**32**

# V850ES/JG3-U, V850ES/JH3-U

## User's Manual: Hardware

RENESAS MCU
V850ES/Jx3-U Microcontrollers

| V850ES/JG3-U | V850ES/JH3-U |
| --- | --- |
| μPD70F3763 | μPD70F3768 |
| μPD70F3764 | μPD70F3769 |

**Renesas Electronics**
www.renesas.com

Rev.4.00    Sep, 2011

# NOTES FOR CMOS DEVICES

① **VOLTAGE APPLICATION WAVEFORM AT INPUT PIN**

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between $V_{IL}$ (MAX) and $V_{IH}$ (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between $V_{IL}$ (MAX) and $V_{IH}$ (MIN).

② **HANDLING OF UNUSED INPUT PINS**

Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to $V_{DD}$ or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

③ **PRECAUTION AGAINST ESD**

A strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

④ **STATUS BEFORE INITIALIZATION**

Power-on does not necessarily define the initial status of a MOS device. Immediately after the power source is turned ON, devices with reset functions have not yet been initialized. Hence, power-on does not guarantee output pin levels, I/O settings or contents of registers. A device is not initialized until the reset signal is received. A reset operation must be executed immediately after power-on for devices with reset functions.

⑤ **POWER ON/OFF SEQUENCE**

In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current.

The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

⑥ **INPUT OF SIGNAL DURING POWER OFF STATE**

Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

# How to Use This Manual

**Readers**  This manual is intended for users who wish to understand the functions of the V850ES/JG3-U and V850ES/JH3-U and design application systems using the V850ES/JG3-U and V850ES/JH3-U.

**Purpose**  This manual is intended to give users an understanding of the hardware functions of the V850ES/JG3-U and V850ES/JH3-U shown in the **Organization** below.

**Organization**  The manual of these products is divided into two volumes: Hardware (this volume) and Architecture (**V850ES Architecture User's Manual**).

| Hardware | Architecture |
|---|---|
| • Pin functions | • Data types |
| • CPU function | • Register set |
| • On-chip peripheral functions | • Instruction format and instruction set |
| • Flash memory programming | • Interrupts and exceptions |
| • Electrical specifications | • Pipeline operation |

**How to Read This Manual**  It is assumed that the readers of this manual have general knowledge in the fields of electrical engineering, logic circuits, and microcontrollers.

To understand the overall functions of the V850ES/JG3-U and V850ES/JH3-U
→Read this manual according to the **CONTENTS**.

To find the details of a register where the name is known
→Use **APPENDIX C  REGISTER INDEX**.

Register format
→The name of the bit whose number is in angle brackets (<>) in the figure of the register format of each register is defined as a reserved word in the device file.

To understand the details of an instruction function
→Refer to the **V850ES Architecture User's Manual** available separately.

To know the electrical specifications of the V850ES/JG3-U and V850ES/JH3-U
→Refer to the **CHAPTER 33 ELECTRICAL SPECIFICATIONS**.

The "yyy bit of the xxx register" is described as the "xxx.yyy bit" in this manual.  Note with caution that if "xxx.yyy" is described as is in a program, however, the compiler/assembler cannot recognize it correctly.

The mark <R> shows major revised points. The revised points can be easily searched by copying an "<R>" in the PDF file and specifying it in the "Find what: " field.

**Conventions**

| | |
|---|---|
| Data significance: | Higher digits on the left and lower digits on the right |
| Active low representation: | $\overline{\text{xxx}}$ (overscore over pin or signal name) |
| Memory map address: | Higher addresses on the top and lower addresses on the bottom |
| **Note**: | Footnote for item marked with **Note** in the text |
| **Caution**: | Information requiring particular attention |
| **Remark**: | Supplementary information |
| Numeric representation: | Binary ... xxxx or xxxxB |
| | Decimal ... xxxx |
| | Hexadecimal ... xxxxH |
| Prefix indicating power of 2 (address space, memory capacity): | K (kilo): $2^{10} = 1,024$ |
| | M (mega): $2^{20} = 1,024^2$ |
| | G (giga): $2^{30} = 1,024^3$ |

**Related Documents**  The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

**Documents related to V850ES/JG3-U and V850ES/JH3-U**

| Document Name | Document No. |
|---|---|
| V850ES Architecture User's Manual | U15943E |
| V850ES/JG3-U, V850ES/JH3-U Hardware User's Manual | This manual |

**Documents related to development tools**

| Document Name | | Document No. |
|---|---|---|
| QB-V850ESJX3H In-Circuit Emulator | | U19170E |
| QB-V850MINI On-Chip Debug Emulator | | U17638E |
| QB-MINI2 On-Chip Debug Emulator with Programming Function | | U18371E |
| CA850 Ver. 3.20 C Compiler Package | Operation | U18512E |
| | C Language | U18513E |
| | Assembly Language | U18514E |
| | Link Directives | U18515E |
| PM+ Ver. 6.30 Project Manager | | U18416E |
| ID850QB Ver. 3.40 Integrated Debugger | Operation | U18604E |
| SM850 Ver. 2.50 System Simulator | Operation | U16218E |
| SM850 Ver. 2.00 or Later System Simulator | External Part User Open Interface Specification | U14873E |
| SM+ System Simulator | Operation | U18601E |
| | User Open Interface | U18212E |
| RX850 Ver. 3.20 Real-Time OS | Basics | U13430E |
| | Installation | U17419E |
| | Technical | U13431E |
| | Task Debugger | U17420E |
| RX850 Pro Ver. 3.21 Real-Time OS | Basics | U18165E |
| | Installation | U17421E |
| | Task Debugger | U17422E |
| AZ850 Ver. 3.30 System Performance Analyzer | | U17423E |
| PG-FP5 Flash Memory Programmer | | U18865E |

Caution: This product uses SuperFlash® technology licensed from Silicon Storage Technology, Inc.

**EEPROM is a trademark of Renesas Electronics Corporation.**

**IECUBE is a registered trademark of Renesas Electronics Corporation in Japan and Germany.**

**MINICUBE is a registered trademark of Renesas Electronics Corporation in Japan and Germany or a trademark in the United States of America.**

**Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.**

**SuperFlash is a registered trademark of Silicon Storage Technology, Inc. in several countries, including the United States and Japan.**

**PC/AT is a trademark of International Business Machines Corporation.**

**SPARCstation is a trademark of SPARC International, Inc.**

**Solaris and SunOS are trademarks of Sun Microsystems, Inc.**

**TRON is an abbreviation of The Real-Time Operating system Nucleus.**

**ITRON is an abbreviation of Industrial TRON.**

# Table of Contents

# CHAPTER 1 INTRODUCTION

The V850ES/JG3-U and V850ES/JH3-U are products in the low-power series of Renesas Electronics' V850 single-chip microcontrollers designed for real-time control applications.

## 1.1 General

The V850ES/JG3-U and V850ES/JH3-U are 32-bit single-chip microcontrollers that use the V850ES CPU core and incorporate peripheral functions such as ROM/RAM, a timer/counter, serial interfaces, an A/D converter, a D/A converter, a DMA controller, a USB function controller, and USB host controller.

In addition to high real-time response characteristics and 1-clock-pitch basic instructions, the V850ES/JG3-U and V850ES/JH3-U feature multiply instructions realized by a hardware multiplier, saturated operation instructions, and bit manipulation instructions.

Table 1-1 lists the products of the V850ES/JG3-U, and Table 1-2 lists the products of the V850ES/JH3-U.

**Table 1-1. V850ES/JG3-U Product List**

| Generic Name | | V850ES/JG3-U | |
|---|---|---|---|
| Part Number | | $\mu$PD70F3763 | $\mu$PD70F3764 |
| Internal memory | Flash memory | 384 KB | 512 KB |
| | RAM[Note 1] | 48 KB | 56 KB |
| Memory space | Logical space | 64 MB | |
| | External memory area | 64 KB | |
| External bus interface | | Address data bus: 16<br>Multiplexed bus | |
| General-purpose register | | 32 bits × 32 registers | |
| Clock | Main clock | (PLL mode: $f_X$ = 3 to 6 MHz, $f_{XX}$ = 24 to 48 MHz (multiplied by 8)<br>Clock through mode: $f_X$ = 3 to 6 MHz (internal: $f_{XX}$ = 3 to 6 MHz) | |
| | Subclock | $f_{XT}$ = 32.768 kHz | |
| | Internal oscillator | $f_R$ = 220 kHz (TYP.) | |
| | Minimum instruction execution time | 20.8 ns (main clock ($f_{XX}$) = 48 MHz) | |
| I/O port (5 V tolerant) | | I/O: 75 (22) | |
| Timer | 16-bit TAA | 6 channels (including 1 channel used only for interval function) | |
| | 16-bit TAB | 2 channels | |
| | 16-bit TMM | 4 channels | |
| | 16-bit TMT | 1 channel | |
| | Motor control | 1 channel (functions with combination of TAA and TAB; includes Hi-Z output control function) | |
| | Watch timer | 1 channel (RTC) | |
| | WDT | 1 channel | |
| Real-time output function | | 6 bits × 1 channel | |
| 10-bit A/D converter | | 12 channels | |
| 8-bit D/A converter | | 2 channels | |
| Serial interface | CSIF/UARTC | 2 channels | 2 channels |
| | CSIF/UARTC/I²C | 1 channel | 1 channel |
| | CSIF | 2 channels | 2 channels |
| | UARTC/I²C | 2 channels | 2 channels |
| | USB host | 1 channel | 1 channel |
| | USB function | 1 channel | 1 channel |
| DMA controller | | 4 channels (transfer target: on-chip peripheral I/O, internal RAM, external memory) | |
| Interrupt source | External[Notes 2, 3] | 15 (15) | 15 (15) |
| | Internal | 72 | 72 |
| Power save function | | HALT/IDLE1/IDLE2/STOP/subclock/sub-IDLE mode | |
| Reset source | | $\overline{\text{RESET}}$ pin input, watchdog timer 2 (WDT2), clock monitor (CLM), low-voltage detector (LVI) | |
| On-chip debugging | | MINICUBE®, MINICUBE2 supported | |
| Operating power supply voltage | | 2.85 to 3.6 V | |
| Operating ambient temperature | | −40 to +85°C | |
| Package | | 100-pin plastic LQFP (fine pitch) (14 × 14 mm) | |

**Notes 1.** Including 8 KB of data-only RAM area.

    **2.** The figures in parentheses indicate the number of external interrupts that can release STOP mode.

    **3.** Including NMI.

**Table 1-2. V850ES/JH3-U Product List**

| Generic Name | | V850ES/JH3-U | |
|---|---|---|---|
| Part Number | | $\mu$PD70F3768 | $\mu$PD70F3769 |
| Internal memory | Flash memory | 384 KB | 512 KB |
| | RAM[Note 1] | 48 KB | 56 KB |
| Memory space | Logical space | 64 MB | |
| | External memory area | 13 MB | |
| External bus interface | | Address bus: 24<br>Address data bus: 16<br>Separate bus/Multiplexed bus | |
| General-purpose register | | 32 bits × 32 registers | |
| Clock | Main clock | (PLL mode: $f_X$ = 3 to 6 MHz, $f_{XX}$ = 24 to 48 MHz (multiplied by 8)<br>Clock through mode: $f_X$ = 3 to 6 MHz (internal: $f_{XX}$ = 3 to 6 MHz) | |
| | Subclock | $f_{XT}$ = 32.768 kHz | |
| | Internal oscillator | $f_R$ = 220 kHz (TYP.) | |
| | Minimum instruction execution time | 20.8 ns (main clock ($f_{XX}$) = 48 MHz) | |
| I/O port (5 V tolerant) | | I/O: 96 (25) | |
| Timer | 16-bit TAA | 6 channels (including 1 channel used only for interval function) | |
| | 16-bit TAB | 2 channels | |
| | 16-bit TMM | 4 channels | |
| | 16-bit TMT | 1 channel | |
| | Motor control | 1 channel (functions with combination of TAA and TAB; includes Hi-Z output control function) | |
| | Watch timer | 1 channel (RTC) | |
| | WDT | 1 channel | |
| Real-time output function | | 6 bits × 1 channel | |
| 10-bit A/D converter | | 12 channels | |
| 8-bit D/A converter | | 2 channels | |
| Serial interface | CSIF/UARTC | 2 channels | 2 channels |
| | CSIF/UARTC/I²C | 1 channel | 1 channel |
| | CSIF | 2 channels | 2 channels |
| | UARTC/I²C | 2 channels | 2 channels |
| | USB host | 1 channel | 1 channel |
| | USB function | 1 channel | 1 channel |
| DMA controller | | 4 channels (transfer target: on-chip peripheral I/O, internal RAM, external memory) | |
| Interrupt source | External[Notes 2, 3] | 20 (20) | |
| | Internal | 72 | 72 |
| Power save function | | HALT/IDLE1/IDLE2/STOP/subclock/sub-IDLE mode | |
| Reset source | | $\overline{RESET}$ pin input, watchdog timer 2 (WDT2), clock monitor (CLM), low-voltage detector (LVI) | |
| On-chip debugging | | MINICUBE, MINICUBE2 supported | |
| Operating power supply voltage | | 2.85 to 3.6 V | |
| Operating ambient temperature | | −40 to +85°C | |
| Package | | 128-pin plastic LQFP (fine pitch) (14 × 20 mm) | |

**Notes 1.** Including 8 KB of data-only RAM area.

**2.** The figures in parentheses indicate the number of external interrupts that can release STOP mode.

**3.** Including NMI.

## 1.2  Features

○ Minimum instruction execution time: 20.8 ns (main clock ($f_{XX}$) = 48 MHz: $V_{DD}$ = 2.85 to 3.6 V)

 30.5 $\mu$s (subclock ($f_{XT}$) = 32.768 kHz)

○ General-purpose registers: 32 bits $\times$ 32 registers

○ CPU features: Signed multiplication (16 $\times$ 16 $\rightarrow$ 32): 1 or 2 clocks

 Signed multiplication (32 $\times$ 32 $\rightarrow$ 64): 1 to 5 clocks

 Saturated operations (overflow and underflow detection functions included)

 32-bit shift instruction: 1 clock

 Bit manipulation instructions

 Load/store instructions with long/short format

○ Memory space: 64 MB of linear address space (for programs and data)

 External expansion: Up to 16 MB (including 1 MB used as internal ROM/RAM space)

 • Internal memory: RAM: 48/56 KB (see **Table 1-1** and **Table 1-2**)

 Flash memory: 384/512 KB (see **Table 1-1** and **Table 1-2**)

 • External bus interface: Separate bus/multiplexed bus output selectable

 (Only a multiplexed bus is available in the V850ES/JG3-U)

 8/16-bit data bus sizing function

 Wait function

 • Programmable wait function

 • External wait function

 Idle state function

 Bus hold function

○ Interrupts and exceptions:

| | | Internal | | | External | | |
|---|---|---|---|---|---|---|---|
| | | Non-maskable | Maskable | Total | Non-maskable | Maskable | Total |
| V850ES/JG3-U | $\mu$PD70F3763 | 1 | 71 | 72 | 1 | 14 | 15 |
| | $\mu$PD70F3764 | 1 | 71 | 72 | 1 | 14 | 15 |
| V850ES/JH3-U | $\mu$PD70F3768 | 1 | 71 | 72 | 1 | 19 | 20 |
| | $\mu$PD70F3769 | 1 | 71 | 72 | 1 | 19 | 20 |

 Software exceptions: 32 sources

 Exception trap: 2 sources

○ I/O lines: I/O ports: 75 (V850ES/JG3-U)

 96 (V850ES/JH3-U)

○ Timer function: 16-bit interval timer M (TMM): 4 channels

 16-bit timer/event counter AA (TAA): 6 channels

 16-bit timer/event counter AB (TAB): 2 channels

 16-bit timer/event counter T (TMT): 1 channel

 Motor control function (timers used: TAB1, TAA4)

 6-phase PWM function with dead-time function of 16-bit accuracy

 High-impedance output control function

 A/D trigger generation by timer-tuned operation function

 Arbitrary cycle setting function

 Arbitrary dead-time setting function

 Real-time counter (RTC): 1 channel

 Watchdog timer: 1 channel

○ Real-time output port: 6 bits $\times$ 1 channel

○ Serial interface: Asynchronous serial interface C (UARTC)

 3-wire variable-length serial interface F (CSIF)

$I^2C$ bus interface ($I^2C$)

USB host interface

USB function interface

| | |
|---|---|
| UARTC/CSIF: | 2 channels |
| UARTC/CSIF/$I^2C$: | 1 channel |
| UARTC/$I^2C$: | 2 channels |
| CSIF: | 2 channels |
| USB host: | 1 channel |
| USB function: | 1 channel |

| | | |
|---|---|---|
| ○ | A/D converter: | 10-bit resolution: 12 channels |
| ○ | D/A converter: | 8-bit resolution: 2 channels |
| ○ | DMA controller: | 4 channels |
| ○ | DCU (debug control unit): | JTAG interface |
| ○ | Clock generator: | Main clock or subclock operation: |
| | | 7-level CPU clock ($f_{XX}$, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, $f_{XT}$) |
| | | Clock-through mode/PLL mode selectable |
| ○ | Internal oscillation clock: | 220 kHz (TYP.) |
| ○ | Power-save functions: | HALT/IDLE1/IDLE2/STOP/subclock/sub-IDLE mode |
| ○ | Package: | 100-pin plastic LQFP (fine pitch) (14 × 14) (V850ES/JG3-U) |
| | | 128-pin plastic LQFP (fine pitch) (14 × 20) (V850ES/JH3-U) |

## 1.3    Application Fields

Equipment requiring a USB interface such as home audio systems, printers, and scanners.

## 1.4    Ordering Information

• V850ES/JG3-U

| Part Number | Package | Internal Flash Memory |
|---|---|---|
| $\mu$PD70F3763GC-UEU-AX | 100-pin plastic LQFP (fine pitch) (14 $\times$ 14) | 384 KB |
| $\mu$PD70F3764GC-UEU-AX | 100-pin plastic LQFP (fine pitch) (14 $\times$ 14) | 512 KB |

• V850ES/JH3-U

| Part Number | Package | Internal Flash Memory |
|---|---|---|
| $\mu$PD70F3768GF-GAT-AX | 128-pin plastic LQFP (fine pitch) (14 $\times$ 20) | 384 KB |
| $\mu$PD70F3769GF-GAT-AX | 128-pin plastic LQFP (fine pitch) (14 $\times$ 20) | 512 KB |

**Remark**    The V850ES/JG3-U and V850ES/JH3-U are lead-free products.

## 1.5  Pin Configuration (Top View)

• V850ES/JG3-U

100-pin plastic LQFP (fine pitch) (14 × 14)

μPD70F3763GC-UEU-AX            μPD70F3764GC-UEU-AX



**Notes 1.**  Connect this pin to Vss in the normal mode.

**2.**  Connect the REGC pin to Vss via a 4.7 μF (recommended value) capacitor.
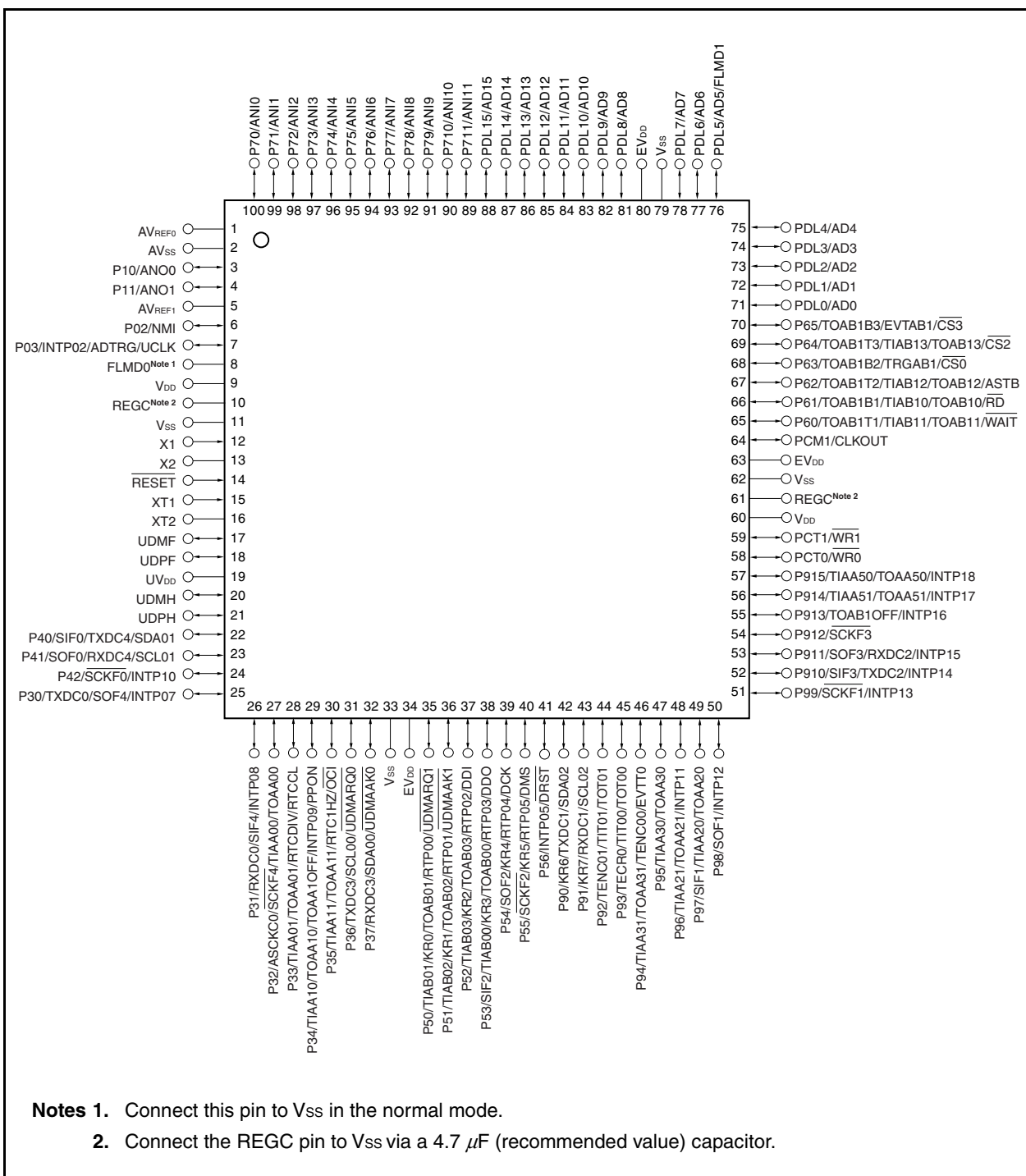
• V850ES/JH3-U

128-pin plastic LQFP (fine pitch) (14 × 20)

μPD70F3768GF-GAT-AX          μPD70F3769GF-GAT-AX



**Notes 1.** Connect this pin to Vss in the normal mode.

**2.** Connect the REGC pin to Vss via a 4.7 μF (recommended value) capacitor.
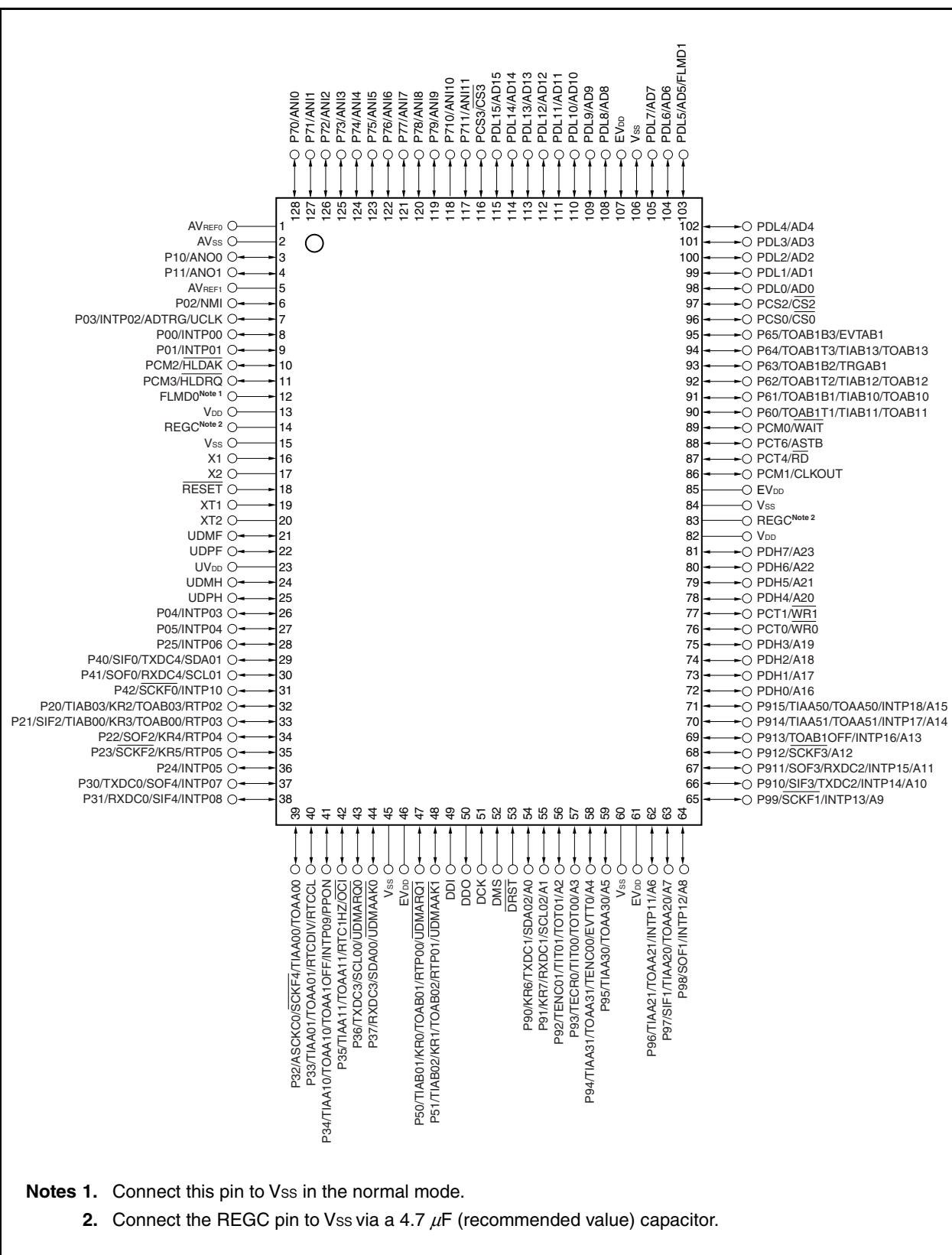
**Pin names**

| | | | | |
|---|---|---|---|---|
| A0 to A23: | Address bus | | RTP00 to RTP05: | Real-time output port |
| AD0 to AD15: | Address/data bus | | RXDC0 to RXDC4: | Receive data |
| ADTRG: | A/D trigger input | | $\overline{\text{SCKF0}}$ to $\overline{\text{SCKF4}}$: | Serial clock |
| ANI0 to ANI11: | Analog input | | SCL00 to SCL02: | Serial clock |
| ANO0, ANO1: | Analog output | | SDA00 to SDA02: | Serial data |
| ASCKC0: | Asynchronous serial clock | | SIF0 to SIB4: | Serial input |
| ASTB: | Address strobe | | SOF0 to SOF4: | Serial output |
| $AV_{REF0}$, $AV_{REF1}$: | Analog reference voltage | | TECR0: | Timer encoder clear input |
| $AV_{SS}$: | Grand for analog pin | | TENC00, TENC01: | Timer encoder input |
| CLKOUT: | Clock output | | TIAA00, TIAA01, | |
| $\overline{\text{CS0}}$, $\overline{\text{CS2}}$, $\overline{\text{CS3}}$: | Chip select | | TIAA10, TIAA11, | |
| DCK: | Debug clock | | TIAA20, TIAA21, | |
| DDI: | Debug data input | | TIAA30, TIAA31, | |
| DDO: | Debug data output | | TIAA50, TIAA51, | |
| DMS: | Debug mode select | | TIAB00 to TIAB03, | |
| $\overline{\text{DRST}}$: | Debug reset | | TIAB10, TIAB13, | |
| $EV_{DD}$: | Power supply for external pin | | TIT00, TIT01: | Timer input |
| EVTT0, EVTAB1: | Timer event count input | | TOAA00, TOAA01, | |
| FLMD0, FLMD1: | Flash programming mode | | TOAA10, TOAA11, | |
| $\overline{\text{HLDAK}}$: | Hold acknowledge | | TOAA20, TOAA21, | |
| $\overline{\text{HLDRQ}}$: | Hold request | | TOAA30, TOAA31, | |
| INTP00 to INTP18: | External interrupt input | | TOAA50 to TOAA51, | |
| KR0 to KR7: | Key return | | TOAB00 to TOAB03, | |
| NMI: | Non-maskable interrupt request | | TOAB10 to TOAB13, | |
| $\overline{\text{OCI}}$: | Over current detection input | | TOAB1B1 to TOAB1B3, | |
| P00 to P05: | Port 0 | | TOAB1T1 to TOAB1T3, | |
| P10, P11: | Port 1 | | TOT00, TOT01: | Timer output |
| P20 to P25: | Port 2 | | TOAA1OFF, | |
| P30 to P37: | Port 3 | | TOAB1OFF: | Timer output off |
| P40 to P42: | Port 4 | | TRGAB1: | Timer trigger input |
| P50 to P56: | Port 5 | | TXDC0 to TXDC4: | Transmit data |
| P60 to P65 | Port 6 | | UCLK: | USB clock |
| P70 to P711: | Port 7 | | $\overline{\text{UDMAAK0}}$, | |
| P90 to P915: | Port 9 | | $\overline{\text{UDMAAK1}}$: | DMA acknowledge for external USB |
| PCM0 to PCM3: | Port CM | | $\overline{\text{UDMARQ0}}$, | |
| PCS0, PCS2, PCS3: | Port CS | | $\overline{\text{UDMARQ1}}$: | DMA request for external USB |
| PCT0, PCT1, | Port CT | | UDMF: | USB data I/O (−) function |
| PCT4, PCT6: | | | UDMH: | USB data I/O (−) Host |
| PDH0 to PDH7: | Port DH | | UDPF: | USB data I/O (+) function |
| PDL0 to PDL15: | Port DL | | UDPH: | USB data I/O (+) Host |
| PPON: | Power supply output for USB | | $UV_{DD}$: | Power supply for USB |
| $\overline{\text{RD}}$: | Read strobe | | $V_{DD}$: | Power supply |
| REGC: | Regulator control | | $V_{SS}$: | Ground |
| $\overline{\text{RESET}}$: | Reset | | $\overline{\text{WAIT}}$: | External wait input |
| RTC1HZ, RTCCL, | | | $\overline{\text{WR0}}$: | Lower byte write strobe |
| RTCDIV: | Real-time counter clock output | | $\overline{\text{WR1}}$: | Upper byte write strobe |
| | | | X1, X2: | Crystal for main clock |
| | | | XT1, XT2: | Crystal for subclock |

RENESAS

## 1.6   Function Block Configuration

### 1.6.1   Internal block diagram

- V850ES/JG3-U



**Notes 1.**  μPD70F3763:    384 KB

　　　　μPD70F3764:    512 KB

**2.**  μPD70F3763:    48 KB (Including a data only RAM: 8 KB)

　　　　μPD70F3764:    56 KB (Including a data only RAM: 8 KB)

• V850ES/JH3-U



**Notes 1.** $\mu$PD70F3768:  384 KB

$\mu$PD70F3769:  512 KB

**2.** $\mu$PD70F3768:  48 KB (Including a data only RAM: 8 KB)

$\mu$PD70F3769:  56 KB (Including a data only RAM: 8 KB)
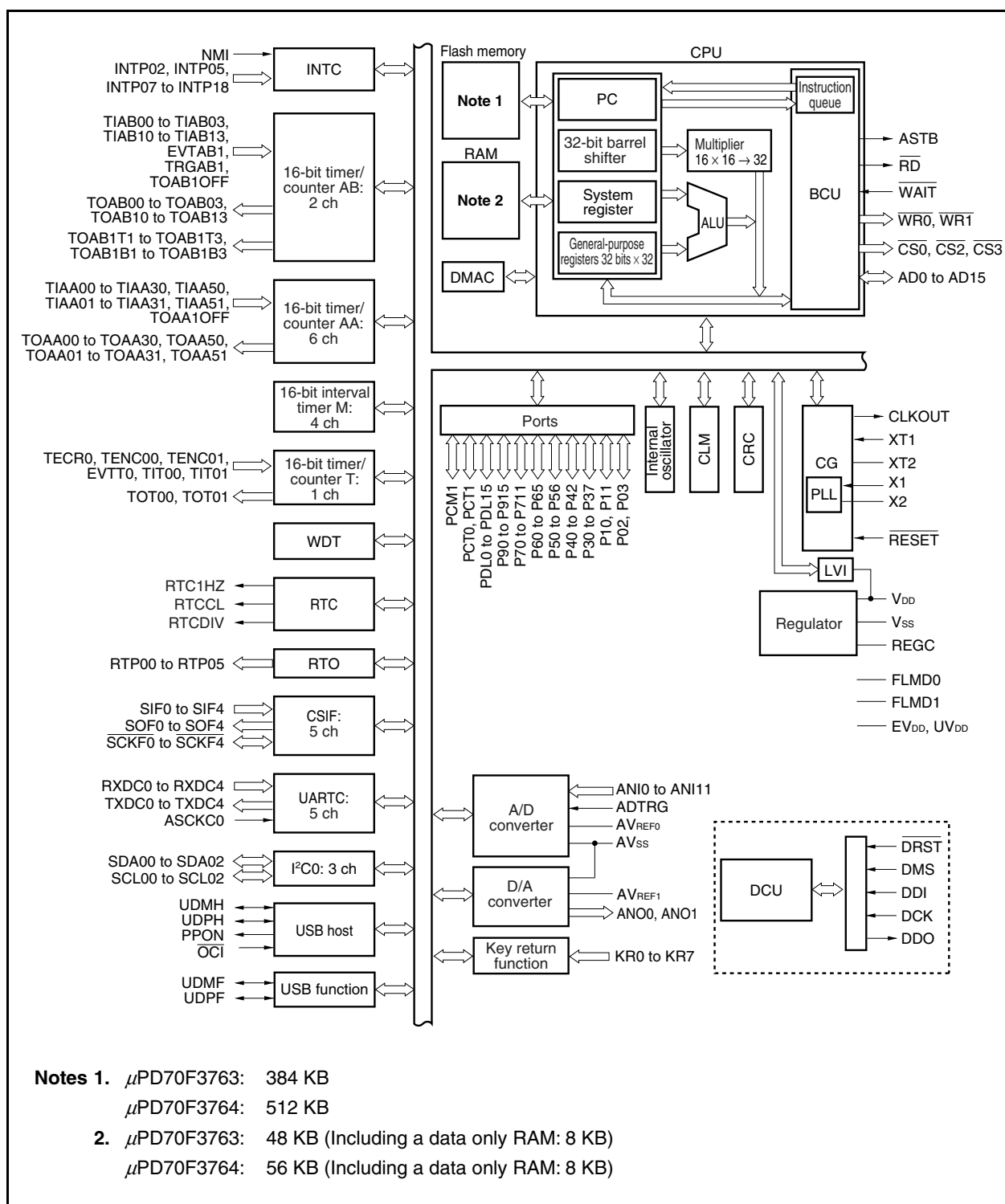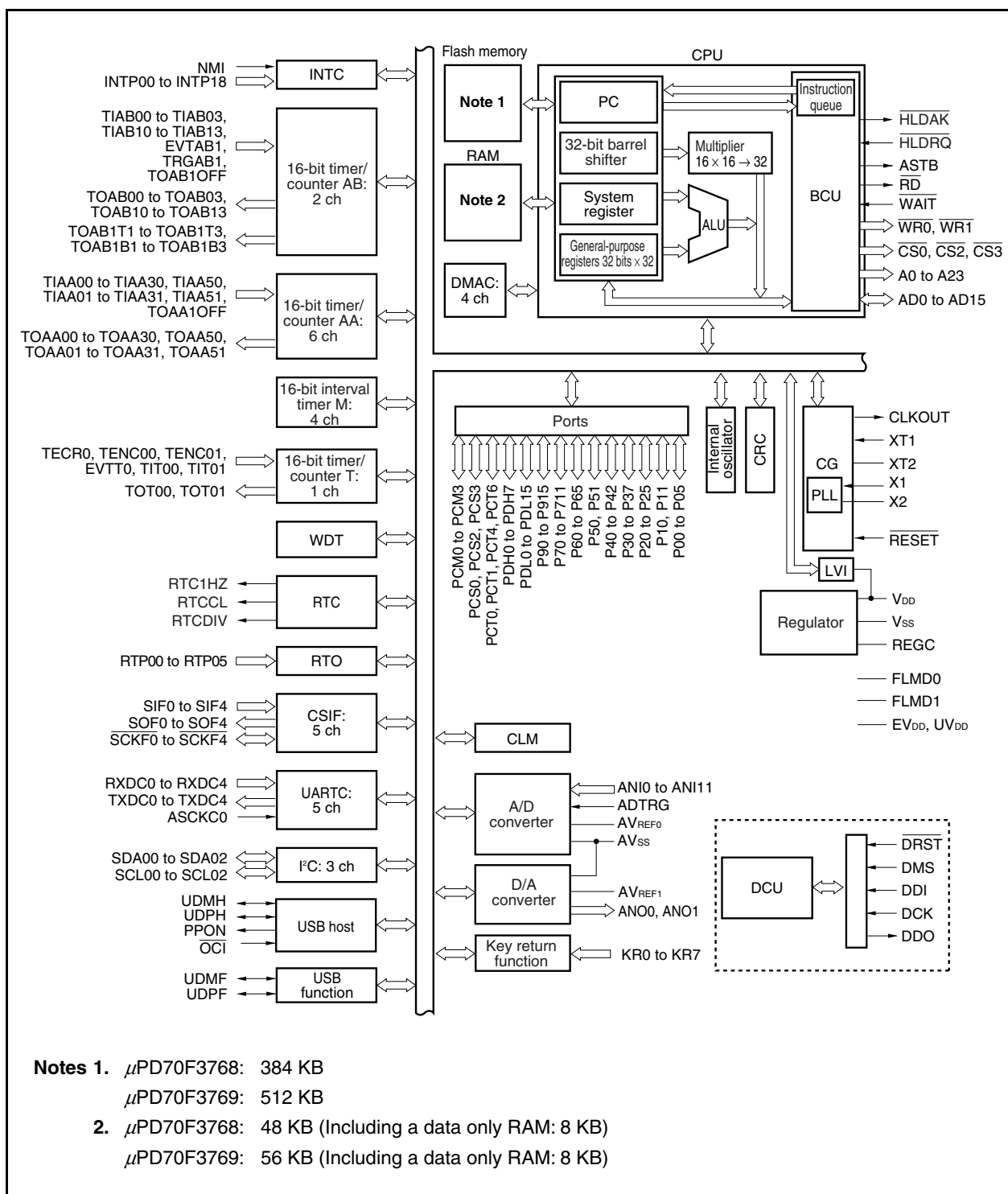
### 1.6.2  Internal units

**(1) CPU**

The CPU uses five-stage pipeline control to enable single-clock execution of address calculations, arithmetic logic operations, data transfers, and almost all other instruction processing.

Other dedicated on-chip hardware, such as a multiplier (16 bits $\times$ 16 bits $\rightarrow$ 32 bits) and a barrel shifter (32 bits) contribute to faster complex processing.

**(2) Bus control unit (BCU)**

The BCU starts a required external bus cycle based on the physical address obtained by the CPU.  When an instruction is fetched from external memory space and the CPU does not send a bus cycle start request, the BCU generates a prefetch address and prefetches the instruction code.  The prefetched instruction code is stored in an instruction queue.

**(3) Flash memory (ROM)**

This is a 512/384 KB flash memory mapped to addresses 0000000H to 007FFFFH/0000000H to 005FFFFH.  It can be accessed from the CPU in one clock during instruction fetch.

**(4) RAM**

This is a 48/40 KB RAM mapped to addresses 3FF3000H to 3FFEFFFH/3FF5000H to 3FFEFFFH.  It can be accessed from the CPU in one clock during data access.  An 8 KB data-only RAM is incorporated at addresses 00280000H to 00281FFFH.

**(5) Interrupt controller (INTC)**

This controller handles hardware interrupt requests (NMI, INTP0 to INTP18) from on-chip peripheral hardware and external hardware.  Eight levels of interrupt priorities can be specified for these interrupt requests, and multiplexed servicing control can be performed.

**(6) Clock generator (CG)**

A main clock oscillator and subclock oscillator are provided and generate the main clock oscillation frequency ($f_X$) and subclock frequency ($f_{XT}$), respectively.  There are two modes: In the clock-through mode, $f_X$ is used as the main clock frequency ($f_{XX}$) as is.  In the PLL mode, $f_X$ is used multiplied by 8.

The CPU clock frequency ($f_{CPU}$) can be selected from among $f_{XX}$, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, and $f_{XT}$.

**(7) Internal oscillator**

An internal oscillator is provided on chip.  The oscillation frequency is 220 kHz (TYP).  The internal oscillator supplies the clock for watchdog timer 2 and timer M.

**(8) Timer/counter**

Six-channel 16-bit timer/event counter AA (TAA), two-channel 16-bit timer/event counter AB (TAB), one-channel 16-bit timer/event counter T (TMT), and four-channel 16-bit interval timer M (TMM) are provided on chip.  The motor control function can be realized using TAB1 and TAA4 in combination.

**(9)  Real-time counter (for watch)**

The real-time counter counts the reference time (one second) for watch counting based on the subclock (32.768 kHz) or main clock.  This can simultaneously be used as the interval timer based on the main clock.  Hardware counters dedicated to year, month, day of week, day, hour, minute, and second are provided, and can count up to 99 years.

**(10)  Watchdog timer 2**

A watchdog timer is provided on chip to detect inadvertent program loops, system abnormalities, etc.

The internal oscillation clock, the main clock, or the subclock can be selected as the source clock.

Watchdog timer 2 generates a non-maskable interrupt request signal (INTWDT2) or a system reset signal (WDT2RES) after an overflow occurs.

**(11)  Serial interface**

The V850ES/JG3-U and V850ES/JH3-U include three kinds of serial interfaces (asynchronous serial interface C (UARTC), 3-wire variable-length serial interface F (CSIF), and an $I^2C$ bus interface ($I^2C$)), a USB function controller (USBF), and USB host controller (USBH).

UARTC transfers data via the TXDC0 to TXDC2 pins and RXDC0 to RXDC2 pins.

CSIF transfers data via the SOF0 to SOF4 pins, SIF0 to SIF4 pins, and $\overline{SCKF0}$ to $\overline{SCKF4}$ pins.

In the case of $I^2C$, data is transferred via the SDA00 to SDA02 and SCL00 to SCL02 pins.

USBF transfers data via the UDMF and UDPF pins.

USBH transfers data via the UDMH and UDPH pins.

**(12)  A/D converter**

This 10-bit A/D converter includes 12 analog input pins.  Conversion is performed using the successive approximation method.

**(13)  D/A converter**

A two-channel, 8-bit-resolution D/A converter that uses the R-2R ladder method is provided on chip.

**(14)  DMA controller**

A 4-channel DMA controller is provided on chip.  This controller transfers data between the internal RAM, on-chip peripheral I/O devices, and external memory in response to interrupt requests sent by on-chip peripheral I/O devices.

**(15)  Key interrupt function**

A key interrupt request signal (INTKR) can be generated by inputting a falling edge to the key input pins (8 channels).

**(16)  Real-time output function**

The real-time output function transfers preset 6-bit data to output latches upon the occurrence of a timer compare register match signal.

**(17)  CRC function**

A CRC operation circuit that generates a 16-bit CRC (Cyclic Redundancy Check) code upon setting of 8-bit data is provided on-chip.

**(18)  DCU (debug control unit)**

An on-chip debug function that uses the JTAG (Joint Test Action Group) communication specifications is provided. Switching between the normal port function and on-chip debugging function is done with the control pin input level and the OCDM register.

**(19)  Ports**

The following general-purpose port functions and control pin functions are available.

• V850ES/JG3-U

| Port | I/O | Alternate Function |
|------|-----|--------------------|
| P0 | 2-bit I/O | NMI, external interrupt, A/D converter trigger, serial interface |
| P1 | 2-bit I/O | D/A converter analog output |
| P3 | 10-bit I/O | External interrupt, real-time counter, serial interface, timer I/O |
| P4 | 3-bit I/O | Serial interface, external interrupt |
| P5 | 7-bit I/O | Timer I/O, serial interface, real-time output, key interrupt input, debug I/O |
| P6 | 6-bit I/O | External bus control signal, timer I/O, external bus control signal |
| P7 | 12-bit I/O | A/D converter analog input |
| P9 | 16-bit I/O | Serial interface, key interrupt input, timer I/O, external interrupt |
| PCM | 1-bit I/O | External bus control signal |
| PCT | 2-bit I/O | External bus control signal |
| PDL | 16-bit I/O | External bus address/data bus |

• V850ES/JH3-U

| Port | I/O | Alternate Function |
|------|-----|--------------------|
| P0 | 6-bit I/O | NMI, external interrupt, A/D converter trigger, serial interface |
| P1 | 2-bit I/O | D/A converter analog output |
| P2 | 6-bit I/O | Timer I/O, real-time output, key interrupt input, serial interface |
| P3 | 10-bit I/O | External interrupt, real-time counter, serial interface, timer I/O |
| P4 | 3-bit I/O | Serial interface, external bus interrupt |
| P5 | 2-bit I/O | Timer I/O, real-time output, key interrupt input |
| P6 | 6-bit I/O | External bus control signal, timer I/O |
| P7 | 12-bit I/O | A/D converter analog input |
| P9 | 16-bit I/O | External address bus, serial interface, key interrupt input, timer I/O, external interrupt |
| PCM | 4-bit I/O | External bus control signal |
| PCS | 3-bit I/O | External bus control signal |
| PCT | 4-bit I/O | External bus control signal |
| PDH | 8-bit I/O | External address bus |
| PDL | 16-bit I/O | External address/data bus |

# CHAPTER 2  PIN  FUNCTIONS

## 2.1   List of Pin Functions

The names and functions of the pins of the V850ES/JG3-U and V850ES/JH3-U are described below.

There are four types of pin I/O buffer power supplies: $AV_{REF0}$, $AV_{REF1}$, $EV_{DD}$, and $UV_{DD}$.  The relationship between these power supplies and the pins is described below.

**Table 2-1.  Pin I/O Buffer Power Supplies**

| Power Supply | Corresponding Pins | |
|---|---|---|
| | V850ES/JG3-U | V850ES/JH3-U |
| $AV_{REF0}$ | Port 7 | Port 7 |
| $AV_{REF1}$ | Port 1 | Port 1 |
| $EV_{DD}$ | $\overline{RESET}$, ports 0, 3 to 6, 9, CM, CT, DL | $\overline{RESET}$, ports 0, 2 to 6, 9, CM, CS, CT, DH, DL |
| $UV_{DD}$ | UDPF, UDMF, UDPH, UDMH | UDPF, UDMF, UDPH, UDMH |

<R>    **(1) Port pins**

(1/4)

| Pin Name | I/O | Function | | Alternate Function | Pin No. | |
|---|---|---|---|---|---|---|
| | | | | | JG3-U | JH3-U |
| P00 | I/O | Port 0 | | INTP00 | – | 8 |
| P01 | | 6-bit I/O port (V850ES/JH3-U) | | INTP01 | – | 9 |
| P02 | | 2-bit I/O port (V850ES/JG3-U) | | NMI | 6 | 6 |
| P03 | | Input/output can be specified in 1-bit units. | | INTP02/ADTRG/UCLK | 7 | 7 |
| P04 | | 5 V tolerant. | | INTP03 | – | 26 |
| P05 | | | | INTP04 | – | 27 |
| P10 | I/O | Port 1 | | ANO0 | 3 | 3 |
| P11 | | 2-bit I/O port<br>Input/output can be specified in 1-bit units. | | ANO1 | 4 | 4 |
| P20 | I/O | Port 2 | | TIAB03/KR2/TOAB03/RTP02 | – | 32 |
| P21 | | 6-bit I/O port | | SIF2/TIAB00/KR3/TOAB00/RTP03 | – | 33 |
| P22 | | Input/output can be specified in 1-bit units. | | SOF2/KR4/RTP04 | – | 34 |
| P23 | | 5 V tolerant. | | $\overline{\text{SCKF2}}$/KR5/RTP05 | – | 35 |
| P24 | | | | INTP05 | – | 36 |
| P25 | | | | INTP06 | – | 28 |
| P30 | I/O | Port 3 | | TXDC0/SOF4/INTP07 | 25 | 37 |
| P31 | | 8-bit I/O port | | RXDC0/SIF4/INTP08 | 26 | 38 |
| P32 | | Input/output can be specified in 1-bit units. | | ASCKC0/$\overline{\text{SCKF4}}$/TIAA00/TOAA00 | 27 | 39 |
| P33 | | 5 V tolerant. | | TIAA01/TOAA01/RTCDIV/RTCCL | 28 | 40 |
| P34 | | | | TIAA10/TOAA10/TOAA1OFF/INTP09<br>/PPON | 29 | 41 |
| P35 | | | | TIAA11/TOAA11/RTC1HZ/$\overline{\text{OCI}}$ | 30 | 42 |
| P36 | | | | TXDC3/SCL00/$\overline{\text{UDMARQ0}}$ | 31 | 43 |
| P37 | | | | RXDC3/SDA00/$\overline{\text{UDMAAK0}}$ | 32 | 44 |
| P40 | I/O | Port 4 | | SIF0/TXDC4/SDA01 | 22 | 29 |
| P41 | | 3-bit I/O port | | SOF0/RXDC4/SCL01 | 23 | 30 |
| P42 | | Input/output can be specified in 1-bit units.<br>5 V tolerant. | | $\overline{\text{SCKF0}}$/INTP10 | 24 | 31 |
| P50 | I/O | Port 5 | | TIAB01/KR0/TOAB01/RTP00<br>/$\overline{\text{UDMARQ1}}$ | 35 | 47 |
| P51 | | 2-bit I/O port (V850ES/JH3-U) | | TIAB02/KR1/TOAB02/RTP01<br>/$\overline{\text{UDMAAK1}}$ | 36 | 48 |
| P52 | | 7-bit I/O port (V850ES/JG3-U) | | TIAB03/KR2/TOAB03/RTP02/DDI | 37 | – |
| P53 | | Input/output can be specified in 1-bit units. | | SIF2/TIAB00/KR3/TOAB00<br>/RTP03/DDO | 38 | – |
| P54 | | 5 V tolerant. | | SOF2/KR4/RTP04/DCK | 39 | – |
| P55 | | | | $\overline{\text{SCKF2}}$/KR5/RTP05/DMS | 40 | – |
| P56 | | | | INTP05/$\overline{\text{DRST}}$ | 41 | – |

**Remark**    JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(2/4)

| Pin Name | I/O | Function | Alternate Function | Pin No. JG3-U | Pin No. JH3-U |
|---|---|---|---|---|---|
| P60 | I/O | Port 6<br>6-bit I/O port<br>Input/output can be specified in 1-bit units. | TOAB1T1/TOAB11/TIAB11/$\overline{\text{WAIT}}$ | 65 | – |
|  |  |  | TOAB1T1/TOAB11/TIAB11 | – | 90 |
| P61 |  |  | TOAB1B1/TIAB10/TOAB10/$\overline{\text{RD}}$ | 66 | – |
|  |  |  | TOAB1B1/TIAB10/TOAB10 | – | 91 |
| P62 |  |  | TOAB1T2/TOAB12/TIAB12/ASTB | 67 | – |
|  |  |  | TOAB1T2/TOAB12/TIAB12 | – | 92 |
| P63 |  |  | TOAB1B2/TRGAB1/$\overline{\text{CS0}}$ | 68 | – |
|  |  |  | TOAB1B2/TRGAB1 | – | 93 |
| P64 |  |  | TOAB1T3/TOAB13/TIAB13/$\overline{\text{CS2}}$ | 69 | – |
|  |  |  | TOAB1T3/TOAB13/TIAB13 | – | 94 |
| P65 |  |  | TOAB1B3/EVTAB1/$\overline{\text{CS3}}$ | 70 | – |
|  |  |  | TOAB1B3/EVTAB1 | – | 95 |
| P70 | I/O | Port 7<br>12-bit I/O port<br>Input/output can be specified in 1-bit units. | ANI0 | 100 | 128 |
| P71 |  |  | ANI1 | 99 | 127 |
| P72 |  |  | ANI2 | 98 | 126 |
| P73 |  |  | ANI3 | 97 | 125 |
| P74 |  |  | ANI4 | 96 | 124 |
| P75 |  |  | ANI5 | 95 | 123 |
| P76 |  |  | ANI6 | 94 | 122 |
| P77 |  |  | ANI7 | 93 | 121 |
| P78 |  |  | ANI8 | 92 | 120 |
| P79 |  |  | ANI9 | 91 | 119 |
| P710 |  |  | ANI10 | 90 | 118 |
| P711 |  |  | ANI11 | 89 | 117 |
| P90 | I/O | Port 9<br>16-bit I/O port<br>Input/output can be specified in 1-bit units. | KR6/TXDC1/SDA02 | 42 | – |
|  |  |  | KR6/TXDC1/SDA02/A0 | – | 54 |
| P91 |  |  | KR7/RXDC1/SCL02 | 43 | – |
|  |  |  | KR7/RXDC1/SCL02/A1 | – | 55 |
| P92 |  |  | TENC01/TIT01/TOT01 | 44 | – |
|  |  |  | TENC01/TIT01/TOT01/A2 | – | 56 |
| P93 |  |  | TECR0/TIT00/TOT00 | 45 | – |
|  |  |  | TECR0/TIT00/TOT00/A3 | – | 57 |
| P94 |  |  | TIAA31/TOAA31/TENC00/EVTT0 | 46 | – |
|  |  |  | TIAA31/TOAA31/TENC00/EVTT0/A4 | – | 58 |
| P95 |  |  | TIAA30/TOAA30 | 47 | – |
|  |  |  | TIAA30/TOAA30/A5 | – | 59 |
| P96 |  |  | TIAA21/TOAA21/INTP11 | 48 | – |
|  |  |  | TIAA21/TOAA21/INTP11/A6 | – | 62 |
| P97 |  |  | SIF1/TIAA20/TOAA20 | 49 | – |
|  |  |  | SIF1/TIAA20/TOAA20/A7 | – | 63 |

**Remark**   JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(3/4)

| Pin Name | I/O | Function | Alternate Function | Pin No. | |
|---|---|---|---|---|---|
| | | | | JG3-U | JH3-U |
| P98 | I/O | Port 9<br>16-bit I/O port<br>Input/output can be specified in 1-bit units. | SOF1/INTP12 | 50 | – |
| | | | SOF1/INTP12/A8 | – | 64 |
| P99 | | | $\overline{\text{SCKF1}}$/INTP13 | 51 | – |
| | | | $\overline{\text{SCKF1}}$/INTP13/A9 | – | 65 |
| P910 | | | SIF3/TXDC2/INTP14 | 52 | – |
| | | | SIF3/TXDC2/INTP14/A10 | – | 66 |
| P911 | | | SOF3/RXDC2/INTP15 | 53 | – |
| | | | SOF3/RXDC2/INTP15/A11 | – | 67 |
| P912 | | | $\overline{\text{SCKF3}}$ | 54 | – |
| | | | $\overline{\text{SCKF3}}$/A12 | – | 68 |
| P913 | | | TOAB1OFF/INTP16 | 55 | – |
| | | | TOAB1OFF/INTP16/A13 | – | 69 |
| P914 | | | TIAA51/TOAA51/INTP17 | 56 | – |
| | | | TIAA51/TOAA51/INTP17/A14 | – | 70 |
| P915 | | | TIAA50/TOAA50/INTP18 | 57 | – |
| | | | TIAA50/TOAA50/INTP18/A15 | – | 71 |
| PCM0 | I/O | Port CM<br>4-bit I/O port (V850ES/JH3-U)<br>1-bit I/O port (V850ES/JG3-U)<br>Input/output can be specified in 1-bit units. | $\overline{\text{WAIT}}$ | – | 89 |
| PCM1 | | | CLKOUT | 64 | 86 |
| PCM2 | | | $\overline{\text{HLDAK}}$ | – | 10 |
| PCM3 | | | $\overline{\text{HLDRQ}}$ | – | 11 |
| PCS0 | I/O | Port CS<br>3-bit I/O port<br>Input/output can be specified in 1-bit units. | $\overline{\text{CS0}}$ | – | 96 |
| PCS2 | | | $\overline{\text{CS2}}$ | – | 97 |
| PCS3 | | | $\overline{\text{CS3}}$ | – | 116 |
| PCT0 | I/O | Port CT<br>4-bit I/O port (V850ES/JH3-U)<br>2-bit I/O port (V850ES/JG3-U)<br>Input/output can be specified in 1-bit units. | $\overline{\text{WR0}}$ | 58 | 76 |
| PCT1 | | | $\overline{\text{WR1}}$ | 59 | 77 |
| PCT4 | | | $\overline{\text{RD}}$ | – | 87 |
| PCT6 | | | ASTB | – | 88 |
| PDH0 | I/O | Port DH<br>8-bit I/O port<br>Input/output can be specified in 1-bit units. | A16 | – | 72 |
| PDH1 | | | A17 | – | 73 |
| PDH2 | | | A18 | – | 74 |
| PDH3 | | | A19 | – | 75 |
| PDH4 | | | A20 | – | 78 |
| PDH5 | | | A21 | – | 79 |
| PDH6 | | | A22 | – | 80 |
| PDH7 | | | A23 | – | 81 |

**Remark** JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(4/4)

| Pin Name | I/O | Function | Alternate Function | Pin No. | |
|---|---|---|---|---|---|
| | | | | JG3-U | JH3-U |
| PDL0 | I/O | Port DL<br>16-bit I/O port<br>Input/output can be specified in 1-bit units. | AD0 | 71 | 98 |
| PDL1 | | | AD1 | 72 | 99 |
| PDL2 | | | AD2 | 73 | 100 |
| PDL3 | | | AD3 | 74 | 101 |
| PDL4 | | | AD4 | 75 | 102 |
| PDL5 | | | AD5/FLMD1 | 76 | 103 |
| PDL6 | | | AD6 | 77 | 104 |
| PDL7 | | | AD7 | 78 | 105 |
| PDL8 | | | AD8 | 81 | 108 |
| PDL9 | | | AD9 | 82 | 109 |
| PDL10 | | | AD10 | 83 | 110 |
| PDL11 | | | AD11 | 84 | 111 |
| PDL12 | | | AD12 | 85 | 112 |
| PDL13 | | | AD13 | 86 | 113 |
| PDL14 | | | AD14 | 87 | 114 |
| PDL15 | | | AD15 | 88 | 115 |

**Remark**   JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

<R>   **(2) Non-port Pins**

(1/9)

| Pin Name | I/O | Function | Alternate Function | Pin No. JG3-U | Pin No. JH3-U |
|---|---|---|---|---|---|
| A0 | Output | Address bus for external memory (when using separate bus) | P90/KR6/TXDC1/SDA02 | – | 54 |
| A1 | | | P91/KR7/RXDC1/SCL02 | – | 55 |
| A2 | | | P92/TENC01/TIT01/TOT01 | – | 56 |
| A3 | | | P93/TECR00/TIT00/TOT00 | – | 57 |
| A4 | | | P94/TIAA31/TOAA31/TENC0/EVTT0 | – | 58 |
| A5 | | | P95/TIAA30/TOAA30 | – | 59 |
| A6 | | | P96/TIAA21/TOAA21/INTP11 | – | 62 |
| A7 | | | P97/SIF1/TIAA20/TOAA20 | – | 63 |
| A8 | | | P98/SOF1/INTP12 | – | 64 |
| A9 | | | P99/$\overline{\text{SCKF1}}$/INTP13 | – | 65 |
| A10 | | | P910/SIF3/TXDC2/INTP14 | – | 66 |
| A11 | | | P911/SOF3/RXDC2/INTP15 | – | 67 |
| A12 | | | P912/$\overline{\text{SCKF3}}$ | – | 68 |
| A13 | | | P913/TOAB1OFF/INTP16 | – | 69 |
| A14 | | | P914/TIAA51/TOAA51/INTP17 | – | 70 |
| A15 | | | P915/TIAA50/TOAA50/INTP18 | – | 71 |
| A16 | | | PDH0 | – | 72 |
| A17 | | | PDH1 | – | 73 |
| A18 | | | PDH2 | – | 74 |
| A19 | | | PDH3 | – | 75 |
| A20 | | | PDH4 | – | 78 |
| A21 | | | PDH5 | – | 79 |
| A22 | | | PDH6 | – | 80 |
| A23 | | | PDH7 | – | 81 |
| AD0 | I/O | Address/data bus for external memory | PDL0 | 71 | 98 |
| AD1 | | | PDL1 | 72 | 99 |
| AD2 | | | PDL2 | 73 | 100 |
| AD3 | | | PDL3 | 74 | 101 |
| AD4 | | | PDL4 | 75 | 102 |
| AD5 | | | PDL5/FLMD1 | 76 | 103 |
| AD6 | | | PDL6 | 77 | 104 |
| AD7 | | | PDL7 | 78 | 105 |
| AD8 | | | PDL8 | 81 | 108 |
| AD9 | | | PDL9 | 82 | 109 |
| AD10 | | | PDL10 | 83 | 110 |
| AD11 | | | PDL11 | 84 | 111 |
| AD12 | | | PDL12 | 85 | 112 |
| AD13 | | | PDL13 | 86 | 113 |
| AD14 | | | PDL14 | 87 | 114 |
| AD15 | | | PDL15 | 88 | 115 |
| ADTRG | Input | External trigger input for A/D converter 5 V tolerant. | P03/INTP02/UCLK | 7 | 7 |

**Remark**   JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(2/9)

| Pin Name | I/O | Function | Alternate Function | Pin No. | |
|---|---|---|---|---|---|
| | | | | JG3-U | JH3-U |
| ANI0 | Input | Analog voltage input for A/D converter | P70 | 100 | 128 |
| ANI1 | | | P71 | 99 | 127 |
| ANI2 | | | P72 | 98 | 126 |
| ANI3 | | | P73 | 97 | 125 |
| ANI4 | | | P74 | 96 | 124 |
| ANI5 | | | P75 | 95 | 123 |
| ANI6 | | | P76 | 94 | 122 |
| ANI7 | | | P77 | 93 | 121 |
| ANI8 | | | P78 | 92 | 120 |
| ANI9 | | | P79 | 91 | 119 |
| ANI10 | | | P710 | 90 | 118 |
| ANI11 | | | P711 | 89 | 117 |
| ANO0 | Output | Analog voltage output for D/A converter | P10 | 3 | 3 |
| ANO1 | | | P11 | 4 | 4 |
| ASCKC0 | Input | UARTC0 baud rate clock input. 5 V tolerant. | P32/$\overline{\text{SCKF4}}$/TIAA00/TOAA00 | 27 | 39 |
| ASTB | Output | Address strobe signal for external memory | P62/TOAB1T2/TIAB12/TOAB12 | 67 | – |
| | | | PCT6 | – | 88 |
| AV$_{\text{REF0}}$ | – | Reference voltage input for A/D converter, positive power supply for port 7 | – | 1 | 1 |
| AV$_{\text{REF1}}$ | – | Reference voltage input for D/A converter, positive power supply for port 1 | – | 5 | 5 |
| AV$_{\text{SS}}$ | – | Ground potential for A/D and D/A converters | – | 2 | 2 |
| CLKOUT | Output | Internal system clock output | PCM1 | 64 | 86 |
| $\overline{\text{CS0}}$ | Output | Chip select output | P63/TOAB1B2/TRGAB1 | 68 | – |
| | | | PCS0 | – | 96 |
| $\overline{\text{CS2}}$ | Output | Chip select output | P64/TOAB1T3/TIAB13/TOAB13 | 69 | – |
| | | | PCS2 | – | 97 |
| $\overline{\text{CS3}}$ | Output | Chip select output | P65/TOAB1B3/EVTAB1 | 70 | – |
| | | | PCS3 | – | 116 |
| DCK | Input | Clock input for on-chip debugging, 5 V tolerant. | P54/SOF2/KR4/RTP04 | 39 | – |
| | | | – | – | 51 |
| DDI | Input | Data input for on-chip debugging, 5 V tolerant. | P52/TIAB03/KR2/TOAB03/RTP02 | 37 | – |
| | | | – | – | 49 |
| DDO | Output | Data output for on-chip debugging In the on-chip debug mode, high-level output is forcibly set. 5 V tolerant. | P53/SIF2/TIAB00/KR3/TOAB00/RTP03 | 38 | – |
| | | | – | – | 50 |
| DMS | Input | Mode select signal input for on-chip debugging, 5 V tolerant. | P55/$\overline{\text{SCKF2}}$/KR5/RTP05 | 40 | – |
| | | | – | – | 52 |
| $\overline{\text{DRST}}$ | Input | Reset signal input for on-chip debugging, 5 V tolerant. | P56/INTP05 | 41 | – |
| | | | – | – | 53 |

**Remark**   JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(3/9)

| Pin Name | I/O | Function | Alternate Function | Pin No. JG3-U | Pin No. JH3-U |
|---|---|---|---|---|---|
| EV$_{DD}$ | – | Positive power supply for external devices (same potential as V$_{DD}$) | – | 34, 63, 80 | 46, 61, 85,107 |
| EVTT0 | Input | External event count input of TMT0 | P94/TIAA31/TOAA31/TENC00 | 46 | – |
| | | | P94/TIAA31/TOAA31/TENC00/A4 | – | 58 |
| EVTAB1 | Input | External event count input of TAB1 | P65/TOAB1B3/$\overline{CS3}$ | 70 | – |
| | | | P65/TOAB1B3 | – | 95 |
| FLMD0 | Input | Flash memory programming mode setting pin | – | 8 | 12 |
| FLMD1 | Input | | PDL5/AD5 | 76 | 103 |
| $\overline{HLDAK}$ | Output | Bus hold acknowledge output | PCM2 | – | 10 |
| $\overline{HLDRQ}$ | Input | Bus hold request input | PCM3 | – | 11 |
| INTP00 | Input | External interrupt request input (maskable, analog noise elimination) Analog noise elimination or digital noise elimination selectable for INTP02 pin. 5 V tolerant (INTP00 to INTP05, INTP07 to INTP10) | P00 | – | 8 |
| INTP01 | | | P01 | – | 9 |
| INTP02 | | | P03/ADTRG/UCLK | 7 | 7 |
| INTP03 | | | P04 | – | 26 |
| INTP04 | | | P05 | – | 27 |
| INTP05 | | | P56/$\overline{DRST}$ | 41 | – |
| | | | P24 | – | 36 |
| INTP06 | | | P25 | – | 28 |
| INTP07 | | | P30/TXDC0/SOF4 | 25 | 37 |
| INTP08 | | | P31/RXDC0/SIF4 | 26 | 38 |
| INTP09 | | | P34/TIAA10/TOAA10/TOAA1OFF/PPON | 29 | 41 |
| INTP10 | | | P42/$\overline{SCKF0}$ | 24 | 31 |
| INTP11 | | | P96/TIAA21/TOAA21/A6 | – | 62 |
| | | | P96/TIAA21/TOAA21 | 48 | – |
| INTP12 | | | P98/SOF1 | 50 | – |
| | | | P98/SOF1/A8 | – | 64 |
| INTP13 | | | P99/$\overline{SCKF1}$ | 51 | – |
| | | | P99/$\overline{SCKF1}$/A9 | – | 65 |
| INTP14 | | | P910/SIF3/TXDC2 | 52 | – |
| | | | P910/SIF3/TXDC2/A10 | – | 66 |
| INTP15 | | | P911/SOF3/RXDC2 | 53 | – |
| | | | P911/SOF3/RXDC2/A11 | – | 67 |
| INTP16 | | | P913/TOAB1OFF | 55 | – |
| | | | P913/TOAB1OFF/A13 | – | 69 |
| INTP17 | | | P914/TIAA51/TOAA51 | 56 | – |
| | | | P914/TIAA51/TOAA51/A14 | – | 70 |
| INTP18 | | | P915/TIAA50/TOAA50 | 57 | – |
| | | | P915/TIAA50/TOAA50/A15 | – | 71 |

**Remark**   JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(4/9)

| Pin Name | I/O | Function | Alternate Function | Pin No. | |
|---|---|---|---|---|---|
| | | | | JG3-U | JH3-U |
| KR0 | Input | Key interrupt input (on-chip analog noise eliminator) | P50/TIAB01/TOAB01/RTP00 /UDMARQ1 | 35 | 47 |
| KR1 | | 5 V tolerant (KR0 to KR5) | P51/TIAB02/TOAB02/RTP01 /UDMARQ1 | 36 | 48 |
| KR2 | | | P52/TIAB03/TOAB03/RTP02/DDI | 37 | – |
| | | | P20/TIAB03/TOAB03/RTP02 | – | 32 |
| KR3 | | | P53/SIF2/TIAB00/TOAB00/RTP03/DDO | 38 | – |
| | | | P21/SIF2/TIAB00/TOAB00/RTP03 | – | 33 |
| KR4 | | | P54/SOF2/RTP04/DCK | 39 | – |
| | | | P22/SOF2/RTP04 | – | 34 |
| KR5 | | | P55/$\overline{\text{SCKF2}}$/RTP05/DMS | 40 | – |
| | | | P23/SCKF2/RTP05 | – | 35 |
| KR6 | | | P90/TXDC1/SDA02 | 42 | – |
| | | | P90/TXDC1/SDA02/A0 | – | 54 |
| KR7 | | | P91/RXDC1/SCL02 | 43 | – |
| | | | P91/RXDC1/SCL02/A1 | – | 55 |
| NMI | Input | External interrupt input (non-maskable, analog noise elimination), 5 V tolerant. | P02 | 6 | 6 |
| $\overline{\text{OCI}}$ | Input | Overcurrent detection input, 5 V tolerant. | P35/TIAA11/TOAA11/RTC1HZ | 30 | 42 |
| PPON | Output | Power supply output for USB, 5 V tolerant. | P34/TIAA10/TOAA10/TOAA1OFF/INTP09 | 29 | 41 |
| $\overline{\text{RD}}$ | Output | Read strobe signal output for external memory | P61/TOAB1B1/TIAB10/TOAB10 | 66 | – |
| | | | PCT4 | – | 87 |
| REGC | – | Connection of regulator output stabilization capacitance (4.7 $\mu$F: recommended value) | – | 10, 61 | 14, 83 |
| $\overline{\text{RESET}}$ | Input | System reset input | – | 14 | 18 |
| RTC1HZ | Output | Real-time counter correction clock (1 Hz) output | P35/TIAA11/TOAA11/$\overline{\text{OCI}}$ | 30 | 42 |
| RTCCL | Output | Real-time counter clock (32 kHz primary oscillation) output, 5 V tolerant. | P33/TIAA01/TOAA01/RTCDIV | 28 | 40 |
| RTCDIV | Output | Real-time counter clock (32 kHz division) output, 5 V tolerant. | P33/TIAA01/TOAA01/RTCCL | 28 | 40 |
| RTP00 | Output | Real-time output port | P50/TIAB01/KR0/TOAB01/$\overline{\text{UDMARQ1}}$ | 35 | 47 |
| RTP01 | | N-ch open-drain output selectable | P51/TIAB02/KR1/TOAB02/$\overline{\text{UDMAAK1}}$ | 36 | 48 |
| RTP02 | | 5 V tolerant. | P52/TIAB03/KR2/TOAB03/DDI | 37 | – |
| | | | P20/TIAB03/KR2/TOAB03 | – | 32 |
| RTP03 | | | P53/SIF2/TIAB00/KR3/TOAB00/DDO | 38 | – |
| | | | P21/SIF2/TIAB00/KR3/TOAB00 | – | 33 |
| RTP04 | | | P54/SOF2/KR4/DCK | 39 | – |
| | | | P22/SOF2/KR4 | – | 34 |
| RTP05 | | | P55/$\overline{\text{SCKF2}}$/KR5/DMS | 40 | – |
| | | | P23/SCKF2/KR5 | – | 35 |

**Remark** JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(5/9)

| Pin Name | I/O | Function | Alternate Function | Pin No. JG3-U | Pin No. JH3-U |
|----------|-----|----------|--------------------|------|------|
| RXDC0 | Input | Serial receive data input (UARTC0 to UARTC4) 5 V tolerant (RXDC0, RXDC3, RXDC4) | P31/SIF4/INTP08 | 26 | 38 |
| RXDC1 | | | P91/KR7/SCL02 | 43 | – |
| | | | P91/KR7/SCL02/A1 | – | 55 |
| RXDC2 | | | P911/SOF3/INTP15 | 53 | – |
| | | | P911/SOF3/INTP15/A11 | – | 67 |
| RXDC3 | | | P37/SDA00/$\overline{\text{UDMAAK0}}$ | 32 | 44 |
| RXDC4 | | | P41/SOF0/SCL01 | 23 | 30 |
| $\overline{\text{SCKF0}}$ | I/O | Serial clock I/O (CSIF0 to CSIF4) N-ch open-drain output selectable 5 V tolerant ($\overline{\text{SCKF0}}$, $\overline{\text{SCKF2}}$, $\overline{\text{SCKF4}}$) | P42/INTP10 | 24 | 31 |
| $\overline{\text{SCKF1}}$ | | | P99/INTP13 | 51 | – |
| | | | P99/INTP13/A9 | – | 65 |
| $\overline{\text{SCKF2}}$ | | | P55/KR5/RTP05/DMS | 40 | – |
| | | | P23/KR5/RTP05 | – | 35 |
| $\overline{\text{SCKF3}}$ | | | P912 | 54 | – |
| | | | P912/A12 | – | 68 |
| $\overline{\text{SCKF4}}$ | | | P32/ASCKC0/TIAA00/TOAA00 | 27 | 39 |
| SCL00 | I/O | Serial clock I/O (I²C00 to I²C02) N-ch open-drain output selectable 5 V tolerant (SCL00, SCL01) | P36/TXDC3/$\overline{\text{UDMARQ0}}$ | 31 | 43 |
| SCL01 | | | P41/SOF0/RXDC4 | 23 | 30 |
| SCL02 | | | P91/KR7/RXDC1 | 43 | – |
| | | | P91/KR7/RXDC1/A1 | – | 55 |
| SDA00 | I/O | Serial transmit/receive data I/O (I²C00 to I²C02) N-ch open-drain output selectable 5 V tolerant (SDA00, SDA01) | P37/RXDC3/$\overline{\text{UDMAAK0}}$ | 32 | 44 |
| SDA01 | | | P40/SIF0/TXDC4 | 22 | 29 |
| SDA02 | | | P90/KR6/TXDC1 | 42 | – |
| | | | P90/KR6/TXDC1/A0 | – | 54 |
| SIF0 | Input | Serial receive data input (CSIF0 to CSIF4) 5 V tolerant (SIF0, SIF2, SIF4) | P40/TXDC4/SDA01 | 22 | 29 |
| SIF1 | | | P97/TIAA20/TOAA20 | 49 | – |
| | | | P97/TIAA20/TOAA20/A7 | – | 63 |
| SIF2 | | | P53/TIAB00/KR3/TOAB00/RTP03/DDO | 38 | – |
| | | | P21/TIAB00/KR3/TOAB00/RTP03 | – | 33 |
| SIF3 | | | P910/TXDC2/INTP14 | 52 | – |
| | | | P910/TXDC2/INTP14/A10 | – | 66 |
| SIF4 | | | P31/RXDC0/INTP08 | 26 | 38 |
| SOF0 | Output | Serial transmit data output (CSIF0 to CSIF4) N-ch open-drain output selectable 5 V tolerant (SOF0, SOF2, SOF4) | P41/RXDC4/SCL01 | 23 | 30 |
| SOF1 | | | P98/INTP12 | 50 | – |
| | | | P98/INTP12/A8 | – | 64 |
| SOF2 | | | P54/KR4/RTP04/DCK | 39 | – |
| | | | P22/KR4/RTP04 | – | 34 |
| SOF3 | | | P911/RXDC2/INTP15 | 53 | – |
| | | | P911/RXDC2/INTP15/A11 | – | 67 |
| SOF4 | | | P30/TXDC0/INTP07 | 25 | 37 |

**Remark**    JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(6/9)

| Pin Name | I/O | Function | Alternate Function | Pin No. JG3-U | Pin No. JH3-U |
|---|---|---|---|---|---|
| TECR0 | Input | TMT0 encoder clear input | P93/TIT00/TOT00 | 45 | – |
| | | | P93/TIT00/TOT00/A3 | – | 57 |
| TENC00 | | TMT0 encoder input | P94/TIAA31/TOAA31/EVTT0 | 46 | – |
| | | | P94/TIAA31/TOAA31/EVTT0/A4 | – | 58 |
| TENC01 | | | P92/TIT01/TOT01 | 44 | – |
| | | | P92/TIT01/TOT01/A2 | – | 56 |
| TIAA00 | Input | External event count input/capture trigger input/external trigger input (TAA0),  5 V tolerant. | P32/ASCKC0/$\overline{\text{SCKF4}}$/TOAA00 | 27 | 39 |
| TIAA01 | | Capture trigger input (TAA0),  5 V tolerant. | P33/TOAA01/RTCDIV/RTCCL | 28 | 40 |
| TIAA10 | | External event count input/capture trigger input/external trigger input (TAA1),  5 V tolerant. | P34/TOAA10/TOAA1OFF/INTP09/PPON | 29 | 41 |
| TIAA11 | | Capture trigger input (TAA1),  5 V tolerant. | P35/TOAA11/RTC1HZ/$\overline{\text{OCI}}$ | 30 | 42 |
| TIAA20 | | External event count input/capture trigger input/external trigger input (TAA2) | P97/SIF1/TOAA20 | 49 | – |
| | | | P97/SIF1/TOAA20/A7 | – | 63 |
| TIAA21 | | Capture trigger input (TAA2) | P96/TOAA21/INTP11 | 48 | – |
| | | | P96/TOAA21/INTP11/A6 | – | 62 |
| TIAA30 | | External event count input/capture trigger input/external trigger input (TAA3) | P95/TOAA30 | 47 | – |
| | | | P95/TOAA30/A5 | – | 59 |
| TIAA31 | | Capture trigger input (TAA3) | P94/TOAA31/TENC00/EVTT0 | 46 | – |
| | | | P94/TOAA31/TENC00/EVTT0/A4 | – | 58 |
| TIAA50 | | External event count input/capture trigger input/external trigger input (TAA5) | P915/TOAA50/INTP18 | 57 | – |
| | | | P915/TOAA50/INTP18/A15 | – | 71 |
| TIAA51 | | Capture trigger input (TAA5) | P914/TOAA51/INTP17 | 56 | – |
| | | | P914/TOAA51/INTP17/A14 | – | 70 |
| TIAB00 | Input | External event count input/capture trigger input/external trigger input (TAB0),  5 V tolerant. | P53/SIF2/KR3/TOAB00/RTP03/DDO | 38 | – |
| | | | P21/SIF2/KR3/TOAB00/RTP03 | – | 33 |
| TIAB01 | | Capture trigger input (TAB0) | P50/KR0/TOAB01/RTP00/$\overline{\text{UDMARQ1}}$ | 35 | 47 |
| TIAB02 | | 5 V tolerant. | P51/KR1/TOAB02/RTP01/$\overline{\text{UDMAAK1}}$ | 36 | 48 |
| TIAB03 | | | P52/KR2/TOAB03/RTP02/DDI | 37 | – |
| | | | P20/KR2/TOAB03/RTP02 | – | 32 |
| TIAB10 | Input | Capture trigger input (TAB1) | P61/TOAB1B1/TOAB10/$\overline{\text{RD}}$ | 66 | – |
| | | | P61/TOAB1B1/TOAB10 | – | 91 |
| TIAB11 | | | P60/TOAB1T1/TOAB11/$\overline{\text{WAIT}}$ | 65 | – |
| | | | P60/TOAB1T1/TOAB11 | – | 90 |
| TIAB12 | | | P62/TOAB1T2/TOAB12/ASTB | 67 | – |
| | | | P62/TOAB1T2/TOAB12 | – | 92 |
| TIAB13 | | | P64/TOAB1T3/TOAB13/$\overline{\text{CS2}}$ | 69 | – |
| | | | P64/TOAB1T3/TOAB13 | – | 94 |

**Remark**   JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(7/9)

| Pin Name | I/O | Function | Alternate Function | Pin No. JG3-U | Pin No. JH3-U |
|----------|-----|----------|--------------------|-----|-----|
| TIT00 | Input | TMT0 external trigger input/capture trigger input | P93/TECR0/TOT00 | 45 | – |
| | | | P93/TECR0/TOT00/A3 | – | 57 |
| TIT01 | Input | TMT0 capture trigger input | P92/TENC01/TOT01 | 44 | – |
| | | | P92/TENC01/TOT01/A2 | – | 56 |
| TOAA00 | Output | Timer output (TAA0) | P32/ASCKC0/$\overline{\text{SCKF4}}$/TIAA00 | 27 | 39 |
| TOAA01 | | N-ch open-drain output selectable, 5 V tolerant. | P33/TIAA01/RTCDIV/RTCCL | 28 | 40 |
| TOAA10 | | Timer output (TAA1) | P34/TIAA10/TOAA1OFF/INTP09/PPON | 29 | 41 |
| TOAA11 | | N-ch open-drain output selectable, 5 V tolerant. | P35/TIAA11/RTC1HZ/$\overline{\text{OCI}}$ | 30 | 42 |
| TOAA1OFF | Input | TAA1 high-impedance output control signal input, 5 V tolerant. | P34/TIAA10/TOAA10/INTP09/PPON | 29 | 41 |
| TOAA20 | Output | Timer output (TAA2) | P97/SIF1/TIAA20 | 49 | – |
| | | N-ch open-drain output selectable | P97/SIF1/TIAA20/A7 | – | 63 |
| TOAA21 | | | P96/TIAA21/INTP11 | 48 | – |
| | | | P96/TIAA21/INTP11/A6 | – | 62 |
| TOAA30 | | Timer output (TAA3) | P95/TIAA30 | 47 | – |
| | | N-ch open-drain output selectable | P95/TIAA30/A5 | – | 59 |
| TOAA31 | | | P94/TIAA31/TENC00/EVTT0 | 46 | – |
| | | | P94/TIAA31/TENC00/EVTT0/A4 | – | 58 |
| TOAA50 | | Timer output (TAA5) | P915/TIAA50/INTP18 | 57 | – |
| | | N-ch open-drain output selectable | P915/TIAA50/INTP18/A15 | – | 71 |
| TOAA51 | | | P914/TIAA51/INTP17 | 56 | – |
| | | | P914/TIAA51/INTP17/A14 | – | 70 |
| TOAB00 | Output | Timer output (TAB0) | P53/SIF2/TIAB00/KR3/RTP03/DDO | 38 | – |
| | | N-ch open-drain output selectable, | P21/SIF2/TIAB00/KR3/RTP03 | – | 33 |
| TOAB01 | | 5 V tolerant. | P50/TIAB01/KR0/RTP00/$\overline{\text{UDMARQ1}}$ | 35 | 47 |
| TOAB02 | | | P51/TIAB02/KR1/RTP01/$\overline{\text{UDMAAK1}}$ | 36 | 48 |
| TOAB03 | | | P52/TIAB03/KR2/RTP02/DDI | 37 | – |
| | | | P20/TIAB03/KR2/RTP02 | – | 32 |
| TOAB1OFF | Input | TAB1 high-impedance output control signal input | P913/INTP16 | 55 | – |
| | | | P913/INTP16/A13 | – | 69 |
| TOAB10 | Output | Timer output (TAB1) | P61/TOAB1B1/TIAB10/$\overline{\text{RD}}$ | 66 | – |
| | | | P61/TOAB1B1/TIAB10 | – | 91 |
| TOAB11 | | | P60/TOAB1T1/TIAB11/$\overline{\text{WAIT}}$ | 65 | – |
| | | | P60/TOAB1T1/TIAB11 | – | 90 |
| TOAB12 | | | P62/TOAB1T2/TIAB12/ASTB | 67 | – |
| | | | P62/TOAB1T2/TIAB12 | – | 92 |
| TOAB13 | | | P64/TOAB1T3/TIAB13/$\overline{\text{CS2}}$ | 69 | – |
| | | | P64/TOAB1T3/TIAB13 | – | 94 |

**Remark** JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

| Pin Name | I/O | Function | Alternate Function | Pin No. JG3-U | Pin No. JH3-U |
|---|---|---|---|---|---|
| TOAB1B1 | Output | Pulse signal output for 6-phase PWM low-arm of TAB1 | P61/TIAB10/TOAB10/$\overline{RD}$ | 66 | – |
| | | | P61/TIAB10/TOAB10 | – | 91 |
| TOAB1B2 | | | P63/TRGAB1/$\overline{CS0}$ | 68 | – |
| | | | P63/TRGAB1 | – | 93 |
| TOAB1B3 | | | P65/EVTAB1/$\overline{CS3}$ | 70 | – |
| | | | P65/EVTAB1 | – | 95 |
| TOAB1T1 | Output | Pulse signal output for 6-phase PWM high-arm of TAB1 | P60/TOAB11/TIAB11/$\overline{WAIT}$ | 65 | – |
| | | | P60/TIAB11/TOAB11 | – | 90 |
| TOAB1T2 | | | P62/TIAB12/TOAB12/ASTB | 67 | – |
| | | | P62/TIAB12/TOAB12 | – | 92 |
| TOAB1T3 | | | P64/TOAB13/TIAB13/$\overline{CS2}$ | 69 | – |
| | | | P64/TIAB13/TOAB13 | – | 94 |
| TOT00 | Output | TMT0 timer output | P93/TECR0/TIT00 | 45 | – |
| | | | P93/TECR0/TIT00/A3 | – | 57 |
| TOT01 | | | P92/TENC01/TIT01 | 44 | – |
| | | | P92/TENC01/TIT01/A2 | – | 56 |
| TRGAB1 | Input | External trigger input of TAB1 | P63/TOAB1B2/$\overline{CS0}$ | 68 | – |
| | | | P63/TOAB1B2 | – | 93 |
| TXDC0 | Output | Serial transmit data output (UARTC0 to UARTC4) N-ch open-drain output selectable, 5 V tolerant (TXDC0, TXDC3, TXDC4) | P30/SOF4/INTP07 | 25 | 37 |
| TXDC1 | | | P90/KR6/SDA02 | 42 | – |
| | | | P90/KR6/SDA02/A0 | – | 54 |
| TXDC2 | | | P910/SIF3/INTP14 | 52 | – |
| | | | P910/SIF3/INTP14/A10 | – | 66 |
| TXDC3 | | | P36/SCL00/$\overline{UDMARQ0}$ | 31 | 43 |
| TXDC4 | | | P40/SIF0/SDA01 | 22 | 29 |
| UCLK | Input | USB clock signal input, 5 V tolerant. | P03/INTP02/ADTRG | 7 | 7 |
| $\overline{UDMAAK0}$ | Output | DMA acknowledge for USB, 5 V tolerant. | P37/RXDC3/SDA00 | 32 | 44 |
| $\overline{UDMAAK1}$ | | DMA acknowledge for USB, 5 V tolerant. | P51/TIAB02/KR1/TOAB02/RTP01 | 36 | 48 |
| $\overline{UDMARQ0}$ | Input | DMA request for USB, 5 V tolerant. | P36/TXDC3/SCL00 | 31 | 43 |
| $\overline{UDMARQ1}$ | | DMA request for USB, 5 V tolerant. | P50/TIAB01/KR0/TOAB01/RTP00 | 35 | 47 |
| UDMF | I/O | USB data I/O (−) function | – | 17 | 21 |
| UDMH | | USB data I/O (−) host | – | 20 | 24 |
| UDPF | | USB data I/O (+) function | – | 18 | 22 |
| UDPH | | USB data I/O (+) host | – | 21 | 25 |
| $UV_{DD}$ | – | 3.3 V Positive power supply for USB | – | 19 | 23 |
| $V_{DD}$ | – | Positive power supply pin for internal unit | – | 9, 60 | 13, 82 |
| $V_{SS}$ | – | Ground potential for internal unit | – | 11, 33 62, 79 | 15, 45 60, 84 106 |

**Remark**  JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

(9/9)

| Pin Name | I/O | Function | Alternate Function | Pin No. JG3-U | Pin No. JH3-U |
|----------|-----|----------|-------------------|------|------|
| $\overline{\text{WAIT}}$ | Input | External wait input | P60/TOAB1T1/TIAB11/TOAB11 | 65 | – |
| | | | PCM0 | – | 89 |
| $\overline{\text{WR0}}$ | Output | Write strobe for external memory (lower 8 bits) | PCT0 | 58 | 76 |
| $\overline{\text{WR1}}$ | | Write strove for external memory (higher 8 bits) | PCT1 | 59 | 77 |
| X1 | Input | Connection of resonator for main clock | – | 12 | 16 |
| X2 | – | | – | 13 | 17 |
| XT1 | Input | Connection of resonator for subclock | – | 15 | 19 |
| XT2 | – | | – | 16 | 20 |

**Remark**  JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

## 2.2  Pin States

The operation states of pins in the various operation modes are described below.

**Table 2-2.  Pin Operation States in Various Modes**

| Pin Name | When Power Is Turned On[1] | During Reset (Other than When Power Is Turned On) | HALT Mode[2] | IDLE1, IDLE2, Sub-IDLE Mode[2] | STOP Mode[2] | Idle State[3] | Bus Hold |
|---|---|---|---|---|---|---|---|
| $\overline{\text{DRST}}$ | Pull down | Pull down[4] | Held | Held | Held | Held | Held |
| P10/ANO0, P11/ANO1 | Undefined | Hi-Z | Held | Held | Hi-Z | Held | Held |
| AD0 to AD15 | Hi-Z[5] | Hi-Z[5] | Notes 6, 7 | Hi-Z | Hi-Z | Held | Hi-Z |
| A0 to A15 | | | Undefined[6,8] | | | | |
| A16 to A21 | | | Undefined[6] | | | | |
| $\overline{\text{WAIT}}$ | | | − | − | − | − | − |
| CLKOUT | | | Operating | L | L | Operating | Operating |
| $\overline{\text{WR0}}$, $\overline{\text{WR1}}$ | | | H[6] | H | H | H | Hi-Z |
| $\overline{\text{RD}}$ | | | | | | | |
| ASTB | | | | | | | |
| $\overline{\text{HLDAK}}$ | | | Operating[6] | | | | L |
| $\overline{\text{HLDRQ}}$ | | | | − | − | − | Operating |
| Other port pins | Hi-Z | Hi-Z | Held | Held | Held | Held | Held |

**Notes 1.** Duration until 1 ms elapses after the supply voltage reaches the operating supply voltage range (lower limit) when the power is turned on.

**2.** Operates while alternate functions are operating.

**3.** The state of the pins in the idle state inserted after the T3 state is shown.

**4.** Pulled down during external reset.  During internal reset by the watchdog timer or clock monitor, etc., the state of this pin differs according to the OCDM.OCDM0 bit setting.

**5.** The bus control pins function alternately as port pins, so they are initialized to the input mode (port mode).

**6.** Operates even in the HALT mode, during DMA operation.

**7.** In separate bus mode:        Hi-Z

In multiplexed bus mode:   Undefined

**8.** In separate bus mode

**Remark**   Hi-Z:   High impedance

Held:   The state during the immediately preceding external bus cycle is held.

L:        Low-level output

H:        High-level output

−:        Input without sampling (not acknowledged)

## 2.3 Pin I/O Circuit Types, I/O Buffer Power Supplies and Connection of Unused Pins

**Table 2-3.  Pin I/O Circuit Types, I/O Buffer Power Supplies and Connection of Unused Pins (1/4)**

| Pin Name | Alternate Function | I/O Circuit Type | Recommended Connection | | JG3-U | JH3-U |
|---|---|---|---|---|---|---|
| P00 | INTP00 | 10-D | Input: | Independently connect to $EV_{DD}$ or $V_{SS}$ via a resistor. | – | √ |
| P01 | INTP01 | | | | – | √ |
| P02 | NMI | | Output: | Leave open. | √ | √ |
| P03 | INTP02/ADTRG/UCLK | | | | √ | √ |
| P04 | INTP03 | | | | – | √ |
| P05 | INTP04 | | | | – | √ |
| P10 | ANO0 | 12-D | Input: | Independently connect to $AV_{REF1}$ or $AV_{SS}$ via a resistor. | √ | √ |
| P11 | ANO1 | | | | √ | √ |
| | | | Output: | Leave open. | | |
| P20 | TIAB03/KR2/TOAB03/RTP02 | 10-D | Input: | Independently connect to $EV_{DD}$ or $V_{SS}$ via a resistor. | – | √ |
| P21 | SIF2/TIAB00/KR3/TOAB00/RTP03 | | | | – | √ |
| P22 | SOF2/KR4/RTP04 | | Output: | Leave open. | – | √ |
| P23 | $\overline{SCKF2}$/KR5/RTP05 | | | | – | √ |
| P24 | INTP05 | | | | – | √ |
| P25 | INTP06 | | | | – | √ |
| P30 | TXDC0/SOF4/INTP07 | 10-D | Input: | Independently connect to $EV_{DD}$ or $V_{SS}$ via a resistor. | √ | √ |
| P31 | RXDC0/SIF4/INTP08 | | | | √ | √ |
| P32 | ASCKC0/$\overline{SCKF4}$/TIAA00/TOAA00 | | Output: | Leave open. | √ | √ |
| P33 | TIAA01/TOAA01/RTCDIV/RTCCL | | | | √ | √ |
| P34 | TIAA10/TOAA10/TOAA1OFF/INTP09/PPON | | | | √ | √ |
| P35 | TIAA11/TOAA11/RTC1HZ/$\overline{OCI}$ | | | | √ | √ |
| P36 | TXDC3/SCL00/$\overline{UDMARQ0}$ | | | | √ | √ |
| P37 | RXDC3/SDA00/$\overline{UDMAAK0}$ | | | | √ | √ |
| P40 | SIF0/TXDC4/SDA01 | 10-D | Input: | Independently connect to $EV_{DD}$ or $V_{SS}$ via a resistor. | √ | √ |
| P41 | SOF0/RXDC4/SCL01 | | | | √ | √ |
| P42 | $\overline{SCKF0}$/INTP10 | | Output: | Leave open. | √ | √ |
| P50 | TIAB01/KR0/TOAB01/RTP00/$\overline{UDMARQ1}$ | 10-D | Input: | Independently connect to $EV_{DD}$ or $V_{SS}$ via a resistor. | √ | √ |
| P51 | TIAB02/KR1/TOAB02/RTP01/$\overline{UDMAAK1}$ | | | | √ | √ |
| P52 | TIAB03/KR2/TOAB03/RTP02/DDI | | Output: | Leave open. | √ | – |
| P53 | SIF2/TIAB00/KR3/TOAB00/RTP03/DDO | | | | √ | – |
| P54 | SOF2/KR4/RTP04/DCK | | | | √ | – |
| P55 | $\overline{SCKF2}$/KR5/RTP05/DMS | | | | √ | – |
| P56 | INTP05/$\overline{DRST}$ | 10-N | Input: | Independently connect to $V_{SS}$ via a resistor.  Fixing to $V_{DD}$ level is prohibited. | √ | – |
| | | | Output: | Leave open. Internally pull-down after reset by $\overline{RESET}$ pin. | | |

**Remark**    JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

**Table 2-3.  Pin I/O Circuit Types, I/O Buffer Power Supplies and Connection of Unused Pins (2/4)**

| Pin Name | Alternate Function | I/O Circuit Type | Recommended Connection | | | JG3-U | JH3-U |
|---|---|---|---|---|---|---|---|
| P60 | TOAB1T1/TIAB11/TOAB11/$\overline{\text{WAIT}}$ | 10-D | Input: | Independently connect to EV$_{DD}$ or V$_{SS}$ via a resistor. | | √ | – |
| | TOAB1T1/TIAB11/TOAB11 | | Output: | Leave open. | | – | √ |
| P61 | TOAB1B1/TIAB10/TOAB10/$\overline{\text{RD}}$ | | | | | √ | – |
| | TOAB1B1/TIAB10/TOAB10 | | | | | – | √ |
| P62 | TOAB1T2/TIAB12/TOAB12/ASTB | | | | | √ | – |
| | TOAB1T2/TIAB12/TOAB12 | | | | | – | √ |
| P63 | TOAB1B2/TRGAB1/$\overline{\text{CS0}}$ | | | | | √ | – |
| | TOAB1B2/TRGAB1 | | | | | – | √ |
| P64 | TOAB1T3/TIAB13/TOAB13/$\overline{\text{CS2}}$ | | | | | √ | – |
| | TOAB1T3/TIAB13/TOAB13 | | | | | – | √ |
| P65 | TOAB1B3/EVTAB1/$\overline{\text{CS3}}$ | | | | | √ | – |
| | TOAB1B3/EVTAB1 | | | | | – | √ |
| P70 to P711 | ANI0 to ANI11 | 11-G | Input: | Independently connect to AV$_{REF0}$ or AV$_{SS}$ via a resistor. | | √ | √ |
| | | | Output: | Leave open. | | | |
| P90 | KR6/TXDC1/SDA02 | 10-D | Input: | Independently connect to EV$_{DD}$ or V$_{SS}$ via a resistor. | | √ | – |
| | KR6/TXDC1/SDA02/A0 | | Output: | Leave open. | | – | √ |
| P91 | KR7/RXDC1/SCL02 | | | | | √ | – |
| | KR7/RXDC1/SCL02/A1 | | | | | – | √ |
| P92 | TENC01/TIT01/TOT01 | | | | | √ | – |
| | TENC01/TIT01/TOT01/A2 | | | | | – | √ |
| P93 | TECR0/TIT00/TOT00 | | | | | √ | – |
| | TECR0/TIT00/TOT00/A3 | | | | | – | √ |
| P94 | TIAA31/TOAA31/TENC00/EVTT0 | | | | | √ | – |
| | TIAA31/TOAA31/TENC00/EVTT0/A4 | | | | | – | √ |
| P95 | TIAA30/TOAA30 | | | | | √ | – |
| | TIAA30/TOAA30/A5 | | | | | – | √ |
| P96 | TIAA21/TOAA21/INTP11 | | | | | √ | – |
| | TIAA21/TOAA21/INTP11/A6 | | | | | – | √ |
| P97 | SIF1/TIAA20/TOAA20 | | | | | √ | – |
| | SIF1/TIAA20/TOAA20/A7 | | | | | – | √ |
| P98 | SOF1/INTP12 | | | | | √ | – |
| | SOF1/INTP12/A8 | | | | | – | √ |
| P99 | $\overline{\text{SCKF1}}$/INTP13 | | | | | √ | – |
| | $\overline{\text{SCKF1}}$/INTP13/A9 | | | | | – | √ |
| P910 | SIF3/TXDC2/INTP14 | | | | | √ | – |
| | SIF3/TXDC2/INTP14/A10 | | | | | – | √ |
| P911 | SOF3/RXDC2/INTP15 | | | | | √ | – |
| | SOF3/RXDC2/INTP15/A11 | | | | | – | √ |
| P912 | $\overline{\text{SCKF3}}$ | | | | | √ | – |
| | $\overline{\text{SCKF3}}$/A12 | | | | | – | √ |

**Remark**   JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

**Table 2-3.  Pin I/O Circuit Types, I/O Buffer Power Supplies and Connection of Unused Pins (3/4)**

| Pin Name | Alternate Function | I/O Circuit Type | Recommended Connection | | | JG3-U | JH3-U |
|---|---|---|---|---|---|---|---|
| P913 | TOAB1OFF/INTP16 | 10-D | Input: | Independently connect to EV$_{DD}$ or V$_{SS}$ via a resistor. | | √ | – |
| | TOAB1OFF/INTP16/A13 | | Output: | Leave open. | | – | √ |
| P914 | TIAA51/TOAA51/INTP17 | | | | | √ | – |
| | TIAA51/TOAA51/INTP17/A14 | | | | | – | √ |
| P915 | TIAA50/TOAA50/INTP18 | | | | | √ | – |
| | TIAA50/TOAA50/INTP18/A15 | | | | | – | √ |
| PCM0 | $\overline{\text{WAIT}}$ | 5 | Input: | Independently connect to EV$_{DD}$ or V$_{SS}$ via a resistor. | | – | √ |
| PCM1 | CLKOUT | | Output: | Leave open. | | √ | √ |
| PCM2 | $\overline{\text{HLDAK}}$ | | | | | – | √ |
| PCM3 | $\overline{\text{HLDRQ}}$ | | | | | – | √ |
| PCS0 | $\overline{\text{CS0}}$ | 5 | Input: | Independently connect to EV$_{DD}$ or V$_{SS}$ via a resistor. | | – | √ |
| PCS2 | $\overline{\text{CS2}}$ | | Output: | Leave open. | | – | √ |
| PCS3 | $\overline{\text{CS3}}$ | | | | | – | √ |
| PCT0 | $\overline{\text{WR0}}$ | 5 | Input: | Independently connect to EV$_{DD}$ or V$_{SS}$ via a resistor. | | √ | √ |
| PCT1 | $\overline{\text{WR1}}$ | | Output: | Leave open. | | √ | √ |
| PCT4 | $\overline{\text{RD}}$ | | | | | – | √ |
| PCT6 | ASTB | | | | | – | √ |
| PDH0 to PDH7 | A16 to A23 | 5 | Input: | Independently connect to EV$_{DD}$ or V$_{SS}$ via a resistor. | | – | √ |
| | | | Output: | Leave open. | | | |
| PDL0 to PDL4 | AD0 to AD4 | 5 | Input: | Independently connect to EV$_{DD}$ or V$_{SS}$ via a resistor. | | √ | √ |
| PDL5 | AD5/FLMD1 | | Output: | Leave open. | | √ | √ |
| PDL6 to PDL15 | AD6 to AD15 | | | | | √ | √ |
| AV$_{REF0}$ | – | – | Always connect to power supply. (The same applies during standby.) | | | √ | √ |
| AV$_{REF1}$ | – | – | | | | √ | √ |
| AV$_{SS}$ | – | – | Always directly connect to ground. (The same applies during standby.) | | | √ | √ |
| DCK | – | – | Always connect to power supply. (The same applies during standby.) | | | – | √ |
| DDI | – | – | | | | – | √ |
| DDO | – | – | Leave open. | | | – | √ |
| DMS | – | – | Always connect to power supply. (The same applies during standby.) | | | – | √ |
| $\overline{\text{DRST}}$ | – | – | Always directly connect to ground. (The same applies during standby.) | | | – | √ |
| EV$_{DD}$ | – | – | Always connect to power supply. (The same applies during standby.) | | | √ | √ |
| FLMD0 | – | – | Directly connect to V$_{SS}$ in other than flash mode. | | | √ | √ |
| REGC | – | – | Connect to regulator output stabilization capacitor. | | | √ | √ |
| $\overline{\text{RESET}}$ | – | 2 | – | | | √ | √ |

**Remark**   JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

**Table 2-3. Pin I/O Circuit Types, I/O Buffer Power Supplies and Connection of Unused Pins (4/4)**

| Pin Name | Alternate Function | I/O Circuit Type | Recommended Connection | JG3-U | JH3-U |
|---|---|---|---|---|---|
| UDMF | – | – | Always connect to power supply. (The same applies during standby.) | √ | √ |
| UDMH | – | – | | √ | √ |
| UDPF | – | – | Always directly connect to ground. (The same applies during standby.) | √ | √ |
| UDPH | – | – | | √ | √ |
| U$_{VDD}$ | – | – | Always connect to power supply. (The same applies during standby.) | √ | √ |
| V$_{DD}$ | – | – | Always connect to power supply. (The same applies during standby.) | √ | √ |
| V$_{SS}$ | – | – | Always directly connect to ground. (The same applies during standby.) | √ | √ |
| X1 | – | – | – | √ | √ |
| X2 | – | – | – | √ | √ |
| XT1 | – | 16-C | Connect to V$_{SS}$ via a resistor. | √ | √ |
| XT2 | – | 16-C | Leave open. | √ | √ |

**Remark**   JG3-U: V850ES/JG3-U, JH3-U: V850ES/JH3-U

**Figure 2-1.  Pin I/O Circuits**
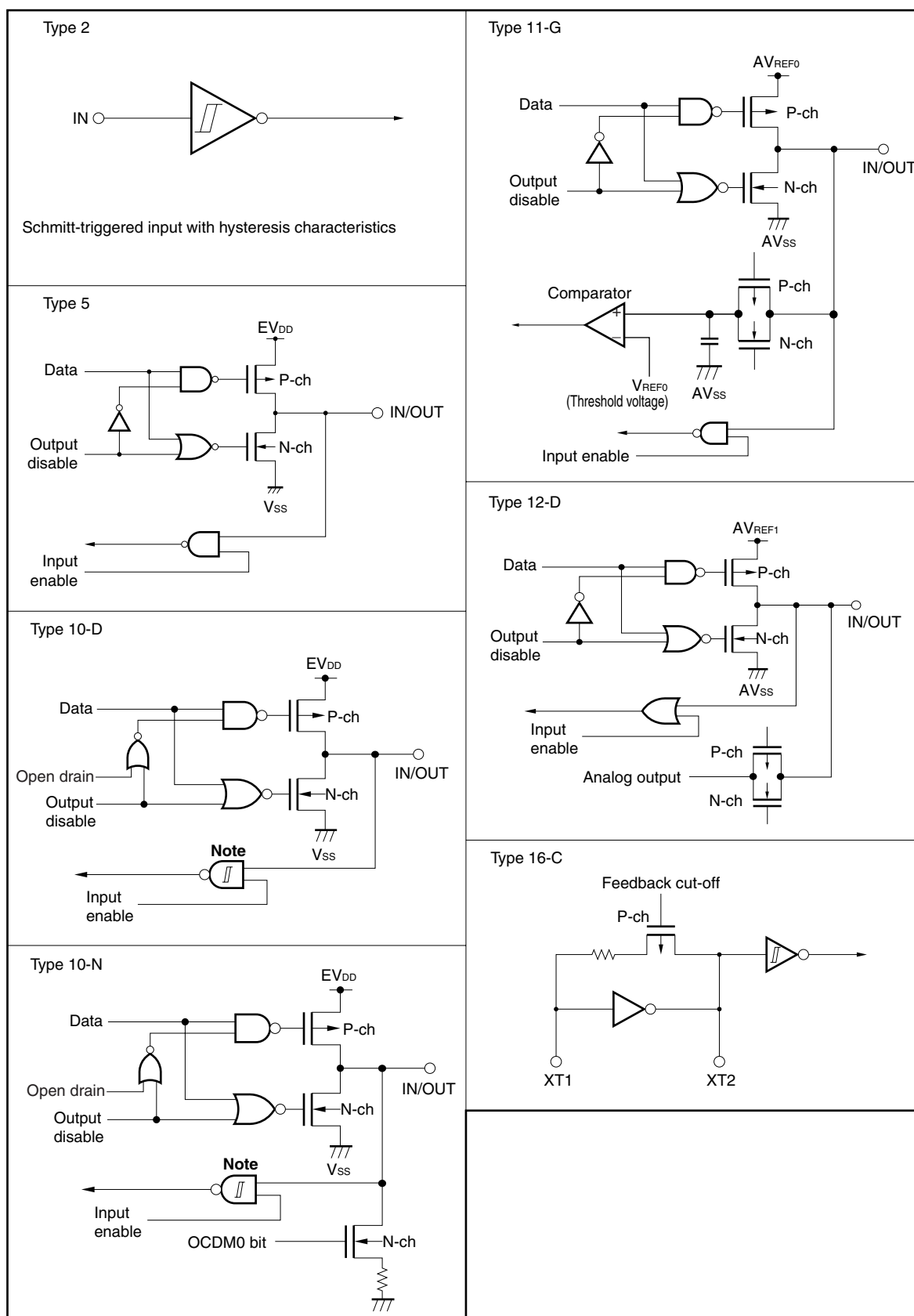


**Note**  Hysteresis characteristics are not available in port mode.

## 2.4  Cautions

When the power is turned on, the following pins may output an undefined level temporarily even during reset.

- P10/ANO0 pin
- P11/ANO1 pin
- DDO pin (V850ES/JH3-U only)
- P53/SIF2/TIAB00/KR3/TOAB00/RTP03/DDO pin (V850ES/JG3-U only)

# CHAPTER 3 CPU FUNCTION

The CPU of the V850ES/JG3-U and V850ES/JH3-U is based on RISC architecture and executes almost all instructions with one clock by using a 5-stage pipeline.

## 3.1 Features

○ Minimum instruction execution time: 20.8 ns (operating with main clock ($f_{XX}$) of 48 MHz: $V_{DD}$ = 2.85 to 3.6 V)

                                         30.5 $\mu$s (operating with subclock ($f_{XT}$) of 32.768 kHz)

○ Memory space    Program (physical address) space: 64 MB linear

                     Data (logical address) space:      4 GB linear

○ General-purpose registers: 32 bits × 32 registers

○ Internal 32-bit architecture

○ 5-stage pipeline control

○ Multiplication/division instruction

○ Saturation operation instruction

○ 32-bit shift instruction: 1 clock

○ Load/store instruction with long/short format

○ Four types of bit manipulation instructions

    • SET1

    • CLR1

    • NOT1

    • TST1

## 3.2  CPU Register Set

The registers of the V850ES/JG3-U and V850ES/JH3-U can be classified into two types: general-purpose program registers and dedicated system registers.  All the registers are 32 bits wide.

For details, refer to the **V850ES Architecture User's Manual**.

---

**(1)  Program register set**

31                                          0

| r0  | (Zero register) |
| r1  | (Assembler-reserved register) |
| r2  | |
| r3  | (Stack pointer (SP)) |
| r4  | (Global pointer (GP)) |
| r5  | (Text pointer (TP)) |
| r6  | |
| r7  | |
| r8  | |
| r9  | |
| r10 | |
| r11 | |
| r12 | |
| r13 | |
| r14 | |
| r15 | |
| r16 | |
| r17 | |
| r18 | |
| r19 | |
| r20 | |
| r21 | |
| r22 | |
| r23 | |
| r24 | |
| r25 | |
| r26 | |
| r27 | |
| r28 | |
| r29 | |
| r30 | (Element pointer (EP)) |
| r31 | (Link pointer (LP)) |

31                                          0

| PC  | (Program counter) |

**(2)  System register set**

31                                          0

| EIPC  | (Interrupt status saving register) |
| EIPSW | (Interrupt status saving register) |

| FEPC  | (NMI status saving register) |
| FEPSW | (NMI status saving register) |

| ECR   | (Interrupt source register) |

| PSW   | (Program status word) |

| CTPC  | (CALLT execution status saving register) |
| CTPSW | (CALLT execution status saving register) |

| DBPC  | (Exception/debug trap status saving register) |
| DBPSW | (Exception/debug trap status saving register) |

| CTBP  | (CALLT base pointer) |

### 3.2.1  Program register set

The program registers include general-purpose registers and a program counter.

#### (1)  General-purpose registers (r0 to r31)

Thirty-two general-purpose registers, r0 to r31, are available.  Any of these registers can be used to store a data variable or an address variable.

However, r0 and r30 are implicitly used by instructions and care must be exercised when these registers are used.  r0 always holds 0 and is used for an operation that uses 0 or addressing of offset 0.  r30 is used by the SLD and SST instructions as a base pointer when these instructions access the memory.  r1, r3 to r5, and r31 are implicitly used by the assembler and C compiler.  When using these registers, save their contents for protection, and then restore the contents after using the registers.  r2 is sometimes used by the real-time OS.  If the real-time OS does not use r2, it can be used as a register for variables.

**Table 3-1.  Program Registers**

| Name | Usage | Operation |
|------|-------|-----------|
| r0 | Zero register | Always holds 0. |
| r1 | Assembler-reserved register | Used as working register to create 32-bit immediate data |
| r2 | Register for address/data variable (if real-time OS does not use r2) | |
| r3 | Stack pointer | Used to create a stack frame when a function is called |
| r4 | Global pointer | Used to access a global variable in the data area |
| r5 | Text pointer | Used as register that indicates the beginning of a text area (area where program codes are located) |
| r6 to r29 | Register for address/data variable | |
| r30 | Element pointer | Used as base pointer to access memory |
| r31 | Link pointer | Used when the compiler calls a function |
| PC | Program counter | Holds the instruction address during program execution |

**Remark**  For further details on the r1, r3 to r5, and r31 that are used in the assembler and C compiler, refer to the **CA850 (C Compiler Package) Assembly Language User's Manual**.

#### (2)  Program counter (PC)

The program counter holds the instruction address during program execution.  The lower 32 bits of this register are valid.  Bits 31 to 26 are fixed to 0.  A carry from bit 25 to 26 is ignored even if it occurs.

Bit 0 is fixed to 0.  This means that execution cannot branch to an odd address.
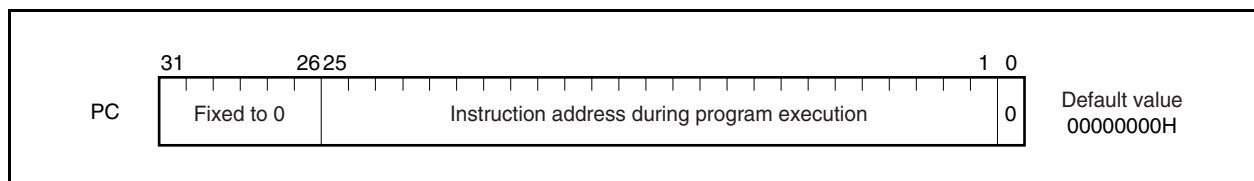
### 3.2.2  System register set

The system registers control the status of the CPU and hold interrupt information.

These registers can be read or written by using system register load/store instructions (LDSR and STSR), using the system register numbers listed below.

**Table 3-2.  System Register Numbers**

| System Register Number | System Register Name | Operand Specification | |
|---|---|---|---|
| | | LDSR Instruction | STSR Instruction |
| 0 | Interrupt status saving register (EIPC)[Note 1] | √ | √ |
| 1 | Interrupt status saving register (EIPSW)[Note 1] | √ | √ |
| 2 | NMI status saving register (FEPC)[Note 1] | √ | √ |
| 3 | NMI status saving register (FEPSW)[Note 1] | √ | √ |
| 4 | Interrupt source register (ECR) | × | √ |
| 5 | Program status word (PSW) | √ | √ |
| 6 to 15 | Reserved for future function expansion (operation is not guaranteed if these registers are accessed) | × | × |
| 16 | CALLT execution status saving register (CTPC) | √ | √ |
| 17 | CALLT execution status saving register (CTPSW) | √ | √ |
| 18 | Exception/debug trap status saving register (DBPC) | √[Note 2] | √[Note 2] |
| 19 | Exception/debug trap status saving register (DBPSW) | √[Note 2] | √[Note 2] |
| 20 | CALLT base pointer (CTBP) | √ | √ |
| 21 to 31 | Reserved for future function expansion (operation is not guaranteed if these registers are accessed) | × | × |

**Notes 1.** Because only one set of these registers is available, the contents of these registers must be saved by program if multiple interrupts are enabled.

**2.** These registers can be accessed only during the interval between the execution of the DBTRAP instruction or illegal opcode and DBRET instruction execution.

**Caution  Even if EIPC or FEPC, or bit 0 of CTPC is set to 1 by the LDSR instruction, bit 0 is ignored when execution is returned to the main routine by the RETI instruction after interrupt servicing (this is because bit 0 of the PC is fixed to 0).  Set an even value to EIPC, FEPC, and CTPC (bit 0 = 0).**

**Remark  √: Can be accessed**
**×: Access prohibited**

**(1)  Interrupt status saving registers (EIPC and EIPSW)**

EIPC and EIPSW are used to save the status when an interrupt occurs.

If a software exception or a maskable interrupt occurs, the contents of the program counter (PC) are saved to EIPC, and the contents of the program status word (PSW) are saved to EIPSW (these contents are saved to the NMI status saving registers (FEPC and FEPSW) if a non-maskable interrupt occurs).

The address of the instruction next to the instruction under execution, except some instructions (see **23.8  Periods in Which Interrupts Are Not Acknowledged by CPU**), is saved to EIPC when a software exception or a maskable interrupt occurs.

The current contents of the PSW are saved to EIPSW.

Because only one set of interrupt status saving registers is available, the contents of these registers must be saved by program when multiple interrupts are enabled.

Bits 31 to 26 of EIPC and bits 31 to 8 of EIPSW are reserved for future function expansion (these bits are always fixed to 0).

The value of EIPC is restored to the PC and the value of EIPSW to the PSW by the RETI instruction.

**(2) NMI status saving registers (FEPC and FEPSW)**

FEPC and FEPSW are used to save the status when a non-maskable interrupt (NMI) occurs.

If an NMI occurs, the contents of the program counter (PC) are saved to FEPC, and those of the program status word (PSW) are saved to FEPSW.

The address of the instruction next to the one of the instruction under execution, except some instructions, is saved to FEPC when an NMI occurs.

The current contents of the PSW are saved to FEPSW.

Because only one set of NMI status saving registers is available, the contents of these registers must be saved by program when multiple interrupts are enabled.

Bits 31 to 26 of FEPC and bits 31 to 8 of FEPSW are reserved for future function expansion (these bits are always fixed to 0).

The value of FEPC is restored to the PC and the value of FEPSW to the PSW by the RETI instruction.



**(3) Interrupt source register (ECR)**

The interrupt source register (ECR) holds the source of an exception or interrupt if an exception or interrupt occurs. This register holds the exception code of each interrupt source. Because this register is a read-only register, data cannot be written to this register using the LDSR instruction.



| Bit position | Bit name | Meaning |
|---|---|---|
| 31 to 16 | FECC | Exception code of non-maskable interrupt (NMI) |
| 15 to 0 | EICC | Exception code of exception or maskable interrupt |

**(4) Program status word (PSW)**

The program status word (PSW) is a collection of flags that indicate the status of the program (result of instruction execution) and the status of the CPU.

If the contents of a bit of this register are changed by using the LDSR instruction, the new contents are validated immediately after completion of LDSR instruction execution.  However if the ID flag is set to 1, interrupt requests will not be acknowledged while the LDSR instruction is being executed.

Bits 31 to 8 of this register are reserved for future function expansion (these bits are fixed to 0).

(1/2)

| Bit position | Flag name | Meaning |
|---|---|---|
| 31 to 8 | RFU | Reserved field.  Fixed to 0. |
| 7 | NP | Indicates that a non-maskable interrupt (NMI) is being serviced.  This bit is set to 1 when an NMI request is acknowledged, disabling multiple interrupts.<br>    0:  NMI is not being serviced.<br>    1:  NMI is being serviced. |
| 6 | EP | Indicates that an exception is being processed.  This bit is set to 1 when an exception occurs. Even if this bit is set, interrupt requests are acknowledged.<br>    0:  Exception is not being processed.<br>    1:  Exception is being processed. |
| 5 | ID | Indicates whether a maskable interrupt can be acknowledged.<br>    0:  Interrupt enabled<br>    1:  Interrupt disabled |
| 4 | SAT[Note] | Indicates that the result of a saturation operation has overflowed and is saturated.  Because this is a cumulative flag, it is set to 1 when the result of a saturation operation instruction is saturated, and is not cleared to 0 even if the subsequent operation result is not saturated. Use the LDSR instruction to clear this bit.  This flag is neither set to 1 nor cleared to 0 by execution of an arithmetic operation instruction.<br>    0:  Not saturated<br>    1:  Saturated |
| 3 | CY | Indicates whether a carry or a borrow occurs as a result of an operation.<br>    0:  Carry or borrow does not occur.<br>    1:  Carry or borrow occurs. |
| 2 | OV[Note] | Indicates whether an overflow occurs during operation.<br>    0:  Overflow does not occur.<br>    1:  Overflow occurs. |
| 1 | S[Note] | Indicates whether the result of an operation is negative.<br>    0:  The result is positive or 0.<br>    1:  The result is negative. |
| 0 | Z | Indicates whether the result of an operation is 0.<br>    0:  The result is not 0.<br>    1:  The result is 0. |

**Remark**   Also read **Note** on the next page.

(2/2)

**Note**  The result of the operation that has performed saturation processing is determined by the contents of the OV and S flags.  The SAT flag is set to 1 only when the OV flag is set to 1 when a saturation operation is performed.

| Status of Operation Result | Flag Status | | | Result of Operation of Saturation Processing |
|---|---|---|---|---|
| | SAT | OV | S | |
| Maximum positive value is exceeded | 1 | 1 | 0 | 7FFFFFFFH |
| Maximum negative value is exceeded | 1 | 1 | 1 | 80000000H |
| Positive (maximum value is not exceeded) | Holds value before operation | 0 | 0 | Operation result itself |
| Negative (maximum value is not exceeded) | | | 1 | |

**(5) CALLT execution status saving registers (CTPC and CTPSW)**

CTPC and CTPSW are CALLT execution status saving registers.

When the CALLT instruction is executed, the contents of the program counter (PC) are saved to CTPC, and those of the program status word (PSW) are saved to CTPSW.

The contents saved to CTPC are the address of the instruction next to CALLT.

The current contents of the PSW are saved to CTPSW.

Bits 31 to 26 of CTPC and bits 31 to 8 of CTPSW are reserved for future function expansion (fixed to 0).
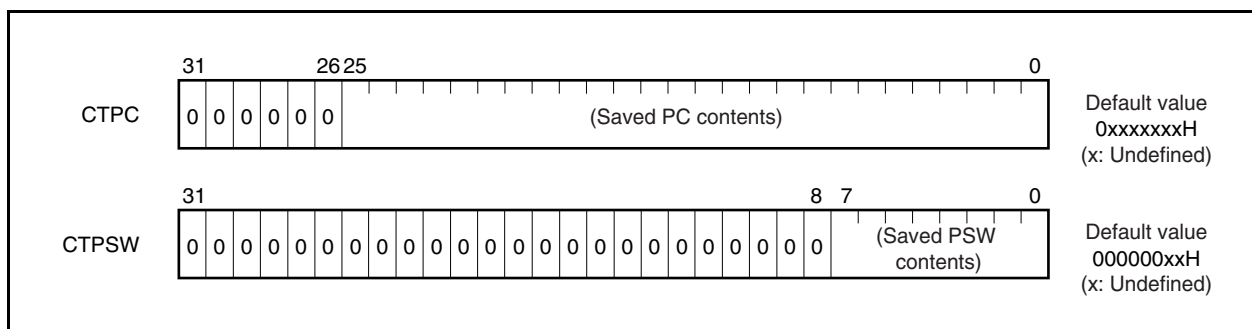
**(6) Exception/debug trap status saving registers (DBPC and DBPSW)**

DBPC and DBPSW are exception/debug trap status registers.

If an exception trap or debug trap occurs, the contents of the program counter (PC) are saved to DBPC, and those of the program status word (PSW) are saved to DBPSW.

The contents to be saved to DBPC are the address of the instruction next to the one that is being executed when an exception trap or debug trap occurs.

The current contents of the PSW are saved to DBPSW.

This register can be read or written only during the interval between the execution of the DBTRAP instruction or illegal opcode and the DBRET instruction.

Bits 31 to 26 of DBPC and bits 31 to 8 of DBPSW are reserved for future function expansion (fixed to 0).

The value of DBPC is restored to the PC and the value of DBPSW to the PSW by the DBRET instruction.
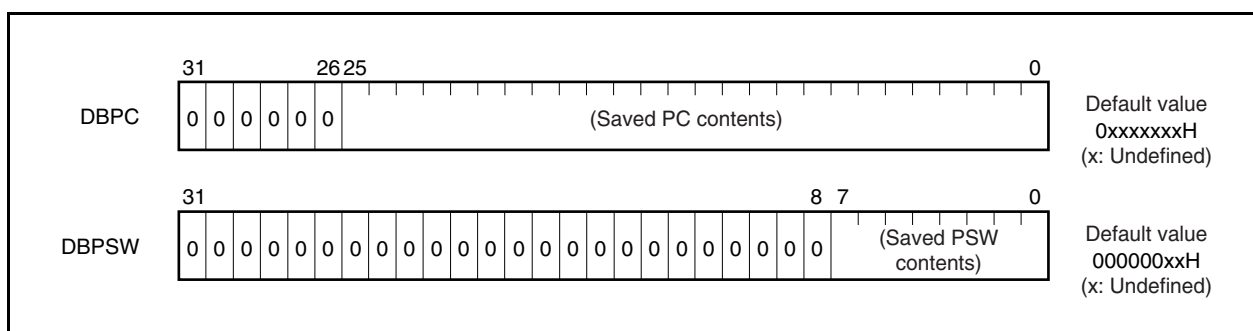


**(7) CALLT base pointer (CTBP)**

The CALLT base pointer (CTBP) is used to specify a table address or generate a target address (bit 0 is fixed to 0).

Bits 31 to 26 of this register are reserved for future function expansion (fixed to 0).
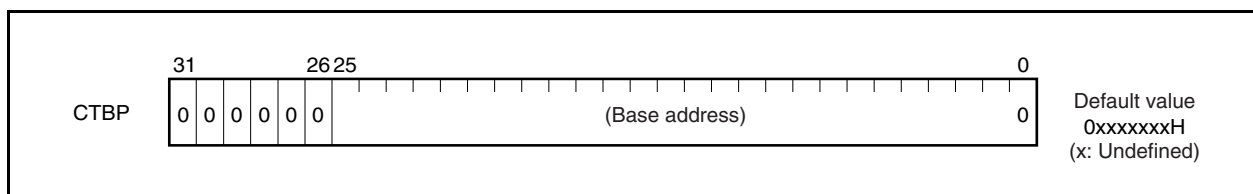
## 3.3  Operation Modes

The V850ES/JG3-U and V850ES/JH3-U have the following operation modes.

**(1)  Normal operation mode**

In this mode, each pin related to the bus interface is set to the port mode after system reset has been released. Execution branches to the reset entry address of the internal ROM, and then instruction processing is started.

**(2)  Flash memory programming mode**

In this mode, the internal flash memory can be programmed by using a flash programmer.

**(3)  On-chip debug mode**

The V850ES/JG3-U and V850ES/JH3-U are provided with an on-chip debug function that employs the JTAG (Joint Test Action Group) communication specifications.

For details, see **CHAPTER 32  ON-CHIP DEBUG FUNCTION**.

### 3.3.1  Specifying operation mode

Specify the operation mode by using the FLMD0 and FLMD1 pins.

In the normal mode, make sure that a low level is input to the FLMD0 pin when reset is released.

In the flash memory programming mode, a high level is input to the FLMD0 pin from the flash programmer if a flash programmer is connected, but it must be input from an external circuit in the self-programming mode.

| Operation When Reset Is Released | | Operation Mode After Reset |
|---|---|---|
| FLMD0 | FLMD1 | |
| L | × | Normal operation mode |
| H | L | Flash memory programming mode |
| H | H | Setting prohibited |

**Remark**  L:  Low-level input

H:  High-level input

×:  Don't care

## 3.4    Address Space

### 3.4.1    CPU address space

For instruction addressing, up to a combined total of 16 MB of external memory area and internal ROM area, plus an internal RAM area, are supported in a linear address space (program space) of up to 64 MB.  For operand addressing (data access), up to 4 GB of a linear address space (data space) is supported.  The 4 GB address space, however, is viewed as 64 images of a 64 MB physical address space.  This means that the same 64 MB physical address space is accessed regardless of the value of bits 31 to 26.

**Figure 3-1.  Image on Address Space**
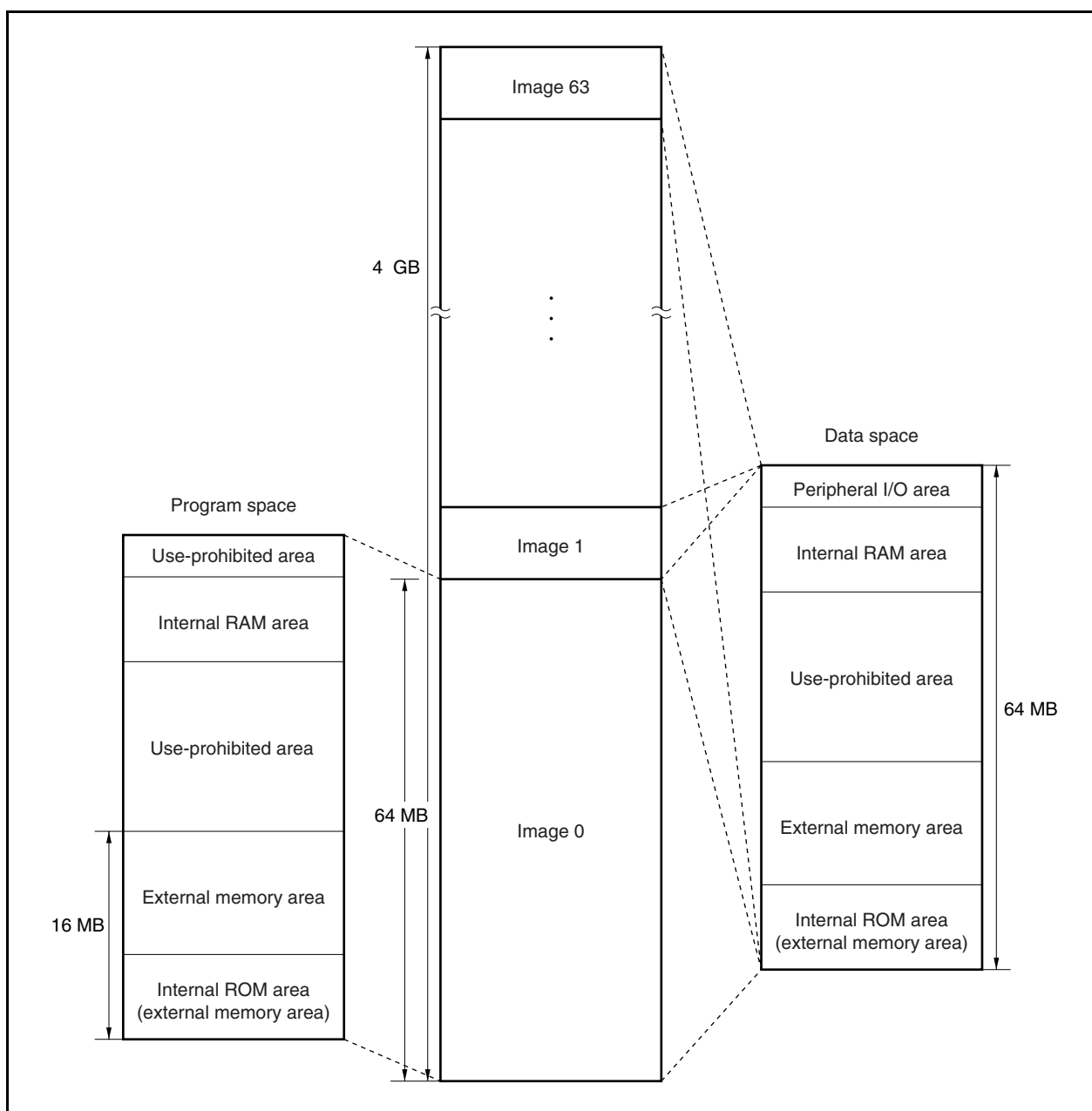
### 3.4.2   Wraparound of CPU address space

**(1)  Program space**

Of the 32 bits of the PC (program counter), the higher 6 bits are fixed to 0 and only the lower 26 bits are valid.  The higher 6 bits ignore a carry or borrow from bit 25 to 26 during branch address calculation.

Therefore, the highest address of the program space, 03FFFFFFH, and the lowest address, 00000000H, are contiguous addresses.  That the highest address and the lowest address of the program space are contiguous in this way is called wraparound.

**Caution   Because the 4 KB area of addresses 03FFF000H to 03FFFFFFH is an on-chip peripheral I/O area, instructions cannot be fetched from this area.  Therefore, do not execute an operation in which the result of a branch address calculation affects this area.**
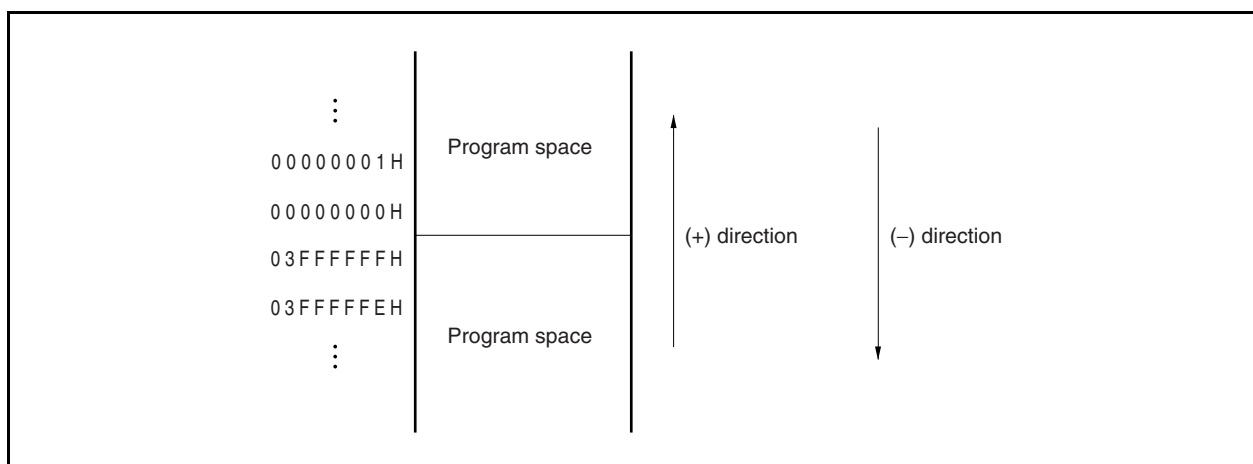


**(2)  Data space**

The result of an operand address calculation operation that exceeds 32 bits is ignored.

Therefore, the highest address of the data space, FFFFFFFFH, and the lowest address, 00000000H, are contiguous, and wraparound occurs at the boundary of these addresses.
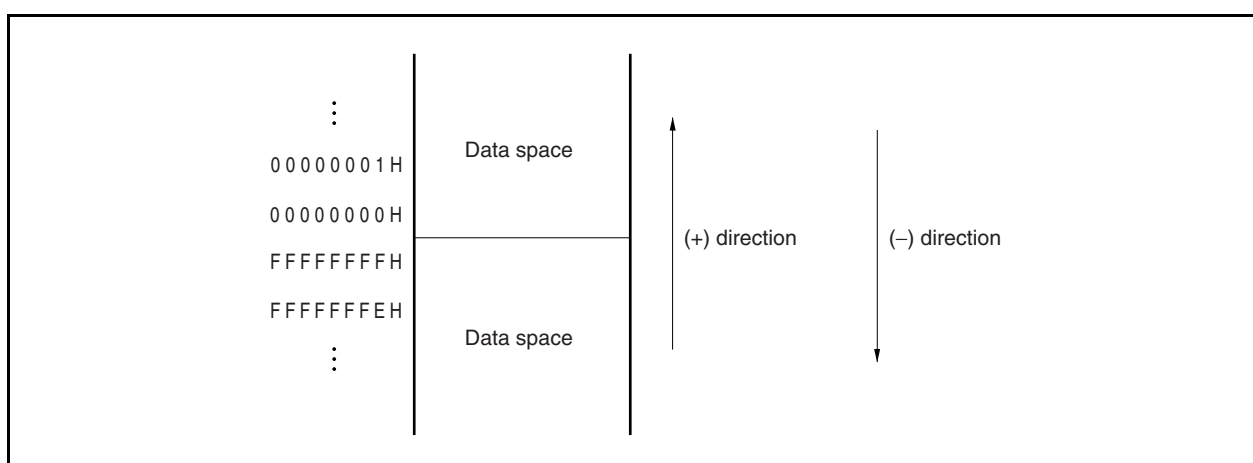
### 3.4.3 Memory map

The areas shown below are reserved in the V850ES/JG3-U and V850ES/JH3-U.

**Figure 3-2. Data Memory Map (Physical Addresses)**



Notes 1. $\overline{\text{CS1}}$ is not provided as an external signal of the V850ES/Jx3-U; it is used internally as a chip select signal for the USB.

2. Use of addresses 03FEF000H to 03FEFFFFH is prohibited because they overlap an on-chip peripheral I/O area.

3. This area is used as an external memory area when data write access to this area is executed.
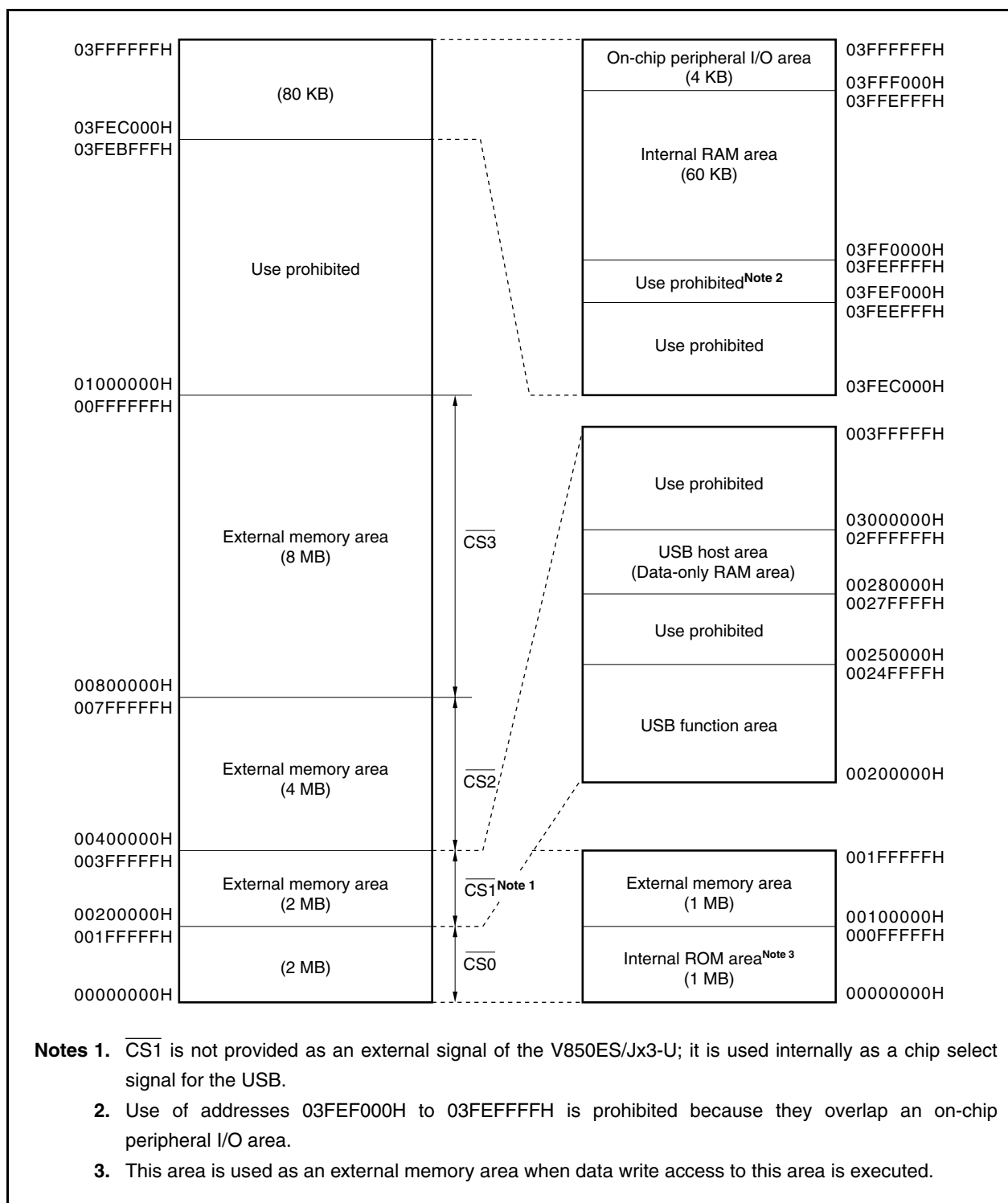
**Figure 3-3.  Program Memory Map**

```
                     03FFFFFFH ┌──────────────────────┐
                               │     Use prohibited    │
                               │(program fetch prohibited area)│
                     03FFF000H │                      │
                     03FFEFFFH ├──────────────────────┤
                               │                      │
                               │                      │
                               │  Internal RAM area (60 KB) │
                               │                      │
                               │                      │
                     03FF0000H │                      │
                     03FEFFFFH ├──────────────────────┤
                               │     Use prohibited    │
                               │(program fetch prohibited area)│
                     01000000H │                      │
                     00FFFFFFH ├──────────────────────┤
                               │                      │
                               │                      │
                               │                      │
                               │                      │
                               │   External memory area │
                               │        (14 MB)        │
                               │                      │
                               │                      │
                               │                      │
                               │                      │
                     00200000H │                      │
                     001FFFFFH ├──────────────────────┤
                               │   External memory area │
                     00100000H │        (1 MB)        │
                     000FFFFFH ├──────────────────────┤
                               │   Internal ROM area   │
                     00000000H │        (1 MB)        │
                               └──────────────────────┘
```
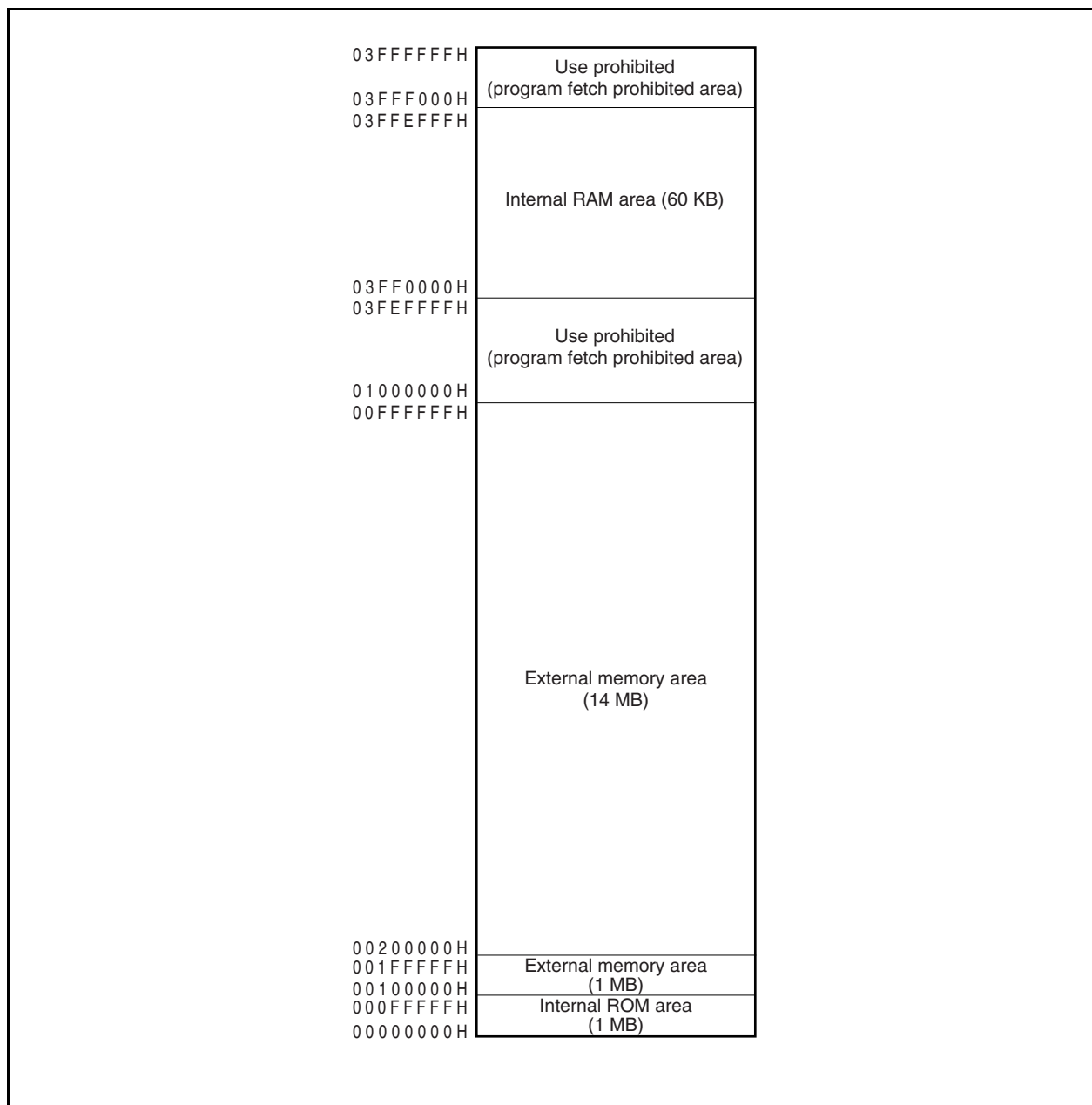
**3.4.4   Areas**

**(1) Internal ROM area**

Up to 1 MB is reserved as an internal ROM area.

**(a) Internal ROM (384 KB)**

384 KB are allocated to addresses 00000000H to 0005FFFFH in the following products.

Accessing addresses 00060000H to 000FFFFFH is prohibited.

- μPD70F3763, 70F3768

**Figure 3-4.  Internal ROM Area (384 KB)**
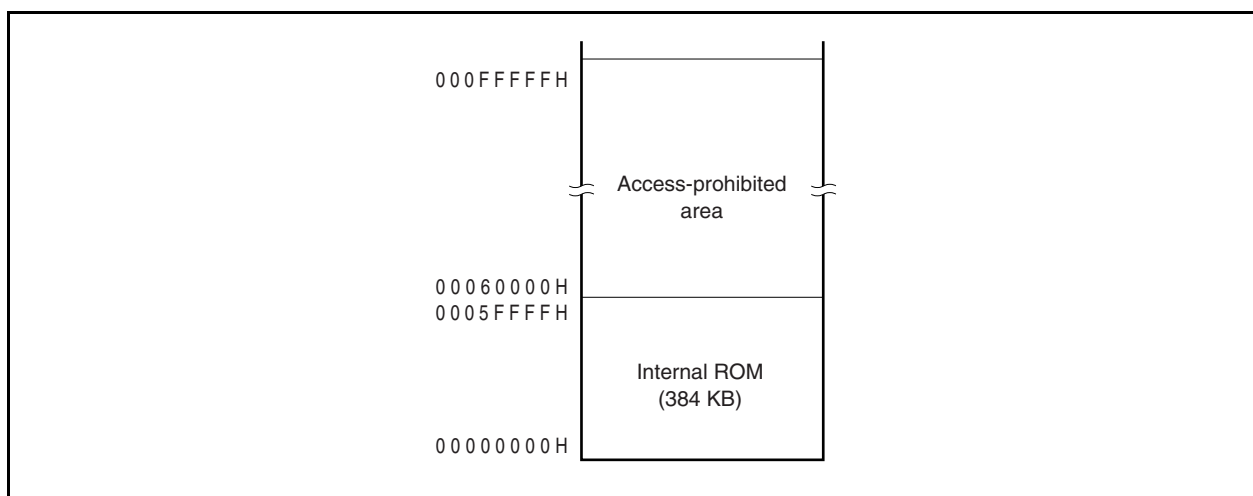


**(b) Internal ROM (512 KB)**

512 KB are allocated to addresses 00000000H to 0007FFFFH in the following products.

Accessing addresses 00080000H to 000FFFFFH is prohibited.

- μPD70F3764, 70F3769

**Figure 3-5.  Internal ROM Area (512 KB)**
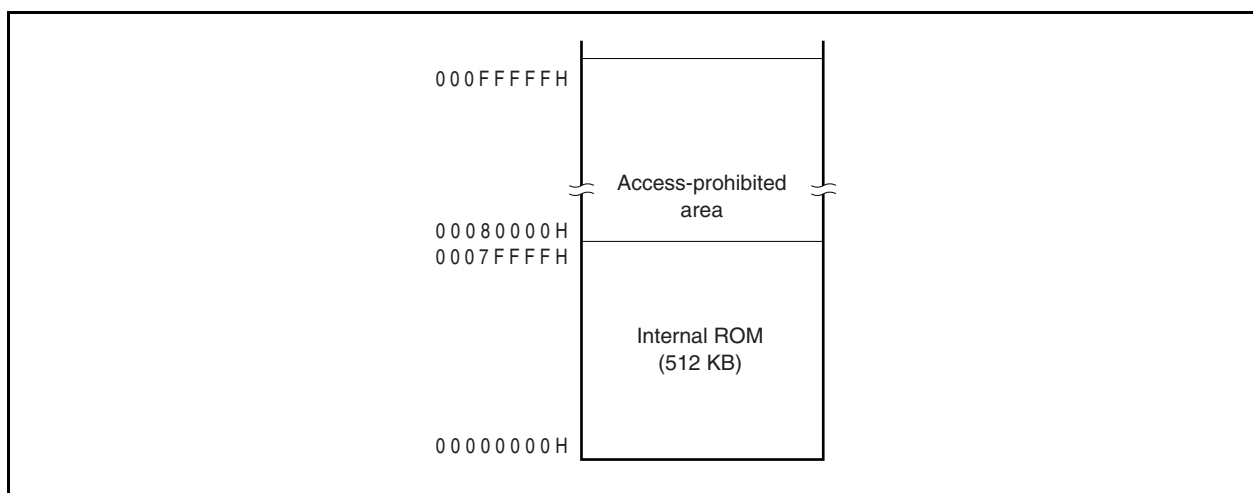
**(2) Internal RAM area**

Up to 60 KB are reserved as the internal RAM area.

The V850ES/JG3-U and V850ES/JH3-U include a data-only RAM of 8 KB in addition to the internal RAM.  If the USB host or USB function is not used, the data-only RAM can be used as the internal RAM.

The RAM capacity of V850ES/JG3-U and V850ES/JH3-U is as follows.

**Table 3-3.  RAM Area**

| Generic Name | Product Name | Internal RAM | Data-only RAM | Total RAM |
|---|---|---|---|---|
| V850ES/JG3-U | $\mu$PD70F3763 | 40 KB | 8 KB | 48 KB |
| | $\mu$PD70F3764 | 48 KB | | 56 KB |
| V850ES/JH3-U | $\mu$PD70F3768 | 40 KB | | 48 KB |
| | $\mu$PD70F3769 | 48 KB | | 56KB |

**(a) Internal RAM (40 KB)**

40 KB are allocated to addresses 03FF5000H to 03FFEFFFH in the following products.

Accessing addresses 03FF0000H to 03FF4FFFH is prohibited.

- $\mu$PD70F3763, 70F3768

**Figure 3-6.  Internal RAM Area (40 KB)**
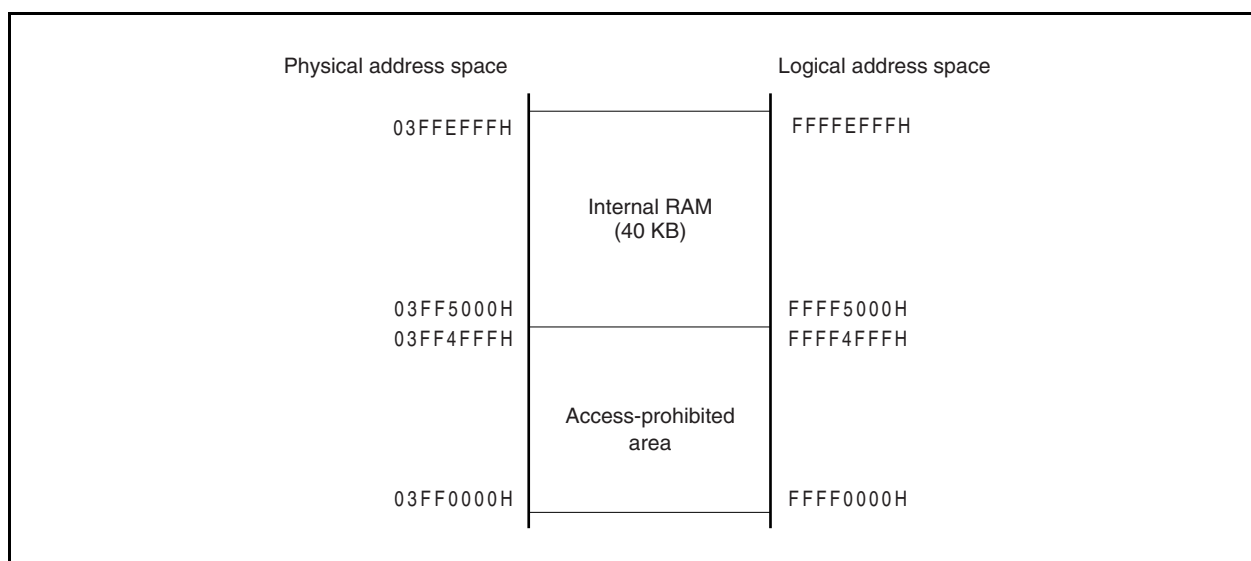
**(b) Internal RAM (48 KB)**

48 KB are allocated to addresses 03FF3000H to 03FFEFFFH in the following products.

Accessing addresses 03FF0000H to 03FF2FFFH is prohibited.

- μPD70F3764, 70F3769

**Figure 3-7.  Internal RAM Area (48 KB)**
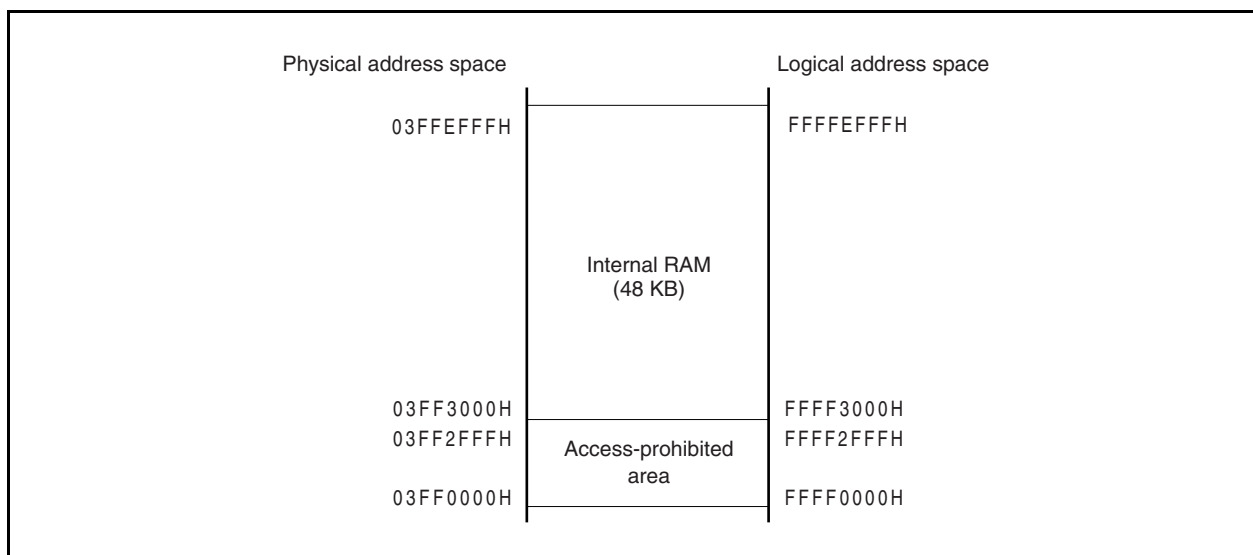


**(c) Data-only RAM (8 KB)**

A data-only RAM of 8 KB is allocated to addresses 00280000H to 00281FFFH in the V850ES/JG3-U and V850ES/JH3-U.

The data-only RAM is used as a USB host area when the USB host controller is used.  If the USB host controller is not used, it can be used as a data-only RAM.

**Figure 3-8.  Data-Only RAM Area (8 KB)**



**Caution   When using the data-only RAM area, the following two register settings are needed.**
- **Select the clock that can be supplied to the data-only RAM by using the USB clock select register (UCKSEL) (refer to 20.6.1(1)).**
- **Enable the data-only RAM operation by using the USB function select register (UHCKMSK) (UHCKMSK.UHMSK bit = 0) (refer to 20.6.1(3)).**
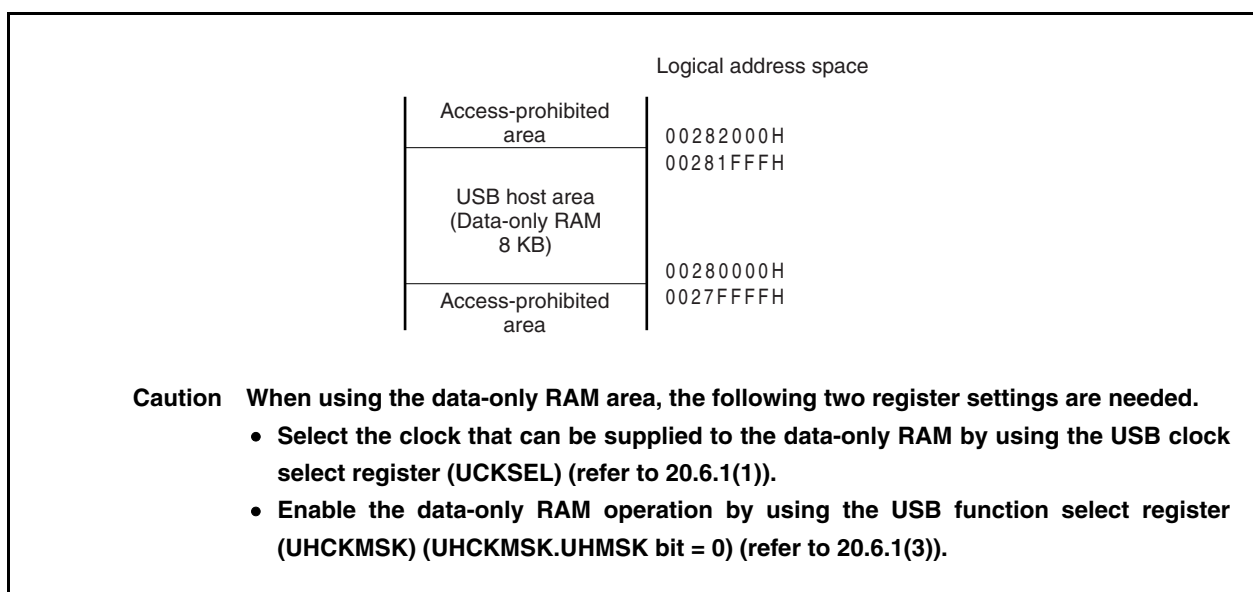
**(3) On-chip peripheral I/O area**

    4 KB of addresses 03FFF000H to 03FFFFFFH are reserved as the on-chip peripheral I/O area.

**Figure 3-9.  On-Chip Peripheral I/O Area**
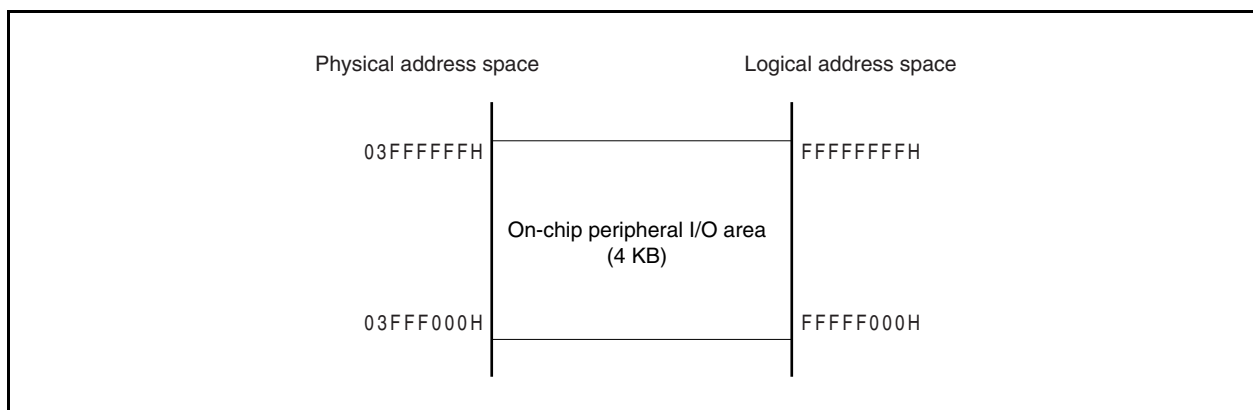


Peripheral I/O registers that have functions to specify the operation mode for and monitor the status of the on-chip peripheral I/O are mapped to the on-chip peripheral I/O area.  Program cannot be fetched from this area.

**Cautions 1. When a register is accessed in word units, a word area is accessed twice in halfword units in the order of lower area and higher area, with the lower 2 bits of the address ignored.**

           **2. If a register that can be accessed in byte units is accessed in halfword units, the higher 8 bits are undefined when the register is read, and data is written to the lower 8 bits.**

           **3. Addresses not defined as registers are reserved for future expansion.  The operation is undefined and not guaranteed when these addresses are accessed.**

           **4. The internal ROM/RAM area and on-chip peripheral I/O area are assigned to successive addresses.**

               **When accessing the internal ROM/RAM area by incrementing or decrementing addresses using a pointer operation or such, be careful not to access the on-chip peripheral I/O area by mistakenly extending over the internal ROM/RAM area boundary.**

**(4) External memory area**

    13 MB (00100000H to 001FFFFFH, 00400000H to 00FFFFFFH) are allocated as the external memory area.  For details, see **CHAPTER 5  BUS CONTROL FUNCTION**.

### 3.4.5   Recommended use of address space

The architecture of the V850ES/JG3-U and V850ES/JH3-U requires that a register that serves as a pointer be secured for address generation when operand data in the data space is accessed.  The address stored in this pointer ±32 KB can be directly accessed by an instruction for operand data.  Because the number of general-purpose registers that can be used as a pointer is limited, however, by keeping the performance from dropping during address calculation when a pointer value is changed, as many general-purpose registers as possible can be secured for variables, and the program size can be reduced.

### (1)  Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are fixed to 0, and only the lower 26 bits are valid.  Regarding the program space, therefore, a 64 MB space of contiguous addresses starting from 00000000H unconditionally corresponds to the memory map.

To use the internal RAM area as the program space, access the following addresses.

**Caution   If a branch instruction is at the upper limit of the internal RAM area, a prefetch operation (invalid fetch) straddling the on-chip peripheral I/O area does not occur.**

| RAM Size | Access Address |
|----------|----------------|
| 48 KB    | 03FF3000H to 03FFEFFFH |
| 40 KB    | 03FF5000H to 03FFEFFFH |

**(2) Data space**

With the V850ES/JG3-U and V850ES/JH3-U, it seems that there are sixty-four 64 MB address spaces on the 4 GB CPU address space. Therefore, the least significant bit (bit 25) of a 26-bit address is sign-extended to 32 bits and allocated as an address.

**(a) Application example of wraparound**

If R = r0 (zero register) is specified for the LD/ST disp16 [R] instruction, a range of addresses 00000000H ±32 KB can be addressed by sign-extended disp16. All the resources, including the internal hardware, can be addressed by one pointer.

The zero register (r0) is a register fixed to 0 by hardware, and practically eliminates the need for registers dedicated to pointers.

**Example**: $\mu$PD70F3769

| | | |
|---|---|---|
| 0007FFFFH | | |
| 00007FFFH | | |
| | Internal ROM area | 32 KB |
| (R = ) 00000000H | On-chip peripheral I/O area | 4 KB |
| FFFFF000H | | |
| FFFFEFFFH | | |
| | Internal RAM area | 28 KB |
| FFFF8000H | | |
| FFFF3000H | | |
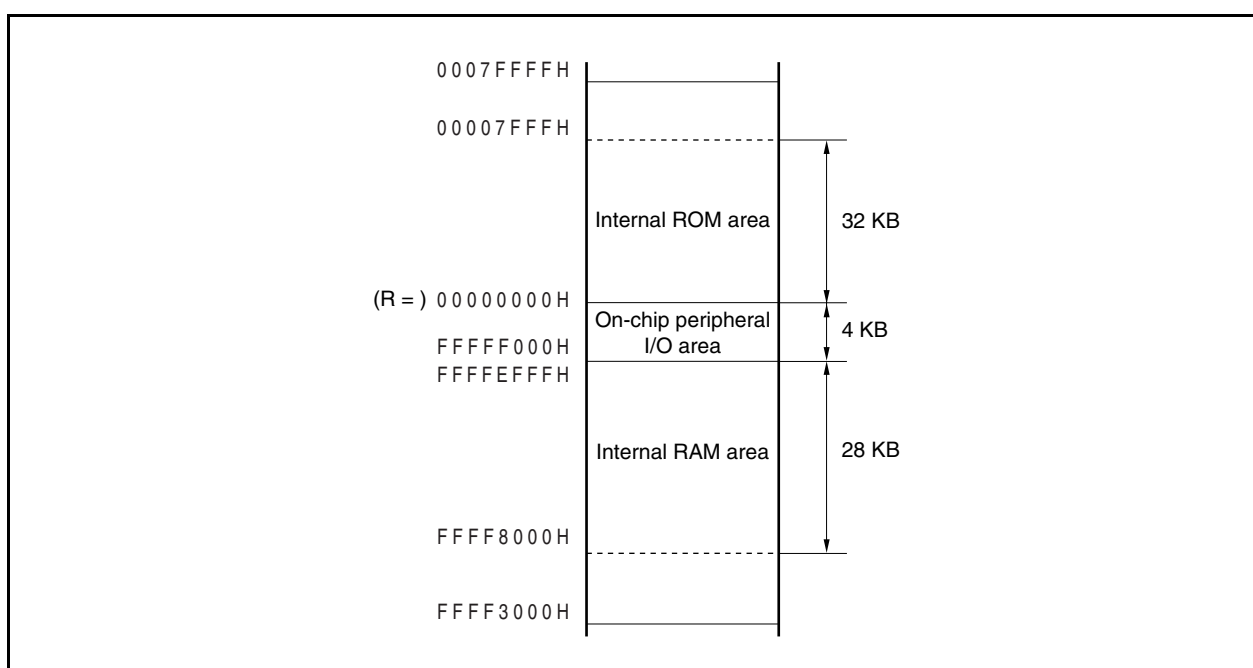
**Figure 3-10. Recommended Memory Map**



**Remarks 1.** ↕ indicates the recommended area.

      **2.** This figure is the recommended memory map of the μPD70F3769.
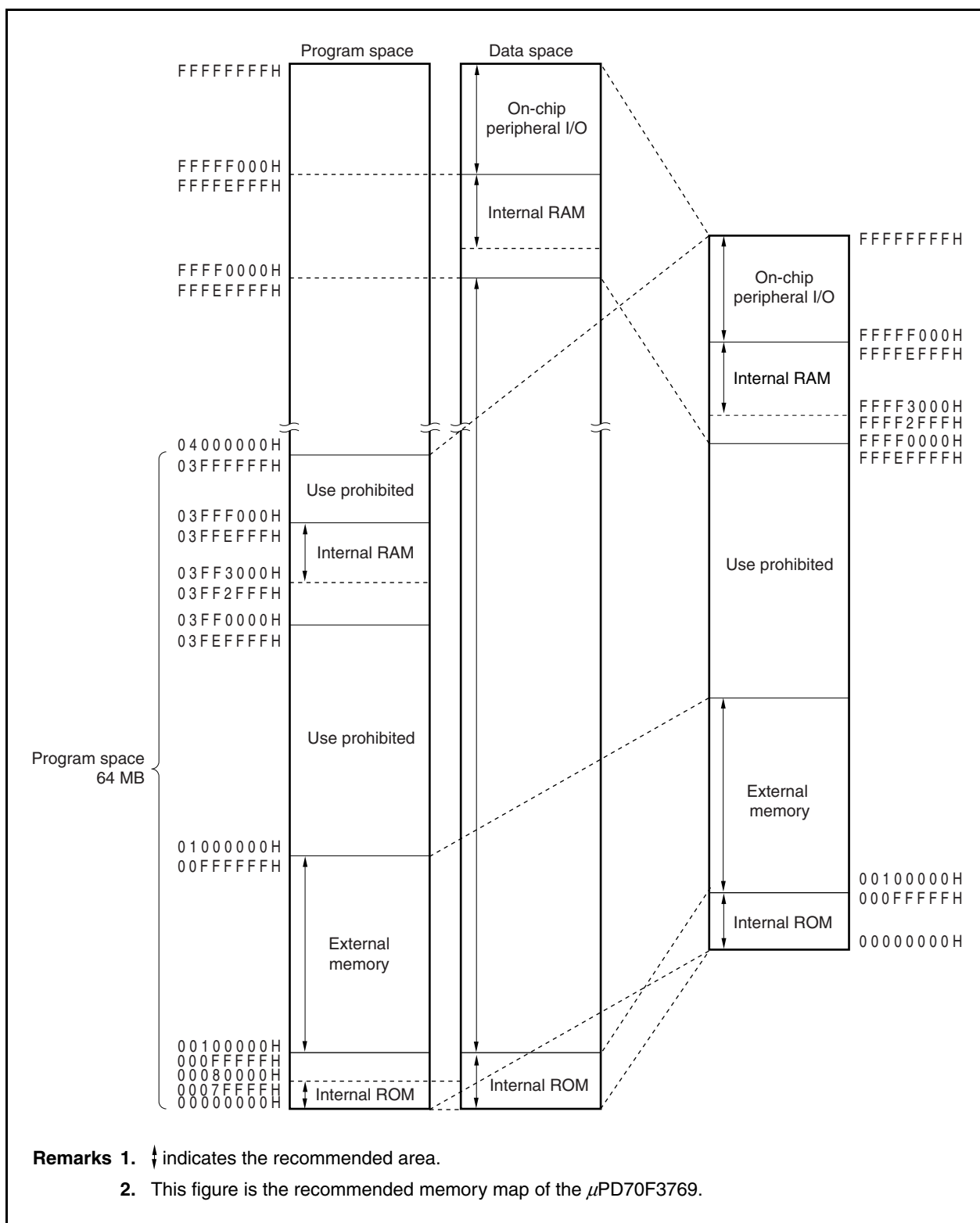
### 3.4.6  Peripheral I/O registers

(1/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|---|---------------|
| | | | | 1 | 8 | 16 | |
| FFFFF004H | Port DL register | PDL | R/W | | | √ | 0000H[Note 1] |
| FFFFF004H | Port DL register L | PDLL | | √ | √ | | 00H[Note 1] |
| FFFFF005H | Port DL register H | PDLH | | √ | √ | | 00H[Note 1] |
| FFFFF006H | Port DH register[Note 2] | PDH[Note 2] | | √ | √ | | 00H[Note 1] |
| FFFFF008H | Port CS register[Note 2] | PCS[Note 2] | | √ | √ | | 00H[Note 1] |
| FFFFF00AH | Port CT register | PCT | | √ | √ | | 00H[Note 1] |
| FFFFF00CH | Port CM register | PCM | | √ | √ | | 00H[Note 1] |
| FFFFF024H | Port DL mode register | PMDL | | | | √ | FFFFH |
| FFFFF024H | Port DL mode register L | PMDLL | | √ | √ | | FFH |
| FFFFF025H | Port DL mode register H | PMDLH | | √ | √ | | FFH |
| FFFFF026H | Port DH mode register[Note 2] | PMDH[Note 2] | | √ | √ | | FFH |
| FFFFF028H | Port CS mode register[Note 2] | PMCS[Note 2] | | √ | √ | | FFH |
| FFFFF02AH | Port CT mode register | PMCT | | √ | √ | | FFH |
| FFFFF02CH | Port CM mode register | PMCM | | √ | √ | | FFH |
| FFFFF044H | Port DL mode control register | PMCDL | | | | √ | 0000H |
| FFFFF044H | Port DL mode control register L | PMCDLL | | √ | √ | | 00H |
| FFFFF045H | Port DL mode control register H | PMCDLH | | √ | √ | | 00H |
| FFFFF046H | Port DH mode control register[Note 2] | PMCDH[Note 2] | | √ | √ | | 00H |
| FFFFF048H | Port CS mode control register[Note 2] | PMCCS[Note 2] | | √ | √ | | 00H |
| FFFFF04AH | Port CT mode control register | PMCCT | | √ | √ | | 00H |
| FFFFF04CH | Port CM mode control register | PMCCM | | √ | √ | | 00H |
| FFFFF066H | Bus size configuration register | BSC | | | | √ | 5555H |
| FFFFF06EH | System wait control register | VSWC | | | √ | | 77H |
| FFFFF080H | DMA source address register 0L | DSA0L | | | | √ | Undefined |
| FFFFF082H | DMA source address register 0H | DSA0H | | | | √ | Undefined |
| FFFFF084H | DMA destination address register 0L | DDA0L | | | | √ | Undefined |
| FFFFF086H | DMA destination address register 0H | DDA0H | | | | √ | Undefined |
| FFFFF088H | DMA source address register 1L | DSA1L | | | | √ | Undefined |
| FFFFF08AH | DMA source address register 1H | DSA1H | | | | √ | Undefined |
| FFFFF08CH | DMA destination address register 1L | DDA1L | | | | √ | Undefined |
| FFFFF08EH | DMA destination address register 1H | DDA1H | | | | √ | Undefined |
| FFFFF090H | DMA source address register 2L | DSA2L | | | | √ | Undefined |
| FFFFF092H | DMA source address register 2H | DSA2H | | | | √ | Undefined |
| FFFFF094H | DMA destination address register 2L | DDA2L | | | | √ | Undefined |
| FFFFF096H | DMA destination address register 2H | DDA2H | | | | √ | Undefined |
| FFFFF098H | DMA source address register 3L | DSA3L | | | | √ | Undefined |
| FFFFF09AH | DMA source address register 3H | DSA3H | | | | √ | Undefined |
| FFFFF09CH | DMA destination address register 3L | DDA3L | | | | √ | Undefined |
| FFFFF09EH | DMA destination address register 3H | DDA3H | | | | √ | Undefined |

**Notes 1.** The output latch is 00H or 0000H.  When these registers are in the input mode, the pin statuses are read.

   **2.** V850ES/JH3-U only

(2/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits 1 | 8 | 16 | Default Value |
|---------|----------------------|--------|-----|---|---|----|---------------|
| FFFFF0C0H | DMA transfer count register 0 | DBC0 | R/W | | | √ | Undefined |
| FFFFF0C2H | DMA transfer count register 1 | DBC1 | | | | √ | Undefined |
| FFFFF0C4H | DMA transfer count register 2 | DBC2 | | | | √ | Undefined |
| FFFFF0C6H | DMA transfer count register 3 | DBC3 | | | | √ | Undefined |
| FFFFF0D0H | DMA addressing control register 0 | DADC0 | | | | √ | 0000H |
| FFFFF0D2H | DMA addressing control register 1 | DADC1 | | | | √ | 0000H |
| FFFFF0D4H | DMA addressing control register 2 | DADC2 | | | | √ | 0000H |
| FFFFF0D6H | DMA addressing control register 3 | DADC3 | | | | √ | 0000H |
| FFFFF0E0H | DMA channel control register 0 | DCHC0 | | √ | √ | | 00H |
| FFFFF0E2H | DMA channel control register 1 | DCHC1 | | √ | √ | | 00H |
| FFFFF0E4H | DMA channel control register 2 | DCHC2 | | √ | √ | | 00H |
| FFFFF0E6H | DMA channel control register 3 | DCHC3 | | √ | √ | | 00H |
| FFFFF100H | Interrupt mask register 0 | IMR0 | | | | √ | FFFFH |
| FFFFF100H | Interrupt mask register 0L | IMR0L | | √ | √ | | FFH |
| FFFFF101H | Interrupt mask register 0H | IMR0H | | √ | √ | | FFH |
| FFFFF102H | Interrupt mask register 1 | IMR1 | | | | √ | FFFFH |
| FFFFF102H | Interrupt mask register 1L | IMR1L | | √ | √ | | FFH |
| FFFFF103H | Interrupt mask register 1H | IMR1H | | √ | √ | | FFH |
| FFFFF104H | Interrupt mask register 2 | IMR2 | | | | √ | FFFFH |
| FFFFF104H | Interrupt mask register 2L | IMR2L | | √ | √ | | FFH |
| FFFFF105H | Interrupt mask register 2H | IMR2H | | √ | √ | | FFH |
| FFFFF106H | Interrupt mask register 3 | IMR3 | | | | √ | FFFFH |
| FFFFF106H | Interrupt mask register 3L | IMR3L | | √ | √ | | FFH |
| FFFFF107H | Interrupt mask register 3H | IMR3H | | √ | √ | | FFH |
| FFFFF108H | Interrupt mask register 4 | IMR4 | | | | √ | FFFFH |
| FFFFF108H | Interrupt mask register 4L | IMR4L | | √ | √ | | FFH |
| FFFFF109H | Interrupt mask register 4H | IMR4H | | √ | √ | | FFH |
| FFFFF10AH | Interrupt mask register 5 | IMR5 | | | | √ | FFFFH |
| FFFFF10AH | Interrupt mask register 5L | IMR5L | | √ | √ | | FFH |
| FFFFF10BH | Interrupt mask register 5H | IMR5H | | √ | √ | | FFH |
| FFFFF110H | Interrupt control register | LVIIC | | √ | √ | | 47H |
| FFFFF112H | Interrupt control register | PIC00 | | √ | √ | | 47H |
| FFFFF114H | Interrupt control register | PIC01 | | √ | √ | | 47H |
| FFFFF116H | Interrupt control register | PIC02 | | √ | √ | | 47H |
| FFFFF118H | Interrupt control register | PIC03[Note] | | √ | √ | | 47H |
| FFFFF11AH | Interrupt control register | PIC04[Note] | | √ | √ | | 47H |
| FFFFF11CH | Interrupt control register | PIC05 | | √ | √ | | 47H |
| FFFFF11EH | Interrupt control register | PIC06 | | √ | √ | | 47H |
| FFFFF120H | Interrupt control register | PIC07 | | √ | √ | | 47H |
| FFFFF122H | Interrupt control register | PIC08 | | √ | √ | | 47H |

**Note**  V850ES/JH3-U only

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| FFFFF124H | Interrupt control register | PIC09 | R/W | √ | √ | | 47H |
| FFFFF126H | Interrupt control register | PIC10 | | √ | √ | | 47H |
| FFFFF128H | Interrupt control register | PIC11 | | √ | √ | | 47H |
| FFFFF12AH | Interrupt control register | PIC12 | | √ | √ | | 47H |
| FFFFF12CH | Interrupt control register | PIC13 | | √ | √ | | 47H |
| FFFFF12EH | Interrupt control register | PIC14 | | √ | √ | | 47H |
| FFFFF130H | Interrupt control register | PIC15 | | √ | √ | | 47H |
| FFFFF132H | Interrupt control register | PIC16 | | √ | √ | | 47H |
| FFFFF134H | Interrupt control register | PIC17 | | √ | √ | | 47H |
| FFFFF136H | Interrupt control register | PIC18 | | √ | √ | | 47H |
| FFFFF138H | Interrupt control register | TAB0OVIC | | √ | √ | | 47H |
| FFFFF13AH | Interrupt control register | TAB0CCIC0 | | √ | √ | | 47H |
| FFFFF13CH | Interrupt control register | TAB0CCIC1 | | √ | √ | | 47H |
| FFFFF13EH | Interrupt control register | TAB0CCIC2 | | √ | √ | | 47H |
| FFFFF140H | Interrupt control register | TAB0CCIC3 | | √ | √ | | 47H |
| FFFFF142H | Interrupt control register | TAB1OVIC | | √ | √ | | 47H |
| FFFFF144H | Interrupt control register | TAB1CCIC0 | | √ | √ | | 47H |
| FFFFF146H | Interrupt control register | TAB1CCIC1 | | √ | √ | | 47H |
| FFFFF148H | Interrupt control register | TAB1CCIC2 | | √ | √ | | 47H |
| FFFFF14AH | Interrupt control register | TAB1CCIC3 | | √ | √ | | 47H |
| FFFFF14CH | Interrupt control register | TT0OVIC | | √ | √ | | 47H |
| FFFFF14EH | Interrupt control register | TT0CCIC0 | | √ | √ | | 47H |
| FFFFF150H | Interrupt control register | TT0CCIC1 | | √ | √ | | 47H |
| FFFFF152H | Interrupt control register | TT0IECIC | | √ | √ | | 47H |
| FFFFF154H | Interrupt control register | TAA0OVIC | | √ | √ | | 47H |
| FFFFF156H | Interrupt control register | TAA0CCIC0 | | √ | √ | | 47H |
| FFFFF158H | Interrupt control register | TAA0CCIC1 | | √ | √ | | 47H |
| FFFFF15AH | Interrupt control register | TAA1OVIC | | √ | √ | | 47H |
| FFFFF15CH | Interrupt control register | TAA1CCIC0 | | √ | √ | | 47H |
| FFFFF15EH | Interrupt control register | TAA1CCIC1 | | √ | √ | | 47H |
| FFFFF160H | Interrupt control register | TAA2OVIC | | √ | √ | | 47H |
| FFFFF162H | Interrupt control register | TAA2CCIC0 | | √ | √ | | 47H |
| FFFFF164H | Interrupt control register | TAA2CCIC1 | | √ | √ | | 47H |
| FFFFF166H | Interrupt control register | TAA3OVIC | | √ | √ | | 47H |
| FFFFF168H | Interrupt control register | TAA3CCIC0 | | √ | √ | | 47H |
| FFFFF16AH | Interrupt control register | TAA3CCIC1 | | √ | √ | | 47H |
| FFFFF16CH | Interrupt control register | TAA4OVIC | | √ | √ | | 47H |
| FFFFF16EH | Interrupt control register | TAA4CCIC0 | | √ | √ | | 47H |
| FFFFF170H | Interrupt control register | TAA4CCIC1 | | √ | √ | | 47H |
| FFFFF172H | Interrupt control register | TAA5OVIC | | √ | √ | | 47H |
| FFFFF174H | Interrupt control register | TAA5CCIC0 | | √ | √ | | 47H |

(4/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| FFFFF176H | Interrupt control register | TAA5CCIC1 | R/W | √ | √ | | 47H |
| FFFFF178H | Interrupt control register | TM0EQIC0 | | √ | √ | | 47H |
| FFFFF17AH | Interrupt control register | TM1EQIC0 | | √ | √ | | 47H |
| FFFFF17CH | Interrupt control register | TM2EQIC0 | | √ | √ | | 47H |
| FFFFF17EH | Interrupt control register | TM3EQIC0 | | √ | √ | | 47H |
| FFFFF180H | Interrupt control register | CF0RIC/IICIC1 | | √ | √ | | 47H |
| FFFFF182H | Interrupt control register | CF0TIC | | √ | √ | | 47H |
| FFFFF184H | Interrupt control register | CF1RIC | | √ | √ | | 47H |
| FFFFF186H | Interrupt control register | CF1TIC | | √ | √ | | 47H |
| FFFFF188H | Interrupt control register | CF2RIC | | √ | √ | | 47H |
| FFFFF18AH | Interrupt control register | CF2TIC | | √ | √ | | 47H |
| FFFFF18CH | Interrupt control register | CF3RIC | | √ | √ | | 47H |
| FFFFF18EH | Interrupt control register | CF3TIC | | √ | √ | | 47H |
| FFFFF190H | Interrupt control register | CF4RIC | | √ | √ | | 47H |
| FFFFF192H | Interrupt control register | CF4TIC | | √ | √ | | 47H |
| FFFFF194H | Interrupt control register | UC0RIC | | √ | √ | | 47H |
| FFFFF196H | Interrupt control register | UC0TIC | | √ | √ | | 47H |
| FFFFF198H | Interrupt control register | UC1RIC/IICIC2 | | √ | √ | | 47H |
| FFFFF19AH | Interrupt control register | UC1TIC | | √ | √ | | 47H |
| FFFFF19CH | Interrupt control register | UC2RIC | | √ | √ | | 47H |
| FFFFF19EH | Interrupt control register | UC2TIC | | √ | √ | | 47H |
| FFFFF1A0H | Interrupt control register | UC3RIC/IICIC0 | | √ | √ | | 47H |
| FFFFF1A2H | Interrupt control register | UC3TIC | | √ | √ | | 47H |
| FFFFF1A4H | Interrupt control register | UC4RIC | | √ | √ | | 47H |
| FFFFF1A6H | Interrupt control register | UC4TIC | | √ | √ | | 47H |
| FFFFF1A8H | Interrupt control register | ADIC | | √ | √ | | 47H |
| FFFFF1AAH | Interrupt control register | DMAIC0 | | √ | √ | | 47H |
| FFFFF1ACH | Interrupt control register | DMAIC1 | | √ | √ | | 47H |
| FFFFF1AEH | Interrupt control register | DMAIC2 | | √ | √ | | 47H |
| FFFFF1B0H | Interrupt control register | DMAIC3 | | √ | √ | | 47H |
| FFFFF1B2H | Interrupt control register | KRIC | | √ | √ | | 47H |
| FFFFF1B4H | Interrupt control register | RTC0IC | | √ | √ | | 47H |
| FFFFF1B6H | Interrupt control register | RTC1IC | | √ | √ | | 47H |
| FFFFF1B8H | Interrupt control register | RTC2IC | | √ | √ | | 47H |
| FFFFF1C2H | Interrupt control register | UHIC0 | | √ | √ | | 47H |
| FFFFF1C4H | Interrupt control register | UHIC1 | | √ | √ | | 47H |
| FFFFF1C6H | Interrupt control register | UHIC2 | | √ | √ | | 47H |
| FFFFF1C8H | Interrupt control register | UFIC0 | | √ | √ | | 47H |
| FFFFF1CAH | Interrupt control register | UFIC1 | | √ | √ | | 47H |
| FFFFF1FAH | In-service priority register | ISPR | R | √ | √ | | 00H |
| FFFFF1FCH | Command register | PRCMD | W | | √ | | Undefined |
| FFFFF1FEH | Power save control register | PSC | R/W | √ | √ | | 00H |

RENESAS

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|:---:|:---:|:---:|---------------|
| | | | | 1 | 8 | 16 | |
| FFFFF200H | A/D converter mode register 0 | ADA0M0 | R/W | √ | √ | | 00H |
| FFFFF201H | A/D converter mode register 1 | ADA0M1 | | √ | √ | | 00H |
| FFFFF202H | A/D converter channel specification register | ADA0S | | √ | √ | | 00H |
| FFFFF203H | A/D converter mode register 2 | ADA0M2 | | √ | √ | | 00H |
| FFFFF204H | Power-fail compare mode register | ADA0PFM | | √ | √ | | 00H |
| FFFFF205H | Power-fail compare threshold value register | ADA0PFT | | √ | √ | | 00H |
| FFFFF210H | A/D conversion result register 0 | ADA0CR0 | R | | | √ | Undefined |
| FFFFF211H | A/D conversion result register 0H | ADA0CR0H | | | √ | | Undefined |
| FFFFF212H | A/D conversion result register 1 | ADA0CR1 | | | | √ | Undefined |
| FFFFF213H | A/D conversion result register 1H | ADA0CR1H | | | √ | | Undefined |
| FFFFF214H | A/D conversion result register 2 | ADA0CR2 | | | | √ | Undefined |
| FFFFF215H | A/D conversion result register 2H | ADA0CR2H | | | √ | | Undefined |
| FFFFF216H | A/D conversion result register 3 | ADA0CR3 | | | | √ | Undefined |
| FFFFF217H | A/D conversion result register 3H | ADA0CR3H | | | √ | | Undefined |
| FFFFF218H | A/D conversion result register 4 | ADA0CR4 | | | | √ | Undefined |
| FFFFF219H | A/D conversion result register 4H | ADA0CR4H | | | √ | | Undefined |
| FFFFF21AH | A/D conversion result register 5 | ADA0CR5 | | | | √ | Undefined |
| FFFFF21BH | A/D conversion result register 5H | ADA0CR5H | | | √ | | Undefined |
| FFFFF21CH | A/D conversion result register 6 | ADA0CR6 | | | | √ | Undefined |
| FFFFF21DH | A/D conversion result register 6H | ADA0CR6H | | | √ | | Undefined |
| FFFFF21EH | A/D conversion result register 7 | ADA0CR7 | | | | √ | Undefined |
| FFFFF21FH | A/D conversion result register 7H | ADA0CR7H | | | √ | | Undefined |
| FFFFF220H | A/D conversion result register 8 | ADA0CR8 | | | | √ | Undefined |
| FFFFF221H | A/D conversion result register 8H | ADA0CR8H | | | √ | | Undefined |
| FFFFF222H | A/D conversion result register 9 | ADA0CR9 | | | | √ | Undefined |
| FFFFF223H | A/D conversion result register 9H | ADA0CR9H | | | √ | | Undefined |
| FFFFF224H | A/D conversion result register 10 | ADA0CR10 | | | | √ | Undefined |
| FFFFF225H | A/D conversion result register 10H | ADA0CR10H | | | √ | | Undefined |
| FFFFF226H | A/D conversion result register 11 | ADA0CR11 | | | | √ | Undefined |
| FFFFF227H | A/D conversion result register 11H | ADA0CR11H | | | √ | | Undefined |
| FFFFF280H | D/A conversion value setting register 0 | DA0CS0 | R/W | | √ | | 00H |
| FFFFF281H | D/A conversion value setting register 1 | DA0CS1 | | | √ | | 00H |
| FFFFF282H | D/A converter mode register | DA0M | | √ | √ | | 00H |
| FFFFF300H | Key return mode register | KRM | | √ | √ | | 00H |
| FFFFF308H | Selector operation control register 0 | SELCNT0 | | √ | √ | | 00H |
| FFFFF310H | CRC input register | CRCIN | | | √ | | 00H |
| FFFFF312H | CRC data register | CRCD | | | | √ | 0000H |
| FFFFF320H | Prescaler mode register 1 | PRSM1 | | √ | √ | | 00H |
| FFFFF321H | Prescaler compare register 1 | PRSCM1 | | | √ | | 00H |
| FFFFF324H | Prescaler mode register 2 | PRSM2 | | √ | √ | | 00H |
| FFFFF325H | Prescaler compare register 2 | PRSCM2 | | | √ | | 00H |
| FFFFF328H | Prescaler mode register 3 | PRSM3 | | √ | √ | | 00H |
| FFFFF329H | Prescaler compare register 3 | PRSCM3 | | | √ | | 00H |

(6/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| FFFFF340H | IIC division clock select register 0 | OCKS0 | R/W | | √ | | 00H |
| FFFFF344H | IIC division clock select register 1 | OCKS1 | | | √ | | 00H |
| FFFFF400H | Port 0 register | P0 | | √ | √ | | 00H[Note 1] |
| FFFFF402H | Port 1 register | P1 | | √ | √ | | 00H[Note 1] |
| FFFFF404H | Port 2 register[Note 2] | P2[Note 2] | | √ | √ | | 00H[Note 1] |
| FFFFF406H | Port 3 register | P3 | | √ | √ | | 00H[Note 1] |
| FFFFF408H | Port 4 register | P4 | | √ | √ | | 00H[Note 1] |
| FFFFF40AH | Port 5 register | P5 | | √ | √ | | 00H[Note 1] |
| FFFFF40CH | Port 6 register | P6 | | √ | √ | | 00H[Note 1] |
| FFFFF40EH | Port 7 register L | P7L | | √ | √ | | 00H[Note 1] |
| FFFFF40FH | Port 7 register H | P7H | | √ | √ | | 00H[Note 1] |
| FFFFF412H | Port 9 register | P9 | | | | √ | 0000H[Note 1] |
| FFFFF412H | Port 9 register L | P9L | | √ | √ | | 00H[Note 1] |
| FFFFF413H | Port 9 register H | P9H | | √ | √ | | 00H[Note 1] |
| FFFFF420H | Port 0 mode register | PM0 | | √ | √ | | FFH |
| FFFFF422H | Port 1 mode register | PM1 | | √ | √ | | FFH |
| FFFFF424H | Port 2 mode register[Note 2] | PM2[Note 2] | | √ | √ | | FFH |
| FFFFF426H | Port 3 mode register | PM3 | | √ | √ | | FFH |
| FFFFF428H | Port 4 mode register | PM4 | | √ | √ | | FFH |
| FFFFF42AH | Port 5 mode register | PM5 | | √ | √ | | FFH |
| FFFFF42CH | Port 6 mode register | PM6 | | √ | √ | | FFH |
| FFFFF42EH | Port 7 mode register L | PM7L | | √ | √ | | FFH |
| FFFFF42FH | Port 7 mode register H | PM7H | | √ | √ | | FFH |
| FFFFF432H | Port 9 mode register | PM9 | | | | √ | FFFFH |
| FFFFF432H | Port 9 mode register L | PM9L | | √ | √ | | FFH |
| FFFFF433H | Port 9 mode register H | PM9H | | √ | √ | | FFH |
| FFFFF440H | Port 0 mode control register | PMC0 | | √ | √ | | 00H |
| FFFFF444H | Port 2 mode control register[Note 2] | PMC2[Note 2] | | √ | √ | | 00H |
| FFFFF446H | Port 3 mode control register | PMC3 | | √ | √ | | 00H |
| FFFFF448H | Port 4 mode control register | PMC4 | | √ | √ | | 00H |
| FFFFF44AH | Port 5 mode control register | PMC5 | | √ | √ | | 00H |
| FFFFF44CH | Port 6 mode control register | PMC6 | | √ | √ | | 00H |
| FFFFF452H | Port 9 mode control register | PMC9 | | | | √ | 0000H |
| FFFFF452H | Port 9 mode control register L | PMC9L | | √ | √ | | 00H |
| FFFFF453H | Port 9 mode control register H | PMC9H | | √ | √ | | 00H |
| FFFFF460H | Port 0 function control register | PFC0 | | √ | √ | | 00H |
| FFFFF464H | Port 2 function control register[Note 2] | PFC2[Note 2] | | √ | √ | | 00H |
| FFFFF466H | Port 3 function control register | PFC3 | | √ | √ | | 00H |
| FFFFF468H | Port 4 function control register | PFC4 | | √ | √ | | 00H |

**Notes 1.** The output latch is 00H or 0000H. When these registers are in the input mode, the pin statuses are read.

    **2.** V850ES/JH3-U only

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| FFFFF46AH | Port 5 function control register | PFC5 | R/W | √ | √ | | 00H |
| FFFFF46CH | Port 6 function control register | PFC6 | | √ | √ | | 00H |
| FFFFF472H | Port 9 function control register | PFC9 | | | | √ | 0000H |
| FFFFF472H | Port 9 function control register L | PFC9L | | √ | √ | | 00H |
| FFFFF473H | Port 9 function control register H | PFC9H | | √ | √ | | 00H |
| FFFFF484H | Data wait control register 0 | DWC0 | | | | √ | 7777H |
| FFFFF488H | Address wait control register | AWC | | | | √ | FFFFH |
| FFFFF48AH | Bus cycle control register | BCC | | | | √ | AAAAH |
| FFFFF540H | TAB0 control register 0 | TAB0CTL0 | | √ | √ | | 00H |
| FFFFF541H | TAB0 control register 1 | TAB0CTL1 | | √ | √ | | 00H |
| FFFFF542H | TAB0 I/O control register 0 | TAB0IOC0 | | √ | √ | | 00H |
| FFFFF543H | TAB0 I/O control register 1 | TAB0IOC1 | | √ | √ | | 00H |
| FFFFF544H | TAB0 I/O control register 2 | TAB0IOC2 | | √ | √ | | 00H |
| FFFFF545H | TAB0 option register 0 | TAB0OPT0 | | √ | √ | | 00H |
| FFFFF546H | TAB0 capture/compare register 0 | TAB0CCR0 | | | | √ | 0000H |
| FFFFF548H | TAB0 capture/compare register 1 | TAB0CCR1 | | | | √ | 0000H |
| FFFFF54AH | TAB0 capture/compare register 2 | TAB0CCR2 | | | | √ | 0000H |
| FFFFF54CH | TAB0 capture/compare register 3 | TAB0CCR3 | | | | √ | 0000H |
| FFFFF54EH | TAB0 counter read buffer register | TAB0CNT | R | | | √ | 0000H |
| FFFFF550H | TAB0 I/O control register 4 | TAB0IOC4 | R/W | √ | √ | | 00H |
| FFFFF560H | TAB1 control register 0 | TAB1CTL0 | | √ | √ | | 00H |
| FFFFF561H | TAB1 control register 1 | TAB1CTL1 | | √ | √ | | 00H |
| FFFFF562H | TAB1 I/O control register 0 | TAB1IOC0 | | √ | √ | | 00H |
| FFFFF563H | TAB1 I/O control register 1 | TAB1IOC1 | | √ | √ | | 00H |
| FFFFF564H | TAB1 I/O control register 2 | TAB1IOC2 | | √ | √ | | 00H |
| FFFFF565H | TAB1 option register 0 | TAB1OPT0 | | √ | √ | | 00H |
| FFFFF566H | TAB1 capture/compare register 0 | TAB1CCR0 | | | | √ | 0000H |
| FFFFF568H | TAB1 capture/compare register 1 | TAB1CCR1 | | | | √ | 0000H |
| FFFFF56AH | TAB1 capture/compare register 2 | TAB1CCR2 | | | | √ | 0000H |
| FFFFF56CH | TAB1 capture/compare register 3 | TAB1CCR3 | | | | √ | 0000H |
| FFFFF56EH | TAB1 counter read buffer register | TAB1CNT | R | | | √ | 0000H |
| FFFFF570H | TAB1 I/O control register 4 | TAB1IOC4 | R/W | √ | √ | | 00H |
| FFFFF580H | TAB1 option register 1 | TAB1OPT1 | | √ | √ | | 00H |
| FFFFF581H | TAB1 option register 2 | TAB1OPT2 | | √ | √ | | 00H |
| FFFFF582H | TAB1 I/O control register 3 | TAB1IOC3 | | √ | √ | | A8H |
| FFFFF584H | TAB1 dead time compare register 1 | TAB1DTC | | | | √ | 0000H |
| FFFFF590H | High impedance output control register 0 | HZACTL0 | | √ | √ | | 00H |
| FFFFF591H | High impedance output control register 1 | HZACTL1 | | √ | √ | | 00H |
| FFFFF600H | TMT0 control register 0 | TT0CTL0 | | √ | √ | | 00H |
| FFFFF601H | TMT0 control register 1 | TT0CTL1 | | √ | √ | | 00H |
| FFFFF602H | TMT0 control register 2 | TT0CTL2 | | √ | √ | | 00H |
| FFFFF603H | TMT0 I/O control register 0 | TT0IOC0 | | √ | √ | | 00H |
| FFFFF604H | TMT0 I/O control register 1 | TT0IOC1 | | √ | √ | | 00H |

(8/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| FFFFF605H | TMT0 I/O control register 2 | TT0IOC2 | R/W | √ | √ | | 00H |
| FFFFF606H | TMT0 I/O control register 3 | TT0IOC3 | | √ | √ | | 00H |
| FFFFF607H | TMT0 option register 0 | TT0OPT0 | | √ | √ | | 00H |
| FFFFF608H | TMT0 option register 1 | TT0OPT1 | | √ | √ | | 00H |
| FFFFF609H | TMT0 option register 2 | TT0OPT2 | | √ | √ | | 00H |
| FFFFF60AH | TMT0 capture/compare register 0 | TT0CCR0 | | | | √ | 0000H |
| FFFFF60CH | TMT0 capture/compare register 1 | TT0CCR1 | | | | √ | 0000H |
| FFFFF60EH | TMT0 counter read buffer register | TT0CNT | R | | | √ | 0000H |
| FFFFF610H | TMT0 counter write register | TT0TCW | R/W | | | √ | 0000H |
| FFFFF630H | TAA0 control register 0 | TAA0CTL0 | | √ | √ | | 00H |
| FFFFF631H | TAA0 control register 1 | TAA0CTL1 | | √ | √ | | 00H |
| FFFFF632H | TAA0 I/O control register 0 | TAA0IOC0 | | √ | √ | | 00H |
| FFFFF633H | TAA0 I/O control register 1 | TAA0IOC1 | | √ | √ | | 00H |
| FFFFF634H | TAA0 I/O control register 2 | TAA0IOC2 | | √ | √ | | 00H |
| FFFFF635H | TAA0 option register 0 | TAA0OPT0 | | √ | √ | | 00H |
| FFFFF636H | TAA0 capture/compare register 0 | TAA0CCR0 | | | | √ | 0000H |
| FFFFF638H | TAA0 capture/compare register 1 | TAA0CCR1 | | | | √ | 0000H |
| FFFFF63AH | TAA0 counter read buffer register | TAA0CNT | R | | | √ | 0000H |
| FFFFF63CH | TAA0 I/O control register 4 | TAA0IOC4 | R/W | √ | √ | | 00H |
| FFFFF63DH | TAA0 option register 1 | TAA0OPT1 | | √ | √ | | 00H |
| FFFFF640H | TAA1 control register 0 | TAA1CTL0 | | √ | √ | | 00H |
| FFFFF641H | TAA1 control register 1 | TAA1CTL1 | | √ | √ | | 00H |
| FFFFF642H | TAA1 I/O control register 0 | TAA1IOC0 | | √ | √ | | 00H |
| FFFFF643H | TAA1 I/O control register 1 | TAA1IOC1 | | √ | √ | | 00H |
| FFFFF644H | TAA1 I/O control register 2 | TAA1IOC2 | | √ | √ | | 00H |
| FFFFF645H | TAA1 option register 0 | TAA1OPT0 | | √ | √ | | 00H |
| FFFFF646H | TAA1 capture/compare register 0 | TAA1CCR0 | | | | √ | 0000H |
| FFFFF648H | TAA1 capture/compare register 1 | TAA1CCR1 | | | | √ | 0000H |
| FFFFF64AH | TAA1 counter read buffer register | TAA1CNT | R | | | √ | 0000H |
| FFFFF64CH | TAA1 I/O control register 4 | TAA1IOC4 | R/W | √ | √ | | 00H |
| FFFFF650H | TAA2 control register 0 | TAA2CTL0 | | √ | √ | | 00H |
| FFFFF651H | TAA2 control register 1 | TAA2CTL1 | | √ | √ | | 00H |
| FFFFF652H | TAA2 I/O control register 0 | TAA2IOC0 | | √ | √ | | 00H |
| FFFFF653H | TAA2 I/O control register 1 | TAA2IOC1 | | √ | √ | | 00H |
| FFFFF654H | TAA2 I/O control register 2 | TAA2IOC2 | | √ | √ | | 00H |
| FFFFF655H | TAA2 option register 0 | TAA2OPT0 | | √ | √ | | 00H |
| FFFFF656H | TAA2 capture/compare register 0 | TAA2CCR0 | | | | √ | 0000H |
| FFFFF658H | TAA2 capture/compare register 1 | TAA2CCR1 | | | | √ | 0000H |
| FFFFF65AH | TAA2 counter read buffer register | TAA2CNT | R | | | √ | 0000H |
| FFFFF65CH | TAA2 I/O control register 4 | TAA2IOC4 | R/W | √ | √ | | 00H |
| FFFFF65DH | TAA2 option register 1 | TAA2OPT1 | | √ | √ | | 00H |
| FFFFF660H | TAA3 control register 0 | TAA3CTL0 | | √ | √ | | 00H |
| FFFFF661H | TAA3 control register 1 | TAA3CTL1 | | √ | √ | | 00H |

RENESAS

(9/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| FFFFF662H | TAA3 I/O control register 0 | TAA3IOC0 | R/W | √ | √ | | 00H |
| FFFFF663H | TAA3 I/O control register 1 | TAA3IOC1 | | √ | √ | | 00H |
| FFFFF664H | TAA3 I/O control register 2 | TAA3IOC2 | | √ | √ | | 00H |
| FFFFF665H | TAA3 option register 0 | TAA3OPT0 | | √ | √ | | 00H |
| FFFFF666H | TAA3 capture/compare register 0 | TAA3CCR0 | | | | √ | 0000H |
| FFFFF668H | TAA3 capture/compare register 1 | TAA3CCR1 | | | | √ | 0000H |
| FFFFF66AH | TAA3 counter read buffer register | TAA3CNT | R | | | √ | 0000H |
| FFFFF66CH | TAA3 I/O control register 4 | TAA3IOC4 | R/W | √ | √ | | 00H |
| FFFFF670H | TAA4 control register 0 | TAA4CTL0 | | √ | √ | | 00H |
| FFFFF671H | TAA4 control register 1 | TAA4CTL1 | | √ | √ | | 00H |
| FFFFF676H | TAA4 capture/compare register 0 | TAA4CCR0 | | | | √ | 0000H |
| FFFFF678H | TAA4 capture/compare register 1 | TAA4CCR1 | | | | √ | 0000H |
| FFFFF67AH | TAA4 counter read buffer register | TAA4CNT | R | | | √ | 0000H |
| FFFFF680H | TAA5 control register 0 | TAA5CTL0 | R/W | √ | √ | | 00H |
| FFFFF681H | TAA5 control register 1 | TAA5CTL1 | | √ | √ | | 00H |
| FFFFF682H | TAA5 I/O control register 0 | TAA5IOC0 | | √ | √ | | 00H |
| FFFFF683H | TAA5 I/O control register 1 | TAA5IOC1 | | √ | √ | | 00H |
| FFFFF684H | TAA5 I/O control register 2 | TAA5IOC2 | | √ | √ | | 00H |
| FFFFF685H | TAA5 option register 0 | TAA5OPT0 | | √ | √ | | 00H |
| FFFFF686H | TAA5 capture/compare register 0 | TAA5CCR0 | | | | √ | 0000H |
| FFFFF688H | TAA5 capture/compare register 1 | TAA5CCR1 | | | | √ | 0000H |
| FFFFF68AH | TAA5 counter read buffer register | TAA5CNT | R | | | √ | 0000H |
| FFFFF68CH | TAA5 I/O control register 4 | TAA5IOC4 | R/W | √ | √ | | 00H |
| FFFFF6C0H | Oscillation stabilization time select register | OSTS | | | √ | | 06H |
| FFFFF6C1H | PLL lockup time specification register | PLLS | | | √ | | 03H |
| FFFFF6D0H | Watchdog timer mode register 2 | WDTM2 | | | √ | | 67H |
| FFFFF6D1H | Watchdog timer enable register | WDTE | | | √ | | 9AH |
| FFFFF6E0H | Real-time output buffer register 0L | RTBL0 | | √ | √ | | 00H |
| FFFFF6E2H | Real-time output buffer register 0H | RTBH0 | | √ | √ | | 00H |
| FFFFF6E4H | Real-time output port mode register 0 | RTPM0 | | √ | √ | | 00H |
| FFFFF6E5H | Real-time output port control register 0 | RTPC0 | | √ | √ | | 00H |
| FFFFF700H | Port 0 function control expansion register | PFCE0 | | √ | √ | | 00H |
| FFFFF704H | Port 2 function control expansion register[Note] | PFCE2[Note] | | √ | √ | | 00H |
| FFFFF706H | Port 3 function control expansion register | PFCE3 | | √ | √ | | 00H |
| FFFFF708H | Port 4 function control expansion register | PFCE4 | | √ | √ | | 00H |
| FFFFF70AH | Port 5 function control expansion register | PFCE5 | | √ | √ | | 00H |
| FFFFF70CH | Port 6 function control expansion register | PFCE6 | | √ | √ | | 00H |
| FFFFF712H | Port 9 function control expansion register | PFCE9 | | | | √ | 0000H |
| FFFFF712H | Port 9 function control expansion register L | PFCE9L | | √ | √ | | 00H |
| FFFFF713H | Port 9 function control expansion register H | PFCE9H | | √ | √ | | 00H |
| FFFFF724H | TAA noise elimination control register | TANFC | | √ | √ | | 00H |
| FFFFF726H | TMT noise elimination control register | TTNFC | | √ | √ | | 00H |

**Note**   V850ES/JH3-U only

(10/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|----|----|-----|---------------|
| | | | | 1 | 8 | 16 | |
| FFFFF728H | Noise elimination control register | INTNFC | R/W | √ | √ | | 00H |
| FFFFF802H | System status register | SYS | | √ | √ | | 00H |
| FFFFF80CH | Internal oscillation mode register | RCM | | √ | √ | | 00H |
| FFFFF810H | DMA trigger factor register 0 | DTFR0 | | √ | √ | | 00H |
| FFFFF812H | DMA trigger factor register 1 | DTFR1 | | √ | √ | | 00H |
| FFFFF814H | DMA trigger factor register 2 | DTFR2 | | √ | √ | | 00H |
| FFFFF816H | DMA trigger factor register 3 | DTFR3 | | √ | √ | | 00H |
| FFFFF820H | Power save mode register | PSMR | | √ | √ | | 00H |
| FFFFF822H | Clock control register | CKC | | √ | √ | | 0AH |
| FFFFF824H | Lock register | LOCKR | R | √ | √ | | 00H |
| FFFFF828H | Processor clock control register | PCC | R/W | √ | √ | | 03H |
| FFFFF82CH | PLL control register | PLLCTL | | √ | √ | | 01H |
| FFFFF82EH | CPU operation clock status register | CCLS | R | √ | √ | | 00H |
| FFFFF870H | Clock monitor mode register | CLM | R/W | √ | √ | | 00H |
| FFFFF888H | Reset source flag register | RESF | | √ | √ | | 00H |
| FFFFF890H | Low-voltage detection register | LVIM | | √ | √ | | 00H |
| FFFFF892H | Internal RAM data status register | RAMS | | √ | √ | | 01H |
| FFFFF8B0H | Prescaler mode register 0 | PRSM0 | | √ | √ | | 00H |
| FFFFF8B1H | Prescaler compare register 0 | PRSCM0 | | | √ | | 00H |
| FFFFF9FCH | On-chip debug mode register[Note] | OCDM[Note] | | √ | √ | | 01H |
| FFFFFA00H | UARTC0 control register 0 | UC0CTL0 | | √ | √ | | 10H |
| FFFFFA01H | UARTC0 control register 1 | UC0CTL1 | | | √ | | 00H |
| FFFFFA02H | UARTC0 control register 2 | UC0CTL2 | | | √ | | FFH |
| FFFFFA03H | UARTC0 option control register 0 | UC0OPT0 | | √ | √ | | 14H |
| FFFFFA04H | UARTC0 status register | UC0STR | | √ | √ | | 00H |
| FFFFFA06H | UARTC0 receive data register | UC0RX | R | | | √ | 01FFH |
| FFFFFA06H | UARTC0 receive data register L | UC0RXL | | | √ | | FFH |
| FFFFFA08H | UARTC0 transmit data register | UC0TX | R/W | | | √ | 01FFH |
| FFFFFA08H | UARTC0 transmit data register L | UC0TXL | | | √ | | FFH |
| FFFFFA0AH | UARTC0 option control register 1 | UC0OPT1 | | √ | √ | | 00H |
| FFFFFA10H | UARTC1 control register 0 | UC1CTL0 | | √ | √ | | 10H |
| FFFFFA11H | UARTC1 control register 1 | UC1CTL1 | | | √ | | 00H |
| FFFFFA12H | UARTC1 control register 2 | UC1CTL2 | | | √ | | FFH |
| FFFFFA13H | UARTC1 option control register 0 | UC1OPT0 | | √ | √ | | 14H |
| FFFFFA14H | UARTC1 status register | UC1STR | | √ | √ | | 00H |
| FFFFFA16H | UARTC1 receive data register | UC1RX | R | | | √ | 01FFH |
| FFFFFA16H | UARTC1 receive data register L | UC1RXL | | | √ | | FFH |
| FFFFFA18H | UARTC1 transmit data register | UC1TX | R/W | | | √ | 01FFH |
| FFFFFA18H | UARTC1 transmit data register L | UC1TXL | | | √ | | FFH |
| FFFFFA1AH | UARTC1 option control register 1 | UC1OPT1 | | √ | √ | | 00H |
| FFFFFA20H | UARTC2 control register 0 | UC2CTL0 | | √ | √ | | 10H |
| FFFFFA21H | UARTC2 control register 1 | UC2CTL1 | | | √ | | 00H |
| FFFFFA22H | UARTC2 control register 2 | UC2CTL2 | | | √ | | FFH |

**Note** V850ES/JG3-U only

(11/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits 1 | Manipulatable Bits 8 | Manipulatable Bits 16 | Default Value |
|---|---|---|---|---|---|---|---|
| FFFFFA23H | UARTC2 option control register 0 | UC2OPT0 | R/W | √ | √ | | 14H |
| FFFFFA24H | UARTC2 status register | UC2STR | | √ | √ | | 00H |
| FFFFFA26H | UARTC2 receive data register | UC2RX | R | | | √ | 01FFH |
| FFFFFA26H | UARTC2 receive data register L | UC2RXL | | | √ | | FFH |
| FFFFFA28H | UARTC2 transmit data register | UC2TX | R/W | | | √ | 01FFH |
| FFFFFA28H | UARTC2 transmit data register L | UC2TXL | | | √ | | FFH |
| FFFFFA2AH | UARTC2 option control register 1 | UC2OPT1 | | √ | √ | | 00H |
| FFFFFA30H | UARTC3 control register 0 | UC3CTL0 | | √ | √ | | 10H |
| FFFFFA31H | UARTC3 control register 1 | UC3CTL1 | | | √ | | 00H |
| FFFFFA32H | UARTC3 control register 2 | UC3CTL2 | | | √ | | FFH |
| FFFFFA33H | UARTC3 option control register 0 | UC3OPT0 | | √ | √ | | 14H |
| FFFFFA34H | UARTC3 status register | UC3STR | | √ | √ | | 00H |
| FFFFFA36H | UARTC3 receive data register | UC3RX | R | | | √ | 01FFH |
| FFFFFA36H | UARTC3 receive data register L | UC3RXL | | | √ | | FFH |
| FFFFFA38H | UARTC3 transmit data register | UC3TX | R/W | | | √ | 01FFH |
| FFFFFA38H | UARTC3 transmit data register L | UC3TXL | | | √ | | FFH |
| FFFFFA3AH | UARTC3 option control register 1 | UC3OPT1 | | √ | √ | | 00H |
| FFFFFA40H | UARTC4 control register 0 | UC4CTL0 | | √ | √ | | 10H |
| FFFFFA41H | UARTC4 control register 1 | UC4CTL1 | | | √ | | 00H |
| FFFFFA42H | UARTC4 control register 2 | UC4CTL2 | | | √ | | FFH |
| FFFFFA43H | UARTC4 option control register 0 | UC4OPT0 | | √ | √ | | 14H |
| FFFFFA44H | UARTC4 status register | UC4STR | | √ | √ | | 00H |
| FFFFFA46H | UARTC4 receive data register | UC4RX | R | | | √ | 01FFH |
| FFFFFA46H | UARTC4 receive data register L | UC4RXL | | | √ | | FFH |
| FFFFFA48H | UARTC4 transmit data register | UC4TX | R/W | | | √ | 01FFH |
| FFFFFA48H | UARTC4 transmit data register L | UC4TXL | | | √ | | FFH |
| FFFFFA4AH | UARTC4 option control register 1 | UC4OPT1 | | √ | √ | | 00H |
| FFFFFA80H | TMM0 control register 0 | TM0CTL0 | | √ | √ | | 00H |
| FFFFFA84H | TMM0 compare register 0 | TM0CMP0 | | | | √ | 0000H |
| FFFFFA90H | TMM1 control register 0 | TM1CTL0 | | √ | √ | | 00H |
| FFFFFA94H | TMM1 compare register 0 | TM1CMP0 | | | | √ | 0000H |
| FFFFFAA0H | TMM2 control register 0 | TM2CTL0 | | √ | √ | | 00H |
| FFFFFAA4H | TMM2 compare register 0 | TM2CMP0 | | | | √ | 0000H |
| FFFFFAB0H | TMM3 control register 0 | TM3CTL0 | | √ | √ | | 00H |
| FFFFFAB4H | TMM3 compare register 0 | TM3CMP0 | | | | √ | 0000H |
| FFFFFAD0H | Sub-count register | RC1SUBC | R | | | √ | 0000H |
| FFFFFAD2H | Second count register | RC1SEC | R/W | | √ | | 00H |
| FFFFFAD3H | Minute count register | RC1MIN | | | √ | | 00H |
| FFFFFAD4H | Hour count register | RC1HOUR | | | √ | | 12H |
| FFFFFAD5H | Week count register | RC1WEEK | | | √ | | 00H |
| FFFFFAD6H | Day count register | RC1DAY | | | √ | | 01H |
| FFFFFAD7H | Month count register | RC1MONTH | | | √ | | 01H |
| FFFFFAD8H | Year count register | RC1YEAR | | | √ | | 00H |

(12/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits 1 | 8 | 16 | Default Value |
|---|---|---|---|---|---|---|---|
| FFFFFAD9H | Time error correction register | RC1SUBU | R/W | √ | √ | | 00H |
| FFFFFADAH | Alarm minute set register | RC1ALM | | | √ | | 00H |
| FFFFFADBH | Alarm time set register | RC1ALH | | | √ | | 12H |
| FFFFFADCH | Alarm week set register | RC1ALW | | √ | √ | | 00H |
| FFFFFADDH | RTC control register 0 | RC1CC0 | | √ | √ | | 00H |
| FFFFFADEH | RTC control register 1 | RC1CC1 | | √ | √ | | 00H |
| FFFFFADFH | RTC control register 2 | RC1CC2 | | √ | √ | | 00H |
| FFFFFAE0H | RTC control register 3 | RC1CC3 | | √ | √ | | 00H |
| FFFFFC00H | External interrupt falling edge specification register 0 | INTF0 | | √ | √ | | 00H |
| FFFFFC04H | External interrupt falling edge specification register 2[Note 1] | INTF2[Note 1] | | √ | √ | | 00H |
| FFFFFC06H | External interrupt falling edge specification register 3 | INTF3 | | √ | √ | | 00H |
| FFFFFC08H | External interrupt falling edge specification register 4 | INTF4 | | √ | √ | | 00H |
| FFFFFC0AH | External interrupt falling edge specification register 5[Note 2] | INTF5[Note 2] | | √ | √ | | 00H |
| FFFFFC12H | External interrupt falling edge specification register 9 | INTF9 | | | | √ | 0000H |
| FFFFFC12H | External interrupt falling edge specification register 9L | INTF9L | | √ | √ | | 00H |
| FFFFFC13H | External interrupt falling edge specification register 9H | INTF9H | | √ | √ | | 00H |
| FFFFFC20H | External interrupt rising edge specification register 0 | INTR0 | | √ | √ | | 00H |
| FFFFFC24H | External interrupt rising edge specification register 2[Note 1] | INTR2[Note 1] | | √ | √ | | 00H |
| FFFFFC26H | External interrupt rising edge specification register 3 | INTR3 | | √ | √ | | 00H |
| FFFFFC28H | External interrupt rising edge specification register 4 | INTR4 | | √ | √ | | 00H |
| FFFFFC2AH | External interrupt rising edge specification register 5[Note 2] | INTR5[Note 2] | | √ | √ | | 00H |
| FFFFFC32H | External interrupt rising edge specification register 9 | INTR9 | | | | √ | 0000H |
| FFFFFC32H | External interrupt rising edge specification register 9L | INTR9L | | √ | √ | | 00H |
| FFFFFC33H | External interrupt rising edge specification register 9H | INTR9H | | √ | √ | | 00H |
| FFFFFC60H | Port 0 function register | PF0 | | √ | √ | | 00H |
| FFFFFC64H | Port 2 function register[Note 1] | PF2[Note 1] | | √ | √ | | 00H |
| FFFFFC66H | Port 3 function register | PF3 | | √ | √ | | 00H |
| FFFFFC68H | Port 4 function register | PF4 | | √ | √ | | 00H |
| FFFFFC6AH | Port 5 function register | PF5 | | √ | √ | | 00H |
| FFFFFC72H | Port 9 function register | PF9 | | | | √ | 0000H |
| FFFFFC72H | Port 9 function register L | PF9L | | √ | √ | | 00H |
| FFFFFD00H | CSIF0 control register 0 | CF0CTL0 | | √ | √ | | 01H |
| FFFFFD01H | CSIF0 control register 1 | CF0CTL1 | | √ | √ | | 00H |
| FFFFFD02H | CSIF0 control register 2 | CF0CTL2 | | | √ | | 00H |
| FFFFFD03H | CSIF0 status register | CF0STR | | √ | √ | | 00H |
| FFFFFD04H | CSIF0 receive data register | CF0RX | R | | | √ | 0000H |
| FFFFFD04H | CSIF0 receive data register L | CF0RXL | | | √ | | 00H |
| FFFFFD06H | CSIF0 transmit data register | CF0TX | R/W | | | √ | 0000H |
| FFFFFD06H | CSIF0 transmit data register L | CF0TXL | | | √ | | 00H |
| FFFFFD10H | CSIF1 control register 0 | CF1CTL0 | | √ | √ | | 01H |
| FFFFFD11H | CSIF1 control register 1 | CF1CTL1 | | √ | √ | | 00H |
| FFFFFD12H | CSIF1 control register 2 | CF1CTL2 | | | √ | | 00H |

**Notes 1.** V850ES/JH3-U only
**2.** V850ES/JG3-U only

(13/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|-----|-----|-----|---------------|
| | | | | 1 | 8 | 16 | |
| FFFFFD13H | CSIF1 status register | CF1STR | R/W | √ | √ | | 00H |
| FFFFFD14H | CSIF1 receive data register | CF1RX | R | | | √ | 0000H |
| FFFFFD14H | CSIF1 receive data register L | CF1RXL | | | √ | | 00H |
| FFFFFD16H | CSIF1 transmit data register | CF1TX | R/W | | | √ | 0000H |
| FFFFFD16H | CSIF1 transmit data register L | CF1TXL | | | √ | | 00H |
| FFFFFD20H | CSIF2 control register 0 | CF2CTL0 | | √ | √ | | 01H |
| FFFFFD21H | CSIF2 control register 1 | CF2CTL1 | | √ | √ | | 00H |
| FFFFFD22H | CSIF2 control register 2 | CF2CTL2 | | | √ | | 00H |
| FFFFFD23H | CSIF2 status register | CF2STR | | √ | √ | | 00H |
| FFFFFD24H | CSIF2 receive data register | CF2RX | R | | | √ | 0000H |
| FFFFFD24H | CSIF2 receive data register L | CF2RXL | | | √ | | 00H |
| FFFFFD26H | CSIF2 transmit data register | CF2TX | R/W | | | √ | 0000H |
| FFFFFD26H | CSIF2 transmit data register L | CF2TXL | | | √ | | 00H |
| FFFFFD30H | CSIF3 control register 0 | CF3CTL0 | | √ | √ | | 01H |
| FFFFFD31H | CSIF3 control register 1 | CF3CTL1 | | √ | √ | | 00H |
| FFFFFD32H | CSIF3 control register 2 | CF3CTL2 | | | √ | | 00H |
| FFFFFD33H | CSIF3 status register | CF3STR | | √ | √ | | 00H |
| FFFFFD34H | CSIF3 receive data register | CF3RX | R | | | √ | 0000H |
| FFFFFD34H | CSIF3 receive data register L | CF3RXL | | | √ | | 00H |
| FFFFFD36H | CSIF3 transmit data register | CF3TX | R/W | | | √ | 0000H |
| FFFFFD36H | CSIF3 transmit data register L | CF3TXL | | | √ | | 00H |
| FFFFFD40H | CSIF4 control register 0 | CF4CTL0 | | √ | √ | | 01H |
| FFFFFD41H | CSIF4 control register 1 | CF4CTL1 | | √ | √ | | 00H |
| FFFFFD42H | CSIF4 control register 2 | CF4CTL2 | | | √ | | 00H |
| FFFFFD43H | CSIF4 status register | CF4STR | | √ | √ | | 00H |
| FFFFFD44H | CSIF4 receive data register | CF4RX | R | | | √ | 0000H |
| FFFFFD44H | CSIF4 receive data register L | CF4RXL | | | √ | | 00H |
| FFFFFD46H | CSIF4 transmit data register | CF4TX | R/W | | | √ | 0000H |
| FFFFFD46H | CSIF4 transmit data register L | CF4TXL | | | √ | | 00H |
| FFFFFD80H | IIC shift register 0 | IIC0 | | | √ | | 00H |
| FFFFFD82H | IIC control register 0 | IICC0 | | √ | √ | | 00H |
| FFFFFD83H | Slave address register 0 | SVA0 | | | √ | | 00H |
| FFFFFD84H | IIC clock select register 0 | IICCL0 | | √ | √ | | 00H |
| FFFFFD85H | IIC function expansion register 0 | IICX0 | | √ | √ | | 00H |
| FFFFFD86H | IIC status register 0 | IICS0 | R | √ | √ | | 00H |
| FFFFFD8AH | IIC flag register 0 | IICF0 | R/W | √ | √ | | 00H |
| FFFFFD90H | IIC shift register 1 | IIC1 | | | √ | | 00H |
| FFFFFD92H | IIC control register 1 | IICC1 | | √ | √ | | 00H |
| FFFFFD93H | Slave address register 1 | SVA1 | | | √ | | 00H |
| FFFFFD94H | IIC clock select register 1 | IICCL1 | | √ | √ | | 00H |
| FFFFFD95H | IIC function expansion register 1 | IICX1 | | √ | √ | | 00H |
| FFFFFD96H | IIC status register1 | IICS1 | R | √ | √ | | 00H |
| FFFFFD9AH | IIC flag register 1 | IICF1 | R/W | √ | √ | | 00H |

(14/14)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| FFFFFDA0H | IIC shift register 2 | IIC2 | R/W | | √ | | 00H |
| FFFFFDA2H | IIC control register 2 | IICC2 | | √ | √ | | 00H |
| FFFFFDA3H | Slave address register 2 | SVA2 | | | √ | | 00H |
| FFFFFDA4H | IIC clock select register 2 | IICCL2 | | √ | √ | | 00H |
| FFFFFDA5H | IIC function expansion register 2 | IICX2 | | √ | √ | | 00H |
| FFFFFDA6H | IIC status register 2 | IICS2 | R | √ | √ | | 00H |
| FFFFFDAAH | IIC flag register 2 | IICF2 | R/W | √ | √ | | 00H |
| FFFFFF40H | USB clock select register | UCKSEL | | √ | √ | | 00H |
| FFFFFF41H | USB function control register | UFCKMSK | | √ | √ | | 03H |
| FFFFFF42H | USB function selection register | UHCKMSK | | √ | √ | | 03H |
| FFFFFF60H | External DMA request enable register | EXDRQEN | | | √ | | 00H |

### 3.4.7  Special registers

Special registers are registers that are protected from being written with illegal data due to a program loop.  The V850ES/JG3-U and V850ES/JH3-U have the following eight special registers.

- Power save control register (PSC)
- Clock control register (CKC)
- Processor clock control register (PCC)
- Clock monitor mode register (CLM)
- Reset source flag register (RESF)
- Low-voltage detection register (LVIM)
- Internal RAM data status register (RAMS)
- On-chip debug mode register (OCDM) (V850ES/JG3-U only)

In addition, the PRCDM register is provided to protect against a write access to the special registers so that the application system does not inadvertently stop due to a program loop.  A write access to the special registers is made in a specific sequence, and an illegal store operation is reported to the SYS register.

**(1) Setting data to special registers**

Set data to the special registers in the following sequence.

<1>     Disable DMA operation.

<2>     Prepare data to be set to the special register in a general-purpose register.

<3>     Write the data prepared in <2> to the PRCMD register.

<4>     Write the setting data to the special register (by using the following instructions).

- Store instruction (ST/SST instruction)

- Bit manipulation instruction (SET1/CLR1/NOT1 instruction)

(<5> to <9> Insert NOP instructions (5 instructions).)**Note**

<10>    Enable DMA operation if necessary.

[Example]  With PSC register (setting standby mode)

```
      ST.B r11, PSMR[r0]     ; Set PSMR register (setting IDLE1, IDLE2, and STOP modes).
<1>CLR1 0, DCHCn[r0]         ; Disable DMA operation.  n = 0 to 3
<2>MOV0x02, r10
<3>ST.B r10, PRCMD[r0]       ; Write PRCMD register.
<4>ST.B r10, PSC[r0]         ; Set PSC register.
<5>NOP^Note                  ; Dummy instruction
<6>NOP^Note                  ; Dummy instruction
<7>NOP^Note                  ; Dummy instruction
<8>NOP^Note                  ; Dummy instruction
<9>NOP^Note                  ; Dummy instruction
<10>SET1 0, DCHCn[r0]        ; Enable DMA operation.  n = 0 to 3
(next instruction)
```

There is no special sequence to read a special register.

**Note**   Five NOP instructions or more must be inserted immediately after setting the IDLE1 mode, IDLE2 mode, or STOP mode (by setting the PSC.STP bit to 1).

**Cautions 1.   When a store instruction is executed to store data in the command register, interrupts are not acknowledged.  This is because it is assumed that steps <3> and <4> above are performed by successive store instructions.  If another instruction is placed between <3> and <4>, and if an interrupt is acknowledged by that instruction, the above sequence may not be established, causing malfunction.**

**2.   Although dummy data is written to the PRCMD register, use the same general-purpose register used to set the special register (<4> in Example) to write data to the PRCMD register (<3> in Example).  The same applies when a general-purpose register is used for addressing.**

**(2) Command register (PRCMD)**

The PRCMD register is an 8-bit register that protects the registers that may seriously affect the application system from being written, so that the system does not inadvertently stop due to a program hang-up.  The first write access to a special register is valid after data has been written in advance to the PRCMD register.  In this way, the value of the special register can be rewritten only in a specific sequence, so as to protect the register from an illegal write access.

The PRCMD register is write-only, in 8-bit units (undefined data is read when this register is read).

After reset: Undefined        W        Address: FFFFF1FCH

|       | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|-------|------|------|------|------|------|------|------|------|
| PRCMD | REG7 | REG6 | REG5 | REG4 | REG3 | REG2 | REG1 | REG0 |
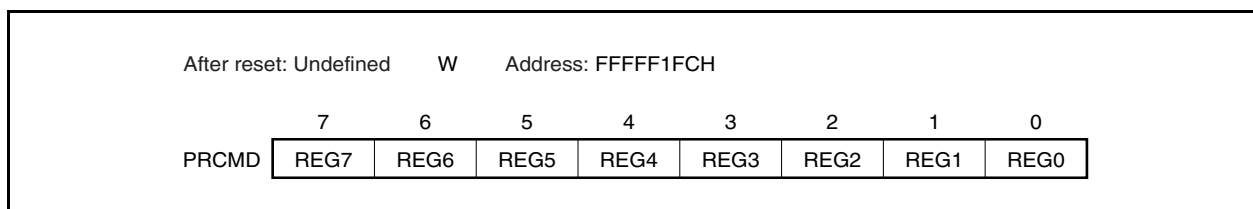
**(3) System status register (SYS)**

Status flags that indicate the operation status of the overall system are allocated to this register.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H        R/W        Address: FFFFF802H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| SYS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PRERR |

| PRERR | Detects protection error |
|---|---|
| 0 | Protection error did not occur |
| 1 | Protection error occurred |

The PRERR flag operates under the following conditions.

**(a) Set condition (PRERR flag = 1)**

(i) When data is written to a special register without writing anything to the PRCMD register (when <4> is executed without executing <3> in **3.4.7 (1)  Setting data to special registers**)

(ii) When data is written to an on-chip peripheral I/O register other than a special register (including execution of a bit manipulation instruction) after writing data to the PRCMD register (if <4> in **3.4.7 (1)  Setting data to special registers** is not the setting of a special register)

**Remark**   Even if an on-chip peripheral I/O register is read (except by a bit manipulation instruction) between an operation to write the PRCMD register and an operation to write a special register, the PRERR flag is not set, and the set data can be written to the special register.

**(b) Clear condition (PRERR flag = 0)**

(i) When 0 is written to the PRERR flag

(ii) When the system is reset

**Cautions 1.** If 0 is written to the PRERR bit of the SYS register, which is not a special register, immediately after a write access to the PRCMD register, the PRERR bit is cleared to 0 (the write access takes precedence).

**2.** If data is written to the PRCMD register, which is not a special register, immediately after a write access to the PRCMD register, the PRERR bit is set to 1.

### 3.4.8  Cautions

**(1)  Registers to be set first**

Be sure to set the following registers first when using the V850ES/JG3-U and V850ES/JH3-U.

- System wait control register (VSWC)
- On-chip debug mode register (OCDM) (V850ES/JG3-U only)
- Watchdog timer mode register 2 (WDTM2)

After setting the VSWC, OCDM, and WDTM2 registers, set the other registers as necessary.
When using the external bus, set each pin to the alternate-function bus control pin mode by using the port-related registers after setting the above registers.

**(a)  System wait control register (VSWC)**

The VSWC register controls wait of bus access to the on-chip peripheral I/O registers.
Three clocks are required to access an on-chip peripheral I/O register (without a wait cycle).  The V850ES/JG3-U and V850ES/JH3-U require wait cycles according to the operating frequency.  Set the following value to the VSWC register in accordance with the frequency used.
The VSWC register can be read or written in 8-bit units.
Reset sets this register to 77H.

After reset: 77H     R/W     Address: FFFFF06EH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| VSWC | | | | | | | | |

| Operating Frequency ($f_{CPU}$) | Set Value of VSWC | Number of Waits |
|---|---|---|
| $f_{CPU} < 16.6$ MHz | 00H | 0 (no waits) |
| $16.6$ MHz $\leq f_{CPU} < 25$ MHz | 01H | 1 |
| $25$ MHz $\leq f_{CPU} < 33.3$ MHz | 11H | 2 |
| $33.3$ MHz $\leq f_{CPU} \leq 48$ MHz | 12H | 3 |

**(b)  On-chip debug mode register (OCDM)**

For details, see **CHAPTER 32  ON-CHIP DEBUG FUNCTION**.

**(c)  Watchdog timer mode register 2 (WDTM2)**

The WDTM2 register sets the overflow time and the operation clock of watchdog timer 2.
Watchdog timer 2 automatically starts in the reset mode after reset is released.  Write the WDTM2 register to activate this operation.
For details, see **CHAPTER 13  FUNCTIONS OF WATCHDOG TIMER 2**.

**(2)  Accessing specific on-chip peripheral I/O registers**

This product has two types of internal system buses.

One is a CPU bus and the other is a peripheral bus that interfaces with low-speed peripheral hardware.

The clock of the CPU bus and the clock of the peripheral bus are asynchronous.  If an access to the CPU and an access to the peripheral hardware conflict, therefore, unexpected illegal data may be transferred.  If there is a possibility of a conflict, the number of cycles for accessing the CPU changes when the peripheral hardware is accessed, so that correct data is transferred.  As a result, the CPU does not start processing of the next instruction but enters the wait status.  If this wait status occurs, the number of clocks required to execute an instruction increases by the number of wait clocks shown below.

This must be taken into consideration if real-time processing is required.

When specific on-chip peripheral I/O registers are accessed, more wait states may be required in addition to the wait states set by the VSWC register.

The access conditions and how to calculate the number of wait states to be inserted (number of CPU clocks) at this time are shown below.

| Peripheral Function | Register Name | Access | k |
|---|---|---|---|
| 16-bit timer/event counter AA (TAA) (n = 0 to 5, m = 0 to 3, 5) | TAAnCNT | Read | 1 or 2 |
| | TAAnCCR0, TAAnCCR1 | Write | • 1st access: No wait<br>• Continuous write: 0 to 3 |
| | | Read | 1 or 2 |
| | TAAmIOC4 | Write | • 1st access: No wait<br>• Continuous write: 0 to 3 |
| | | Read | 1 or 2 |
| 16-bit timer/event counter AB (TAB) (n = 0, 1) | TABnCNT | Read | 1 or 2 |
| | TABnCCR0 to TABnCCR3 | Write | • 1st access: No wait<br>• Continuous write: 0 to 3 |
| | | Read | 1 or 2 |
| | TABnIOC4 | Write | • 1st access: No wait<br>• Continuous write: 0 to 3 |
| | | Read | 1 or 2 |
| Motor control | TAB0OPT1 | Write | • 1st access: No wait<br>• Continuous write: 0 to 3 |
| | TAB0DTC | Write | • 1st access: No wait<br>• Continuous write: 0 to 3 |
| TMT | TT0CNT | Read | 1 or 2 |
| | TT0TCR0, TT0TCR1 | Write | • 1st access: No wait<br>• Continuous write: 0 to 3 |
| | | Read | 1 or 2 |
| Watchdog timer 2 (WDT2) | WDTM2 | Write (when WDT2 operating) | 3 |
| Real-time output function (RTO) | RTBL0, RTBH0 | Write (RTPC0.RTPOE0 bit = 0) | 1 |
| A/D converter | ADA0M0 | Read | 1 or 2 |
| | ADA0CR0 to ADA0CR11 | Read | 1 or 2 |
| | ADA0CR0H to ADA0CR11H | Read | 1 or 2 |
| I²C00 to I²C02 | IICS0 to IICS2 | Read | 1 |
| CRC | CRCD | Write | 1 |

Number of clocks necessary for access = $3 + i + j + (2 + j) \times k$

**Caution  Accessing the above registers is prohibited in the following statuses.  If a wait cycle is generated, it can only be cleared by a reset.**
- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

**Remark  i:  Values (0) of higher 4 bits of VSWC register**
**            j:  Values (0 or 1) of lower 4 bits of VSWC register**

**(3) Restriction on conflict between sld instruction and interrupt request**

**(a) Description**

If a conflict occurs between the decode operation of an instruction in <2> immediately before the sld instruction following an instruction in <1> and an interrupt request before the instruction in <1> is complete, the execution result of the instruction in <1> may not be stored in a register.

Instruction <1>
- ld instruction:              ld.b, ld.h, ld.w, ld.bu, ld.hu
- sld instruction:             sld.b, sld.h, sld.w, sld.bu, sld.hu
- Multiplication instruction:  mul, mulh, mulhi, mulu

Instruction <2>

| mov    reg1, reg2 | not    reg1, reg2 | satsubr reg1, reg2 | satsub  reg1, reg2 |
| satadd reg1, reg2 | satadd imm5, reg2 | or reg1, reg2 <br> subr    reg1, reg2 | xor     reg1, reg2 |
| and    reg1, reg2 | tst reg1, reg2 <br> add    imm5, reg2 | cmp    reg1, reg2 | sub     reg1, reg2 |
| add    reg1, reg2 | shr    imm5, reg2 | sar    imm5, reg2 | cmp    imm5, reg2 |
| mulh   reg1, reg2 | | | shl     imm5, reg2 |

<Example>

<i>   ld.w  [r11], r10     If the decode operation of the mov instruction <ii> immediately before the sld
      •                   instruction <iii> and an interrupt request conflict before execution of the ld instruction
      •                   <i> is complete, the execution result of instruction <i> may not be stored in a register.
      •
<ii>  mov  r10, r28
<iii> sld.w 0x28, r10

**(b) Countermeasure**

<1>  When compiler (CA850) is used
     Use CA850 Ver. 2.61 or later because generation of the corresponding instruction sequence can be automatically suppressed.
<2>  For assembler
     When executing the sld instruction immediately after instruction <ii>, avoid the above operation using either of the following methods.

- Insert a nop instruction immediately before the sld instruction.
- Do not use the same register as the sld instruction destination register in the above instruction <ii> executed immediately before the sld instruction.

# CHAPTER 4  PORT FUNCTIONS

## 4.1  Features

○ I/O ports
- V850ES/JG3-U:  75

    5 V tolerant/N-ch open-drain output selectable:  22
- V850ES/JH3-U:  96

    5 V tolerant/N-ch open-drain output selectable:  25

○ Input/output specifiable in 1-bit units

## 4.2  Basic Port Configuration

The V850ES/JG3-U features a total of 75 I/O ports consisting of ports 0, 1, 3 to 7, 9, CM, CT, and DL.

The V850ES/JH3-U features a total of 96 I/O ports consisting of ports 0 to 7, 9, CM, CS, CT, DH, and DL.

The port configuration is shown below.

**Table 4-1.  I/O Buffer Power Supplies for Pins (V850ES/JG3-U)**

| Power Supply | Corresponding Pins |
|---|---|
| $AV_{REF0}$ | Port 7 |
| $AV_{REF1}$ | Port 1 |
| $EV_{DD}$ | $\overline{RESET}$, ports 0, 3 to 6, 9, CM, CT, DL |

**Table 4-2.  I/O Buffer Power Supplies for Pins (V850ES/JH3-U)**

| Power Supply | Corresponding Pins |
|---|---|
| $AV_{REF0}$ | Port 7 |
| $AV_{REF1}$ | Port 1 |
| $EV_{DD}$ | $\overline{RESET}$, ports 0, 2 to 6, 9, CM, CS, CT, DH, DL |

**Figure 4-1. Port Configuration Diagram (V850ES/JG3-U)**
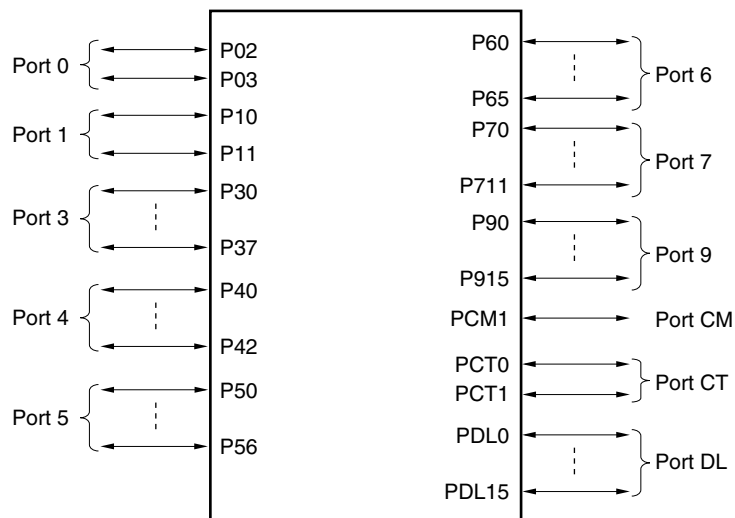


**Caution   Ports 0, 3 to 5 are 5 V tolerant.**

\<R\>

**Figure 4-2. Port Configuration Diagram (V850ES/JH3-U)**
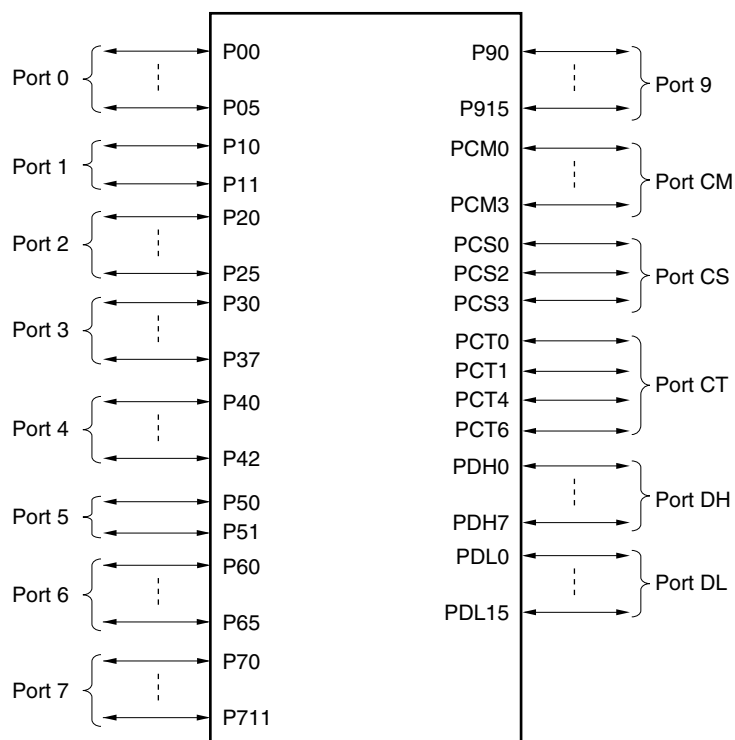


**Caution   Ports 0, 3 to 5 are 5 V tolerant.**

\<R\>

## 4.3  Port Configuration

**Table 4-3.  Port Configuration (V850ES/JG3-U)**

| Item | Configuration |
|---|---|
| Control register | Port n mode register (PMn: n = 0, 1, 3 to 7, 9, CM, CT, DL)<br>Port n mode control register (PMCn: n = 0, 3 to 5, 9, CM, CT, DL)<br>Port n function control register (PFCn: n = 0, 3 to 6, 9)<br>Port n function control expansion register (PFCEn: n = 4 to 6, 9)<br>Port n function register (PFn: n = 0, 3 to 5, 9) |
| Ports | I/O: 75 |

**Table 4-4.  Port Configuration (V850ES/JH3-U)**

| Item | Configuration |
|---|---|
| Control register | Port n mode register (PMn: n = 0 to 7, 9, CM, CS, CT, DH, DL)<br>Port n mode control register (PMCn: n = 0, 2 to 6, 9, CM, CS, CT, DH, DL)<br>Port n function control register (PFCn: n = 0, 2 to 6, 9)<br>Port n function control expansion register (PFCEn: n = 4 to 6, 9)<br>Port n function register (PFn: n = 0, 2 to 5, 9) |
| Ports | I/O: 96 |

**(1) Port n register (Pn)**

Data is input from or output to an external device by writing or reading the Pn register.

The Pn register consists of a port latch that holds output data, and a circuit that reads the status of pins.

Each bit of the Pn register corresponds to one pin of port n, and can be read or written in 1-bit units.

After reset: 00H (output latch)       R/W

| | 7 | 6 | 5 | 7 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Pn | Pn7 | Pn6 | Pn5 | Pn4 | Pn3 | Pn2 | Pn1 | Pn0 |

| Pnm | Control of output data (in output mode) |
|-----|------------------------------------------|
| 0 | Outputs 0. |
| 1 | Outputs 1. |
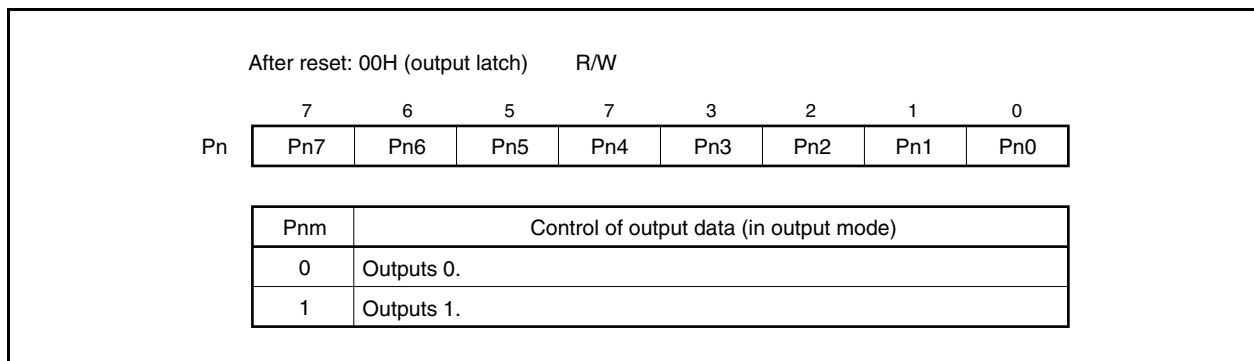
Data is written to or read from the Pn register as follows, regardless of the setting of the PMCn register.

**Table 4-5.  Writing/Reading Pn Register**

| Setting of PMn Register | Writing to Pn Register | Reading from Pn Register |
|--------------------------|-------------------------|----------------------------|
| Output mode<br>(PMnm = 0) | Data is written to the output latch[Note].<br>In the port mode (PMCn = 0), the contents of the output latch are output from the pins. | The value of the output latch is read. |
| Input mode<br>(PMnm = 1) | Data is written to the output latch.<br>The pin status is not affected[Note]. | The pin status is read. |

**Note**  The value written to the output latch is retained until a new value is written to the output latch.

**(2) Port n mode register (PMn)**

The PMn register specifies the input or output mode of the corresponding port pin.

Each bit of this register corresponds to one pin of port n, and the input or output mode can be specified in 1-bit units.

After reset: FFH      R/W

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMn | PMn7 | PMn6 | PMn5 | PMn4 | PMn3 | PMn2 | PMn1 | PMn0 |

| PMnm | Control of input/output mode |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(3) Port n mode control register (PMCn)**

The PMCn register specifies the port mode or alternate function.

Each bit of this register corresponds to one pin of port n, and the mode of the port can be specified in 1-bit units.

After reset: 00H      R/W

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCn | PMCn7 | PMCn6 | PMCn5 | PMCn4 | PMCn3 | PMCn2 | PMCn1 | PMCn0 |

| PMCnm | Specification of operation mode |
|---|---|
| 0 | Port mode |
| 1 | Alternate-function mode |

**(4) Port n function control register (PFCn)**

The PFCn register specifies the alternate function of a port pin to be used if the pin has two alternate functions.

Each bit of this register corresponds to one pin of port n, and the alternate function of a port pin can be specified in 1-bit units.

After reset: 00H      R/W

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCn | PFCn7 | PFCn6 | PFCn5 | PFCn4 | PFCn3 | PFCn2 | PFCn1 | PFCn0 |

| PFCnm | Specification of alternate function |
|---|---|
| 0 | Alternate function 1 |
| 1 | Alternate function 2 |

**(5) Port n function control expansion register (PFCEn)**

The PFCEn register specifies the alternate function of a port pin to be used if the pin has three or more alternate functions.

Each bit of this register corresponds to one pin of port n, and the alternate function of a port pin can be specified in 1-bit units.

After reset: 00H      R/W

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCEn | PFCEn7 | PFCEn6 | PFCEn5 | PFCEn4 | PFCEn3 | PFCEn2 | PFCEn1 | PFCEn0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCn | PFCn7 | PFCn6 | PFCn5 | PFCn4 | PFCn3 | PFCn2 | PFCn1 | PFCn0 |

| PFCEnm | PFCnm | Specification of alternate function |
|---|---|---|
| 0 | 0 | Alternate function 1 |
| 0 | 1 | Alternate function 2 |
| 1 | 0 | Alternate function 3 |
| 1 | 1 | Alternate function 4 |

**(6) Port n function register (PFn)**

The PFn register specifies normal output or N-ch open-drain output.

Each bit of this register corresponds to one pin of port n, and the output mode of the port pin can be specified in 1-bit units.

After reset: 00H      R/W

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFn | PFn7 | PFn6 | PFn5 | PFn4 | PFn3 | PFn2 | PFn1 | PFn0 |

| PFnm[Note] | Control of normal output/N-ch open-drain output |
|---|---|
| 0 | Normal output (CMOS output) |
| 1 | N-ch open-drain output |

**Note**   The PFnm bit of the PFn register is valid only when the PMnm bit of the PMn register is 0 (when the output mode is specified) in port mode (PMCnm bit = 0).  When the PMnm bit is 1 (when the input mode is specified), the set value of the PFn register is invalid.

**(7) Port setting**

Set a port as illustrated below.

**Figure 4-3.  Setting of Each Register and Pin Function**



**Remark**   Set the alternate functions in the following sequence.

<1> Set the PFCn and PFCEn registers.
<2> Set the PMCn register.
<3> Set the INTRn or INTFn register (to specify an external interrupt pin).

If the PMCn register is set first, an unintended function may be set while the PFCn and PFCEn registers are being set.
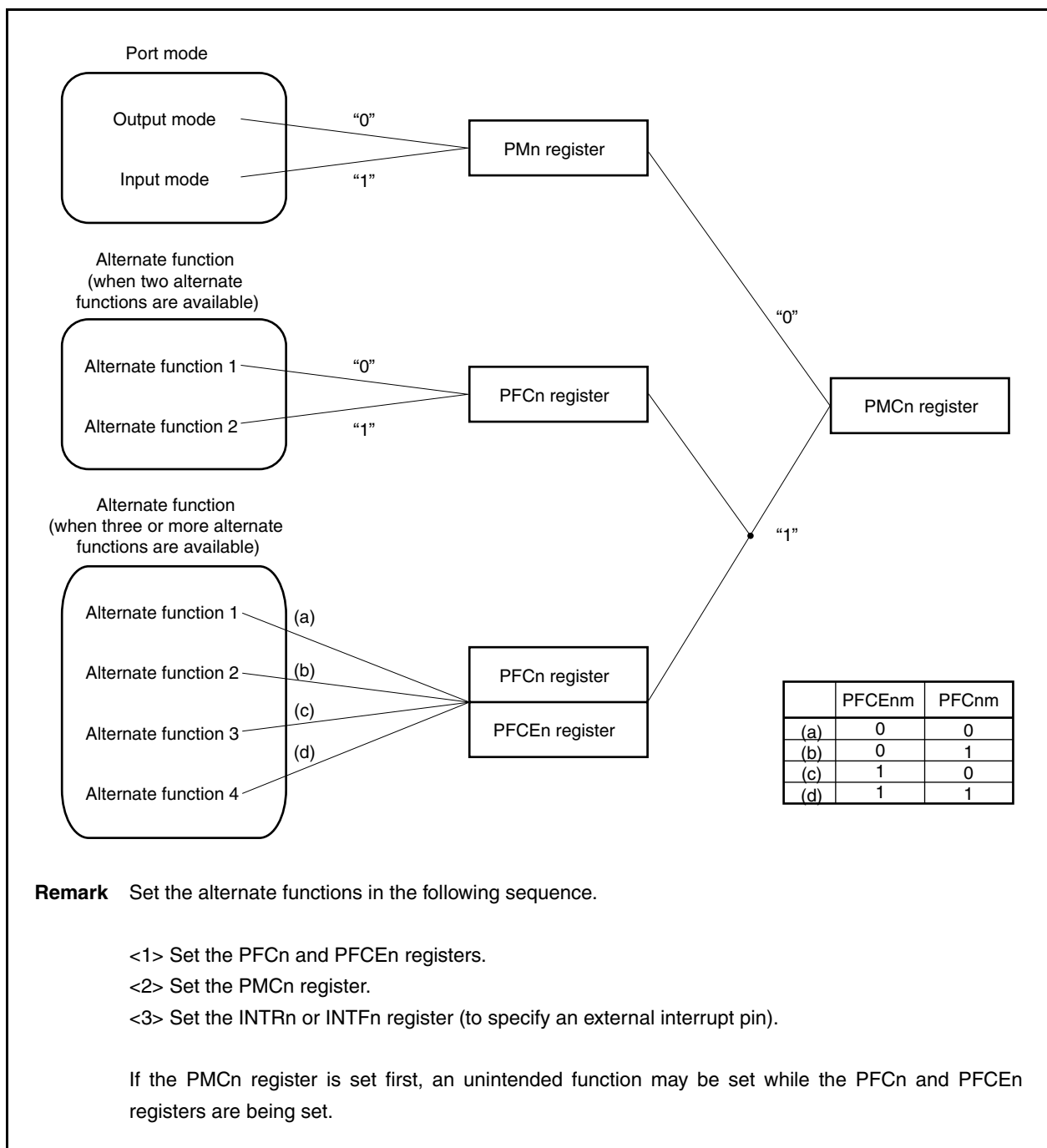
### 4.3.1  Port 0

Port 0 is 2-bit (V850ES/JG3-U)/6-bit (V850ES/JH3-U) port for which I/O settings can be controlled in 1-bit units.

Port 0 includes the following alternate-function pins.

**Table 4-6.  Port 0 Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| P00 | – | 8 | INTP00 | Input | Selectable as N-ch open-drain output |
| P01 | – | 9 | INTP01 | Input | |
| P02 | 6 | 6 | NMI | Input | |
| P03 | 7 | 7 | INTP02/ADTRG/UCLK | Input | |
| P04 | – | 26 | INTP03 | Input | |
| P05 | – | 27 | INTP04 | Input | |

**Caution   The P00 to P05 pins have hysteresis characteristics in the input mode of the alternate function, but do not have hysteresis characteristics in the port mode.**

**(1)  Port 0 register (P0)**

**(a)  V850ES/JG3-U**

After reset: 00H (output latch)      R/W      Address:  FFFFF400H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P0 | 0 | 0 | 0 | 0 | P03 | P02 | 0 | 0 |

| P0n | Output data control (in output mode) (n = 2, 3) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(b)  V850ES/JH3-U**

After reset: 00H (output latch)      R/W      Address:  FFFFF400H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P0 | 0 | 0 | P05 | P04 | P03 | P02 | P01 | P00 |

| P0n | Output data control (in output mode) (n = 2 to 5) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(2) Port 0 mode register (PM0)**

**(a) V850ES/JG3-U**

After reset: FFH     R/W     Address:  FFFFF420H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|------|------|---|---|
| PM0 | 1 | 1 | 1 | 1 | PM03 | PM02 | 1 | 1 |

| PM0n | I/O mode control (n = 2, 3) |
|------|------------------------------|
| 0 | Output mode |
| 1 | Input mode |

**(b) V850ES/JH3-U**

After reset: FFH     R/W     Address:  FFFFF420H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|------|------|------|------|------|------|
| PM0 | 1 | 1 | PM05 | PM04 | PM03 | PM02 | PM01 | PM00 |

| PM0n | I/O mode control (n = 0 to 5) |
|------|-------------------------------|
| 0 | Output mode |
| 1 | Input mode |

**(3)  Port 0 mode control register (PMC0)**

**(a)  V850ES/JG3-U**

After reset: 00H      R/W      Address:  FFFFF440H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC0 | 0 | 0 | 0 | 0 | PMC03 | PMC02 | 0 | 0 |

| PMC03 | Specification of P03 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | INTP02 input/ADTRG input/UCLK input |

| PMC02 | Specification of P02 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | NMI input |

**(b)  V850ES/JH3-U**

After reset: 00H      R/W      Address:  FFFFF440H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC0 | 0 | 0 | PMC05 | PMC04 | PMC03 | PMC02 | PMC01 | PMC00 |

| PMC05 | Specification of P05 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | INTP04 input |

| PMC04 | Specification of P04 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | INTP03 input |

| PMC03 | Specification of P03 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | INTP02 input/ADTRG input/UCLK input |

| PMC02 | Specification of P02 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | NMI input |

| PMC01 | Specification of P01 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | INTP01 input |

| PMC00 | Specification of P00 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | INTP00 input |

**(4)  Port 0 function control register (PFC0)**

After reset:  00H      R/W      Address:  FFFFF460H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC0 | 0 | 0 | 0 | 0 | PFC03 | 0 | 0 | 0 |

**Remark**  For details of alternate function specification, see **4.3.1 (6)  Port 0 alternate function specifications**.

**(5)  Port 0 function control expansion register (PFCE0)**

After reset:  00H      R/W      Address:  FFFFF700H

|       | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|---|---|---|---|
| PFCE0 | 0 | 0 | 0 | 0 | PFCE03 | 0 | 0 | 0 |

**Remark**   For details of alternate function specification, see **4.3.1 (6)  Port 0 alternate function specifications**.

**(6)  Port 0 alternate function specifications**

| PFCE03 | PFC03 | Specification of P03 pin alternate function |
|--------|-------|---------------------------------------------|
| 0 | 0 | INTP02 input |
| 0 | 1 | ADTRG input |
| 1 | 0 | UCLK input |
| 1 | 1 | Setting prohibited |

**(7)  Port 0 function register (PF0)**

**(a)  V850ES/JG3-U**

After reset:  00H      R/W      Address:  FFFFFC60H

|     | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|---|---|---|---|---|---|---|---|
| PF0 | 0 | 0 | 0 | 0 | PF03 | PF02 | 0 | 0 |

| PF0n | Control of normal output or N-ch open-drain output (n = 2, 3) |
|------|---------------------------------------------------------------|
| 0 | Normal output |
| 1 | N-ch open-drain output |

**(b)  V850ES/JH3-U**

After reset:  00H      R/W      Address:  FFFFFC60H

|     | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|---|---|---|---|---|---|---|---|
| PF0 | 0 | 0 | PF05 | PF04 | PF03 | PF02 | PF01 | PF00 |

| PF0n | Control of normal output or N-ch open-drain output (n = 0 to 5) |
|------|-----------------------------------------------------------------|
| 0 | Normal output |
| 1 | N-ch open-drain output |

#### 4.3.2  Port 1

Port 1 is a 2-bit port for which I/O settings can be controlled in 1-bit units.

Port 1 includes the following alternate-function pins.

**Table 4-7.  Port 1 Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| P10 | 3 | 3 | ANO0 | Output | – |
| P11 | 4 | 4 | ANO1 | Output | |

**Caution    When the power is turned on, the P10 and P11 pins may output an undefined level temporarily even during reset.**

**(1)  Port 1 register (P1)**

After reset:  00H (output latch)      R/W      Address:  FFFFF402H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P1 | 0 | 0 | 0 | 0 | 0 | 0 | P11 | P10 |

| P1n | Output data control (in output mode) (n = 0, 1) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**Caution    Do not read or write the P1 register during D/A conversion (see 16.4.3  Cautions).**

**(2)  Port 1 mode register (PM1)**

After reset:  FFH      R/W      Address:  FFFFF422H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM1 | 1 | 1 | 1 | 1 | 1 | 1 | PM11 | PM10 |

| PM1n | I/O mode control (n = 0, 1) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**Cautions 1.  When using P1n as the alternate function (ANOn pin output), set the PM1n bit to 1.**

**2.  When using one of the PM10 and PM11 pins as an I/O port and the other as a D/A output pin, do so in an application where the port I/O level does not change during D/A output.**

### 4.3.3 Port 2 (V850ES/JH3-U only)

Port 2 is a 6-bit port for which I/O settings can be controlled in 1-bit units.

Port 2 includes the following alternate-function pins.

**Table 4-8. Port 2 Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|----------|---------|---------|------------------------------|-----|--------|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| P20 | – | 32 | TIAB03/KR2/TOAB03/RTP02 | I/O | Selectable as N-ch open-drain output |
| P21 | – | 33 | SIF2/KR3/TIAB00/TOAB00 /RTP03 | I/O | |
| P22 | – | 34 | SOF2/KR4/RTP04 | I/O | |
| P23 | – | 35 | SCKF2/KR5/RTP05 | I/O | |
| P24 | – | 36 | INTP05 | Input | |
| P25 | – | 28 | INTP06 | Input | |

**Caution   The P20 to P25 pins have hysteresis characteristics in the input mode of the alternate-function pin, but do not have the hysteresis characteristics in the port mode.**

### (1) Port 2 register (P2)

After reset: 00H (output latch)     R/W     Address: FFFFF404H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|---|---|-----|-----|-----|-----|-----|-----|
| P2 | 0 | 0 | P25 | P24 | P23 | P22 | P21 | P20 |

| P2n | Output data control (in output mode) (n = 0 to 5) |
|-----|---------------------------------------------------|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

### (2) Port 2 mode register (PM2)

After reset: FFH     R/W     Address: FFFFF424H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|---|---|------|------|------|------|------|------|
| PM2 | 1 | 1 | PM25 | PM24 | PM23 | PM22 | PM21 | PM20 |

| PM2n | I/O mode control (n = 0 to 5) |
|------|-------------------------------|
| 0 | Output mode |
| 1 | Input mode |

**(3)  Port 2 mode control register (PMC2)**

After reset:  00H     R/W     Address:  FFFFF444H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC2 | 0 | 0 | PMC25 | PMC24 | PMC23 | PMC22 | PMC21 | PMC20 |

| PMC25 | Specification of P25 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | INTP06 input |

| PMC24 | Specification of P24 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | INTP05 input |

| PMC23 | Specification of P23 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{SCKF2}}$ I/O/KR5 input/RTP05 output |

| PMC22 | Specification of P22 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SOF2 output/KR4 input/RTP04 output |

| PMC21 | Specification of P21 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SIF2 output/KR3 input/TIAB00 input/TOAB00 output/RTP03 output |

| PMC20 | Specification of P20 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAB03 input/KR2 input/TOAB03 output/RTP02 output |

**(4)  Port 2 function control register (PFC2)**

After reset:  00H     R/W     Address:  FFFFF464H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC2 | 0 | 0 | 0 | 0 | PFC23 | PFC22 | PFC21 | PFC20 |

**Remark**   For details of alternate function specification, see **4.3.3 (6)  Port 2 alternate function specifications**.

**(5)  Port 2 function control expansion register (PFCE2)**

After reset:  00H      R/W      Address:  FFFFF704H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE2 | 0 | 0 | 0 | 0 | PFCE23 | PFCE22 | PFCE21 | PFCE20 |

**Remark**   For details of alternate function specification, see **4.3.3 (6)  Port 2 alternate function specifications**.

**(6)  Port 2 alternate function specifications**

| PFCE23 | PFC23 | Specification of P23 pin alternate function |
|---|---|---|
| 0 | 0 | SCKF2 I/O |
| 0 | 1 | KR5 input |
| 1 | 0 | RTP05 output |
| 1 | 1 | Setting prohibited |

| PFCE22 | PFC22 | Specification of P22 pin alternate function |
|---|---|---|
| 0 | 0 | SOF2 output |
| 0 | 1 | KR4 input |
| 1 | 0 | RTP04 output |
| 1 | 1 | Setting prohibited |

| PFCE21 | PFC21 | Specification of P21 pin alternate function |
|---|---|---|
| 0 | 0 | SIF2 input |
| 0 | 1 | KR3 input/TIAB00 input[Note] |
| 1 | 0 | TOAB00 output |
| 1 | 1 | RTP03 output |

**Note**   KR3 and TIAB00 are alternate functions.  When using the pin as the TIAB00 pin, disable the KR3 pin key return detection, which is the alternate function (clear the KRM.KRM3 bit to 0).  Also, when using the pin as the KRn pin, disable TIAB00 pin edge detection, which is the alternate function (TAB0IOC1.TAB0TIG0, TAB0TIG1 bit = 00 B, TAB0IOC2 register = 00H).

| PFCE20 | PFC20 | Specification of P20 pin alternate function |
|---|---|---|
| 0 | 0 | TIAB03 input/KR2 input |
| 0 | 1 | KR2 input |
| 1 | 0 | TOAB03 output |
| 1 | 1 | Setting prohibited |

**(7) Port 2 function register (PF2)**

After reset:  00H     R/W     Address:  FFFFFC64H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PF2 | 0 | 0 | PF25 | PF24 | PF23 | PF22 | PF21 | PF20 |

| PF2n | Control of normal output or N-ch open-drain output (n = 0 to 5) |
|---|---|
| 0 | Normal output |
| 1 | N-ch open-drain output |

### 4.3.4 Port 3

Port 3 is a 10-bit port that controls I/O in 1-bit units.

Port 3 includes the following alternate-function pins.

**Table 4-9. Port 3 Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| P30 | 25 | 37 | TXDC0/SOF4/INTP07 | I/O | Selectable as N-ch open-drain output |
| P31 | 26 | 38 | RXDC0/SIF4/INTP08 | Input | |
| P32 | 27 | 39 | ASCKC0/SCKF4/TIAA00/TOAA00 | I/O | |
| P33 | 28 | 40 | TIAA01/TOAA01/RTCDIV/RTCCL | I/O | |
| P34 | 29 | 41 | TIAA10/TOAA10/TOAA1OFF/INTP09 /PPON | I/O | |
| P35 | 30 | 42 | TIAA11/TOAA11/RTC1HZ/$\overline{\text{OCI}}$ | I/O | |
| P36 | 31 | 43 | TXDC3/SCL00/$\overline{\text{UDMARQ0}}$ | I/O | |
| P37 | 32 | 44 | RXDC3/SDA00/$\overline{\text{UDMAAK0}}$ | I/O | |

**Caution  The P30 to P37 pins have hysteresis characteristics in the input mode of the alternate-function pin, but do not have the hysteresis characteristics in the port mode.**

### (1) Port 3 register (P3)

After reset: 00H (output latch)    R/W    Address: FFFFF406H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P3 | P37 | P36 | P35 | P34 | P33 | P32 | P31 | P30 |

| P3n | Output data control (in output mode) (n = 0 to 7) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

### (2) Port 3 mode register (PM3)

After reset: FFH    R/W    Address: FFFFF426H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM3 | PM37 | PM36 | PM35 | PM34 | PM33 | PM32 | PM31 | PM30 |

| PM3n | I/O mode control (n = 0 to 7) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(3) Port 3 mode control register (PMC3)**

After reset: 00H     R/W     Address: FFFFF446H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC3 | PMC37 | PMC36 | PMC35 | PMC34 | PMC33 | PMC32 | PMC31 | PMC30 |

| PMC37 | Specification of P37 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | RXDC3 input/SDA00 I/O/$\overline{\text{UDMAAK0}}$ output |

| PMC36 | Specification of P36 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TXDC3 output/SCL00 I/O/$\overline{\text{UDMARQ0}}$ input |

| PMC35 | Specification of P35 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAA11 input/TOAA11 output/RTC1HZ output/$\overline{\text{OCI}}$ input |

| PMC34 | Specification of P34 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAA10 input/TOAA10 output/TOAA1OFF input/INTP09 input/PPON output |

| PMC33 | Specification of P33 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAA01 input/TOAA01 output/RTCDIV output/RTCCL output |

| PMC32 | Specification of P32 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | ASCKA0 input/$\overline{\text{SCKF4}}$ I/O/TIAA00 input/TOAA00 output |

| PMC31 | Specification of P31 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | RXDC0 input/SIF4 input/INTP08 input |

| PMC30 | Specification of P30 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TXDC0 output/SOF4 output/INTP07 input |

**(4) Port 3 function control register (PFC3)**

After reset: 00H    R/W    Address:    FFFFF466H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC3 | PFC37 | PFC36 | PFC35 | PFC34 | PFC33 | PFC32 | PFC31 | PFC30 |

**Remark** For details of alternate function specification, see **4.3.4 (6)  Port 3 alternate function specifications**.

**(5) Port 3 function control expansion register (PFCE3)**

After reset: 00H    R/W    Address: FFFFF706H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE3 | PFCE37 | PFCE36 | PFCE35 | PFCE34 | PFCE33 | PFCE32 | PFCE31 | PFCE30 |

**Remark** For details of alternate function specification, see **4.3.4 (6)  Port 3 alternate function specifications**.

**(6) Port 3 alternate function specifications**

| PFCE37 | PFC37 | Specification of P37 pin alternate function |
|---|---|---|
| 0 | 0 | RXDC3 input |
| 0 | 1 | SDA00 I/O |
| 1 | 0 | Setting prohibited |
| 1 | 1 | $\overline{\text{UDMAAK0}}$ output |

| PFCE36 | PFC36 | Specification of P36 pin alternate function |
|---|---|---|
| 0 | 0 | TXDC3 output |
| 0 | 1 | SCL00 I/O |
| 1 | 0 | Setting prohibited |
| 1 | 1 | $\overline{\text{UDMARQ0}}$ input |

| PFCE35 | PFC35 | Specification of P35 pin alternate function |
|---|---|---|
| 0 | 0 | TIAA11 input |
| 0 | 1 | TOAA11 output |
| 1 | 0 | RTC1HZ output |
| 1 | 1 | $\overline{\text{OCI}}$ input |

| PFCE34 | PFC34 | Specification of P34 pin alternate function |
|--------|-------|---------------------------------------------|
| 0 | 0 | TIAA10 input |
| 0 | 1 | TOAA10 output |
| 1 | 0 | TOAA1OFF input/INTP09 input[Note] |
| 1 | 1 | PPON output |

**Note** TOAA1OFF and INTP09 are alternate functions. When using the pin as the TOAA1OFF pin, disable INTP09 pin edge detection, which is the alternate function. Also, when using the pin as the INTP09 pin, stop the high-impedance output controller.

| PFCE33 | PFC33 | Specification of P33 pin alternate function |
|--------|-------|---------------------------------------------|
| 0 | 0 | TIAA01 iput |
| 0 | 1 | TOAA01 output |
| 1 | 0 | RTCDIV output |
| 1 | 1 | RTCCL output |

| PFCE32 | PFC32 | Specification of P32 pin alternate function |
|--------|-------|---------------------------------------------|
| 0 | 0 | ASCKC0 input |
| 0 | 1 | $\overline{\text{SCKF4}}$ I/O |
| 1 | 0 | TIAA00 input |
| 1 | 1 | TOAA00 output |

| PFCE31 | PFC31 | Specification of P31 pin alternate function |
|--------|-------|---------------------------------------------|
| 0 | 0 | RXDC0 input |
| 0 | 1 | SIF4 input |
| 1 | 0 | INTP08 input |
| 1 | 1 | Setting prohibited |

| PFCE30 | PFC30 | Specification of P30 pin alternate function |
|--------|-------|---------------------------------------------|
| 0 | 0 | TXDC0 output |
| 0 | 1 | SOF4 output |
| 1 | 0 | INTP07 input |
| 1 | 1 | Setting prohibited |

**(7) Port 3 function register (PF3)**

After reset: 00H     R/W     Address:     FFFFFC66H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PF3 | PF37 | PF36 | PF35 | PF34 | PF33 | PF32 | PF31 | PF30 |

| PF3n | Control of normal output or N-ch open-drain output (n = 0 to 7) |
|---|---|
| 0 | Normal output (CMOS output) |
| 1 | N-ch open-drain output |

### 4.3.5  Port 4

Port 4 is a 3-bit port that controls I/O in 1-bit units.

Port 4 includes the following alternate-function pins.

**Table 4-10.  Port 4 Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|----------|---------|---------|-----------------------------|-----|--------|
|          | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| P40 | 22 | 29 | SIF0/TXDC4/SDA01 | I/O | Selectable as N-ch open-drain output |
| P41 | 23 | 30 | SOF0/RXDC4/SCL01 | I/O | |
| P42 | 24 | 31 | $\overline{\text{SCKF0}}$/INTP10 | I/O | |

**Caution**   **The P40 to P42 pins have hysteresis characteristics in the input mode of the alternate-function pin, but do not have the hysteresis characteristics in the port mode.**

### (1)  Port 4 register (P4)

After reset:  00H (output latch)     R/W     Address:  FFFFF408H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P4 | 0 | 0 | 0 | 0 | 0 | P42 | P41 | P40 |

| P4n | Output data control (in output mode) (n = 0 to 2) |
|-----|---------------------------------------------------|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

### (2)  Port 4 mode register (PM4)

After reset:  FFH     R/W     Address:  FFFFF428H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM4 | 1 | 1 | 1 | 1 | 1 | PM42 | PM41 | PM40 |

| PM4n | I/O mode control (n = 0 to 2) |
|------|-------------------------------|
| 0 | Output mode |
| 1 | Input mode |

**(3)  Port 4 mode control register (PMC4)**

After reset:  00H      R/W      Address:  FFFFF448H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC4 | 0 | 0 | 0 | 0 | 0 | PMC42 | PMC41 | PMC40 |

| PMC42 | Specification of P42 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{SCKF0}}$ I/O/INTP10 input |

| PMC41 | Specification of P41 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SOF0 output/RXDC4 input/SCL01 I/O |

| PMC40 | Specification of P40 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SIF0 input/TXDC4 output/SDA01 I/O |

**(4)  Port 4 function control register (PFC4)**

After reset:  00H      R/W      Address:  FFFFF468H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC4 | 0 | 0 | 0 | 0 | 0 | PFC42 | PFC41 | PFC40 |

**Remark**   For details of alternate function specification, see **4.3.5 (6)  Port 4 alternate function specifications**.

**(5)  Port 4 function control expansion register (PFCE4)**

After reset:  00H      R/W      Address:  FFFFF708H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE4 | 0 | 0 | 0 | 0 | 0 | 0 | PFCE41 | PFCE40 |

**Remark**   For details of alternate function specification, see **4.3.5 (6)  Port 4 alternate function specifications**.

**(6) Port 4 alternate function specifications**

| PFC42 | Specification of P42 pin alternate function |
|-------|----------------------------------------------|
| 0 | $\overline{\text{SCKF0}}$ I/O |
| 1 | INTP10 input |

| PFCE41 | PFC41 | Specification of P41 pin alternate function |
|--------|-------|----------------------------------------------|
| 0 | 0 | SOF0 output |
| 0 | 1 | RXDC4 input |
| 1 | 0 | SCL01 I/O |
| 1 | 1 | Setting prohibited |

| PFCE40 | PFC40 | Specification of P40 pin alternate function |
|--------|-------|----------------------------------------------|
| 0 | 0 | SIF0 input |
| 0 | 1 | TXDC4 output |
| 1 | 0 | SDA01 I/O |
| 1 | 1 | Setting prohibited |

**(5) Port 4 function register (PF4)**

After reset:  00H     R/W     Address:  FFFFFC68H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|---|------|------|------|
| PF4 | 0 | 0 | 0 | 0 | 0 | PF42 | PF41 | PF40 |

| PF4n | Control of normal output or N-ch open-drain output (n = 0 to 2) |
|------|-----------------------------------------------------------------|
| 0 | Normal output (CMOS output) |
| 1 | N-ch open-drain output |

### 4.3.6 Port 5

Port 5 is 6-bit (V850ES/JG3-U)/2-bit (V850ES/JH3-U) port that controls I/O in 1-bit units.

Port 5 includes the following alternate-function pins.

**Table 4-11. Port 5 Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| P50 | 35 | 47 | TIAB01/KR0/TOAB01/RTP00 /$\overline{\text{UDMARQ1}}$ | I/O | Selectable as N-ch open-drain output |
| P51 | 36 | 48 | TIAB02/KR1/TOAB02/RTP01 /$\overline{\text{UDMAAK1}}$ | I/O | |
| P52 | 37 | – | TIAB03/KR2/TOAB13/RTP02 /DDI[Note] | I/O | |
| P53 | 38 | – | SIF2/TIAB00/KR3/TOAB10 /RTP03/DDO[Note] | I/O | |
| P54 | 39 | – | SOF2/KR4/RTP04/DCK[Note] | I/O | |
| P55 | 40 | – | SCKF2/KR5/RTP05/DMS[Note] | I/O | |
| P56 | 41 | – | INTP05/$\overline{\text{DRST}}$[Note] | Input | |

**Note** The DDI, DDO, DCK, DMS, and $\overline{\text{DRST}}$ pins are used for on-chip debugging.

If on-chip debugging is not used, fix the P05/INTP02/$\overline{\text{DRST}}$ pin to low level between when the reset by the $\overline{\text{RESET}}$ pin is released and when the OCDM.OCDM0 bit is cleared (0).

For details, see **4.5.3 Cautions on on-chip debug pins**.

**Cautions 1. When the power is turned on, the P53 pin may output an undefined level temporarily even during reset.**

**2. The P50 to P56 pins have hysteresis characteristics in the input mode of the alternate-function pin, but do not have the hysteresis characteristics in the port mode.**

**(1) Port 5 register (P5)**

**(a) V850ES/JG3-U**

After reset:  00H (output latch)      R/W      Address:  FFFFF40AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P5 | 0 | P56 | P55 | P54 | P53 | P52 | P51 | P50 |

| P5n | Output data control (in output mode) (n = 0 to 6) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(b) V850ES/JH3-U**

After reset:  00H (output latch)      R/W      Address:  FFFFF40AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P5 | 0 | 0 | 0 | 0 | 0 | 0 | P51 | P50 |

| P5n | Output data control (in output mode) (n = 0, 1) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(2) Port 5 mode register (PM5)**

**(a) V850ES/JG3-U**

After reset:  FFH     R/W     Address:  FFFFF42AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM5 | 1 | PM56 | PM55 | PM54 | PM53 | PM52 | PM51 | PM50 |

| PM5n | I/O mode control (n = 0 to 6) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(b) V850ES/JH3-U**

After reset:  FFH     R/W     Address:  FFFFF42AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM5 | 1 | 1 | 1 | 1 | 1 | 1 | PM51 | PM50 |

| PM5n | I/O mode control (n = 0, 1) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(3)  Port 5 mode control register (PMC5)**

**(a)  V850ES/JG3-U**

After reset:  00H    R/W    Address:  FFFFF44AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC5 | 0 | PMC56 | PMC55 | PMC54 | PMC53 | PMC52 | PMC51 | PMC50 |

| PMC56 | Specification of P56 pin operation mode |
|---|---|
| 1 | I/O port |
| | INTP05 input |

| PMC55 | Specification of P55 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{SCKF2}}$ I/O/KR5 input/RTP05 output |

| PMC54 | Specification of P54 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SOF2 output/KR4 input/RTP04 output |

| PMC53 | Specification of P53 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SIF2 input/KR3 input/TIAB00 input/TOAB00 output/RTP03 output |

| PMC52 | Specification of P52 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAB03 input/KR2 input/TOAB03 output/RTP02 output |

| PMC51 | Specification of P51 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAB02 input/KR1 input/TOAB02 output/RTP01 output/$\overline{\text{UDMAAK1}}$ output |

| PMC50 | Specification of P50 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAB01 input/KR0 input/TOAB01 output/RTP00 output/$\overline{\text{UDMARQ1}}$ input |

**(b)  V850ES/JH3-U**

After reset:  00H    R/W    Address:  FFFFF44AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC5 | 0 | 0 | 0 | 0 | 0 | 0 | PMC51 | PMC50 |

| PMC51 | Specification of P51 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAB02 input/KR1 input/TOAB02 output/RTP01 output/$\overline{\text{UDMAAK1}}$ output |

| PMC50 | Specification of P50 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAB01 input/KR0 input/TOAB01 output/RTP00 output/$\overline{\text{UDMARQ1}}$ input |

**(4)  Port 5 function control register (PFC5)**

**(a)  V850ES/JG3-U**

After reset:  00H     R/W     Address:  FFFFF46AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC5 | 0 | 0 | PFC55 | PFC54 | PFC53 | PFC52 | PFC51 | PFC50 |

**(b)  V850ES/JH3-U**

After reset:  00H     R/W     Address:  FFFFF46AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC5 | 0 | 0 | 0 | 0 | 0 | 0 | PFC51 | PFC50 |

**Remark**   For details of alternate function specification, see **4.3.6 (6)  Port 5 alternate function specifications**.

**(5)  Port 5 function control expansion register (PFCE5)**

**(a)  V850ES/JG3-U**

After reset:  00H     R/W     Address:  FFFFF70AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE5 | 0 | 0 | PFCE55 | PFCE54 | PFCE53 | PFCE52 | PFCE51 | PFCE50 |

**(b)  V850ES/JH3-U**

After reset:  00H     R/W     Address:  FFFFF70AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE5 | 0 | 0 | 0 | 0 | 0 | 0 | PFCE51 | PFCE50 |

**Remark**   For details of alternate function specification, see **4.3.6 (6)  Port 5 alternate function specifications**.

**(6)  Port 5 alternate function specifications**

| PFCE55[Note 1] | PFC55[Note 1] | Specification of P55 pin alternate function[Note 1] |
|---|---|---|
| 0 | 0 | $\overline{\text{SCKF2}}$ I/O |
| 0 | 1 | KR5 input |
| 1 | 0 | RTP05 output |
| 1 | 1 | Setting prohibited |

| PFCE54[Note 1] | PFC54[Note 1] | Specification of P54 pin alternate function[Note 1] |
|---|---|---|
| 0 | 0 | SOF2 output |
| 0 | 1 | KR4 input |
| 1 | 0 | RTP04 output |
| 1 | 1 | Setting prohibited |

| PFCE53[Note 1] | PFC53[Note 1] | Specification of P53 pin alternate function[Note 1] |
|---|---|---|
| 0 | 0 | SIF2 input |
| 0 | 1 | TIAB00 input/KR3[Note 2] input |
| 1 | 0 | TOAB00 output |
| 1 | 1 | RTP03 output |

| PFCE52[Note1] | PFC52[Note1] | Specification of P52 pin alternate function[Note 1] |
|---|---|---|
| 0 | 0 | TIAB03 input/KR2[Note 2] input |
| 0 | 1 | TOAB03 output |
| 1 | 0 | RTP02 output |
| 1 | 1 | Setting prohibited |

| PFCE51 | PFC51 | Specification of P51 pin alternate function |
|---|---|---|
| 0 | 0 | TIAB02 input/KR1[Note 2] input |
| 0 | 1 | TOAB02 output |
| 1 | 0 | RTP01 output |
| 1 | 1 | $\overline{\text{UDMAAK1}}$ output |

| PFCE50 | PFC50 | Specification of P50 pin alternate function |
|---|---|---|
| 0 | 0 | TIAB01 input/KR0[Note 2] input |
| 0 | 1 | TOAB01 output |
| 1 | 0 | RTP00 output |
| 1 | 1 | $\overline{\text{UDMARQ1}}$ input |

**Notes 1.** V850ES/JG3-U only

2. KRn and TIAB0m are alternate functions.  When using the pin as the TIAB0m pin, disable KRn pin key return detection, which is the alternate function (clear the KRM.KRMn bit to 0).  Also, when using the pin as the KRn pin, disable TIAB0m pin edge detection, which is the alternate function (n = 0 to 3, m = 0 to 3).

| Pin Name | Use as TIAB0m Pin | Using as KRn Pin |
|---|---|---|
| KR0/TIAB01 | KRM.KRM0 bit = 0 | TAB0IOC1.TAB0TIG2, TAB0TIG3 bits = 0 |
| KR1/TIAB02 | KRM.KRM1 bit = 0 | TAB0IOC1.TAB0TIG4, TAB0TIG5 bits = 0 |
| KR2/TIAB03 | KRM.KRM2 bit = 0 | TAB0IOC1.TAB0TIG6, TAB0TIG7 bits = 0 |
| KR3/TIAB00 | KRM.KRM3 bit = 0 | TAB0IOC1.TAB0TIG0, TAB0TIG1 bits = 0<br>TAB0IOC2.TAB0EES0, TAB0EES1 bits = 0<br>TAB0IOC2.TAB0ETS0, TAB0ETS1 bits = 0 |

**(7) Port 5 function register (PF5)**

**(a) V850ES/JG3-U**

After reset: 00H    R/W    Address: FFFFFC6AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PF5 | 0 | PF56 | PF55 | PF54 | PF53 | PF52 | PF51 | PF50 |

| PF5n | Control of normal output or N-ch open-drain output (n = 0 to 6) |
|---|---|
| 0 | Normal output (CMOS output) |
| 1 | N-ch open-drain output |

**(b) V850ES/JH3-U**

After reset: 00H    R/W    Address: FFFFFC6AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PF5 | 0 | 0 | 0 | 0 | 0 | 0 | PF51 | PF50 |

| PF5n | Control of normal output or N-ch open-drain output (n = 0, 1) |
|---|---|
| 0 | Normal output (CMOS output) |
| 1 | N-ch open-drain output |

### 4.3.7 Port 6

Port 6 is a 6-bit port for which I/O settings can be controlled in 1-bit units.

Port 6 includes the following alternate-function pins.

**Table 4-12. Port 6 Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| P60 | 65 | 90 | TOAB1T1/TOAB11/TIAB11 /$\overline{\text{WAIT}}$[Note] | I/O | – |
| P61 | 66 | 91 | TOAB1B1/TOAB10/TIAB10 /$\overline{\text{RD}}$[Note] | I/O | |
| P62 | 67 | 92 | TOAB1T2/TOAB12/TIAB12 /ASTB[Note] | I/O | |
| P63 | 68 | 93 | TOAB1B2/TRGAB1/$\overline{\text{CS0}}$[Note] | I/O | |
| P64 | 69 | 94 | TOAB1T3/TOAB13/TIAB13 /$\overline{\text{CS2}}$[Note] | I/O | |
| P65 | 70 | 95 | TOAB1B3/EVTAB1/$\overline{\text{CS3}}$[Note] | I/O | |

**Note** V850ES/JG3-U only

**Caution** **The P60 to P65 pins have hysteresis characteristics in the input mode of the alternate-function pin, but do not have the hysteresis characteristics in the port mode.**

### (1) Port 6 register (P6)

After reset: 00H (output latch)     R/W     Address: FFFFF40CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P6 | 0 | 0 | P65 | P64 | P63 | P62 | P61 | P60 |

| P6n | Output data control (in output mode) (n = 0 to 5) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(2)  Port 6 mode register (PM6)**

After reset:  FFH      R/W      Address:  FFFFF42CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM6 | 1 | 1 | PM65 | PM64 | PM63 | PM62 | PM61 | PM60 |

| PM6n | Output data control (in output mode) (n = 0 to 5) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(3)  Port 6 mode control register (PMC6)**

After reset:  00H      R/W      Address:  FFFFF44CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC6 | 0 | 0 | PMC65 | PMC64 | PMC63 | PMC62 | PMC61 | PMC60 |

| PMC65 | Specification of P65 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TOAB1B3 output/EVTAB1 input/$\overline{\text{CS3}}$ output[Note] |

| PMC64 | Specification of P64 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TOAB1T3 output/TOAB13 output/TIAB13 input/$\overline{\text{CS2}}$ output[Note] |

| PMC63 | Specification of P63 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TOAB1B2 output/TRGAB1 input/$\overline{\text{CS0}}$ output[Note] |

| PMC62 | Specification of P62 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TOAB1T2 output/TOAB12 output/TIAB12 output/ASTB output[Note] |

| PMC61 | Specification of P61 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TOAB1B1 output/TIAB10 input/TOAB10 output/$\overline{\text{RD}}$ output[Note] |

| PMC60 | Specification of P60 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TOAB1T1 output/TOAB11 output/TIAB11 input/$\overline{\text{WAIT}}$ output[Note] |

**Note**  V850ES/JG3-U only

**(4)  Port 6 function control register (PFC6)**

After reset:  00H      R/W      Address:  FFFFF46CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC6 | 0 | 0 | PFC65 | PFC64 | PFC63 | PFC62 | PFC61 | PFC60 |

**Remark**  For details of alternate function specification, see **4.3.7 (6)  Port 6 alternate function specifications**.

**(5)  Port 6 function control expansion register (PFCE6)**

**(a)  V850ES/JG3-U**

After reset:  00H      R/W      Address:  FFFFF70CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE6 | 0 | 0 | PFCE65 | PFCE64 | PFCE63 | PFCE62 | PFCE61 | PFCE60 |

**(b)  V850ES/JH3-U**

After reset:  00H      R/W      Address:  FFFFF70CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE6 | 0 | 0 | 0 | 0 | 0 | 0 | PFCE61 | 0 |

**Remark**  For details of alternate function specification, see **4.3.7 (6)  Port 6 alternate function specifications**.

**(6)  Port 6 alternate function specifications**

| PFCE65[Note] | PFC65 | Specification of P65 pin alternate function |
|---|---|---|
| 0 | 0 | TOAB1B3 output |
| 0 | 1 | EVTAB1 input |
| 1 | 0 | $\overline{CS3}$ output[Note] |
| 1 | 1 | Setting prohibited[Note] |

| PFCE64[Note] | PFC64 | Specification of P64 pin alternate function |
|---|---|---|
| 0 | 0 | TOAB1T3 output/TOAB13 output |
| 0 | 1 | TIAB13 input |
| 1 | 0 | $\overline{CS2}$ output[Note] |
| 1 | 1 | Setting prohibited[Note] |

| PFCE63[Note] | PFC63 | Specification of P63 pin alternate function |
|---|---|---|
| 0 | 0 | TOAB1B2 output |
| 0 | 1 | TRGAB1 input |
| 1 | 0 | $\overline{CS0}$ output[Note] |
| 1 | 1 | Setting prohibited[Note] |

| PFCE62[Note] | PFC62 | Specification of P62 pin alternate function |
|---|---|---|
| 0 | 0 | TOAB1T2 output/TOAB12 output |
| 0 | 1 | TIAB12 input |
| 1 | 0 | ASTB output[Note] |
| 1 | 1 | Setting prohibited[Note] |

| PFCE61 | PFC61 | Specification of P61 pin alternate function |
|---|---|---|
| 0 | 0 | TOAB1B1 output |
| 0 | 1 | TIAB10 input |
| 1 | 0 | TOAB10 output |
| 1 | 1 | $\overline{RD}$ output (V850ES/JG3-U)<br>Setting prohibited (V850ES/JH3-U) |

| PFCE60[Note] | PFC60 | Specification of P60 pin alternate function |
|---|---|---|
| 0 | 0 | TOAB1T1 output/TOAB11 output |
| 0 | 1 | TIAB11 input |
| 1 | 0 | $\overline{WAIT}$ output[Note] |
| 1 | 1 | Setting prohibited[Note] |

**Note**  V850ES/JG3-U only

## 4.3.8  Port 7

Port 7 is a 12-bit port for which I/O settings can be controlled in 1-bit units.

Port 7 includes the following alternate-function pins.

**Table 4-13.  Port 7 Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| P70 | 100 | 128 | ANI0 | Input | – |
| P71 | 99 | 127 | ANI1 | Input | |
| P72 | 98 | 126 | ANI2 | Input | |
| P73 | 97 | 125 | ANI3 | Input | |
| P74 | 96 | 124 | ANI4 | Input | |
| P77 | 95 | 123 | ANI5 | Input | |
| P76 | 94 | 122 | ANI6 | Input | |
| P77 | 93 | 121 | ANI7 | Input | |
| P78 | 92 | 120 | ANI8 | Input | |
| P79 | 91 | 119 | ANI9 | Input | |
| P710 | 90 | 118 | ANI10 | Input | |
| P711 | 89 | 117 | ANI11 | Input | |

### (1)  Port 7 register H, port 7 register L (P7H, P7L)

After reset: 00H (output latch)       R/W       Address:  P7L FFFFF40EH, P7H FFFFF40FH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P7H | 0 | 0 | 0 | 0 | P711 | P710 | P79 | P78 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P7L | P77 | P76 | P75 | P74 | P73 | P72 | P71 | P70 |

| P7n | Output data control (in output mode) (n = 0 to 11) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**Caution**   **Do not read or write the P7H and P7L registers during A/D conversion (see 15.6 (4) Alternate I/O).**

**Remark**   These registers cannot be accessed in 16-bit units as the P7 register.  They can be read or written in 8-bit or 1-bit units as the P7H and P7L registers.

**(2)  Port 7 mode register H, port 7 mode register L (PM7H, PM7L)**

After reset:  FFH     R/W     Address:  PM7L FFFFF42EH, PM7H FFFFF42FH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM7H | 1 | 1 | 1 | 1 | PM711 | PM710 | PM79 | PM78 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM7L | PM77 | PM76 | PM75 | PM74 | PM73 | PM72 | PM71 | PM70 |

| PM7n | I/O mode control (n = 0 to 11) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**Caution   When using the P7n pin as its alternate function (ANIn pin), set the PM7n bit to 1.**

**Remark**   These registers cannot be accessed in 16-bit units as the PM7 register.  They can be read or written in 8-bit or 1-bit units as the PM7H and PM7L registers.

**4.3.9   Port 9**

Port 9 is a 16-bit port for which I/O settings can be controlled in 1-bit units.

Port 9 includes the following alternate-function pins.

**Table 4-14.  Port 9 Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/<br>JG3-U | V850ES/<br>JH3-U | | | |
| P90 | 42 | 54 | KR6/TXDC1/SDA02/A0[Note] | I/O | Selectable as N-ch open-drain output |
| P91 | 43 | 55 | KR7/RXDC1/SCL02/A1[Note] | I/O | |
| P92 | 44 | 56 | TENC01/TIT01/TOT01/A2[Note] | I/O | – |
| P93 | 45 | 57 | TECR0/TIT00/TOT00/A3[Note] | I/O | |
| P94 | 46 | 58 | TIAA31/TOAA31/TENC00<br>/EVTT0/A4[Note] | I/O | |
| P95 | 47 | 59 | TIAA30/TOAA30/A5[Note] | I/O | |
| P96 | 48 | 62 | TIAA21/TOAA21/INTP11/A6[Note] | I/O | |
| P97 | 49 | 63 | SIF1/TIAA20/TOAA20/A7[Note] | I/O | |
| P98 | 50 | 64 | SOF1/INTP12/A8[Note] | I/O | |
| P99 | 51 | 65 | $\overline{SCKF1}$/INTP13/A9[Note] | I/O | |
| P910 | 52 | 66 | SIF3/TXDC2/INTP14/A10[Note] | I/O | |
| P911 | 53 | 67 | SOF3/RXDC2/INTP15/A11[Note] | I/O | |
| P912 | 54 | 68 | $\overline{SCKF3}$/A12[Note] | I/O | |
| P913 | 55 | 69 | TOAB1OFF/INTP16/A13[Note] | I/O | |
| P914 | 56 | 70 | TIAA51/TOAA51/INTP17/A14[Note] | I/O | |
| P915 | 57 | 71 | TIAA50/TOAA50/INTP18/A15[Note] | I/O | |

**Note**   V850ES/JH3-U only

**Caution   The P90 to P915 pins have hysteresis characteristics in the input mode of the alternate-function pin, but do not have the hysteresis characteristics in the port mode.**

**(1)  Port 9 register (P9)**

After reset:  0000H (output latch)      R/W      Address:  P9 FFFFF412H,
                                                            P9L FFFFF412H, P9H FFFFF413H

|  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| P9 (P9H) | P915 | P914 | P913 | P912 | P911 | P910 | P99 | P98 |

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (P9L) | P97 | P96 | P95 | P94 | P93 | P92 | P91 | P90 |

| P9n | Output data control (in output mode) (n = 0 to 15) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**Remarks 1.** The P9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the P9 register as the P9H register and the lower 8 bits as the P9L register, they can be read or written in 8-bit or 1-bit units.
**2.** To read/write bits 8 to 15 of the P9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the P9H register.

**(2)  Port 9 mode register (PM9)**

After reset:  FFFFH      R/W      Address:  PM9 FFFFF432H,
                                            PM9L FFFFF432H, PM9H FFFFF433H

|  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PM9 (PM9H) | PM915 | PM914 | PM913 | PM912 | PM911 | PM910 | PM99 | PM98 |

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (PM9L) | PM97 | PM96 | PM95 | PM94 | PM93 | PM92 | PM91 | PM90 |

| PM9n | I/O mode control (in output mode) (n = 0 to 15) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**Remarks 1.** The PM9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PM9 register as the PM9H register and the lower 8 bits as the PM9L register, they can be read or written in 8-bit and 1-bit units.
**2.** To read/write bits 8 to 15 of the PM9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PM9H register.

**(3) Port 9 mode control register (PMC9)**

After reset: 0000H     R/W     Address: PMC9 FFFFF452H,
                                        PMC9L FFFFF452H, PMC9H FFFFF453H

|  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PMC9 (PMC9H) | PMC915 | PMC914 | PMC913 | PMC912 | PMC911 | PMC910 | PMC99 | PMC98 |

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (PMC9L) | PMC97 | PMC96 | PMC95 | PMC94 | PMC93 | PMC92 | PMC91 | PMC90 |

| PMC915 | Specification of P915 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAA50 input/TOAA50 output/INTP18 input/A15 output[Note] |

| PMC914 | Specification of P914 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAA51 input/TOAA51 output/INTP17 input/A14 output[Note] |

| PMC913 | Specification of P913 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TOAB1OFF input/INTP16 input/A13 output[Note] |

| PMC912 | Specification of P912 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{SCKF3}}$ I/O/A12 output[Note] |

| PMC911 | Specification of P911 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SOF3 output/RXDC2 input/INTP15 input/A11 output[Note] |

| PMC910 | Specification of P910 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SIF3 input/TXDC2 output/INTP14 input/A10 output[Note] |

| PMC99 | Specification of P99 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{SCKF1}}$ I/O/INTP13 input/A9 output[Note] |

| PMC98 | Specification of P98 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SOF1 output/INTP12 input/A8 output[Note] |

**Note**  V850ES/JH3-U only

**Remarks 1.** The PMC9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PMC9 register as the PMC9H register and the lower 8 bits as the PMC9L register, they can be read or written in 8-bit or 1-bit units.
**2.** To read/write bits 8 to 15 of the PMC9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMC9H register.

| PMC97 | Specification of P97 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | SIF1 input/TIAA20 input/TOAA20 output/A7 output[Note] |

| PMC96 | Specification of P96 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAA21 input/TOAA21 output/INTP11 input/A6 output[Note] |

| PMC95 | Specification of P95 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAA30 input/TOAA30 output/A5 output[Note] |

| PMC94 | Specification of P94 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TIAA31 input/TOAA31 output/TENC00 input/EVTT0 input/A4 output[Note] |

| PMC93 | Specification of P93 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TECR0 input/TIT00 input/TOT00 output/A3 output[Note] |

| PMC92 | Specification of P92 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | TENC01 input/TIT01 input/TOT01 output/A2 output[Note] |

| PMC91 | Specification of P91 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | KR7 input/RXDC1 input/SCL02 I/O/A1 output[Note] |

| PMC90 | Specification of P90 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | KR6 input/TXDC1 output/SDA02 I/O/A0 output[Note] |

**Note**   V850ES/JH3-U only

**Caution   When using the A0 to A15 pins as the alternate functions of the P90 to P915 pins, be sure to set all 16 bits of the PMC9 register to FFFFH at once (V850ES/JH3-U only).**

**(4) Port 9 function control register (PFC9)**

**Caution When performing separate address bus output (A0 to A15), set the PMC9 register to FFFFH for all 16 bits at once after setting the PFC9 registers to FCDFH and the PFCE9 registers to CFFFH (V850ES/JH3-U only).**

**(a) V850ES/JG3-U**

After reset: 0000H   R/W   Address: PFC9 FFFFF472H,
PFC9L FFFFF472H, PFC9H FFFFF473H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFC9 (PFC9H) | PFC915 | PFC914 | 0 | 0 | PFC911 | PFC910 | PFC99 | PFC98 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (PFC9L) | PFC97 | PFC96 | PFC95 | PFC94 | PFC93 | PFC92 | PFC91 | PFC90 |

**(b) V850ES/JH3-U**

After reset: 0000H   R/W   Address: PFC9 FFFFF472H,
PFC9L FFFFF472H, PFC9H FFFFF473H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFC9 (PFC9H) | PFC915 | PFC914 | PFC913 | PFC912 | PFC911 | PFC910 | PFC99 | PFC98 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (PFC9L) | PFC97 | PFC96 | PFC95 | PFC94 | PFC93 | PFC92 | PFC91 | PFC90 |

**Remarks 1.** For details of alternate function specification, see **4.3.9 (6) Port 9 alternate function specifications**.

      **2.** The PFC9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PFC9 register as the PFC9H register and the lower 8 bits as the PFC9L register, they can be read or written in 8-bit or 1-bit units.

      **3.** To read/write bits 8 to 15 of the PFC9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PFC9H register.

**(5) Port 9 function control expansion register (PFCE9)**

**Caution   When performing separate address bus output (A0 to A15), set the PMC9 register to FFFFH for all 16 bits at once after setting the PFC9 registers to FCDFH and the PFCE9 registers to CFFFH (V850ES/JH3-U only).**

**(a)  V850ES/JG3-U**

After reset:  0000H     R/W     Address:  PFCE9 FFFFF712H,
                                          PFCE9L FFFFF712H, PFCE9H FFFFF713H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFCE9 (PFCE9H) | PFCE915 | PFCE914 | 0 | 0 | PFCE911 | PFCE910 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (PFCE9L) | PFCE97 | PFCE96 | 0 | PFCE94 | PFCE93 | PFCE92 | PFCE91 | PFCE90 |

**(b)  V850ES/JH3-U**

After reset:  0000H     R/W     Address:  PFCE9 FFFFF712H,
                                          PFCE9L FFFFF712H, PFCE9H FFFFF713H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFCE9 (PFCE9H) | PFCE915 | PFCE914 | 0 | 0 | PFCE911 | PFCE910 | PFCE99 | PFCE98 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (PFCE9L) | PFCE97 | PFCE96 | PFCE95 | PFCE94 | PFCE93 | PFCE92 | PFCE91 | PFCE90 |

**Remarks 1.**  For details of alternate function specification, see **4.3.9 (6)  Port 9 alternate function specifications**.

**2.**  The PFCE9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PFCE9 register as the PFCE9H register and the lower 8 bits as the PFCE9L register, they can be read or written in 8-bit or 1-bit units.

**3.**  To read/write bits 8 to 15 of the PFCE9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PFCE9H register.

**(6) Port 9 alternate function specifications**

| PFCE915 | PFC915 | Specification of P915 pin alternate function |
|---------|--------|----------------------------------------------|
| 0 | 0 | TIAA50 input |
| 0 | 1 | TOAA50 output |
| 1 | 0 | INTP18 input |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A15 output (V850ES/JH3-U) |

| PFCE914 | PFC914 | Specification of P914 pin alternate function |
|---------|--------|----------------------------------------------|
| 0 | 0 | TIAA51 input |
| 0 | 1 | TOAA51 output |
| 1 | 0 | INTP17 input |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A14 output (V850ES/JH3-U) |

| PFC913[Note] | Specification of P913 pin alternate function |
|--------------|----------------------------------------------|
| 0 | TOAB1OFF input/INTP16 input |
| 1 | A13 output[Note] |

| PFC912[Note] | Specification of P912 pin alternate function |
|--------------|----------------------------------------------|
| 0 | $\overline{\text{SCKF3}}$ I/O |
| 1 | A12 output[Note] |

| PFCE911 | PFC911 | Specification of P911 pin alternate function |
|---------|--------|----------------------------------------------|
| 0 | 0 | SOF3 output |
| 0 | 1 | RXDC2 input |
| 1 | 0 | INTP15 input |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A11 output (V850ES/JH3-U) |

| PFCE910 | PFC910 | Specification of P910 pin alternate function |
|---------|--------|----------------------------------------------|
| 0 | 0 | SIF3 input |
| 0 | 1 | TXDC2 output |
| 1 | 0 | INTP14 input |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A10 output (V850ES/JH3-U) |

| PFCE99[Note] | PFC99 | Specification of P99 pin alternate function |
|--------------|-------|---------------------------------------------|
| 0 | 0 | $\overline{\text{SCKF1}}$ I/O |
| 0 | 1 | INTP13 input |
| 1 | 0 | A9 output[Note] |
| 1 | 1 | Setting prohibited[Note] |

| PFCE98[Note] | PFC98 | Specification of P98 pin alternate function |
|---|---|---|
| 0 | 0 | SOF1 output |
| 0 | 1 | INTP12 input |
| 1 | 0 | A8 output[Note] |
| 1 | 1 | Setting prohibited[Note] |

| PFCE97 | PFC97 | Specification of P97 pin alternate function |
|---|---|---|
| 0 | 0 | SIF1 input |
| 0 | 1 | TIAA20 input |
| 1 | 0 | TOAA20 output |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A7 output (V850ES/JH3-U) |

| PFCE96 | PFC96 | Specification of P96 pin alternate function |
|---|---|---|
| 0 | 0 | TIAA21 input |
| 0 | 1 | TOAA21 output |
| 1 | 0 | INTP11 input |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A6 output (V850ES/JH3-U) |

| PFCE95[Note] | PFC95 | Specification of P95 pin alternate function |
|---|---|---|
| 0 | 0 | TIAA30 input |
| 0 | 1 | TOAA30 output |
| 1 | 0 | A5 output[Note] |
| 1 | 1 | Setting prohibited[Note] |

| PFCE94 | PFC94 | Specification of P94 pin alternate function |
|---|---|---|
| 0 | 0 | TIAA31 input |
| 0 | 1 | TOAA31 output |
| 1 | 0 | TENC00 input/EVTT0 input |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A4 output (V850ES/JH3-U) |

| PFCE93 | PFC93 | Specification of P93 pin alternate function |
|---|---|---|
| 0 | 0 | TECR0 input |
| 0 | 1 | TIT00 input |
| 1 | 0 | TOT00 output |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A3 output (V850ES/JH3-U) |

**Note** V850ES/JH3-U only

| PFCE92 | PFC92 | Specification of P92 pin alternate function |
|--------|-------|---------------------------------------------|
| 0 | 0 | TENC01 input |
| 0 | 1 | TIT01 input |
| 1 | 0 | TOT01 output |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A2 output (V850ES/JH3-U) |

| PFCE91 | PFC91 | Specification of P91 pin alternate function |
|--------|-------|---------------------------------------------|
| 0 | 0 | KR7 input |
| 0 | 1 | RXDC1 input |
| 1 | 0 | SCL02 I/O |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A1 output (V850ES/JH3-U) |

| PFCE90 | PFC90 | Specification of P90 pin alternate function |
|--------|-------|---------------------------------------------|
| 0 | 0 | KR6 input |
| 0 | 1 | TXDC1 output |
| 1 | 0 | SDA02 I/O |
| 1 | 1 | Setting prohibited (V850ES/JG3-U)<br>A0 output (V850ES/JH3-U) |

**(7) Port 9 function register (PF9)**

After reset: 0000H    R/W    Address: PF9 FFFFFC72H,
                                      PF9L FFFFFC72H

|      | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|------|----|----|----|----|----|----|---|---|
| PF9  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 |

|        | 7 | 6 | 5 | 4 | 3 | 2 | 1    | 0    |
|--------|---|---|---|---|---|---|------|------|
| (PF9L) | 0 | 0 | 0 | 0 | 0 | 0 | PF91 | PF90 |

| PF9n | Control of normal output or N-ch open-drain output (n = 0, 1) |
|------|----------------------------------------------------------------|
| 0    | Normal output (CMOS output)                                    |
| 1    | N-ch open-drain output                                         |

**Caution   When output pins P90 and P91 are pulled up to EV$_{DD}$ or higher, be sure to set the PF9n bit to 1.**

**Remark**   The PF9 register can be read or written in 16-bit units.
However, when using the lower 8 bits of the PF9 register as the PF9L register, they can be read or written in 8-bit or 1-bit units.

#### 4.3.10  Port CM

Port CM is 1-bit (V850ES/JG3-U)/4-bit (V850ES/JH3-U) port for which I/O settings can be controlled in 1-bit units.

Port CM includes the following alternate-function pins.

**Table 4-15.  Port CM Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| PCM0 | – | 89 | $\overline{\text{WAIT}}$ | Input | – |
| PCM1 | 64 | 86 | CLKOUT | Output | |
| PCM2 | – | 10 | $\overline{\text{HLDAK}}$ | Output | |
| PCM3 | – | 11 | $\overline{\text{HLDRQ}}$ | Input | |

#### (1)  Port CM register (PCM)

**(a)  V850ES/JG3-U**

After reset:  00H (output latch)  R/W  Address:  FFFFF00CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PCM | 0 | 0 | 0 | 0 | 0 | 0 | PCM1 | 0 |

| PCM1 | Output data control (in output mode) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(b)  V850ES/JH3-U**

After reset:  00H (output latch)  R/W  Address:  FFFFF00CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PCM | 0 | 0 | 0 | 0 | PCM3 | PCM2 | PCM1 | PCM0 |

| PCMn | Output data control (in output mode) (n = 0 to 3) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(2)  Port CM mode register (PMCM)**

**(a)  V850ES/JG3-U**

After reset:  FFH      R/W      Address:  FFFFF02CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCM | 1 | 1 | 1 | 1 | 1 | 1 | PMCM1 | 1 |

| PMCM1 | I/O mode control |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(b)  V850ES/JH3-U**

After reset:  FFH      R/W      Address:  FFFFF02CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCM | 1 | 1 | 1 | 1 | PMCM3 | PMCM2 | PMCM1 | PMCM0 |

| PMCMn | I/O mode control (n = 0 to 3) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(3)  Port CM mode control register (PMCCM)**

**(a)  V850ES/JG3-U**

After reset:  00H    R/W    Address:  FFFFF04CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCCM | 0 | 0 | 0 | 0 | 0 | 0 | PMCCM1 | 0 |

| PMCCM1 | Specification of PCM1 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | CLKOUT output |

**(b)  V850ES/JH3-U**

After reset:  00H    R/W    Address:  FFFFF04CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCCM | 0 | 0 | 0 | 0 | PMCCM3 | PMCCM2 | PMCCM1 | PMCCM0 |

| PMCCM3 | Specification of PCM3 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{HLDRQ}}$ input |

| PMCCM2 | Specification of PCM2 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{HLDAK}}$ output |

| PMCCM1 | Specification of PCM1 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | CLKOUT output |

| PMCCM0 | Specification of PCM0 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{WAIT}}$ input |

**4.3.11  Port CS (V850ES/JH3-U only)**

Port CS is a 3-bit port for which I/O settings can be controlled in 1-bit units.

Port CS includes the following alternate-function pins.

**Table 4-16.  Port CM Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|----------|---------|---------|-----------------------------|-----|--------|
|          | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| PCS0 | – | 96 | $\overline{CS0}$ | Output | – |
| PCS2 | – | 97 | $\overline{CS2}$ | Output | |
| PCS3 | – | 116 | $\overline{CS3}$ | Output | |

**(1)  Port CS register (PCS)**

After reset:  00H (output latch)     R/W      Address:  FFFFF008H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|------|------|---|------|
| PCS | 0 | 0 | 0 | 0 | PCS3 | PCS2 | 0 | PCS0 |

| PCSn | Output data control (in output mode) (n = 0, 2, 3) |
|------|----------------------------------------------------|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(2)  Port CS mode register (PMCS)**

After reset:  FFH      R/W      Address:  FFFFF028H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|-------|-------|---|-------|
| PMCS | 1 | 1 | 1 | 1 | PMCS3 | PMCS2 | 1 | PMCS0 |

| PMCSn | I/O mode control (n = 0, 2, 3) |
|-------|--------------------------------|
| 0 | Output mode |
| 1 | Input mode |

**(3) Port CS mode control register (PMCCS)**

After reset: 00H    R/W    Address: FFFFF048H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCCS | 0 | 0 | 0 | 0 | PMCCS3 | PMCCS2 | 0 | PMCCS0 |

| PMCCS3 | Specification of PCS3 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{CS3}}$ output |

| PMCCS2 | Specification of PCS2 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{CS2}}$ output |

| PMCCS0 | Specification of PCS0 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{\text{CS0}}$ output |

#### 4.3.12  Port CT

Port CT is a 2-bit (V850ES/JG3-U)/4-bit (V850ES/JH3-U) port for which I/O settings can be controlled in 1-bit units.
Port CT includes the following alternate-function pins.

**Table 4-17.  Port CT Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| PCT0 | 58 | 76 | $\overline{WR0}$ | Output | – |
| PCT1 | 59 | 77 | $\overline{WR1}$ | Output | |
| PCT4 | – | 87 | $\overline{RD}$ | Output | |
| PCT6 | – | 88 | ASTB | Output | |

### (1)  Port CT register (PCT)

#### (a)  V850ES/JG3-U

After reset:  00H (output latch)       R/W       Address:  FFFFF00AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PCT | 0 | 0 | 0 | 0 | 0 | 0 | PCT1 | PCT0 |

| PCTn | Output data control (in output mode) (n = 0, 1) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

#### (b)  V850ES/JH3-U

After reset:  00H (output latch)       R/W       Address:  FFFFF00AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PCT | 0 | PCT6 | 0 | PCT4 | 0 | 0 | PCT1 | PCT0 |

| PCTn | Output data control (in output mode) (n = 0, 1, 4, 6) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(2) Port CT mode register (PMCT)**

**(a) V850ES/JG3-U**

After reset: FFH    R/W    Address: FFFFF02AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCT | 1 | 1 | 1 | 1 | 1 | 1 | PMCT1 | PMCT0 |

| PMCTn | I/O mode control (n = 0, 1) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(b) V850ES/JH3-U**

After reset: FFH    R/W    Address: FFFFF02AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCT | 1 | PMCT6 | 1 | PMCT4 | 1 | 1 | PMCT1 | PMCT0 |

| PMCTn | I/O mode control (n = 0, 1, 4, 6) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(3) Port CT mode control register (PMCCT)**

**(a) V850ES/JG3-U**

After reset: 00H    R/W    Address: FFFFF04AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCCT | 0 | 0 | 0 | 0 | 0 | 0 | PMCCT1 | PMCCT0 |

| PMCCT1 | Specification of PCT1 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{WR1}$ output |

| PMCCT0 | Specification of PCT0 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{WR0}$ output |

**(b) V850ES/JH3-U**

After reset: 00H    R/W    Address: FFFFF04AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCCT | 0 | PMCCT6 | 0 | PMCCT4 | 0 | 0 | PMCCT1 | PMCCT0 |

| PMCCT6 | Specification of PCT6 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | ASTB output |

| PMCCT4 | Specification of PCT4 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{RD}$ output |

| PMCCT1 | Specification of PCT1 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{WR1}$ output |

| PMCCT0 | Specification of PCT0 pin operation mode |
|---|---|
| 0 | I/O port |
| 1 | $\overline{WR0}$ output |

**4.3.13  Port DH (V850ES/JH3-U only)**

Port DH is an 8-bit port for which I/O settings can be controlled in 1-bit units.

Port DH includes the following alternate-function pins.

**Table 4-18.  Port DH Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|----------|---------|---------|----------------------------|-----|--------|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| PDH0 | – | 72 | A16 | Output | – |
| PDH1 | – | 73 | A17 | Output | |
| PDH2 | – | 74 | A18 | Output | |
| PDH3 | – | 75 | A19 | Output | |
| PDH4 | – | 78 | A20 | Output | |
| PDH5 | – | 79 | A21 | Output | |
| PDH6 | – | 80 | A22 | Output | |
| PDH7 | – | 81 | A23 | Output | |

**(1)  Port DH register (PDH)**

After reset:  00H (output latch)      R/W      Address:  FFFFF006H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PDH | PDH7 | PDH6 | PDH5 | PDH4 | PDH3 | PDH2 | PDH1 | PDH0 |

| PDHn | Output data control (in output mode) (n = 0 to 7) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**(2)  Port DH mode register (PMDH)**

After reset:  FFH      R/W      Address:  FFFFF026H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMDH | PMDH7 | PMDH6 | PMDH5 | PMDH4 | PMDH3 | PMDH2 | PMDH1 | PMDH0 |

| PMDHn | I/O mode control (n = 0, 7) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**(3)  Port DH mode control register (PMCDH)**

After reset:  00H      R/W      Address:  FFFFF046H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMCDH | PMCDH7 | PMCDH6 | PMCDH5 | PMCDH4 | PMCDH3 | PMCDH2 | PMCDH1 | PMCDH0 |

| PMCDHn | Specification of PDHn pin operation mode (n = 0 to 7) |
|---|---|
| 0 | I/O port |
| 1 | Am output (Address bus) (m = 16 to 23) |

**4.3.14  Port DL**

Port DL is a 16-bit port for which I/O settings can be controlled in 1-bit units.

Port DL includes the following alternate-function pins.

**Table 4-19.  Port DL Alternate-Function Pins**

| Pin Name | Pin No. | | Alternate-Function Pin Name | I/O | Remark |
|---|---|---|---|---|---|
| | V850ES/ JG3-U | V850ES/ JH3-U | | | |
| PDL0 | 71 | 98 | AD0 | I/O | – |
| PDL1 | 72 | 99 | AD1 | I/O | |
| PDL2 | 73 | 100 | AD2 | I/O | |
| PDL3 | 74 | 101 | AD3 | I/O | |
| PDL4 | 75 | 102 | AD4 | I/O | |
| PDL5 | 78 | 103 | AD5/FLMD1[Note] | I/O | |
| PDL6 | 79 | 104 | AD6 | I/O | |
| PDL7 | 80 | 105 | AD7 | I/O | |
| PDL8 | 81 | 108 | AD8 | I/O | |
| PDL9 | 82 | 109 | AD9 | I/O | |
| PDL10 | 83 | 110 | AD10 | I/O | |
| PDL11 | 84 | 111 | AD11 | I/O | |
| PDL12 | 85 | 112 | AD12 | I/O | |
| PDL13 | 86 | 113 | AD13 | I/O | |
| PDL14 | 87 | 114 | AD14 | I/O | |
| PDL15 | 88 | 115 | AD15 | I/O | |

**Note**   Since this pin is set in the flash memory programming mode, it does not need to be manipulated with the port control register.  For details, see **CHAPTER 31  FLASH MEMORY**.

**(1)  Port DL register (PDL)**

After reset:  0000H (output latch)      R/W      Address:  PDL FFFFF004H,
                                                          PDLL FFFFF004H, PDLH FFFFF005H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PDL (PDLH) | PDL15 | PDL14 | PDL13 | PDL12 | PDL11 | PDL10 | PDL9 | PDL8 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (PDLL) | PDL7 | PDL6 | PDL5 | PDL4 | PDL3 | PDL2 | PDL1 | PDL0 |

| PDLn | Output data control (in output mode) (n = 0 to 15) |
|---|---|
| 0 | Outputs 0. |
| 1 | Outputs 1. |

**Remarks 1.** The PDL register can be read or written in 16-bit units.
           However, when using the higher 8 bits of the PDL register as the PDLH register and
           the lower 8 bits as the PDLL register, they can be read or written in 8-bit or 1-bit units.
**2.** To read/write bits 8 to 15 of the PDL register in 8-bit or 1-bit units, specify them as
           bits 0 to 7 of the PDLH register.

**(2)  Port DL mode register (PMDL)**

After reset:  FFFFH      R/W      Address:    PMDL FFFFF024H,
                                             PMDLL FFFFF024H, PMDLH FFFFF025H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PMDL (PMDLH) | PMDL15 | PMDL14 | PMDL13 | PMDL12 | PMDL11 | PMDL10 | PMDL9 | PMDL8 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (PMDLL) | PMDL7 | PMDL6 | PMDL5 | PMDL4 | PMDL3 | PMDL2 | PMDL1 | PMDL0 |

| PMDLn | I/O mode control (n = 0 to 15) |
|---|---|
| 0 | Output mode |
| 1 | Input mode |

**Remarks 1.** The PMDL register can be read or written in 16-bit units.
           However, when using the higher 8 bits of the PMDL register as the PMDLH register
           and the lower 8 bits as the PMDLL register, they can be read or written in 8-bit or 1-bit
           units.
**2.** To read/write bits 8 to 15 of the PMDL register in 8-bit or 1-bit units, specify them as
           bits 0 to 7 of the PMDLH register.

**(3) Port DL mode control register (PMCDL)**

After reset:  0000H      R/W      Address:  PMCDL FFFFF044H,
                                                PMCDLL FFFFF044H, PMCDLH FFFFF045H

|   | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PMCDL (PMCDLH) | PMCDL15 | PMCDL14 | PMCDL13 | PMCDL12 | PMCDL11 | PMCDL10 | PMCDL9 | PMCDL8 |

|   | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (PMCDLL) | PMCDL7 | PMCDL6 | PMCDL5 | PMCDL4 | PMCDL3 | PMCDL2 | PMCDL1 | PMCDL0 |

| PMCDLn | Specification of PDLn pin operation mode (n = 0 to 15) |
|---|---|
| 0 | I/O port |
| 1 | ADn I/O (address/data bus I/O) |

Remarks  **1.** The PMCDL register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PMCDL register as the PMCDLH register and the lower 8 bits as the PMCDLL register, they can be read or written in 8-bit or 1-bit units.

**2.** To read/write bits 8 to 15 of the PMCDL register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMCDLH register.

## 4.4   Port Register Settings When Alternate Function Is Used

Table 4-20 shows the port register settings when each port is used for an alternate function. When using a port pin as an alternate-function pin, refer to the description of each pin.

**Table 4-20.   Using Port Pin as Alternate-Function Pin (1/10)**

| Pin Name | Alternate Function | | Pnx Bit of Pn Register | PMnx Bit of PMn Register | PMCnx Bit of PMCn Register | PFCEnx Bit of PFCEn Register | PFCnx Bit of PFCn Register | Other Bits (Registers) |
|---|---|---|---|---|---|---|---|---|
| | Name | I/O | | | | | | |
| P00[Note 1] | INTP00 | Input | P00 = Setting not required | PM00 = Setting not required | PMC00 = 1 | – | – | |
| P01[Note 1] | INTP01 | Input | P01 = Setting not required | PM01 = Setting not required | PMC01 = 1 | – | – | |
| P02 | NMI | Input | P02 = Setting not required | PM02 = Setting not required | PMC02 = 1 | – | – | |
| P03 | INTP02 | Input | P03 = Setting not required | PM03 = Setting not required | PMC03 = 1 | PFCE03 = 0 | PFC03 = 0 | |
| | ADTRG | Input | P03 = Setting not required | PM03 = Setting not required | PMC03 = 1 | PFCE03 = 0 | PFC03 = 1 | |
| | UCLK | Input | P03 = Setting not required | PM03 = Setting not required | PMC03 = 1 | PFCE03 = 1 | PFC03 = 0 | |
| P04[Note 1] | INTP03[Note 1] | Input | P04 = Setting not required | PM04 = Setting not required | PMC04 = 1 | – | – | |
| P05[Note 1] | INTP04[Note 1] | Input | P05 = Setting not required | PM05 = Setting not required | PMC05 = 1 | – | – | |
| P10 | ANO0 | Output | P10 = Setting not required | PM10 = 1 | – | – | – | |
| P11 | ANO1 | Output | P11 = Setting not required | PM11 = 1 | – | – | – | |
| P20[Note 1] | TIAB03 | Input | P20 = Setting not required | PM20 = Setting not required | PMC20 = 1 | PFCE20 = 0 | PFC20 = 0 | |
| | KR2 | Input | P20 = Setting not required | PM20 = Setting not required | PMC20 = 1 | PFCE20 = 0 | PFC20 = 0 | |
| | TOAB03 | Output | P20 = Setting not required | PM20 = Setting not required | PMC20 = 1 | PFCE20 = 0 | PFC20 = 1 | |
| | RTP02 | Output | P20 = Setting not required | PM20 = Setting not required | PMC20 = 1 | PFCE20 = 1 | PFC20 = 0 | |
| P21[Note 1] | SIF2 | Input | P21 = Setting not required | PM21 = Setting not required | PMC21 = 1 | PFCE21 = 0 | PFC21 = 0 | |
| | KR3[Note 2] | Input | P21 = Setting not required | PM21 = Setting not required | PMC21 = 1 | PFCE21 = 0 | PFC21 = 1 | |
| | TIAB00[Note 2] | Input | P21 = Setting not required | PM21 = Setting not required | PMC21 = 1 | PFCE21 = 0 | PFC21 = 1 | |
| | TOAB00 | Output | P21 = Setting not required | PM21 = Setting not required | PMC21 = 1 | PFCE21 = 1 | PFC21 = 0 | |
| | RTP03 | Output | P21 = Setting not required | PM21 = Setting not required | PMC21 = 1 | PFCE21 = 1 | PFC21 = 1 | |

**Notes 1.** V850ES/JH3-U only

**2.** The KR3 pin and TIAB00 pin are alternate-function pins.   When using this pin as the TIAB00 pin, do not use the alternate function KR3 pin.   Similarly, when using this pin as the KR3 pin, do not use the TIAB00 pin.

**Caution   When the power is turned on, the P10 and P11 pins may output an undefined level temporarily even during reset.**

**Table 4-20.   Using Port Pin as Alternate-Function Pin (2/10)**

| Pin Name | Alternate Function | | Pnx Bit of | PMnx Bit of | PMCnx Bit of | PFCEnx Bit of | PFCnx Bit of | Other Bits |
|---|---|---|---|---|---|---|---|---|
| | Name | I/O | Pn Register | PMn Register | PMCn Register | PFCEn Register | PFCn Register | (Registers) |
| P22[Note] | SOF2 | Output | P22 = Setting not required | PM22 = Setting not required | PMC22= 1 | PFCE22 = 0 | PFC22 = 0 | |
| | KR4 | Input | P22 = Setting not required | PM22 = Setting not required | PMC22= 1 | PFCE22 = 0 | PFC22 = 1 | |
| | RTP04 | Output | P22 = Setting not required | PM22 = Setting not required | PMC22= 1 | PFCE22 = 1 | PFC22 = 0 | |
| P23[Note] | SCKF2 | I/O | P23 = Setting not required | PM23 = Setting not required | PMC23= 1 | PFCE23 = 0 | PFC23 = 0 | |
| | KR5 | Input | P23 = Setting not required | PM23 = Setting not required | PMC23= 1 | PFCE23 = 0 | PFC23 = 1 | |
| | RTP05 | Output | P23 = Setting not required | PM23 = Setting not required | PMC23= 1 | PFCE23 = 1 | PFC23 = 0 | |
| P24[Note] | INTP05 | Input | P24 = Setting not required | PM24 = Setting not required | PMC24= 1 | – | – | |
| P25[Note] | INTP06 | Input | P25 = Setting not required | PM25 = Setting not required | PMC25= 1 | – | – | |
| P30 | TXDC0 | Output | P30 = Setting not required | PM30 = Setting not required | PMC30 = 1 | PFCE30 = 0 | PFC30 = 0 | |
| | SOF4 | Output | P30 = Setting not required | PM30 = Setting not required | PMC30 = 1 | PFCE30 = 0 | PFC30 = 1 | |
| | INTP07 | Input | P30 = Setting not required | PM30 = Setting not required | PMC30 = 1 | PFCE30 = 1 | PFC30 = 0 | |
| P31 | RXDC0 | Input | P31 = Setting not required | PM31 = Setting not required | PMC31 = 1 | PFCE31 = 0 | PFC31 = 0 | |
| | SIF4 | Input | P31 = Setting not required | PM31 = Setting not required | PMC31 = 1 | PFCE31 = 0 | PFC31 = 1 | |
| | INTP08 | Input | P31 = Setting not required | PM31 = Setting not required | PMC31 = 1 | PFCE31 = 1 | PFC31 = 0 | |
| P32 | ASCKC0 | Input | P32 = Setting not required | PM32 = Setting not required | PMC32 = 1 | PFCE32 = 0 | PFC32 = 0 | |
| | SCKF4 | I/O | P32 = Setting not required | PM32 = Setting not required | PMC32 = 1 | PFCE32 = 0 | PFC32 = 1 | |
| | TIAA00 | Input | P32 = Setting not required | PM32 = Setting not required | PMC32 = 1 | PFCE32 = 1 | PFC32 = 0 | |
| | TOAA00 | Output | P32 = Setting not required | PM32 = Setting not required | PMC32 = 1 | PFCE32 = 1 | PFC32 = 1 | |
| P33 | TIAA01 | Input | P33 = Setting not required | PM33 = Setting not required | PMC33 = 1 | PFCE33 = 0 | PFC33 = 0 | |
| | TOAA01 | Output | P33 = Setting not required | PM33 = Setting not required | PMC33 = 1 | PFCE33 = 0 | PFC33 = 1 | |
| | RTCDIV | Output | P33 = Setting not required | PM33 = Setting not required | PMC33 = 1 | PFCE33 = 1 | PFC33 = 0 | |
| | RTCCL | Output | P33 = Setting not required | PM33 = Setting not required | PMC33 = 1 | PFCE33 = 1 | PFC33 = 1 | |

**Note**   V850ES/JH3-U only

**Table 4-20.   Using Port Pin as Alternate-Function Pin (3/10)**

| Pin Name | Alternate Function | | Pnx Bit of Pn Register | PMnx Bit of PMn Register | PMCnx Bit of PMCn Register | PFCEnx Bit of PFCEn Register | PFCnx Bit of PFCn Register | Other Bits (Registers) |
|---|---|---|---|---|---|---|---|---|
| | Name | I/O | | | | | | |
| P34 | TIAA10 | Input | P34 = Setting not required | PM34 = Setting not required | PMC34 = 1 | PFCE34 = 0 | PFC34 = 0 | |
| | TOAA10 | Output | P34 = Setting not required | PM34 = Setting not required | PMC34 = 1 | PFCE34 = 0 | PFC34 = 1 | |
| | TOAA1OFF[Note] | Input | P34 = Setting not required | PM34 = Setting not required | PMC34 = 1 | PFCE34 = 1 | PFC34 = 0 | |
| | INTP09[Note] | Input | P34 = Setting not required | PM34 = Setting not required | PMC34 = 1 | PFCE34 = 1 | PFC34 = 0 | |
| | PPON | Output | P34 = Setting not required | PM34 = Setting not required | PMC34 = 1 | PFCE34 = 1 | PFC34 = 1 | |
| P35 | TIAA11 | Input | P35 = Setting not required | PM35 = Setting not required | PMC35 = 1 | PFCE35 = 0 | PFC35 = 0 | |
| | TOAA11 | Output | P35 = Setting not required | PM35 = Setting not required | PMC35 = 1 | PFCE35 = 0 | PFC35 = 1 | |
| | RTC1HZ | Output | P35 = Setting not required | PM35 = Setting not required | PMC35 = 1 | PFCE35 = 1 | PFC35 = 0 | |
| | $\overline{\text{OCI}}$ | Input | P35 = Setting not required | PM35 = Setting not required | PMC35 = 1 | PFCE35 = 1 | PFC35 = 1 | |
| P36 | TXDC3 | Output | P36 = Setting not required | PM36 = Setting not required | PMC36 = 1 | PFCE36 = 0 | PFC36 = 0 | |
| | SCL00 | I/O | P36 = Setting required | PM36 = Setting not required | PMC36 = 1 | PFCE36 = 0 | PFC36 = 1 | PF36 (PF3) = 1 |
| | $\overline{\text{UDMARQ0}}$ | Input | P36 = Setting not required | PM36 = Setting not required | PMC36 = 1 | PFCE36 = 1 | PFC36 = 1 | |
| P37 | RXDC3 | Input | P37 = Setting not required | PM37 = Setting not required | PMC37 = 1 | PFCE37 = 0 | PFC37 = 0 | |
| | SDA00 | I/O | P37 = Setting required | PM37 = Setting not required | PMC37 = 1 | PFCE37 = 0 | PFC37 = 1 | PF37 (PF3) = 1 |
| | $\overline{\text{UDMAAK0}}$ | Output | P37 = Setting not required | PM37 = Setting not required | PMC37 = 1 | PFCE37 = 1 | PFC37 = 1 | |
| P40 | SIF0 | Input | P40 = Setting not required | PM40 = Setting not required | PMC40 = 1 | PFCE40 = 0 | PFC40 = 0 | |
| | TXDC4 | Output | P40 = Setting not required | PM40 = Setting not required | PMC40 = 1 | PFCE40 = 0 | PFC40 = 1 | |
| | SDA01 | I/O | P40 = Setting required | PM40 = Setting not required | PMC40 = 1 | PFCE40 = 1 | PFC40 = 0 | PF40 (PF4) = 1 |
| P41 | SOF0 | Output | P41 = Setting not required | PM41 = Setting not required | PMC41 = 1 | PFCE41 = 0 | PFC41 = 0 | |
| | RXDC4 | Input | P41 = Setting not required | PM41 = Setting not required | PMC41 = 1 | PFCE41 = 0 | PFC41 = 1 | |
| | SCL01 | I/O | P41 = Setting required | PM41 = Setting not required | PMC41 = 1 | PFCE41 = 1 | PFC41 = 0 | PF41 (PF4) = 1 |
| P42 | $\overline{\text{SCKF0}}$ | I/O | P42 = Setting not required | PM42 = Setting not required | PMC42 = 1 | − | PFC42 = 0 | |
| | INTP10 | Input | P42 = Setting not required | PM42 = Setting not required | PMC42 = 1 | − | PFC42 = 1 | |

**Note** The TOAA1OFF pin and INTP09 pin are alternate-function pins.   When using this pin as the TOAA1OFF pin, disable edge detection of the INTP09 pin, which is the alternate function (set to INTF3.INTF34 = 0, INTR3.INTR34 = 0).   Similarly, when using this pin as the INTP09 pin, be sure to stop the high impedance output circuit.

**Table 4-20.   Using Port Pin as Alternate-Function Pin (4/10)**

| Pin Name | Alternate Function | | Pnx Bit of Pn Register | PMnx Bit of PMn Register | PMCnx Bit of PMCn Register | PFCEnx Bit of PFCEn Register | PFCnx Bit of PFCn Register | Other Bits (Registers) |
|---|---|---|---|---|---|---|---|---|
| | Name | I/O | | | | | | |
| P50 | TIAB01 | Input | P50 = Setting not required | PM50 = Setting not required | PMC50 = 1 | PFCE50 = 0 | PFC50 = 0 | KRM0 (KRM)= 0 |
| | KR0 | Input | P50 = Setting not required | PM50 = Setting not required | PMC50 = 1 | PFCE50 = 0 | PFC50 = 0 | TAB0TIG2, TAB0TIG3 (TAB0IOC1) = 0 |
| | TOAB01 | Output | P50 = Setting not required | PM50 = Setting not required | PMC50 = 1 | PFCE50 = 0 | PFC50 = 1 | |
| | RTP00 | Output | P50 = Setting not required | PM50 = Setting not required | PMC50 = 1 | PFCE50 = 1 | PFC50 = 0 | |
| | UDMARQ1 | Input | P50 = Setting not required | PM50 = Setting not required | PMC50 = 1 | PFCE50 = 1 | PFC50 = 1 | |
| P51 | TIAB02 | Input | P51 = Setting not required | PM51 = Setting not required | PMC51 = 1 | PFCE51 = 0 | PFC51 = 0 | KRM1 (KRM) = 0 |
| | KR1 | Input | P51 = Setting not required | PM51 = Setting not required | PMC51 = 1 | PFCE51 = 0 | PFC51 = 0 | TAB0TIG4, TAB0TIG5 (TAB0IOC1) = 0 |
| | TOAB02 | Output | P51 = Setting not required | PM51 = Setting not required | PMC51 = 1 | PFCE51 = 0 | PFC51 = 1 | |
| | RTP01 | Output | P51 = Setting not required | PM51 = Setting not required | PMC51 = 1 | PFCE51 = 1 | PFC51 = 0 | |
| | UDMAAK1 | Output | P51 = Setting not required | PM51 = Setting not required | PMC51 = 1 | PFCE51 = 1 | PFC51 = 1 | |
| P52[Note] | TIAB03 | Input | P52 = Setting not required | PM52 = Setting not required | PMC52 = 1 | PFCE52 = 0 | PFC52 = 0 | KRM2 (KRM) = 0 |
| | KR2 | Input | P52 = Setting not required | PM52 = Setting not required | PMC52 = 1 | PFCE52 = 0 | PFC52 = 0 | TAB0TIG6, TAB0TIG7 (TAB0I0C1) = 0 |
| | TOAB03 | Output | P52 = Setting not required | PM52 = Setting not required | PMC52 = 1 | PFCE52 = 0 | PFC52 = 1 | |
| | RTP02 | Output | P52 = Setting not required | PM52 = Setting not required | PMC52 = 1 | PFCE52 = 1 | PFC52 = 0 | |
| P53[Note] | SIF2 | Input | P53 = Setting not required | PM53 = Setting not required | PMC53 = 1 | PFCE53 = 0 | PFC53 = 0 | |
| | TIAB00 | Input | P53 = Setting not required | PM53 = Setting not required | PMC53 = 1 | PFCE53 = 0 | PFC53 = 1 | KRM3 (KRM) = 0 |
| | KR3 | Input | P53 = Setting not required | PM53 = Setting not required | PMC53 = 1 | PFCE53 = 0 | PFC53 = 1 | TAB0TIG0, TAB0TIG1 (TAB0IOC1) = 0, TAB0EES0, TAB0EES1 (TAB0IOC2) = 0, TAB0ETS0, TAB0ETS1 (TAB0IOC2) = 0 |
| | TOAB00 | Output | P53 = Setting not required | PM53 = Setting not required | PMC53 = 1 | PFCE53 = 1 | PFC53 = 0 | |
| | RTP03 | Output | P53 = Setting not required | PM53 = Setting not required | PMC53 = 1 | PFCE53 = 1 | PFC53 = 1 | |
| P54[Note] | SOF2 | Output | P54 = Setting not required | PM54 = Setting not required | PMC54 = 1 | PFCE54 = 0 | PFC54 = 0 | |
| | KR4 | Input | P54 = Setting not required | PM54 = Setting not required | PMC54 = 1 | PFCE54 = 0 | PFC54 = 1 | |
| | RTP04 | Output | P54 = Setting not required | PM54 = Setting not required | PMC54 = 1 | PFCE54 = 1 | PFC54 = 0 | |

**Note**   V850ES/JG3-U only

**Table 4-20. Using Port Pin as Alternate-Function Pin (5/10)**

| Pin Name | Alternate Function Name | I/O | Pnx Bit of Pn Register | PMnx Bit of PMn Register | PMCnx Bit of PMCn Register | PFCEnx Bit of PFCEn Register | PFCnx Bit of PFCn Register | Other Bits (Registers) |
|---|---|---|---|---|---|---|---|---|
| P55[Note] | SCKF2 | I/O | P55 = Setting not required | PM55 = Setting not required | PMC55 = 1 | PFCE55 = 0 | PFC55 = 0 | |
| | KR5 | Input | P55 = Setting not required | PM55 = Setting not required | PMC55 = 1 | PFCE55 = 0 | PFC55 = 1 | |
| | RTP05 | Output | P55 = Setting not required | PM55 = Setting not required | PMC55 = 1 | PFCE55 = 1 | PFC55 = 0 | |
| P60 | TOAB1T1 | Output | P60 = Setting not required | PM60 = Setting not required | PMC60 = 1 | PFCE60 = 0[Note] | PFC60 = 0 | |
| | TOAB11 | Output | P60 = Setting not required | PM60 = Setting not required | PMC60 = 1 | PFCE60 = 0[Note] | PFC60 = 0 | |
| | TIAB11 | Input | P60 = Setting not required | PM60 = Setting not required | PMC60 = 1 | PFCE60 = 0[Note] | PFC60 = 1 | |
| | $\overline{\text{WAIT}}$[Note] | Output | P60 = Setting not required | PM60 = Setting not required | PMC60 = 1 | PFCE60 = 1[Note] | PFC60 = 0 | |
| P61 | TOAB1B1 | Output | P61 = Setting not required | PM61 = Setting not required | PMC61 = 1 | PFCE61 = 0 | PFC61 = 0 | |
| | TIAB10 | Input | P61 = Setting not required | PM61 = Setting not required | PMC61 = 1 | PFCE61 = 0 | PFC61 = 1 | |
| | TOAB10 | Output | P61 = Setting not required | PM61 = Setting not required | PMC61 = 1 | PFCE61 = 1 | PFC61 = 0 | |
| | $\overline{\text{RD}}$[Note] | Output | P61 = Setting not required | PM61 = Setting not required | PMC61 = 1 | PFCE61 = 1 | PFC61 = 1 | |
| P62 | TOAB1T2 | Output | P62 = Setting not required | PM62 = Setting not required | PMC62 = 1 | PFCE62 = 0[Note] | PFC62 = 0 | |
| | TOAB12 | Output | P62 = Setting not required | PM62 = Setting not required | PMC62 = 1 | PFCE62 = 0[Note] | PFC62 = 0 | |
| | TIAB12 | Input | P62 = Setting not required | PM62 = Setting not required | PMC62 = 1 | PFCE62 = 0[Note] | PFC62 = 1 | |
| | ASTB[Note] | Output | P62 = Setting not required | PM62 = Setting not required | PMC62 = 1 | PFCE62 = 1[Note] | PFC62 = 0 | |
| P63 | TOAB1B2 | Output | P63 = Setting not required | PM63 = Setting not required | PMC63 = 1 | PFCE63 = 0[Note] | PFC63 = 0 | |
| | TRGAB1 | Input | P63 = Setting not required | PM63 = Setting not required | PMC63 = 1 | PFCE63 = 0[Note] | PFC63 = 1 | |
| | $\overline{\text{CS0}}$[Note] | Output | P63 = Setting not required | PM63 = Setting not required | PMC63 = 1 | PFCE63 = 1[Note] | PFC63 = 0 | |
| P64 | TOAB1T3 | Output | P64 = Setting not required | PM64 = Setting not required | PMC64 = 1 | PFCE64 = 0[Note] | PFC64 = 0 | |
| | TOAB13 | Output | P64 = Setting not required | PM64 = Setting not required | PMC64 = 1 | PFCE64 = 0[Note] | PFC64 = 0 | |
| | TIAB13 | Input | P64 = Setting not required | PM64 = Setting not required | PMC64 = 1 | PFCE64 = 0[Note] | PFC64 = 1 | |
| | $\overline{\text{CS2}}$[Note] | Output | P64 = Setting not required | PM64 = Setting not required | PMC64 = 1 | PFCE64 = 1[Note] | PFC64 = 0 | |
| P65 | TOAB1B3 | Output | P65 = Setting not required | PM65 = Setting not required | PMC65 = 1 | PFCE65 = 0[Note] | PFC63 = 0 | |
| | EVTAB1 | Input | P65 = Setting not required | PM65 = Setting not required | PMC65 = 1 | PFCE65 = 0[Note] | PFC65 = 1 | |
| | $\overline{\text{CS3}}$[Note] | Output | P65 = Setting not required | PM65 = Setting not required | PMC65 = 1 | PFCE65 = 1[Note] | PFC65 = 0 | |

**Note** V850ES/JG3-U only

**Table 4-20.   Using Port Pin as Alternate-Function Pin (6/10)**

| Pin Name | Alternate Function | | Pnx Bit of Pn Register | PMnx Bit of PMn Register | PMCnx Bit of PMCn Register | PFCEnx Bit of PFCEn Register | PFCnx Bit of PFCn Register | Other Bits (Registers) |
|---|---|---|---|---|---|---|---|---|
| | Name | I/O | | | | | | |
| P70 | ANI0 | Input | P70 = Setting not required | PM70 = 1 | – | – | – | |
| P71 | ANI1 | Input | P71 = Setting not required | PM71 = 1 | – | – | – | |
| P72 | ANI2 | Input | P72 = Setting not required | PM72 = 1 | – | – | – | |
| P73 | ANI3 | Input | P73 = Setting not required | PM73 = 1 | – | – | – | |
| P74 | ANI4 | Input | P74 = Setting not required | PM74 = 1 | – | – | – | |
| P75 | ANI5 | Input | P75 = Setting not required | PM75 = 1 | – | – | – | |
| P76 | ANI6 | Input | P76 = Setting not required | PM76 = 1 | – | – | – | |
| P77 | ANI7 | Input | P77 = Setting not required | PM77 = 1 | – | – | – | |
| P78 | ANI8 | Input | P78 = Setting not required | PM78 = 1 | – | – | – | |
| P79 | ANI9 | Input | P79 = Setting not required | PM79 = 1 | – | – | – | |
| P710 | ANI10 | Input | P710 = Setting not required | PM710 = 1 | – | – | – | |
| P711 | ANI11 | Input | P711 = Setting not required | PM711 = 1 | – | – | – | |
| P90 | KR6 | Input | P90 = Setting not required | PM90 = Setting not required | PMC90 = 1 | PFCE90 = 0 | PFC90 = 0 | |
| | TXDC1 | Output | P90 = Setting not required | PM90 = Setting not required | PMC90 = 1 | PFCE90 = 0 | PFC90 = 1 | |
| | SDA02 | I/O | P90 = Setting not required | PM90 = Setting not required | PMC90 = 1 | PFCE90 = 1 | PFC90 = 0 | PF90 (PF9) = 1 |
| | A0[Note 1] | Output | P90 = Setting not required | PM90 = Setting not required | PMC90 = 1 | PFCE90 = 1 | PFC90 = 1 | **Note 2** |
| P91 | KR7 | Input | P91 = Setting not required | PM91 = Setting not required | PMC91 = 1 | PFCE91 = 0 | PFC91 = 0 | |
| | RXDC1 | Input | P91 = Setting not required | PM91 = Setting not required | PMC91 = 1 | PFCE91 = 0 | PFC91 = 1 | |
| | SCL02 | I/O | P91 = Setting not required | PM91 = Setting not required | PMC91 = 1 | PFCE91 = 1 | PFC91 = 0 | PF91 (PF9) = 1 |
| | A1[Note 1] | Output | P91 = Setting not required | PM91 = Setting not required | PMC91 = 1 | PFCE91 = 1 | PFC91 = 1 | **Note 2** |
| P92 | TENC01 | Input | P92 = Setting not required | PM92 = Setting not required | PMC92 = 1 | PFCE92 = 0 | PFC92 = 0 | |
| | TIT01 | Input | P92 = Setting not required | PM92 = Setting not required | PMC92 = 1 | PFCE92 = 0 | PFC92 = 1 | |
| | TOT01 | Output | P92 = Setting not required | PM92 = Setting not required | PMC92 = 1 | PFCE92 = 1 | PFC92 = 0 | |
| | A2[Note 1] | Output | P92 = Setting not required | PM92 = Setting not required | PMC92 = 1 | PFCE92 = 1 | PFC92 = 1 | **Note 2** |

**Notes 1.** V850ES/JH3-U only

**2.** When using as the A0 to A15 pins, be sure to set all 16 bits of the PMC9 register to FFFFH at once.

**Table 4-20.  Using Port Pin as Alternate-Function Pin (7/10)**

| Pin Name | Alternate Function | | Pnx Bit of Pn Register | PMnx Bit of PMn Register | PMCnx Bit of PMCn Register | PFCEnx Bit of PFCEn Register | PFCnx Bit of PFCn Register | Other Bits (Registers) |
|---|---|---|---|---|---|---|---|---|
| | Name | I/O | | | | | | |
| P93 | TECR0 | Input | P93 = Setting not required | PM93 = Setting not required | PMC93 = 1 | PFCE93 = 0 | PFC93 = 0 | |
| | TIT00 | Input | P93 = Setting not required | PM93 = Setting not required | PMC93 = 1 | PFCE93 = 0 | PFC93 = 1 | |
| | TOT00 | Output | P93 = Setting not required | PM93 = Setting not required | PMC93 = 1 | PFCE93 = 1 | PFC93 = 0 | |
| | A3[Note 1] | Output | P93 = Setting not required | PM93 = Setting not required | PMC93 = 1 | PFCE93 = 1 | PFC93 = 1 | **Note 2** |
| P94 | TIAA31 | Input | P94 = Setting not required | PM94 = Setting not required | PMC94 = 1 | PFCE94 = 0 | PFC94 = 0 | |
| | TOAA31 | Output | P94 = Setting not required | PM94 = Setting not required | PMC94 = 1 | PFCE94 = 0 | PFC94 = 1 | |
| | TENC00 | Input | P94 = Setting not required | PM94 = Setting not required | PMC94 = 1 | PFCE94 = 1 | PFC94 = 0 | |
| | EVTT0 | Input | P94 = Setting not required | PM94 = Setting not required | PMC94 = 1 | PFCE94 = 1 | PFC94 = 0 | |
| | A4[Note 1] | Output | P94 = Setting not required | PM94 = Setting not required | PMC94 = 1 | PFCE94 = 1 | PFC94 = 1 | **Note 2** |
| P95 | TIAA30 | Input | P95 = Setting not required | PM95 = Setting not required | PMC95 = 1 | PFCE95 = 0 | PFC95 = 0 | |
| | TOAA30 | Output | P95 = Setting not required | PM95 = Setting not required | PMC95 = 1 | PFCE95 = 0 | PFC95 = 1 | |
| | A5[Note 1] | Output | P95 = Setting not required | PM95 = Setting not required | PMC95 = 1 | PFCE95 = 1 | PFC95 = 0 | **Note 2** |
| P96 | TIAA21 | Input | P96 = Setting not required | PM96 = Setting not required | PMC96 = 1 | PFCE96 = 0 | PFC96 = 0 | |
| | TOAA21 | Output | P96 = Setting not required | PM96 = Setting not required | PMC96 = 1 | PFCE96 = 1 | PFC96 = 1 | |
| | INTP11 | Input | P96 = Setting not required | PM96 = Setting not required | PMC96 = 1 | PFCE96 = 1 | PFC96 = 0 | |
| | A6[Note 1] | Output | P96 = Setting not required | PM96 = Setting not required | PMC96 = 1 | PFCE96 = 1 | PFC96 = 1 | **Note 2** |
| P97 | SIF1 | Input | P97 = Setting not required | PM97 = Setting not required | PMC97 = 1 | PFCE97 = 0 | PFC97 = 0 | |
| | TIAA20 | Input | P97 = Setting not required | PM97 = Setting not required | PMC97 = 1 | PFCE97 = 0 | PFC97 = 1 | |
| | TOAA20 | Output | P97 = Setting not required | PM97 = Setting not required | PMC97 = 1 | PFCE97 = 1 | PFC97 = 0 | |
| | A7[Note 1] | Output | P97 = Setting not required | PM97 = Setting not required | PMC97 = 1 | PFCE97 = 1 | PFC97 = 1 | **Note 2** |
| P98 | SOF1 | Output | P98 = Setting not required | PM98 = Setting not required | PMC98 = 1 | PFCE98 = 0 | PFC98 = 0 | |
| | INTP12 | Input | P98 = Setting not required | PM98 = Setting not required | PMC98 = 1 | PFCE98 = 0 | PFC98 = 1 | |
| | A8[Note 1] | Output | P98 = Setting not required | PM98 = Setting not required | PMC98 = 1 | PFCE98 = 1 | PFC98 = 0 | **Note 2** |

**Notes 1.** V850ES/JH3-U only

**2.** When using as the A0 to A15 pins, be sure to set all 16 bits of the PMC9 register to FFFFH at once.

**Table 4-20.   Using Port Pin as Alternate-Function Pin (8/10)**

| Pin Name | Alternate Function | | Pnx Bit of Pn Register | PMnx Bit of PMn Register | PMCnx Bit of PMCn Register | PFCEnx Bit of PFCEn Register | PFCnx Bit of PFCn Register | Other Bits (Registers) |
|---|---|---|---|---|---|---|---|---|
| | Name | I/O | | | | | | |
| P99 | SCKF1 | I/O | P99 = Setting not required | PM99 = Setting not required | PMC99 = 1 | PFCE99 = 0 | PFC99 = 0 | |
| | INTP13 | Input | P99 = Setting not required | PM99 = Setting not required | PMC99 = 1 | PFCE99 = 0 | PFC99 = 1 | |
| | A9[Note 1] | Output | P99 = Setting not required | PM99 = Setting not required | PMC99 = 1 | PFCE99 = 1 | PFC99 = 0 | **Note 2** |
| P910 | SIF3 | Input | P910 = Setting not required | PM910 = Setting not required | PMC910 = 1 | PFCE910 = 0 | PFC910 = 0 | |
| | TXDC2 | Output | P910 = Setting not required | PM910 = Setting not required | PMC910 = 1 | PFCE910 = 0 | PFC910 = 1 | |
| | INTP14 | Input | P910 = Setting not required | PM910 = Setting not required | PMC910 = 1 | PFCE910 = 1 | PFC910 = 0 | |
| | A10[Note 1] | Output | P910 = Setting not required | PM910 = Setting not required | PMC910 = 1 | PFCE910 = 1 | PFC910 = 1 | **Note 2** |
| P911 | SOF3 | Output | P911 = Setting not required | PM911 = Setting not required | PMC911 = 1 | PFCE911 = 0 | PFC911 = 0 | |
| | RXDC2 | Input | P911 = Setting not required | PM911 = Setting not required | PMC911 = 1 | PFCE911 = 0 | PFC911 = 1 | |
| | INTP15 | Input | P911 = Setting not required | PM911 = Setting not required | PMC911 = 1 | PFCE911 = 1 | PFC911 = 0 | |
| | A11[Note 1] | Output | P911 = Setting not required | PM911 = Setting not required | PMC911 = 1 | PFCE911 = 1 | PFC911 = 1 | **Note 2** |
| P912 | SCKF3 | I/O | P912 = Setting not required | PM912 = Setting not required | PMC912 = 1 | – | PFC912 = 0 | |
| | A12[Note 1] | Output | P912 = Setting not required | PM912 = Setting not required | PMC912 = 1 | – | PFC912 = 1 | **Note 2** |
| P913 | TOAB1OFF | Input | P913 = Setting not required | PM913 = Setting not required | PMC913 = 1 | – | PFC913 = 0 | |
| | INTP16 | Input | P913 = Setting not required | PM913 = Setting not required | PMC913 = 1 | – | PFC913 = 0 | |
| | A13[Note 1] | Output | P913 = Setting not required | PM913 = Setting not required | PMC913 = 1 | – | PFC913 = 1 | **Note 2** |
| P914 | TIAA51 | Input | P914 = Setting not required | PM914 = Setting not required | PMC914 = 1 | PFCE914 = 0 | PFC914 = 0 | |
| | TOAA51 | Output | P914 = Setting not required | PM914 = Setting not required | PMC914 = 1 | PFCE914 = 0 | PFC914 = 1 | |
| | INTP17 | Input | P914 = Setting not required | PM914 = Setting not required | PMC914 = 1 | PFCE914 = 1 | PFC914 = 0 | |
| | A14[Note 1] | Output | P914 = Setting not required | PM914 = Setting not required | PMC914 = 1 | PFCE914 = 1 | PFC914 = 1 | **Note 2** |
| P915 | TIAA50 | Input | P915 = Setting not required | PM915 = Setting not required | PMC915 = 1 | PFCE915 = 0 | PFC915 = 0 | |
| | TOP50 | Output | P915 = Setting not required | PM915 = Setting not required | PMC915 = 1 | PFCE915 = 0 | PFC915 = 1 | |
| | INTP18 | Input | P915 = Setting not required | PM915 = Setting not required | PMC915 = 1 | PFCE915 = 1 | PFC915 = 0 | |
| | A15[Note 1] | Output | P915 = Setting not required | PM915 = Setting not required | PMC915 = 1 | PFCE915 = 1 | PFC915 = 1 | **Note 2** |

**Notes 1.**  V850ES/JH3-U only

　　　**2.**  When using as the A0 to A15 pins, be sure to set all 16 bits of the PMC9 register to FFFFH at once.

**Table 4-20. Using Port Pin as Alternate-Function Pin (9/10)**

| Pin Name | Alternate Function | | Pnx Bit of Pn Register | PMnx Bit of PMn Register | PMCnx Bit of PMCn Register | PFCEnx Bit of PFCEn Register | PFCnx Bit of PFCn Register | Other Bits (Registers) |
|---|---|---|---|---|---|---|---|---|
| | Name | I/O | | | | | | |
| PCM0 | $\overline{\text{WAIT}}$[Note] | Input | PCM0 = Setting not required | PMCM0 = Setting not required | PMCCM0 = 1 | – | – | |
| PCM1 | CLKOUT | Output | PCM1 = Setting not required | PMCM1 = Setting not required | PMCCM1 = 1 | – | – | |
| PCM2 | $\overline{\text{HLDAK}}$[Note] | Output | PCM2 = Setting not required | PMCM2 = Setting not required | PMCCM2 = 1 | – | – | |
| PCM3 | $\overline{\text{HLDRQ}}$[Note] | Input | PCM3 = Setting not required | PMCM3 = Setting not required | PMCCM3 = 1 | – | – | |
| PCS0 | $\overline{\text{CS0}}$[Note] | Output | PCS0 = Setting not required | PMCS0 = Setting not required | PMCCS0 = 1 | – | – | |
| PCS2 | $\overline{\text{CS2}}$[Note] | Output | PCS2 = Setting not required | PMCS2 = Setting not required | PMCCS2 = 1 | – | – | |
| PCS3 | $\overline{\text{CS3}}$[Note] | Output | PCS3 = Setting not required | PMCS3 = Setting not required | PMCCS3 = 1 | – | – | |
| PCT0 | $\overline{\text{WR0}}$ | Output | PCT0 = Setting not required | PMCT0 = Setting not required | PMCCT0 = 1 | – | – | |
| PCT1 | $\overline{\text{WR1}}$ | Output | PCT1 = Setting not required | PMCT1 = Setting not required | PMCCT1 = 1 | – | – | |
| PCT4 | $\overline{\text{RD}}$[Note] | Output | PCT4 = Setting not required | PMCT4 = Setting not required | PMCCT4 = 1 | – | – | |
| PCT6 | ASTB[Note] | Output | PCT6 = Setting not required | PMCT6 = Setting not required | PMCCT6 = 1 | – | – | |
| PDH0 | A16 | Output | PDH0 = Setting not required | PMDH0 = Setting not required | PMCDH0 = 1 | – | – | |
| PDH1 | A17 | Output | PDH1 = Setting not required | PMDH1 = Setting not required | PMCDH1 = 1 | – | – | |
| PDH2 | A18 | Output | PDH2 = Setting not required | PMDH2 = Setting not required | PMCDH2 = 1 | – | – | |
| PDH3 | A19 | Output | PDH3 = Setting not required | PMDH3 = Setting not required | PMCDH3 = 1 | – | – | |
| PDH4 | A20 | Output | PDH4 = Setting not required | PMDH4 = Setting not required | PMCDH4 = 1 | – | – | |
| PDH5 | A21 | Output | PDH5 = Setting not required | PMDH5 = Setting not required | PMCDH5 = 1 | – | – | |
| PDL0 | AD0 | I/O | PDL0 = Setting not required | PMDL0 = Setting not required | PMCDL0 = 1 | – | – | |
| PDL1 | AD1 | I/O | PDL1 = Setting not required | PMDL1 = Setting not required | PMCDL1 = 1 | – | – | |
| PDL2 | AD2 | I/O | PDL2 = Setting not required | PMDL2 = Setting not required | PMCDL2 = 1 | – | – | |
| PDL3 | AD3 | I/O | PDL3 = Setting not required | PMDL3 = Setting not required | PMCDL3 = 1 | – | – | |
| PDL4 | AD4 | I/O | PDL4 = Setting not required | PMDL4 = Setting not required | PMCDL4 = 1 | – | – | |

**Note** V850ES/JH3-U only

**Table 4-20. Using Port Pin as Alternate-Function Pin (10/10)**

| Pin Name | Alternate Function | | Pnx Bit of Pn Register | PMnx Bit of PMn Register | PMCnx Bit of PMCn Register | PFCEnx Bit of PFCEn Register | PFCnx Bit of PFCn Register | Other Bits (Registers) |
|---|---|---|---|---|---|---|---|---|
| | Name | I/O | | | | | | |
| PDL5 | AD5 | I/O | PDL5 = Setting not required | PMDL5 = Setting not required | PMCDL5 = 1 | – | – | |
| | FLMD1[Note] | Input | PDL5 = Setting not required | PMDL5 = Setting not required | PMCDL5 = Setting not required | – | – | |
| PDL6 | AD6 | I/O | PDL6 = Setting not required | PMDL6 = Setting not required | PMCDL6 = 1 | – | – | |
| PDL7 | AD7 | I/O | PDL7 = Setting not required | PMDL7 = Setting not required | PMCDL7 = 1 | – | – | |
| PDL8 | AD8 | I/O | PDL8 = Setting not required | PMDL8 = Setting not required | PMCDL8 = 1 | – | – | |
| PDL9 | AD9 | I/O | PDL9 = Setting not required | PMDL9 = Setting not required | PMCDL9 = 1 | – | – | |
| PDL10 | AD10 | I/O | PDL10 = Setting not required | PMDL10 = Setting not required | PMCDL10 = 1 | – | – | |
| PDL11 | AD11 | I/O | PDL11 = Setting not required | PMDL11 = Setting not required | PMCDL11 = 1 | – | – | |
| PDL12 | AD12 | I/O | PDL12 = Setting not required | PMDL12 = Setting not required | PMCDL12 = 1 | – | – | |
| PDL13 | AD13 | I/O | PDL13 = Setting not required | PMDL13 = Setting not required | PMCDL13 = 1 | – | – | |
| PDL14 | AD14 | I/O | PDL14 = Setting not required | PMDL14 = Setting not required | PMCDL14 = 1 | – | – | |
| PDL15 | AD15 | I/O | PDL15 = Setting not required | PMDL15 = Setting not required | PMCDL15 = 1 | – | – | |

**Note** Since this pin is set in the flash memory programming mode, it does not need to be manipulated using the port control register. For details, see **CHAPTER 31 FLASH MEMORY**.

## 4.5 Cautions

### 4.5.1 Cautions on setting port pins

(1) In the V850ES/JG3-U and V850ES/JH3-U, the general-purpose port functions share pins with several peripheral function I/O pins. Switch between the general-purpose port (port mode) and the peripheral function I/O pin (alternate-function mode) by setting the PMCn register. Note the following cautions with regards to this register setting sequence.

(a) Cautions on switching from port mode to alternate-function mode
Switch from the port mode to alternate-function mode in the following order.

&lt;1&gt; Set the PFn register[Note 1]:                N-ch open-drain setting
&lt;2&gt; Set the PFCn and PFCEn registers:        Alternate-function selection
&lt;3&gt; Set the corresponding bit of the PMCn register to 1:  Switch to alternate-function mode
&lt;4&gt; Set the INTRn and INTFn registers[Note 2]:  External interrupt setting

If the PMCn register is set first, note that unexpected operations may occur at that moment or depending on the change of the pin states in accordance with the setting of the PFn, PFCn, and PFCEn registers.
A concrete example is shown in [Example] below.

**Notes 1.** N-ch open-drain output pin only
   **2.** Only when the external interrupt function is selected

**Caution  Regardless of the port mode/alternate-function mode, the Pn register is read and written as follows.**
- **Pn register read:  Read the port output latch value (when PMn.PMnm bit = 0), or read the pin states (PMn.PMnm bit = 1).**
- **Pn register write:  Write to the port output latch**

[Example]  SCL01 pin setting example
The SCL01 pin is used alternately as the P41/SOF0 pin. Select the valid pin function using the PMC4, PFC4, and PF4 registers.

| PMC41 Bit | PFC41 Bit | PF41 Bit | Valid Pin Function |
|---|---|---|---|
| 0 | don't care | 1 | P41 (in output port mode, N-ch open-drain output) |
| 1 | 0 | 1 | SOF0 output (N-ch open-drain output) |
|   | 1 | 1 | SCL01 I/O (N-ch open-drain output) |

The setting procedure that may cause malfunction on switching from the P41 pin to the SCL01 pin is shown below.

| Setting Procedure | Setting Contents | Pin State | Pin Level |
|---|---|---|---|
| <1> | Initial value (PMC41 bit = 0, PFC41 bit = 0, PF41 bit = 0) | Port mode (input) | Hi-Z |
| <2> | PMC41 bit ← 1 | SOF0 output | Low level (high level depending on the CSIF0 setting) |
| <3> | PFC41 bit ← 1 | SCL01 I/O | High level (CMOS output) |
| <4> | PF41 bit ← 1 | SCL01 I/O | Hi-Z (N-ch open-drain output) |

In <2>, I²C communication may be affected since the alternate-function SOF0 output is output to the pin. In the CMOS output period of <2> or <3>, unnecessary current may be generated.

(b) Cautions on alternate-function mode (input)

The signal input to the alternate-function block is low level when the PMCn.PMCnm bit is 0 due to the AND output of the PMCn register set value and the pin level. Thus, depending on the port setting and alternate-function operation enable timing, unexpected operations may occur. Therefore, switch between the port mode and alternate-function mode in the following sequence.

- To switch from port mode to alternate-function mode (input)

  Set the pins to the alternate-function mode using the PMCn register and then enable the alternate-function operation.

- To switch from alternate-function mode (input) to port mode

  Stop the alternate-function operation and then switch the pins to the port mode.

Concrete examples are shown in [Example 1] and [Example 2].

[Example 1]   Switching from general-purpose port (P02) to external interrupt pin (NMI)

When the P02/NMI pin is pulled up as shown in Figure 4-4 and the rising edge is specified by the NMI pin edge detection setting, even though a high level is input continuously to the NMI pin when switching from the P02 pin to the an NMI pin (PMC02 bit = 0 → 1), this is detected as a rising edge as if a low level changed to a high level, and an NMI interrupt occurs.

To avoid this, set the NMI pin's valid edge after switching from the P02 pin to the NMI pin.

**Figure 4-4.  Example of Switching from P02 to NMI (Incorrect)**



**Remark**   m = 2 to 6
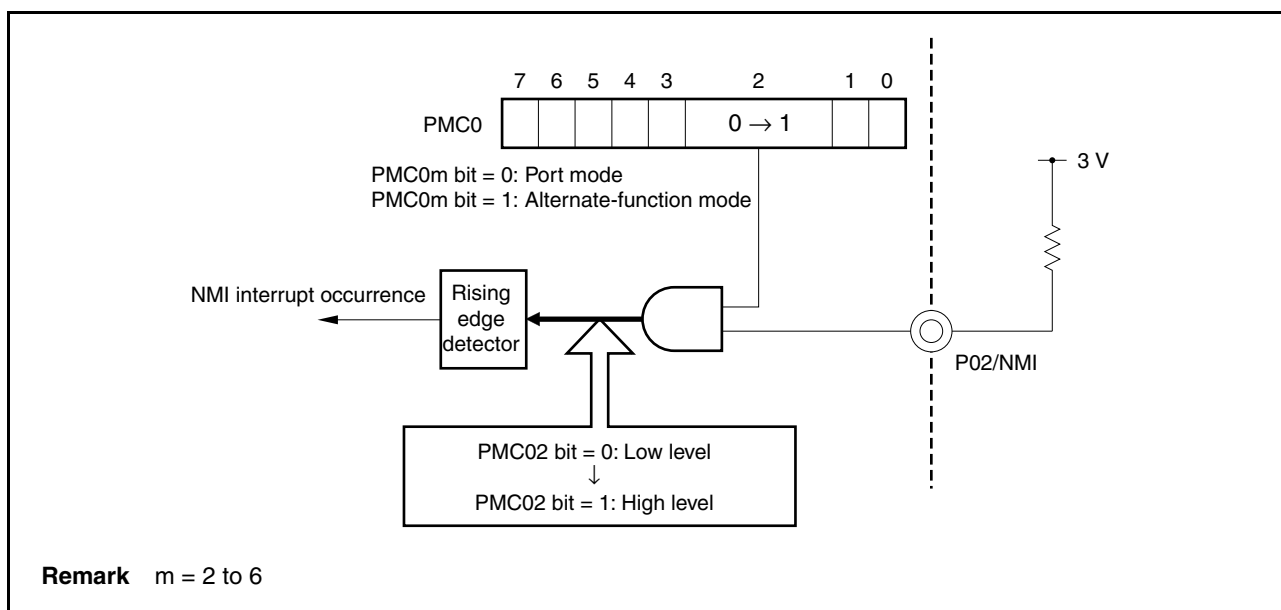
[Example 2]   Switching from external pin (NMI) to general-purpose port (P02)

When the P02/NMI pin is pulled up as shown in Figure 4-5 and the falling edge is specified by the NMI pin edge detection setting, even though a high level is input continuously to the NMI pin when switching from the NMI pin to the P02 pin (PMC02 bit = 1 → 0), this is detected as a falling edge as if a high level changed to a low level, and an NMI interrupt occurs.

To avoid this, set the NMI pin edge detection as "No edge detected" before switching to the P02 pin.

**Figure 4-5.  Example of Switching from NMI to P02 (Incorrect)**



**Remark**   m = 2 to 6
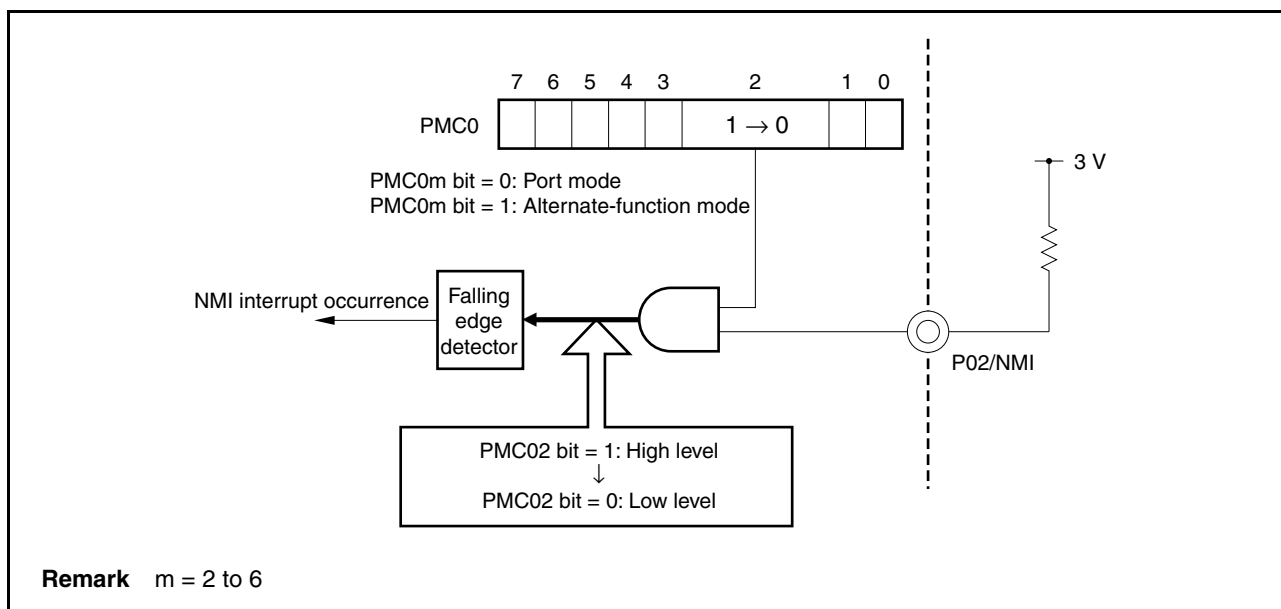
(2)  In port mode, the PFn.PFnm bit is valid only in the output mode (PMn.PMnm bit = 0).  In the input mode (PMnm bit = 1), the value of the PFnm bit is not reflected in the buffer.

### 4.5.2  Cautions on bit manipulation instruction for port n register (Pn)

When a 1-bit manipulation instruction is executed on a port that provides both input and output functions, the value of the output latch of an input port that is not subject to manipulation may be written in addition to the targeted bit.

Therefore, it is recommended to rewrite the output latch when switching a port from input mode to output mode.

<Example>    When the P90 pin is an output port, the P91 to P97 pins are input ports (all pin statuses are high level), and the value of the port latch is 00H, if the output of the P90 pin is changed from low level to high level via a bit manipulation instruction, the value of the port latch is FFH.

Explanation: The targets of writing to and reading from the Pn register of a port whose PMnm bit is 1 are the output latch and pin status, respectively.

A bit manipulation instruction is executed in the following order in the V850ES/JG3-U and V850ES/JH3-U.

<1>  The Pn register is read in 8-bit units.
<2>  The targeted bit is manipulated.
<3>  The Pn register is written in 8-bit units.

In step <1>, the value of the output latch (0) of the P90 pin, which is an output port, is read, while the pin statuses of the P91 to P97 pins, which are input ports, are read.  If the pin statuses of the P91 to P97 pins are high level at this time, the read value is FEH.

The value is changed to FFH by the manipulation in <2>.

FFH is written to the output latch by the manipulation in <3>.

**Figure 4-6.  Bit Manipulation Instruction (P90 Pin)**
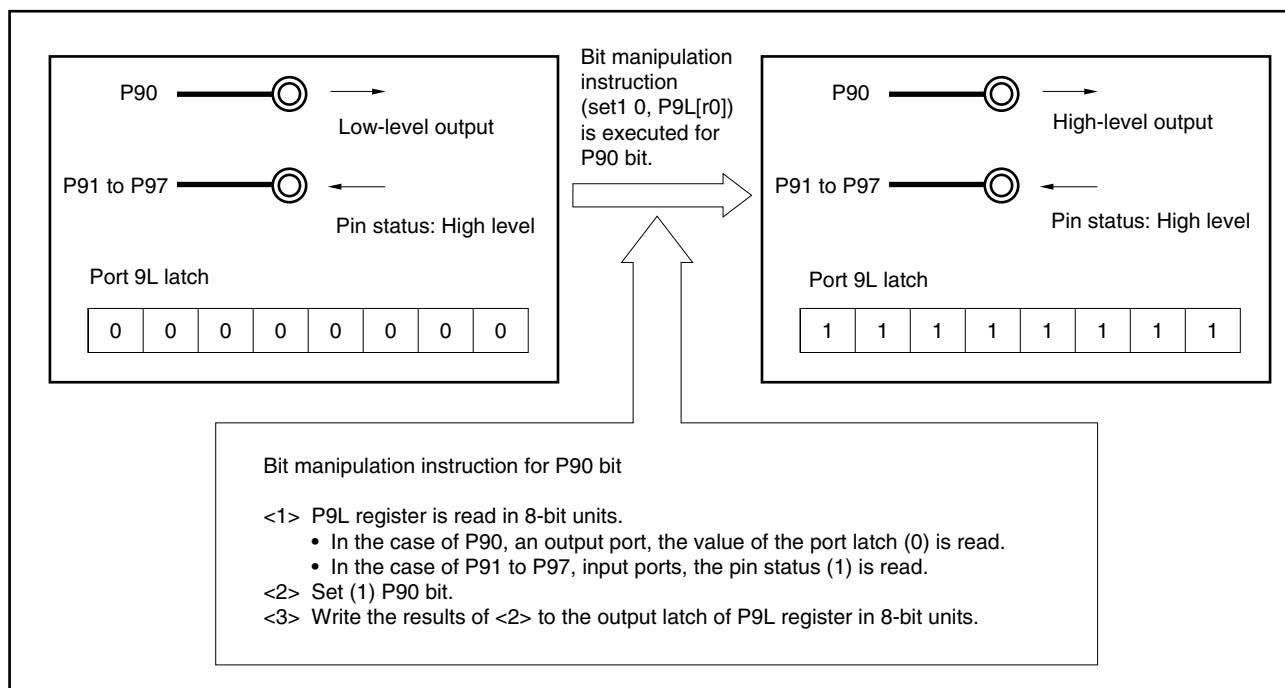
### 4.5.3  Cautions on on-chip debug pins (V850ES/JG3-U only)

The $\overline{\text{DRST}}$, DCK, DMS, DDI, and DDO pins are on-chip debug pins.

After reset by the $\overline{\text{RESET}}$ pin, the P56/INTP05/$\overline{\text{DRST}}$ pin is initialized to function as an on-chip debug pin ($\overline{\text{DRST}}$).  If a high level is input to the $\overline{\text{DRST}}$ pin at this time, the on-chip debug mode is set, and the DCK, DMS, DDI, and DDO pins can be used.

The following action must be taken if on-chip debugging is not used.

- Clear the OCDM0 bit of the OCDM register (special register) (0)

At this time, fix the P56/INTP05/$\overline{\text{DRST}}$ pin to low level from when reset by the $\overline{\text{RESET}}$ pin is released until the above action is taken.

If a high level is input to the $\overline{\text{DRST}}$ pin before the above action is taken, it may cause a malfunction (CPU deadlock). Handle the P56 pin with the utmost care.

**Caution  After reset by the WDT2RES signal, clock monitor (CLM), or low-voltage detector (LVI), the P56/INTP05/$\overline{\text{DRST}}$ pin is not initialized to function as an on-chip debug pin ($\overline{\text{DRST}}$).  The OCDM register holds the current value.**

### 4.5.4  Cautions on P56/INTP05/$\overline{\text{DRST}}$ pin

The P56/INTP05/$\overline{\text{DRST}}$ pin has an internal pull-down resistor (30 kΩ TYP.).  After a reset by the $\overline{\text{RESET}}$ pin, a pull-down resistor is connected.  The pull-down resistor is disconnected when the OCDM0 bit is cleared (0).

### 4.5.5  Cautions on P10, P11, and P53 pins when power is turned on

When the power is turned on, the following pins may output an undefined level temporarily even during reset.

- P10/ANO0 pin
- P11/ANO1 pin
- P53/SIF2/TIAB00/KR3/TOAB00/RTP03/DDO pin (V850ES/JG3-U only)

### 4.5.6  Hysteresis characteristics

In port mode, the following port pins do not have hysteresis characteristics.

P00 to P05
P20 to P25
P30 to P37
P40 to P42
P50 to P56
P60 to P65
P90 to P915

## CHAPTER 5  BUS CONTROL FUNCTION

The V850ES/JG3-U and V850ES/JH3-U are provided with an external bus interface function by which external memories such as ROM and RAM, and I/O can be connected.

## 5.1  Features

○ Output is selectable from multiplexed bus output with a minimum of 3 bus cycles and separate bus output
  (V850ES/JH3-U only; the V850ES/JG3-U only supports the multiplexed bus.)
○ 8-bit/16-bit data bus selectable
○ Wait function
  • Programmable wait function of up to 7 states
  • External wait function using $\overline{\text{WAIT}}$ pin
○ Idle state function
○ Bus hold function

## 5.2  Bus Control Pins

The pins used to connect an external device are listed in the table below.

**Table 5-1.  V850ES/JH3-U Bus Control Pins (Multiplexed Bus)**

| Bus Control Pin | Alternate-Function Pin | I/O | Function |
|---|---|---|---|
| AD0 to AD15 | PDL0 to PDL15 | I/O | Address/data bus |
| A16 to A23 | PDH0 to PDH5 | Output | Address bus |
| $\overline{\text{WAIT}}$ | PCM0 | Input | External wait control |
| CLKOUT | PCM1 | Output | Internal system clock |
| $\overline{\text{WR0}}$, $\overline{\text{WR1}}$ | PCT0, PCT1 | Output | Write strobe signal |
| $\overline{\text{RD}}$ | PCT4 | Output | Read strobe signal |
| ASTB | PCT6 | Output | Address strobe signal |
| $\overline{\text{HLDRQ}}$ | PCM3 | Input | Bus hold control |
| $\overline{\text{HLDAK}}$ | PCM2 | Output | |
| $\overline{\text{CS0}}$, $\overline{\text{CS2}}$, $\overline{\text{CS3}}$ | PCS0, PCS2, PCS3 | Output | Chip select |

**Table 5-2.  V850ES/JH3-U Bus Control Pins (Separate Bus)**

| Bus Control Pin | Alternate-Function Pin | I/O | Function |
|---|---|---|---|
| AD0 to AD15 | PDL0 to PDL15 | I/O | Data bus |
| A0 to A15 | P90 to P915 | Output | Address bus |
| A16 to A21 | PDH0 to PDH5 | Output | Address bus |
| $\overline{\text{WAIT}}$ | PCM0 | Input | External wait control |
| CLKOUT | PCM1 | Output | Internal system clock |
| $\overline{\text{WR0}}$, $\overline{\text{WR1}}$ | PCT0, PCT1 | Output | Write strobe signal |
| $\overline{\text{RD}}$ | PCT4 | Output | Read strobe signal |
| $\overline{\text{HLDRQ}}$ | PCM3 | Input | Bus hold control |
| $\overline{\text{HLDAK}}$ | PCM2 | Output | |
| $\overline{\text{CS0}}$, $\overline{\text{CS2}}$, $\overline{\text{CS3}}$ | PCS0, PCS2, PCS3 | Output | Chip select |

**Table 5-3.  V850ES/JG3-U Bus Control Pins (Multiplexed Bus)**

| Bus Control Pin | Alternate-Function Pin | I/O | Function |
|---|---|---|---|
| AD0 to AD15 | PDL0 to PDL15 | I/O | Address/data bus |
| $\overline{\text{WAIT}}$ | PCM0 | Input | External wait control |
| CLKOUT | PCM1 | Output | Internal system clock |
| $\overline{\text{WR0}}$, $\overline{\text{WR1}}$ | PCT0, PCT1 | Output | Write strobe signal |
| $\overline{\text{RD}}$ | P61 | Output | Read strobe signal |
| ASTB | P62 | Output | Address strobe signal |
| $\overline{\text{CS0}}$, $\overline{\text{CS2}}$, $\overline{\text{CS3}}$ | P63, P64, P65 | Output | Chip select |

**5.2.1 Pin status when internal ROM, internal RAM, or on-chip peripheral I/O is accessed**

When the internal ROM, internal RAM, or on-chip peripheral I/O are accessed, the status of each pin is as follows.

**Table 5-4. Pin Statuses When Internal ROM, Internal RAM, or On-Chip Peripheral I/O Is Accessed**

| Bus Control Pin | Separate Bus Mode | | Multiplexed Bus Mode | |
|---|---|---|---|---|
| | Internal ROM/RAM | Peripheral I/O | Internal ROM/RAM | Peripheral I/O |
| Address/data bus (AD15 to AD0) | Undefined | Undefined | Undefined | Undefined |
| Address bus (A23 to A16) | Undefined | Undefined (Address output during access) | Undefined | Undefined (Address output during access) |
| Address bus (A15 to A0) | Undefined | Undefined (Address output during access) | Undefined | Undefined (Address output during access) |
| Control signal | Inactive | Inactive | Inactive | Inactive |

**Caution  When a write access is performed to the internal ROM area, address, data, and control signals are activated in the same way as access to the external memory area.**

**5.2.2 Pin status in each operation mode**

For the pin status of the V850ES/JG3-U and V850ES/JH3-U in each operation mode, see **2.2 Pin States**.

## 5.3  Memory Block Function

The 16 MB external memory space is divided into memory blocks of 2 MB, 4 MB, and 8 MB from the lowest of the memory space.  The programmable wait function and bus cycle operation mode for each of these blocks can be independently controlled in one-block units.

**Figure 5-1.  Data Memory Map: Physical Address**



Notes **1.** $\overline{\text{CS1}}$ is not provided as an external signal of the V850ES/Jx3-U; it is used internally as a chip select signal for the USB.

**2.** Use of addresses 03FEF000H to 03FEFFFFH is prohibited because these addresses are in the same area as the on-chip peripheral I/O area.

**3.** This area is an external memory area in the case of a data write access.
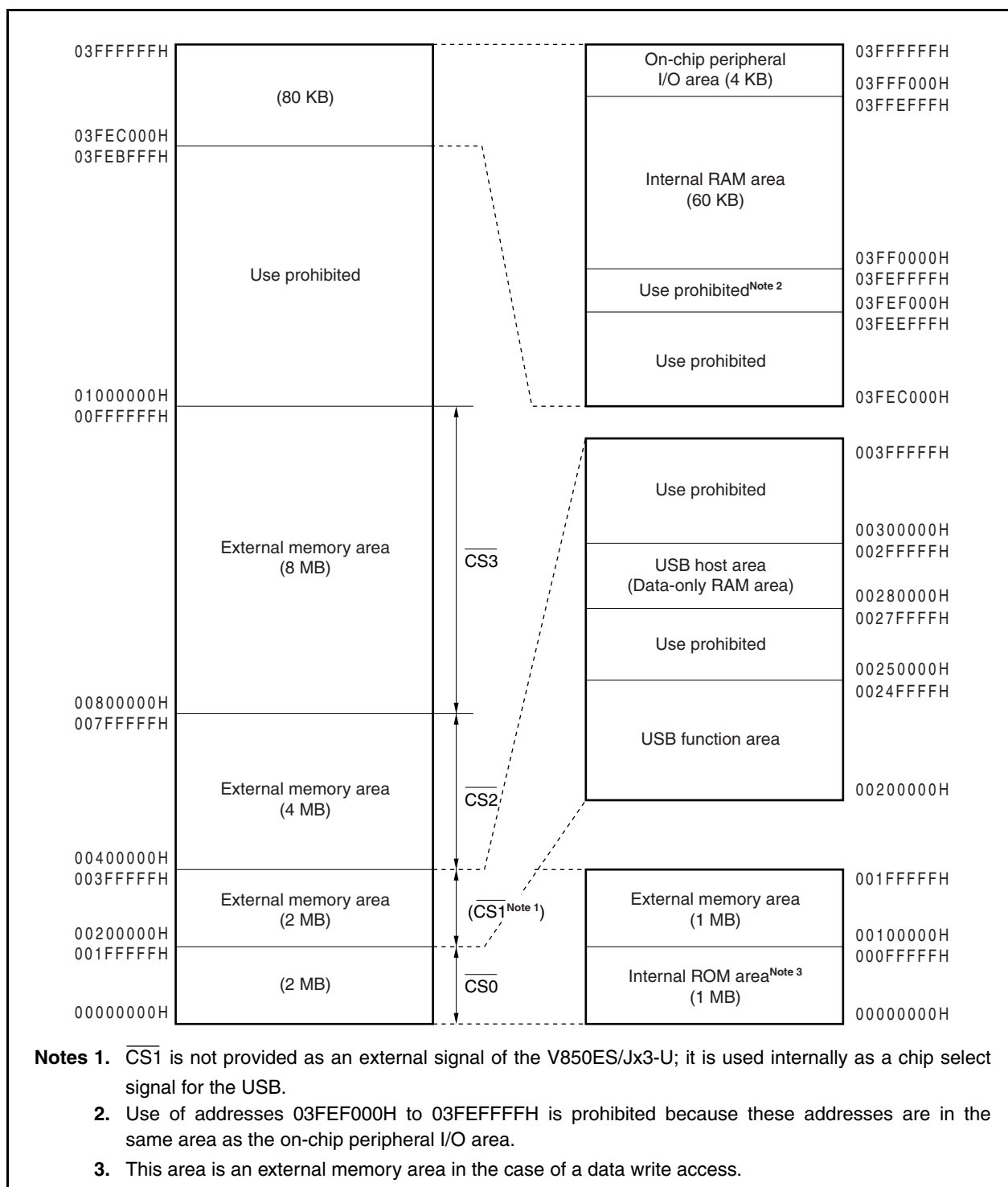
## 5.4  Bus Access

### 5.4.1  Number of clocks for access

The following table shows the number of basic clocks required for accessing each resource.

| Bus Cycle Type \ Area (Bus Width) | Internal ROM (32 Bits) | Internal RAM (32 Bits) | External Memory (16 Bits) | |
|---|---|---|---|---|
| | | | Multiplexed | Separate[Note 1] |
| Instruction fetch (normal access) | 1 | 1[Note 2] | 3 + n | |
| Instruction fetch (branch) | 3 | 2[Note 1] | 3 + n | |
| Operand data access | 5 | 1 | 3 + n | |

**Notes 1.** V850ES/JH3-U only

     **2.** Increases by 1 if a conflict with a data access occurs.

**Remark**  Unit: Clocks/access

### 5.4.2  Bus size setting function

Each external memory area selected by memory block $\overline{CSn}$ can be set by using the BSC register.  However, the bus size can be set to 8 bits and 16 bits only.

The external memory area of the V850ES/JG3-U and V850ES/JH3-U is selected by memory blocks $\overline{CS0}$, $\overline{CS2}$, and $\overline{CS3}$.

#### (1)  Bus size configuration register (BSC)

The BSC register can be read or written in 16-bit units.

Reset sets this register to 5555H.

**Caution**  Write to the BSC register after reset, and then do not change the set values.  Also, do not access an external memory area until the initial settings of the BSC register are complete.

After reset: 5555H    R/W    Address: FFFFF066H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| BSC | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | BS30 | 0 | BS20 | 0 | 1 | 0 | BS00 |
| | | $\overline{CS3}$ | | $\overline{CS2}$ | | | | $\overline{CS0}$ |

| BSn0 | Data bus width of memory block CSn space (n = 0, 2, 3) |
|---|---|
| 0 | 8 bits |
| 1 | 16 bits |

**Caution**  Be sure to set bits 14, 12, 10, 8, and 2 to "1", and clear bits 15, 13, 11, 9, 7, 5, 3, and 1 to "0".

**5.4.3  Access by bus size**

The V850ES/JG3-U and V850ES/JH3-U access the on-chip peripheral I/O and external memory in 8-bit, 16-bit, or 32-bit units. The bus size is as follows.

- The bus size of the on-chip peripheral I/O is fixed to 16 bits.
- The bus size of the external memory is selectable from 8 bits or 16 bits (by using the BSC register).

The operation when each of the above is accessed is described below. All data is accessed starting from the lower side. The V850ES/JG3-U and V850ES/JH3-U support only the little-endian format.

**Figure 5-2.  Little-Endian Address in Word**

| 31 | 24 23 | 16 15 | 8 7 | 0 |
|---|---|---|---|---|
| 000BH | 000AH | 0009H | 0008H | |
| 0007H | 0006H | 0005H | 0004H | |
| 0003H | 0002H | 0001H | 0000H | |
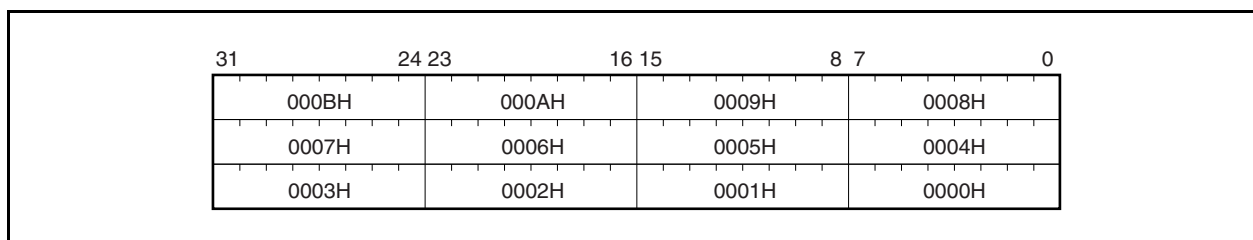
**(1) Data space**

The V850ES/JG3-U and V850ES/JH3-U have an address misalign function.

With this function, data can be placed at all addresses, regardless of the format of the data (word data or halfword data). However, if the word data or halfword data is not aligned at the boundary, a bus cycle is generated at least twice, causing the bus efficiency to drop.

**(a) Halfword-length data access**

A byte-length bus cycle is generated twice if the least significant bit of the address is 1.

**(b) Word-length data access**

(i) A byte-length bus cycle, halfword-length bus cycle, and byte-length bus cycle are generated in that order if the least significant bit of the address is 1.

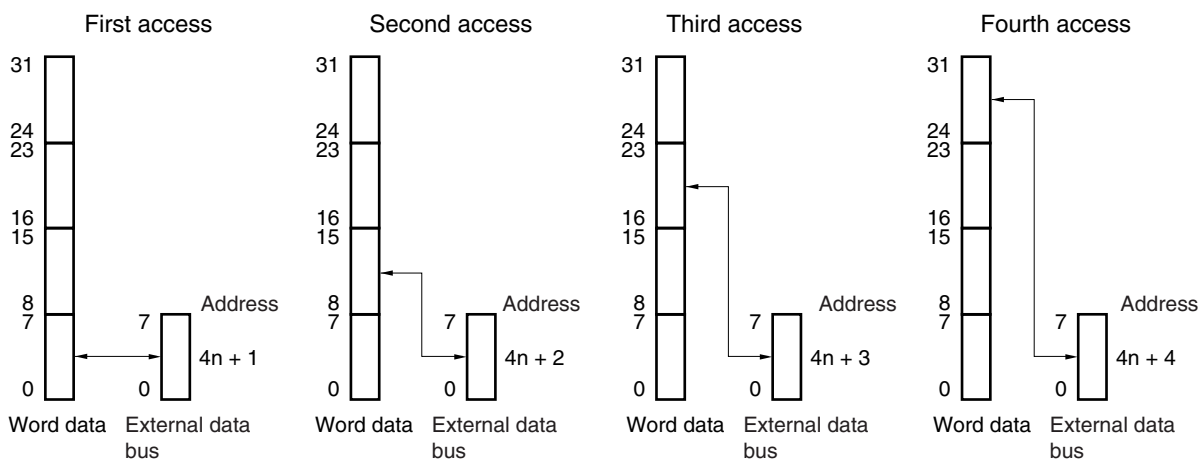(ii) A halfword-length bus cycle is generated twice if the lower 2 bits of the address are 10.

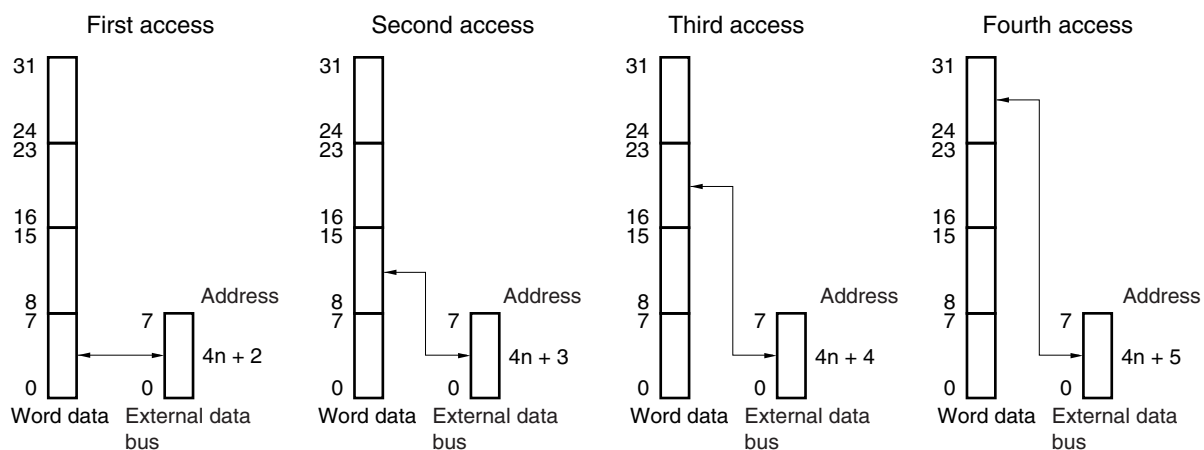**(2) Byte access (8 bits)**

**(a) 16-bit data bus width**

<1>  **Access to even address (2n)**                 <2>  **Access to odd address (2n + 1)**



**(b) 8-bit data bus width**

<1>  **Access to even address (2n)**                 <2>  **Access to odd address (2n + 1)**
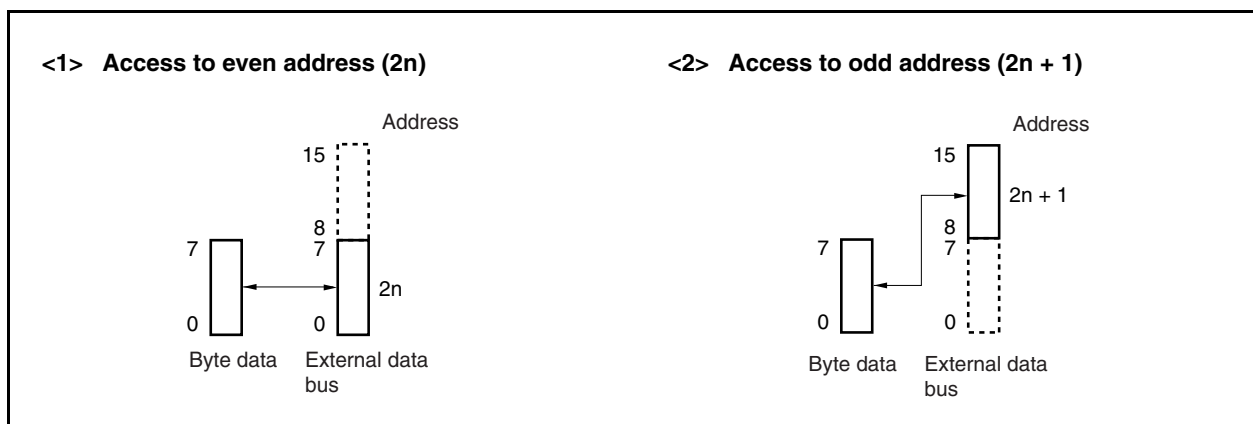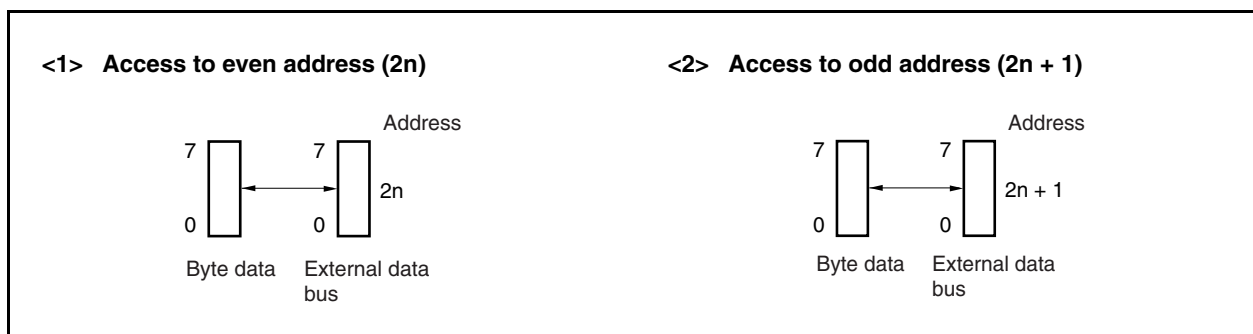
**(3) Halfword access (16 bits)**

**(a) 16-bit data bus width**

**<1> Access to even address (2n)**

**<2> Access to odd address (2n + 1)**

**(b) 8-bit data bus width**

**<1> Access to even address (2n)**

**<2> Access to odd address (2n + 1)**
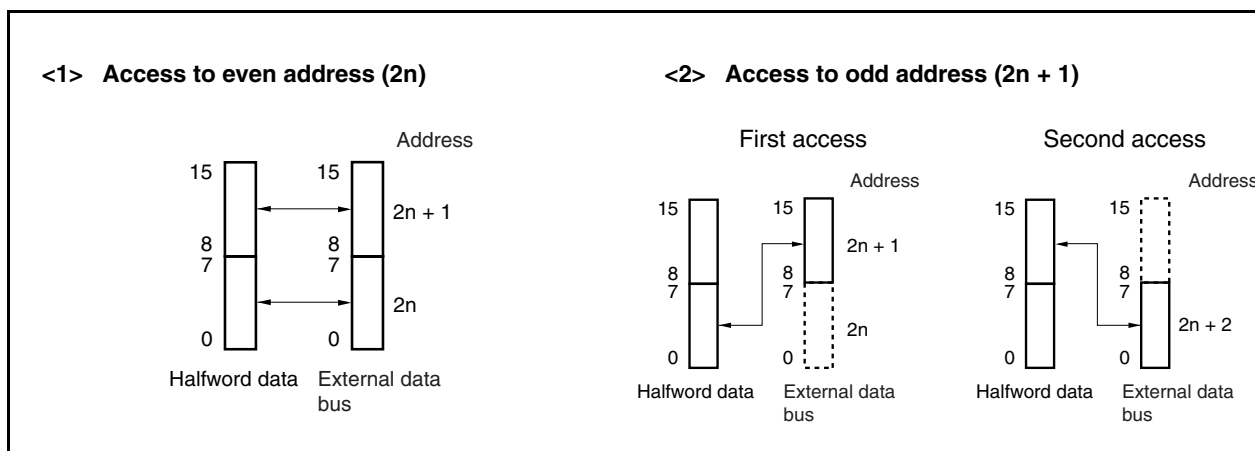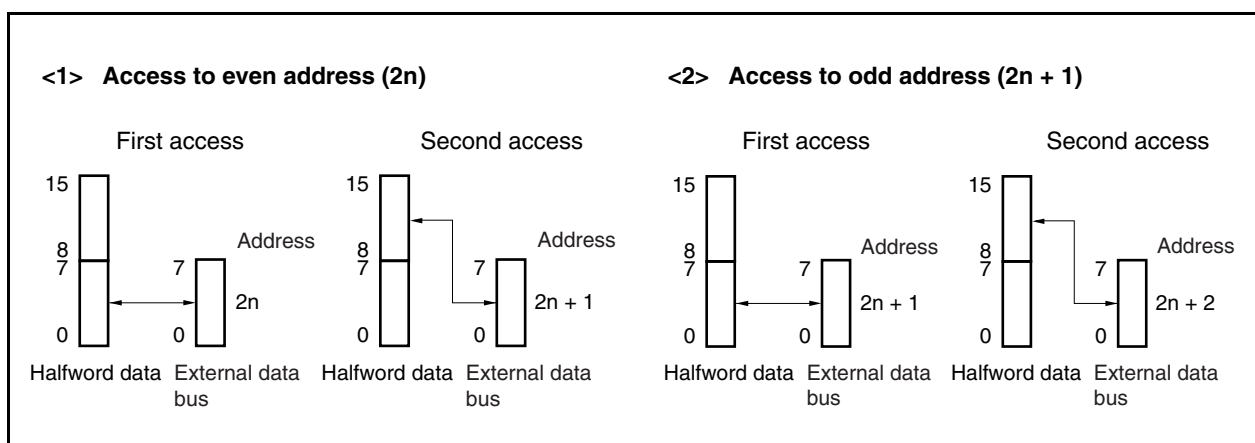
**(4)  Word access (32 bits)**

**(a)  16-bit data bus width (1/2)**

**<1>  Access to address (4n)**
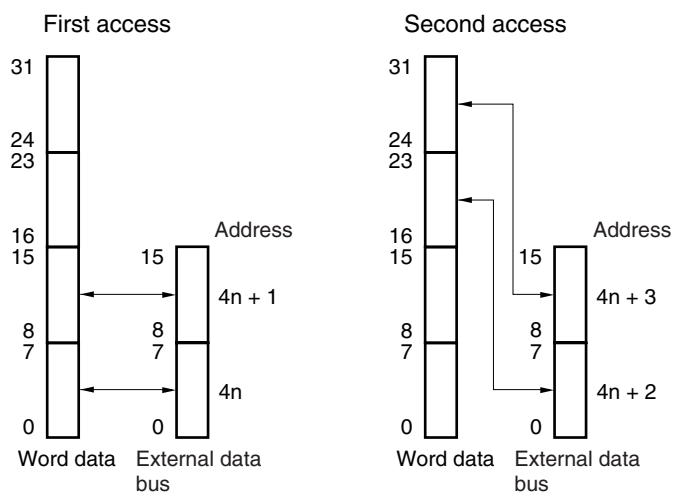


**<2>  Access to address (4n + 1)**
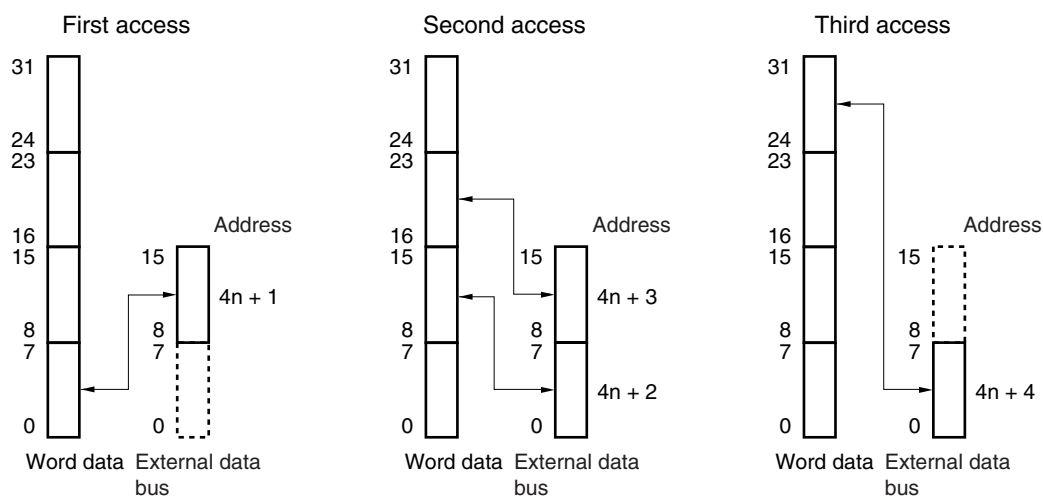
**(a)  16-bit data bus width (2/2)**

**<3>  Access to address (4n + 2)**
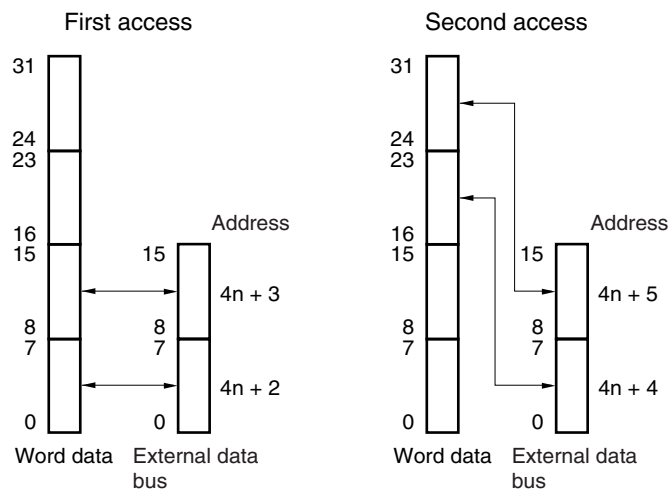


**<4>  Access to address (4n + 3)**
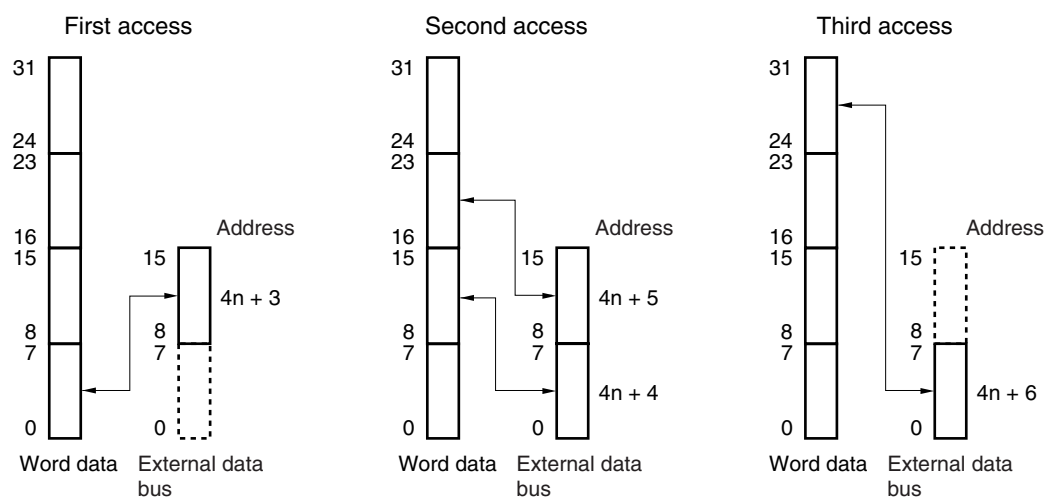
**(b)  8-bit data bus width (1/2)**

**<1>  Access to address (4n)**
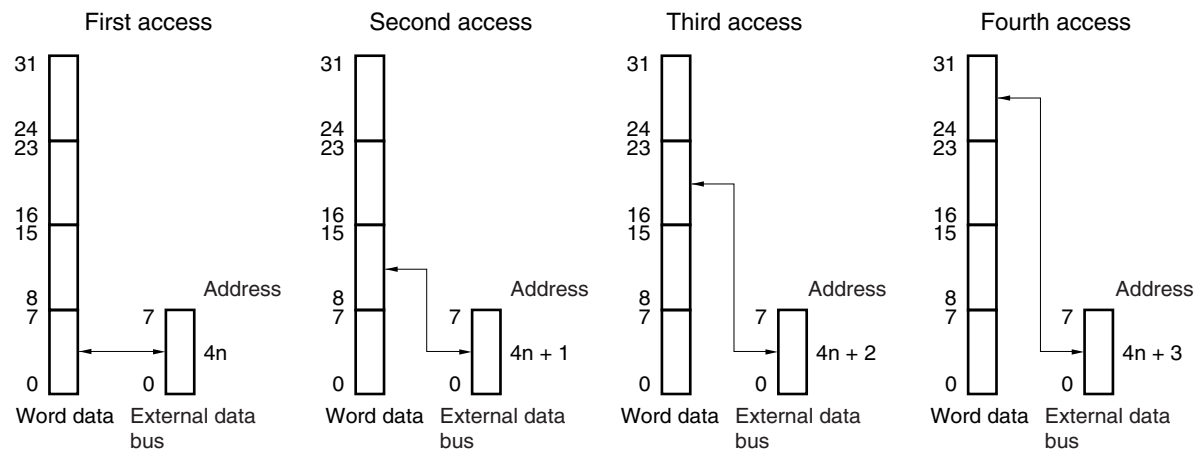


**<2>  Access to address (4n + 1)**

**(b)  8-bit data bus width (2/2)**

**<3>  Access to address (4n + 2)**

| First access | Second access | Third access | Fourth access |



**<4>  Access to address (4n + 3)**

| First access | Second access | Third access | Fourth access |
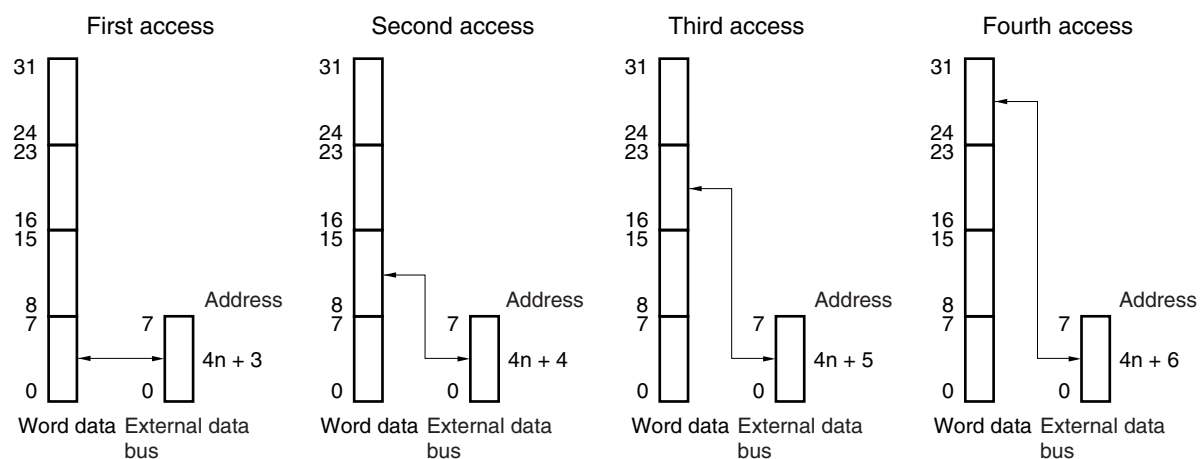
## 5.5  Wait Function

### 5.5.1  Programmable wait function

**(1) Data wait control register 0 (DWC0)**

To realize interfacing with a low-speed memory or I/O, up to seven data wait states can be inserted in the bus cycle that is executed for each CS space.

The number of wait states can be programmed by using the DWC0 register.  Immediately after system reset, 7 data wait states are inserted for all the blocks.

The DWC0 register can be read or written in 16-bit units.

Reset sets this register to 7777H.

**Cautions 1.  The internal ROM and internal RAM areas are not subject to programmable wait, and are always accessed without a wait state.  The on-chip peripheral I/O area is also not subject to programmable wait, and only wait control from each peripheral function is performed.**

**2.  Write to the DWC0 register after reset, and then do not change the set values.  Also, do not access an external memory area until the initial settings of the DWC0 register are complete.**

After reset: 7777H      R/W      Address: FFFFF484H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| DWC0 | 0 | DW32 | DW31 | DW30 | 0 | DW22 | DW21 | DW20 |
| | | | $\overline{\text{CS3}}$ | | | | $\overline{\text{CS2}}$ | |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | DW12[Note] | DW11[Note] | DW10[Note] | 0 | DW02 | DW01 | DW00 |
| | | | | | | | $\overline{\text{CS0}}$ | |

| DWn2 | DWn1 | DWn0 | Number of wait states inserted in CSn space (n = 0 to 3) |
|---|---|---|---|
| 0 | 0 | 0 | None |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 2 |
| 0 | 1 | 1 | 3 |
| 1 | 0 | 0 | 4 |
| 1 | 0 | 1 | 5 |
| 1 | 1 | 0 | 6 |
| 1 | 1 | 1 | 7 |
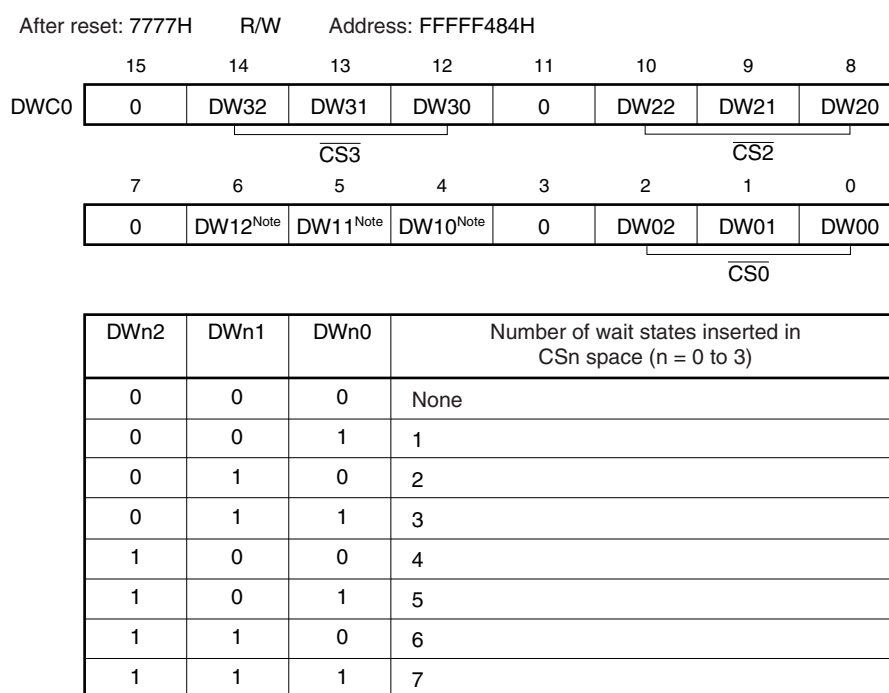
**Note**  The DW12 to DW10 bits set wait of access to the USB function area.

It is recommended to set the DW12 to DW10 bits to 001B (1 wait).

**Caution  Be sure to set bits 6 to 4 to "1", and be sure to clear bits 15, 11, 7, and 3 to "0".**

### 5.5.2  External wait function

To synchronize an extremely slow external memory, I/O, or asynchronous system, any number of wait states can be inserted in the bus cycle by using the external wait pin ($\overline{\text{WAIT}}$).

When the P60[Note 1] or PCM0[Note 2] pin is set to its alternate function, the external wait function is enabled.

Access to each area of the internal ROM, internal RAM, and on-chip peripheral I/O is not subject to control by the external wait function, in the same manner as the programmable wait function.

The $\overline{\text{WAIT}}$ signal can be input asynchronously to CLKOUT, and is sampled at the falling edge of the clock in the T2 and TW states of the bus cycle. If the setup/hold time of the sampling timing is not satisfied, a wait state is inserted in the next state, or not inserted at all.

**Notes 1.** V850ES/JG3-U
     **2.** V850ES/JH3-U

### 5.5.3  Relationship between programmable wait and external wait

Wait cycles are inserted as the result of an OR operation between the wait cycles specified by the set value of the programmable wait and the wait cycles controlled by the $\overline{\text{WAIT}}$ pin.
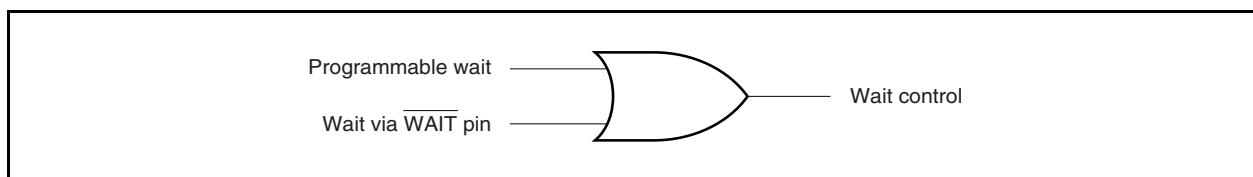


For example, if the timing of the programmable wait and the $\overline{\text{WAIT}}$ pin signal is as illustrated below, three wait states will be inserted in the bus cycle.

**Figure 5-3.  Inserting Wait Example**



**Remark**   The circles indicate the sampling timing.
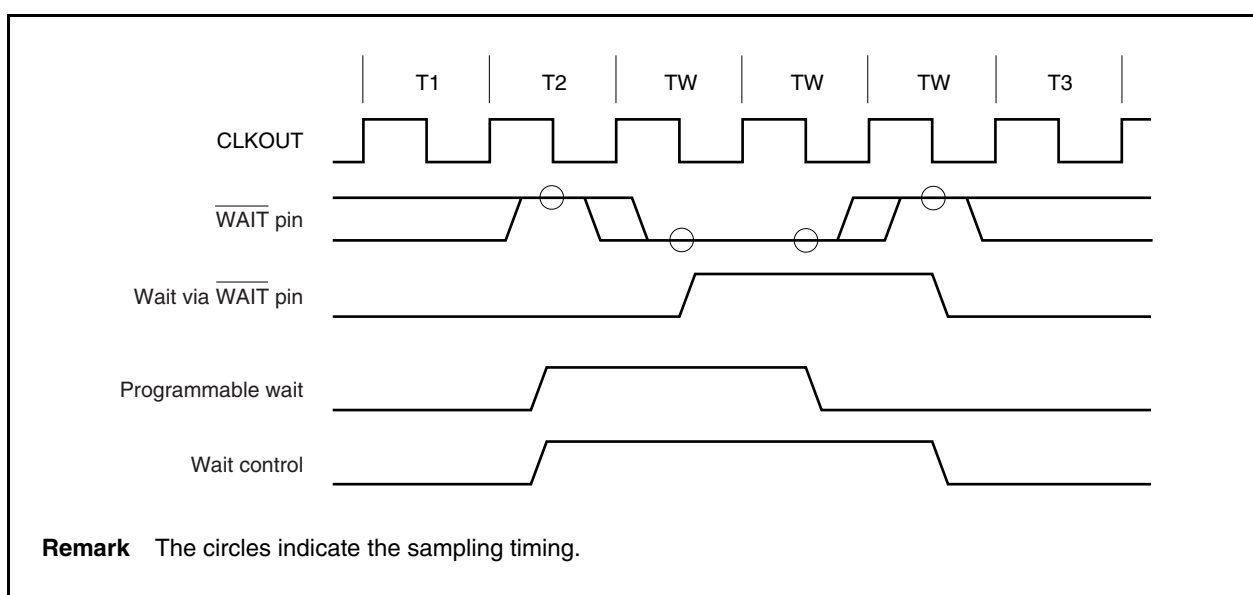
### 5.5.4  Programmable address wait function

Address-setup or address-hold waits to be inserted in each bus cycle can be set by using the AWC register. Address wait insertion is set for each chip select area ($\overline{CS0}$, $\overline{CS2}$, $\overline{CS3}$).

If an address setup wait is inserted, it seems that the high-clock period of the T1 state is extended by 1 clock. If an address-hold wait is inserted, it seems that the low-clock period of the T1 state is extended by 1 clock.

**(1)  Address wait control register (AWC)**

The AWC register can be read or written in 16-bit units.

Reset sets this register to FFFFH.

**Cautions 1.  Address-setup wait and address-hold wait cycles are not inserted when the internal ROM area, internal RAM area, and on-chip peripheral I/O areas are accessed.**

**2.  Write to the AWC register after reset, and then do not change the set values. Also, do not access an external memory area until the initial settings of the AWC register are complete.**

After reset: FFFFH       R/W       Address: FFFFF488H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| AWC | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AHW3 | ASW3 | AHW2 | ASW2 | AHW1Note | ASW1Note | AHW0 | ASW0 |

$\overline{CS3}$         $\overline{CS2}$                     $\overline{CS0}$

| AHWn | Specifies insertion of address-hold wait (n = 0 to 3) |
|---|---|
| 0 | Not inserted |
| 1 | Inserted |

| ASWn | Specifies insertion of address-setup wait (n = 0 to 3) |
|---|---|
| 0 | Not inserted |
| 1 | Inserted |

**Note**    It is recommended to clear the AHW1 bit and the ASW1 bit to 0.

**Caution**   Be sure to set bits 15 to 8 to "1".

## 5.6  Idle State Insertion Function

To facilitate interfacing with low-speed memories, one idle state (TI) can be inserted after the T3 state in the bus cycle that is executed for each space selected by the chip select.  By inserting an idle state, the data output float delay time of the memory can be secured during read access (an idle state cannot be inserted during write access).

Whether the idle state is to be inserted can be programmed by using the BCC register.

An idle state is inserted for all the areas immediately after system reset.

**(1)  Bus cycle control register (BCC)**

The BCC register can be read or written in 16-bit units.

Reset sets this register to AAAAH.

**Cautions 1.  The internal ROM, internal RAM, and on-chip peripheral I/O areas are not subject to idle state insertion.**

**2.  Write to the BCC register after reset, and then do not change the set values.  Also, do not access an external memory area until the initial settings of the BCC register are complete.**

After reset: AAAAH    R/W    Address: FFFFF48AH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|------|----|----|----|----|----|----|----|----|
| BCC | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|----|------|----|-----------|----|------|----|
| | BC31 | 0 | BC21 | 0 | BC11[Note] | 0 | BC01 | 0 |
| | CS3 | | CS2 | | | | CS0 | |

| BCn1 | Specifies insertion of idle state (n = 0 to 3) |
|------|------------------------------------------------|
| 0 | Not inserted |
| 1 | Inserted |

**Note**  It is recommended to clear the BC11 bit to 0.

**Caution**  Be sure to set bits 15, 13, and 11, 9 to "1", and clear bits 14, 12, 10, 8, 6, 4, 2, and 0 to "0".

## 5.7  Bus Hold Function (V850ES/JH3-U only)

### 5.7.1  Functional outline

The $\overline{\text{HLDRQ}}$ and $\overline{\text{HLDAK}}$ functions are valid if the PCM2 and PCM3 pins are set to their alternate function.

When the $\overline{\text{HLDRQ}}$ pin is asserted (low level), indicating that another bus master has requested bus mastership, the external address/data bus goes into a high-impedance state and is released (bus hold status).  If the request for the bus mastership is cleared and the $\overline{\text{HLDRQ}}$ pin is deasserted (high level), driving these pins is started again.

During the bus hold period, execution of the program in the internal ROM and internal RAM is continued until an on-chip peripheral I/O register or the external memory is accessed.

The bus hold status is indicated by assertion of the $\overline{\text{HLDAK}}$ pin (low level).  The bus hold function enables the configuration of multi-processor type systems in which two or more bus masters exist.

Note that the bus hold request is not acknowledged during a multiple-access cycle initiated by the bus sizing function or a bit manipulation instruction.

| Status | Data Bus Width | Access Type | Timing at Which Bus Hold Request Is Not Acknowledged |
|---|---|---|---|
| CPU bus lock | 16 bits | Word access to even address | Between first and second access |
|  |  | Word access to odd address | Between first and second access |
|  |  |  | Between second and third access |
|  |  | Halfword access to odd address | Between first and second access |
|  | 8 bits | Word access | Between first and second access |
|  |  |  | Between second and third access |
|  |  |  | Between third and fourth access |
|  |  | Halfword access | Between first and second access |
| Read-modify-write access of bit manipulation instruction | – | – | Between read access and write access |

### 5.7.2 Bus hold procedure

The bus hold status transition procedure is shown below.

<1> $\overline{\text{HLDRQ}}$ = 0 acknowledged

<2> All bus cycle start requests inhibited

<3> End of current bus cycle

<4> Shift to bus idle status

<5> $\overline{\text{HLDAK}}$ = 0

Normal status

Bus hold status

<6> $\overline{\text{HLDRQ}}$ = 1 acknowledged

<7> $\overline{\text{HLDAK}}$ = 1

<8> Bus cycle start request inhibition released

<9> Bus cycle starts

Normal status

$\overline{\text{HLDRQ}}$ (input)

$\overline{\text{HLDAK}}$ (output)

<1>  <2>  <3><4>  <5>                <6>  <7><8><9>
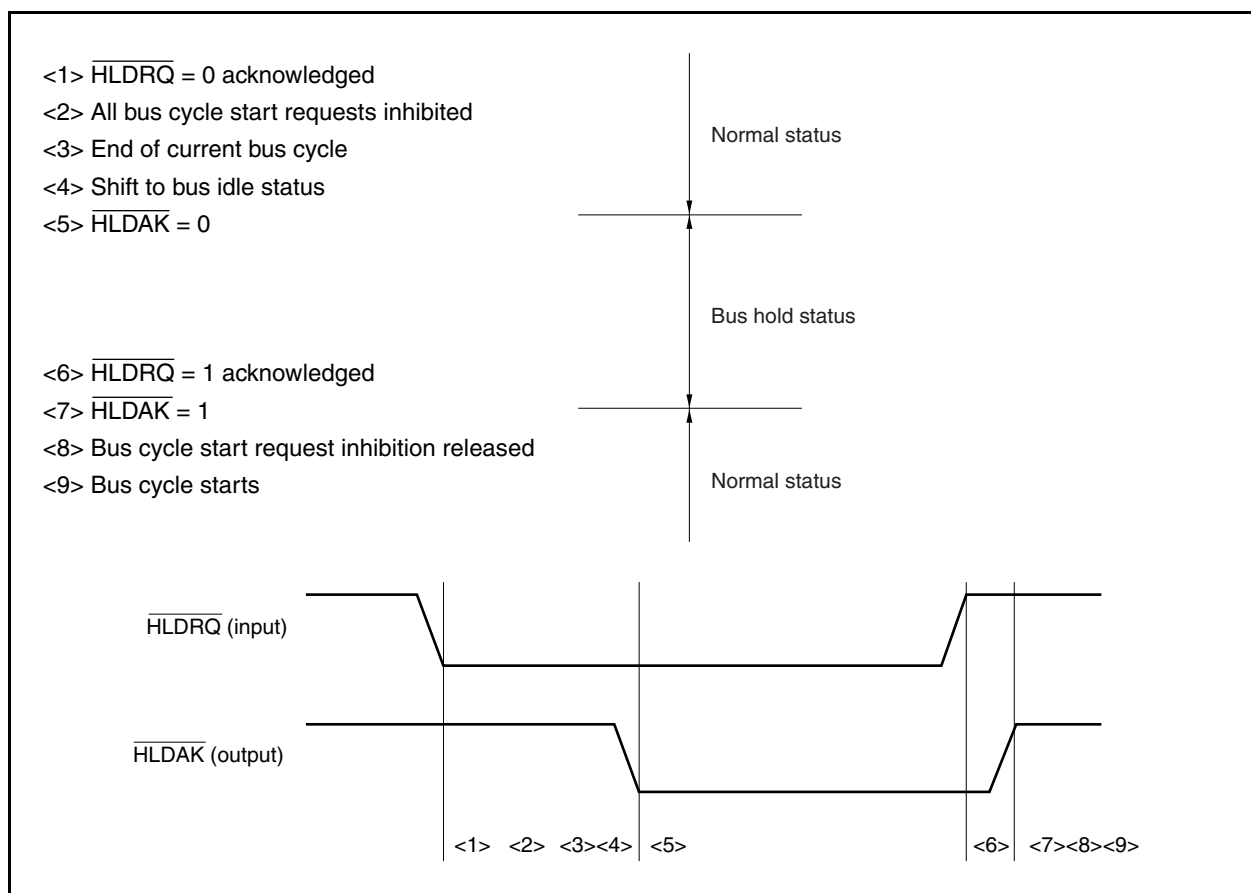
### 5.7.3 Operation in power save mode

Because the internal system clock is stopped in the STOP, IDLE1, and IDLE2 modes, the bus hold status is not entered even if the $\overline{\text{HLDRQ}}$ pin is asserted.

In the HALT mode, the $\overline{\text{HLDAK}}$ pin is asserted as soon as the $\overline{\text{HLDRQ}}$ pin has been asserted, and the bus hold status is entered. When the $\overline{\text{HLDRQ}}$ pin is later deasserted, the $\overline{\text{HLDAK}}$ pin is also deasserted, and the bus hold status is cleared.

## 5.8 Bus Priority
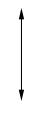
Bus hold, DMA transfer, operand data accesses, instruction fetch (branch), and instruction fetch (successive) are executed in the external bus cycle.

Bus hold has the highest priority, followed by DMA transfer, operand data access, instruction fetch (branch), and instruction fetch (successive).

An instruction fetch may be inserted between the read access and write access in a read-modify-write access.

If an instruction is executed for two or more accesses, an instruction fetch and bus hold are not inserted between accesses due to bus size limitations.

**Table 5-5. Bus Priority**

| Priority | External Bus Cycle | Bus Master |
|---|---|---|
| High | Bus hold | External device |
| ↑ | DMA transfer | DMAC |
| | Operand data access | CPU |
| ↓ | Instruction fetch (branch) | CPU |
| Low | Instruction fetch (successive) | CPU |

## 5.9  Bus Timing

**Figure 5-4.  Multiplexed/Separate Bus Read Timing (Bus Size: 16 Bits, 16-Bit Access)**



| 8-bit Access | Odd Address | Even Address |
|---|---|---|
| AD15 to AD8 | Active | Hi-Z |
| AD7 to AD0 | Hi-Z | Active |

**Notes 1.** V850ES/JH3-U only

**2.** Only the $\overline{CS}$ space subject to access is active.

**Remark**   The broken lines indicate high impedance.
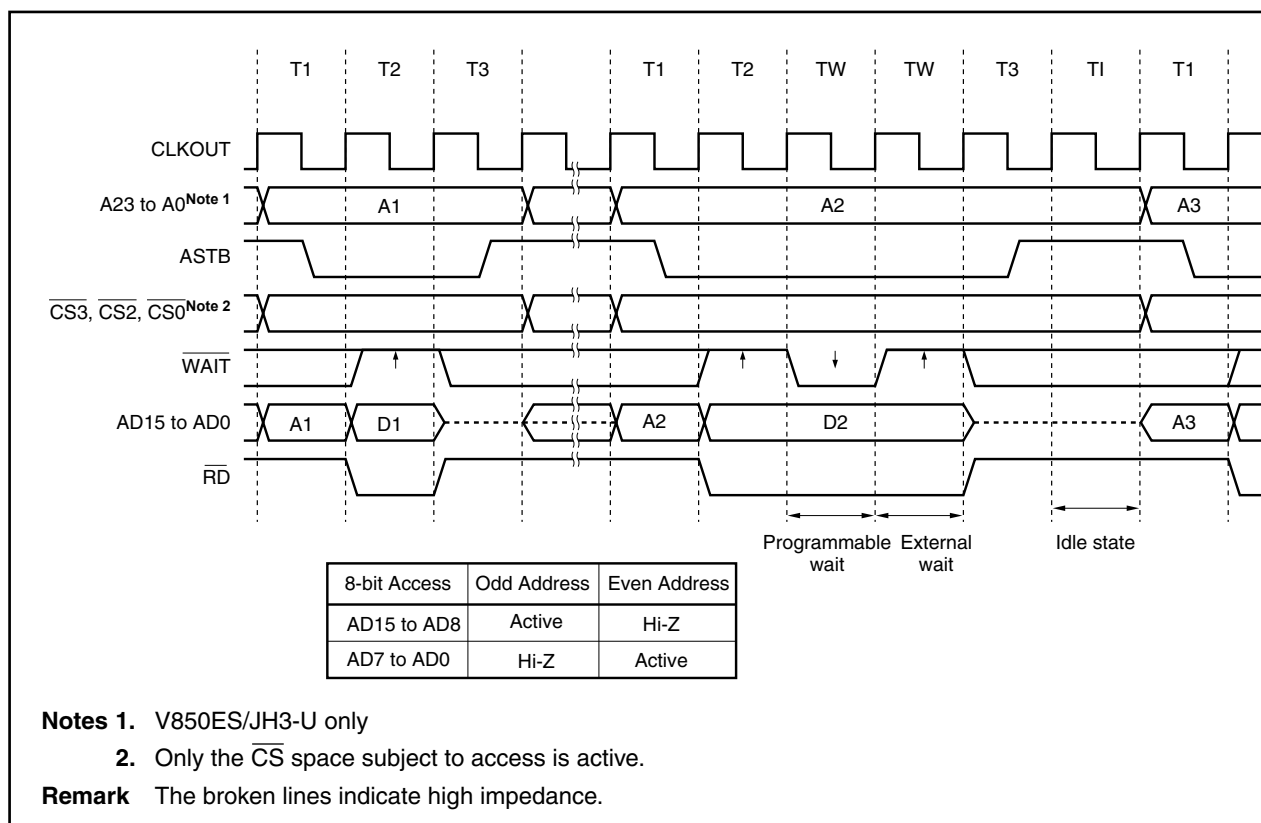
**Figure 5-5.  Multiplexed/Separate Bus Read Timing (Bus Size: 8 Bits)**



**Notes 1.** V850ES/JH3-U only

**2.** Only the $\overline{CS}$ space subject to access is active.

**Remark**   The broken lines indicate high impedance.
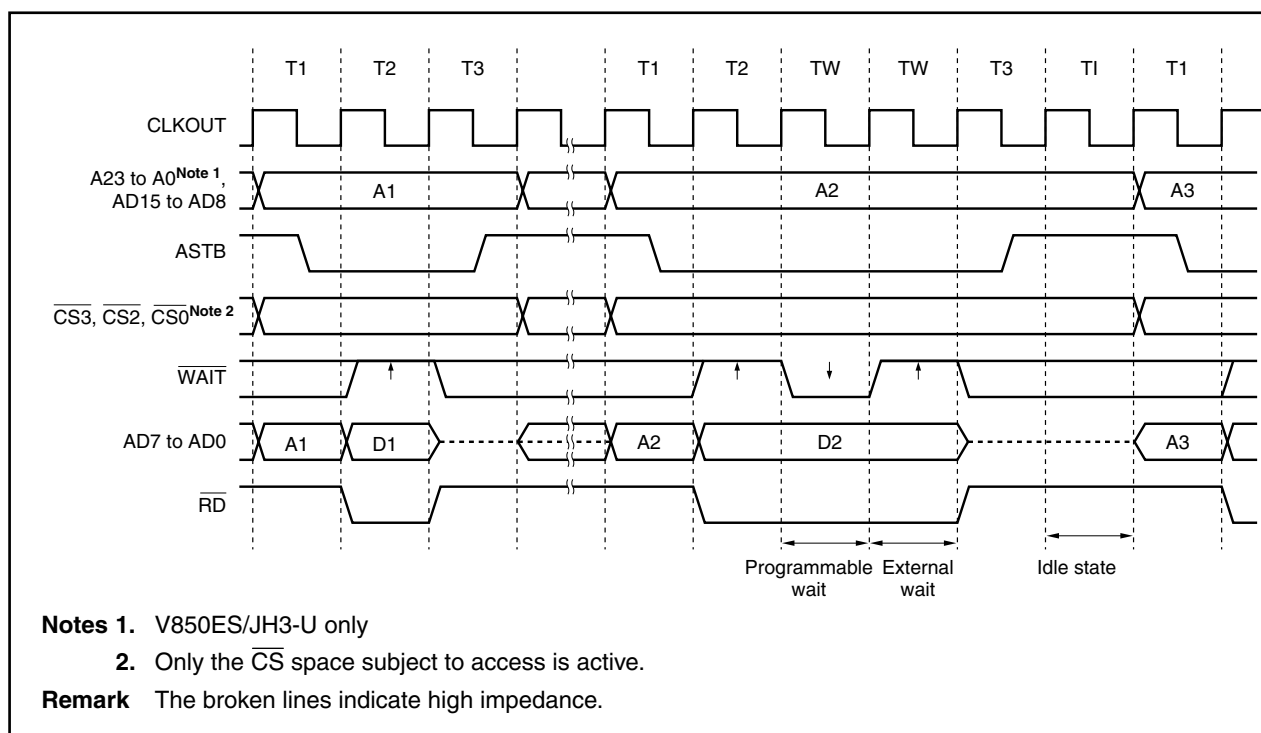
**Figure 5-6.  Multiplexed/Separate Bus Write Timing (Bus Size: 16 Bits, 16-Bit Access)**



| 8-bit Access | Odd Address | Even Address |
|---|---|---|
| AD15 to AD8 | Active | Hi-Z |
| AD7 to AD0 | Undefined | Active |
| $\overline{WR1}$, $\overline{WR0}$ | 01 | 10 |

**Notes 1.**  V850ES/JH3-U only
    **2.**  Only the $\overline{CS}$ space subject to access is active.
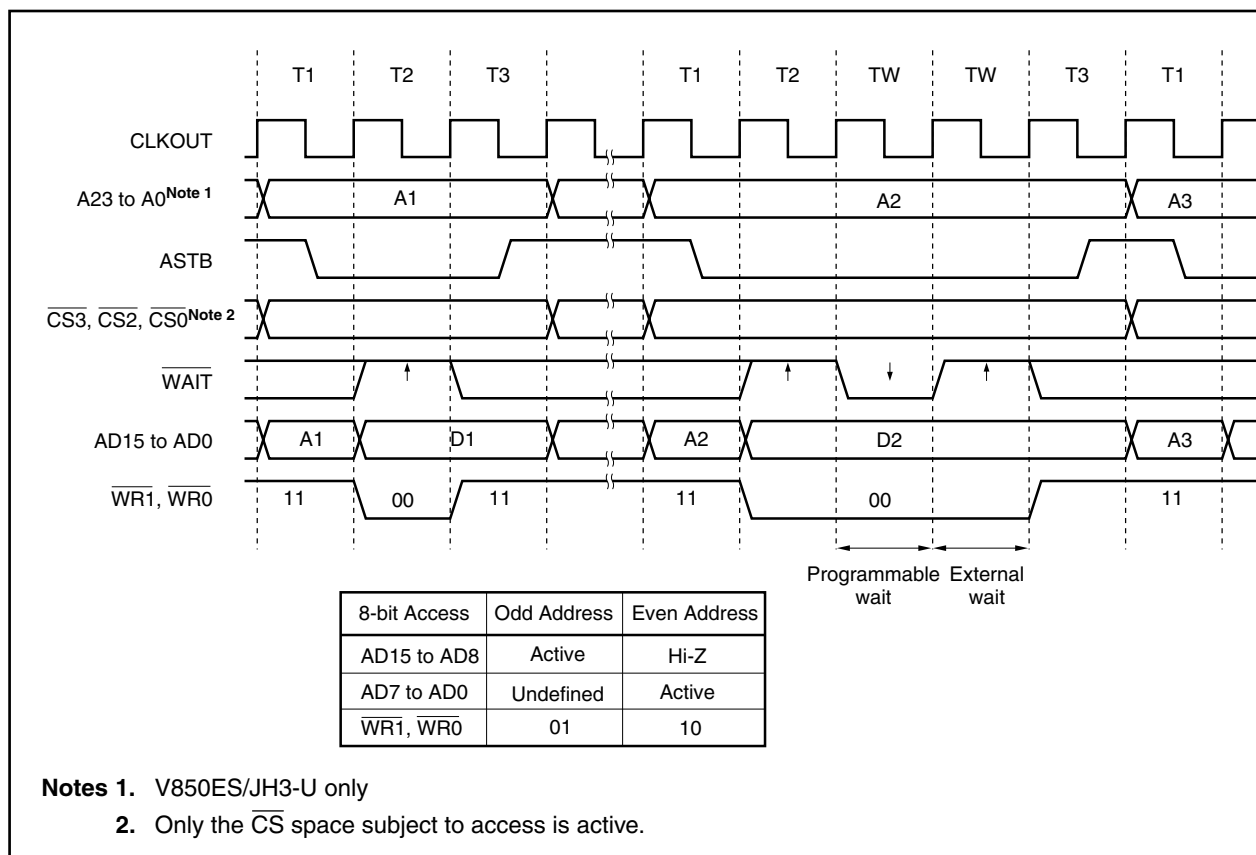
**Figure 5-7.  Multiplexed/Separate Bus Write Timing (Bus Size: 8 Bits)**



**Notes 1.**  V850ES/JH3-U only
    **2.**  Only the $\overline{CS}$ space subject to access is active.
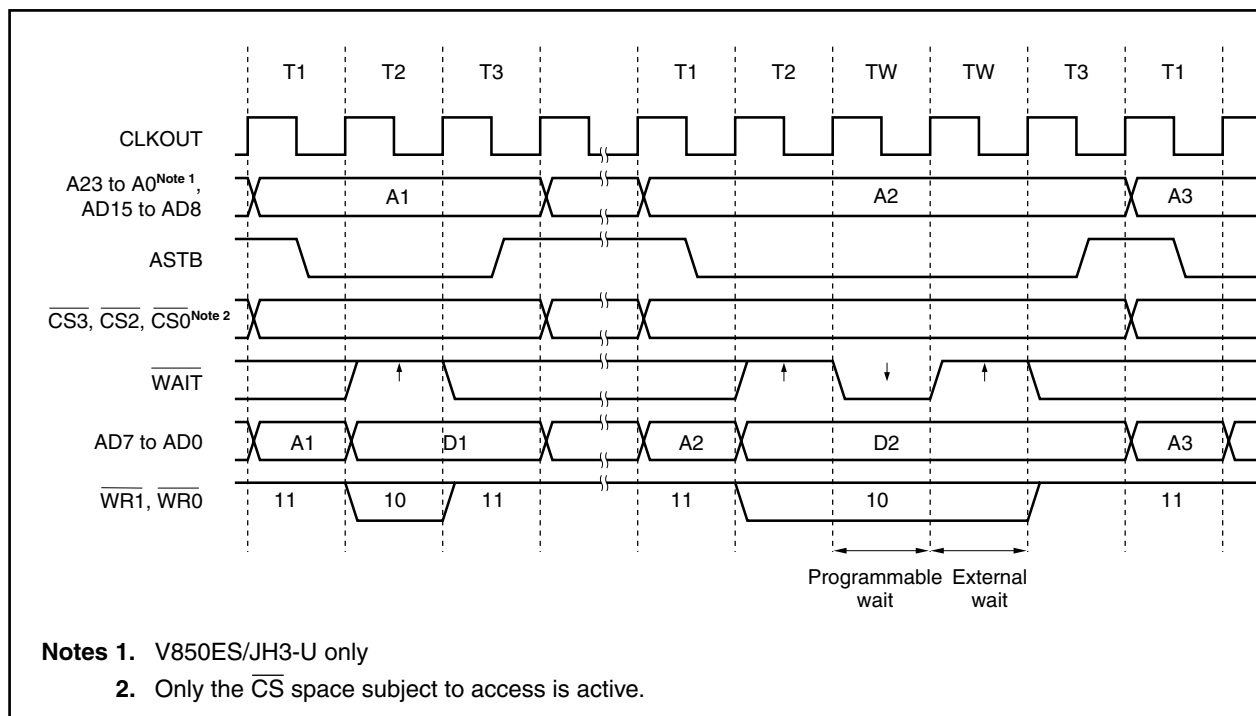
**Figure 5-8.  Multiplexed/Separate Bus Hold Timing (Bus Size: 16 Bits, 16-Bit Access) (V850ES/JH3-U only)**



Notes **1.** This idle state (TI) does not depend on the BCC register settings.

   **2.** Only the $\overline{CS}$ space subject to access is active.

Remarks **1.** See **Table 2-2** for the pin statuses in the bus hold mode.

   **2.** The broken lines indicate high impedance.
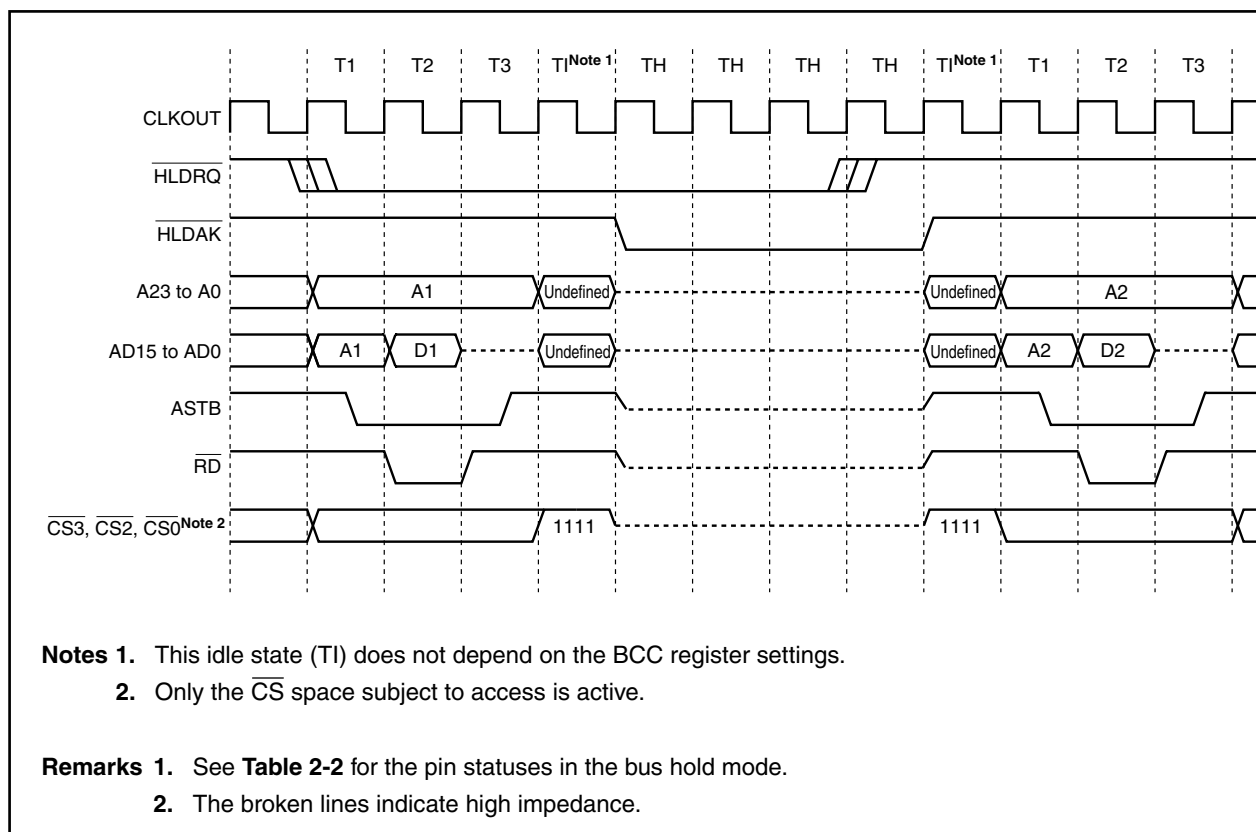
## CHAPTER 6  CLOCK GENERATION FUNCTION

### 6.1  Overview

The following clock generation functions are available.

○ Main clock oscillator
  - In clock-through mode
    $f_X$ = 3.0 to 6.0 MHz ($f_{XX}$ = 3.0 to 6.0 MHz)
  - In PLL mode
    $f_X$ = 3.0 to 6.0 MHz (×8: $f_{XX}$ = 24 to 48 MHz)
○ Subclock oscillator
  - $f_{XT}$ = 32.768 kHz
○ Multiply (×8) function by PLL (Phase Locked Loop)
  - Clock-through mode/PLL mode selectable
○ Internal oscillator
  - $f_R$ = 220 kHz (TYP.)
○ Internal system clock generation
  - 7 steps ($f_{XX}$, $f_{XX}$/2, $f_{XX}$/4, $f_{XX}$/8, $f_{XX}$/16, $f_{XX}$/32, $f_{XT}$)
○ Peripheral clock generation
○ Clock output function

**Remark**  $f_X$:   Main clock oscillation frequency
  $f_{XX}$:  Main clock frequency
  $f_{XT}$:  Subclock frequency
  $f_R$:   Internal oscillation clock frequency

## 6.2  Configuration

**Figure 6-1.  Clock Generator**



**Remark**  fx:    Main clock oscillation frequency

fxx:   Main clock frequency

fCLK:  Internal system clock frequency

fXT:   Subclock frequency

fCPU:  CPU clock frequency

fR:    Internal oscillation clock frequency
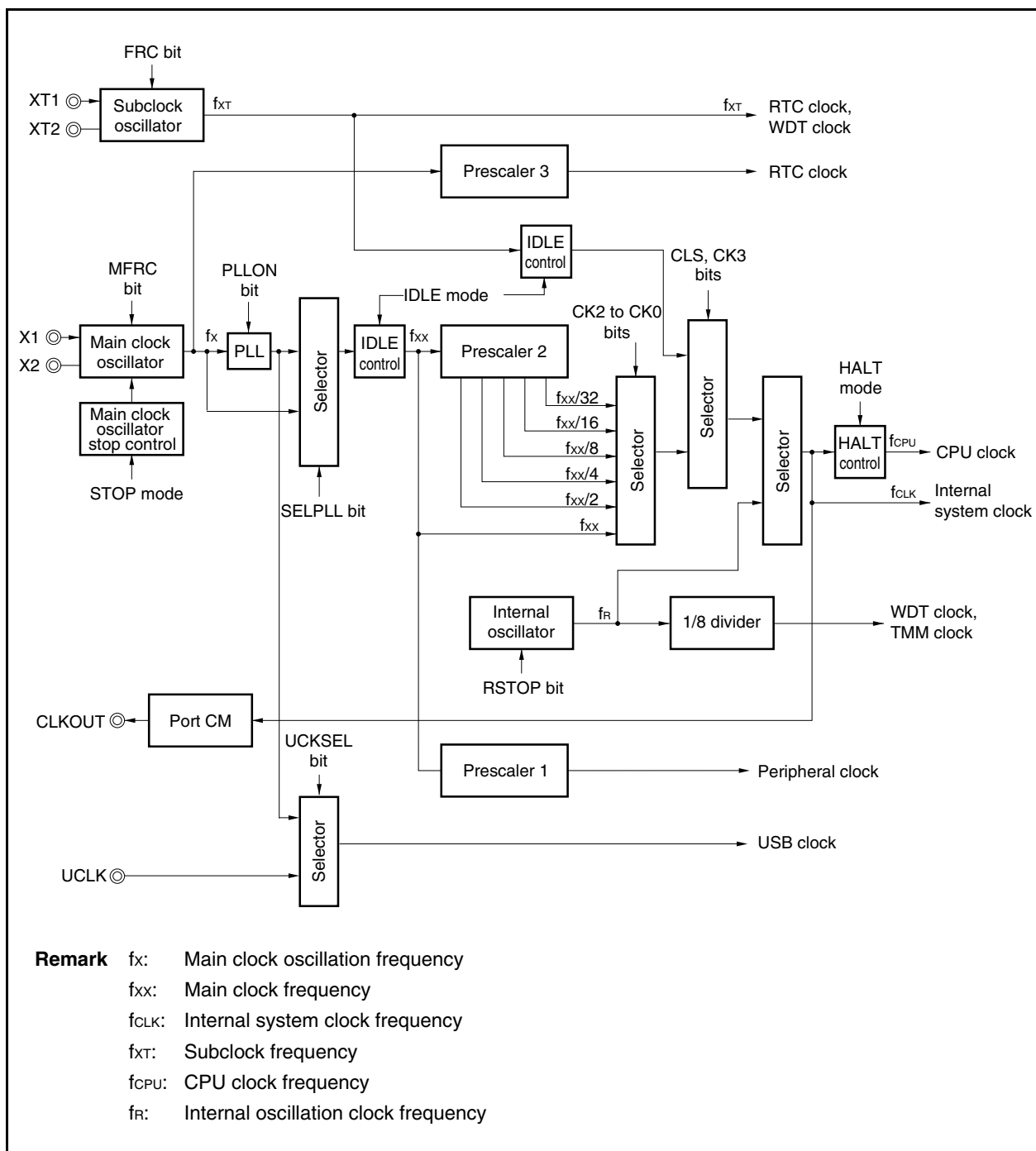
**(1) Main clock oscillator**

The main clock oscillator oscillates the following frequencies ($f_X$).

- In clock-through mode
  $f_X$ = 3.0 to 6.0 MHz
- In PLL mode
  $f_X$ = 3.0 to 6.0 MHz ($\times 8$)

**(2) Subclock oscillator**

The sub-resonator oscillates a frequency of 32.768 kHz ($f_{XT}$).

**(3) Main clock oscillator stop control**

This circuit generates a control signal that stops oscillation of the main clock oscillator.

Oscillation of the main clock oscillator is stopped in the STOP mode or when the PCC.MCK bit = 1 (valid only when the PCC.CLS bit = 1).

**(4) Internal oscillator**

Oscillates a frequency ($f_R$) of 220 kHz (TYP.).

**(5) Prescaler 1**

This prescaler generates the clock ($f_{XX}$ to $f_{XX}$/1,024) to be supplied to the following on-chip peripheral functions: TAA, TAB, TMM, TMT, CSIF, UARTC, I²C, ADC, DAC, WDT2

**(6) Prescaler 2**

This circuit divides the main clock ($f_{XX}$).

The clock generated by prescaler 2 ($f_{XX}$ to $f_{XX}$/32) is supplied to the selector that generates the CPU clock ($f_{CPU}$) and internal system clock ($f_{CLK}$).

$f_{CLK}$ is the clock supplied to the INTC, ROM, and RAM blocks, and can be output from the CLKOUT pin.

**(7) Prescaler 3**

This circuit divides the clock generated by the main clock oscillator ($f_X$) to a specific frequency (32.768 kHz) and supplies that clock to the real-time counter (RTC) block.

**(8) PLL**

This circuit multiplies the clock generated by the main clock oscillator ($f_X$) by 8.

It operates in two modes: clock-through mode in which $f_X$ is output as is, and PLL mode in which a multiplied clock is output.  These modes can be selected by using the PLLCTL.SELPLL bit.

## 6.3  Registers

**(1)  Processor clock control register (PCC)**

The PCC register is a special register.  Data can be written to this register only in combination of specific sequences (see **3.4.7  Special registers**).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 03H.

After reset:  03H    R/W    Address:  FFFFF828H

| | 7 | <6> | 5 | <4> | <3> | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PCC | FRC | MCK | MFRC | CLS$^{Note}$ | CK3 | CK2 | CK1 | CK0 |

| FRC | Use of subclock on-chip feedback resistor |
|---|---|
| 0 | Used |
| 1 | Not used |

| MCK | Main clock oscillator control |
|---|---|
| 0 | Oscillation enabled |
| 1 | Oscillation stopped |

- Even if the MCK bit is set (1) while the system is operating with the main clock as the CPU clock, the operation of the main clock does not stop.  It stops after the CPU clock has been changed to the subclock.
- Before setting the MCK bit from 0 to 1, stop the on-chip peripheral functions operating with the main clock.
- When the main clock is stopped and the device is operating with the subclock, clear (0) the MCK bit and secure the oscillation stabilization time by software before switching the CPU clock to the main clock or operating the on-chip peripheral functions.

| MFRC | Use of main clock on-chip feedback resistor |
|---|---|
| 0 | Used |
| 1 | Not used |

| CLS$^{Note}$ | Status of CPU clock ($f_{CPU}$) |
|---|---|
| 0 | Main clock operation |
| 1 | Subclock operation |

| CK3 | CK2 | CK1 | CK0 | Clock selection ($f_{CLK}/f_{CPU}$) |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | $f_{XX}$ |
| 0 | 0 | 0 | 1 | $f_{XX}/2$ |
| 0 | 0 | 1 | 0 | $f_{XX}/4$ |
| 0 | 0 | 1 | 1 | $f_{XX}/8$ |
| 0 | 1 | 0 | 0 | $f_{XX}/16$ |
| 0 | 1 | 0 | 1 | $f_{XX}/32$ |
| 0 | 1 | 1 | × | Setting prohibited |
| 1 | × | × | × | $f_{XT}$ |

**Note**  The CLS bit is a read-only bit.

**Cautions 1.  Do not change the CPU clock (by using the CK3 to CK0 bits) while CLKOUT is being output.**

**2.  Use a bit manipulation instruction to manipulate the CK3 bit.  When using an 8-bit manipulation instruction, do not change the set values of the CK2 to CK0 bits.**

**Remark**  ×: don't care

**(a) Example of setting main clock operation → subclock operation**

&lt;1&gt; CK3 bit ← 1:     Use of a bit manipulation instruction is recommended. Do not change the CK2 to CK0 bits.

&lt;2&gt; Subclock operation: Read the CLS bit to check if subclock operation has started. It takes the following time after the CK3 bit is set until subclock operation is started.

       Max.: $1/f_{XT}$ (1/subclock frequency)

&lt;3&gt; MCK bit ← 1:     Set the MCK bit to 1 only when stopping the main clock.

**Cautions 1.  When stopping the main clock, stop the PLL. Also stop the operations of the on-chip peripheral functions operating with the main clock.**

**2.     If the following conditions are not satisfied, change the CK2 to CK0 bits so that the conditions are satisfied, then change to the subclock operation mode.**

**Internal system clock ($f_{CLK}$) > Subclock ($f_{XT}$: 32.768 kHz) × 4**

**Remark**   Internal system clock ($f_{CLK}$): Clock generated from the main clock ($f_{XX}$) by setting the CK2 to CK0 bits

[Description example]

```
      _DMA_DISABLE:
      clrl      0, DCHCn[r0]          -- DMA operation disabled.  n = 0 to 3
<1>   _SET_SUB_RUN :
      st.b      r0, PRCMD[r0]
      set1      3, PCC[r0]            -- CK3 bit ← 1
<2>   _CHECK_CLS :
      tst1      4, PCC[r0]            -- Wait until subclock operation starts.
      bz        _CHECK_CLS
<3>   _STOP_MAIN_CLOCK :
      st.b      r0, PRCMD[r0]
      set1      6, PCC[r0]            -- MCK bit ← 1, main clock is stopped.
      _DMA_ENABLE:
      setl      0, DCHCn[r0]          -- DMA operation enabled.  n = 0 to 3
```

**Remark**   The description above is simply an example. Note that in &lt;2&gt; above, the CLS bit is read in a closed loop.

**(b) Example of setting subclock operation → main clock operation**

<1> MCK bit ← 0:        Main clock starts oscillating

<2> Insert waits by the program and wait until the oscillation stabilization time of the main clock elapses.

<3> CK3 bit ← 0:        Use of a bit manipulation instruction is recommended.  Do not change the CK2
to CK0 bits.

<4> Main clock operation: It takes the following time after the CK3 bit is set until main clock operation is
started.

Max.: $1/f_{XT}$ (1/subclock frequency)

Therefore, insert one NOP instruction immediately after setting the CK3 bit to 0
or read the CLS bit to check if main clock operation has started.

**Caution  Enable operation of the on-chip peripheral functions operating with the main clock only after
the oscillation of the main clock stabilizes.  If their operations are enabled before the lapse of
the oscillation stabilization time, a malfunction may occur.**

[Description example]

```
    _DMA_DISABLE:
    clrl      0, DCHCn[r0]              -- DMA operation disabled.  n = 0 to 3
<1> _START_MAIN_OSC :
    st.b      r0, PRCMD[r0]            -- Release of protection of special registers
    clr1      6, PCC[r0]               -- Main clock starts oscillating.
<2> movea     0x55, r0, r11            -- Wait for oscillation stabilization time.
    _WAIT_OST :
    nop
    nop
    nop
    addi      -1, r11, r11
    cmp       r0, r11
    bne               _WAIT_OST
<3> st.b      r0, PRCMD[r0]
    clr1      3, PCC[r0]               -- CK3 ← 0
<4> _CHECK_CLS :
    tst1      4, PCC[r0]               -- Wait until main clock operation starts.
    bnz       _CHECK_CLS
    _DMA_ENABLE:
    setl      0, DCHCn[r0]             -- DMA operation enabled.  n = 0 to 3
```

**Remark**  The description above is simply an example.  Note that in <4> above, the CLS bit is read in a closed
loop.

**(2) Internal oscillation mode register (RCM)**

The RCM register is an 8-bit register that sets the operation mode of the internal oscillator.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset:  00H　　R/W　　Address:  FFFFF80CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| RCM | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RSTOP |

| RSTOP | Oscillation/stop of internal oscillator |
|---|---|
| 0 | Internal oscillator oscillation |
| 1 | Internal oscillator stopped |

**Cautions 1. The internal oscillator cannot be stopped while the CPU is operating on the internal oscillation clock (CCLS.CCLSF bit = 1).  Do not set the RSTOP bit to 1.**

**2. The internal oscillator oscillates if the CCLS.CCLSF bit is set to 1 (when WDT overflow occurs during oscillation stabilization) even when the RSTOP bit is set to 1.  At this time, the RSTOP bit remains being set to 1.**

**(3) CPU operation clock status register (CCLS)**

The CCLS register indicates the status of the CPU operation clock.

This register is read-only, in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset:  00H[Note]　　R　　Address:  FFFFF82EH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CCLS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | CCLSF |

| CCLSF | CPU operation clock status |
|---|---|
| 0 | Operating on main clock ($f_X$) or subclock ($f_{XT}$). |
| 1 | Operating on internal oscillation clock ($f_R$). |

**Note**  If WDT overflow occurs during oscillation stabilization after a reset is released, the CCLSF bit is set to 1 and the reset value is 01H.

## 6.4  Operation

### 6.4.1  Operation of each clock

The following table shows the operation status of each clock.

**Table 6-1.  Operation Status of Each Clock**

| Register Setting and Operation Status / Target Clock | PCC Register | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | CLK Bit = 0, MCK Bit = 0 | | | | | CLS Bit = 1, MCK Bit = 0 | | CLS Bit = 1, MCK Bit = 1 | |
| | During Reset | During Oscillation Stabilization Time Count | HALT Mode | IDLE1, IDLE2 Mode | STOP Mode | Subclock Mode | Sub-IDLE Mode | Subclock Mode | Sub-IDLE Mode |
| Main clock oscillator ($f_X$) | × | ○ | ○ | ○ | × | ○ | ○ | × | × |
| Subclock oscillator ($f_{XT}$) | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| CPU clock ($f_{CPU}$) | × | × | × | × | × | ○ | × | ○ | × |
| Internal system clock ($f_{CLK}$) | × | × | ○ | × | × | ○ | × | ○ | × |
| Main clock (in PLL mode, $f_{XX}$) | × | ○**Note** | ○ | × | × | ○ | ○ | × | × |
| Peripheral clock ($f_{XX}$ to $f_{XX}/1{,}024$) | × | × | ○ | × | × | ○ | × | × | × |
| WT clock (main) | × | ○ | ○ | ○ | × | ○ | ○ | × | × |
| WT clock (sub) | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| WDT2 clock (internal oscillation) | × | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| WDT2 clock (main) | × | × | ○ | × | × | ○ | × | × | × |
| WDT2 clock (sub) | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |

**Note**  Lockup time

**Remark**   ○: Operable
        ×:  Stopped

### 6.4.2  Clock output function

The clock output function is used to output the internal system clock ($f_{CLK}$) from the CLKOUT pin.

The internal system clock ($f_{CLK}$) is selected by using the PCC.CK3 to PCC.CK0 bits.

The CLKOUT pin functions alternately as the PCM1 pin and functions as a clock output pin if so specified by the control register of port CM.

The status of the CLKOUT pin is the same as the internal system clock in Table 6-1 and the pin can output the clock when it is in the operable status.  It outputs a low level in the stopped status.  However, the CLKOUT pin is in the port mode (PCM1 pin: input mode) after reset and until it is set in the output mode.  Therefore, the status of the pin is Hi-Z.

## 6.5  PLL Function

### 6.5.1  Overview

In the V850ES/JG3-U and V850ES/JH3-U, an operating clock that is 8 times higher than the oscillation frequency output by the PLL function or the clock-through mode can be selected as the operating clock of the CPU and on-chip peripheral functions.

When PLL function is used (×8):  Input clock = 3.0 to 6.0 MHz (output: 24 to 48 MHz)
Clock-through mode:                Input clock = 3.0 to 6.0 MHz (output: 3.0 to 6.0 MHz)

### 6.5.2  Registers

**(1)  PLL control register (PLLCTL)**

The PLLCTL register is an 8-bit register that controls the PLL function.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 01H.

After reset: 01H    R/W    Address:  FFFFF82CH

|  | 7 | 6 | 5 | 4 | 3 | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PLLCTL | 0 | 0 | 0 | 0 | 0 | 0 | SELPLL | PLLON |

| PLLON | PLL operation stop register |
|---|---|
| 0 | PLL stopped |
| 1 | PLL operating<br>(After PLL operation starts, a lockup time is required for frequency stabilization) |

| SELPLL | CPU operation clock selection register |
|---|---|
| 0 | Clock-through mode |
| 1 | PLL mode |

**Cautions 1.  When the PLLON bit is cleared to 0, the SELPLL bit is automatically cleared to 0 (clock-through mode).**

**2.  The SELPLL bit can be set to 1 only when the PLL clock frequency is stabilized.  If not (unlocked), "0" is written to the SELPLL bit if data is written to it.**

**(2) Clock control register (CKC)**

The CKC register is a special register.  Data can be written to this register only in a combination of specific sequence (see **3.4.7  Special registers**).

The CKC register controls the internal system clock in the PLL mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 0AH.

After reset:  0AH        R/W        Address:  FFFFF822H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CKC | 0 | 0 | 0 | 0 | 1 | 0 | 1 | CKDIV0 |

| CKDIV0 | Internal system clock ($f_{XX}$) in PLL mode |
|---|---|
| 0 | Setting prohibited |
| 1 | $f_{XX} = 8 \times f_X$ ($f_X$ = 3.0 to 6.0 MHz) |

**Cautions 1.  Be sure to set the CKC register to 0BH.  When setting this register to a value other than 0BH or leaving it set to its initial value without setting it to 0BH, enabling PLL operation (PLLCTL.SELPLL = 1) is prohibited.**

**2.  Be sure to set bits 3 and 1 to "1" and set bits 7 to 4 and 2 to "0".**

**Remark**   Both the CPU clock and peripheral clock are divided by the CKC register, but only the CPU clock is divided by the PCC register.

**(3) Lock register (LOCKR)**

Phase lock occurs at a given frequency following power application or immediately after the STOP mode is released, and the time required for stabilization is the lockup time (frequency stabilization time).  This state until stabilization is called the lockup status, and the stabilized state is called the locked status.

The LOCKR register includes a LOCK bit that reflects the PLL frequency stabilization status.

This register is read-only, in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset:  00H    R    Address:  FFFFF824H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| LOCKR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | LOCK |

| LOCK | PLL lock status check |
|---|---|
| 0 | Locked status |
| 1 | Unlocked status |

**Caution   The LOCK register does not reflect the lock status of the PLL in real time.  The set/clear conditions are as follows.**

**[Set conditions]**
- Upon system reset[Note]
- In IDLE2 or STOP mode
- Upon setting of PLL stop (clearing of PLLCTL.PLLON bit to 0)
- Upon stopping main clock and using CPU with subclock (setting of PCC.CK3 bit to 1 and setting of PCC.MCK bit to 1)

  **Note**  This register is set to 01H by reset and cleared to 00H after the reset has been released and the oscillation stabilization time has elapsed.

**[Clear conditions]**
- Upon overflow of oscillation stabilization time following reset release (OSTS register default time (see **25.2 (3)  Oscillation stabilization time select register (OSTS)**))
- Upon oscillation stabilization timer overflow (time set by OSTS register) following STOP mode release, when the STOP mode was set in the PLL operating status
- Upon PLL lockup time timer overflow (time set by PLLS register) when the PLLCTL.PLLON bit is changed from 0 to 1
- After the setup time inserted upon release of the IDLE2 mode is released (time set by the OSTS register) when the IDLE2 mode is set during PLL operation.

**(4) PLL lockup time specification register (PLLS)**

The PLLS register is an 8-bit register used to select the PLL lockup time when the PLLCTL.PLLON bit is changed from 0 to 1.

This register can be read or written in 8-bit units.

Reset sets this register to 03H.

After reset:  03H      R/W      Address:  FFFFF6C1H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PLLS | 0 | 0 | 0 | 0 | 0 | 0 | PLLS1 | PLLS0 |

| PLLS1 | PLLS0 | Selection of PLL lockup time |
|---|---|---|
| 0 | 0 | $2^{10}/f_X$ |
| 0 | 1 | $2^{11}f_X$ |
| 1 | 0 | $2^{12}/f_X$ |
| 1 | 1 | $2^{13}/f_X$ (default value) |

**Cautions  1.  Set so that the lockup time is 800 $\mu$s or longer.**

**2.  Do not change the PLLS register setting during the lockup period.**

## 6.5.3  Usage

**(1) When PLL is used**

- After the reset signal has been released, the PLL operates (PLLCTL.PLLON bit = 1), but because the default mode is the clock-through mode (PLLCTL.SELPLL bit = 0), select the PLL mode (SELPLL bit = 1).
- To enable PLL operation, first set the PLLON bit to 1, and then set the SELPLL bit to 1 after the LOCKR.LOCK bit = 0.  To stop the PLL, first select the clock-through mode (SELPLL bit = 0), wait for 8 clocks or more, and then stop the PLL (PLLON bit = 0).
- The PLL stops during transition to the IDLE2 or STOP mode regardless of the setting and is restored from the IDLE2 or STOP mode to the status before transition.  The time required for restoration is as follows.

(a) When transiting to the IDLE2 or STOP mode from the clock through mode

- STOP mode:  Set the OSTS register so that the oscillation stabilization time is 1 ms (min.) or longer.
- IDLE2 mode:  Set the OSTS register so that the setup time is 350 $\mu$s (min.) or longer.

(b) When transiting to the IDLE 2 or STOP mode while remaining in the PLL operation mode

- STOP mode:  Set the OSTS register so that the oscillation stabilization time is 1 ms (min.) or longer.
- IDLE2 mode:  Set the OSTS register so that the setup time is 800 $\mu$s (min.) or longer.

When transiting to the IDLE1 mode, the PLL does not stop.  Stop the PLL if necessary.

**(2) When PLL is not used**

- The clock-through mode (SELPLL bit = 0) is selected after the reset signal has been released, but the PLL is operating (PLLON bit = 1) and must therefore be stopped (PLLON bit = 0).

# CHAPTER 7  16-BIT TIMER/EVENT COUNTER AA (TAA)

Timer AA (TAA) is 16-bit timer/event counter.
The V850ES/JG3-U and V850ES/JH3-U have TAA0 to TAA5.

## 7.1  Overview

An overview of TAAn is shown below.

- Clock selection:         8 ways
- Capture/trigger input pins:     2
- External event count input pins[Note]:  1
- External trigger input pin[Note]:   1
- Timer/counter:         1
- Capture/compare registers:    2
  (32-bit capture timer function available by using a cascade connection of TAA0 and TAA1, TAA2 and TAA3.)
- Capture/compare match interrupt request signals: 2
- Timer output pins:        2

**Note** External event count input pins and external trigger input pins are alternately used as capture/trigger input pins (TIAAm0).

**Remark** n = 0 to 5, m = 0 to 3, 5

## 7.2  Functions

TAAn has the following functions.

- Interval timer
- External event counter
- External trigger pulse output
- One-shot pulse output
- PWM output
- Free-running timer
- Pulse width measurement
- Timer-tuned function
- Simultaneous-start function

## 7.3 Configuration

TAAn includes the following hardware.

**Table 7-1. Configuration of TAAn**

| Item | Configuration |
|---|---|
| Registers | 16-bit counter<br>TAAn capture/compare registers 0, 1 (TAAnCCR0, TAAnCCR1)<br>TAAn counter read buffer register (TAAnCNT)<br>CCR0, CCR1 buffer registers<br>TAAn control registers 0, 1 (TAAnCTL0, TAAnCTL1)<br>TAAm I/O control registers 0 to 2, 4 (TAAmIOC0 to TAAmIOC2, TAAmIOC4)<br>TAAm option registers 0, 1 (TAAmOPT0, TAAmOPT1)<br>TAA noise elimination control register (TANFC) |
| Timer inputs[Note 1] | 2 (TIAAm0[Note 2], TIAAm1 pins) |
| Timer outputs[Note 1] | 2 (TOAAm0, TOAAm1 pins) |

**Notes 1.** When using the functions of the TIAAm0, TIAAm1, TOAAm0, and TOAAm1 pins, see **Table 4-20 Using Port Pin as Alternate-Function Pin**.

    **2.** The TIAAm0 pin functions alternately as a capture trigger input signal, external event count input signal, and external trigger input signal.

**Remark**   n = 0 to 5, m = 0 to 3, 5

**Figure 7-1. Block Diagram of TAAn**
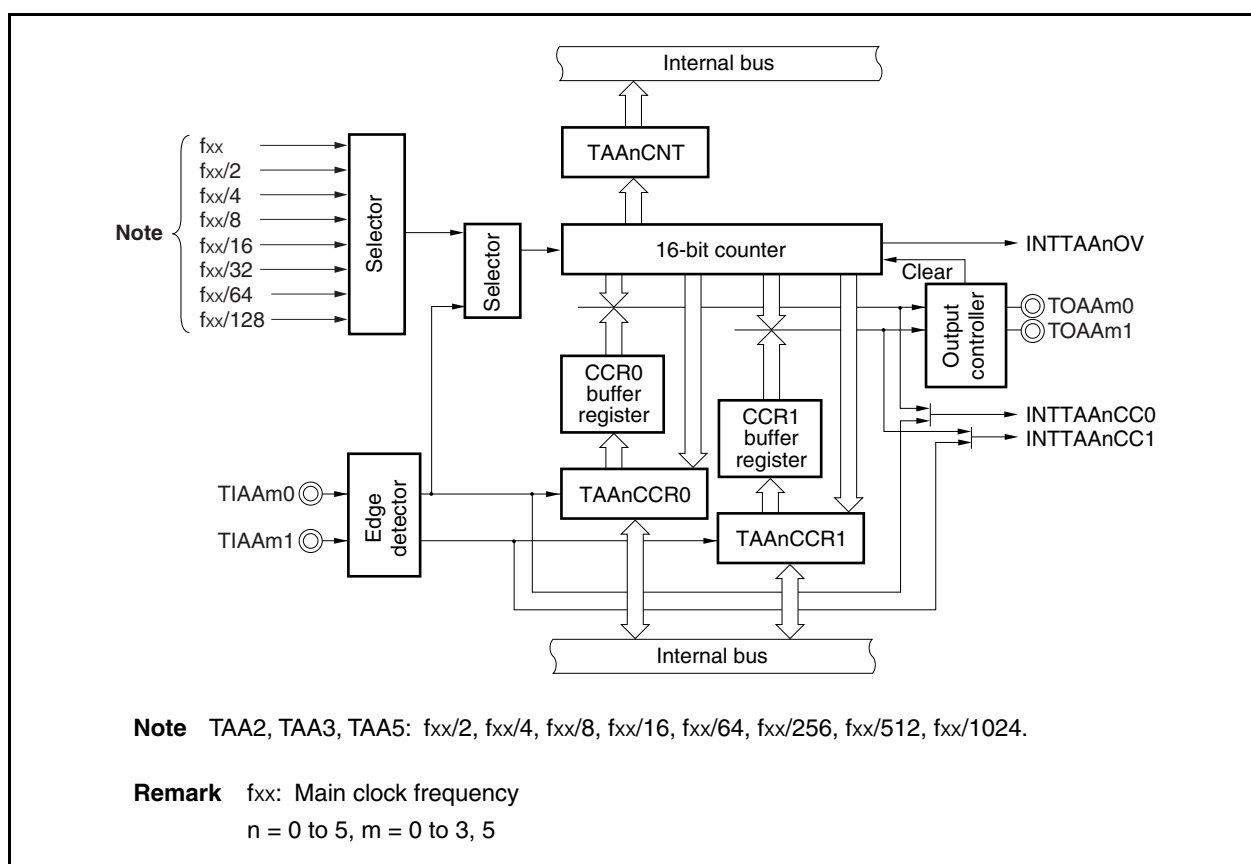


**Note**   TAA2, TAA3, TAA5: $f_{xx}/2$, $f_{xx}/4$, $f_{xx}/8$, $f_{xx}/16$, $f_{xx}/64$, $f_{xx}/256$, $f_{xx}/512$, $f_{xx}/1024$.

**Remark**   $f_{xx}$: Main clock frequency

          n = 0 to 5, m = 0 to 3, 5

**(1)  16-bit counter**

This 16-bit counter can count internal clocks or external events.

The count value of this counter can be read by using the TAAnCNT register.

When the TAAnCTL0.TAAnCE bit = 0, the value of the 16-bit counter is FFFFH.  If the TAAnCNT register is read at this time, 0000H is read.

Reset sets the TAAnCE bit to 0.  Therefore, the 16-bit counter is set to FFFFH.

**(2)  CCR0 buffer register**

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TAAnCCR0 register is used as a compare register, the value written to the TAAnCCR0 register is transferred to the CCR0 buffer register.  When the count value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTAAnCC0) is generated.

The CCR0 buffer register cannot be read or written directly.

Reset clears the TAAnCCR0 register to 0000H.  Therefore, the CCR0 buffer register is cleared to 0000H.

**(3)  CCR1 buffer register**

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TAAnCCR1 register is used as a compare register, the value written to the TAAnCCR1 register is transferred to the CCR1 buffer register.  When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTAAnCC1) is generated.

The CCR1 buffer register cannot be read or written directly.

Reset clears the TAAnCCR1 register to 0000H.  Therefore, the CCR1 buffer register is cleared to 0000H.

**(4)  Edge detector**

This circuit detects the valid edges input to the TIAAm0 and TIAAm1 pins.  No edge, rising edge, falling edge, or both the rising and falling edges can be selected as the valid edge by using the TAAmIOC1 and TAAmIOC2 registers.

**(5)  Output controller**

This circuit controls the output of the TOAAm0 and TOAAm1 pins.  The outputs of the TOAAm0 and TOAAm1 pins are controlled by the TAAmIOC0 register.

**(6)  Selector**

This selector selects the count clock for the 16-bit counter.  Eight types of internal clocks or an external event can be selected as the count clock.

### 7.3.1   Pin configuration

The timer inputs and outputs that configure TAAn are shared with the following ports.  The port functions must be set when using each pin (see **Table 4-20 When Using Port Pins as Alternate-Function Pins**).

**Table 7-2  Pin Configuration**

| Channel | Port | Timer AA Input | Timer AA Output | Other Alternate Function |
|---------|------|----------------|-----------------|--------------------------|
| TAA0 | P32 | TIAA00[Note 1] | TOAA00 | ASCK0/$\overline{\text{SCKF4}}$ |
|      | P33 | TIAA01 | TOAA01 | RTCDIV/RTCCL |
| TAA1 | P34 | TIAA10[Note 1] | TOAA10 | TOAA1OFF/INTP09/PPON |
|      | P35 | TIAA11 | TOAA11 | RTC1HZ/$\overline{\text{OCI}}$ |
| TAA2 | P97 | TIAA20[Note 1] | TOAA20 | SIF1/A7[Note 2] |
|      | P96 | TIAA21 | TOAA21 | INTP11/A6[Note 2] |
| TAA3 | P95 | TIAA30[Note 1] | TOAA30 | A5[Note 2] |
|      | P94 | TIAA31 | TOAA31 | TENC00/EVTT0/A4[Note 2] |
| TAA4 | – | – | – | – |
|      | – | – | – | – |
| TAA5 | P915 | TIAA50[Note 1] | TOAA50 | INTP18/A15[Note 2] |
|      | P914 | TIAA51 | TOAA51 | INTP17/A14[Note 2] |

**Notes 1.** The TAAm0 pin functions alternately as a capture trigger input function, external event input function, and external trigger input function.

    **2.** V850ES/JH3-U only

**Remark**   TAA4 has neither timer inputs nor outputs.  Consequently, only the interval timer function can use TAA4 by itself.  However, the 6-phase PWM output function can be achieved by using TAA4 together with TAB1.

## 7.4   Registers

The registers that control TAAn are as follows.

- TAAn control register 0 (TAAnCTL0)
- TAAn control register 1 (TAAnCTL1)
- TAAn I/O control register 0 (TAAmIOC0)
- TAAn I/O control register 1 (TAAmIOC1)
- TAAn I/O control register 2 (TAAmIOC2)
- TAAn I/O control register 4 (TAAmIOC4)
- TAAn option register 0 (TAAmOPT0)
- TAAn option register 1 (TAAmOPT1)
- TAAn capture/compare register 0 (TAAnCCR0)
- TAAn capture/compare register 1 (TAAnCCR1)
- TAAn counter read buffer register (TAAnCNT)
- TAA noise elimination control register (TANFC)

**Remarks 1.** When using the functions of the TIAAm0, TIAAm1, TOAAm0, and TOAAm1 pins, see **Table 4-20   Using Port Pin as Alternate-Function Pin**.

**2.** n = 0 to 5, m = 0 to 3, 5

**(1)  TAAn control register 0 (TAAnCTL0)**

The TAAnCTL0 register is an 8-bit register that controls the operation of TAAn.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

The same value can always be written to the TAAnCTL0 register by software.

After reset: 00H    R/W    Address:    TAA0CTL0 FFFFF630H, TAA1CTL0 FFFFF640H,
TAA2CTL0 FFFFF650H, TAA3CTL0 FFFFF660H,
TAA4CTL0 FFFFF670H, TAA5CTL0 FFFFF680H

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TAAnCTL0 | TAAnCE | 0 | 0 | 0 | 0 | TAAnCKS2 | TAAnCKS1 | TAAnCKS0 |

(n = 0 to 5)

| TAAnCE | TAAn operation control |
|---|---|
| 0 | TAAn operation disabled (TAAn reset asynchronously[Note] ). |
| 1 | TAAn operation enabled.  TAAn operation started. |

| TAAnCKS2 | TAAnCKS1 | TAAnCKS0 | Internal count clock selection | | | |
|---|---|---|---|---|---|---|
| | | | n = 0, 1, 4 | | n = 2, 3, 5 | |
| 0 | 0 | 0 | $f_{XX}$ | (20.8 ns) | $f_{XX}/2$ | (41.7 ns) |
| 0 | 0 | 1 | $f_{XX}/2$ | (41.7 ns) | $f_{XX}/4$ | (83.3 ns) |
| 0 | 1 | 0 | $f_{XX}/4$ | (83.3 ns) | $f_{XX}/8$ | (166.7 ns) |
| 0 | 1 | 1 | $f_{XX}/8$ | (166.7 ns) | $f_{XX}/16$ | (333.3 ns) |
| 1 | 0 | 0 | $f_{XX}/16$ | (333.3 ns) | $f_{XX}/64$ | (1.3333 $\mu$s) |
| 1 | 0 | 1 | $f_{XX}/32$ | (666.7 ns) | $f_{XX}/256$ | (5.3333 $\mu$s) |
| 1 | 1 | 0 | $f_{XX}/64$ | (1.3333 $\mu$s) | $f_{XX}/512$ | (10.6667 $\mu$s) |
| 1 | 1 | 1 | $f_{XX}/128$ | (2.6667 $\mu$s) | $f_{XX}/1024$ | (21.3333 $\mu$s) |

**Note**    TAAnOPT0.TAAnOVF bit, 16-bit counter, timer output (TOAAn0, TOAAn1 pins)

**Cautions  1. Set the TAAnCKS2 to TAAnCKS0 bits when the TAAnCE bit = 0.**
**When  the  value  of  the  TAAnCE  bit  is  changed  from  0  to  1,  the**
**TAAnCKS2 to TAAnCKS0 bits can be set simultaneously.**
**2. Be sure to set bits 3 to 6 to "0".**

**Remark**    $f_{XX}$: Main clock frequency
The values in parentheses indicate the cycles when $f_{XX}$ = 48 MHz.

**(2) TAAn control register 1 (TAAnCTL1)**

The TAAnCTL1 register is an 8-bit register that controls the operation of TAAn.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.
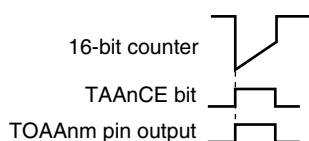
(1/2)

After reset: 00H    R/W    Address:   TAA0CTL1 FFFFF631H, TAA1CTL1 FFFFF641H,
TAA2CTL1 FFFFF651H, TAA3CTL1 FFFFF661H,
TAA4CTL1 FFFFF671H, TAA5CTL1 FFFFF681H

|          | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|---|---|---|---|---|---|---|---|
| TAA0CTL1 | TAA0SYE | TAA0EST | TAA0EEE | TAA0SYM | 0 | TAA0MD2 | TAA0MD1 | TAA0MD0 |
| TAA1CTL1 | 0 | TAA1EST | TAA1EEE | 0 | 0 | TAA1MD2 | TAA1MD1 | TAA1MD0 |
| TAA2CTL1 | TAA2SYE | TAA2EST | TAA2EEE | TAA2SYM | 0 | TAA2MD2 | TAA2MD1 | TAA2MD0 |
| TAA3CTL1 | 0 | TAA3EST | TAA3EEE | 0 | 0 | TAA3MD2 | TAA3MD1 | TAA3MD0 |
| TAA4CTL1 | TAA4SYE | 0 | 0 | TAA4SYM | 0 | TAA4MD2 | TAA4MD1 | TAA4MD0 |
| TAA5CTL1 | TAA5SYE | TAA5EST | TAA5EEE | TAA5SYM | 0 | TAA5MD2 | TAA5MD1 | TAA5MD0 |

| TAAmSYE | TAAmSYM | Tuned operation mode enable control (m = 0, 2, 4, 5) |
|---------|---------|-------------------------------------------------------|
| 0 | 0 | Independent operation mode (asynchronous operation mode) |
| 0 | 1 | Setting prohibited |
| 1 | 0 | Tuned-operation function (specification of slave operation) |
| 1 | 1 | Simultaneous-start function (specification of slave timer) |

These bits can be set only for the slave timer (setting them for the master timer is prohibited).
The relationship between the master timer and slave timer is as follows.

| Master timer | Slave timer |
|--------------|-------------|
| TAA1 | TAA0 |
| TAA3 | TAA2 |
| TAB0 | TAA5 |
| TAB1 | TAA4 |

For the tuned-operation function, see **7.6  Timer-Tuned Operation Function**.
For the simultaneous-start function, see **7.7  Simultaneous-Start Function**.

| TAAnEST | Software trigger control (n = 0 to 3, 5) |
|---------|------------------------------------------|
| 0 | – |
| 1 | Generates a valid signal for external trigger input.<br>• In one-shot pulse output mode:<br>  A one-shot pulse is output with writing 1 to the TAAnEST bit as the trigger.<br>• In external trigger pulse output mode:<br>  A PWM waveform is output with writing 1 to the TAAnEST bit as the trigger. |

**Cautions 1. The TAAnEST bit is valid only in the external trigger pulse output mode or one-shot pulse output mode.  In any other mode, writing 1 to this bit is ignored.**

**2. Be sure to clear the sections of the TAAnCTL1 register of each channel, where 0 is specified, to 0.**

| TAAmEEE | Count clock selection |
|---------|-----------------------|
| 0 | Disables operation with external event count input. (Performs counting with the count clock selected by the TAAmCTL0.TAAmCK0 to TAAmCK2 bits.) |
| 1 | Enables operation with external event count input. (Performs counting at every valid edge of the external event count input signal.) |
| | The TAAmEEE bit selects whether counting is performed with the internal count clock or the valid edge of the external event count input. |

| TAAnMD2 | TAAnMD1 | TAAnMD0 | Timer mode selection |
|---------|---------|---------|----------------------|
| 0 | 0 | 0 | Interval timer mode |
| 0 | 0 | 1 | External event count mode |
| 0 | 1 | 0 | External trigger pulse output mode |
| 0 | 1 | 1 | One-shot pulse output mode |
| 1 | 0 | 0 | PWM output mode |
| 1 | 0 | 1 | Free-running timer mode |
| 1 | 1 | 0 | Pulse width measurement mode |
| 1 | 1 | 1 | Setting prohibited |

**Cautions 1. External event count input is selected in the external event count mode regardless of the value of the TAAmEEE bit.**

**2. Set the TAAmEEE and TAAmMD2 to TAAmMD0 bits when the TAAmCTL0.TAAmCE bit = 0. (The same value can be written when the TAAmCE bit = 1.) The operation is not guaranteed when rewriting is performed with the TAAmCE bit = 1. If rewriting was mistakenly performed, clear the TAAnCE bit to 0 and then set the bits again (m = 0 to 3, 5).**

**(3) TAAn I/O control register 0 (TAAnIOC0)**

The TAAnIOC0 register is an 8-bit register that controls the timer output (TOAAn0, TOAAn1 pins).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address:   TAA0IOC0 FFFFF632H, TAA1IOC0 FFFFF642H,
TAA2IOC0 FFFFF652H, TAA3IOC0 FFFFF662H,
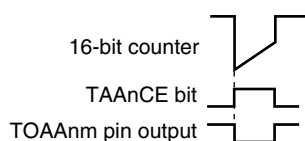TAA5IOC0 FFFFF682H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TAAnIOC0 | 0 | 0 | 0 | 0 | TAAnOL1 | TAAnOE1 | TAAnOL0 | TAAnOE0 |

(n = 0 to 3, 5)

| TAAnOL1 | TOAAn1 pin output level setting[Note] |
|---|---|
| 0 | TOAAn1 pin output starts at high level |
| 1 | TOAAn1 pin output starts at low level |

| TAAnOE1 | TOAAn1 pin output setting |
|---|---|
| 0 | Timer output disabled<br>• When TAAnOL1 bit = 0: Low level is output from the TOAAn1 pin<br>• When TAAnOL1 bit = 1: High level is output from the TOAAn1 pin |
| 1 | Timer output enabled (a square wave is output from the TOAAn1 pin). |

| TAAnOL0 | TOAAn0 pin output level setting[Note] |
|---|---|
| 0 | TOAAn0 pin output starts at high level |
| 1 | TOAAn0 pin output starts at low level |

| TAAnOE0 | TOAAn0 pin output setting |
|---|---|
| 0 | Timer output disabled<br>• When TAAnOL0 bit = 0: Low level is output from the TOAAn0 pin<br>• When TAAnOL0 bit = 1: High level is output from the TOAAn0 pin |
| 1 | Timer output enabled (a square wave is output from the TOAAn0 pin). |

**Note**   The output level of the timer output pin (TOAAnm) specified by the TAAnOLm bit is shown below.

• When TAAnOLm bit = 0                    • When TAAnOLm bit = 1

16-bit counter                                16-bit counter

TAAnCE bit                                    TAAnCE bit

TOAAnm pin output                         TOAAnm pin output

**Cautions  1. Rewrite the TAAnOL1, TAAnOE1, TAAnOL0, and TAAnOE0 bits when the TAAnCTL0.TAAnCE bit = 0. (The same value can be written when the TAAnCE bit = 1.) If rewriting was mistakenly performed, clear the TAAnCE bit to 0 and then set the bits again.**

**2. Even if the TAAnOLm bit is manipulated when the TAAnCE and TAAnOEm bits are 0, the TOAAnm pin output level varies.**

**Remark**   m = 0, 1

**(4) TAAn I/O control register 1 (TAAnIOC1)**

The TAAnIOC1 register is an 8-bit register that controls the valid edge of the capture trigger input signals (TIAAn0, TIAAn1 pins).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

---

After reset: 00H     R/W     Address:     TAA0IOC1 FFFFF633H, TAA1IOC1 FFFFF643H,
TAA2IOC1 FFFFF653H, TAA3IOC1 FFFFF663H,
TAA5IOC1 FFFFF683H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TAAnIOC1 | 0 | 0 | 0 | 0 | TAAnIS3 | TAAnIS2 | TAAnIS1 | TAAnIS0 |

(n = 0 to 3, 5)

| TAAnIS3 | TAAnIS2 | Capture trigger input signal (TIAAn1 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (capture operation invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

| TAAnIS1 | TAAnIS0 | Capture trigger input signal (TIAAn0 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (capture operation invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

**Cautions 1. Rewrite the TAAnIS3 to TAAnIS0 bits when the TAAnCTL0.TAAnCE bit = 0. (The same value can be written when the TAAnCE bit = 1.) If rewriting was mistakenly performed, clear the TAAnCE bit to 0 and then set the bits again.**

**2. The TAAnIS3 to TAAnIS0 bits are valid only in the free-running timer mode and the pulse width measurement mode. In all other modes, a capture operation is not performed.**

---

**(5) TAAn I/O control register 2 (TAAnIOC2)**

The TAAnIOC2 register is an 8-bit register that controls the valid edge of the external event count input signal (TIAAn0 pin) and external trigger input signal (TIAAn0 pin).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address:    TAA0IOC2  FFFFF634H, TAA1IOC2  FFFFF644H,
                                       TAA2IOC2  FFFFF654H, TAA3IOC2  FFFFF664H.
                                       TAA5IOC2  FFFFF684H

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TAAnIOC2 (n = 0 to 3, 5) | 0 | 0 | 0 | 0 | TAAnEES1 | TAAnEES0 | TAAnETS1 | TAAnETS0 |

| TAAnEES1 | TAAnEES0 | External event count input signal (TIAAn0 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (external event count invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

| TAAnETS1 | TAAnETS0 | External trigger input signal (TIAAn0 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (external trigger invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

**Cautions  1. Rewrite the TAAnEES1, TAAnEES0, TAAnETS1, and TAAnETS0 bits when the TAAnCTL0.TAAnCE bit = 0. (The same value can be written when the TAAnCE bit = 1.)  If rewriting was mistakenly performed, clear the TAAnCE bit to 0 and then set the bits again.**

**2. The TAAnEES1 and TAAnEES0 bits are valid only when the TAAnCTL1.TAAnEEE bit = 1 or when the external event count mode (TAAnCTL1.TAAnMD2 to TAAnCTL1.TAAnMD0 bits = 001) has been set.**

**3. The TAAnETS1 and TAAnETS0 bits are valid only when the external trigger pulse output mode (TAAnCTL1.TAAnMD2 to TAAnCTL1.TAAnMD0 bits = 010) or the one-shot pulse output mode (TAAnCTL1.TAAnMD2 to TAAnCTL1.TAAnMD0 = 011) is set.**

**(6)  TAAn I/O control register 4 (TAAnIOC4)**

The TAAnIOC4 register is an 8-bit register that controls the timer output.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.  This register is not reset by stopping the timer operation (TAAnCTL0.TAAnCE = 0).

**Cautions 1.  Accessing the TAAnIOC4 register is prohibited in the following statuses.  For details, see 3.4.8**
**(2)  Accessing specific on-chip peripheral I/O registers.**
- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

**2.  The TAAnIOC4 register can be set only in the interval timer mode and free-running timer mode.**
**Be sure to set the TAAnIOC4 register to 00H in all other modes (for details of the mode setting,**
**see 7.4 (2) TAAn control register 1 (TAAnCTL1)).  The TAAnIOC4 register setting is invalid if the**
**TAAnCCR0 and TAAnCCR1 registers are set to the capture function, even if the free-running**
**timer mode is set.**

After reset: 00H     R/W     Address:     TAA0IOC4 FFFFF63CH, TAA1IOC4 FFFFF64CH,
                                          TAA2IOC4 FFFFF65CH, TAA3IOC4 FFFFF66CH,
                                          TAA5IOC4 FFFFF68CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TAAnIOC4 | 0 | 0 | 0 | 0 | TAAnOS1 | TAAnOR1 | TAAnOS0 | TAAnOR0 |

(n = 0 to 3, 5)

| TAAnOS1 | TAAnOR1 | Toggle control of TIAAn1 pin |
|---|---|---|
| 0 | 0 | No request.  Normal toggle operation. |
| 0 | 1 | Reset request<br>Fix to inactive level upon next match between value of 16-bit counter and value of TAAnCCR1 register. |
| 1 | 0 | Set request<br>Fix to active level upon next match between value of 16-bit counter and value of TAAnCCR1 register. |
| 1 | 1 | Keep request<br>Keep current output level. |

| TAAnOS0 | TAAnOR0 | Toggle control of TIAAn0 |
|---|---|---|
| 0 | 0 | No request.  Normal toggle operation. |
| 0 | 1 | Reset request<br>Fix to inactive level upon next match between value of 16-bit counter and value of TAAnCCR0 register. |
| 1 | 0 | Set request<br>Fix to active level upon next match between value of 16-bit counter and value of TAAnCCR0 register. |
| 1 | 1 | Keep request<br>Keep current output level. |

**(7) TAAn option register 0 (TAAnOPT0)**

The TAAnOPT0 register is an 8-bit register used to set the capture/compare operation and detect an overflow.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H     R/W     Address:     TAA0OPT0 FFFFF635H, TAA1OPT0 FFFFF645H,
TAA2OPT0 FFFFF655H, TAA3OPT0 FFFFF665H,
TAA5OPT0 FFFFF685H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TAAnOPT0 | 0 | 0 | TAAnCCS1 | TAAnCCS0 | 0 | 0 | 0 | TAAnOVF |

(n = 0 to 3, 5)

| TAAnCCS1 | TAAnCCR1 register capture/compare selection |
|---|---|
| 0 | Compare register selected |
| 1 | Capture register selected |
| The TAAnCCS1 bit setting is valid only in the free-running timer mode. | |

| TAAnCCS0 | TAAnCCR0 register capture/compare selection |
|---|---|
| 0 | Compare register selected |
| 1 | Capture register selected |
| The TAAnCCS0 bit setting is valid only in the free-running timer mode. | |

| TAAnOVF | TAAn overflow detection flag |
|---|---|
| Set (1) | Overflow occurred |
| Reset (0) | 0 written to TAAnOVF bit or TAAnCTL0.TAAnCE bit = 0 |

- The TAAnOVF bit is set to 1 when the 16-bit counter count value overflows from FFFFH to 0000H in the free-running timer mode or the pulse width measurement mode.
- An interrupt request signal (INTTAAnOV) is generated at the same time that the TAAnOVF bit is set to 1. The INTTAAnOV signal is not generated in modes other than the free-running timer mode and the pulse width measurement mode.
- The TAAnOVF bit is not cleared even when the TAAnOVF bit or the TAAnOPT0 register are read when the TAAnOVF bit = 1.
- The TAAnOVF bit can be both read and written, but the TAAnOVF bit cannot be set to 1 by software. Writing 1 has no influence on the operation of TAAn.

**Cautions 1. Rewrite the TAAnCCS1 and TAAnCCS0 bits when the TAAnCE bit = 0. (The same value can be written when the TAAnCE bit = 1.) If rewriting was mistakenly performed, clear the TAAnCE bit to 0 and then set the bits again.**

**2. Be sure to set bits 1 to 3, 6, and 7 to "0".**

**(8) TAAn option register 1 (TAAnOPT1)**

The TAAnOPT1 register is an 8-bit register that controls the 32-bit capture function realized by a cascade connection.

Rewriting this register is prohibited while the timer is operating (TAAnCTL0.TAAnCE = 1).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H     R/W     Address:     TAA0OPT1 FFFFF63DH, TAA2OPT1 FFFFF65DH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TAAnOPT1 | TAAnCSE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

(n = 0, 2)

| TAAnCSE | Cascade control |
|---|---|
| 0 | Individual operation or operation as lower side of cascade function |
| 1 | Operation as higher side of cascade function |

**Cautions  1. Cascade connection and timer-tuned operation cannot be used together.  Be sure to set TAAnCTL1.TAAnSYE to 0 for a cascade connection.**

**2. For a cascade connection, set the free-running timer mode and use the TAAnCCR0 and TAAnCCR1 registers as capture registers.**

**For details of cascade connection, see 7.8  Cascade Connection.**

RENESAS

**(9)  TAAn capture/compare register 0 (TAAnCCR0)**

The TAAnCCR0 register can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, depending on the setting of the TAAnOPT0.TAAnCCS0 bit.  In the pulse width measurement mode, the TAAnCCR0 register can be used only as a capture register.  In any other mode, this register can be used only as a compare register.

The TAAnCCR0 register can be read or written during operation.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

**Caution   Accessing the TAAnCCR0 register is prohibited in the following statuses.  For details, see 3.4.8**
**(2)  Accessing specific on-chip peripheral I/O registers.**
- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

```
        After reset:  0000H     R/W     Address:     TAA0CCR0  FFFFF636H, TAA1CCR0  FFFFF646H,
                                                      TAA2CCR0  FFFFF656H, TAA3CCR0  FFFFF666H,
                                                      TAA4CCR0  FFFFF676H, TAA5CCR0  FFFFF686H

                        15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
              TAAnCCR0 [   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   ]
              (n = 0 to 5)
```
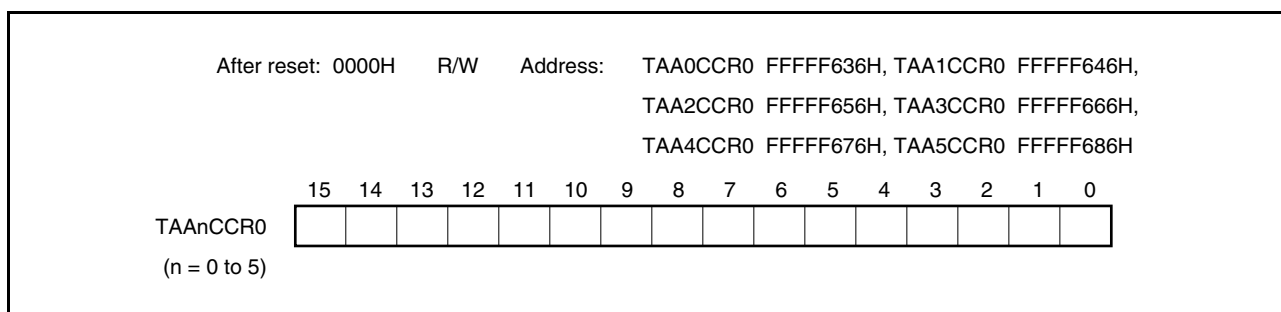
**(a) Function as compare register**

The TAAnCCR0 register can be rewritten even when the TAAnCTL0.TAAnCE bit = 1.

The set value of the TAAnCCR0 register is transferred to the CCR0 buffer register. When the value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTAAnCC0) is generated. If TOAAn0 pin output is enabled at this time, the output of the TOAAn0 pin is inverted.

When the TAAnCCR0 register is used as a cycle register in the interval timer mode, external event count mode, external trigger pulse output mode, one-shot pulse output mode, or PWM output mode, the value of the 16-bit counter is cleared (0000H) if its count value matches the value of the CCR0 buffer register.

**(b) Function as capture register**

When the TAAnCCR0 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TAAnCCR0 register if the valid edge of the capture trigger input pin (TIAAn0 pin) is detected. In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TAAnCCR0 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIAAn0) is detected.

Even if the capture operation and reading the TAAnCCR0 register conflict, the correct value of the TAAnCCR0 register can be read.

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

**Table 7-3.  Function of Capture/Compare Register in Each Mode and How to Write Compare Register**

| Operation Mode | Capture/Compare Register | How to Write Compare Register |
|---|---|---|
| Interval timer | Compare register | Anytime write |
| External event counter | Compare register | Anytime write |
| External trigger pulse output | Compare register | Batch write |
| One-shot pulse output | Compare register | Anytime write |
| PWM output | Compare register | Batch write |
| Free-running timer | Capture/compare register | Anytime write |
| Pulse width measurement | Capture register | – |

**(10) TAAn capture/compare register 1 (TAAnCCR1)**

The TAAnCCR1 register can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, depending on the setting of the TAAnOPT0.TAAnCCS1 bit. In the pulse width measurement mode, the TAAnCCR1 register can be used only as a capture register. In any other mode, this register can be used only as a compare register.

The TAAnCCR1 register can be read or written during operation.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

**Caution**   **Accessing the TAAnCCR1 register is prohibited in the following statuses. For details, see 3.4.8 (2) Accessing specific on-chip peripheral I/O registers.**

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

After reset: 0000H    R/W    Address:   TAA0CCR1 FFFFF638H, TAA1CCR1 FFFFF648H,
                                        TAA2CCR1 FFFFF658H, TAA3CCR1 FFFFF668H,
                                        TAA4CCR1 FFFFF678H, TAA5CCR1 FFFFF688H

|  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TAAnCCR1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

(n = 0 to 5)

**(a) Function as compare register**

The TAAnCCR1 register can be rewritten even when the TAAnCTL0.TAAnCE bit = 1.

The set value of the TAAnCCR1 register is transferred to the CCR1 buffer register. When the value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTAAnCC1) is generated. If TOAAn1 pin output is enabled at this time, the output of the TOAAn1 pin is inverted.

**(b) Function as capture register**

When the TAAnCCR1 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TAAnCCR1 register if the valid edge of the capture trigger input pin (TIAAn1 pin) is detected. In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TAAnCCR1 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIAAn1) is detected.

Even if the capture operation and reading the TAAnCCR1 register conflict, the correct value of the TAAnCCR1 register can be read.

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

**Table 7-4.  Function of Capture/Compare Register in Each Mode and How to Write Compare Register**

| Operation Mode | Capture/Compare Register | How to Write Compare Register |
|---|---|---|
| Interval timer | Compare register | Anytime write |
| External event counter | Compare register | Anytime write |
| External trigger pulse output | Compare register | Batch write |
| One-shot pulse output | Compare register | Anytime write |
| PWM output | Compare register | Batch write |
| Free-running timer | Capture/compare register | Anytime write |
| Pulse width measurement | Capture register | – |

**(11) TAAn counter read buffer register (TAAnCNT)**

The TAAnCNT register is a read buffer register that can read the count value of the 16-bit counter.

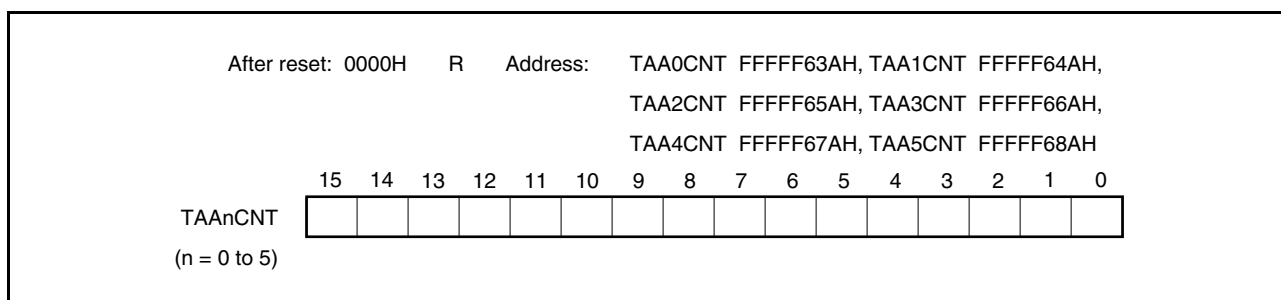If this register is read when the TAAnCTL0.TAAnCE bit = 1, the count value of the 16-bit timer can be read.

This register is read-only, in 16-bit units.

The value of the TAAnCNT register is cleared to 0000H when the TAAnCE bit = 0. If the TAAnCNT register is read at this time, the value of the 16-bit counter (FFFFH) is not read, but 0000H is read.

Reset clears the TAAnCE bit to 0. Therefore, the value of the TAAnCNT register is cleared to 0000H.

**Caution**  **Accessing the TAAnCNT register is prohibited in the following statuses. For details, see 3.4.8 (2)
Accessing specific on-chip peripheral I/O registers.**

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

After reset: 0000H    R    Address:    TAA0CNT  FFFFF63AH, TAA1CNT  FFFFF64AH,
TAA2CNT  FFFFF65AH, TAA3CNT  FFFFF66AH,
TAA4CNT  FFFFF67AH, TAA5CNT  FFFFF68AH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TAAnCNT | | | | | | | | | | | | | | | | |

(n = 0 to 5)

**(12) Noise elimination control register (TANFC)**

Digital noise elimination can be selected for the TIAAn0 and TIAAn1 pins.  The noise elimination setting is selected using the TANFC register.

When digital noise elimination is selected, the sampling clock for digital sampling can be selected from among $f_{XX}$ and $f_{XX}/4$.  Sampling is performed 3 times.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**Caution　Time equal to the sampling clock × 3 clocks is required until the digital noise eliminator is initialized after the sampling clock has been changed.  If the valid edge of TIAAn0 and TIAAn1 is input after the sampling clock has been changed and before the time of the sampling clock × 3 clocks passes, therefore, an interrupt request signal may be generated.  Therefore, when using the external trigger function, the external event function, and the capture trigger function of TAA, enable TAA operation after the time of the sampling clock × 3 clocks has elapsed.**

**Remark**　n = 0 to 3, 5

After reset:  00H　　　R/W　　　Address:  FFFFF724H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TANFC | TANFEN | 0 | 0 | 0 | 0 | 0 | 0 | TANFC0 |

| TANFEN | Setting of digital noise elimination |
|---|---|
| 0 | Does not perform digital noise elimination |
| 1 | Performs digital noise elimination |

| TANFC0 | Digital sampling clock |
|---|---|
| 0 | $f_{XX}$ |
| 1 | $f_{XX}/4$ |

**Remarks　1.**　Since sampling is performed 3 times, the noise width for reliably eliminating noise is 2 sampling clocks.
**2.**　In the case of noise with a width smaller than 2 sampling clocks, an interrupt request signal is generated if noise synchronized with the sampling clock is input.

A timing example of noise elimination performed by the timer AA input pin digital filter is shown Figure 7-2.

**Figure 7-2.  Example of Digital Noise Elimination Timing**



**Remark**    If there are two or fewer noise elimination clocks while the TIAAn0 or TIAAn1 input signal is high
level (or low level), that input signal is eliminated as noise.  If it is sampled three times or more,
the edge is detected as a valid input (n = 0 to 3, 5).
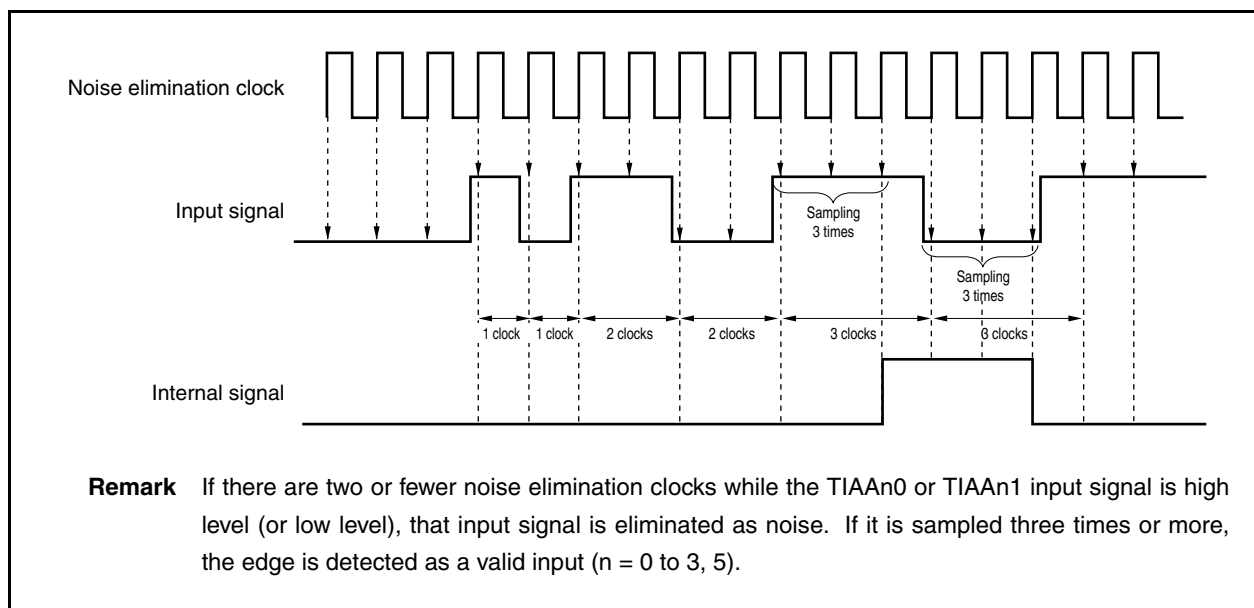
## 7.5 Operation

TAAn can perform the following operations.

| Operation | TAAnCTL1.TAAmEST Bit (Software Trigger Bit) | TIAAn0 Pin (External Trigger Input) | Capture/Compare Register Setting | Compare Register Write |
|---|---|---|---|---|
| Interval timer mode | Invalid | Invalid | Compare only | Anytime write |
| External event count mode[Note 1] | Invalid | Invalid | Compare only | Anytime write |
| External trigger pulse output mode[Note 2] | Valid | Valid | Compare only | Batch write |
| One-shot pulse output mode[Note 2] | Valid | Valid | Compare only | Anytime write |
| PWM output mode | Invalid | Invalid | Compare only | Batch write |
| Free-running timer mode | Invalid | Invalid | Switching enabled | Anytime write |
| Pulse width measurement mode[Note 2] | Invalid | Invalid | Capture only | Not applicable |

**Notes 1.** To use the external event count mode, specify that the valid edge of the TIAAn0 pin capture trigger input is not detected (by clearing the TAAnIOC1.TAAnIS1 and TAAnIOC1.TAAnIS0 bits to "00").

    **2.** When using the external trigger pulse output mode, one-shot pulse output mode, and pulse width measurement mode, select the internal clock as the count clock (by clearing the TAAnCTL1.TAAnEEE bit to 0).

**Remark**    n = 0 to 3, 5

**(1) Anytime write and batch write**

With TAAn, the TAAnCCR0 and TAAnCCR1 registers can be rewritten during timer operation (TAAnCTL0.TAAnCE bit = 1), but the write method (anytime write, batch write) of the CCR0 and CCR1 buffer registers differs depending on the mode.

**(a) Anytime write**

In this mode, data is transferred at any time from the TAAnCCR0 and TAAnCCR1 registers to the CCR0 and CCR1 buffer registers during timer operation.

**Figure 7-3. Example of Basic Anytime Write Operation Flowchart (Interval Timer Mode of TAA0)**



**Note** The 16-bit counter is not cleared upon a match between the 16-bit counter value and the CCR1 buffer register value. It is cleared upon a match between the 16-bit counter value and the CCR0 buffer register value.

**Remark** n = 0, 1
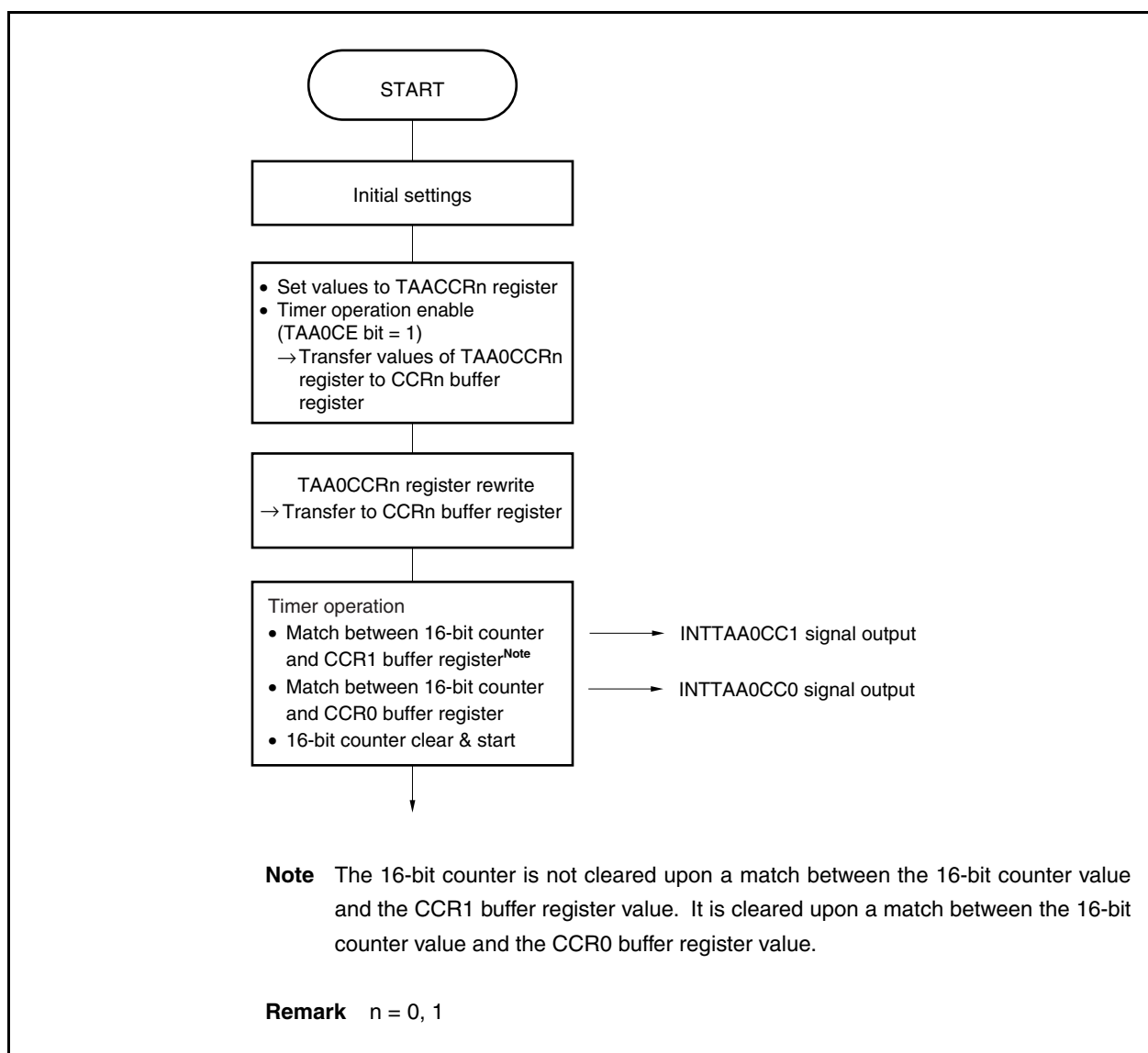
**Figure 7-4. Example of Anytime Write Timing (Interval Timer Mode of TAA0)**



Remark    $D_{01}$, $D_{02}$: Set values of TAA0CCR0 register
              $D_{11}$, $D_{12}$: Set values of TAA0CCR1 register
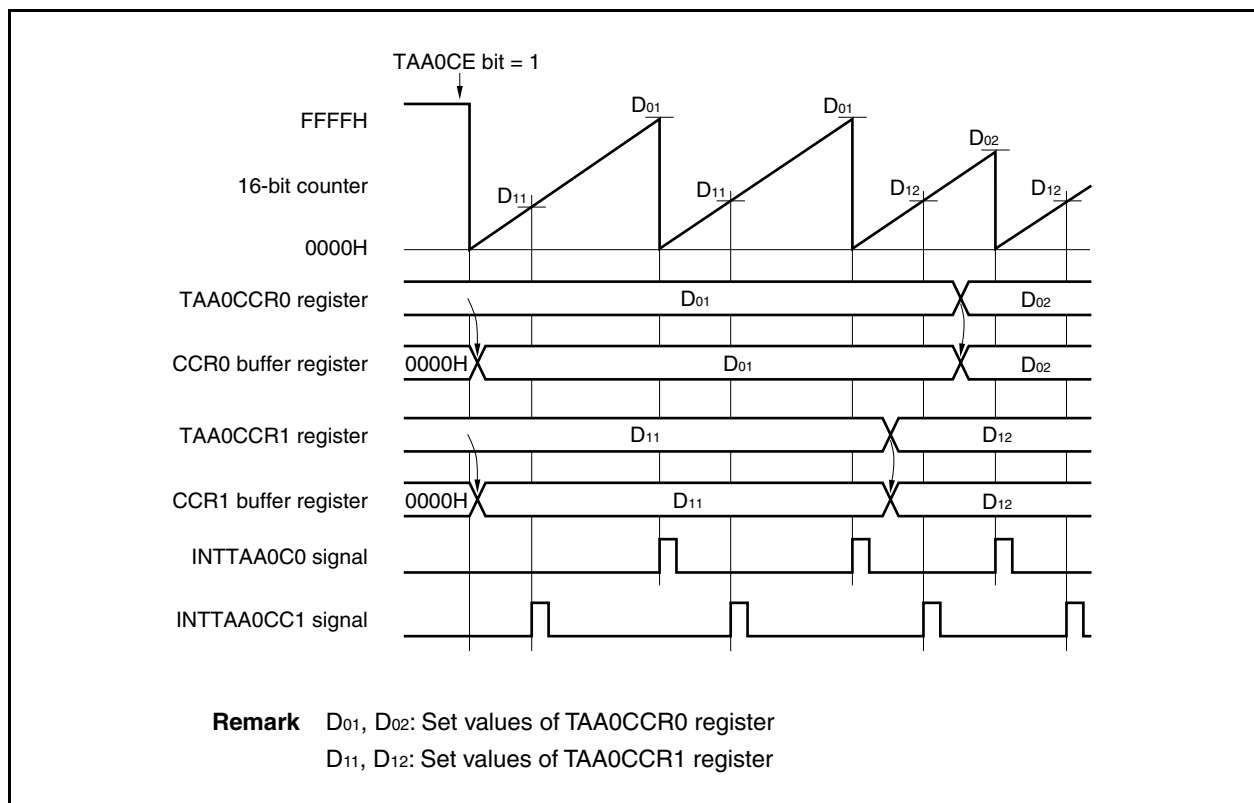
**(b) Batch write**

In this mode, data is transferred all at once from the TAAnCCR0 and TAAnCCR1 registers to the CCR0 and CCR1 buffer registers during timer operation. This data is transferred upon a match between the value of the CCR0 buffer register and the value of the 16-bit counter. Transfer is enabled by writing to the TAAnCCR1 register. Whether to enable or disable the next transfer timing is controlled by writing or not writing to the TAAnCCR1 register.

In order for the set value when the TAAnCCR0 and TAAnCCR1 registers are rewritten to become the 16-bit counter comparison value (in other words, in order for this value to be transferred to the CCR0 and CCR1 buffer registers), it is necessary to rewrite the TAAnCCR0 register and then write to the TAAnCCR1 register before the 16-bit counter value and the CCR0 buffer register value match. Therefore, the values of the TAAnCCR0 and TAAnCCR1 registers are transferred to the CCR0 and CCR1 buffer registers upon a match between the count value of the 16-bit counter and the value of the CCR0 buffer register. Thus even when wishing only to rewrite the value of the TAAnCCR0 register, also write the same value (same as preset value of the TAAnCCR1 register) to the TAAnCCR1 register.

**Figure 7-5. Example of Basic Batch Write Operation Flowchart (PWM Output Mode of TAA0)**



**Note**  The 16-bit counter is not cleared upon a match between the 16-bit counter value and the CCR1 buffer register value.  It is cleared upon a match between the 16-bit counter value and the CCR0 buffer register value.

**Caution**   **Writing to the TAA0CCR1 register includes enabling of batch write.  Thus, rewrite the TAA0CCR1 register after rewriting the TAA0CCR0 register.**

**Remark**   n = 0, 1

**Figure 7-6.  Timing of Batch Write (Interval Timer Mode of TAA0)**
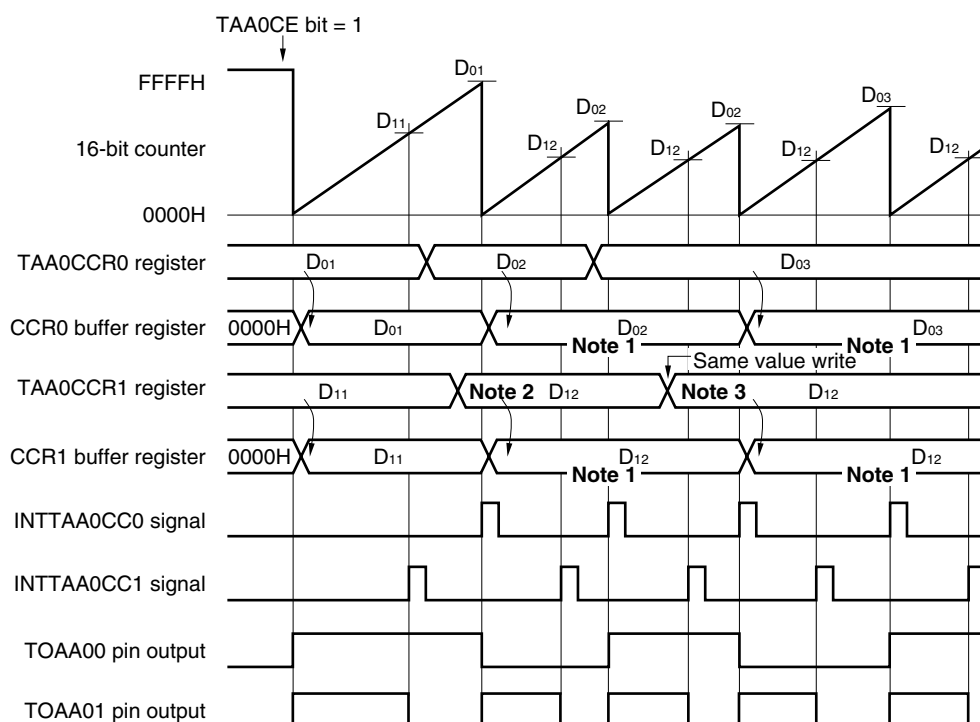


**Notes 1.** Because the TAA0CCR1 register was not rewritten, $D_{03}$ is not transferred.

**2.** Because the TAA0CCR1 register has been written ($D_{12}$), data is transferred to the CCR1 buffer register upon a match between the value of the 16-bit counter and the value of the TAA0CCR0 register ($D_{01}$).

**3.** Because the TAA0CCR1 register has been written ($D_{12}$), data is transferred to the CCR1 buffer register upon a match between the value of the 16-bit counter and the value of the TAA0CCR0 register ($D_{02}$).

**Remark**   $D_{01}$, $D_{02}$, $D_{03}$: Set values of TAA0CCR0 register
$D_{11}$, $D_{12}$:      Set values of TAA0CCR1 register

### 7.5.1 Interval timer mode (TAAmMD2 to TAAmMD0 bits = 000)

In the interval timer mode, an interrupt request signal (INTTAAnCC0) is generated at any interval if the TAAnCTL0.TAAnCE bit is set to 1. A square wave whose half cycle is equal to the interval can be output from the TOAAn0 pin.

Usually, the TAAnCCR1 register is not used in the interval timer mode.

**Figure 7-7. Configuration of Interval Timer**



Remark  m = 0 to 3, 5
n = 0 to 5
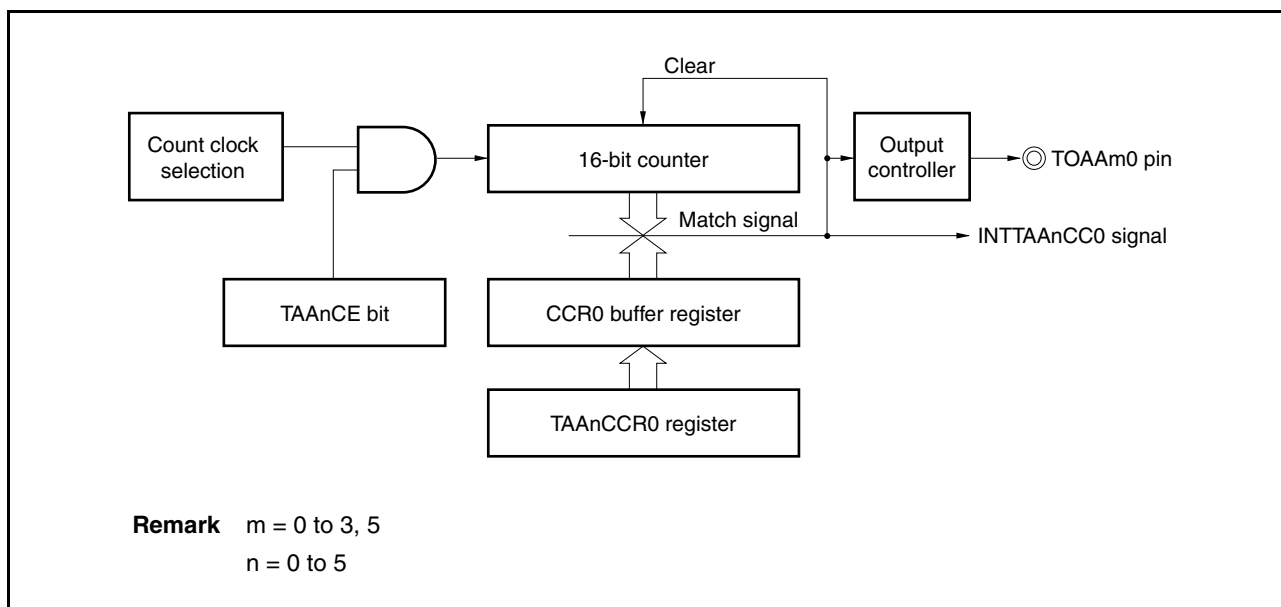
**Figure 7-8. Basic Timing of Operation in Interval Timer Mode**



Remark  m = 0 to 3, 5
n = 0 to 5

When the TAAnCE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H in synchronization with the count clock, and the counter starts counting. At this time, the output of the TOAAn0 pin is inverted. Additionally, the set value of the TAAnCCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, the output of the TOAAn0 pin is inverted, and a compare match interrupt request signal (INTTAAnCC0) is generated.

The interval can be calculated by the following expression.

Interval = (Set value of TAAnCCR0 register + 1) $\times$ Count clock cycle

**Remark**   m = 0 to 3, 5
            n = 0 to 5

**Figure 7-9.  Register Settings for Interval Timer Mode Operation (1/2)**



**Note**   This bit can be set to 1 only when the interrupt request signals (INTTAAmCC0 and INTTAAmCC1) are masked by the interrupt mask flags (TAAmCCMK0 and TAAmCCMK1) and timer output (TOAAm1) is performed. However, set the TAAmCCR0 and TAAmCCR1 registers to the same value (see **7.5.1 (2) (d) Operation of TAAnCCR1 register**).

**Remark**   m = 0 to 3, 5
            n = 0 to 5

**Figure 7-9.  Register Settings for Interval Timer Mode Operation (2/2)**

**(c)  TAAm I/O control register 0 (TAAmIOC0)**

TAAmOL1 TAAmOE1 TAAmOL0 TAAmOE0

| TAAmIOC0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |
|---|---|---|---|---|---|---|---|---|

0: Disables TOAAm0 pin output
1: Enables TOAAm0 pin output

Setting of output level with
operation of TOAAm0 pin disabled
0: Low level
1: High level

0: Disables TOAAm1 pin output
1: Enables TOAAm1 pin output

Setting of output level with
operation of TOAAm1 pin disabled
0: Low level
1: High level

**(d)  TAAn counter read buffer register (TAAnCNT)**

By reading the TAAnCNT register, the count value of the 16-bit counter can be read.

**(e)  TAAn capture/compare register 0 (TAAnCCR0)**

If the TAAnCCR0 register is set to $D_0$, the interval is as follows.

Interval = $(D_0 + 1) \times$ Count clock cycle

**(f)  TAAn capture/compare register 1 (TAAnCCR1)**

Usually, the TAAnCCR1 register is not used in the interval timer mode.  However, the set value of the TAAnCCR1 register is transferred to the CCR1 buffer register.  A compare match interrupt request signal (INTTAAnCC1) is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

Therefore, mask the interrupt request by using the corresponding interrupt mask flag (TAAnCCMK1).

**Remarks  1.**  TAAm I/O control register 1 (TAAmIOC1), TAAm I/O control register 2 (TAAmIOC2), and TAAm option register 0 (TAAmOPT0) are not used in the interval timer mode.

**2.**  m = 0 to 3, 5
n = 0 to 5

**(1) Interval timer mode operation flow**

**Figure 7-10. Software Processing Flow in Interval Timer Mode**



**Remark** m = 0 to 3, 5

n = 0 to 5

**(2)  Interval timer mode operation timing**

**(a)  Operation if TAAnCCR0 register is set to 0000H**

If the TAAnCCR0 register is set to 0000H, the INTTAAnCC0 signal is generated at each count clock subsequent to the first count clock, and the output of the TOAAm0 pin is inverted.

The value of the 16-bit counter is always 0000H.



**Remark**   m = 0 to 3, 5

n = 0 to 5
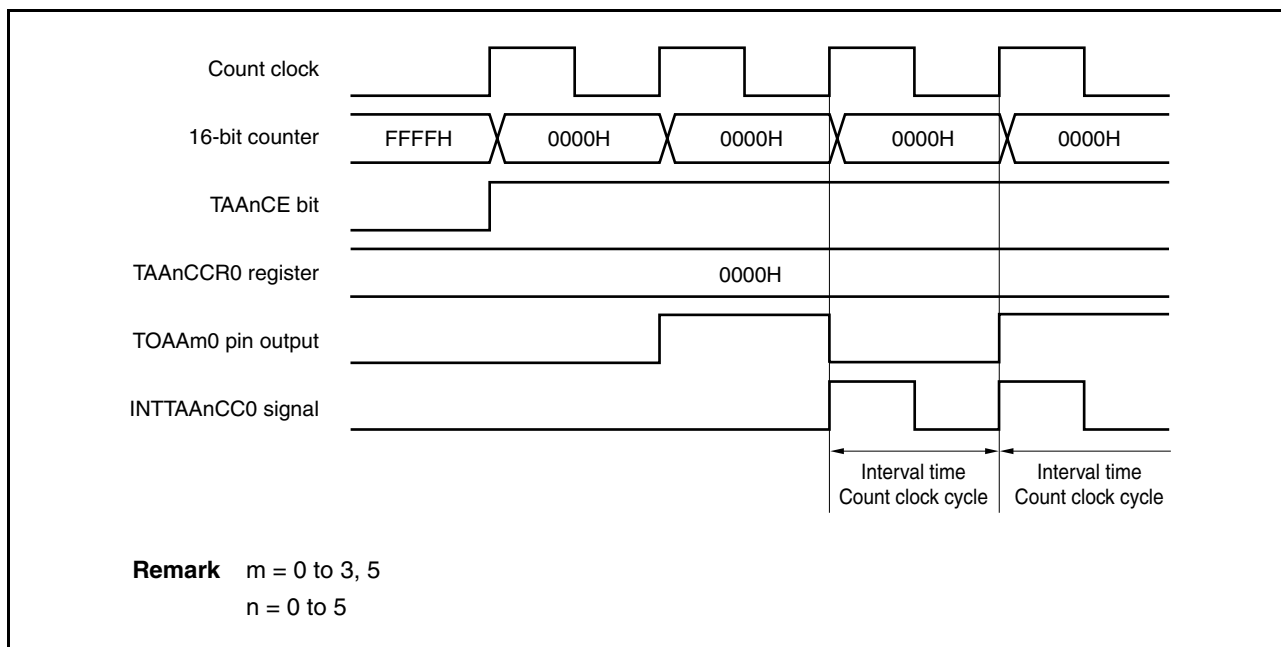
**(b) Operation if TAAnCCR0 register is set to FFFFH**

If the TAAnCCR0 register is set to FFFFH, the 16-bit counter counts up to FFFFH.  The counter is cleared to 0000H in synchronization with the next count-up timing.  The INTTAAnCC0 signal is generated and the output of the TOAAm0 pin is inverted.  At this time, an overflow interrupt request signal (INTTAAnOV) is not generated, nor is the overflow flag (TAAmOPT0.TAAmOVF bit) set to 1.
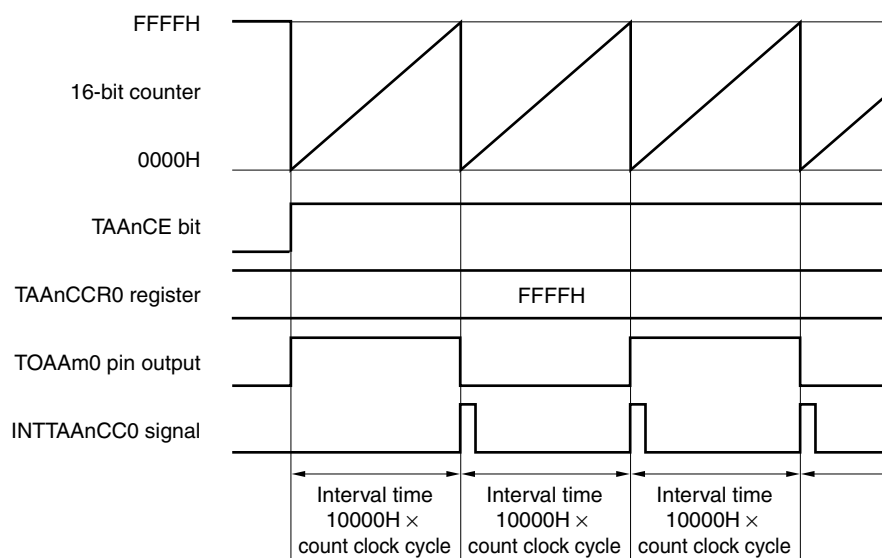


Remark   m = 0 to 3, 5
         n = 0 to 5

**(c) Notes on rewriting TAAnCCR0 register**

To change the value of the TAAnCCR0 register to a smaller value, stop counting once and then change the set value.

If the value of the TAAnCCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow.
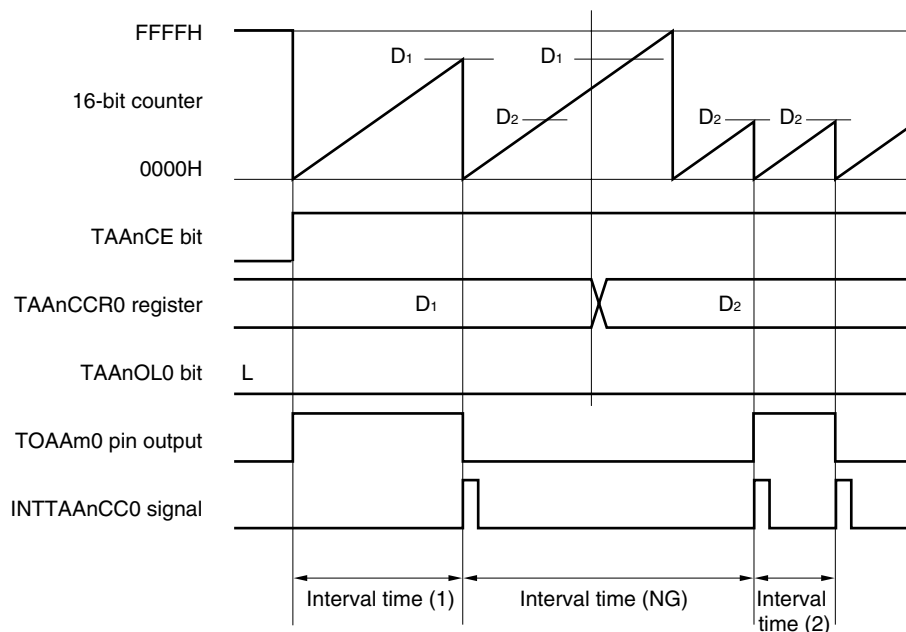


Remarks   **1.**   Interval time (1):     $(D_1 + 1) \times$ Count clock cycle

                       Interval time (NG): $(10000H + D_2 + 1) \times$ Count clock cycle

                       Interval time (2):     $(D_2 + 1) \times$ Count clock cycle

       **2.**   m = 0 to 3, 5

              n = 0 to 5

If the value of the TAAnCCR0 register is changed from $D_1$ to $D_2$ while the count value is greater than $D_2$ but less than $D_1$, the count value is transferred to the CCR0 buffer register as soon as the TAAnCCR0 register has been rewritten. Consequently, the value of the 16-bit counter that is compared is $D_2$.

Because the count value has already exceeded $D_2$, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H. When the count value matches $D_2$, the INTTAAnCC0 signal is generated and the output of the TOAAm0 pin is inverted.

Therefore, the INTTAAnCC0 signal may not be generated at the interval time "$(D_1 + 1) \times$ Count clock cycle" or "$(D_2 + 1) \times$ Count clock cycle" as originally expected, but may be generated at an interval of "$(10000H + D_2 + 1) \times$ Count clock period".

**(d) Operation of TAAnCCR1 register**

**Figure 7-11. Configuration of TAAnCCR1 Register**



**Remark**    n = 0 to 3, 5
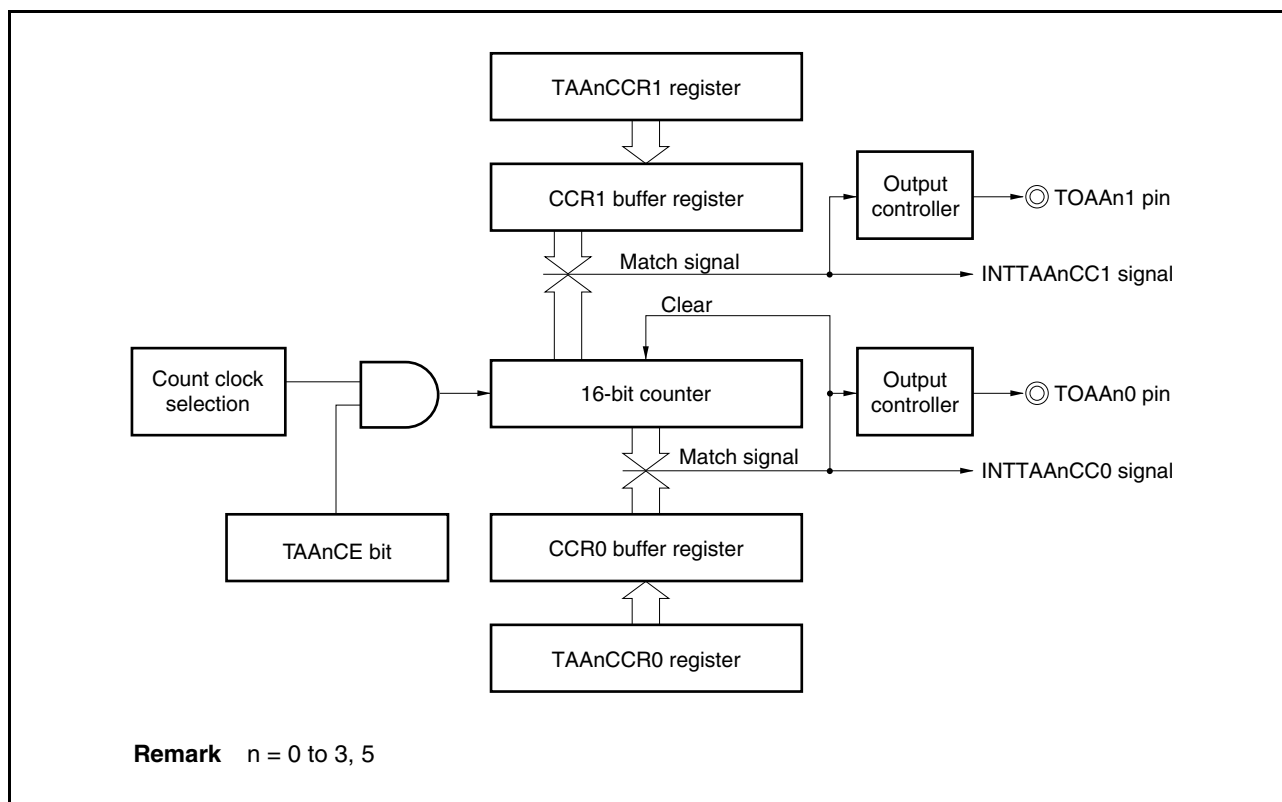
If the set value of the TAAnCCR1 register is less than the set value of the TAAnCCR0 register, the INTTAAnCC1 signal is generated once per cycle. At the same time, the output of the TOAAn1 pin is inverted. The TOAAn1 pin outputs a square wave with the same cycle as that output by the TOAAn0 pin.

**Figure 7-12.  Timing Chart When $D_{01} \geq D_{11}$**
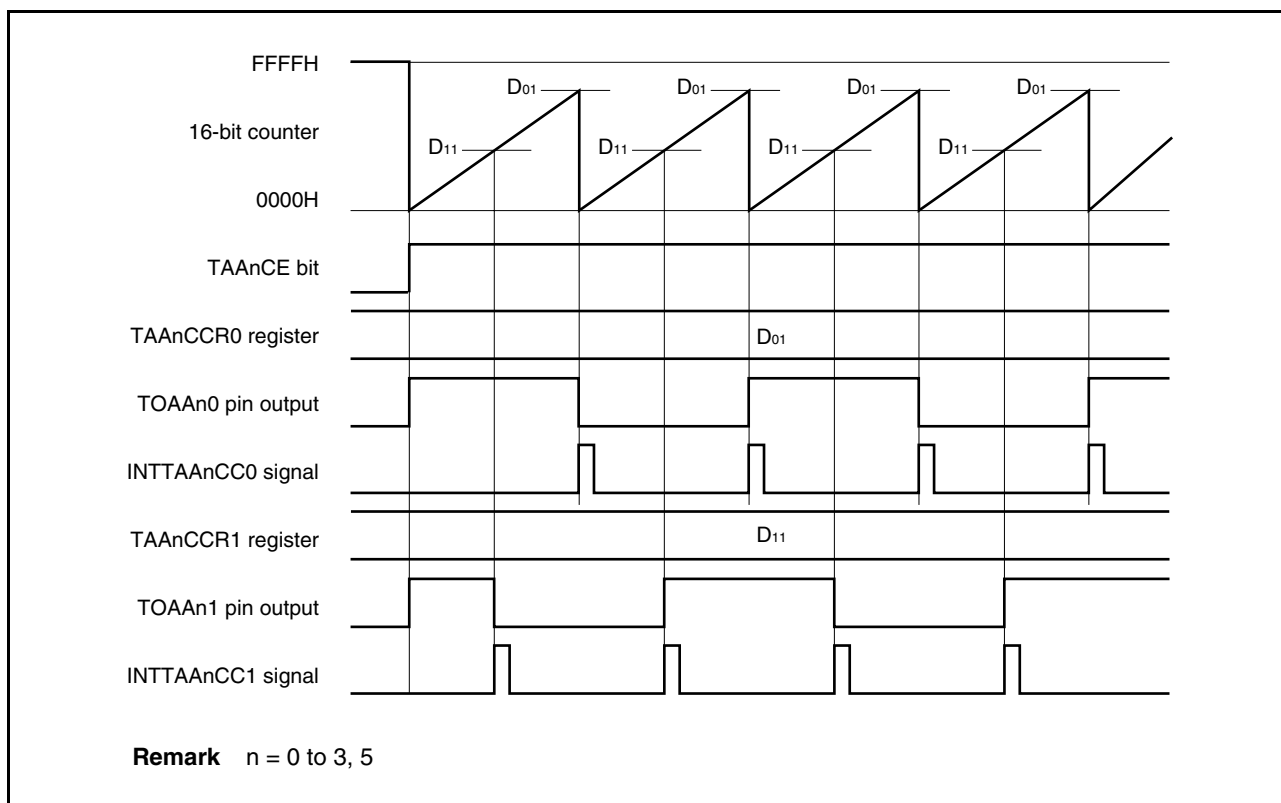


**Remark**   n = 0 to 3, 5

If the set value of the TAAnCCR1 register is greater than the set value of the TAAnCCR0 register, the count value of the 16-bit counter does not match the value of the TAAnCCR1 register.  Consequently, the INTTAAnCC1 signal is not generated, nor is the output of the TOAAn1 pin changed.

**Figure 7-13.  Timing Chart When $D_{01} < D_{11}$**



**Remark**    n = 0 to 3, 5
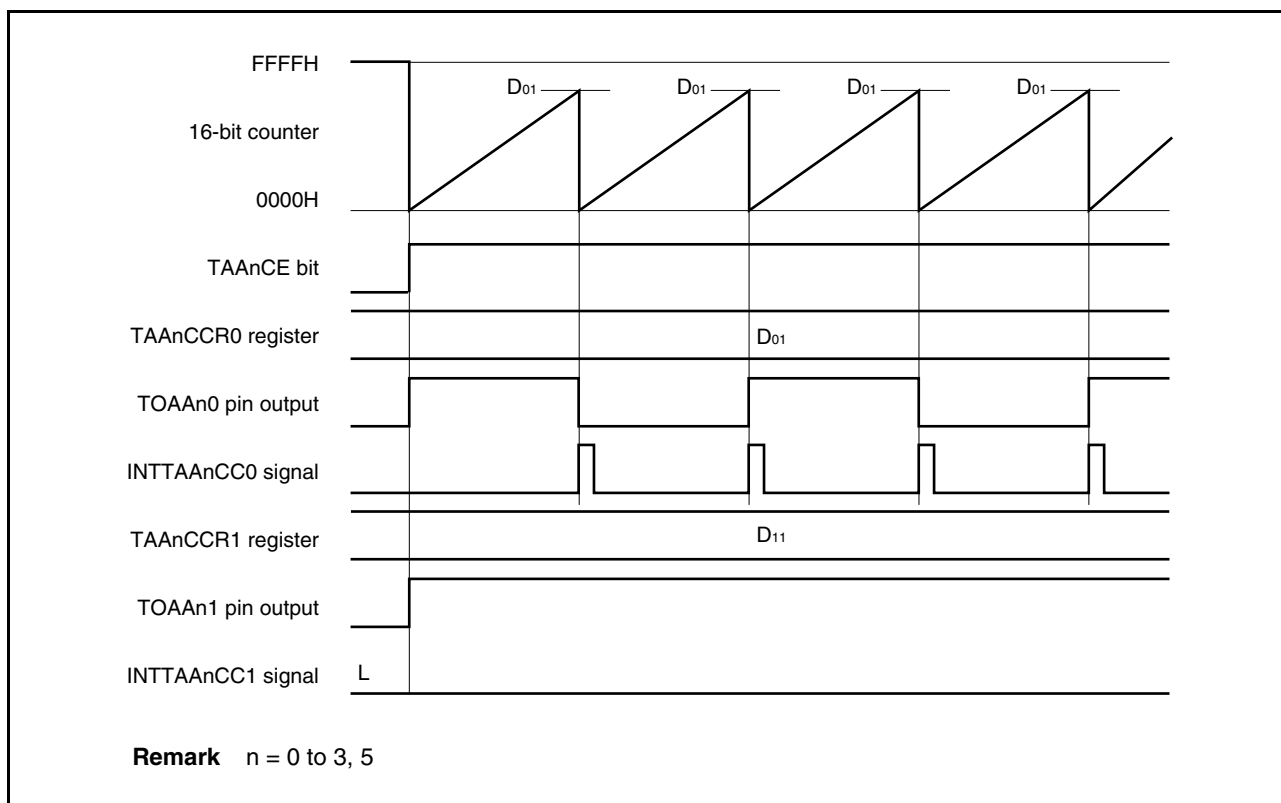
### 7.5.2   External event count mode (TAAnMD2 to TAAnMD0 bits = 001)

In the external event count mode, the valid edge of the external event count input is counted when the TAAnCTL0.TAAnCE bit is set to 1, and an interrupt request signal (INTTAAnCC0) is generated each time the specified number of edges have been counted. The TOAAn0 pin cannot be used.

Usually, the TAAnCCR1 register is not used in the external event count mode.

#### Figure 7-14.  Configuration in External Event Count Mode



**Remark**   n = 0 to 3, 5
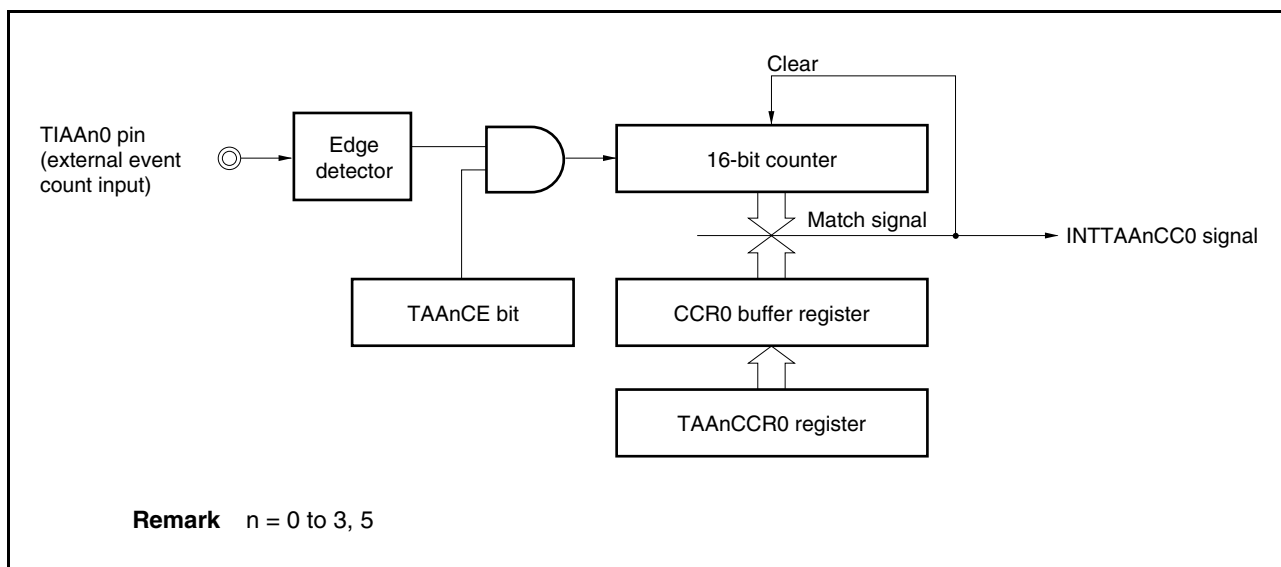
#### Figure 7-15.  Basic Timing in External Event Count Mode



**Remarks  1.**  This figure shows the basic timing when the rising edge is specified as the valid edge of the external event count input.

**2.**  n = 0 to 3, 5

When the TAAnCE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H.  The counter counts each time the valid edge of external event count input is detected.  Additionally, the set value of the TAAnCCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, and a compare match interrupt request signal (INTTAAnCC0) is generated.

The INTTAAnCC0 signal is generated each time the valid edge of the external event count input has been detected (set value of TAAnCCR0 register + 1) times.

**Figure 7-16.  Register Setting for Operation in External Event Count Mode (1/2)**
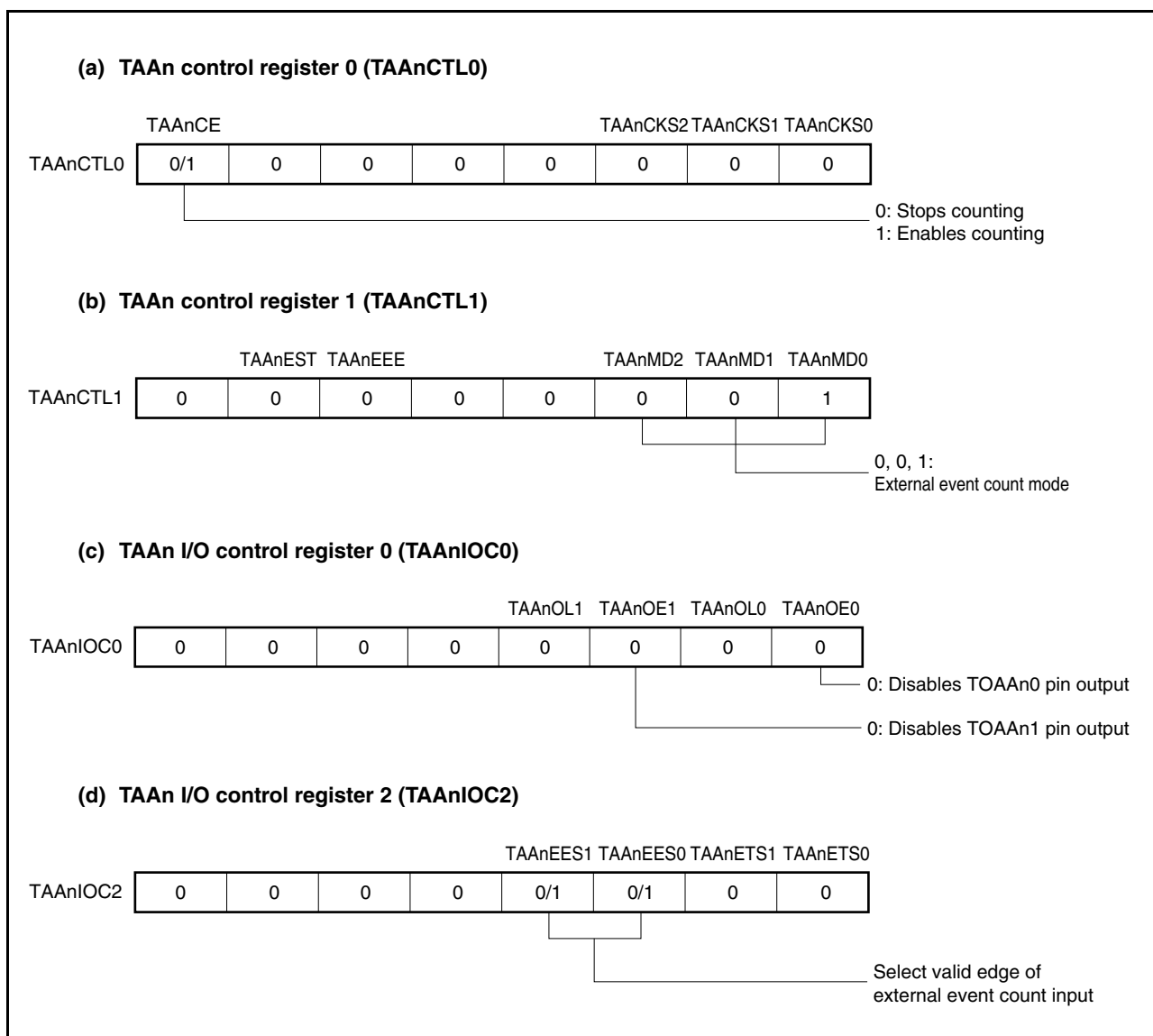
**Figure 7-16.  Register Setting for Operation in External Event Count Mode (2/2)**

**(e) TAAn counter read buffer register (TAAnCNT)**

The count value of the 16-bit counter can be read by reading the TAAnCNT register.

**(f) TAAn capture/compare register 0 (TAAnCCR0)**

If $D_0$ is set to the TAAnCCR0 register, the counter is cleared and a compare match interrupt request signal (INTTAAnCC0) is generated when the number of external event counts reaches ($D_0 + 1$).

**(g) TAAn capture/compare register 1 (TAAnCCR1)**

Usually, the TAAnCCR1 register is not used in the external event count mode. However, the set value of the TAAnCCR1 register is transferred to the CCR1 buffer register. When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTAAnCC1) is generated.

Therefore, mask the interrupt signal by using the interrupt mask flag (TAAnCCMK1).

**Caution   When an external clock is used as the count clock, the external clock can be input only from the TIAAn0 pin.  At this time, set the TAAnIOC1.TAAnIS1 and TAAnIOC1.TAAnIS0 bits to 00 (capture trigger input (TIAAn0 pin): no edge detection).**

**Remarks  1.** TAAn I/O control register 1 (TAAnIOC1) and TAAn option register 0 (TAAnOPT0) are not used in the external event count mode.

**2.** n = 0 to 3, 5

**(1) External event count mode operation flow**

**Figure 7-17.  Flow of Software Processing in External Event Count Mode**
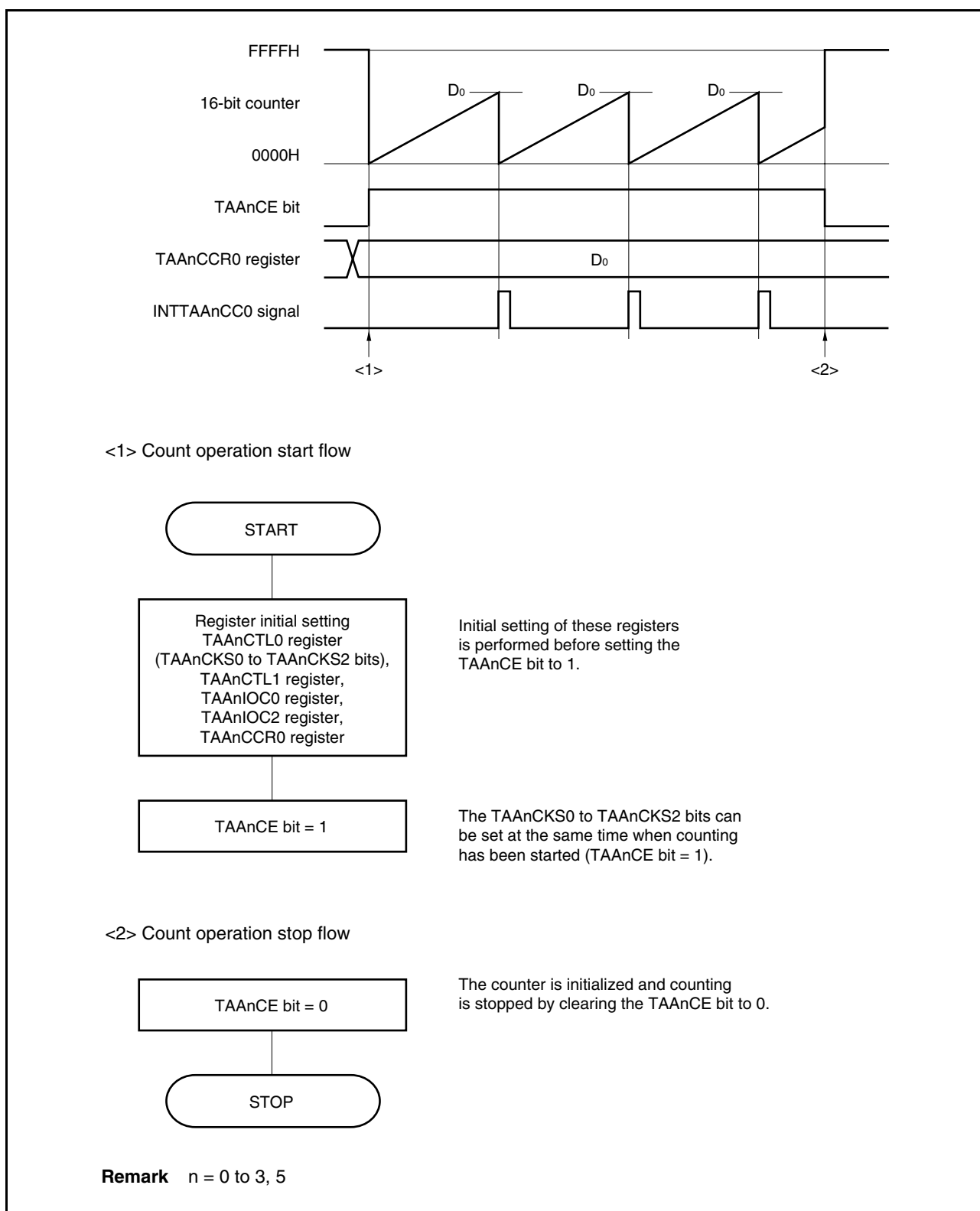


<1> Count operation start flow
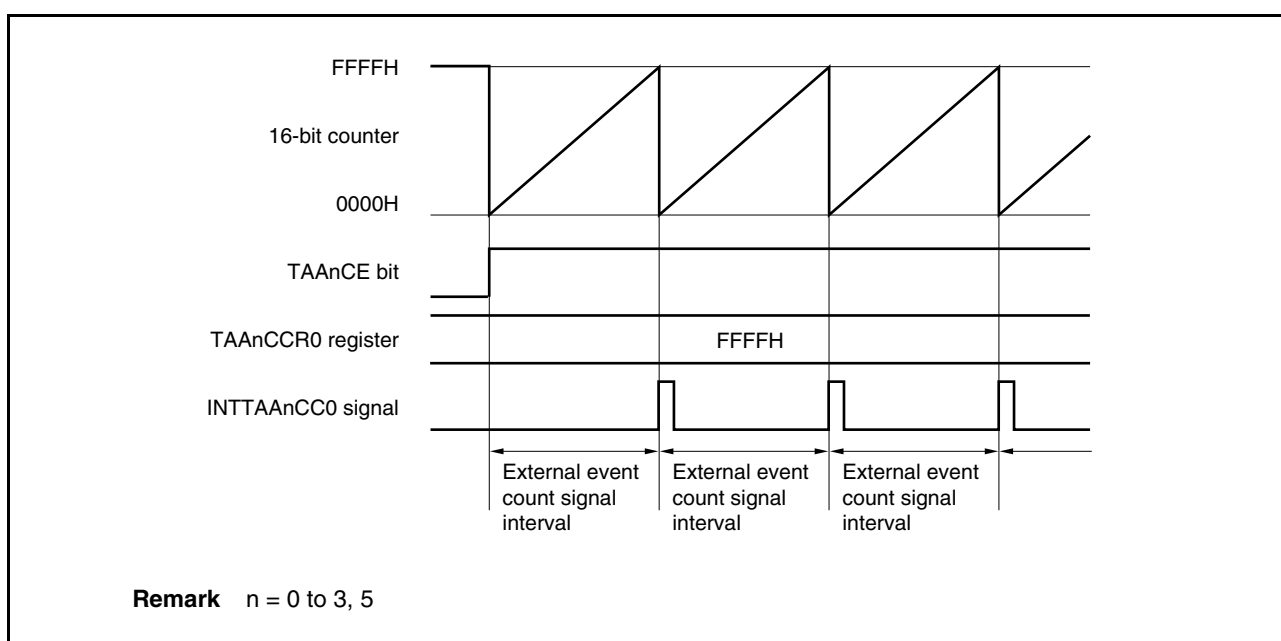
<2> Count operation stop flow

**Remark**  n = 0 to 3, 5

**(2)  Operation timing in external event count mode**

**Cautions 1.  In the external event count mode, do not set the TAAnCCR0 register to 0000H.**

**2.  In the external event count mode, use of the timer output is disabled.  If performing timer output using external event count input, set the interval timer mode, and select the operation of the count clock to be enabled by the external event count input (TAAnCTL1.TAAnMD2 to TAAnCTL1.TAAnMD0 bits = 000, TAAnCTL1.TAAnEEE bit = 1).**

**(a)  Operation if TAAnCCR0 register is set to FFFFH**

If the TAAnCCR0 register is set to FFFFH, the 16-bit counter counts to FFFFH each time the valid edge of the external event count signal has been detected.  The 16-bit counter is cleared to 0000H in synchronization with the next count-up timing, and the INTTAAnCC0 signal is generated.  At this time, the TAAnOPT0.TAAnOVF bit is not set.



**Remark**   n = 0 to 3, 5

**(b) Notes on rewriting the TAAnCCR0 register**

To change the value of the TAAnCCR0 register to a smaller value, stop counting once and then change the set value.
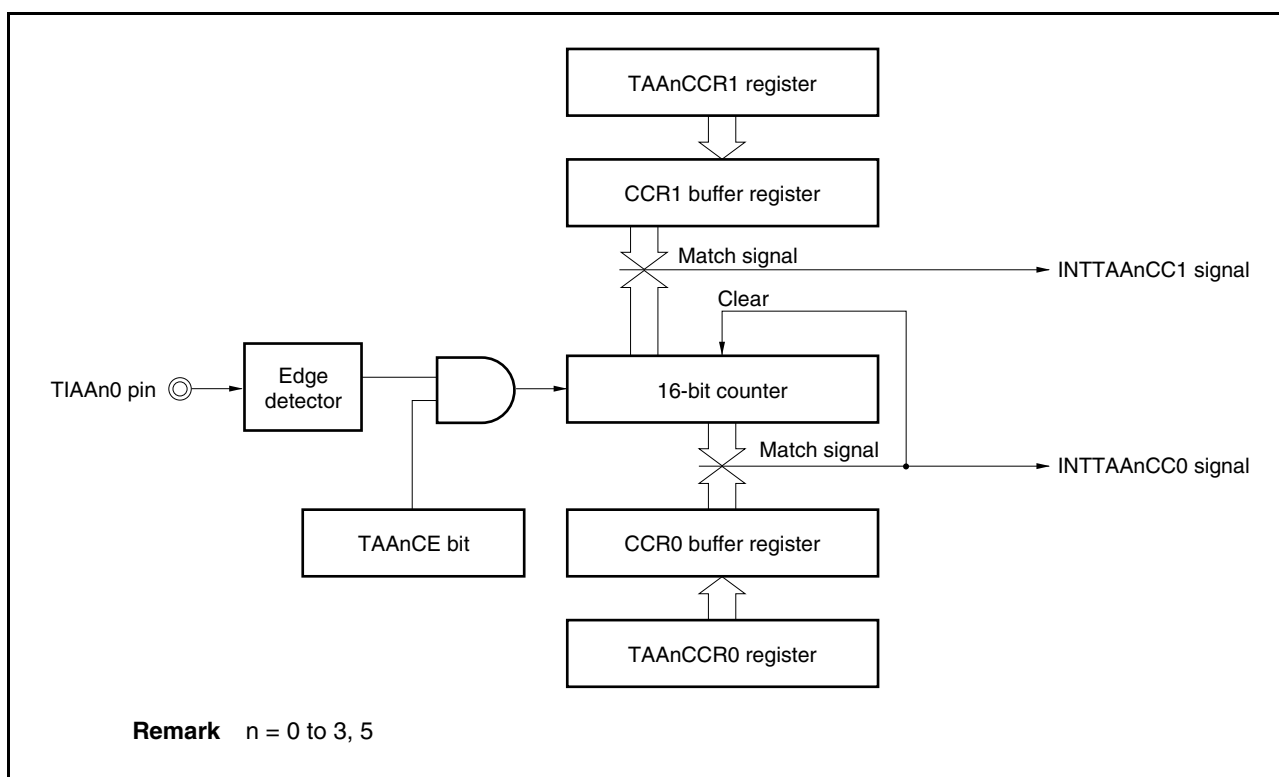
If the value of the TAAnCCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow.



**Remark**   n = 0 to 3, 5

If the value of the TAAnCCR0 register is changed from $D_1$ to $D_2$ while the count value is greater than $D_2$ but less than $D_1$, the count value is transferred to the CCR0 buffer register as soon as the TAAnCCR0 register has been rewritten. Consequently, the value that is compared with the 16-bit counter is $D_2$.

Because the count value has already exceeded $D_2$, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H. When the count value matches $D_2$, the INTTAAnCC0 signal is generated.

Therefore, the INTTAAnCC0 signal may not be generated at the valid edge count of "($D_1$ + 1) times" or "($D_2$ + 1) times" as originally expected, but may be generated at the valid edge count of "(10000H + $D_2$ + 1) times".

**(c) Operation of TAAnCCR1 register**

**Figure 7-18.  Configuration of TAAnCCR1 Register**



**Remark**   n = 0 to 3, 5

If the set value of the TAAnCCR1 register is smaller than the set value of the TAAnCCR0 register, the INTTAAnCC1 signal is generated once per cycle.

**Figure 7-19.  Timing Chart When $D_{01} \geq D_{11}$**
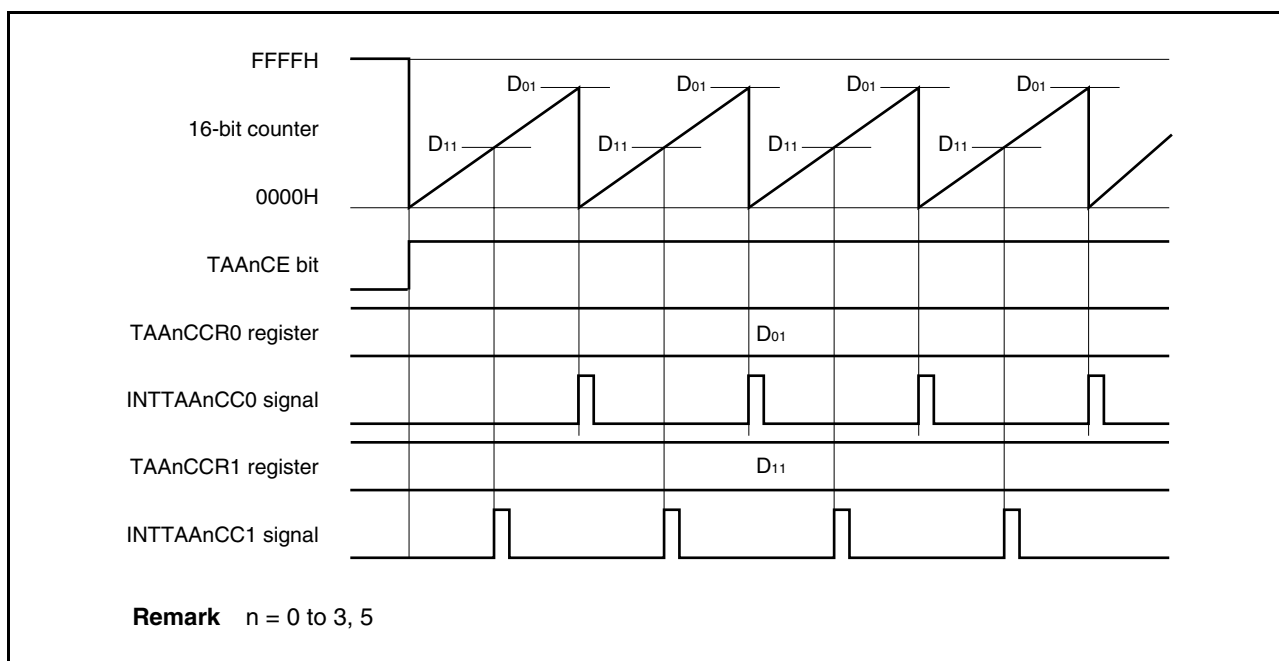


**Remark**   n = 0 to 3, 5

If the set value of the TAAnCCR1 register is greater than the set value of the TAAnCCR0 register, the INTTAAnCC1 signal is not generated because the count value of the 16-bit counter and the value of the TAAnCCR1 register do not match.

**Figure 7-20.  Timing Chart When $D_{01} < D_{11}$**



**Remark**   n = 0 to 3, 5
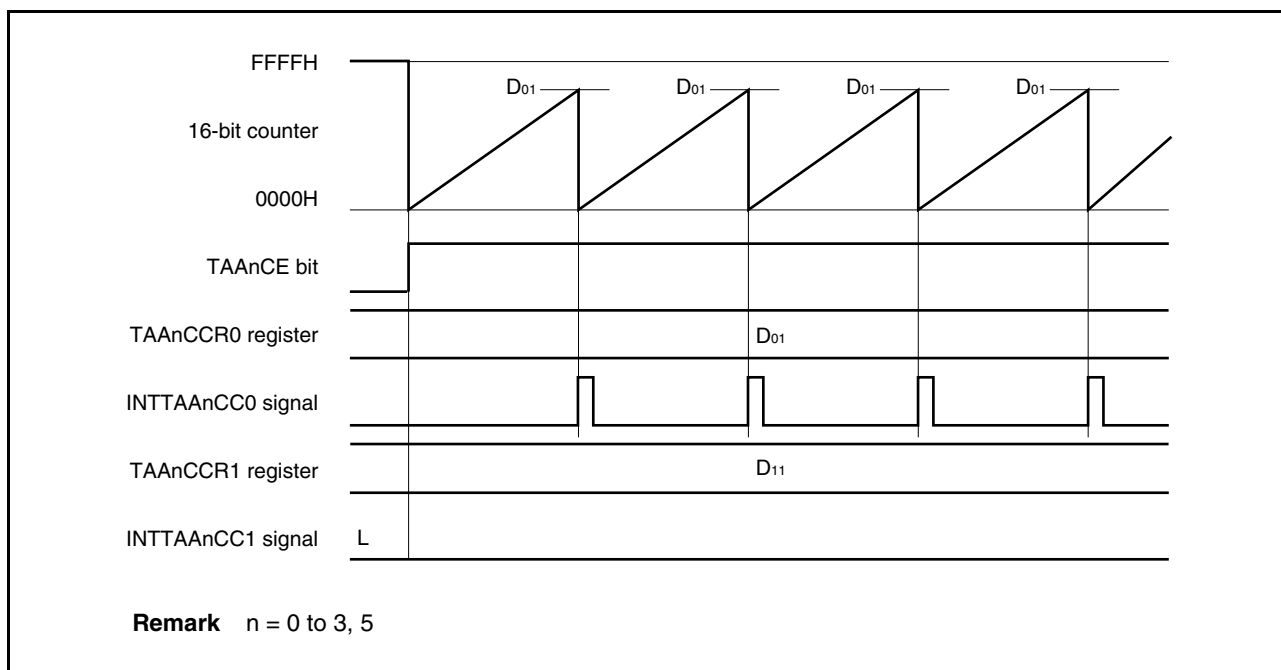
### 7.5.3    External trigger pulse output mode (TAAnMD2 to TAAnMD0 bits = 010)

In the external trigger pulse output mode, 16-bit timer/event counter AA waits for a trigger when the TAAnCTL0.TAAnCE bit is set to 1.  When the valid edge of an external trigger input signal is detected, 16-bit timer/event counter AA starts counting, and outputs a PWM waveform from the TOAAn1 pin.

Pulses can also be output by generating a software trigger instead of using the external trigger.  When using a software trigger, a square wave that has one cycle of the PWM waveform as half its cycle can also be output from the TOAAn0 pin.

**Figure 7-21.  Configuration in External Trigger Pulse Output Mode**
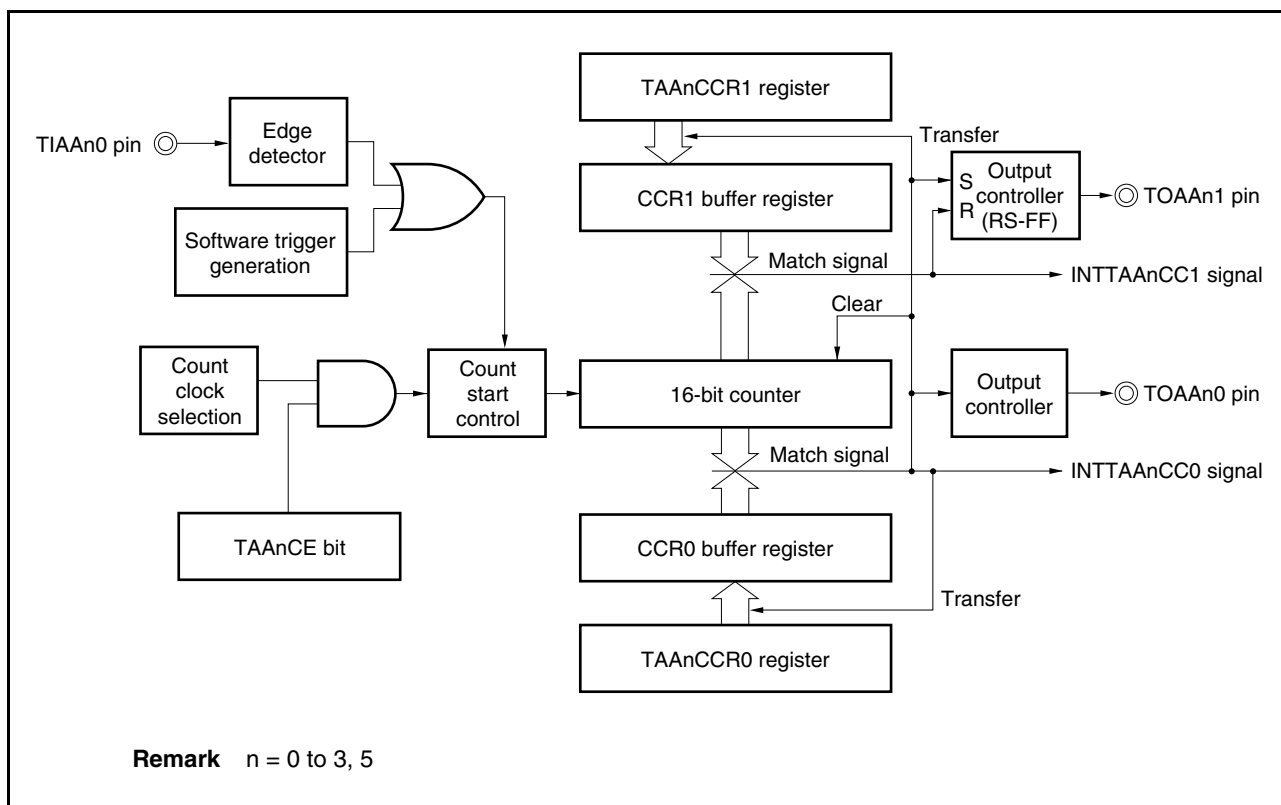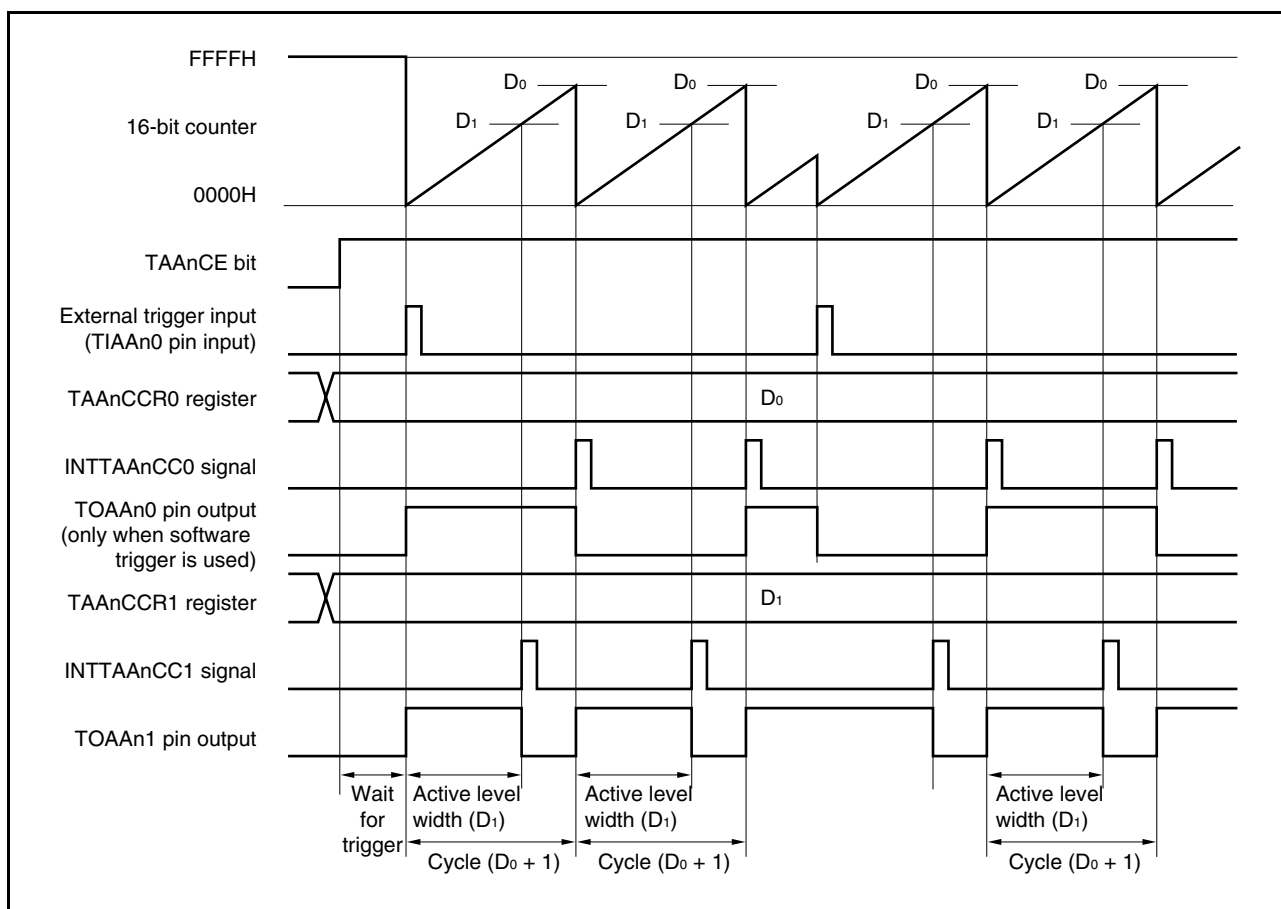


**Remark**    n = 0 to 3, 5

**Figure 7-22.  Basic Timing in External Trigger Pulse Output Mode**



16-bit timer/event counter AA waits for a trigger when the TAAnCE bit is set to 1.  When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting at the same time, and outputs a PWM waveform from the TOAAn1 pin.  If the trigger is generated again while the counter is operating, the counter is cleared to 0000H and restarted. (The output of the TOAAn0 pin is inverted.  The TOAAn1 pin outputs a high level regardless of the status (high/low) when a trigger occurs.)

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

Active level width = (Set value of TAAnCCR1 register) $\times$ Count clock cycle

Cycle = (Set value of TAAnCCR0 register + 1) $\times$ Count clock cycle

Duty factor = (Set value of TAAnCCR1 register)/(Set value of TAAnCCR0 register + 1)

The compare match request signal INTTAAnCC0 is generated the next time the 16-bit counter counts after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H at the same time.  The compare match interrupt request signal INTTAAnCC1 is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The value set to the TAAnCCRm register is transferred to the CCRm buffer register when the count value of the 16-bit counter matches the value of the CCRm buffer register and the 16-bit counter is cleared to 0000H.

The valid edge of an external trigger input signal, or setting the software trigger (TAAnCTL1.TAAnEST bit) to 1 is used as the trigger.

**Remark**   n = 0 to 3, 5
          m = 0, 1

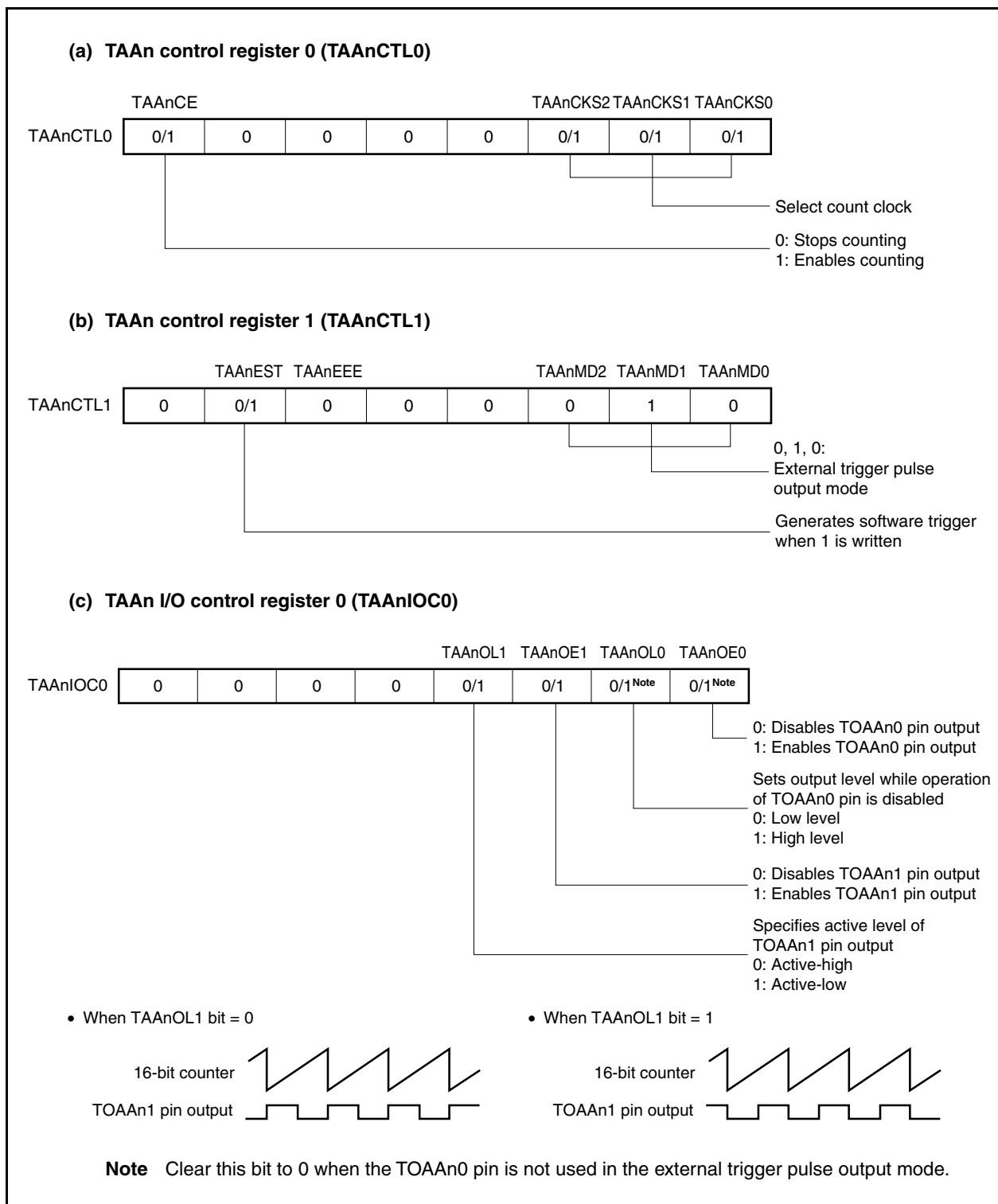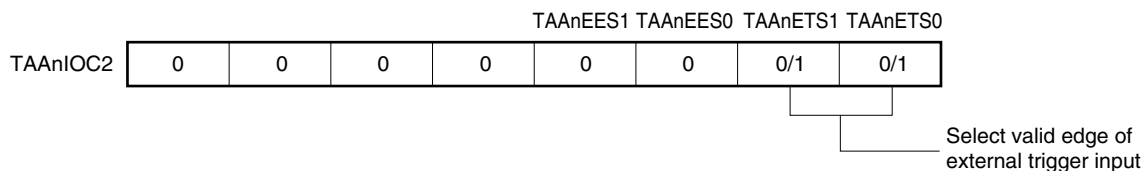**Figure 7-23.  Setting of Registers in External Trigger Pulse Output Mode (1/2)**



**(a)  TAAn control register 0 (TAAnCTL0)**

| | TAAnCE | | | | | TAAnCKS2 | TAAnCKS1 | TAAnCKS0 |
|---|---|---|---|---|---|---|---|---|
| TAAnCTL0 | 0/1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 |

Select count clock

0: Stops counting
1: Enables counting

**(b)  TAAn control register 1 (TAAnCTL1)**

| | TAAnEST | TAAnEEE | | | | TAAnMD2 | TAAnMD1 | TAAnMD0 |
|---|---|---|---|---|---|---|---|---|
| TAAnCTL1 | 0 | 0/1 | 0 | 0 | 0 | 0 | 1 | 0 |

0, 1, 0:
External trigger pulse
output mode

Generates software trigger
when 1 is written

**(c)  TAAn I/O control register 0 (TAAnIOC0)**

| | | | | | TAAnOL1 | TAAnOE1 | TAAnOL0 | TAAnOE0 |
|---|---|---|---|---|---|---|---|---|
| TAAnIOC0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 [Note] | 0/1 [Note] |

0: Disables TOAAn0 pin output
1: Enables TOAAn0 pin output

Sets output level while operation
of TOAAn0 pin is disabled
0: Low level
1: High level

0: Disables TOAAn1 pin output
1: Enables TOAAn1 pin output

Specifies active level of
TOAAn1 pin output
0: Active-high
1: Active-low

- When TAAnOL1 bit = 0

16-bit counter

TOAAn1 pin output

- When TAAnOL1 bit = 1

16-bit counter

TOAAn1 pin output

**Note**   Clear this bit to 0 when the TOAAn0 pin is not used in the external trigger pulse output mode.

**Figure 7-23. Setting of Registers in External Trigger Pulse Output Mode (2/2)**

**(d) TAAn I/O control register 2 (TAAnIOC2)**

| | | | | | TAAnEES1 | TAAnEES0 | TAAnETS1 | TAAnETS0 |
|---|---|---|---|---|---|---|---|---|
| TAAnIOC2 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |

Select valid edge of external trigger input

**(e) TAAn counter read buffer register (TAAnCNT)**

The value of the 16-bit counter can be read by reading the TAAnCNT register.

**(f) TAAn capture/compare registers 0 and 1 (TAAnCCR0 and TAAnCCR1)**

If $D_0$ is set to the TAAnCCR0 register and $D_1$ to the TAAnCCR1 register, the cycle and active level of the PWM waveform are as follows.

Cycle = $(D_0 + 1) \times$ Count clock cycle

Active level width = $D_1 \times$ Count clock cycle

**Remarks 1.** TAAn I/O control register 1 (TAAnIOC1) and TAAn option register 0 (TAAnOPT0) are not used in the external trigger pulse output mode.

**2.** n = 0 to 3, 5

**(1) Operation flow in external trigger pulse output mode**

**Figure 7-24. Software Processing Flow in External Trigger Pulse Output Mode (1/2)**
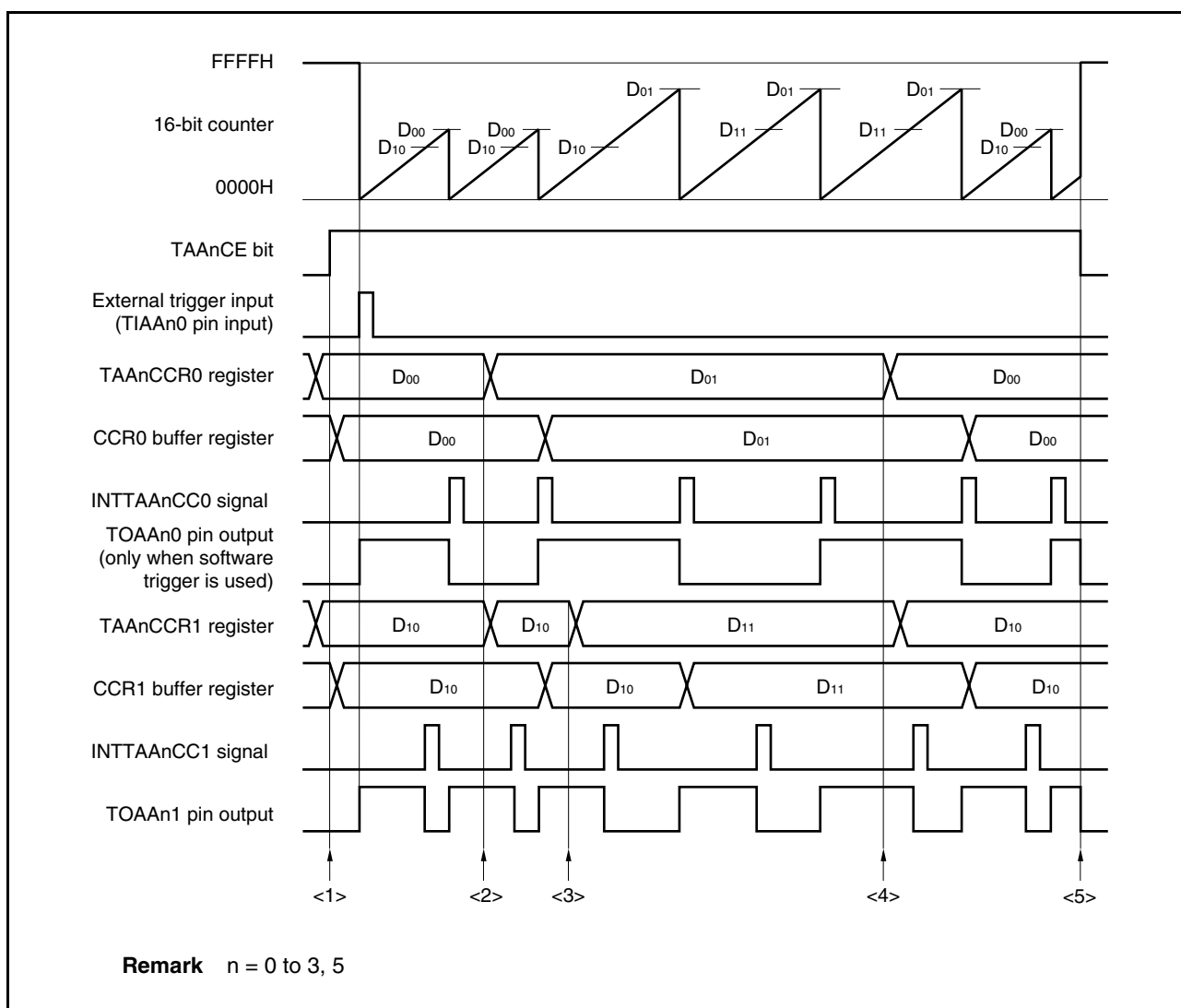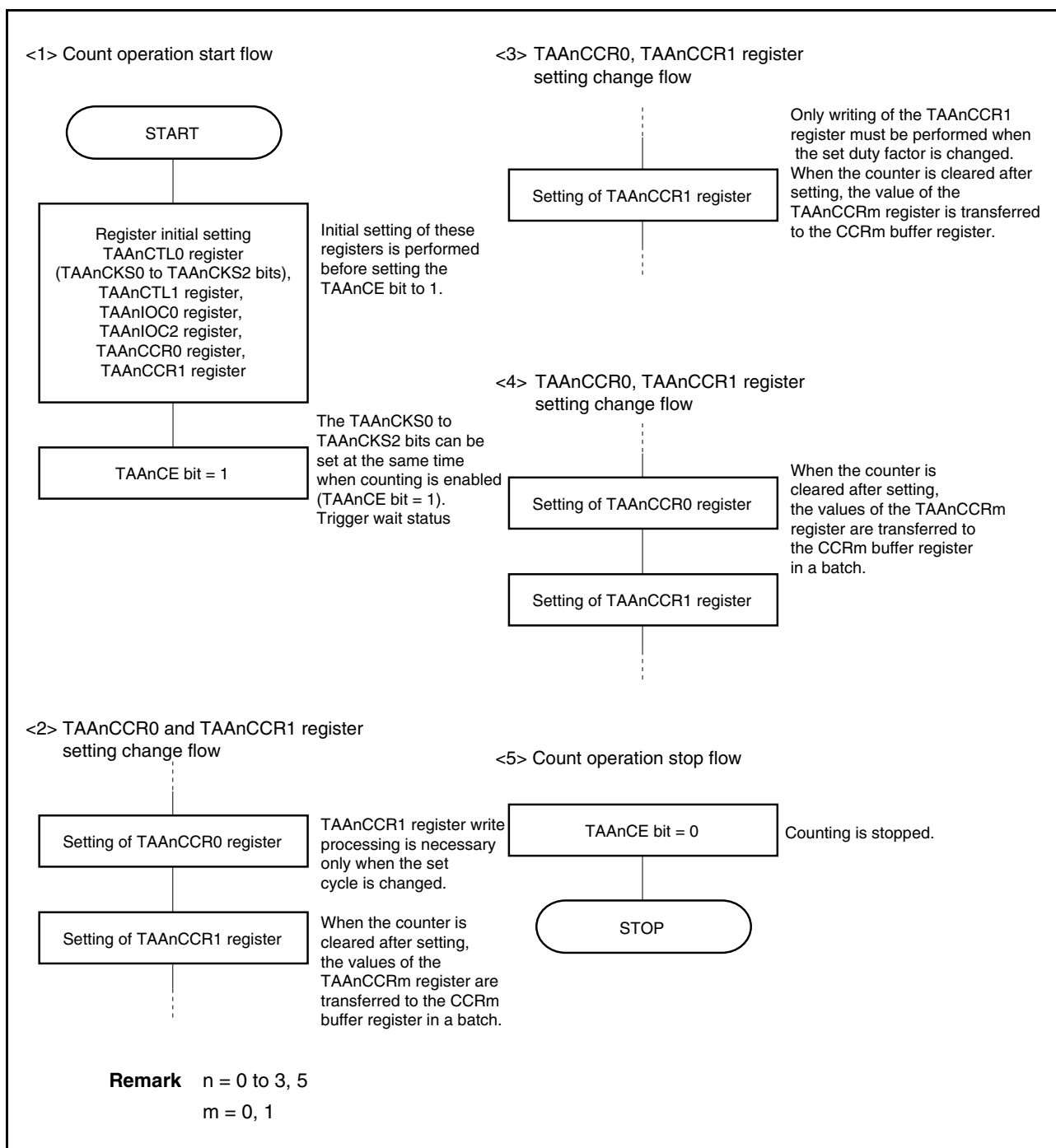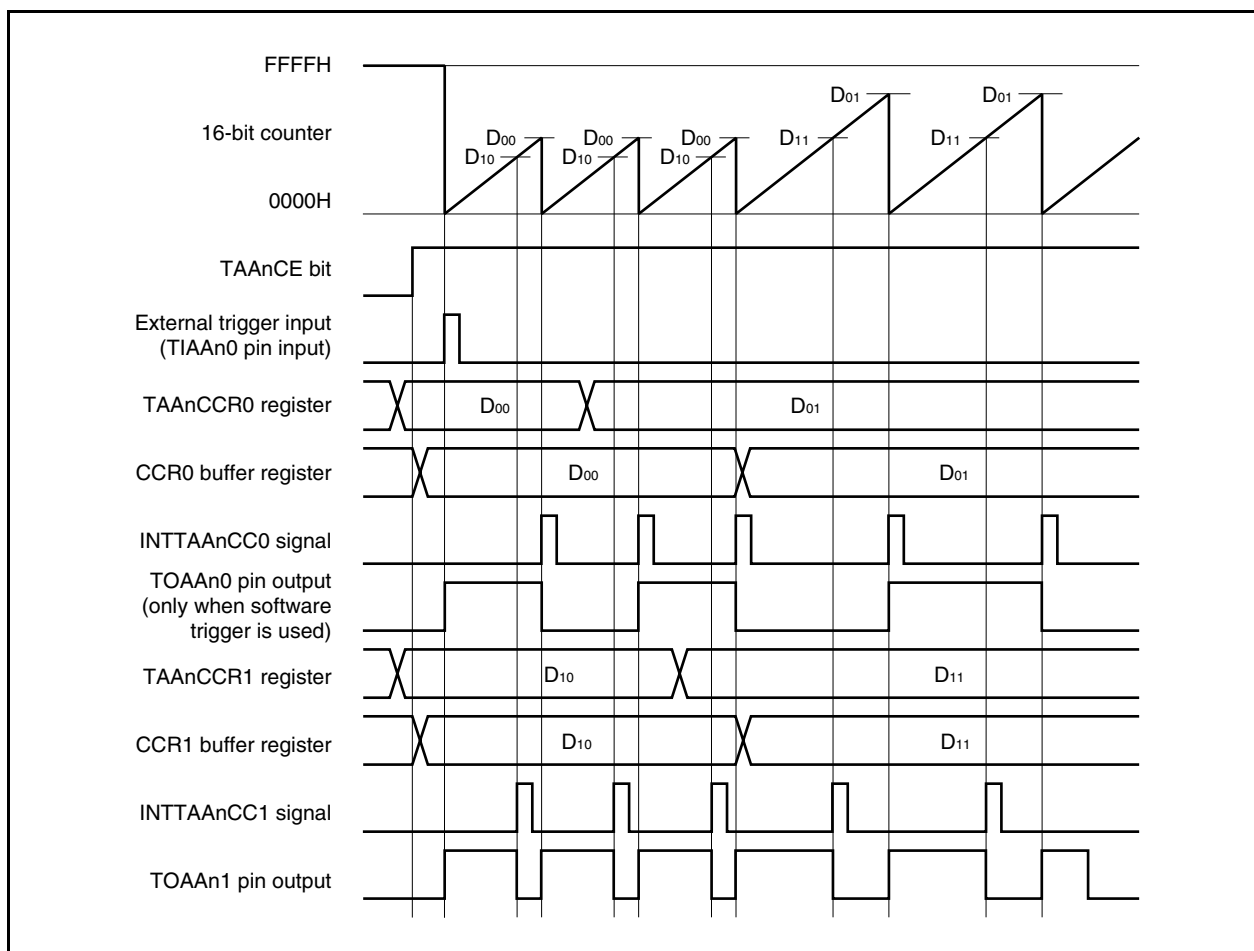


**Remark** n = 0 to 3, 5

**Figure 7-24.  Software Processing Flow in External Trigger Pulse Output Mode (2/2)**

<1> Count operation start flow

START

Register initial setting
TAAnCTL0 register
(TAAnCKS0 to TAAnCKS2 bits),
TAAnCTL1 register,
TAAnIOC0 register,
TAAnIOC2 register,
TAAnCCR0 register,
TAAnCCR1 register

Initial setting of these registers is performed before setting the TAAnCE bit to 1.

TAAnCE bit = 1

The TAAnCKS0 to TAAnCKS2 bits can be set at the same time when counting is enabled (TAAnCE bit = 1). Trigger wait status

<3> TAAnCCR0, TAAnCCR1 register
    setting change flow

Setting of TAAnCCR1 register

Only writing of the TAAnCCR1 register must be performed when the set duty factor is changed. When the counter is cleared after setting, the value of the TAAnCCRm register is transferred to the CCRm buffer register.

<4> TAAnCCR0, TAAnCCR1 register
    setting change flow

Setting of TAAnCCR0 register

Setting of TAAnCCR1 register

When the counter is cleared after setting, the values of the TAAnCCRm register are transferred to the CCRm buffer register in a batch.

<2> TAAnCCR0 and TAAnCCR1 register
    setting change flow

Setting of TAAnCCR0 register

Setting of TAAnCCR1 register

TAAnCCR1 register write processing is necessary only when the set cycle is changed.

When the counter is cleared after setting, the values of the TAAnCCRm register are transferred to the CCRm buffer register in a batch.

<5> Count operation stop flow

TAAnCE bit = 0

Counting is stopped.

STOP

**Remark**   n = 0 to 3, 5
            m = 0, 1

**(2) External trigger pulse output mode operation timing**

**(a) Note on changing pulse width during operation**

To change the PWM waveform while the counter is operating, write the TAAnCCR1 register last.

Rewrite the TAAnCCRm register after writing the TAAnCCR1 register after the INTTAAnCC0 signal is detected.

In order to transfer data from the TAAnCCRm register to the CCRm buffer register, the TAAnCCR1 register must be written.

To change both the cycle and active level width of the PWM waveform at this time, first set the cycle to the TAAnCCR0 register and then set the active level width to the TAAnCCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TAAnCCR0 register, and then write the same value to the TAAnCCR1 register.

To change only the active level width (duty factor) of the PWM waveform, only the TAAnCCR1 register has to be set.

After data is written to the TAAnCCR1 register, the value written to the TAAnCCRm register is transferred to the CCRm buffer register in synchronization with clearing of the 16-bit counter, and is used as the value compared with the 16-bit counter.

To write the TAAnCCR0 or TAAnCCR1 register again after writing the TAAnCCR1 register once, do so after the INTTAAnCC0 signal is generated.  Otherwise, the value of the CCRm buffer register may become undefined because the timing of transferring data from the TAAnCCRm register to the CCRm buffer register conflicts with writing the TAAnCCRm register.
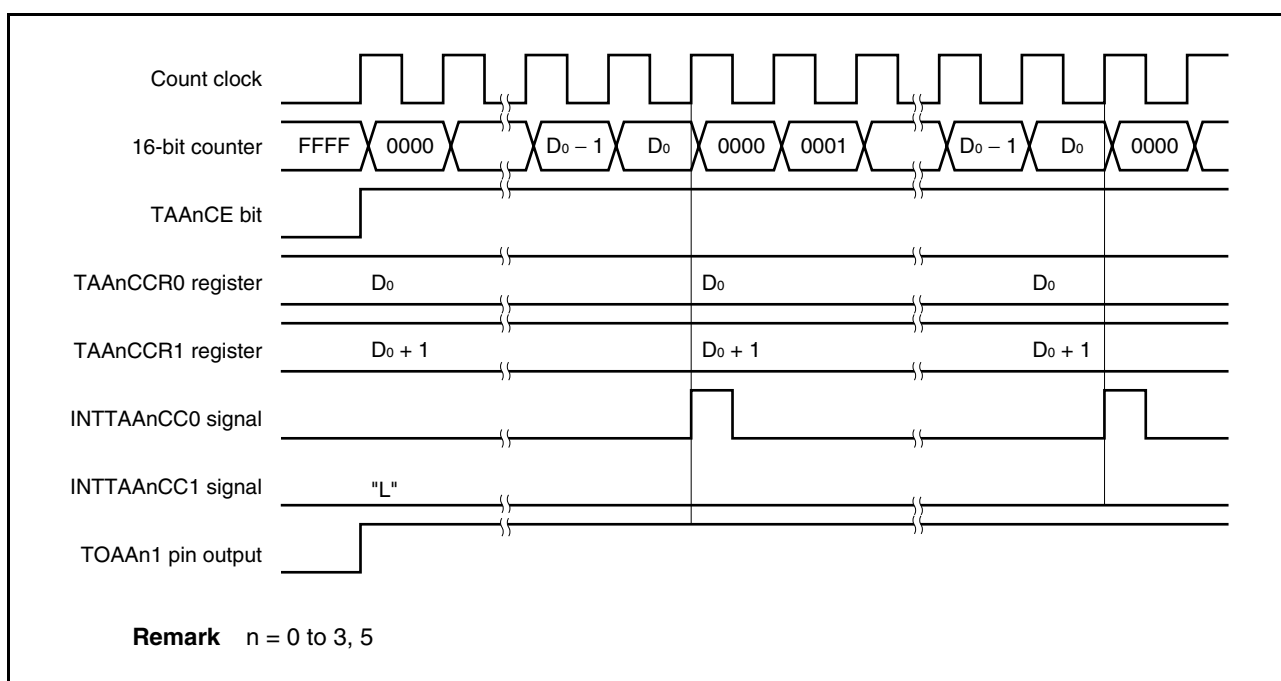
**Remark**   n = 0 to 3, 5
          m = 0, 1

**(b) 0%/100% output of PWM waveform**

To output a 0% waveform, set the TAAnCCR1 register to 0000H. If the set value of the TAAnCCR0 register is FFFFH, the INTTAAnCC1 signal is generated periodically.

Count clock

16-bit counter   FFFF   0000   $D_0 - 1$   $D_0$   0000   0001   $D_0 - 1$   $D_0$   0000

TAAnCE bit

TAAnCCR0 register   $D_0$    $D_0$    $D_0$

TAAnCCR1 register   0000H    0000H    0000H

INTTAAnCC0 signal

INTTAAnCC1 signal

TOAAn1 pin output   "L"

**Remark**   n = 0 to 3, 5

To output a 100% waveform, set a value of (set value of TAAnCCR0 register + 1) to the TAAnCCR1 register. If the set value of the TAAnCCR0 register is FFFFH, 100% output cannot be produced.

Count clock

16-bit counter   FFFF   0000   $D_0 - 1$   $D_0$   0000   0001   $D_0 - 1$   $D_0$   0000

TAAnCE bit

TAAnCCR0 register   $D_0$    $D_0$    $D_0$

TAAnCCR1 register   $D_0 + 1$    $D_0 + 1$    $D_0 + 1$

INTTAAnCC0 signal

INTTAAnCC1 signal   "L"

TOAAn1 pin output

**Remark**   n = 0 to 3, 5

**(c) Conflict between trigger detection and match with TAAnCCR1 register**

If the trigger is detected immediately after the INTTAAnCC1 signal is generated, the 16-bit counter is cleared to 0000H at the same time, the output signal of the TOAAn1 pin is asserted, and the counter continues counting. Consequently, the inactive period of the PWM waveform is shortened.
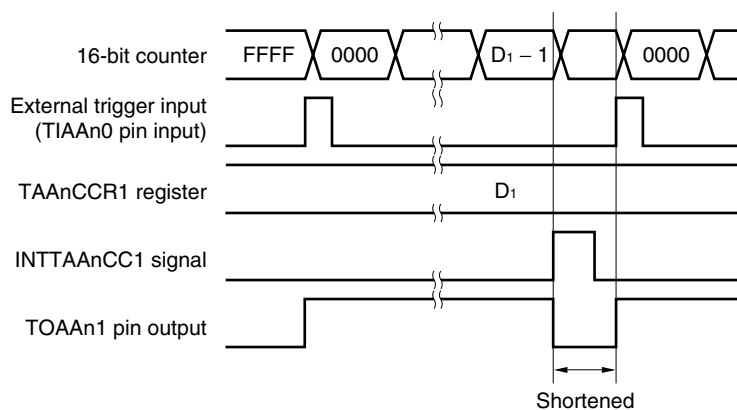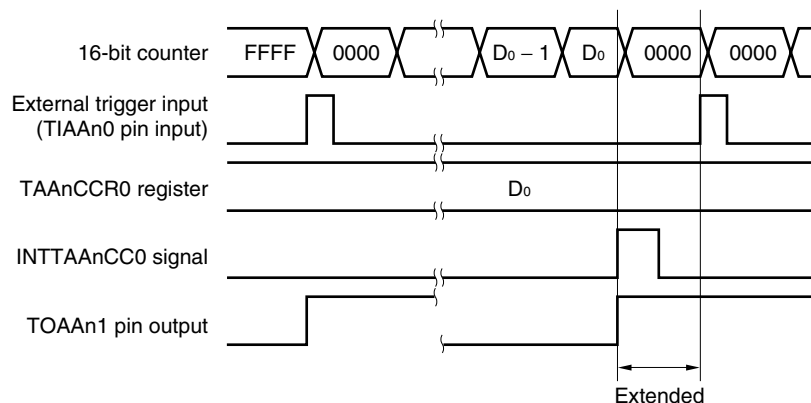


**Remark**  n = 0 to 3, 5

If the trigger is detected immediately before the INTTAAnCC1 signal is generated, the INTTAAnCC1 signal is not generated, and the 16-bit counter is cleared to 0000H and continues counting. The output signal of the TOAAn1 pin remains active. Consequently, the active period of the PWM waveform is extended.
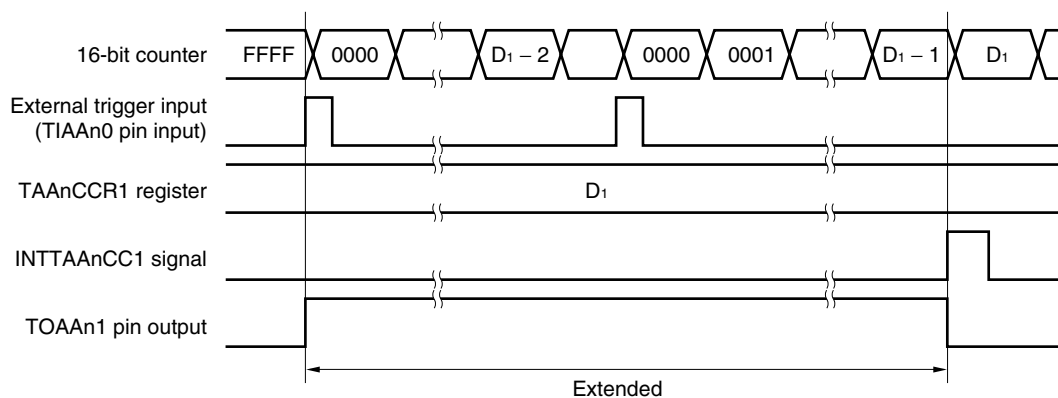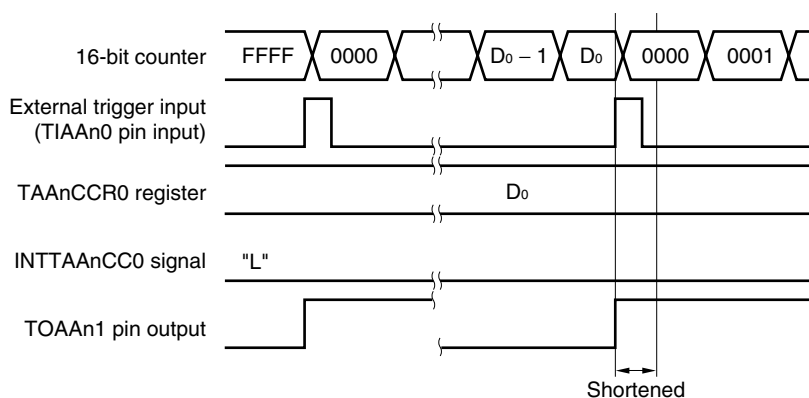


**Remark**  n = 0 to 3, 5

**(d) Conflict between trigger detection and match with TAAnCCR0 register**

If the trigger is detected immediately after the INTTAAnCC0 signal is generated, the 16-bit counter is cleared to 0000H again and continues counting up.  Therefore, the active period of the TOAAn1 pin is extended by the time from generation of the INTTAAnCC0 signal to trigger detection.
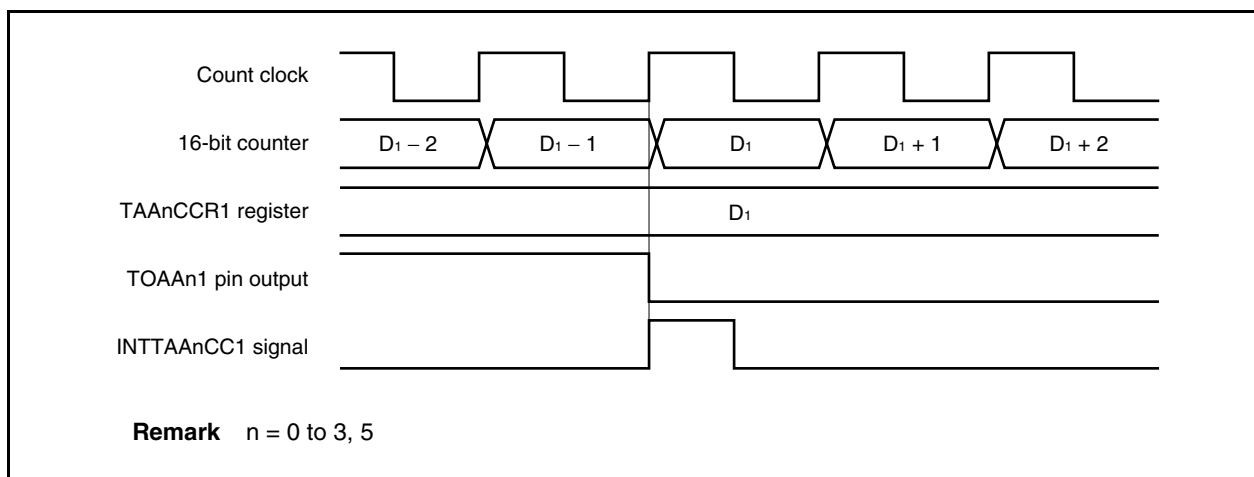


**Remark**   n = 0 to 3, 5

If the trigger is detected immediately before the INTTAAnCC0 signal is generated, the INTTAAnCC0 signal is not generated.  The 16-bit counter is cleared to 0000H, the TOAAn1 pin is asserted, and the counter continues counting.  Consequently, the inactive period of the PWM waveform is shortened.



**Remark**   n = 0 to 3, 5

**(e) Generation timing of compare match interrupt request signal (INTTAAnCC1)**

The timing of generation of the INTTAAnCC1 signal in the external trigger pulse output mode differs from the timing of other INTTAAnCC1 signals; the INTTAAnCC1 signal in the external trigger pulse output mode is generated when the count value of the 16-bit counter matches the value of the TAAnCCR1 register.



Count clock

16-bit counter    $D_1 - 2$    $D_1 - 1$    $D_1$    $D_1 + 1$    $D_1 + 2$

TAAnCCR1 register                              $D_1$

TOAAn1 pin output

INTTAAnCC1 signal

**Remark**    n = 0 to 3, 5

Usually, the INTTAAnCC1 signal is generated in synchronization with the next count-up, after the count value of the 16-bit counter matches the value of the TAAnCCR1 register.

In the external trigger pulse output mode, however, it is generated one clock earlier. This is because the timing is changed to match the timing of changing the output signal of the TOAAn1 pin.

### 7.5.4   One-shot pulse output mode (TAAnMD2 to TAAnMD0 bits = 011)

In the one-shot pulse output mode, 16-bit timer/event counter AA waits for a trigger when the TAAnCTL0.TAAnCE bit is set to 1. When the valid edge of an external trigger input is detected, 16-bit timer/event counter AA starts counting, and outputs a one-shot pulse from the TOAAn1 pin.

Instead of the external trigger, a software trigger can also be generated to output the pulse. When the software trigger is used, the TOAAn0 pin outputs the active level while the 16-bit counter is counting, and the inactive level when the counter is stopped (waiting for a trigger).

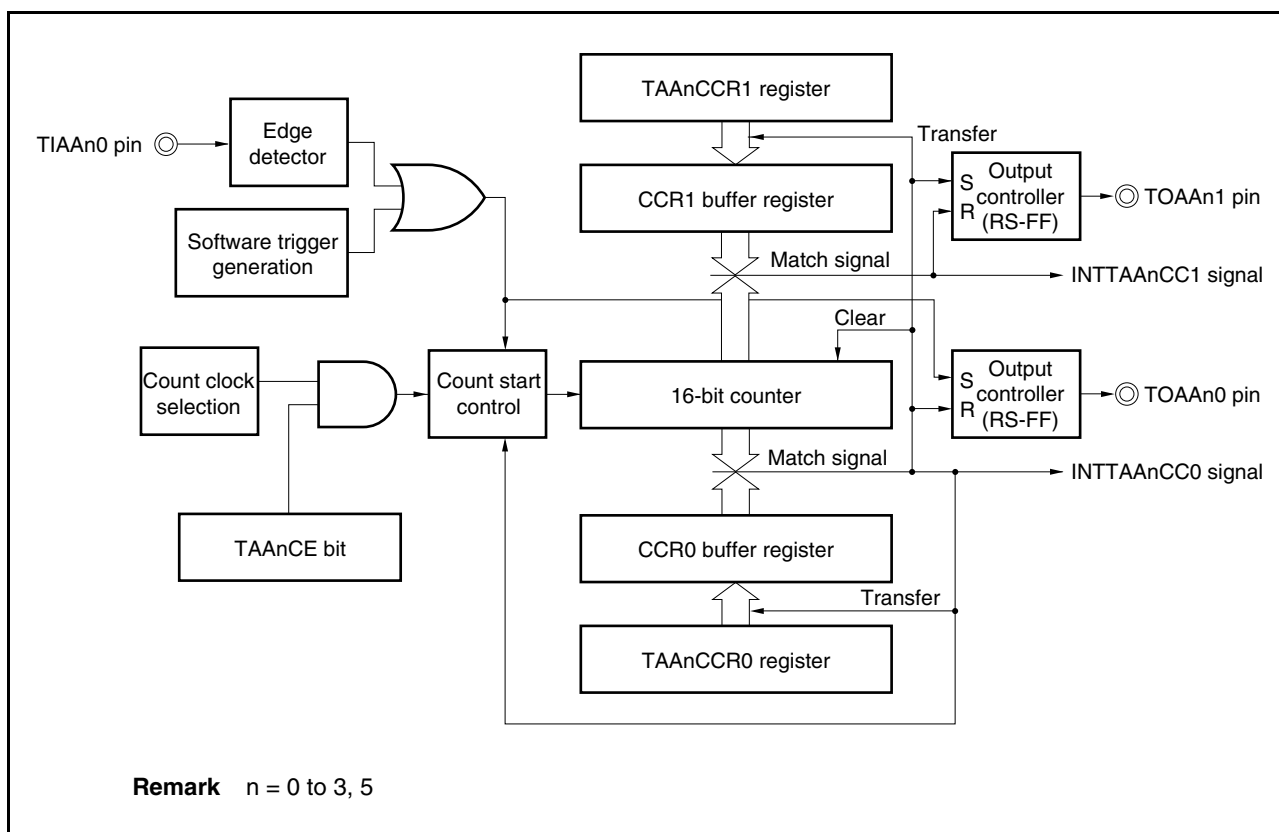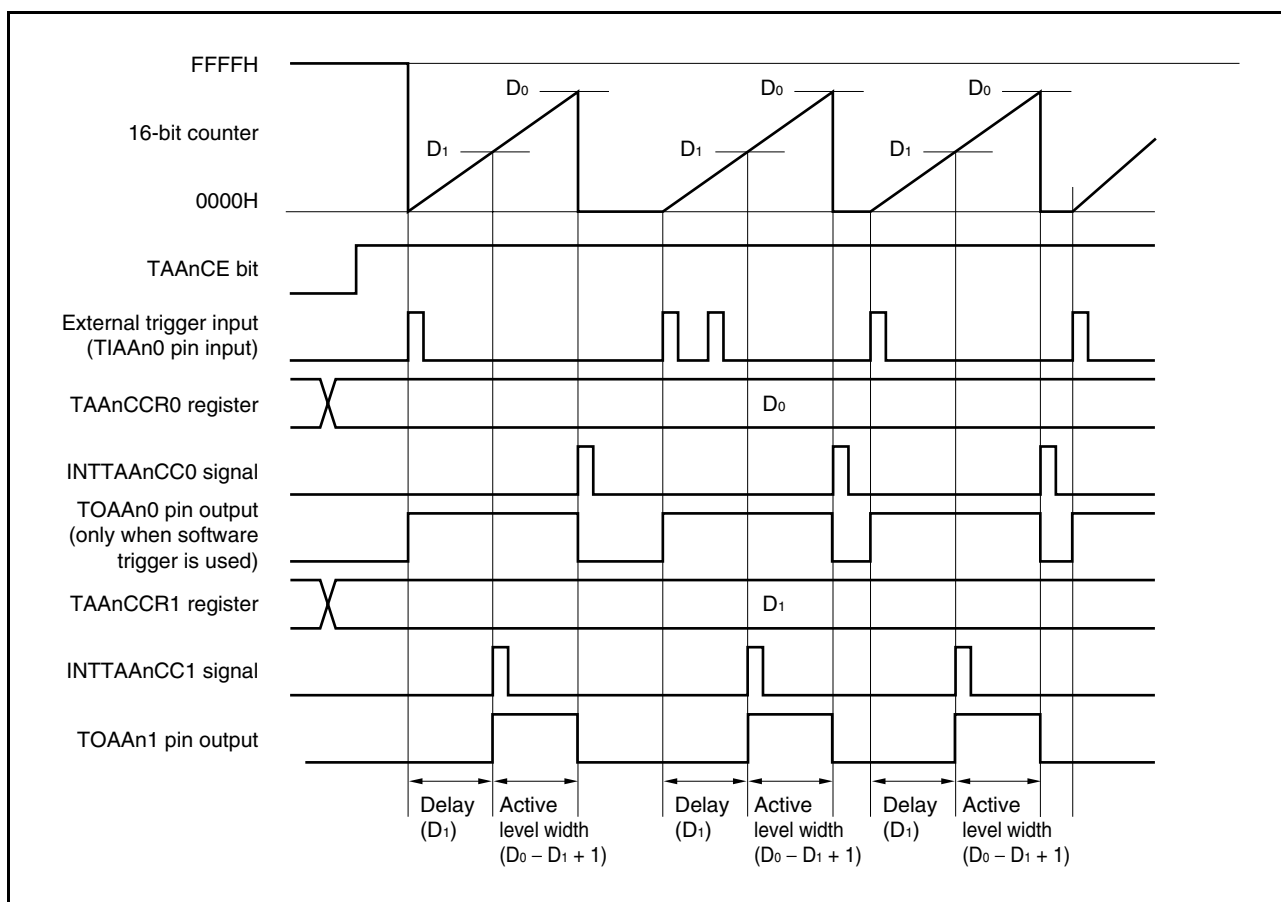**Figure 7-25.  Configuration in One-Shot Pulse Output Mode**



**Remark**   n = 0 to 3, 5

**Figure 7-26. Basic Timing in One-Shot Pulse Output Mode**



When the TAAnCE bit is set to 1, 16-bit timer/event counter AA waits for a trigger.  When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a one-shot pulse from the TOAAn1 pin.  After the one-shot pulse is output, the 16-bit counter is set to FFFFH, stops counting, and waits for a trigger.  If a trigger is generated again while the one-shot pulse is being output, it is ignored.

The output delay period and active level width of the one-shot pulse can be calculated as follows.

Output delay period = (Set value of TAAnCCR1 register) $\times$ Count clock cycle
Active level width = (Set value of TAAnCCR0 register $-$ Set value of TAAnCCR1 register $+$ 1) $\times$ Count clock cycle

The compare match interrupt request signal INTTAAnCC0 is generated when the 16-bit counter counts after its count value matches the value of the CCR0 buffer register.  The compare match interrupt request signal INTTAAnCC1 is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.
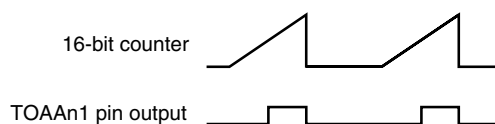
The valid edge of an external trigger input or setting the software trigger (TAAnCTL1.TAAnEST bit) to 1 is used as the trigger.
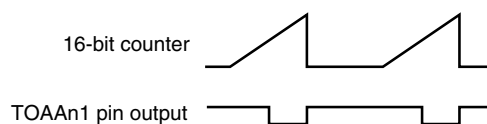
**Remark**  n = 0 to 3, 5

**Figure 7-27. Register Setting for Operation in One-Shot Pulse Output Mode (1/2)**
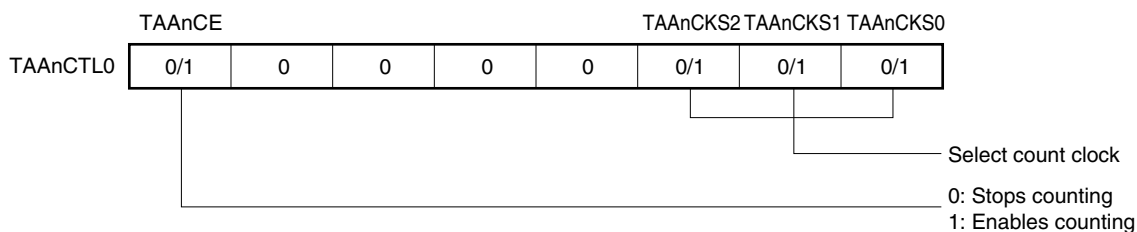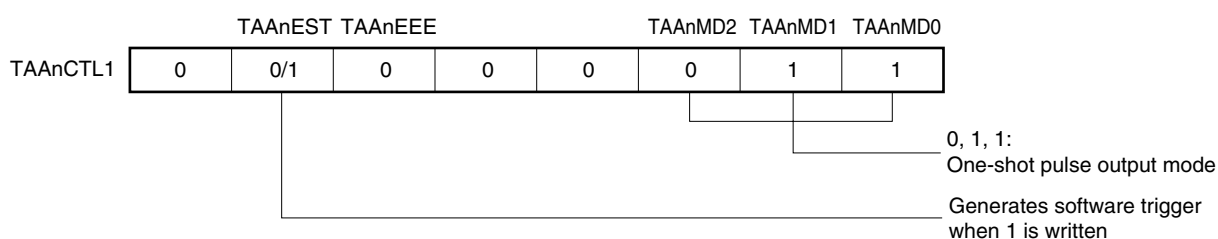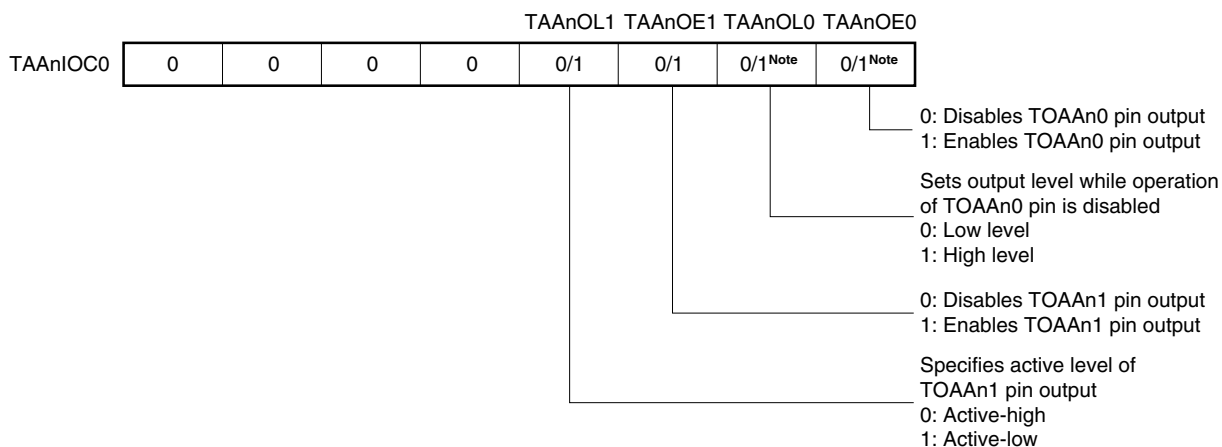


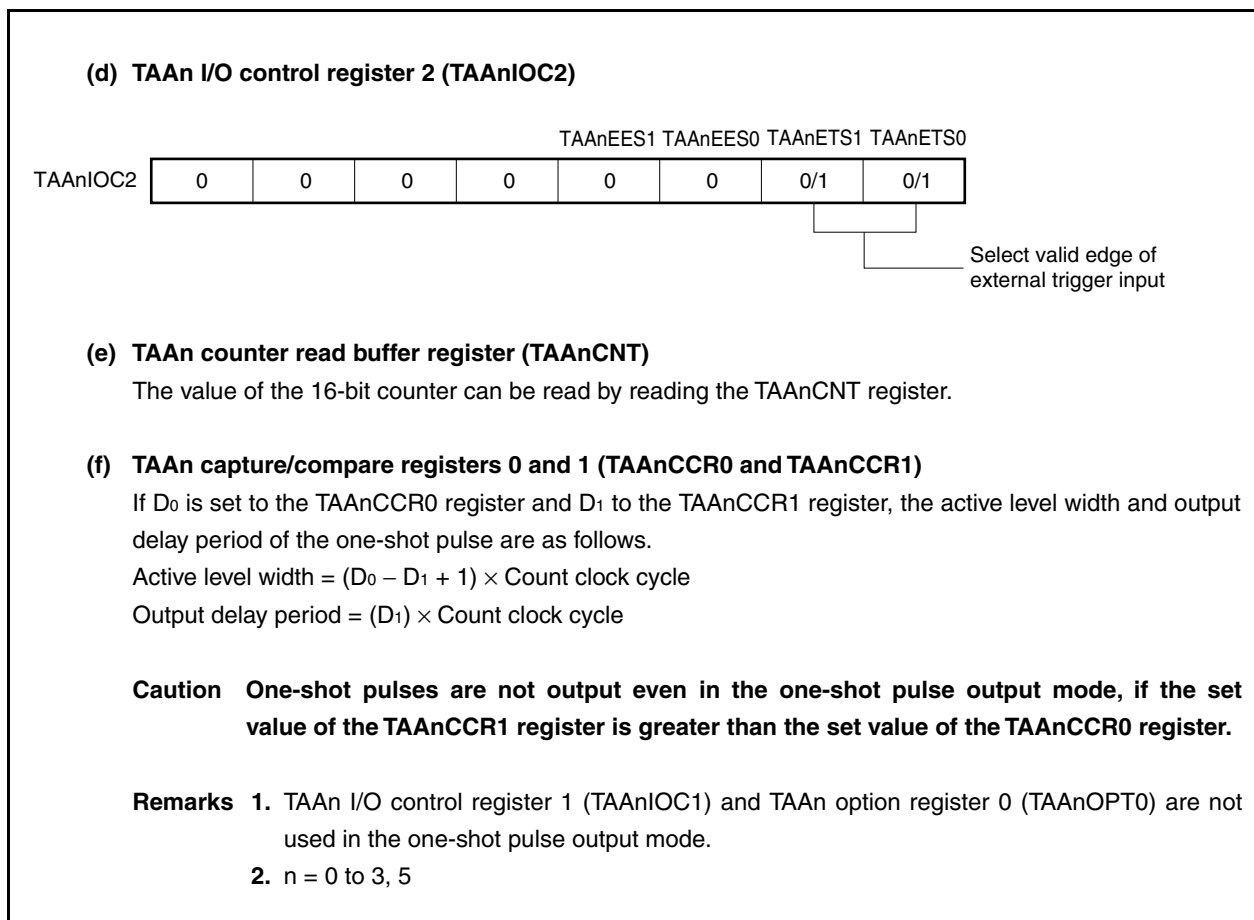**Note** Clear this bit to 0 when the TOAAn0 pin is not used in the one-shot pulse output mode.

**Figure 7-27.  Register Setting for Operation in One-Shot Pulse Output Mode (2/2)**

**(d)  TAAn I/O control register 2 (TAAnIOC2)**

| | | | | | | TAAnEES1 | TAAnEES0 | TAAnETS1 | TAAnETS0 |
|---|---|---|---|---|---|---|---|---|---|

| TAAnIOC2 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
|---|---|---|---|---|---|---|---|---|

Select valid edge of external trigger input

**(e)  TAAn counter read buffer register (TAAnCNT)**

The value of the 16-bit counter can be read by reading the TAAnCNT register.

**(f)  TAAn capture/compare registers 0 and 1 (TAAnCCR0 and TAAnCCR1)**

If $D_0$ is set to the TAAnCCR0 register and $D_1$ to the TAAnCCR1 register, the active level width and output delay period of the one-shot pulse are as follows.

Active level width = $(D_0 - D_1 + 1) \times$ Count clock cycle

Output delay period = $(D_1) \times$ Count clock cycle

**Caution   One-shot pulses are not output even in the one-shot pulse output mode, if the set value of the TAAnCCR1 register is greater than the set value of the TAAnCCR0 register.**

**Remarks  1.** TAAn I/O control register 1 (TAAnIOC1) and TAAn option register 0 (TAAnOPT0) are not used in the one-shot pulse output mode.

**2.** n = 0 to 3, 5

**(1) Operation flow in one-shot pulse output mode**

**Figure 7-28. Software Processing Flow in One-Shot Pulse Output Mode**



<1> Count operation start flow

<2> TAAnCCR0, TAAnCCR1 register setting change flow

**START**

Register initial setting
TAAnCTL0 register
(TAAnCKS0 to TAAnCKS2 bits),
TAAnCTL1 register,
TAAnIOC0 register,
TAAnIOC2 register,
TAAnCCR0 register,
TAAnCCR1 register

Initial setting of these registers is performed before setting the TAAnCE bit to 1.

Setting of TAAnCCR0, TAAnCCR1 registers

As rewriting the TAAnCCRm register immediately forwards to the CCRm buffer register, rewriting immediately after the generation of the INTTAAnCCR0 signal is recommended.

<3> Count operation stop flow

TAAnCE bit = 1

The TAAnCKS0 to TAAnCKS2 bits can be set at the same time when counting has been started (TAAnCE bit = 1). Trigger wait status

TAAnCE bit = 0

Count operation is stopped

**STOP**

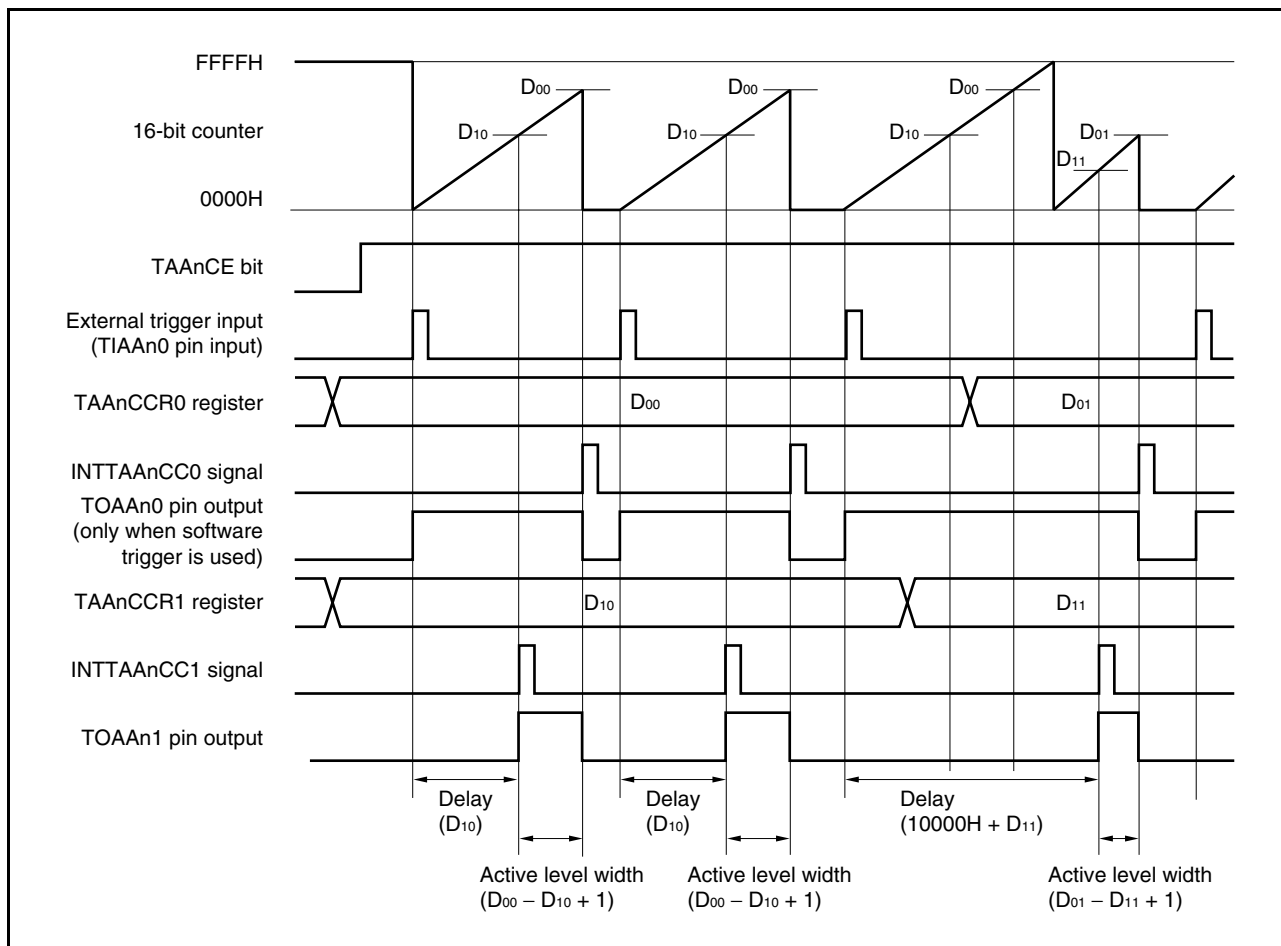**Remark**    n = 0 to 3, 5
                m = 0, 1

**(2)  Operation timing in one-shot pulse output mode**

**(a)  Note on rewriting TAAnCCRm register**

To change the set value of the TAAnCCRm register to a smaller value, stop counting once, and then change the set value.

If the value of the TAAnCCRm register is rewritten to a smaller value during counting, the 16-bit counter may overflow.
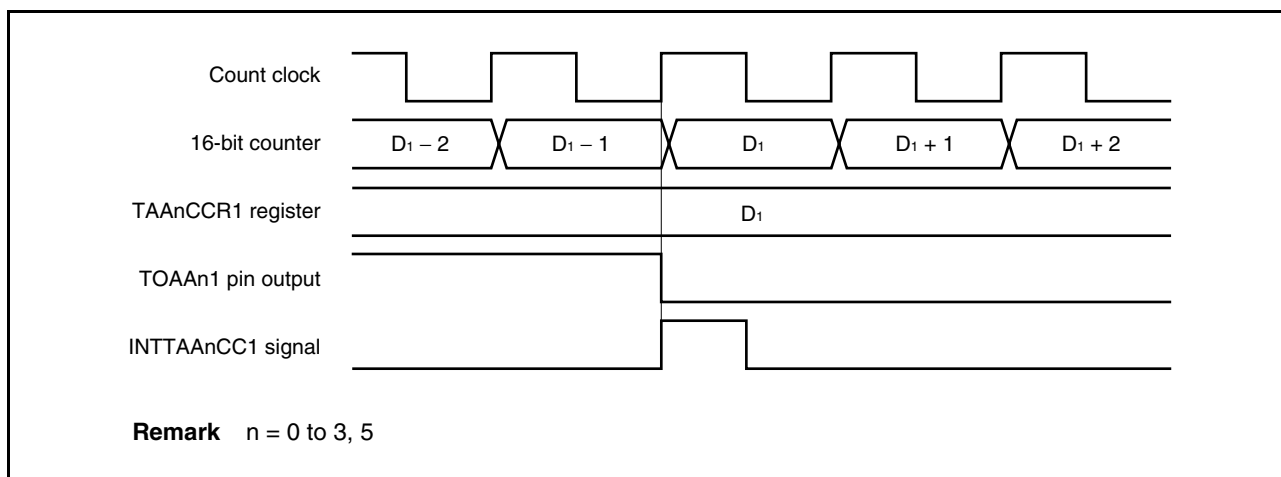


When the TAAnCCR0 register is rewritten from $D_{00}$ to $D_{01}$ and the TAAnCCR1 register from $D_{10}$ to $D_{11}$ where $D_{00} > D_{01}$ and $D_{10} > D_{11}$, if the TAAnCCR1 register is rewritten when the count value of the 16-bit counter is greater than $D_{11}$ and less than $D_{10}$ and if the TAAnCCR0 register is rewritten when the count value is greater than $D_{01}$ and less than $D_{00}$, each set value is reflected as soon as the register has been rewritten and compared with the count value.  The counter counts up to FFFFH and then counts up again from 0000H. When the count value matches $D_{11}$, the counter generates the INTTAAnCC1 signal and asserts the TOAAn1 pin.  When the count value matches $D_{01}$, the counter generates the INTTAAnCC0 signal, deasserts the TOAAn1 pin, and stops counting.

Therefore, the counter may output a pulse with a delay period or active period different from that of the one-shot pulse that is originally expected.

**Remark**   n = 0 to 3, 5
            m = 0, 1

**(b) Generation timing of compare match interrupt request signal (INTTAAnCC1)**

The generation timing of the INTTAAnCC1 signal in the one-shot pulse output mode is different from other INTTAAnCC1 signals; the INTTAAnCC1 signal in the one-shot pulse output mode is generated when the count value of the 16-bit counter matches the value of the TAAnCCR1 register.



| Count clock | | | | | |
|---|---|---|---|---|---|
| 16-bit counter | $D_1 - 2$ | $D_1 - 1$ | $D_1$ | $D_1 + 1$ | $D_1 + 2$ |
| TAAnCCR1 register | | | $D_1$ | | |
| TOAAn1 pin output | | | | | |
| INTTAAnCC1 signal | | | | | |

**Remark**    n = 0 to 3, 5

Usually, the INTTAAnCC1 signal is generated the next time the 16-bit counter counts after its count value matches the value of the TAAnCCR1 register.

In the one-shot pulse output mode, however, it is generated one clock earlier. This is because the timing is changed to match the change timing of the TOAAn1 pin.

### 7.5.5   PWM output mode (TAAnMD2 to TAAnMD0 bits = 100)

In the PWM output mode, a PWM waveform is output from the TOAAn1 pin when the TAAnCTL0.TAAnCE bit is set to 1. In addition, a pulse with one cycle of the PWM waveform as half its cycle is output from the TOAAn0 pin.

**Figure 7-29.  Configuration in PWM Output Mode**
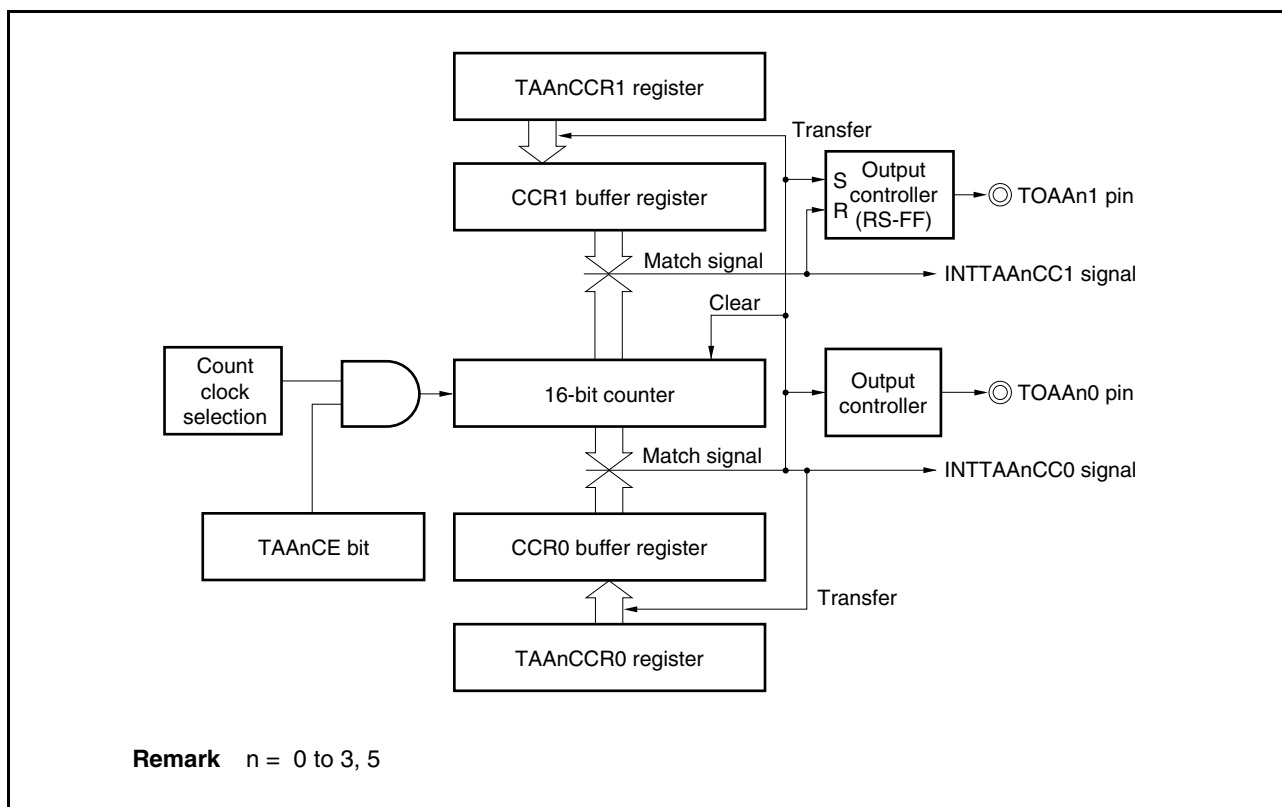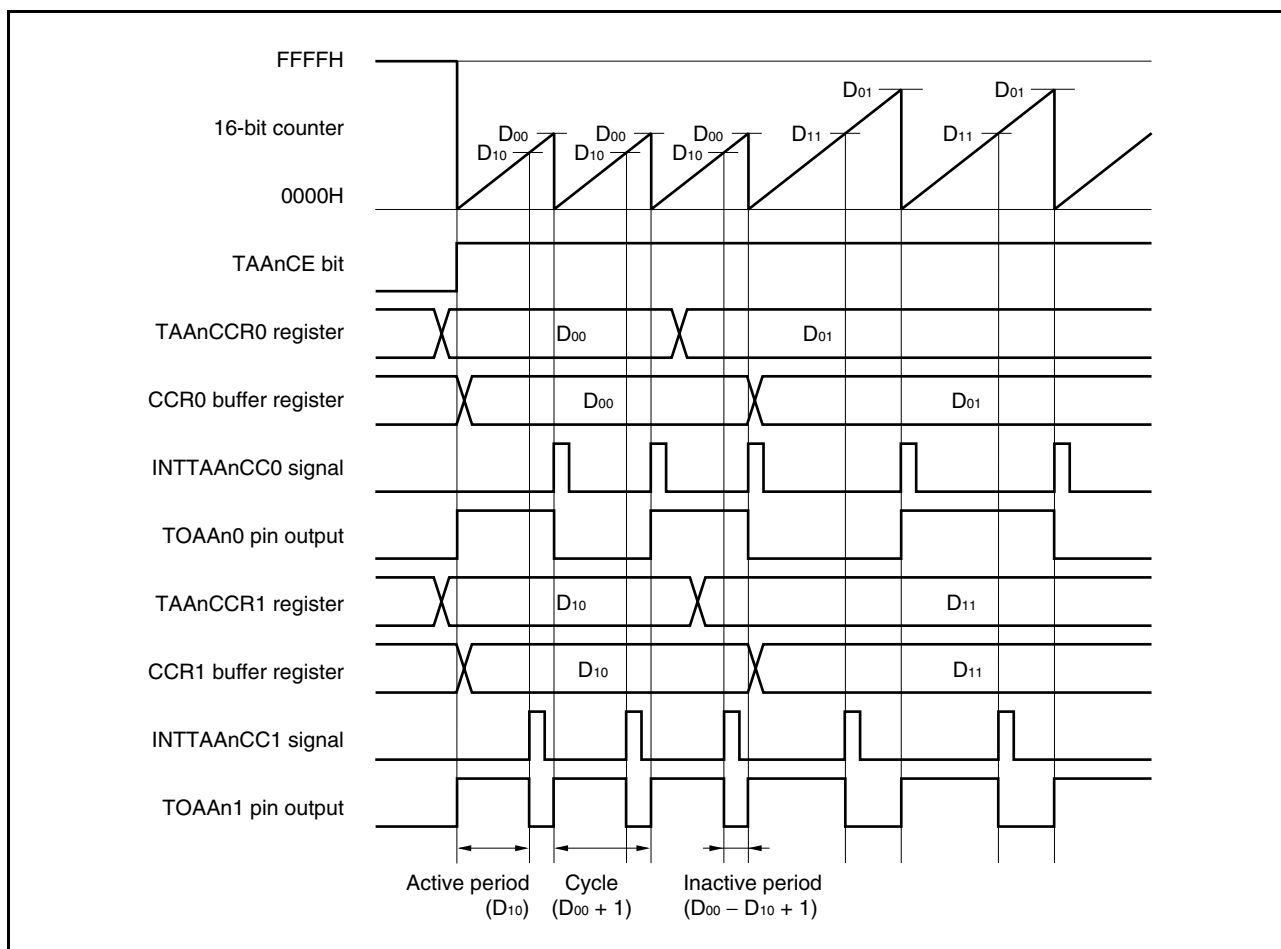


**Remark**   n = 0 to 3, 5

**Figure 7-30. Basic Timing in PWM Output Mode**



When the TAAnCE bit is set to 1, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a PWM waveform from the TOAAn1 pin.

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

    Active level width = (Set value of TAAnCCR1 register) × Count clock cycle

    Cycle = (Set value of TAAnCCR0 register + 1) × Count clock cycle

    Duty factor = (Set value of TAAnCCR1 register)/(Set value of TAAnCCR0 register + 1)

The PWM waveform can be changed by rewriting the TAAnCCRm register while the counter is operating. The newly written value is reflected when the count value of the 16-bit counter matches the value of the CCR0 buffer register and the 16-bit counter is cleared to 0000H.

The compare match interrupt request signal INTTAAnCC0 is generated the next time the 16-bit counter counts after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H. The compare match interrupt request signal INTTAAnCC1 is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The value set to the TAAnCCRm register is transferred to the CCRm buffer register when the count value of the 16-bit counter matches the value of the CCRm buffer register and the 16-bit counter is cleared to 0000H.

**Remark**   n = 0 to 3, 5
          m = 0, 1

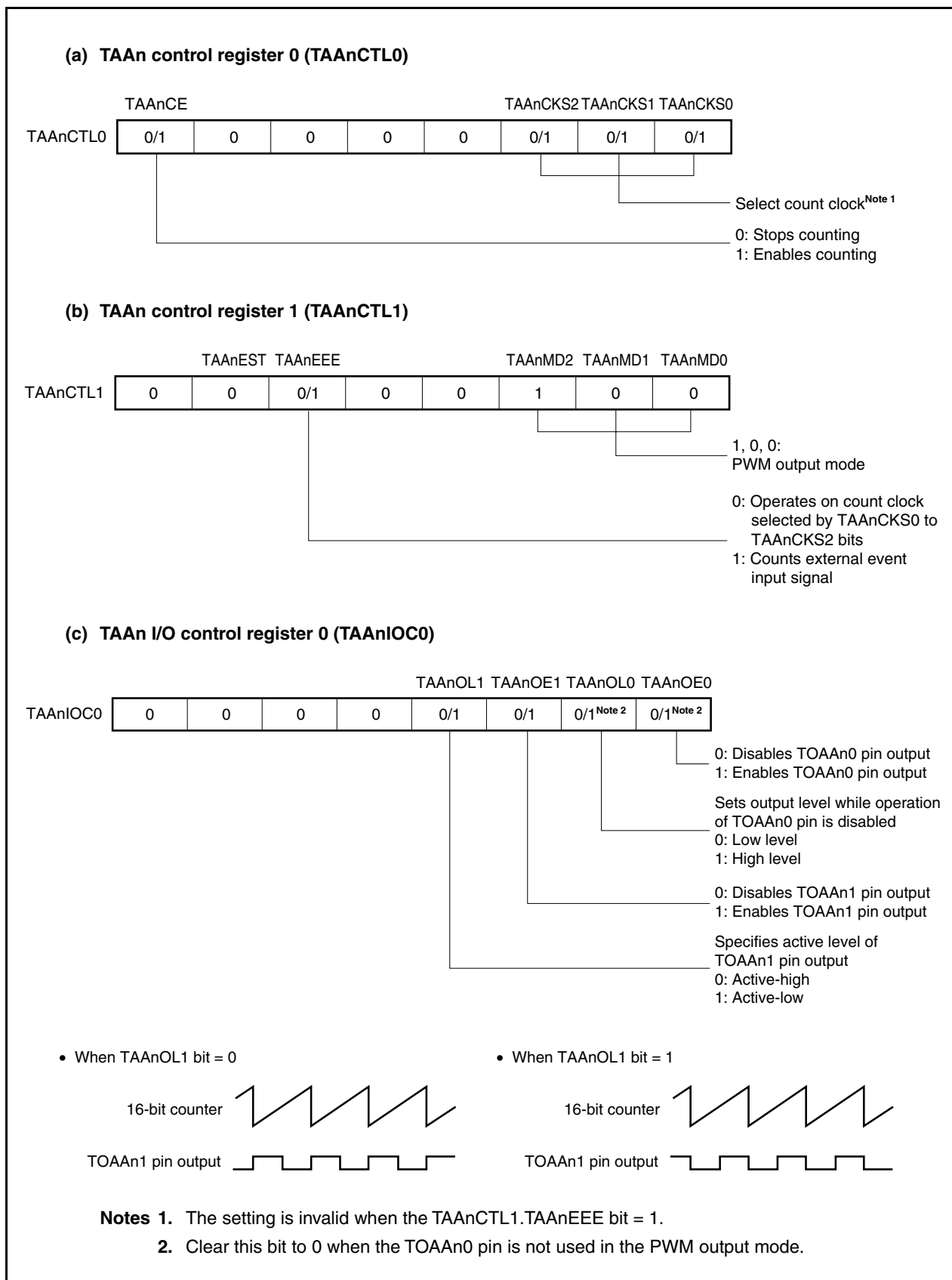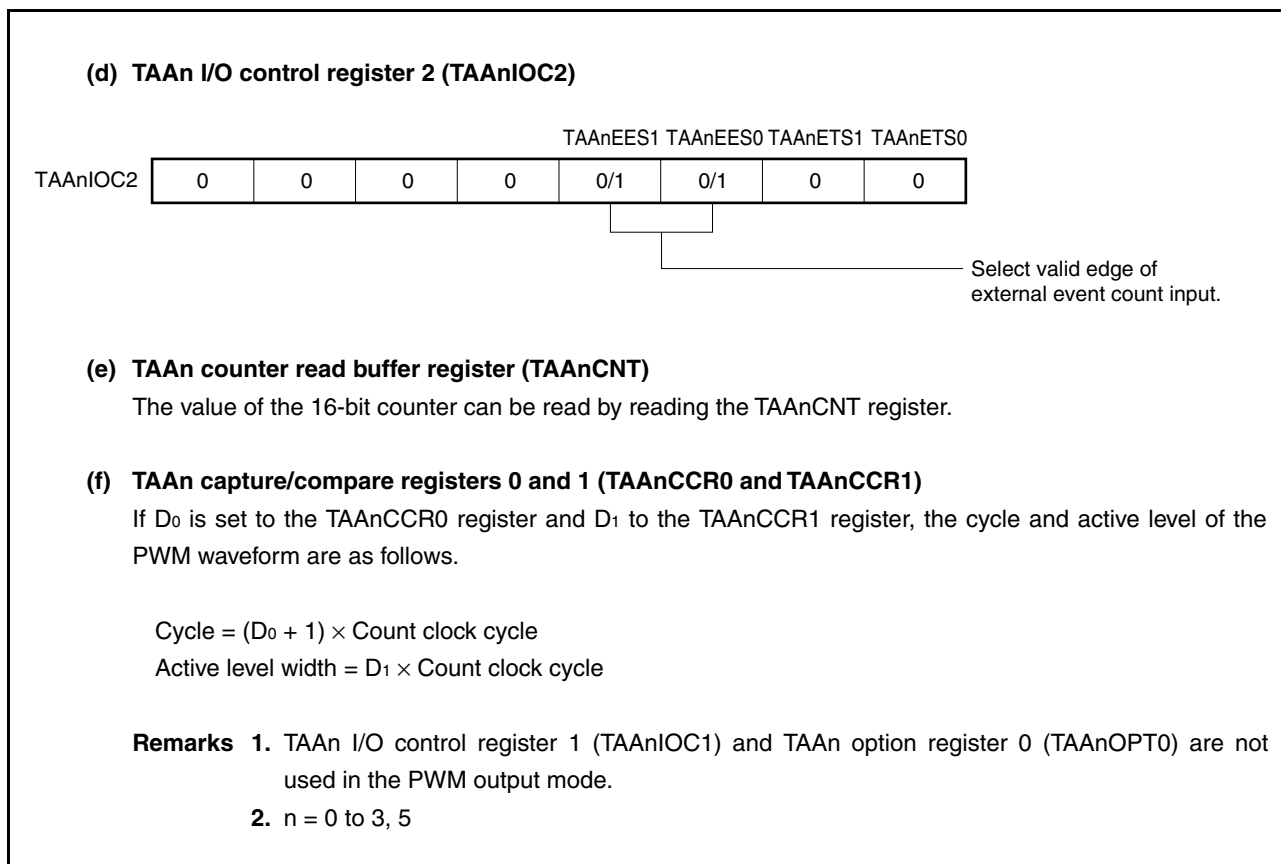**Figure 7-31.  Setting of Registers in PWM Output Mode (1/2)**

**(a)  TAAn control register 0 (TAAnCTL0)**

TAAnCE ... TAAnCKS2 TAAnCKS1 TAAnCKS0

TAAnCTL0 | 0/1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 |

Select count clock[Note 1]

0: Stops counting
1: Enables counting

**(b)  TAAn control register 1 (TAAnCTL1)**

TAAnEST TAAnEEE ... TAAnMD2 TAAnMD1 TAAnMD0

TAAnCTL1 | 0 | 0 | 0/1 | 0 | 0 | 1 | 0 | 0 |

1, 0, 0:
PWM output mode

0: Operates on count clock
selected by TAAnCKS0 to
TAAnCKS2 bits
1: Counts external event
input signal

**(c)  TAAn I/O control register 0 (TAAnIOC0)**

TAAnOL1 TAAnOE1 TAAnOL0 TAAnOE0

TAAnIOC0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1[Note 2] | 0/1[Note 2] |

0: Disables TOAAn0 pin output
1: Enables TOAAn0 pin output

Sets output level while operation
of TOAAn0 pin is disabled
0: Low level
1: High level

0: Disables TOAAn1 pin output
1: Enables TOAAn1 pin output

Specifies active level of
TOAAn1 pin output
0: Active-high
1: Active-low

- When TAAnOL1 bit = 0

16-bit counter

TOAAn1 pin output

- When TAAnOL1 bit = 1

16-bit counter

TOAAn1 pin output



**Notes 1.**  The setting is invalid when the TAAnCTL1.TAAnEEE bit = 1.
　　　**2.**  Clear this bit to 0 when the TOAAn0 pin is not used in the PWM output mode.

**Figure 7-31.  Setting of Registers in PWM Output Mode (2/2)**

**(d)  TAAn I/O control register 2 (TAAnIOC2)**

| | | | | TAAnEES1 | TAAnEES0 | TAAnETS1 | TAAnETS0 |
|---|---|---|---|---|---|---|---|
| TAAnIOC2 | 0 | 0 | 0 | 0/1 | 0/1 | 0 | 0 |

Select valid edge of
external event count input.

**(e)  TAAn counter read buffer register (TAAnCNT)**

The value of the 16-bit counter can be read by reading the TAAnCNT register.

**(f)  TAAn capture/compare registers 0 and 1 (TAAnCCR0 and TAAnCCR1)**

If $D_0$ is set to the TAAnCCR0 register and $D_1$ to the TAAnCCR1 register, the cycle and active level of the PWM waveform are as follows.

Cycle = $(D_0 + 1) \times$ Count clock cycle
Active level width = $D_1 \times$ Count clock cycle

**Remarks  1.** TAAn I/O control register 1 (TAAnIOC1) and TAAn option register 0 (TAAnOPT0) are not used in the PWM output mode.

**2.** n = 0 to 3, 5

**(1) Operation flow in PWM output mode**

**Figure 7-32. Software Processing Flow in PWM Output Mode (1/2)**

**Figure 7-32.  Software Processing Flow in PWM Output Mode (2/2)**



**Remark**  n = 0 to 3, 5
m = 0, 1

**(2) PWM output mode operation timing**

**(a) Changing pulse width during operation**

To change the PWM waveform while the counter is operating, write the TAAnCCR1 register last.

Rewrite the TAAnCCRm register after writing the TAAnCCR1 register after the INTTAAnCC1 signal is detected.



To transfer data from the TAAnCCRm register to the CCRm buffer register, the TAAnCCR1 register must be written.

To change both the cycle and active level of the PWM waveform at this time, first set the cycle to the TAAnCCR0 register and then set the active level to the TAAnCCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TAAnCCR0 register, and then write the same value to the TAAnCCR1 register.

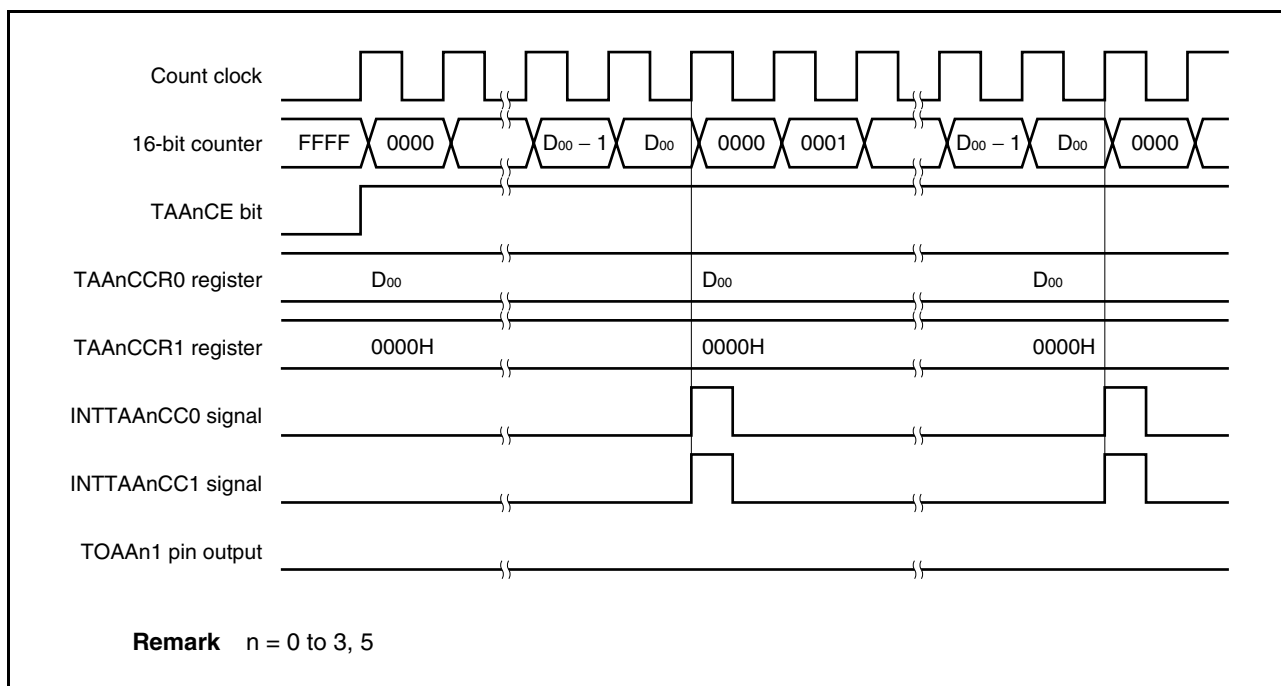To change only the active level width (duty factor) of the PWM waveform, only the TAAnCCR1 register has to be set.

After data is written to the TAAnCCR1 register, the value written to the TAAnCCRm register is transferred to the CCRm buffer register in synchronization with clearing of the 16-bit counter, and is used as the value compared with the 16-bit counter.

To write the TAAnCCR0 or TAAnCCR1 register again after writing the TAAnCCR1 register once, do so after the INTTAAnCC0 signal is generated.  Otherwise, the value of the CCRm buffer register may become undefined because the timing of transferring data from the TAAnCCRm register to the CCRm buffer register conflicts with writing the TAAnCCRm register.
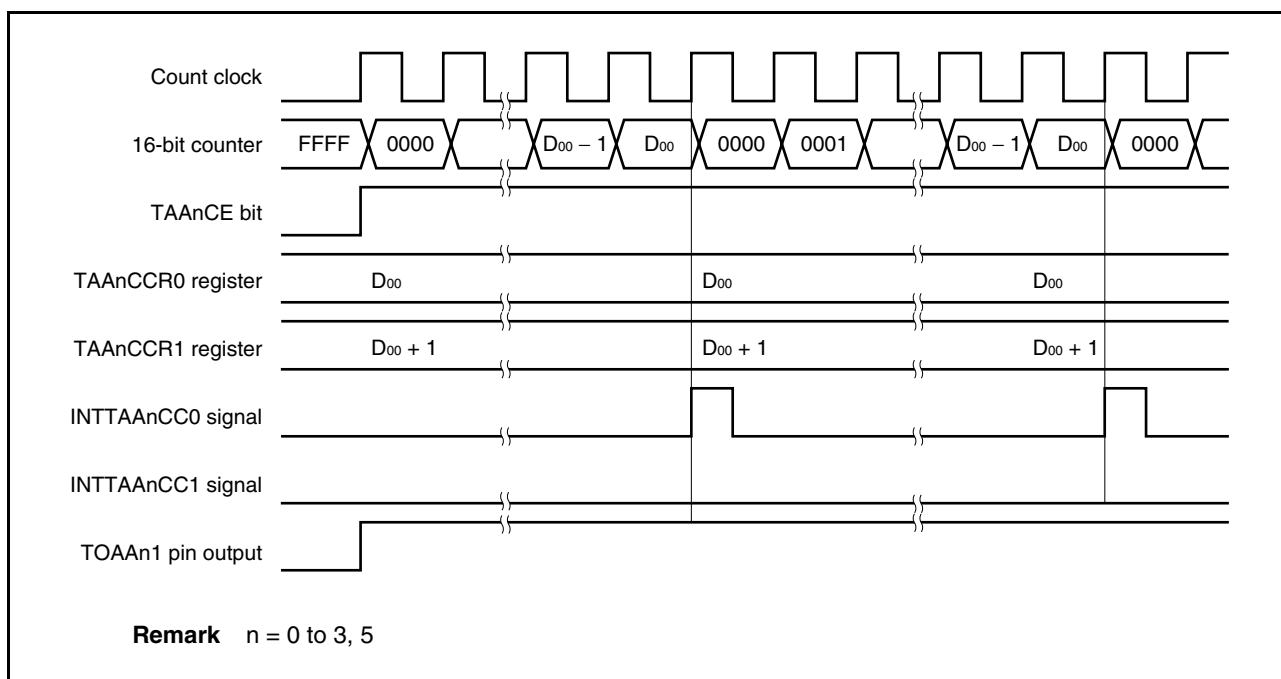
**Remark**    $n$ = 0 to 3, 5

$m$ = 0, 1

**(b) 0%/100% output of PWM waveform**

To output a 0% waveform, set the TAAnCCR1 register to 0000H. If the set value of the TAAnCCR0 register is FFFFH, the INTTAAnCC1 signal is generated periodically.
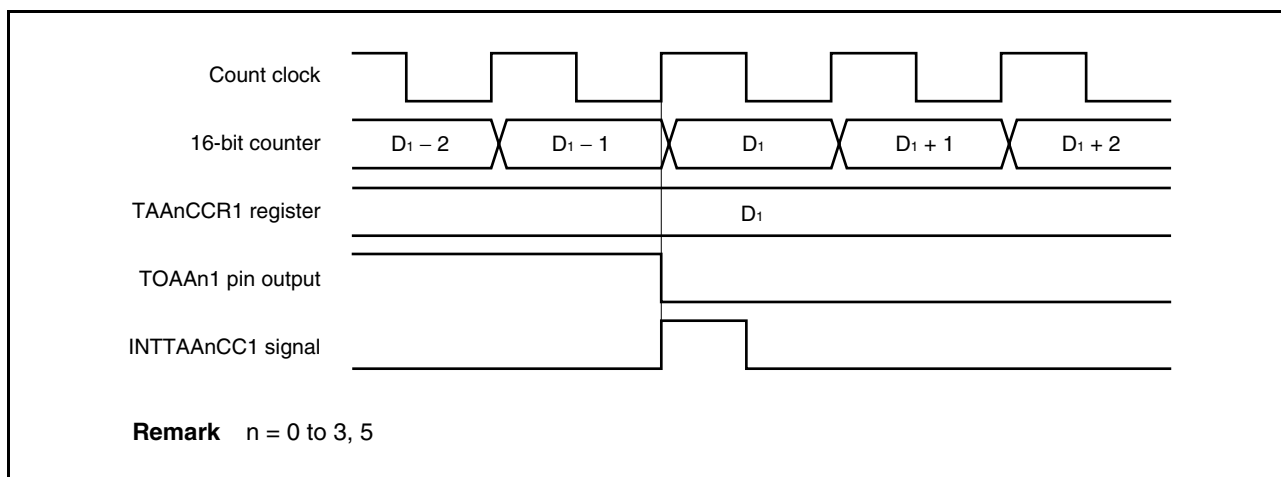


**Remark**   n = 0 to 3, 5

To output a 100% waveform, set a value of (set value of TAAnCCR0 register + 1) to the TAAnCCR1 register. If the set value of the TAAnCCR0 register is FFFFH, 100% output cannot be produced.



**Remark**   n = 0 to 3, 5

**(c) Generation timing of compare match interrupt request signal (INTTAAnCC1)**

The timing of generation of the INTTAAnCC1 signal in the PWM output mode differs from the timing of other INTTAAnCC1 signals; the INTTAAnCC1 signal in the PWM output mode is generated when the count value of the 16-bit counter matches the value of the TAAnCCR1 register.



| | |
|---|---|
| Count clock | |
| 16-bit counter | $D_1 - 2$   $D_1 - 1$   $D_1$   $D_1 + 1$   $D_1 + 2$ |
| TAAnCCR1 register | $D_1$ |
| TOAAn1 pin output | |
| INTTAAnCC1 signal | |

**Remark**   n = 0 to 3, 5

Usually, the INTTAAnCC1 signal is generated in synchronization with the next count-up after the count value of the 16-bit counter matches the value of the TAAnCCR1 register.

In the PWM output mode, however, it is generated one clock earlier.  This is because the timing is changed to match the change timing of the output signal of the TOAAn1 pin.

### 7.5.6   Free-running timer mode (TAAnMD2 to TAAnMD0 bits = 101)

In the free-running timer mode, 16-bit timer/event counter AA starts counting when the TAAnCTL0.TAAnCE bit is set to 1. At this time, the TAAnCCRm register can be used as a compare register or a capture register, depending on the setting of the TAAnOPT0.TAAnCCS0 and TAAnOPT0.TAAnCCS1 bits.
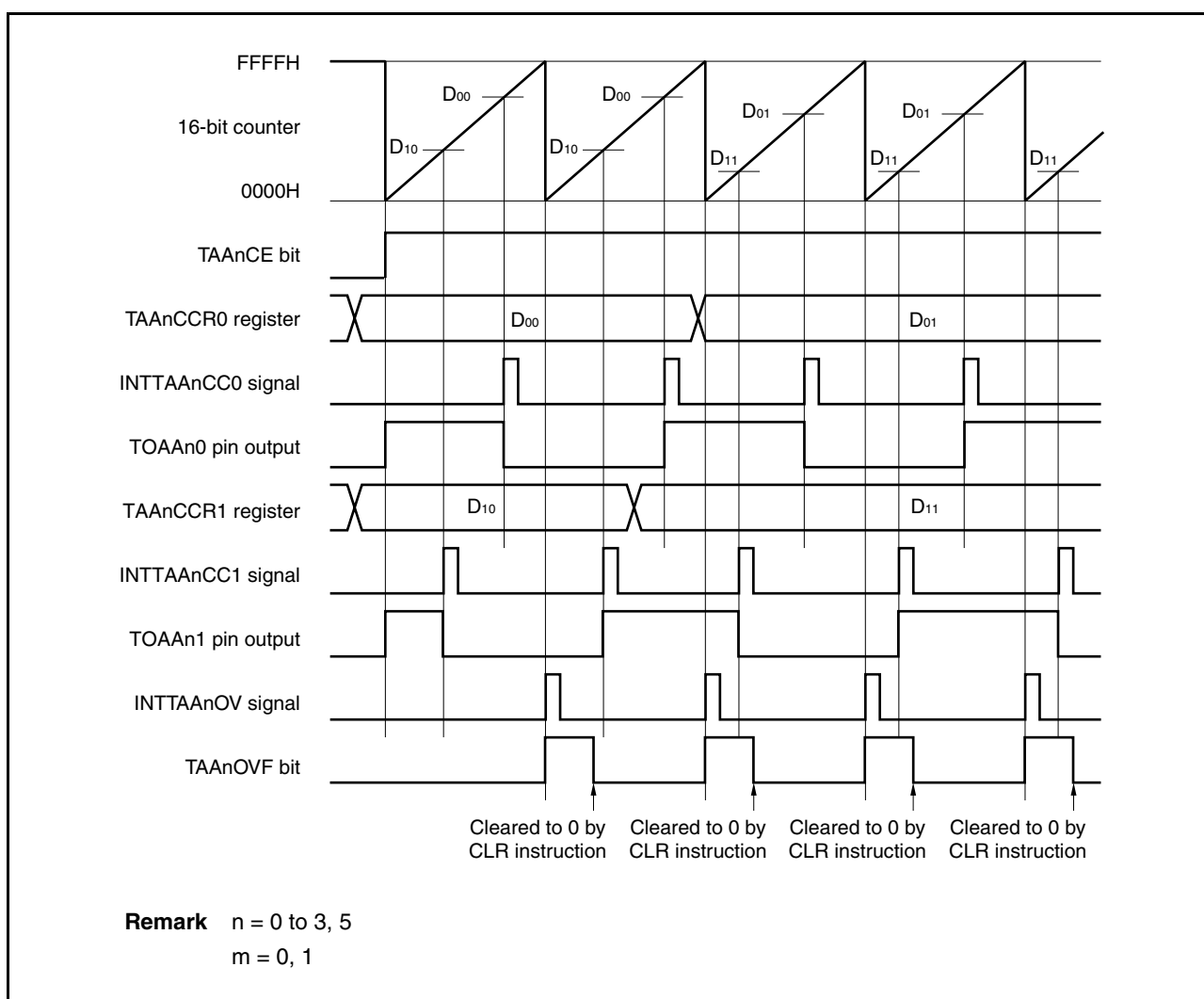
**Figure 7-33.  Configuration in Free-Running Timer Mode**
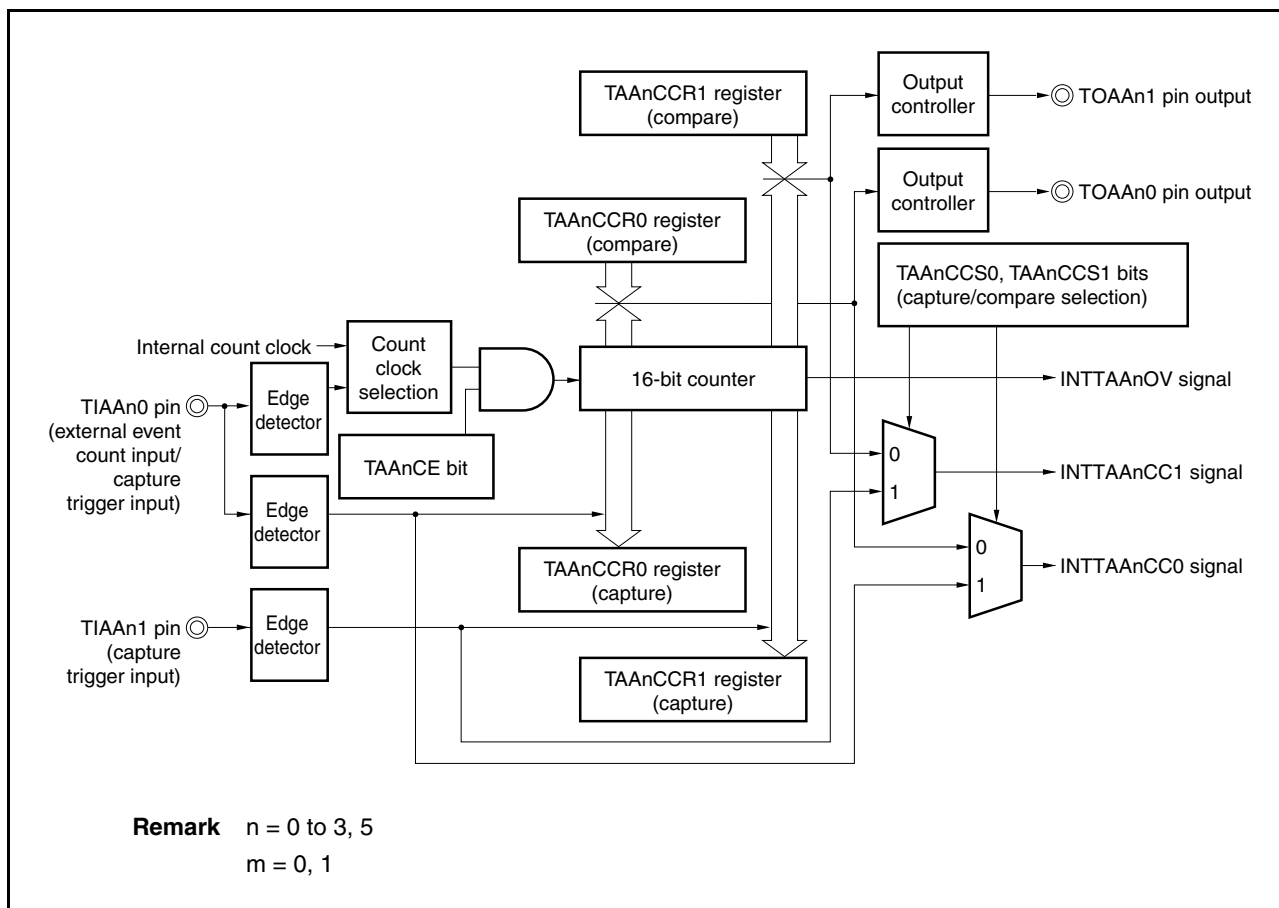
When the TAAnCE bit is set to 1, 16-bit timer/event counter AA starts counting, and the output signals of the TOAAn0 and TOAAn1 pins are inverted. When the count value of the 16-bit counter later matches the set value of the TAAnCCRm register, a compare match interrupt request signal (INTTAAnCCm) is generated, and the output signal of the TOAAnm pin is inverted.

The 16-bit counter continues counting in synchronization with the count clock. When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTAAnOV) at the next clock, is cleared to 0000H, and continues counting. At this time, the overflow flag (TAAnOPT0.TAAnOVF bit) is also set to 1. Clear the overflow flag to 0 by executing the CLR instruction by software.

The TAAnCCRm register can be rewritten while the counter is operating. If it is rewritten, the new value is reflected at that time, and compared with the count value.

**Figure 7-34. Basic Timing in Free-Running Timer Mode (Compare Function)**



Remark n = 0 to 3, 5
m = 0, 1

When the TAAnCE bit is set to 1, the 16-bit counter starts counting.  When the valid edge input to the TIAAnm pin is detected, the count value of the 16-bit counter is stored in the TAAnCCRm register, and a capture interrupt request signal (INTTAAnCCm) is generated.

The 16-bit counter continues counting in synchronization with the count clock.  When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTAAnOV) at the next clock, is cleared to 0000H, and continues counting. At this time, the overflow flag (TAAnOPT0.TAAnOVF bit) is also set to 1.  Clear the overflow flag to 0 by executing the CLR instruction by software.

**Figure 7-35.  Basic Timing in Free-Running Timer Mode (Capture Function)**
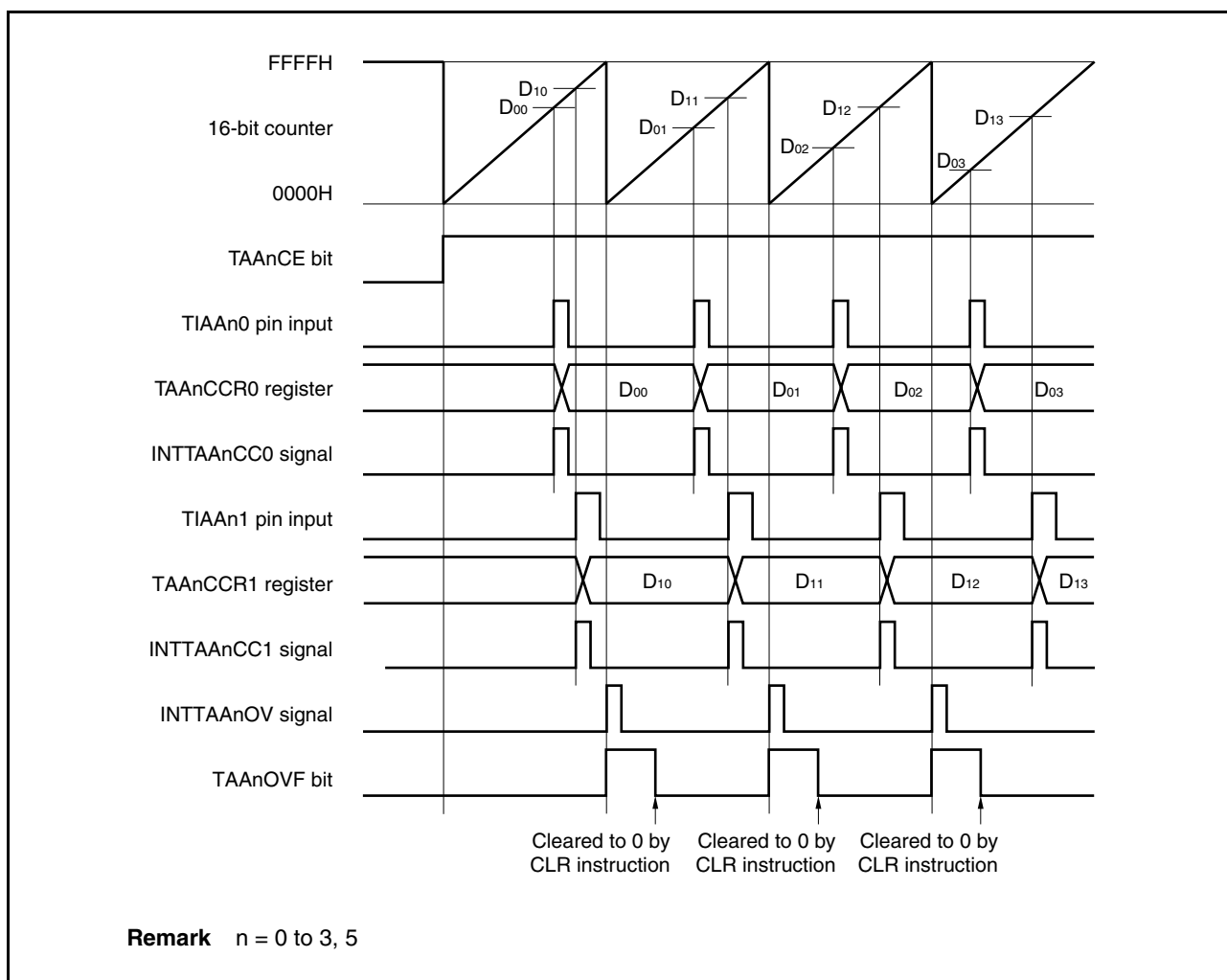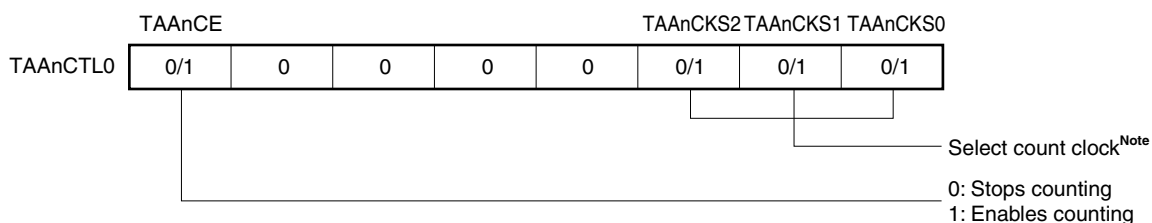


**Remark**    n = 0 to 3, 5

**Figure 7-36. Register Setting in Free-Running Timer Mode (1/2)**



**(a) TAAn control register 0 (TAAnCTL0)**

TAAnCE            TAAnCKS2 TAAnCKS1 TAAnCKS0

| TAAnCTL0 | 0/1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 |
|---|---|---|---|---|---|---|---|---|

Select count clock**Note**

0: Stops counting
1: Enables counting

**Note** The setting is invalid when the TAAnCTL1.TAAnEEE bit = 1

**(b) TAAn control register 1 (TAAnCTL1)**

TAAnEST TAAnEEE        TAAnMD2 TAAnMD1 TAAnMD0

| TAAnCTL1 | 0 | 0 | 0/1 | 0 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|---|---|

1, 0, 1:
Free-running mode

0: Operates with count clock
   selected by TAAnCKS0 to
   TAAnCKS2 bits
1: Counts on external
   event count input signal

**(c) TAAn I/O control register 0 (TAAnIOC0)**

TAAnOL1 TAAnOE1 TAAnOL0 TAAnOE0

| TAAnIOC0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |
|---|---|---|---|---|---|---|---|---|

0: Disables TOAAn0 pin output
1: Enables TOAAn0 pin output

Sets output level with operation
of TOAAn0 pin disabled
0: Low level
1: High level

0: Disables TOAAn1 pin output
1: Enables TOAAn1 pin output

Sets output level with operation
of TOAAn1 pin disabled
0: Low level
1: High level

**Figure 7-36.  Register Setting in Free-Running Timer Mode (2/2)**

**(d) TAAn I/O control register 1 (TAAnIOC1)**

|  | | | | TAAnIS3 | TAAnIS2 | TAAnIS1 | TAAnIS0 |
|---|---|---|---|---|---|---|---|
| TAAnIOC1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |

Select valid edge
of TIAAn0 pin input

Select valid edge
of TIAAn1 pin input

**(e) TAAn I/O control register 2 (TAAnIOC2)**

|  | | | | TAAnEES1 | TAAnEES0 | TAAnETS1 | TAAnETS0 |
|---|---|---|---|---|---|---|---|
| TAAnIOC2 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0 | 0 |

Select valid edge of
external event count input

**(f) TAAn option register 0 (TAAnOPT0)**

|  | | TAAnCCS1 | TAAnCCS0 | | | | TAAnOVF |
|---|---|---|---|---|---|---|---|
| TAAnOPT0 | 0 | 0 | 0/1 | 0/1 | 0 | 0 | 0 | 0/1 |

Overflow flag

Specifies if TAAnCCR0
register functions as
capture or compare register

Specifies if TAAnCCR1
register functions as
capture or compare register
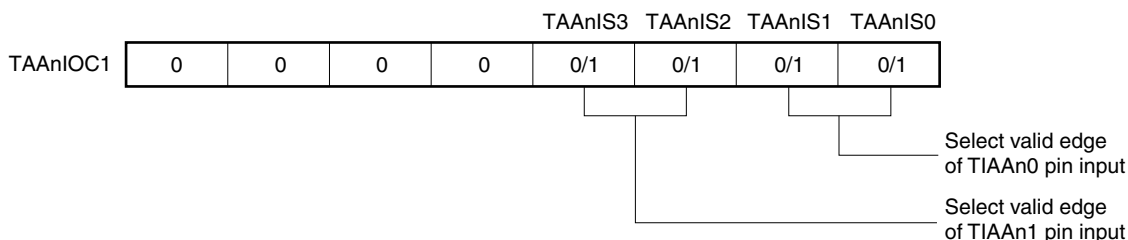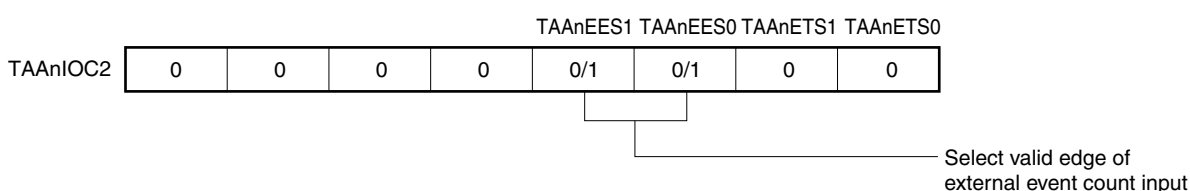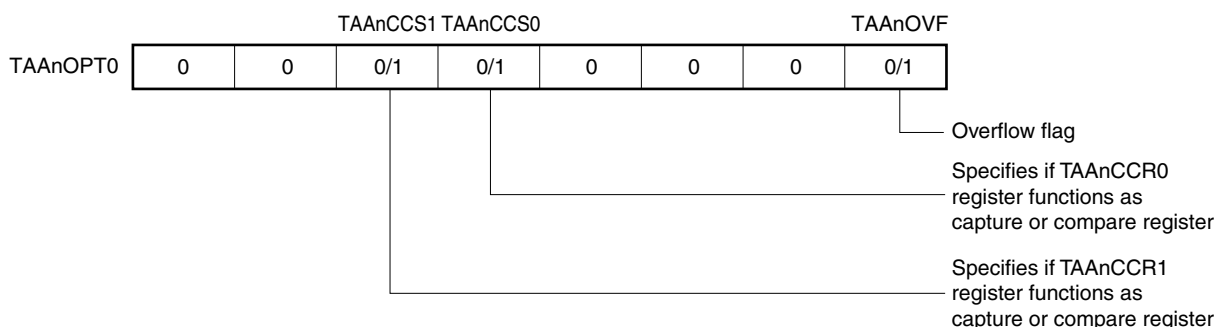
**(g) TAAn counter read buffer register (TAAnCNT)**

The value of the 16-bit counter can be read by reading the TAAnCNT register.

**(h) TAAn capture/compare registers 0 and 1 (TAAnCCR0 and TAAnCCR1)**

These registers function as capture registers or compare registers depending on the setting of the TAAnOPT0.TAAnCCSm bit.

When the registers function as capture registers, they store the count value of the 16-bit counter when the valid edge input to the TIAAnm pin is detected.

When the registers function as compare registers and when $D_m$ is set to the TAAnCCRm register, the INTTAAnCCm signal is generated when the counter reaches ($D_m$ + 1), and the output signal of the TOAAnm pin is inverted.

**Remark**   n = 0 to 3, 5
m = 0, 1

**(1) Operation flow in free-running timer mode**

**(a) When using capture/compare register as compare register**

**Figure 7-37.  Software Processing Flow in Free-Running Timer Mode (Compare Function) (1/2)**
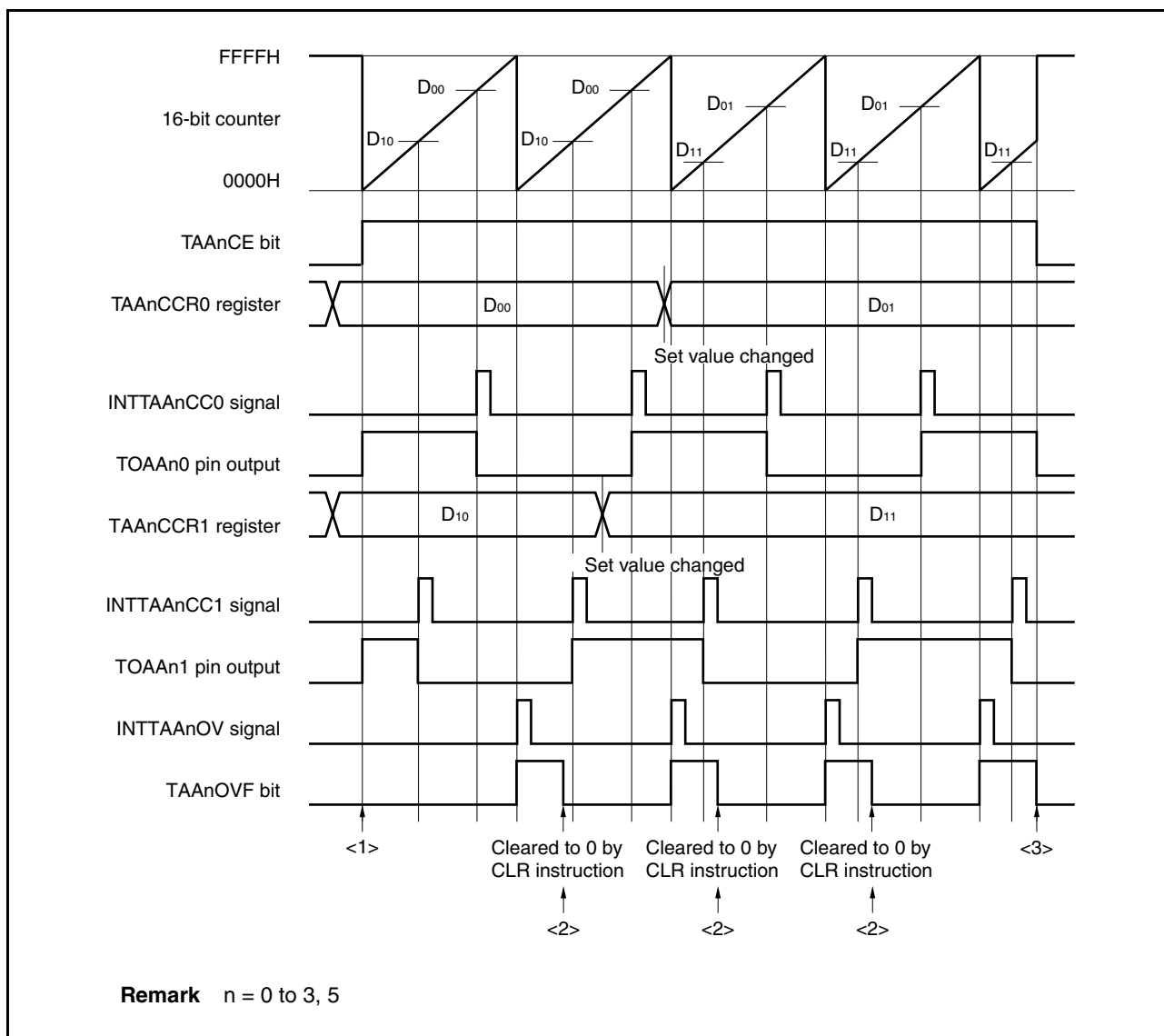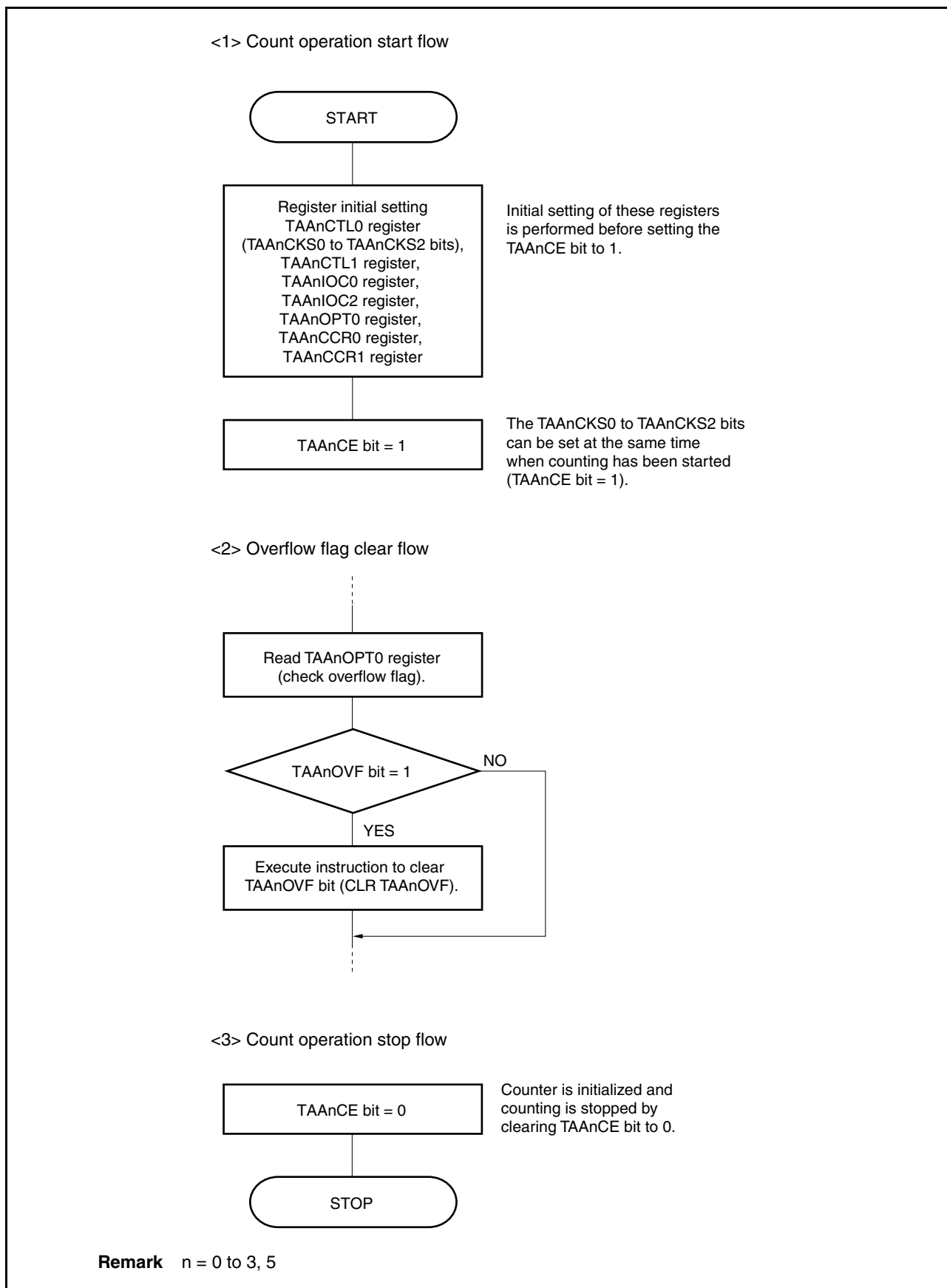


**Remark**   n = 0 to 3, 5

**Figure 7-37. Software Processing Flow in Free-Running Timer Mode (Compare Function) (2/2)**

<1> Count operation start flow

```
            ┌──────────────────┐
            │      START       │
            └──────────────────┘
                     │
    ┌────────────────────────────────┐
    │    Register initial setting     │      Initial setting of these registers
    │       TAAnCTL0 register         │      is performed before setting the
    │  (TAAnCKS0 to TAAnCKS2 bits),   │      TAAnCE bit to 1.
    │       TAAnCTL1 register,        │
    │       TAAnIOC0 register,        │
    │       TAAnIOC2 register,        │
    │       TAAnOPT0 register,        │
    │       TAAnCCR0 register,        │
    │       TAAnCCR1 register         │
    └────────────────────────────────┘
                     │
    ┌────────────────────────────────┐      The TAAnCKS0 to TAAnCKS2 bits
    │         TAAnCE bit = 1          │      can be set at the same time
    └────────────────────────────────┘      when counting has been started
                                             (TAAnCE bit = 1).
```

<2> Overflow flag clear flow

```
    ┌────────────────────────────────┐
    │      Read TAAnOPT0 register      │
    │      (check overflow flag).      │
    └────────────────────────────────┘
                     │
              ╱──────────────╲              NO
             ╱  TAAnOVF bit = 1 ╲ ───────────────┐
             ╲                  ╱                 │
              ╲──────────────╱                    │
                     │ YES                        │
    ┌────────────────────────────────┐            │
    │     Execute instruction to clear │           │
    │     TAAnOVF bit (CLR TAAnOVF).   │           │
    └────────────────────────────────┘            │
                     │ ◄──────────────────────────┘
```

<3> Count operation stop flow

```
    ┌────────────────────────────────┐      Counter is initialized and
    │         TAAnCE bit = 0          │      counting is stopped by
    └────────────────────────────────┘      clearing TAAnCE bit to 0.
                     │
            ┌──────────────────┐
            │       STOP       │
            └──────────────────┘
```

**Remark**    n = 0 to 3, 5

**(b)  When using capture/compare register as capture register**

**Figure 7-38.  Software Processing Flow in Free-Running Timer Mode (Capture Function) (1/2)**
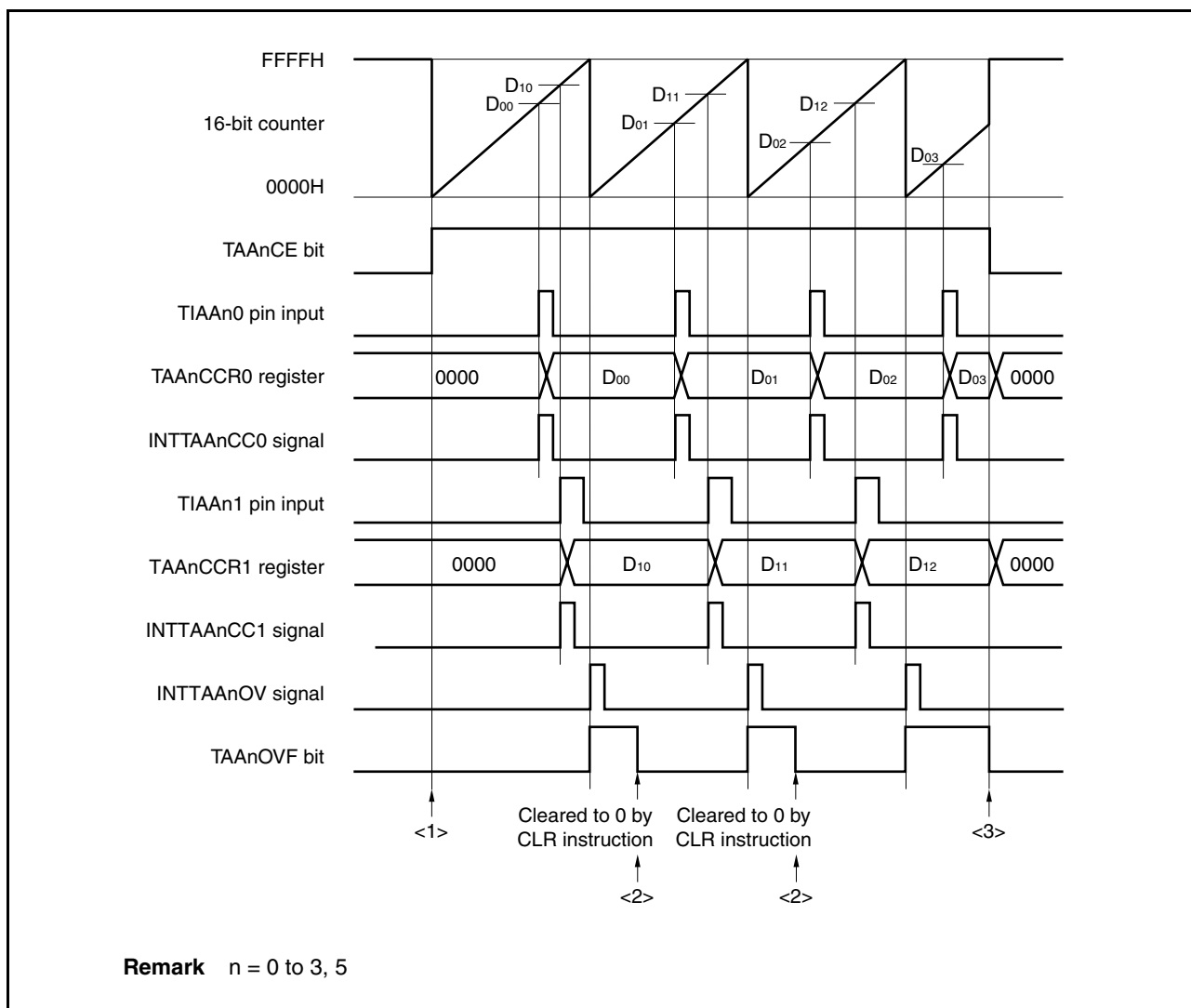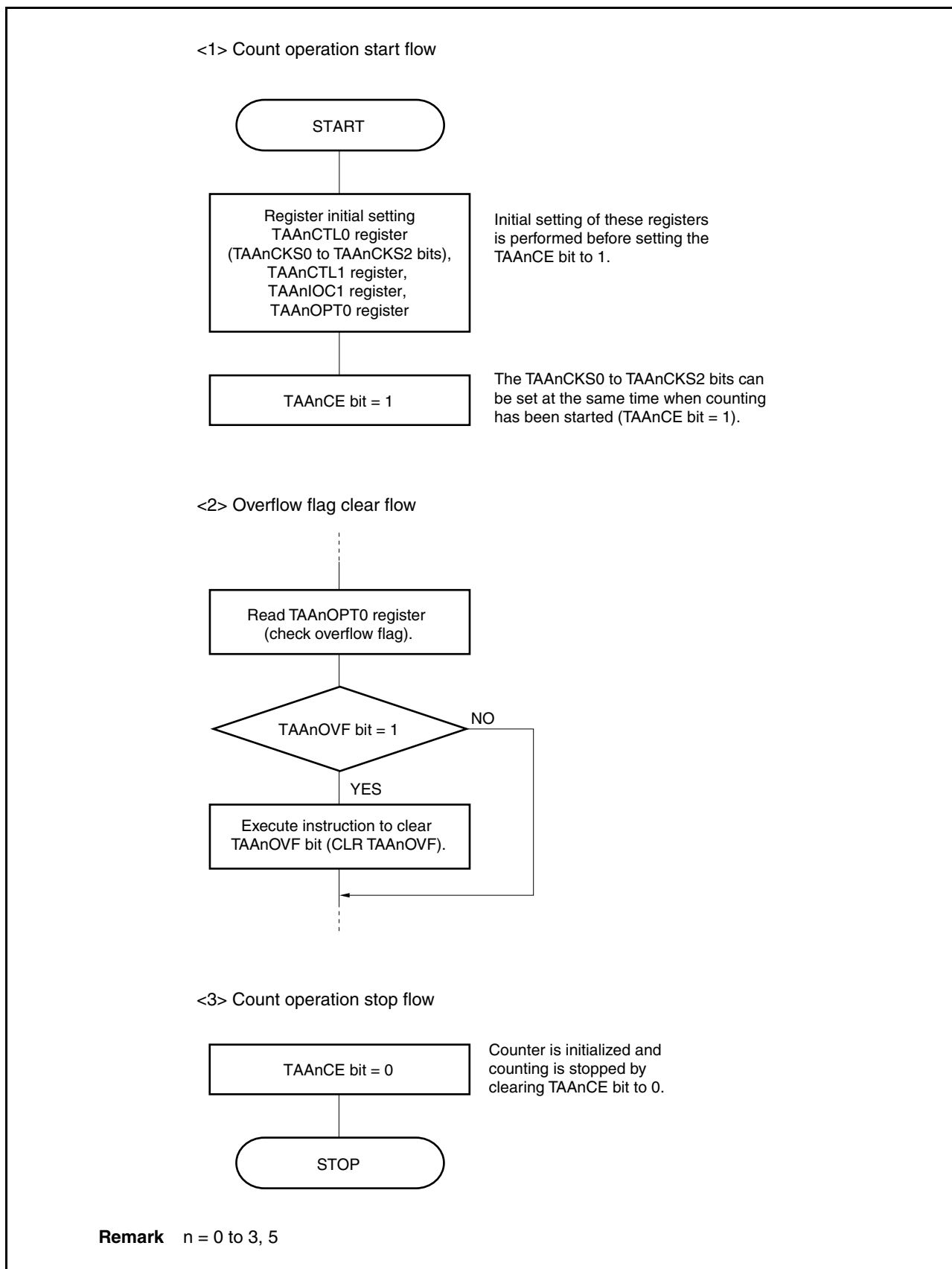
**Figure 7-38.  Software Processing Flow in Free-Running Timer Mode (Capture Function) (2/2)**

<1> Count operation start flow

START

Register initial setting
TAAnCTL0 register
(TAAnCKS0 to TAAnCKS2 bits),
TAAnCTL1 register,
TAAnIOC1 register,
TAAnOPT0 register

Initial setting of these registers
is performed before setting the
TAAnCE bit to 1.

TAAnCE bit = 1

The TAAnCKS0 to TAAnCKS2 bits can
be set at the same time when counting
has been started (TAAnCE bit = 1).

<2> Overflow flag clear flow

Read TAAnOPT0 register
(check overflow flag).

TAAnOVF bit = 1          NO

YES

Execute instruction to clear
TAAnOVF bit (CLR TAAnOVF).

<3> Count operation stop flow

TAAnCE bit = 0

Counter is initialized and
counting is stopped by
clearing TAAnCE bit to 0.

STOP

**Remark**   n = 0 to 3, 5

**(2) Operation timing in free-running timer mode**

**(a) Interval operation with TAAnCCRm register used as compare register**

When 16-bit timer/event counter AA is used as an interval timer with the TAAnCCRm register used as a compare register, software processing is necessary for setting a comparison value to generate the next interrupt request signal each time the INTTAAnCCm signal has been detected.
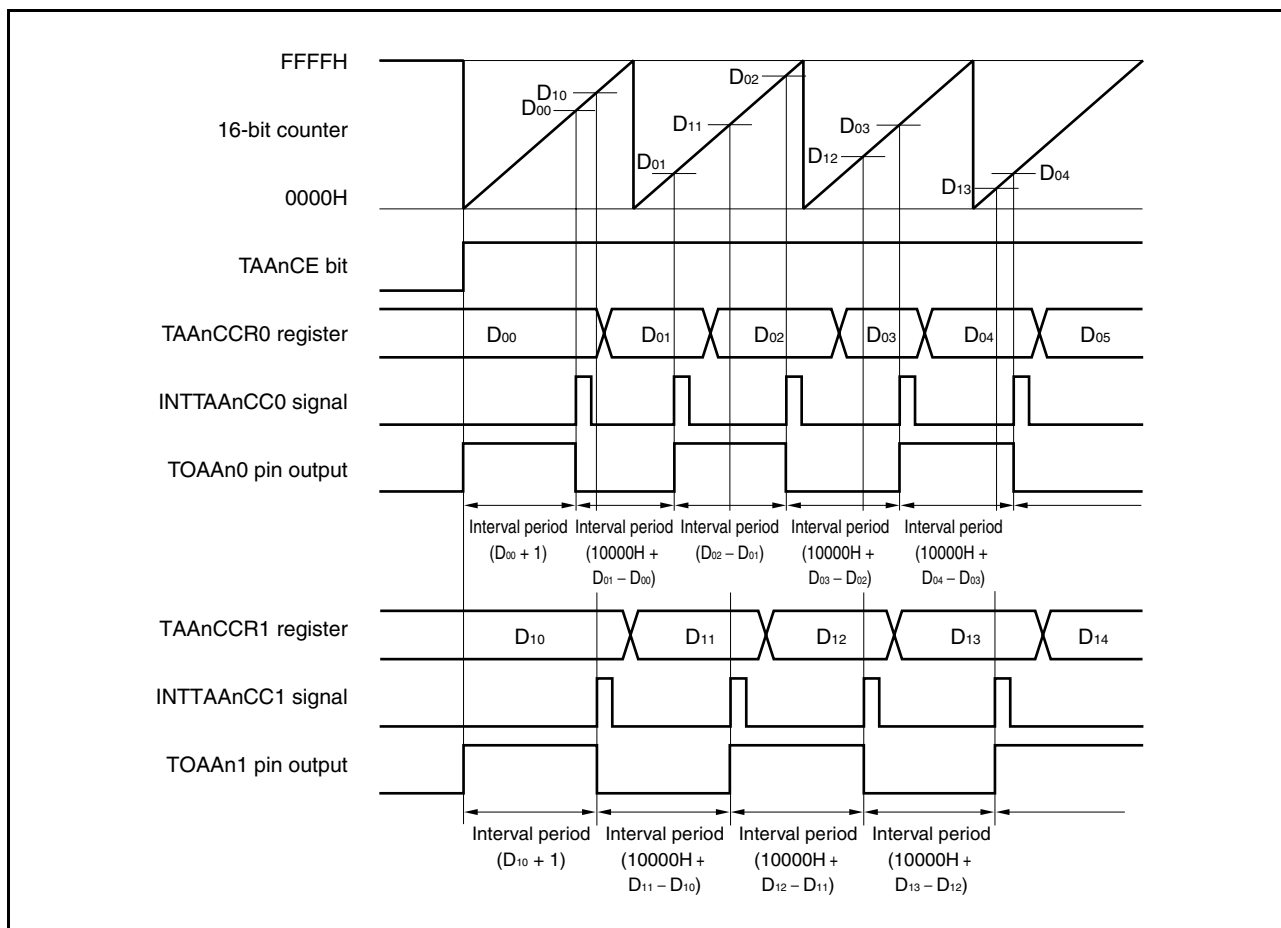


When performing an interval operation in the free-running timer mode, two intervals can be set with one channel.

To perform the interval operation, the value of the corresponding TAAnCCRm register must be re-set in the interrupt servicing that is executed when the INTTAAnCCm signal is detected.

The set value for re-setting the TAAnCCRm register can be calculated by the following expression, where "$D_m$" is the interval period.

    Compare register default value: $D_m - 1$

    Value set to compare register second and subsequent times: Previous set value + $D_m$

    (If the calculation result is greater than FFFFH, subtract 10000H from the result and set this value to the register.)

**Remark**    m = 0, 1

             n = 0 to 3, 5

**(b) Pulse width measurement with TAAnCCRm used as capture register**

When pulse width measurement is performed with the TAAnCCRm register used as a capture register, software processing is necessary for reading the capture register each time the INTTAAnCCm signal has been detected and for calculating the interval.
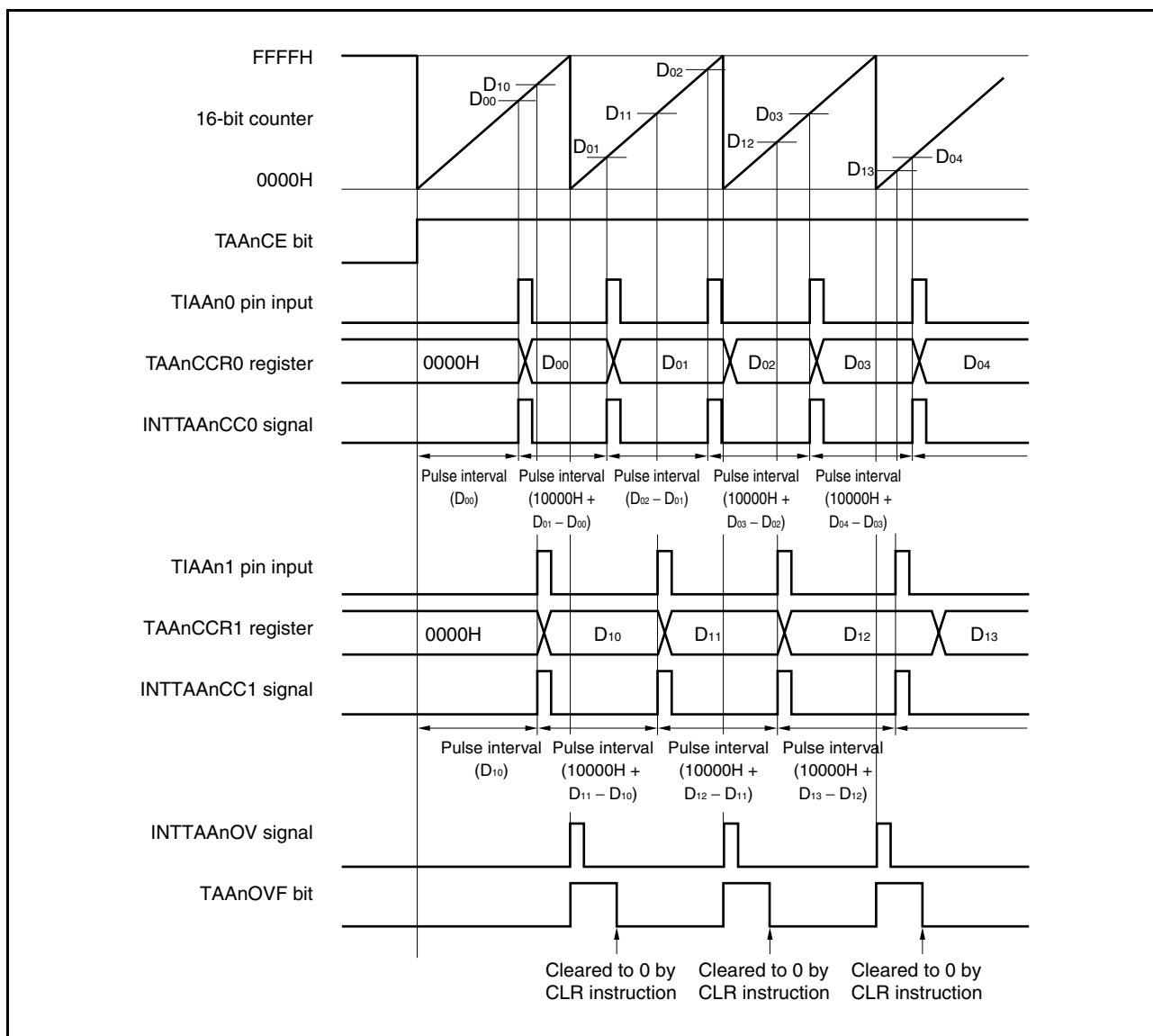


When executing pulse width measurement in the free-running timer mode, two pulse widths can be measured with one channel.

To measure a pulse width, the pulse width can be calculated by reading the value of the TAAnCCRm register in synchronization with the INTTAAnCCm signal, and calculating the difference between the read value and the previously read value.

**Remark** m = 0, 1

n = 0 to 3, 5

**(c) Processing of overflow when two capture registers are used**

Care must be exercised in processing the overflow flag when two capture registers are used. First, an example of incorrect processing is shown below.

**Example of incorrect processing when two capture registers are used**
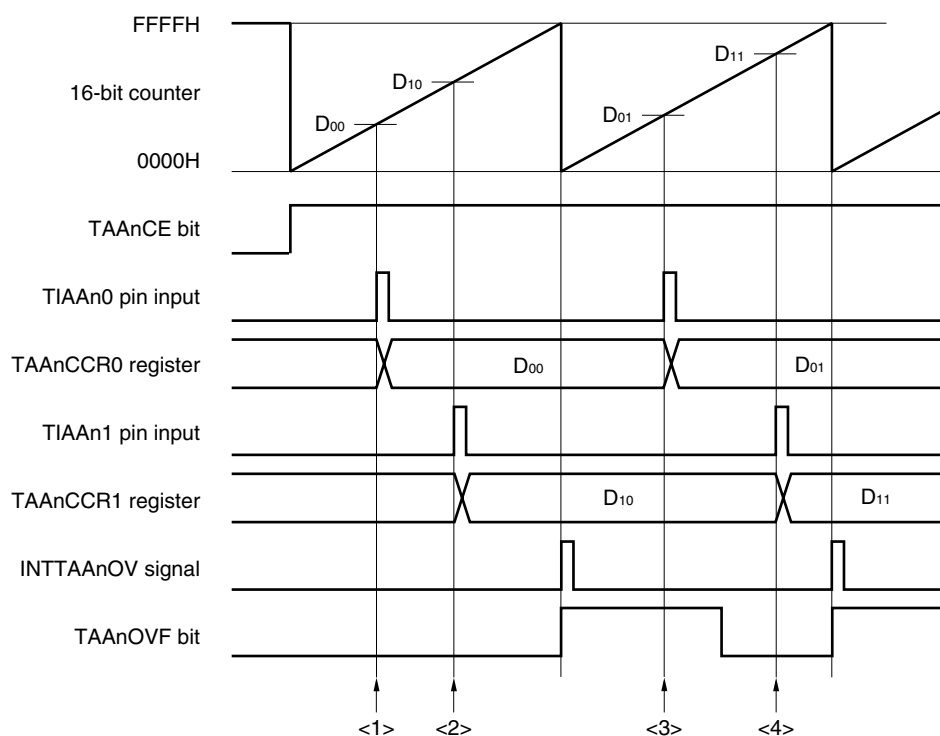


The following problem may occur when two pulse widths are measured in the free-running timer mode.

<1> Read the TAAnCCR0 register (setting of the default value of the TIAAn0 pin input).
<2> Read the TAAnCCR1 register (setting of the default value of the TIAAn1 pin input).
<3> Read the TAAnCCR0 register.
Read the overflow flag. If the overflow flag is 1, clear it to 0.
Because the overflow flag is 1, the pulse width can be calculated by ($10000H + D_{01} - D_{00}$).
<4> Read the TAAnCCR1 register.
Read the overflow flag. Because the flag is cleared in <3>, 0 is read.
Because the overflow flag is 0, the pulse width can be calculated by ($D_{11} - D_{10}$) (incorrect).

When two capture registers are used, and if the overflow flag is cleared to 0 by one capture register, the other capture register may not obtain the correct pulse width.

Use software when using two capture registers. An example of how to use software is shown below.
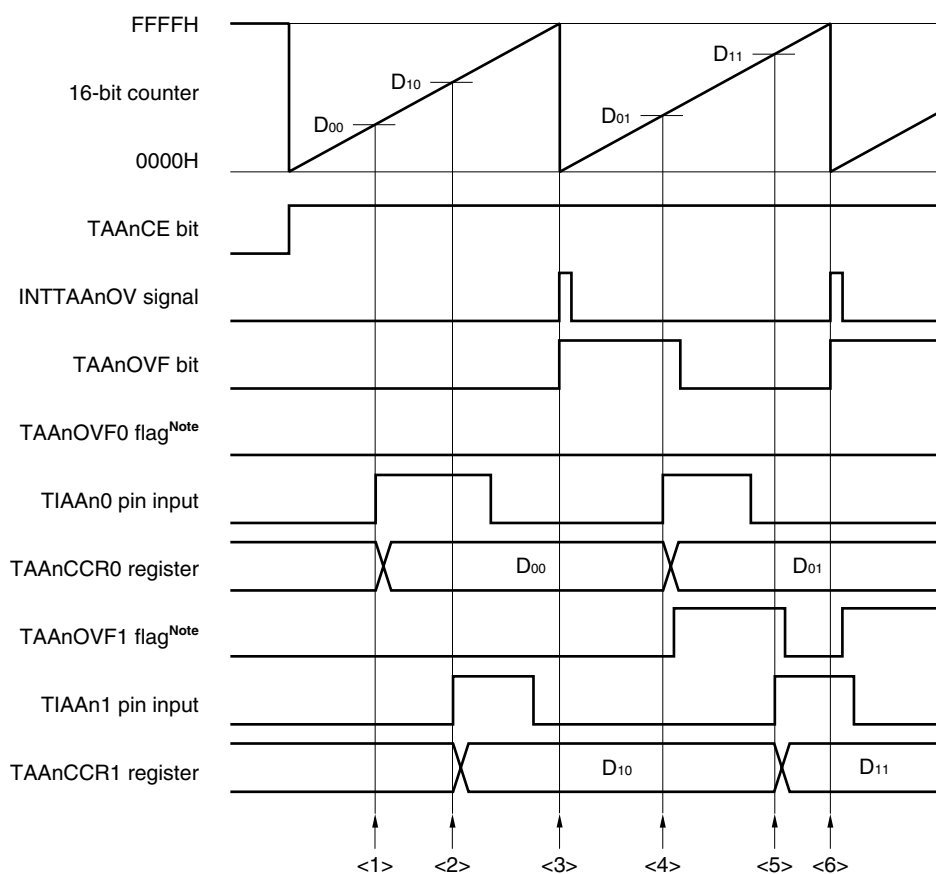
**Example when two capture registers are used (using overflow interrupt)**



**Note**    The TAAnOVF0 and TAAnOVF1 flags are set on the internal RAM by software.

<1> Read the TAAnCCR0 register (setting of the default value of the TIAAn0 pin input).

<2> Read the TAAnCCR1 register (setting of the default value of the TIAAn1 pin input).

<3> An overflow occurs.  Set the TAAnOVF0 and TAAnOVF1 flags to 1 in the overflow interrupt servicing, and clear the overflow flag to 0.

<4> Read the TAAnCCR0 register.

Read the TAAnOVF0 flag.  If the TAAnOVF0 flag is 1, clear it to 0.

Because the TAAnOVF0 flag is 1, the pulse width can be calculated by ($10000H + D_{01} - D_{00}$).

<5> Read the TAAnCCR1 register.

Read the TAAnOVF1 flag.  If the TAAnOVF1 flag is 1, clear it to 0 (the TAAnOVF0 flag is cleared in <4>, and the TAAnOVF1 flag remains 1).

Because the TAAnOVF1 flag is 1, the pulse width can be calculated by ($10000H + D_{11} - D_{10}$) (correct).

<6> Same as <3>

(2/2)

**Example when two capture registers are used (without using overflow interrupt)**



**Note**　The TAAnOVF0 and TAAnOVF1 flags are set on the internal RAM by software.

<1>　Read the TAAnCCR0 register (setting of the default value of the TIAAn0 pin input).

<2>　Read the TAAnCCR1 register (setting of the default value of the TIAAn1 pin input).

<3>　An overflow occurs. Nothing is done by software.

<4>　Read the TAAnCCR0 register.

　　　Read the overflow flag. If the overflow flag is 1, set only the TAAnOVF1 flag to 1, and clear the overflow flag to 0.

　　　Because the overflow flag is 1, the pulse width can be calculated by ($10000H + D_{01} - D_{00}$).

<5>　Read the TAAnCCR1 register.

　　　Read the overflow flag. Because the overflow flag is cleared in <4>, 0 is read.

　　　Read the TAAnOVF1 flag. If the TAAnOVF1 flag is 1, clear it to 0.

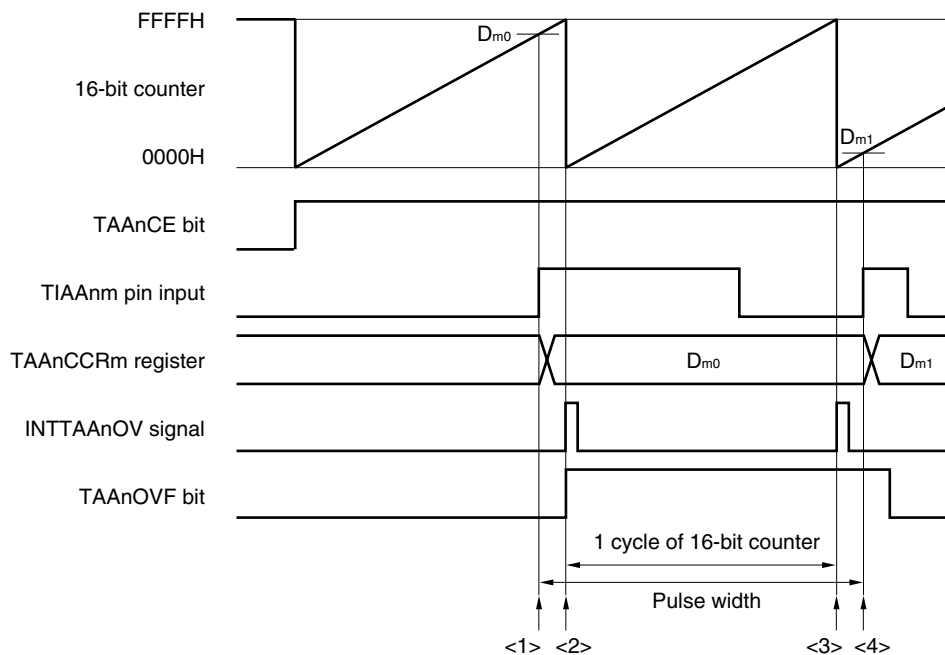　　　Because the TAAnOVF1 flag is 1, the pulse width can be calculated by ($10000H + D_{11} - D_{10}$) (correct).

<6>　Same as <3>

**Remark**　n = 0 to 3, 5

**(d) Processing of overflow if capture trigger interval is long**

If the pulse width is greater than one cycle of the 16-bit counter, care must be exercised because an overflow may occur more than once from the first capture trigger to the next. First, an example of incorrect processing is shown below.

---

**Example of incorrect processing when capture trigger interval is long**



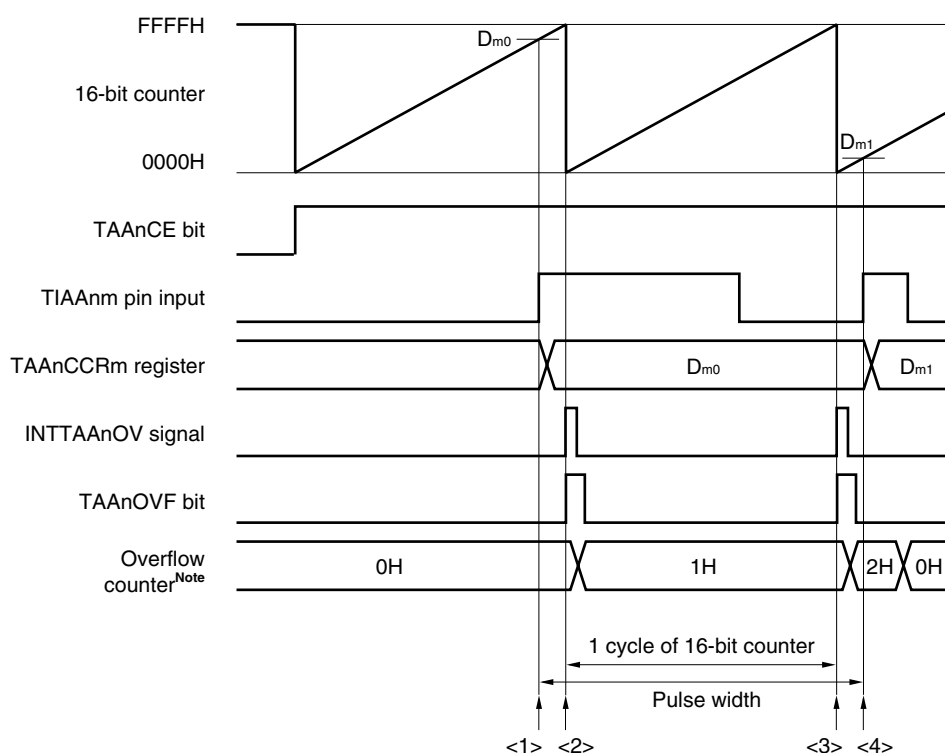The following problem may occur when a long pulse width is measured in the free-running timer mode.

<1> Read the TAAnCCRm register (setting of the default value of the TIAAnm pin input).
<2> An overflow occurs. Nothing is done by software.
<3> An overflow occurs a second time. Nothing is done by software.
<4> Read the TAAnCCRm register.
Read the overflow flag. If the overflow flag is 1, clear it to 0.
Because the overflow flag is 1, the pulse width can be calculated by $(10000H + D_{m1} - D_{m0})$ (incorrect).
Actually, the pulse width must be $(20000H + D_{m1} - D_{m0})$ because an overflow occurs twice.

---

If an overflow occurs twice or more when the capture trigger interval is long, the correct pulse width may not be obtained.

If the capture trigger interval is long, slow the count clock to lengthen one cycle of the 16-bit counter, or use software. An example of how to use software is shown next.

**Remark** m = 0, 1
n = 0 to 3, 5

**Example when capture trigger interval is long**



**Note** The overflow counter is set arbitrarily by software on the internal RAM.

<1> Read the TAAnCCRm register (setting of the default value of the TIAAnm pin input).

<2> An overflow occurs. Increment the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<3> An overflow occurs a second time. Increment (+1) the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<4> Read the TAAnCCRm register.

Read the overflow counter.

$\rightarrow$ When the overflow counter is "N", the pulse width can be calculated by (N $\times$ 10000H + $D_{m1}$ − $D_{m0}$).

In this example, the pulse width is (20000H + $D_{m1}$ − $D_{m0}$) because an overflow occurs twice.

Clear the overflow counter (0H).

**Remark** m = 0, 1

n = 0 to 3, 5

**(e) Clearing overflow flag**

The overflow flag can be cleared to 0 by clearing the TAAnOVF bit to 0 with the CLR instruction and by writing 8-bit data (bit 0 is 0) to the TAAnOPT0 register. To accurately detect an overflow, read the TAAnOVF bit when it is 1, and then clear the overflow flag by using a bit manipulation instruction.
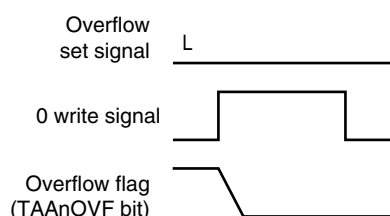


(i) Operation to write 0 (without conflict with setting)

(ii) Operation to write 0 (conflict with setting)
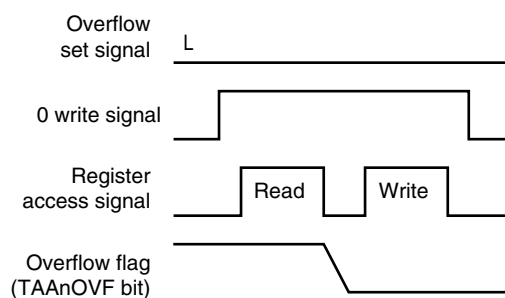
(iii) Operation to clear to 0 (without conflict with setting)
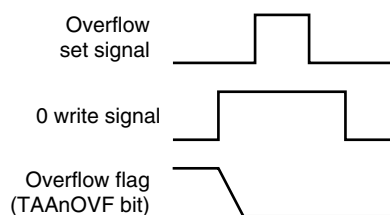
(iv) Operation to clear to 0 (conflict with setting)
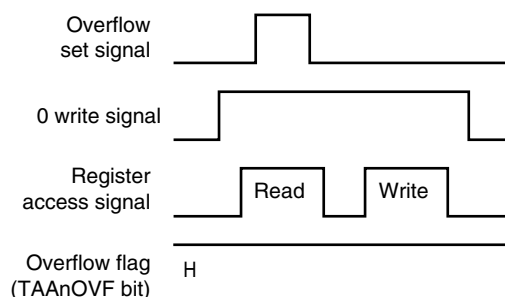
**Remark**   n = 0 to 3, 5

To clear the overflow flag to 0, read the overflow flag to check if it is set to 1, and clear it with the CLR instruction. If 0 is written to the overflow flag without checking if the flag is 1, the set information of the overflow may be erased by writing 0 ((ii) in the above chart). Therefore, software may judge that no overflow has occurred even when an overflow has actually occurred.

If execution of the CLR instruction conflicts with occurrence of an overflow when the overflow flag is cleared to 0 with the CLR instruction, the overflow flag remains set (1) even after execution of the clear instruction.

### 7.5.7   Pulse width measurement mode (TAAnMD2 to TAAnMD0 bits = 110)

In the pulse width measurement mode, 16-bit timer/event counter AA starts counting when the TAAnCTL0.TAAnCE bit is set to 1.  Each time the valid edge input to the TIAAnm pin has been detected, the count value of the 16-bit counter is stored in the TAAnCCRm register, and the 16-bit counter is cleared to 0000H.

The interval of the valid edge can be measured by reading the TAAnCCRm register after a capture interrupt request signal (INTTAAnCCm) occurs.

Select either the TIAAn0 or TIAAn1 pin as the capture trigger input pin.  Specify "No edge detection" for the unused pins by using the TAAnIOC1 register.

When an external clock is used as the count clock, measure the pulse width of the TIAAn1 pin because the external clock is fixed to the TIAAn0 pin.  At this time, clear the TAAnIOC1.TAAnIS1 and TAAnIOC1.TAAnIS0 bits to 00 (capture trigger input (TIAAn0 pin): No edge detection).

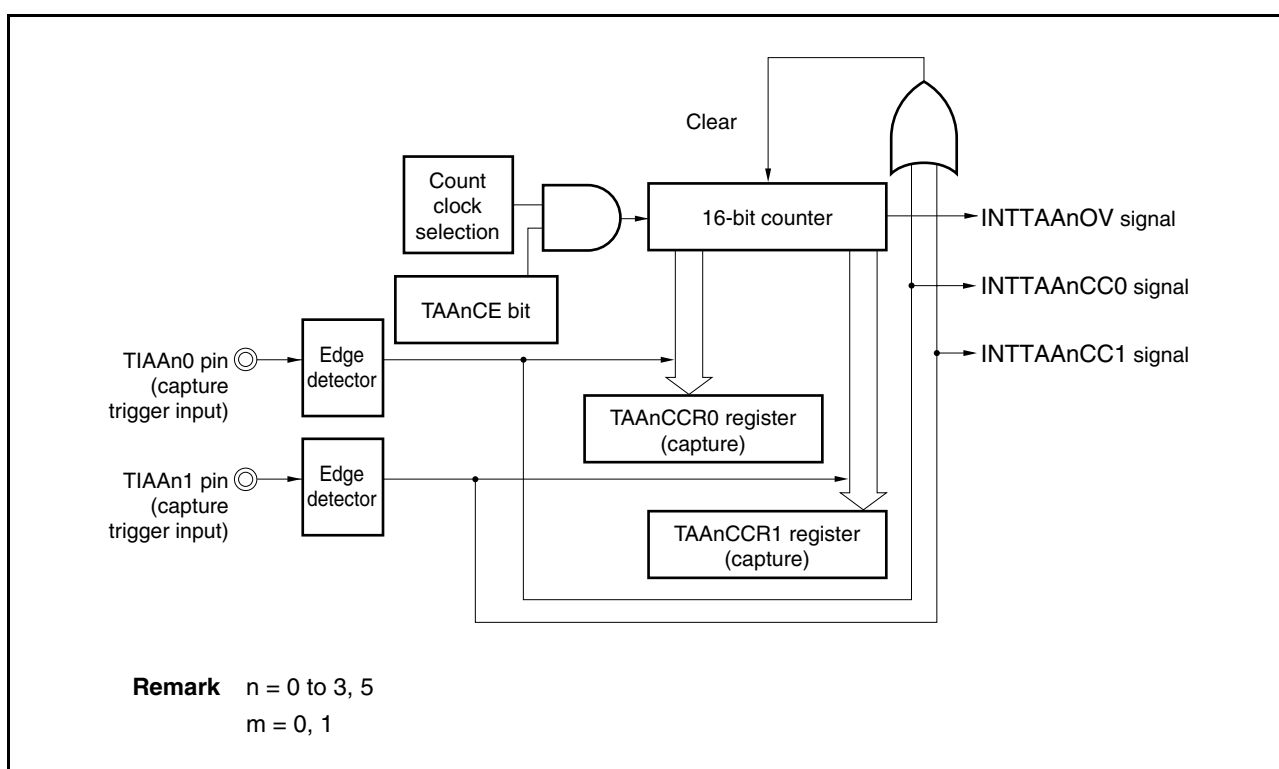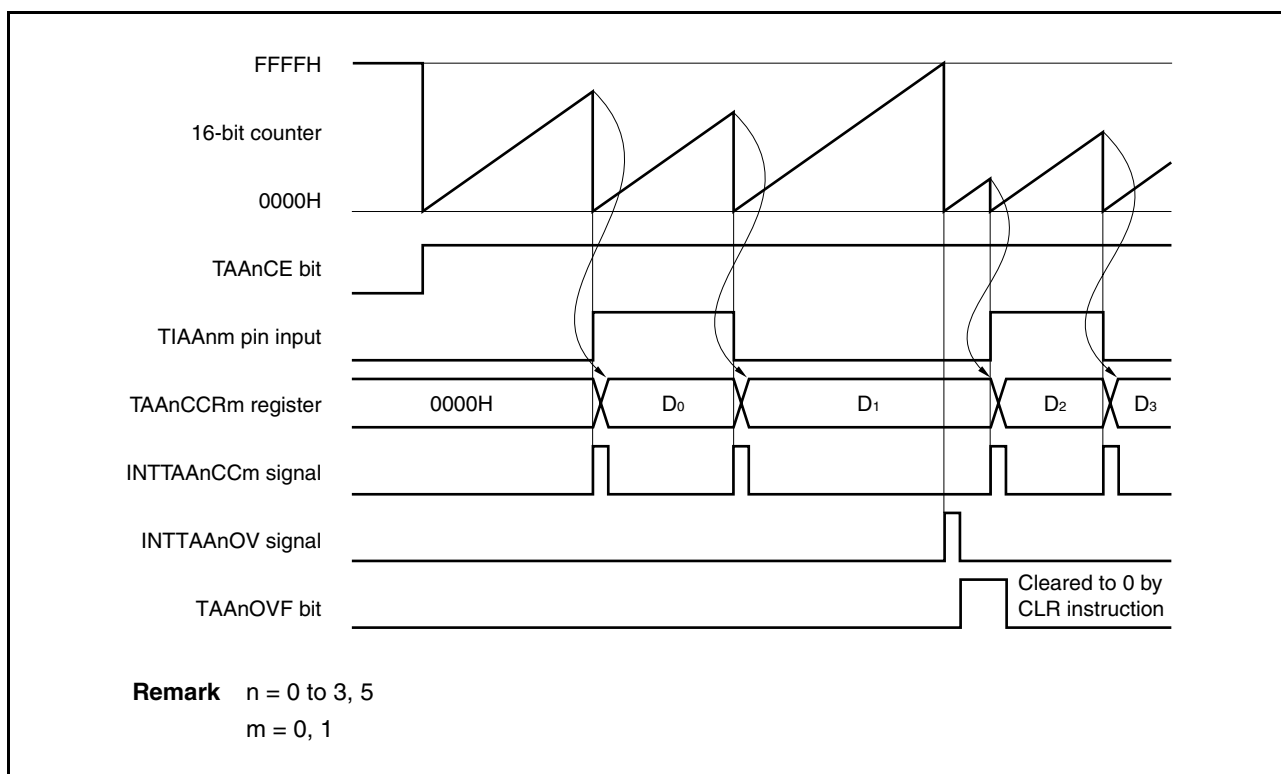#### Figure 7-39.  Configuration in Pulse Width Measurement Mode



**Remark**   n = 0 to 3, 5
            m = 0, 1

**Figure 7-40.  Basic Timing in Pulse Width Measurement Mode**



**Remark**   n = 0 to 3, 5
            m = 0, 1

When the TAAnCE bit is set to 1, the 16-bit counter starts counting.  When the valid edge input to the TIAAnm pin is later detected, the count value of the 16-bit counter is stored in the TAAnCCRm register, the 16-bit counter is cleared to 0000H, and a capture interrupt request signal (INTTAAnCCm) is generated.

The pulse width is calculated as follows.

     Pulse width = Captured value $\times$ Count clock cycle

If the valid edge is not input to the TIAAnm pin even when the 16-bit counter counted up to FFFFH, an overflow interrupt request signal (INTTAAnOV) is generated at the next count clock, and the counter is cleared to 0000H and continues counting.  At this time, the overflow flag (TAAnOPT0.TAAnOVF bit) is also set to 1.  Clear the overflow flag to 0 by executing the CLR instruction via software.

If the overflow flag is set to 1, the pulse width can be calculated as follows.

     Pulse width = (10000H $\times$ TAAnOVF bit set (1) count + Captured value) $\times$ Count clock cycle

**Remark**   n = 0 to 3, 5
            m = 0, 1

**Figure 7-41.  Register Setting in Pulse Width Measurement Mode (1/2)**



**(a)  TAAn control register 0 (TAAnCTL0)**

TAAnCE ... TAAnCKS2 TAAnCKS1 TAAnCKS0

TAAnCTL0 | 0/1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 |

Select count clock

0: Stops counting
1: Enables counting

**(b)  TAAn control register 1 (TAAnCTL1)**

TAAnEST TAAnEEE ... TAAnMD2 TAAnMD1 TAAnMD0

TAAnCTL1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

1, 1, 0:
Pulse width measurement
mode

**(c)  TAAn I/O control register 1 (TAAnIOC1)**

TAAnIS3  TAAnIS2  TAAnIS1  TAAnIS0

TAAnIOC1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |

Select valid edge
of TIAAn0 pin input

Select valid edge
of TIAAn1 pin input
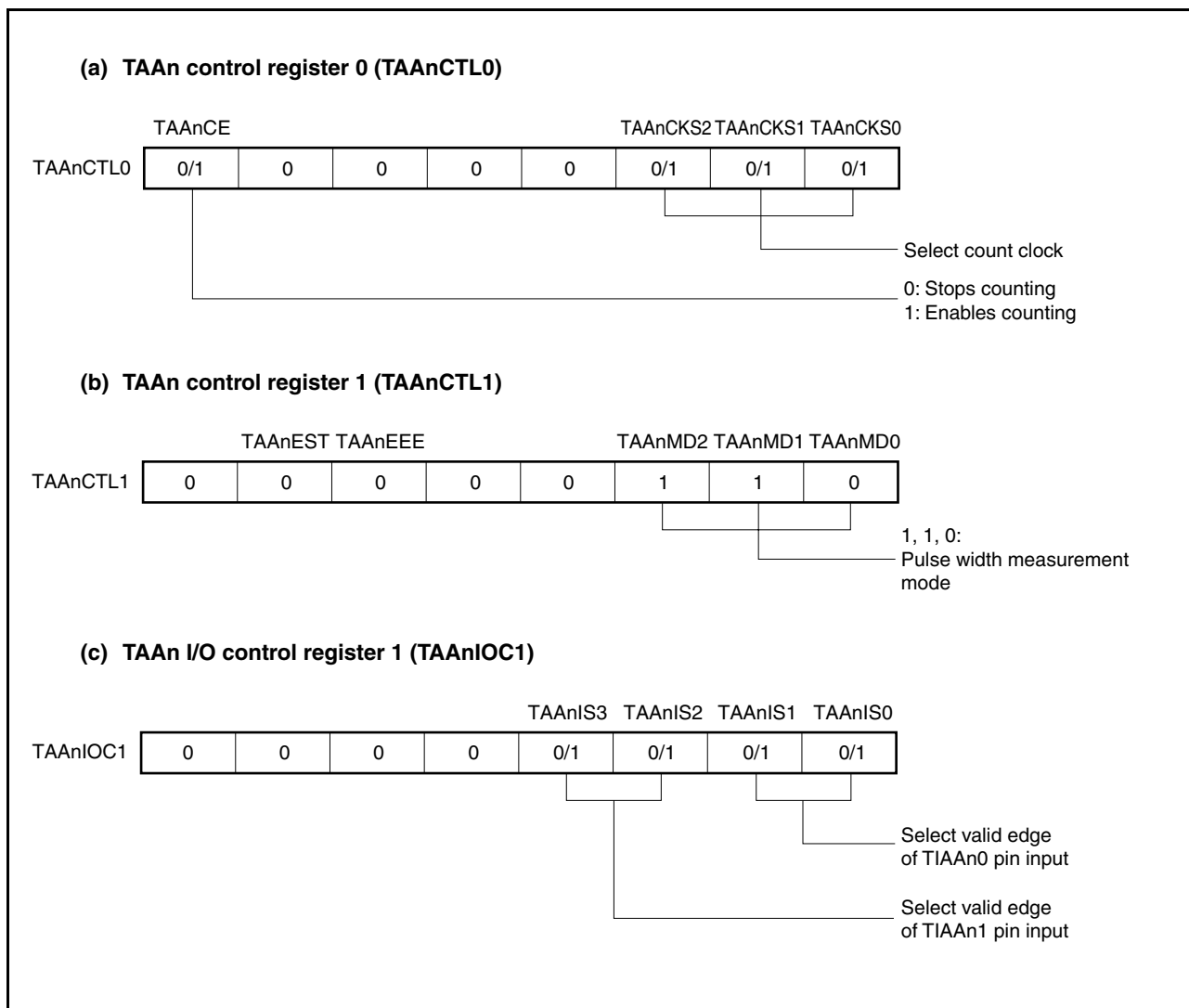
**Figure 7-41.  Register Setting in Pulse Width Measurement Mode (2/2)**

**(d)  TAAn option register 0 (TAAnOPT0)**
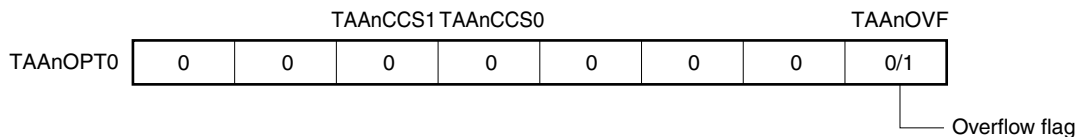
|           |   | TAAnCCS1 | TAAnCCS0 |   |   |   | TAAnOVF |
|-----------|---|----------|----------|---|---|---|---------|
| TAAnOPT0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 |

Overflow flag

**(e)  TAAn counter read buffer register (TAAnCNT)**

The value of the 16-bit counter can be read by reading the TAAnCNT register.

**(f)  TAAn capture/compare registers 0 and 1 (TAAnCCR0 and TAAnCCR1)**

These registers store the count value of the 16-bit counter when the valid edge input to the TIAAnm pin is detected.

**Remarks  1.** TAAn I/O control register 0 (TAAnIOC0), and TAAn I/O control register 2 (TAAnIOC2) are not used in the pulse width measurement mode.
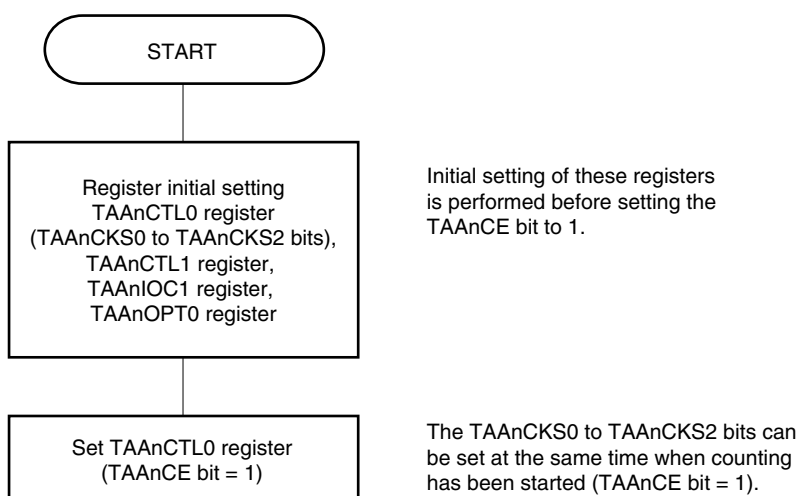
**2.** m = 0, 1
n = 0 to 3, 5

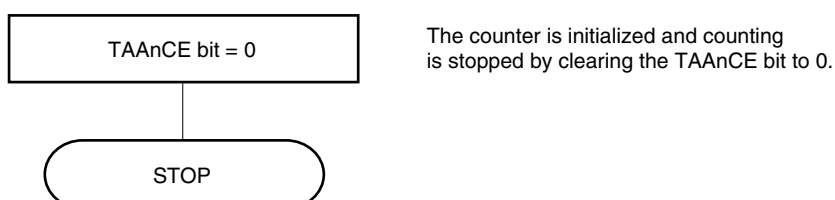**(1) Operation flow in pulse width measurement mode**

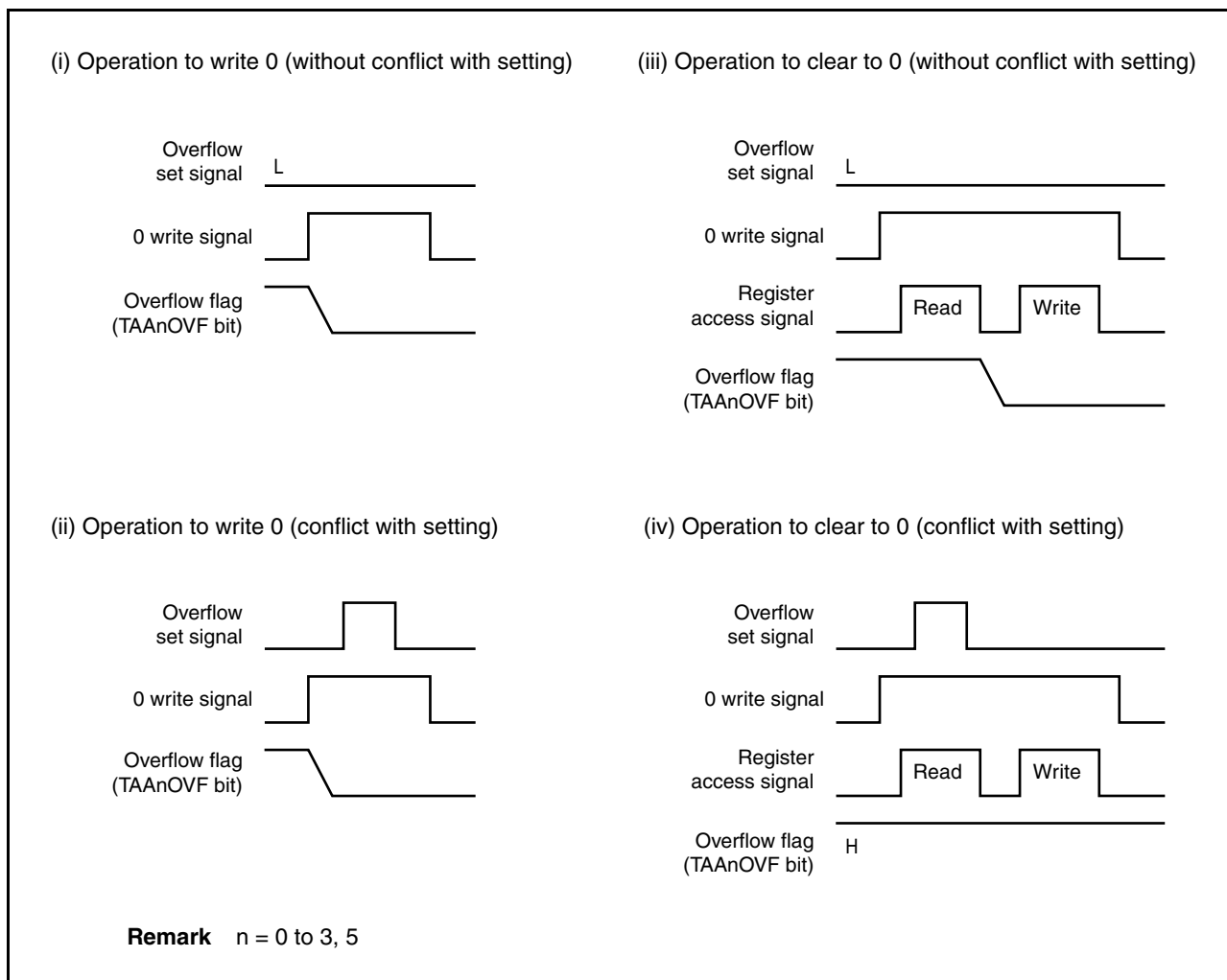**Figure 7-42. Software Processing Flow in Pulse Width Measurement Mode**



**Remark**    n = 0 to 3, 5

**(2)  Operation timing in pulse width measurement mode**

**(a)  Clearing overflow flag**

The overflow flag can be cleared to 0 by clearing the TAAnOVF bit to 0 with the CLR instruction and by writing 8-bit data (bit 0 is 0) to the TAAnOPT0 register.  To accurately detect an overflow, read the TAAnOVF bit when it is 1, and then clear the overflow flag by using a bit manipulation instruction.



(i) Operation to write 0 (without conflict with setting)

Overflow set signal

0 write signal

Overflow flag (TAAnOVF bit)

(ii) Operation to write 0 (conflict with setting)

Overflow set signal

0 write signal

Overflow flag (TAAnOVF bit)

(iii) Operation to clear to 0 (without conflict with setting)

Overflow set signal

0 write signal

Register access signal      Read      Write

Overflow flag (TAAnOVF bit)

(iv) Operation to clear to 0 (conflict with setting)

Overflow set signal

0 write signal

Register access signal      Read      Write

Overflow flag (TAAnOVF bit)      H

**Remark**   n = 0 to 3, 5

To clear the overflow flag to 0, read the overflow flag to check if it is set to 1, and clear it with the CLR instruction.  If 0 is written to the overflow flag without checking if the flag is 1, the set information of the overflow may be erased by writing 0 ((ii) in the above chart).  Therefore, software may judge that no overflow has occurred even when an overflow has actually occurred.

If execution of the CLR instruction conflicts with occurrence of an overflow when the overflow flag is cleared to 0 with the CLR instruction, the overflow flag remains set (1) even after execution of the clear instruction.

#### 7.5.8  Timer output operations

The following table shows the operations and output levels of the TOAAn0 and TOAAn1 pins.

**Table 7-5.  Timer Output Control in Each Mode**

| Operation Mode | TOAAn1 Pin | TOAAn0 Pin |
|---|---|---|
| Interval timer mode | Square wave output | |
| External event count mode | Square wave output | – |
| External trigger pulse output mode | External trigger pulse output | Square wave output |
| One-shot pulse output mode | One-shot pulse output | |
| PWM output mode | PWM output | |
| Free-running timer mode | Square wave output (only when compare function is used) | |
| Pulse width measurement mode | – | |

**Remark**    n = 0 to 3, 5

**Table 7-6.  Truth Table of TOAAn0 and TOAAn1 Pins Under Control of Timer Output Control Bits**

| TAAnIOC0.TAAnOLm Bit | TAAnIOC0.TAAnOEm Bit | TAAnCTL0.TAAnCE Bit | Level of TOAAnm Pin |
|---|---|---|---|
| 0 | 0 | × | Low-level output |
| | 1 | 0 | Low-level output |
| | | 1 | Low level immediately before counting, high level after counting is started |
| 1 | 0 | × | High-level output |
| | 1 | 0 | High-level output |
| | | 1 | High level immediately before counting, low level after counting is started |

**Remark**    n = 0 to 3, 5

m = 0, 1

## 7.6 Timer-Tuned Operation Function

Timer AA and timer AB have a timer-tuned operation function.

The timer-tuned operation function is used to tune the internal timers of the V850ES/JG3-U and V850ES/JH3-U, so that the number of capture or compare registers of the slave timer (the number of timer outputs and the number of compare match interrupts of the slave timer) can be added to the master timer. The timers that can be tuned are listed in Table 7-7.

**Table 7-7. Tuned-Operation Mode of Timers**

| Master Timer | Slave Timer |
| --- | --- |
| TAA1 | TAA0 |
| TAA3 | TAA2 |
| TAB0 | TAA5 |

The tuned-operation function has the following modes.

- PWM output mode
- Free-running timer mode

Figure 7-43 shows an example where individual operation and tuned operation of TAA0 (as the master timer) and TAA1 (as the slave timer) are performed in PWM output mode.

**Figure 7-43. Differences Between Individual Operation and Tuned Operation Using TAA0 and TAA1**
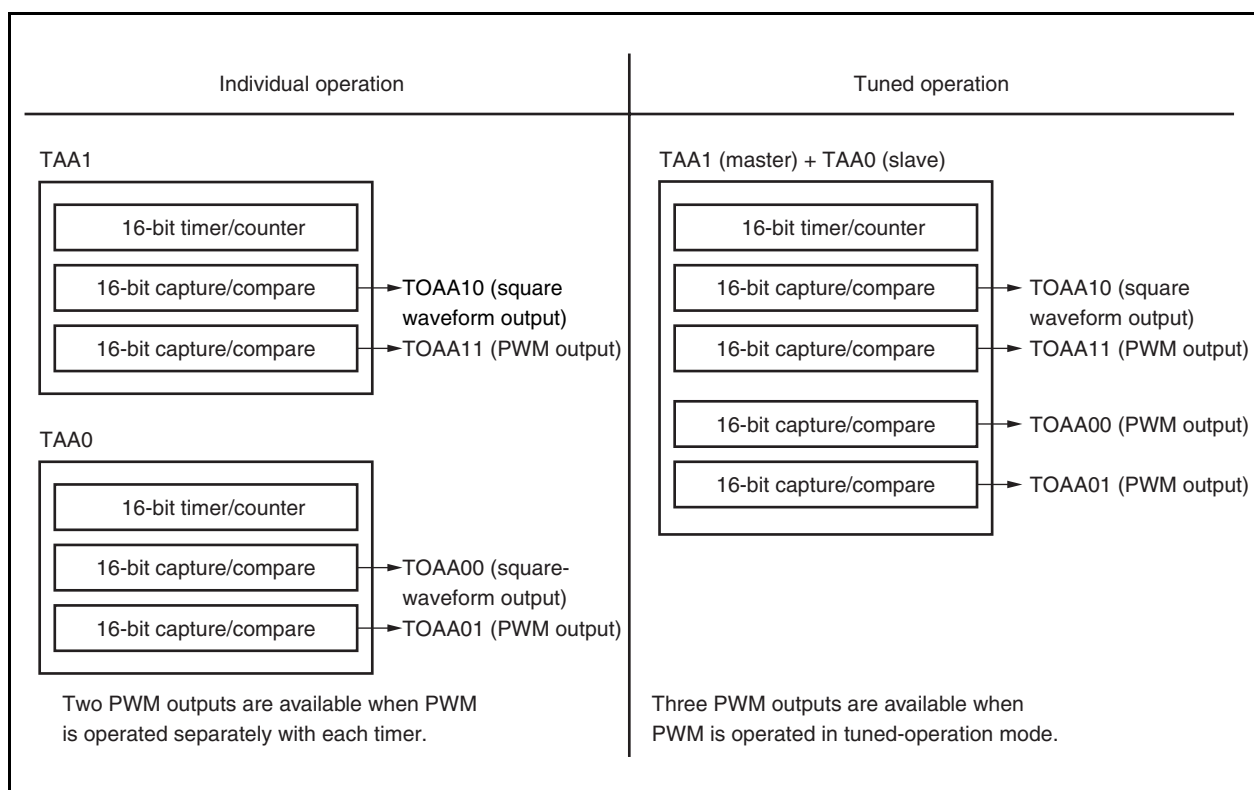
Table 7-8 show the timer modes that can be used in the tuned-operation mode and Table 7-9 shows the differences of the timer output functions between individual operation and tuned operation (√: Settable, ×: Not settable).

**Table 7-8. Timer Modes Usable in Tuned-Operation Mode**

| Master Timer | Slave Timer | Free-Running Timer Mode | PWM Mode |
|---|---|---|---|
| TAA1 | TAA0 | √ | √ |
| TAA3 | TAA2 | √ | √ |
| TAB0 | TAA5 | √ | √ |

**Table 7-9. Timer Output Functions**

| Tuned Channel | Timer | Pin | Free-Running Timer Mode | | PWM Mode | |
|---|---|---|---|---|---|---|
| | | | Individual Operation | Tuned Operation | Individual Operation | Tuned Operation |
| Ch0 | TAA1 (master) | TOAA10 | PPG | ← | Toggle | ← |
| | | TOAA11 | PPG | ← | PWM | ← |
| | TAA0 (slave) | TOAA00 | PGP | ← | Toggle | PWM |
| | | TOAA01 | PPG | ← | PWM | ← |
| Ch1 | TAA3 (master) | TOAA30 | PPG | ← | Toggle | ← |
| | | TOAA31 | PPG | ← | PWM | ← |
| | TAA2 (slave) | TOAA20 | PPG | ← | Toggle | PWM |
| | | TOAA21 | PPG | ← | PWM | ← |
| Ch2 | TAB0 (master) | TOAB00 | PPG | ← | Toggle | ← |
| | | TOAB01 to TOAB03 | PPG | ← | PWM | ← |
| | TAA5 (slave) | TOAA50 | PPG | ← | Toggle | PWM |
| | | TOAA51 | PPG | ← | PWM | ← |

**Remark**   The timing of transmitting data from the compare register of the master timer to the compare register of the slave timer is as follows.

- PPG:                                    CPU write timing
- Toggle, PWM, triangular wave PWM: Timing at which timer counter and compare register match TOAAn0 and TOABm0

**7.6.1    Free-running timer mode (during timer-tuned operation)**

   This section explains the free-running timer mode of the timer-tuned operation.  For the combination of timer-tuned operations, see **Table 7-7**.  In this section, an example of timer-tuned operation using TAA1 and TAA0 is shown.

   **(i)  Selecting capture/compare registers**

   When the free-running timer mode of the timer-tuned operation is used with TAA1 and TAA0 connected to each other, the two capture/compare registers of TAA1 and two capture/compare registers of TAA0 can be used in combination.

   How the capture and compare registers are combined is not restricted and can be selected by using the TAAnCCSn bit of the master or slave timer.  When the compare register is selected, the set value of the compare register can be rewritten during operation and the rewriting method is anytime write (n = 0, 1).

   **(ii) Overflow**

   If the counter overflows, an overflow interrupt (INTTAA1OV) of the master timer is generated and the overflow flag (TAA1OVF) is set to "1".

   The overflow interrupt (INTTAA0OV) and overflow flag (TAA0OVF) of the slave timer do not operate and are always at the low level.

**(1)  Settings in free-running timer mode (compare function)**

**[Initial settings]**

Master timer: TAA1CTL0.TAA1CE = 0 (operation disabled)

Slave timer: TAA0CTL0.TAA0CE = 0 (operation disabled)

**[Initial settings of master timer (TAA1)]**
- TAA1CTL1.TAA1MD2 to TAA1CTL1.TAA1MD0 = 101 (setting of free-running timer mode)
- TAA1OPT0.TAA1CCS1 and TAA1OPT0.TAA1CCS0 = 00 (setting of capture/compare select bit to "compare".)
- TAA1CTL1.TAA1CKS2 to TAA1CTL1.TAA1CKS0 (setting of count clock (any))
- TAA1CCR1 and TAA1CCR0 registers are set.

**[Initial settings of slave timer (TAA0)]**
- TAA0CTL1.TAA0SYE = 1 (setting of timer-tuned operation)
- TAA0CTL1.TAA0MD2 to TAA0CTL1.TAA0MD0 = 101 (setting of free-running timer mode)
- TAA0OPT0.TAA0CCS1 and TAA0OPT0.TAA0CCS0 = 00 (setting of capture/compare select bit to "compare".)
- TAA0CCR0 and TAA0CCR1 registers are set.

**Remark**   The initial settings of the master timer and slave timer may be performed in any order.
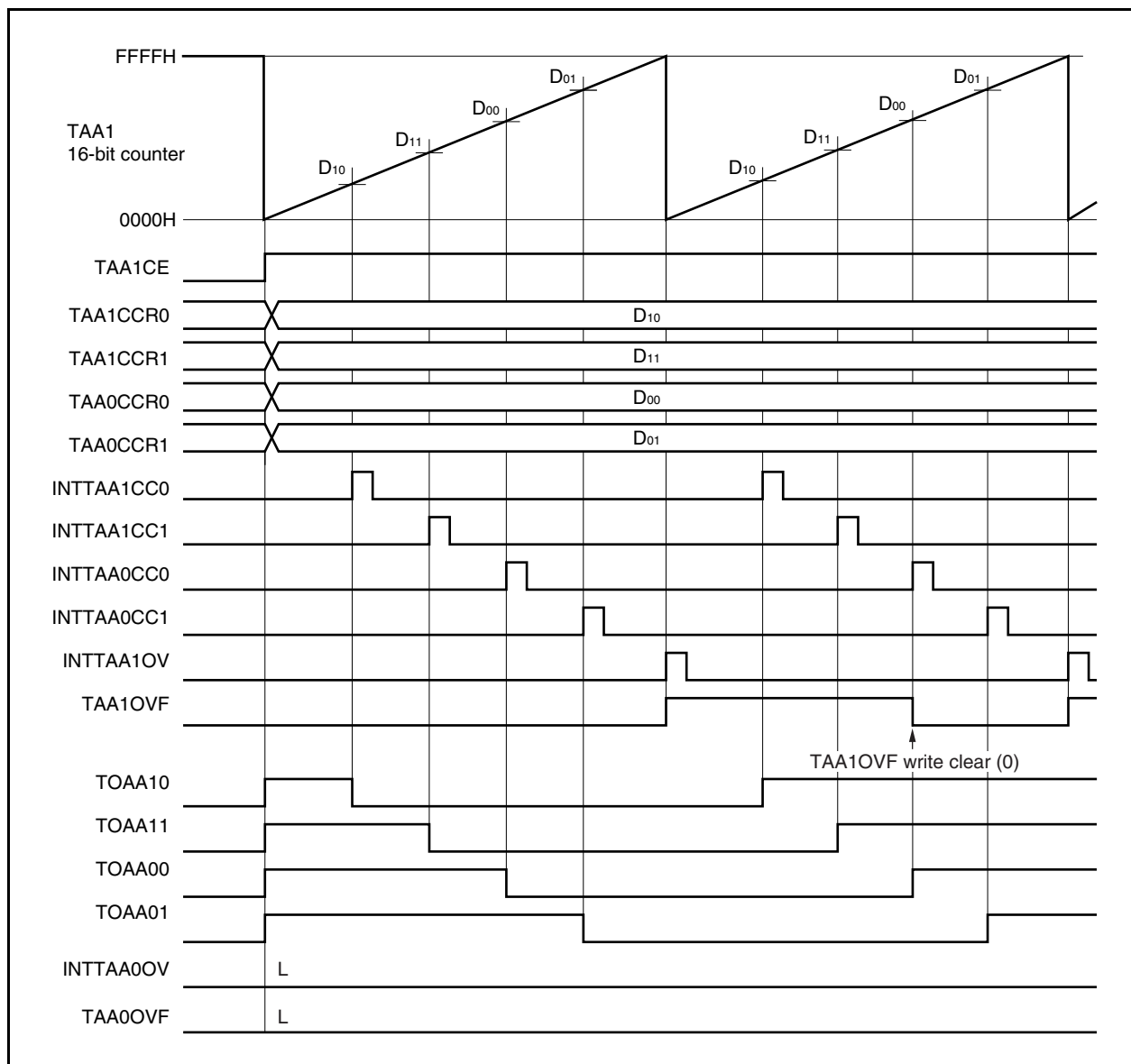
**[Starting counting]**
<1>  Set TAA1CTL0.TAA1CE of the master timer to 1.
<2>  Start counting.
<3>  Changing the setting of the register during operation
- The compare register can be rewritten (anytime write).

**[End condition]**
- Set TAA1CTL0.TAA1CE of the master timer to 0.

**Figure 7-44. Example of Timing in Free-Running Mode (Compare Function)**

**(2)  Settings in free-running timer mode (capture function)**

**[Initial settings]**

Master timer: TAA1CTL0.TAA1CE = 0 (operation disabled)

Slave timer: TAA0CTL0.TAA0CE = 0 (operation disabled)

**[Initial settings of master timer (TAA1)]**
- TAA1CTL1.TAA1MD2 to TAA1CTL1.TAA1MD0 = 101 (setting of free-running timer mode)
- TAA1OPT0.TAA1CCS1 and TAA1OPT0.TAA1CCS0 = 11 (setting of capture/compare select bit to "capture".)
- TAA1CTL1.TAA1CKS2 to TAA1CTL1.TAA1CKS0 (setting of count clock (any))
- TAA1IOC1.TAA1IS3 to TAA1IOC1.TAA1IS0 (specification of valid edge of capture trigger)

**[Initial settings of slave timer (TAA0)]**
- TAA0CTL1.TAA0SYE = 1 (setting of timer-tuned operation)
- TAA0CTL1.TAA0MD2 to TAA0CTL1.TAA0MD0 = 101 (setting of free-running timer mode)
- TAA0OPT0.TAA0CCS1 and TAA0OPT0.TAA0CCS0 = 11 (setting of capture/compare select bit to "capture".)
- TAA0IOC1.TAA0IS3 to TAA0IOC1.TAA0IS0 (specification of valid edge of capture trigger)

**Remark**    The initial settings of the master timer and slave timer may be performed in any order.
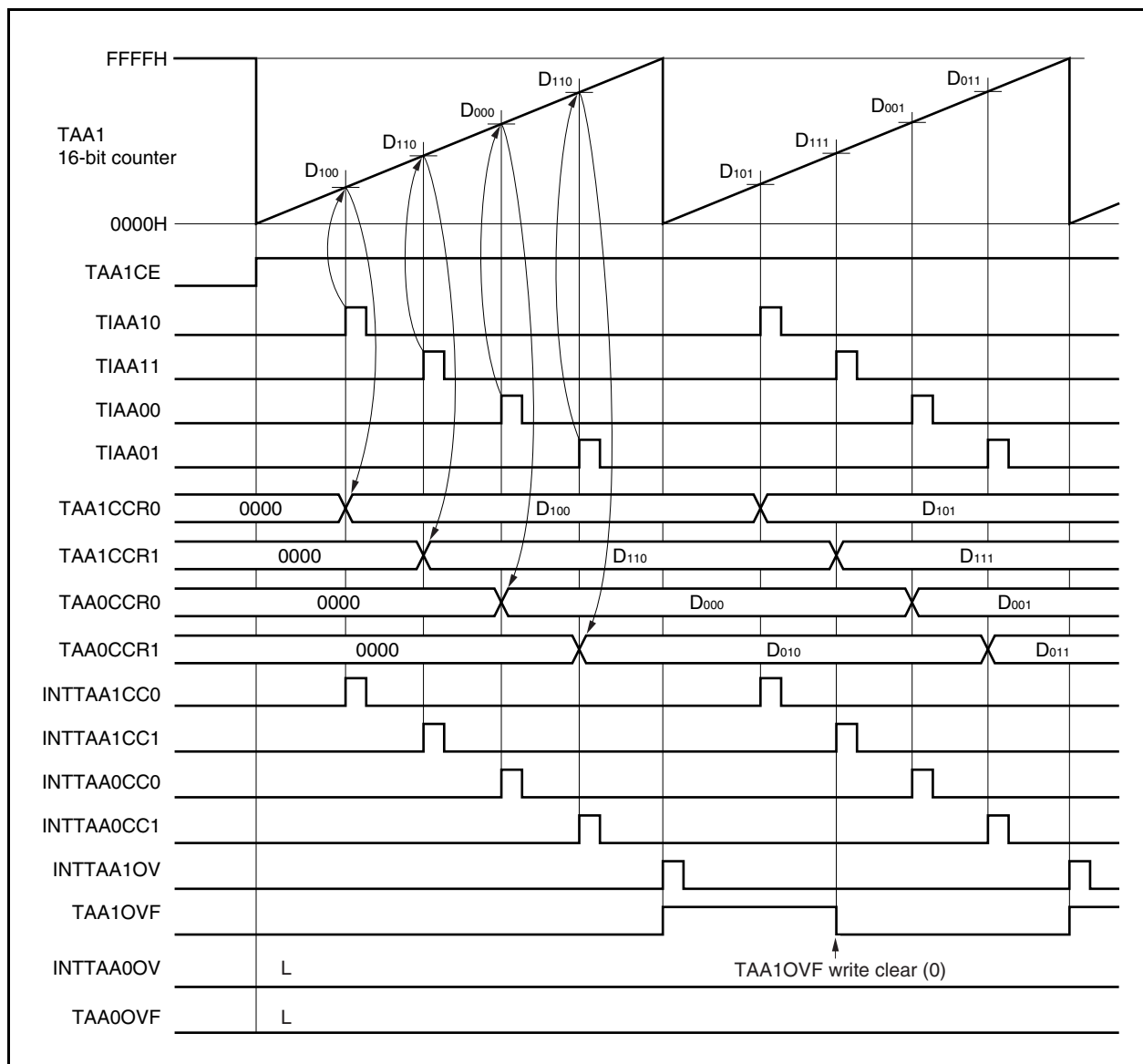
**[Starting counting]**

<1>  Set TAA1CTL0.TAA1CE of the master timer to 1.

<2>  Start counting.

**[End condition]**
- Set TAA1CTL0.TAA1CE of the master timer to 0.

**Figure 7-45.  Example of Timing in Free-Running Mode (Capture Function)**

**(3) Settings in free-running timer mode (capture/compare used together)**

An example of using TAA0 as a capture register and TAA1 as a compare register is shown below.

**[Initial settings]**

Master timer: TAA1CTL0.TAA1CE = 0 (operation disabled)

Slave timer: TAA0CTL0.TAA0CE = 0 (operation disabled)

**[Initial settings of master timer (TAA1)]**

- TAA1CTL1.TAA1MD2 to TAA1CTL1.TAA1MD0 = 101 (setting of free-running timer mode)
- TAA1OPT0.TAA1CCS1 and TAA1OPT0.TAA1CCS0 = 11 (setting of capture/compare select bit to "capture".)
- TAA1CTL1.TAA1CKS2 to TAA1CTL1.TAA1CKS0 (setting of count clock (any))
- TAA1.TAA0IS3 to TAA1.TAA1IS0 (specification of valid edge of capture trigger)

**[Initial settings of slave timer (TAA0)]**

- TAA0CTL1.TAA0SYE = 1 (setting of timer-tuned operation)
- TAA0CTL1.TAA0MD2 to TAA0CTL1.TAA0MD0 = 101 (setting of free-running timer mode)
- TAA0OPT0.TAA0CCS1 and TAA0OPT0.TAA0CCS0 = 00 (setting of capture/compare select bit to "compare".)
- TAA0CCR0 and TAA0CCR1 registers are set.

**Remark** The initial settings of the master timer and slave timer may be performed in any order.

**[Starting counting]**

&lt;1&gt; Set TAA1CTL0.TAA1CE of the master timer to 1.

&lt;2&gt; Start counting.

**[End condition]**

- Set TAA1CTL0.TAA1CE of the master timer to 0.

**Figure 7-46. Example of Timing in Free-Running Mode (Capture/Compare Used Together)**
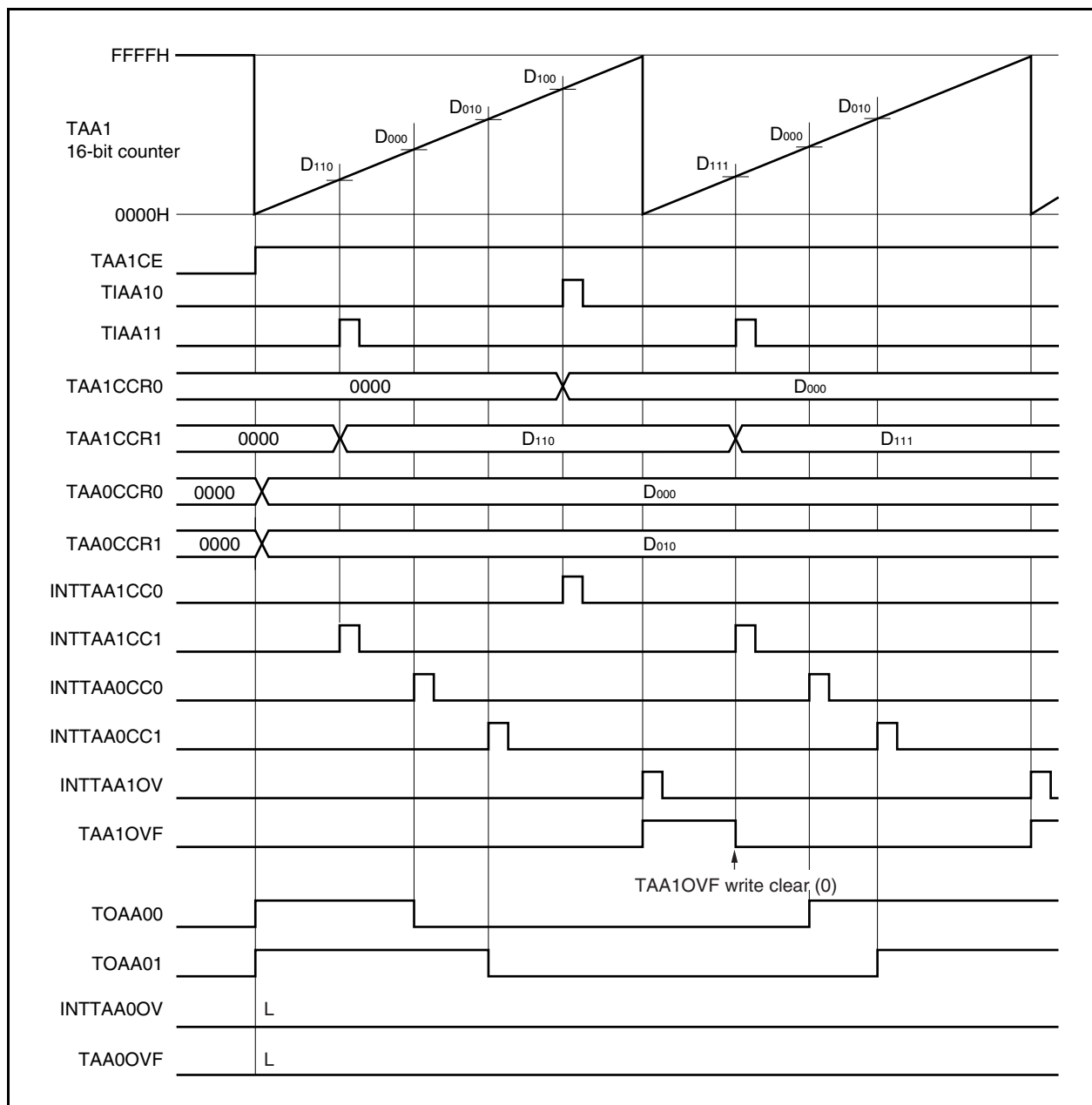
**7.6.2   PWM output mode (during timer-tuned operation)**

This section explains the PWM output mode of timer-tuned operation.  For combinations of timer-tuned operations, see **Table 7-7**.  This section presents an example of a timer-tuned operation with TAB0 and TAA5.

The TAB0CCR0 register of the master timer (TAB0) is used as a compare register for cycle, and the TAB0CCR1, TAB0CCR2, and TAB0CCR3 registers of the master timer (TAB0) and the TAA5CCR0 and TAA5CCR1 registers of the slave timer (TAA5) are used as compare registers for duty.

The compare registers can be rewritten during operation and the rewriting method is batch writing.

Batch writing is enabled when the TAB0CCR1 register of the master timer (TAB0) is written, and all the compare registers of the master and slave timers are rewritten or the same value is written to them when an interrupt, which is generated if the value of the TAB0CCR0 register of the master timer (TAB0) matches the value of the timer counter, is generated.

**(1)  Settings in PWM output mode**

  **[Initials setting]**
    Master timer: TAB0CTL0.TAB0CE = 0 (operation disabled)
    Slave timer: TAA5CTL0.TAA5CE = 0 (operation disabled)

  **[Initial settings of master timer (TAB0)]**
    - TAB0CTL1.TAB0MD2 to TAB0CTL1.TAB0MD0 = 100 (setting of PWM output mode)
    - TAB0OPT0.TAB0CCS3 to  TAB0OPT0.TAB0CCS0 = 0000 (setting of capture/compare select bit to "compare".)
    - TAB0CCR0, TAB0CCR1, TAB0CCR2, and TAB0CCR3 registers are set.

  **[Initial settings of slave timer (TAA5)]**
    - TAA5CTL1.TAA5SYE = 1 (setting of timer-tuned operation)
    - TAA5CTL1.TAA5MD2 to TAA5CTL1.TAA5MD0 = 101 (setting of free-running timer mode)
    - TAA5OPT0.TAA5CCS1  and  TAA5OPT0.TAA5CCS0  =  00  (setting of capture/compare select bit to "compare".)
    - TAA5CCR0 and TAA5CCR1 registers are set.

  **Remark**   The initial settings of the master timer and slave timer may be performed in any order.

  **[Starting counting]**
    <1>  Set TAB0CTL0.TAB0CE of the master timer to 1.
    <2>  Start counting.
    <3>  Changing the setting of the register during operation
        - The compare register can be rewritten (batch rewrite).

  **[End condition]**
    - Set TAB0CTL0.TAB0CE of the master timer to 0.

**[Batch write]**

In the PWM output mode, the next batch write is enabled by writing the TAB0CCR1 register of the master timer (TAB0). After all the compare registers that must be rewritten have been rewritten, therefore, the TAB0CCR1 register of the master timer (TAB0) must be written.

Batch writing is executed when the value of the timer counter matches the value of the compare register for cycle (TAB0CCR0).

If the TAB0CCR1 register of the master timer (TAB0) is not written, batch writing is not enabled even if another compare register is rewritten. Consequently, the value of the compare registers is not rewritten even when the value of the timer counter matches the value of the compare register for cycle (TAB0CCR0).

**Figure 7-47. Timing Example of Tuned PWM Function (TAB0, TAA5)**
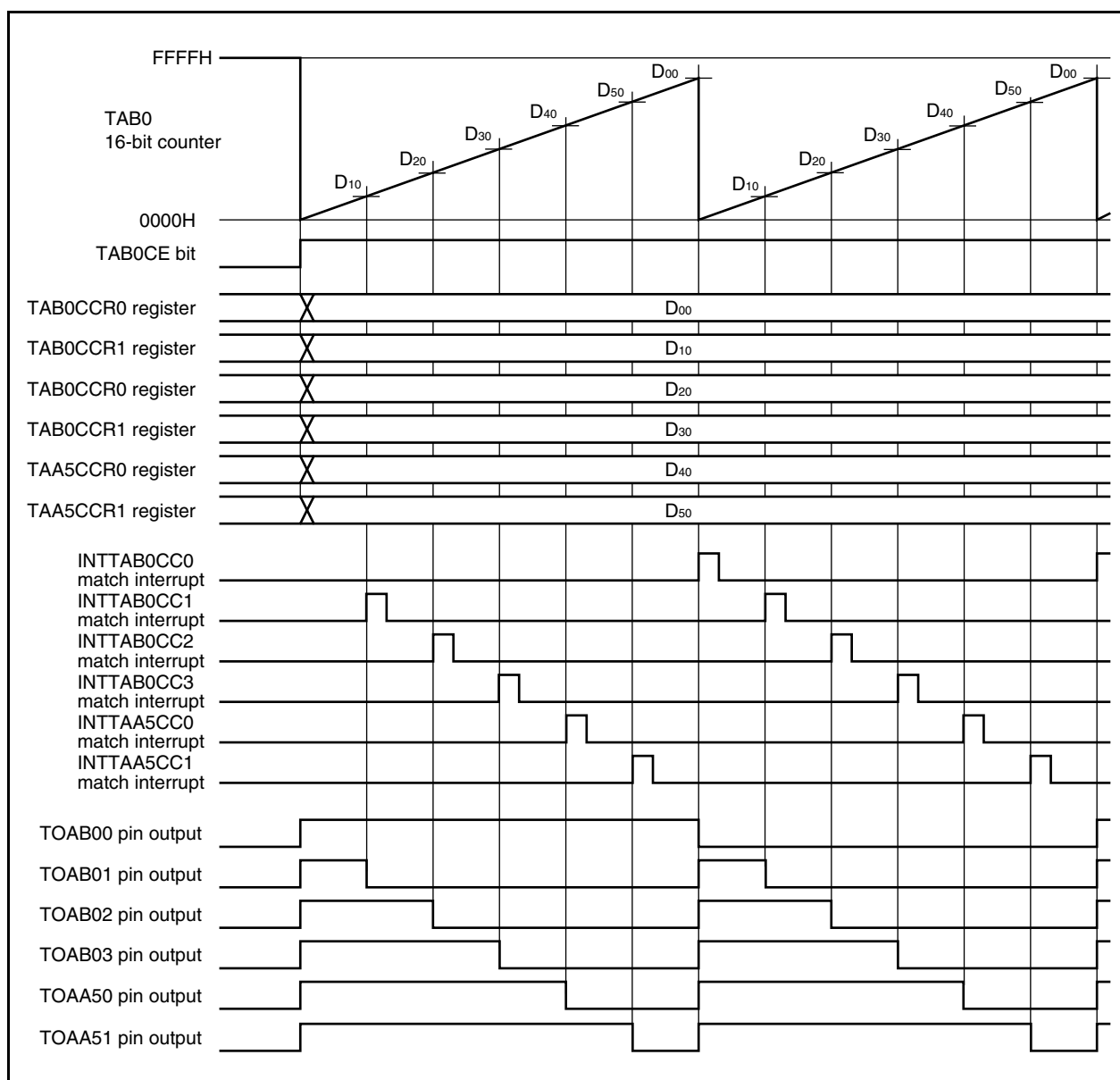
## 7.7 Simultaneous-Start Function

Timer AA and timer AB have a timer-tuned operation function.

By using the simultaneous-start function, a timer operation in which the operation start timing and count up timing of the master timer and slave timer are synchronized can be performed.

Only the PWM output mode can be used in the simultaneous-start function.

The combinations of timers that can use the simultaneous-start function are listed in Table 7-10.

**Table 7-10. Timer Simultaneous-Start Function**

| Master Timer | Slave Timer |
|:---:|:---:|
| TAA1 | TAA0 |
| TAA3 | TAA2 |
| TAB0 | TAA5 |

Figure 7-48 shows an example where individual operation and simultaneous-start operation of TAA0 (as the master timer) and TAA1 (as the slave timer) are performed in PWM output mode.

**Figure 7-48. Differences Between Individual Operation and Simultaneous-Start Operation Using TAA1 and TAA0**
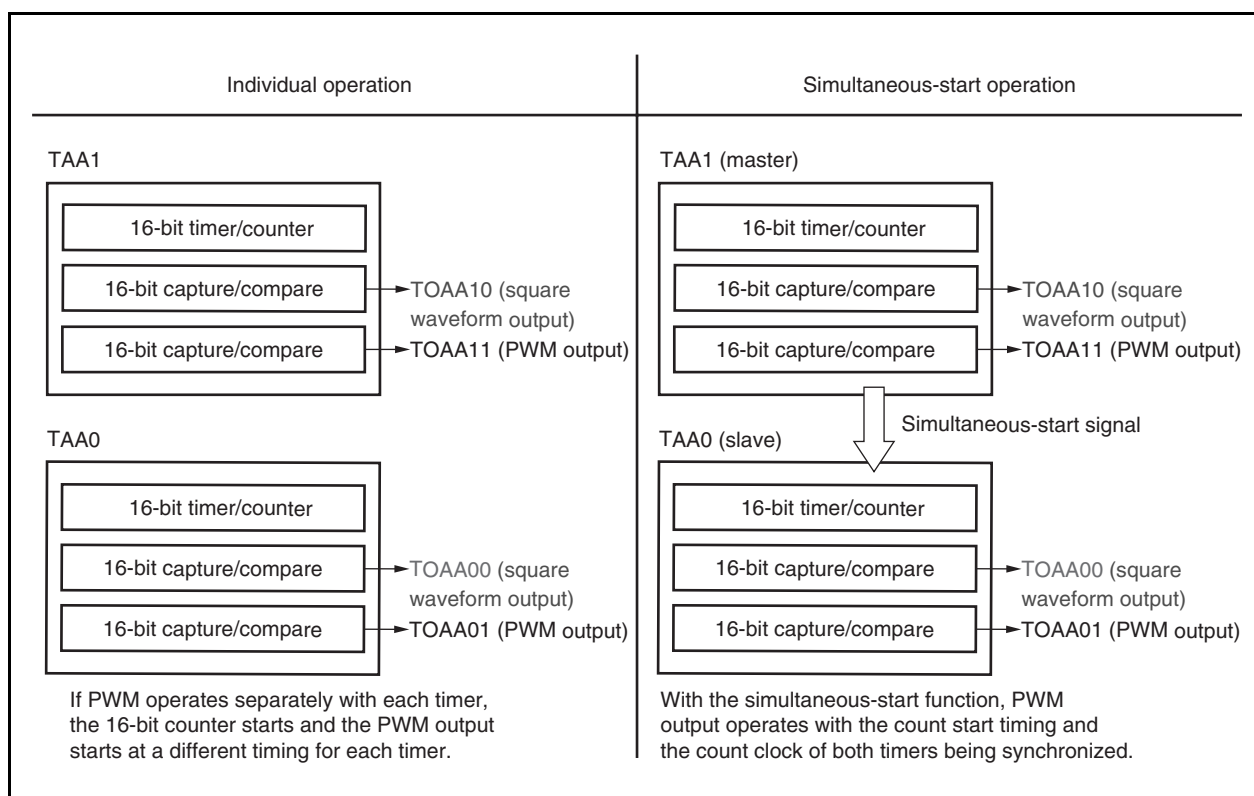
### 7.7.1   PWM output mode (simultaneous-start operation)

In this section, the operation of the simultaneous-start function is shown, where TAA1 is used as the master timer and TAA0 is used as the slave timer.

The master timer (TAA1) and slave timer (TAA0) start operating at the same time when the TAA1CTL0.TAA0CE bit of master timer is set to 1.  The slave timer operates by the count clock supplied from the master timer (TAA1).  After the slave timer starts operating, however, the 16-bit counter of the slave timer (TAA0) is not cleared even if the 16-bit counter of the master timer (TAA1) is cleared to 0000H upon a match between the 16-bit counter value of the master timer (TAA1) and the TAA1CCR0 register value, because each timer operates individually.

In the same manner, if the compare register value of the master timer (TAA1) is rewritten by batch writing, the compare register of the slave timer is not affected.

**[Initial settings]**

    Master timer: TAA1CTL0.TAA1CE = 0 (operation disabled)

    Slave timer: TAA0CTL0.TAA0CE = 0 (operation disabled)

**[Initial settings of master timer (TAA1)]**

- TAA1CTL1.TAA1MD2 to TAA1CTL1.TAA1MD0 = 100 (setting of PMW output mode)
- TAA1CTL1.TAA1CKS2 to TAA1CTL1.TAA1CKS0 (setting of count clock (any))
- TAA1CCR1, TAA1CCR0 (specification of valid edge of capture trigger)
- TAA1IOC0 (specification of valid edge of capture trigger)

**[Initial settings of slave timer (TAA0)]**

- TAA0CTL1.TAA0SYE = 1, TAA0SYM = 1 (simultaneous-start operation)
- TAA0CTL1.TAA0MD2 to TAA0CTL1.TAA0MD0 = 100 (setting of PMW output mode)
- TAA0CCR0, TAA1CCR1 (specification of valid edge of capture trigger)
- TAA0IOC0 (specification of valid edge of capture trigger)

    **Remark**    The initial settings of the master timer and slave timer may be performed in any order.

**[Starting counting]**

    <1>  Set TAA1CTL0.TAA1CE of the master timer to 1.

    <2>  Start counting.

    <3>  Changing the setting of the register during operation

- The compare register can be rewritten (anytime write).

**[End condition]**

- Set TAA1CTL0.TAA0CE of the master timer to 0.

**Figure 7-49. Timing Example of Simultaneous-Start Function (TAA1: Master, TAA0: Slave)**
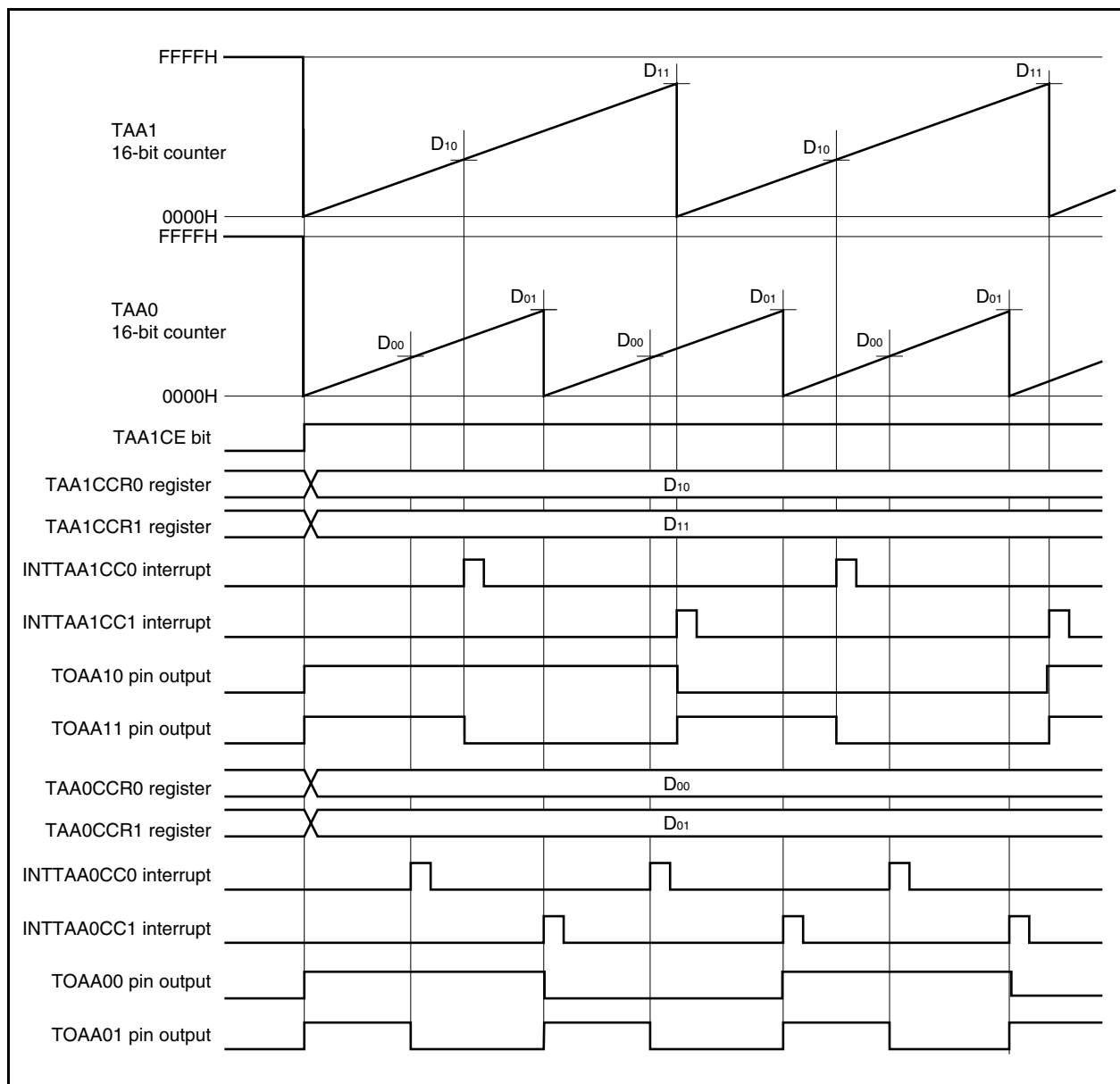
## 7.8　Cascade Connection

This section explains an operation of connecting two channels of TAA in cascade to form a 32-bit capture timer.

For cascade connection, the free-running timer mode must be set and all the capture/compare registers must be set as capture registers (TAA0CCSn = 1).

Combinations of TAA channels that can be connected in cascade are shown in the following table.

**Table 7-11.　Cascade Connection of TAA**

| Lower Timer (Master Timer) | Higher Timer (Slave Timer) |
|---|---|
| TAA1 | TAA0 |
| TAA3 | TAA2 |

In the following example, TAA1 is used as the lower timer (master timer) and TAA0 is used as the higher timer (slave timer) to use them as a 32-bit capture timer by cascade connection.

**Figure 7-50.　Cascade Connection Example**
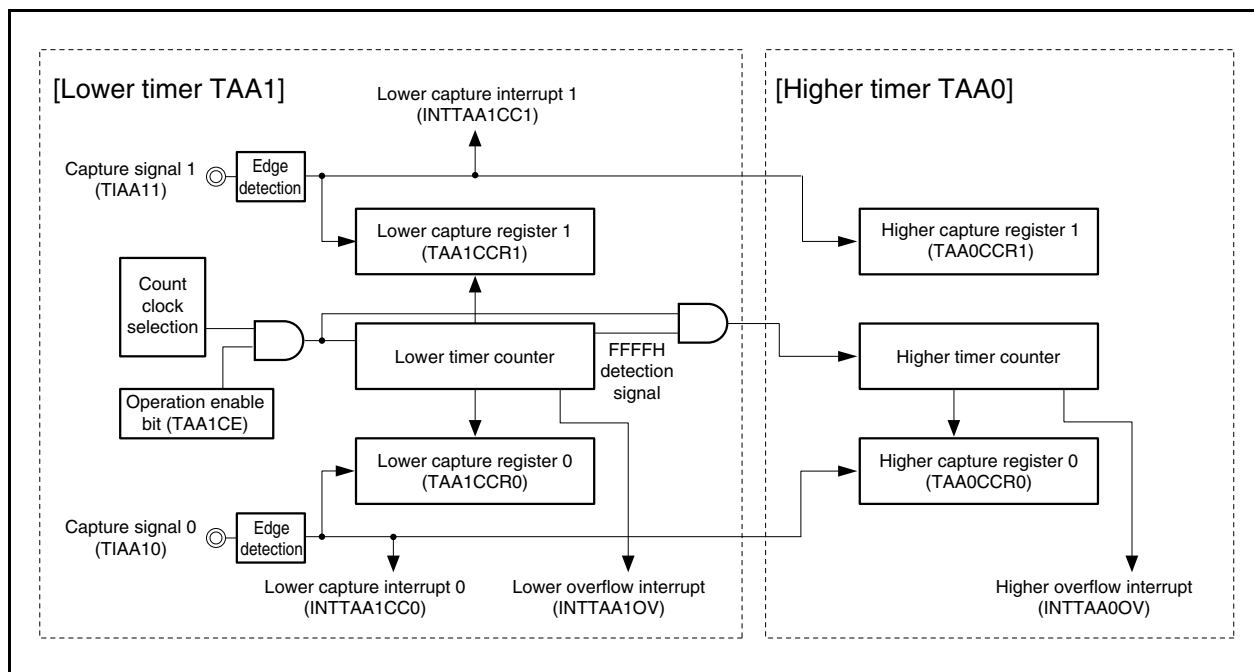
The operation of each pin and signal when TAA1 and TAA0 are connected in cascade is shown below.

**Table 7-12.  Status in Cascade Connection**

| Name | Higher/Lower | Function | Operation |
|---|---|---|---|
| TIAA10 pin input | Lower | Capture input 0 | The value of the lower timer counter is stored in the TAA1CCR0 register and the value of the higher timer counter is stored in the TAA0CCR0 register when the valid edge of this input is detected. |
| TIAA11 pin input | Lower | Capture input 1 | The value of the lower timer counter is stored in the TAA1CCR1 register and the value of the higher timer counter is stored in the TAA0CCR1 register when the valid edge of this input is detected. |
| INTTAA1CCR0 interrupt signal | Lower | Capture interrupt 0 | This interrupt is generated when the valid edge of the TIAA10 pin is detected. |
| INTTAA1CCR1 interrupt signal | Lower | Capture interrupt 1 | This interrupt is generated when the valid edge of the TIAA11 pin is detected. |
| INTTAA1OV interrupt signal | Lower | Overflow interrupt | This interrupt is generated when an overflow of the lower timer counter is detected. |
| TIAA00 pin input | Higher | Capture input 0 | Does not operate. |
| TIAA01 pin input | Higher | Capture input 1 | Does not operate. |
| INTTAA0CCR0 interrupt signal | Higher | Capture interrupt 0 | Does not operate. |
| INTTAA0CCR1 interrupt signal | Higher | Capture interrupt 1 | Does not operate. |
| INTTAA0OV interrupt signal | Higher | Overflow interrupt | This interrupt is generated when an overflow of the higher timer counter is detected. |

**Figure 7-51.  Operation Flow in Cascade Connection of TAA1 and TAA0 (1/2)**
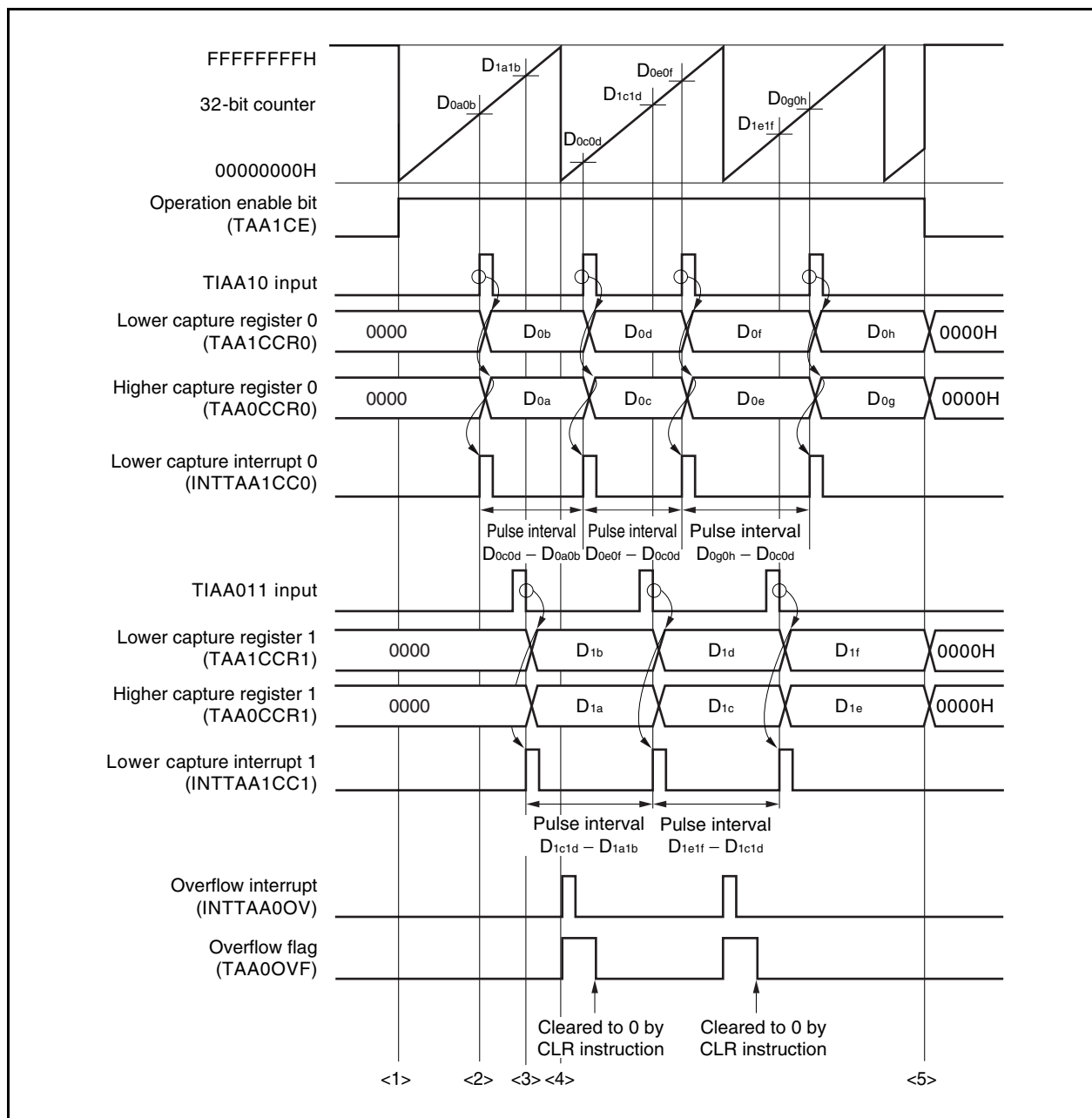
**Figure 7-51. Operation Flow in Cascade Connection of TAA1 and TAA0 (2/2)**
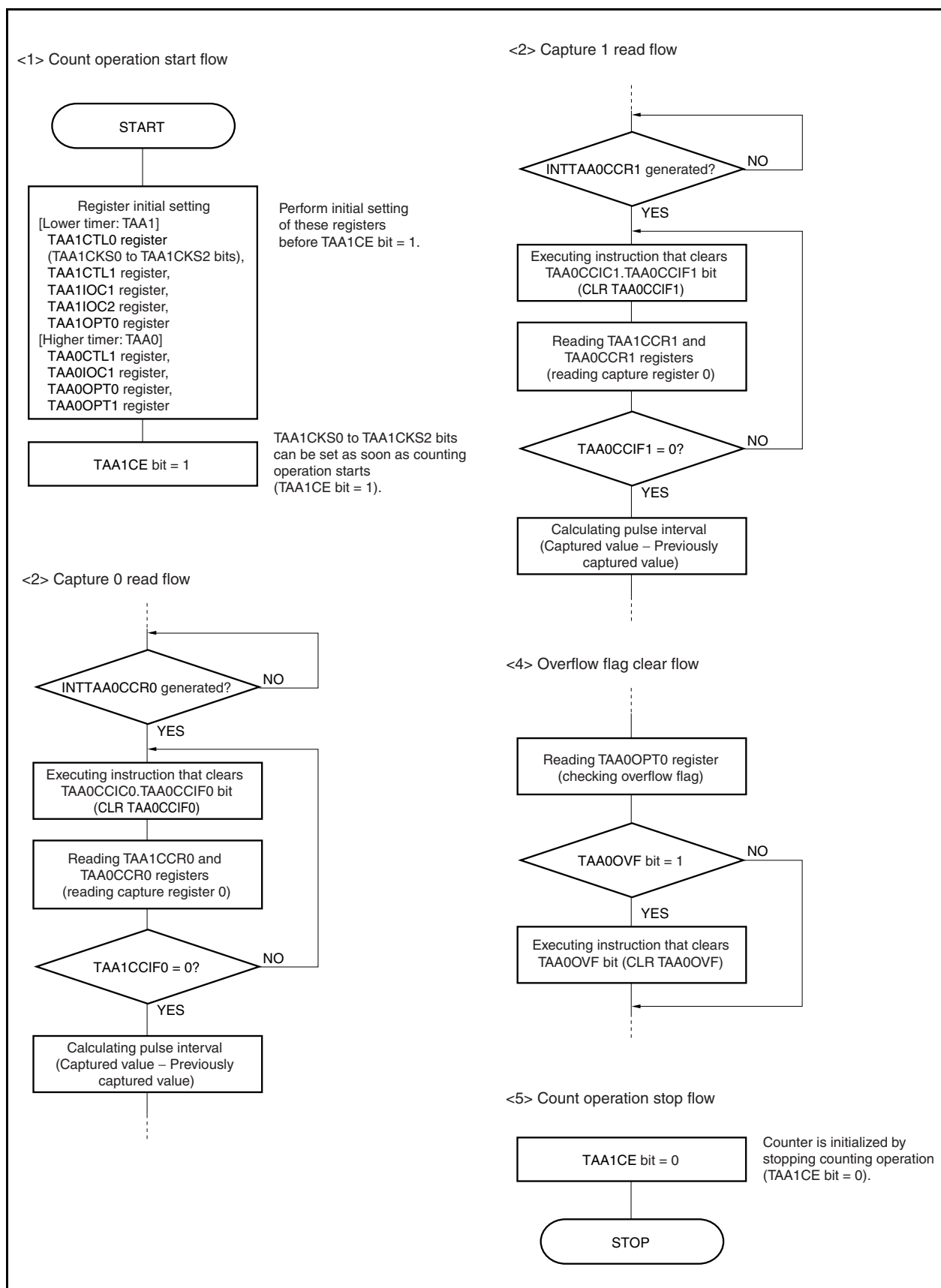
**Figure 7-52.  Example of Basic Timing When TAA1 and TAA0 Are Connected in Cascade**



The counting operation is started when the TAA1CTL.TAA1CE bit is set to 1 and the count clock is supplied. When the valid edge input to the TIAA10 pin is detected, the count value is stored in the capture register 0 (TAA1CCR0 and TAA0CCR0), and capture interrupt 0 signal (INTTAA1CC0) is issued.

The timer counter continues the counting operation in synchronization with the count clock. When it counts up to FFFFFFFFH, the overflow interrupt (INTTAA0OV) is generated at the next clock and the overflow flag (TAA0OVF) is set to 1. The timer counter is cleared to 00000000H and continues counting up.

The overflow flag (TAA0OVF) is cleared by an instruction issued from the CPU that writes "0" to it.

Because the free-running timer mode is set, the timer counter cannot be cleared by detection of the valid edge input to the TIAA10 pin.

Using TOAA10 output is prohibited because it alternately functions as the TIAA10 input.

Capture register 1 (TAA1CCR1 and TAA0CCR1) also operates in the same manner.

If the lower timer counter (TAA1) overflows, an overflow interrupt (TAA1OVF) is generated. However, it is recommended to mask this interrupt because it cannot be used as an overflow interrupt of the 32-bit counter.

## 7.9   Selector Function

In the V850ES/JG3-U and V850ES/JH3-U, the alternate-function pins of ports or peripheral I/O (TAA1, UARTC0, or UARTC1) signals can be selected as the capture trigger input of TAA1.

If the signal input from the UARTCn pin is selected by the selector function when RXDCn is used, baud rate errors of the LIN reception transfer rate of UARTCn can be calculated (n = 0, 1).

**(1)  Selector operation control register 0 (SELCNT0)**

The SELCNT0 register is an 8-bit register that selects the capture trigger for TAA1.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H      R/W      Address: FFFFF308H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| SELCNT0 | 0 | 0 | 0 | ISEL4 | ISEL3 | 0 | 0 | 0 |

| ISEL4 | Selection of TIAA11 capture trigger input signal |
|---|---|
| 0 | TIAA11 (alternately functions as P35) pin |
| 1 | RXDC1 (alternately functions as P91) pin |

| ISEL3 | Selection of TIAA10 capture trigger input signal |
|---|---|
| 0 | TIAA10 (alternately functions as P34) pin |
| 1 | RXDC0 (alternately functions as P31) pin |

**Cautions 1. To set the ISEL3 and ISEL4 bits to 1, set the corresponding function pin to the capture input mode.**

**2. Set the ISEL3 and ISEL4 bits when the operation of TAA1, UARTC0, and UARTC1 are stopped.**

**3. Be sure to set bits 7 to 5, and 2 to 0 to "0".**

## 7.10  Cautions

### (1)  Capture operation

When the capture operation is used and a slow clock is selected as the count clock, FFFFH, not 0000H, may be captured in the TAAnCCR0 and TAAnCCR1 registers if the capture trigger is input immediately after the TAAnCE bit is set to 1.



**(a)  Free-running timer mode**

**(b)  Pulse width measurement mode**
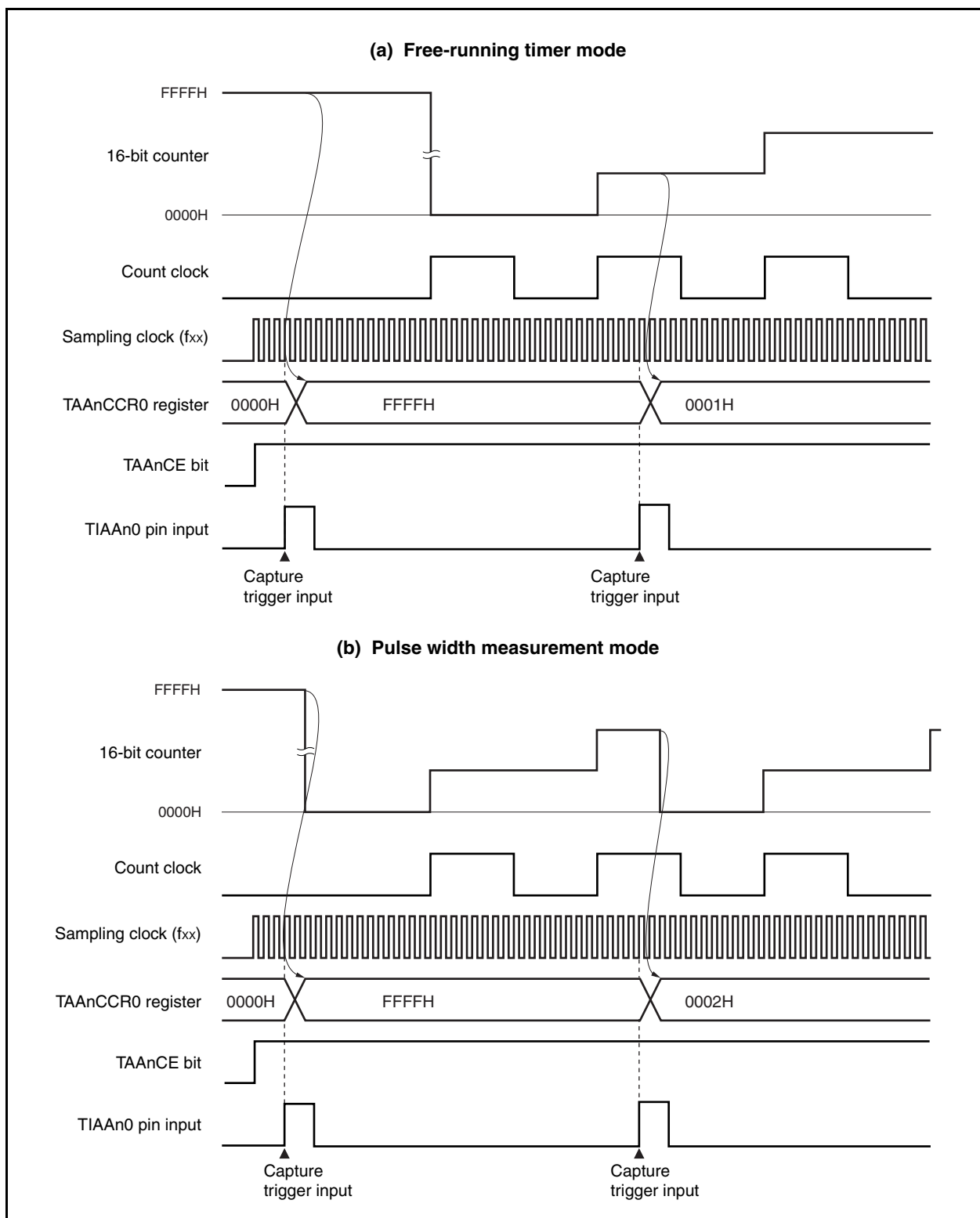
## CHAPTER 8   16-BIT TIMER/EVENT COUNTER AB (TAB)

Timer AB (TAB) is a 16-bit timer/event counter.
The V850ES/JG3-U and V850ES/JH3-U have TAB0 and TAB1.

### 8.1   Overview

An outline of TABn is shown below.

- Clock selection: 8 ways
- Capture/trigger input pins: 4
- External event count input pins: 1
- External trigger input pins: 1
- Timer counters: 1
- Capture/compare registers: 4
- Capture/compare match interrupt request signals: 4
- Timer output pins: 4

**Remark**    n = 0, 1

### 8.2   Functions

TABn has the following functions.

- Interval timer
- External event counter
- External trigger pulse output
- One-shot pulse output
- PWM output
- Free-running timer
- Pulse width measurement
- Triangular wave PWM output
- Timer-tuned operation function
- Simultaneous-start function

**Remark**    n = 0, 1

## 8.3  Configuration

TABn includes the following hardware.

**Table 8-1.  Configuration of TABn**

| Item | Configuration |
|---|---|
| Registers | 16-bit counter <br> TABn capture/compare registers 0 to 3 (TABnCCR0 to TABnCCR3) <br> TABn counter read buffer register (TABnCNT) <br> CCR0 to CCR3 buffer registers <br> TABn control registers 0, 1 (TABnCTL0, TABnCTL1) <br> TABn I/O control registers 0 to 2 (TABnIOC0 to TABnIOC2, TABnIOC4) <br> TABn option register 0 (TABnOPT0) |
| Timer inputs[Note 2] | 4 (TIABn0[Note 1] to TIABn3 pins) |
| Timer outputs[Note 2] | 4 (TOABn0 to TOABn3 pins) |

**Notes 1.** When using the functions of the TIABn0 to TIABn3 and TOABn0 to TOABn3 pins, see **Table 4-20 Using Port Pin as Alternate-Function Pin**.

**2.** The TIAB00 pin functions alternately as a capture trigger input signal, external event count input signal, and external trigger input signal.

**Figure 8-1.  Block Diagram of TABn**
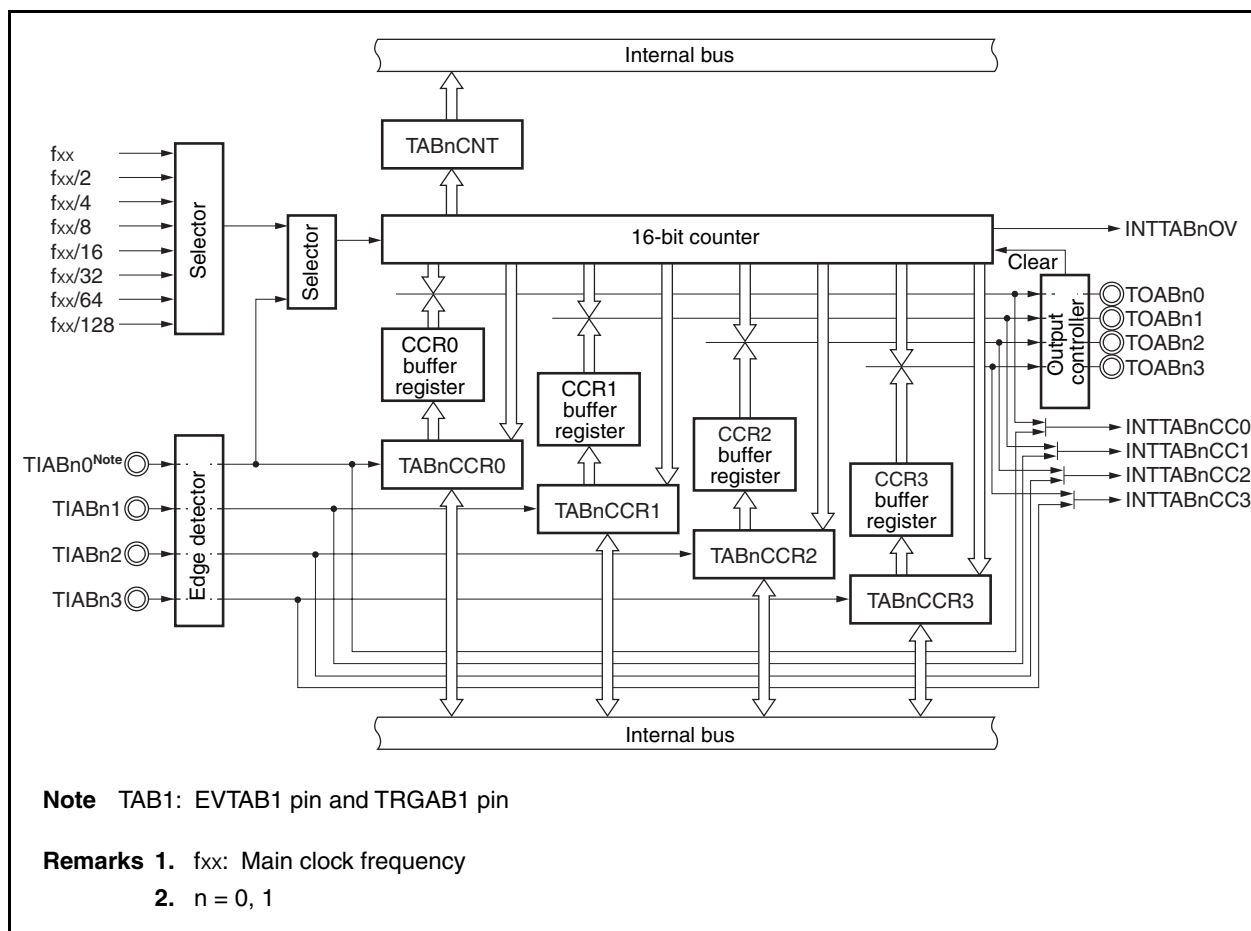


**Note** TAB1: EVTAB1 pin and TRGAB1 pin

**Remarks 1.** fxx:  Main clock frequency

**2.** n = 0, 1

**(1)  16-bit counter**

This 16-bit counter can count internal clocks or external events.

The count value of this counter can be read by using the TABnCNT register.

When the TABnCTL0.TABnCE bit = 0, the value of the 16-bit counter is FFFFH.  If the TABnCNT register is read at this time, 0000H is read.

Reset sets the TABnCE bit to 0.  Therefore, the 16-bit counter is set to FFFFH.

**(2)  CCR0 buffer register**

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TABnCCR0 register is used as a compare register, the value written to the TABnCCR0 register is transferred to the CCR0 buffer register.  When the count value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTABnCC0) is generated.

The CCR0 buffer register cannot be read or written directly.

The CCR0 buffer register is cleared to 0000H after reset, as the TABnCCR0 register is cleared to 0000H.

**(3)  CCR1 buffer register**

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TABnCCR1 register is used as a compare register, the value written to the TABnCCR1 register is transferred to the CCR1 buffer register.  When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTABnCC1) is generated.

The CCR1 buffer register cannot be read or written directly.

The CCR1 buffer register is cleared to 0000H after reset, as the TABnCCR1 register is cleared to 0000H.

**(4)  CCR2 buffer register**

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TABnCCR2 register is used as a compare register, the value written to the TABnCCR2 register is transferred to the CCR2 buffer register. When the count value of the 16-bit counter matches the value of the CCR2 buffer register, a compare match interrupt request signal (INTTABnCC2) is generated.

The CCR2 buffer register cannot be read or written directly.

The CCR2 buffer register is cleared to 0000H after reset, as the TABnCCR2 register is cleared to 0000H.

**(5)  CCR3 buffer register**

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TABnCCR3 register is used as a compare register, the value written to the TABnCCR3 register is transferred to the CCR3 buffer register. When the count value of the 16-bit counter matches the value of the CCR3 buffer register, a compare match interrupt request signal (INTTABnCC3) is generated.

The CCR3 buffer register cannot be read or written directly.

The CCR3 buffer register is cleared to 0000H after reset, as the TABnCCR3 register is cleared to 0000H.

**(6)  Edge detector**

This circuit detects the valid edges input to the TIABn0 to TIABn3 pins.  No edge, rising edge, falling edge, or both the rising and falling edges can be selected as the valid edge by using the TABnIOC1 and TABnIOC2 registers.

RENESAS

**(7)  Output controller**

This circuit controls the output of the TOABn0 to TOABn3 pins.  The output controller is controlled by the TABnIOC0 register.

**(8)  Selector**

This selector selects the count clock for the 16-bit counter.  Eight types of internal clocks or an external event can be selected as the count clock.

## 8.4   Registers

The registers that control TABn are as follows.

- TABn control register 0 (TABnCTL0)
- TABn control register 1 (TABnCTL1)
- TABn I/O control register 0 (TABnIOC0)
- TABn I/O control register 1 (TABnIOC1)
- TABn I/O control register 2 (TABnIOC2)
- TABn I/O control register 4 (TABnIOC4)
- TABn option register 0 (TABnOPT0)
- TABn capture/compare register 0 (TABnCCR0)
- TABn capture/compare register 1 (TABnCCR1)
- TABn capture/compare register 2 (TABnCCR2)
- TABn capture/compare register 3 (TABnCCR3)
- TABn counter read buffer register (TABnCNT)

**Remarks 1.** When using the functions of the TIABn0 to TIABn3 and TOABn0 to TOABn3 pins, see **Table 4-20  Using Port Pin as Alternate-Function Pin**.

   **2.** n = 0, 1

**(1) TABn control register 0 (TABnCTL0)**

The TABnCTL0 register is an 8-bit register that controls the operation of TABn.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

Software can be used to always write the same value to the TABnCTL0 register.

After reset: 00H    R/W    Address:    TAB0CTL0 FFFFF540H, TAB1CTL0 FFFFF560H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TABnCTL0 (n = 0, 1) | TABnCE | 0 | 0 | 0 | 0 | TABnCKS2 | TABnCKS1 | TABnCKS0 |

| TABnCE | TABn operation control |
|---|---|
| 0 | TABn operation disabled (TABn reset asynchronously[Note]). |
| 1 | TABn operation enabled.  TABn operation started. |

| TABnCKS2 | TABnCKS1 | TABnCKS0 | Internal count clock selection |
|---|---|---|---|
| 0 | 0 | 0 | $f_{XX}$ |
| 0 | 0 | 1 | $f_{XX}/2$ |
| 0 | 1 | 0 | $f_{XX}/4$ |
| 0 | 1 | 1 | $f_{XX}/8$ |
| 1 | 0 | 0 | $f_{XX}/16$ |
| 1 | 0 | 1 | $f_{XX}/32$ |
| 1 | 1 | 0 | $f_{XX}/64$ |
| 1 | 1 | 1 | $f_{XX}/128$ |

**Note**    TABnOPT0.TABnOVF bit, 16-bit counter, timer output (TOABn0 to TOABn3 pins)

**Cautions 1. Set the TABnCKS2 to TABnCKS0 bits when the TABnCE bit = 0. When the value of the TABnCE bit is changed from 0 to 1, the TABnCKS2 to TABnCKS0 bits can be set simultaneously.**
**2. Be sure to set bits 3 to 6 to "0".**

**Remark**    $f_{XX}$:  Main clock frequency

**(2) TABn control register 1 (TABnCTL1)**

The TABnCTL1 register is an 8-bit register that controls the operation of TABn.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address: TAB0CTL1 FFFFF541H, TAB1CTL1 FFFFF561H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TABnCTL1 (n = 0, 1) | 0 | TABnEST | TABnEEE | 0 | 0 | TABnMD2 | TABnMD1 | TABnMD0 |

| TABnEST | Software trigger control |
|---|---|
| 0 | − |
| 1 | Generate a valid signal for external trigger input. <br>• In one-shot pulse output mode: A one-shot pulse is output with writing 1 to the TABnEST bit as the trigger. <br>• In external trigger pulse output mode: A PWM waveform is output with writing 1 to the TABnEST bit as the trigger. |

| TABnEEE | Count clock selection |
|---|---|
| 0 | Disable operation with external event count input. (Perform counting with the count clock selected by the TABnCTL0.TABnCK0 to TABnCTL0.TABnCK2 bits.) |
| 1 | Enable operation with external event count input. (Perform counting at the valid edge of the external event count input signal.) |

The TABnEEE bit selects whether counting is performed with the internal count clock or the valid edge of the external event count input.

| TABnMD2 | TABnMD1 | TABnMD0 | Timer mode selection |
|---|---|---|---|
| 0 | 0 | 0 | Interval timer mode |
| 0 | 0 | 1 | External event count mode |
| 0 | 1 | 0 | External trigger pulse output mode |
| 0 | 1 | 1 | One-shot pulse output mode |
| 1 | 0 | 0 | PWM output mode |
| 1 | 0 | 1 | Free-running timer mode |
| 1 | 1 | 0 | Pulse width measurement mode |
| 1 | 1 | 1 | Triangular wave PWM mode |

**Cautions 1. The TABnEST bit is valid only in the external trigger pulse output mode or one-shot pulse output mode. In any other mode, writing 1 to this bit is ignored.**

      **2. Be sure to set bits 3, 4, and 7 to "0".**

      **3. External event count input is selected in the external event count mode regardless of the value of the TABnEEE bit.**

      **4. Set the TABnEEE and TABnMD2 to TABnMD0 bits when the TABnCTL0.TABnCE bit = 0. (The same value can be written when the TABnCE bit = 1.) The operation is not guaranteed when rewriting is performed with the TABnCE bit = 1. If rewriting was mistakenly performed, clear the TABnCE bit to 0 and then set the bits again.**

**(3) TABn I/O control register 0 (TABnIOC0)**

The TABnIOC0 register is an 8-bit register that controls the timer output (TOABn0 to TOABn3 pins).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address:   TAB0IOC0 FFFFF542H, TAB1IOC0 FFFFF562H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC0<br>(n = 0, 1) | TABnOL3 | TABnOE3 | TABnOL2 | TABnOE2 | TABnOL1 | TABnOE1 | TABnOL0 | TABnOE0 |

| TABnOLm | TOABnm pin output level setting (m = 0 to 3)[Note] |
|---|---|
| 0 | TOABnm pin high level start |
| 1 | TOABnm pin low level start |

| TABnOEm | TOABnm pin output setting (m = 0 to 3) |
|---|---|
| 0 | Timer output disabled<br>• When TABnOLm bit = 0: Low level is output from the TOABnm pin<br>• When TABnOLm bit = 1: High level is output from the TOABnm pin |
| 1 | Timer output enabled (a square wave is output from the TOABnm pin). |

**Note**   The output level of the timer output pin (TOABnm) specified by the TABnOLm bit is shown below.

• When TABnOLm bit = 0      • When TABnOLm bit = 1

16-bit counter     16-bit counter

TABnCE bit     TABnCE bit

TOABnm output pin     TOABnm output pin
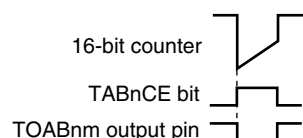
**Cautions 1.** **Rewrite the TABnOLm and TABnOEm bits when the TABnCTL0.TABnCE bit = 0. (The same value can be written when the TABnCE bit = 1.) If rewriting was mistakenly performed, clear the TABnCE bit to 0 and then set the bits again.**

**2.** **Even if the TABnOLm bit is manipulated when the TABnCE and TABnOEm bits are 0, the TOABnm pin output level varies.**

**Remark**   m = 0 to 3

**(4) TABn I/O control register 1 (TABnIOC1)**

The TABnIOC1 register is an 8-bit register that controls the valid edge of the capture trigger input signals (TIABn0 to TIABn3 pins).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address:    TAB0IOC1 FFFFF543H, TAB1IOC1 FFFFF563H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC1 (n = 0, 1) | TABnIS7 | TABnIS6 | TABnIS5 | TABnIS4 | TABnIS3 | TABnIS2 | TABnIS1 | TABnIS0 |

| TABnIS7 | TABnIS6 | Capture trigger input signal (TIABn3 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (capture operation invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

| TABnIS5 | TABnIS4 | Capture trigger input signal (TIABn2 pin) valid edge detection |
|---|---|---|
| 0 | 0 | No edge detection (capture operation invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

| TABnIS3 | TABnIS2 | Capture trigger input signal (TIABn1 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (capture operation invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

| TABnIS1 | TABnIS0 | Capture trigger input signal (TIABn0 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (capture operation invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

**Cautions 1. Rewrite the TABnIS7 to TABnIS0 bits when the TABnCTL0.TABnCE bit = 0. (The same value can be written when the TABnCE bit = 1.) If rewriting was mistakenly performed, clear the TABnCE bit to 0 and then set the bits again.**

**2. The TABnIS7 to TABnIS0 bits are valid only in the free-running timer mode and the pulse width measurement mode. In all other modes, a capture operation is not possible.**

**(5) TABn I/O control register 2 (TABnIOC2)**

The TABnIOC2 register is an 8-bit register that controls the valid edge of the external event count input signal (TIAB00/EVTAB1 pin) and external trigger input signal (TIAB00/TRGAB1 pin).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

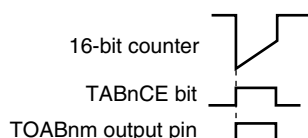After reset:  00H    R/W    Address:    TAB0IOC2 FFFFF544H, TAB1IOC2 FFFFF564H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC2<br>(n = 0, 1) | 0 | 0 | 0 | 0 | TABnEES1 | TABnEES0 | TABnETS1 | TABnETS0 |

| TABnEES1 | TABnEES0 | External event count input signal (TIAB00/EVTAB1 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (external event count invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

| TABnETS1 | TABnETS0 | External trigger input signal (TIAB00/TRGAB1 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (external trigger invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

**Cautions 1. Rewrite the TABnEES1, TABnEES0, TABnETS1, and TABnETS0 bits when the TABnCTL0.TABnCE bit = 0. (The same value can be written when the TABnCE bit = 1.) If rewriting was mistakenly performed, clear the TABnCE bit to 0 and then set the bits again.**

**2. The TABnEES1 and TABnEES0 bits are valid only when the TABnCTL1.TABnEEE bit = 1 or when the external event count mode (TABnCTL1.TABnMD2 to TABnCTL1.TABnMD0 bits = 001) has been set.**

**3. The TABnETS1 and TABnETS0 bits are valid only when the external trigger pulse output mode (TABnCTL1.TABnMD2 to TABnCTL1.TABnMD0 bits = 010) or the one-shot pulse output mode (TABnCTL1.TABnMD2 to TABnCTL1.TABnMD0 = 011) is set.**

**(6) TABn I/O control register 4 (TABnIOC4)**

The TABnIOC4 register is an 8-bit register that controls the timer output.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.  This register is not reset by stopping the timer operation (TABnCTL0.TABnCE = 0).

**Cautions 1. Accessing the TABnIOC4 register is prohibited in the following statuses. For details, see 3.4.8**
**(2) Accessing specific on-chip peripheral I/O registers.**
- **When the CPU operates on the subclock and the main clock oscillation is stopped**
- **When the CPU operates on the internal oscillation clock**

**2. The TABnIOC4 register can be set only in the interval timer mode and free-running timer mode.**
**Be sure to set the TABnIOC4 register to 00H in all other modes (for details of the mode setting,**
**see 8.4 (2) TABn control register 1 (TABnCTL1)).  Even in free-running timer mode, if the**
**TABnCCR0 to TABnCCR3 registers are set to the capture function, the setting of the**
**TABnIOC4 register becomes invalid.**

After reset: 00H      R/W      Address:      TAB0IOC4 FFFFF550H, TAB1IOC4 FFFFF570H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC4 (n = 0, 1) | TABnOS3 | TABnOR3 | TABnOS2 | TABnOR2 | TABnOS1 | TABnOR1 | TABnOS0 | TABnOR0 |

| TABnOSm | TABnORm | Toggle control of TOABnm pin (m = 0 to 3) |
|---|---|---|
| 0 | 0 | No request.  Normal  toggle operation. |
| 0 | 1 | Reset request<br>Fix to inactive level upon next match between value of 16-bit counter and value of TAAnCCRm register. |
| 1 | 0 | Set request<br>Fix to active level upon next match between value of 16-bit counter and value of TAAnCCRm register. |
| 1 | 1 | Keep request<br>Keep the current output level. |

**(7) TABn option register 0 (TABnOPT0)**

The TABnOPT0 register is an 8-bit register used to set the capture/compare operation and detect an overflow.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address:    TAB0OPT0 FFFFF545H, TAB1OPT0 FFFFF565H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TABnOPT0 <br> (n = 0, 1) | TABnCCS3 | TABnCCS2 | TABnCCS1 | TABnCCS0 | 0 | TAB1CMS[Note] | TABnCUF | TABnOVF |

| TABnCCSm | TABnCCRm register capture/compare selection |
|---|---|
| 0 | Compare register selected |
| 1 | Capture register selected |
| The TABnCCSm bit setting is valid only in the free-running timer mode. | |

| TABnOVF | TABn overflow detection |
|---|---|
| Set (1) | Overflow occurred |
| Reset (0) | TABnOVF bit 0 written or TABnCTL0.TABnCE bit = 0 |

- The TABnOVF bit is set to 1 when the 16-bit counter count value overflows from FFFFH to 0000H in the free-running timer mode or the pulse width measurement mode.
- An interrupt request signal (INTTABnOV) is generated at the same time that the TABnOVF bit is set to 1. The INTTABnOV signal is not generated in modes other than the free-running timer mode and the pulse width measurement mode.
- The TABnOVF bit is not cleared even when the TABnOVF bit or the TABnOPT0 register are read when the TABnOVF bit = 1.
- The TABnOVF bit can be both read and written, but the TABnOVF bit cannot be set to 1 by software. Writing 1 has no effect on the operation of TABn.

**Note**   The TAB1CMS bit is used for the motor control function. For details, see **CHAPTER 11 MOTOR CONTROL FUNCTION**.

**Cautions 1.** Rewrite the TABnCCS3 to TABnCCS0 bits when the TABnCTL0.TABnCE bit = 0. (The same value can be written when the TABnCE bit = 1.) If rewriting was mistakenly performed, clear the TABnCE bit to 0 and then set the bits again.

          **2.** Be sure to set bit 3 to "0". When the motor control function is not used, be sure to also set bit 2 to "0".

**Remark**   m = 0 to 3

**(8) TABn capture/compare register 0 (TABnCCR0)**

The TABnCCR0 register can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, according to the setting of the TABnOPT0.TABnCCS0 bit. In the pulse width measurement mode, the TABnCCR0 register can be used only as a capture register. In any other mode, this register can be used only as a compare register.

The TABnCCR0 register can be read or written during operation.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

**Caution   Accessing the TABnCCR0 register is prohibited in the following statuses.  For details, see 3.4.8**
**(2)  Accessing specific on-chip peripheral I/O registers.**
- **When the CPU operates on the subclock and the main clock oscillation is stopped**
- **When the CPU operates on the internal oscillation clock**

After reset: 0000H     R/W     Address:     TAB0CCR0 FFFFF546H, TAB1CCR0 FFFFF566H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TABnCCR0 (n = 0, 1) | | | | | | | | | | | | | | | | |
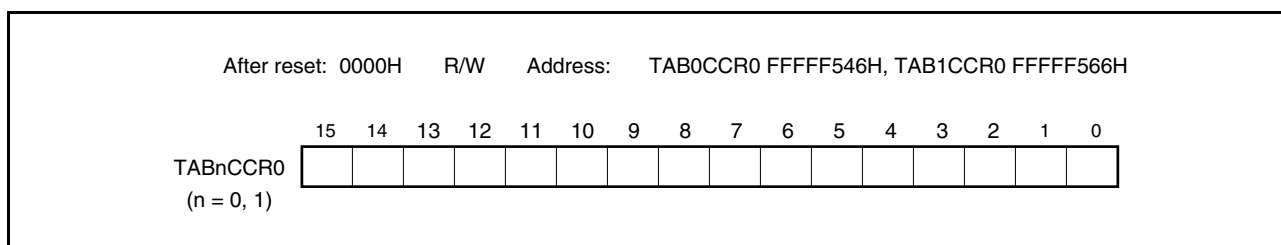
**(a) Function as compare register**

The TABnCCR0 register can be rewritten even when the TABnCTL0.TABnCE bit = 1.

The set value of the TABnCCR0 register is transferred to the CCR0 buffer register.  When the value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTABnCC0) is generated.  If TOABn0 pin output is enabled at this time, the output of the TOABn0 pin is inverted.

When the TABnCCR0 register is used as a cycle register in the interval timer mode, external event count mode, external trigger pulse output mode, one-shot pulse output mode, PWM output mode, or triangular wave PWM mode, the value of the 16-bit counter is cleared (0000H) if its count value matches the value of the CCR0 buffer register.

**(b) Function as capture register**

When the TABnCCR0 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TABnCCR0 register if the valid edge of the capture trigger input pin (TIABn0 pin) is detected.  In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TABnCCR0 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIABn0 pin) is detected.

Even if the capture operation and reading the TABnCCR0 register conflict, the correct value of the TABnCCR0 register can be read.

**Remark**   n = 0, 1

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

**Table 8-2.  Function of Capture/Compare Register in Each Mode and How to Write Compare Register**

| Operation Mode | Capture/Compare Register | How to Write Compare Register |
|---|---|---|
| Interval timer | Compare register | Anytime write |
| External event counter | Compare register | Anytime write |
| External trigger pulse output | Compare register | Batch write |
| One-shot pulse output | Compare register | Anytime write |
| PWM output | Compare register | Batch write |
| Free-running timer | Capture/compare register | Anytime write |
| Pulse width measurement | Capture register | – |
| Triangular wave PWM mode | Compare register | Batch write |

**(9) TABn capture/compare register 1 (TABnCCR1)**

The TABnCCR1 register can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, according to the setting of the TABnOPT0.TABnCCS1 bit.  In the pulse width measurement mode, the TABnCCR1 register can be used only as a capture register.  In any other mode, this register can be used only as a compare register.

The TABnCCR1 register can be read or written during operation.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

**Caution   Accessing the TABnCCR1 register is prohibited in the following statuses.  For details, see 3.4.8**
**(2)  Accessing specific on-chip peripheral I/O registers.**
- **When the CPU operates on the subclock and the main clock oscillation is stopped**
- **When the CPU operates on the internal oscillation clock**

---

After reset:  0000H      R/W      Address:      TAB0CCR1 FFFFF548H, TAB1CCR1 FFFFF568H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| TABnCCR1 (n = 0, 1) | | | | | | | | | | | | | | | | |

---

**(a) Function as compare register**

The TABnCCR1 register can be rewritten even when the TABnCTL0.TABnCE bit = 1.

The set value of the TABnCCR1 register is transferred to the CCR1 buffer register. When the value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTABnCC1) is generated. If TOABn1 pin output is enabled at this time, the output of the TOABn1 pin is inverted.

**(b) Function as capture register**

When the TABnCCR1 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TABnCCR1 register if the valid edge of the capture trigger input pin (TIABn1 pin) is detected. In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TABnCCR1 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIABn1 pin) is detected.

Even if the capture operation and reading the TABnCCR1 register conflict, the correct value of the TABnCCR1 register can be read.

**Remark** n = 0, 1

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

**Table 8-3.  Function of Capture/Compare Register in Each Mode and How to Write Compare Register**

| Operation Mode | Capture/Compare Register | How to Write Compare Register |
|---|---|---|
| Interval timer | Compare register | Anytime write |
| External event counter | Compare register | Anytime write |
| External trigger pulse output | Compare register | Batch write |
| One-shot pulse output | Compare register | Anytime write |
| PWM output | Compare register | Batch write |
| Free-running timer | Capture/compare register | Anytime write |
| Pulse width measurement | Capture register | – |
| Triangular wave PWM mode | Compare register | Batch write |

**(10) TABn capture/compare register 2 (TABnCCR2)**

The TABnCCR2 register can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, according to the setting of the TABnOPT0.TABnCCS2 bit.  In the pulse width measurement mode, the TABnCCR2 register can be used only as a capture register.  In any other mode, this register can be used only as a compare register.

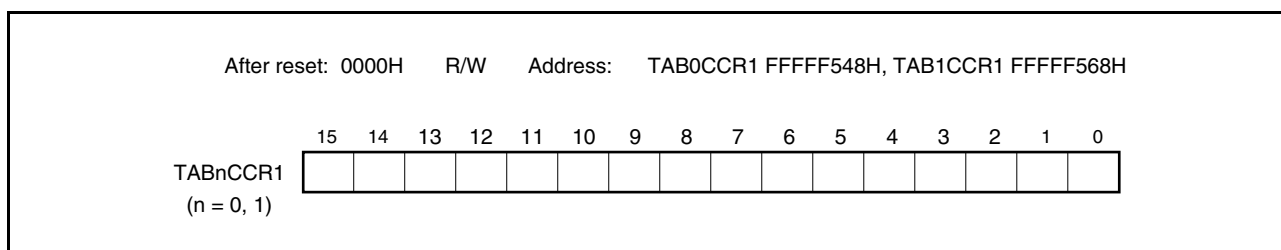The TABnCCR2 register can be read or written during operation.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

**Caution   Accessing the TABnCCR2 register is prohibited in the following statuses.  For details, see 3.4.8**
**(2)  Accessing specific on-chip peripheral I/O registers.**
   - **When the CPU operates with the subclock and the main clock oscillation is stopped**
   - **When the CPU operates with the internal oscillation clock**

After reset:  0000H     R/W     Address:    TAB0CCR2 FFFFF54AH, TAB1CCR2 FFFFF56AH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TABnCCR2 (n = 0, 1) | | | | | | | | | | | | | | | | |

**(a) Function as compare register**

The TABnCCR2 register can be rewritten even when the TABnCTL0.TABnCE bit = 1.

The set value of the TABnCCR2 register is transferred to the CCR2 buffer register. When the value of the 16-bit counter matches the value of the CCR2 buffer register, a compare match interrupt request signal (INTTABnCC2) is generated. If TOABn2 pin output is enabled at this time, the output of the TOABn2 pin is inverted.

**(b) Function as capture register**

When the TABnCCR2 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TABnCCR2 register if the valid edge of the capture trigger input pin (TIABn2 pin) is detected. In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TABnCCR2 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIABn2 pin) is detected.

Even if the capture operation and reading the TABnCCR2 register conflict, the correct value of the TABnCCR2 register can be read.

**Remark**   n = 0, 1

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

**Table 8-4.  Function of Capture/Compare Register in Each Mode and How to Write Compare Register**

| Operation Mode | Capture/Compare Register | How to Write Compare Register |
|---|---|---|
| Interval timer | Compare register | Anytime write |
| External event counter | Compare register | Anytime write |
| External trigger pulse output | Compare register | Batch write |
| One-shot pulse output | Compare register | Anytime write |
| PWM output | Compare register | Batch write |
| Free-running timer | Capture/compare register | Anytime write |
| Pulse width measurement | Capture register | – |
| Triangular wave PWM mode | Compare register | Batch write |

**(11) TABn capture/compare register 3 (TABnCCR3)**

The TABnCCR3 register can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, according to the setting of the TABnOPT0.TABnCCS3 bit. In the pulse width measurement mode, the TABnCCR3 register can be used only as a capture register. In any other mode, this register can be used only as a compare register.

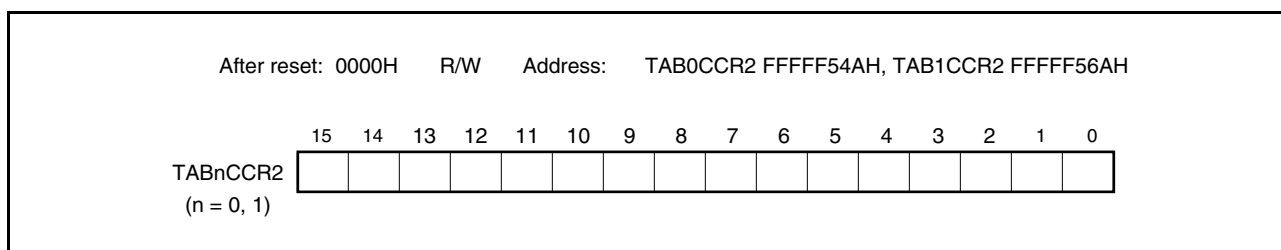The TABnCCR3 register can be read or written during operation.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

**Caution    Accessing the TABnCCR3 register is prohibited in the following statuses. For details, see 3.4.8
(2) Accessing specific on-chip peripheral I/O registers.**

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

After reset: 0000H     R/W     Address:    TAB0CCR3 FFFFF54CH, TAB1CCR3 FFFFF56CH

|  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TABnCCR3 (n = 0, 1) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
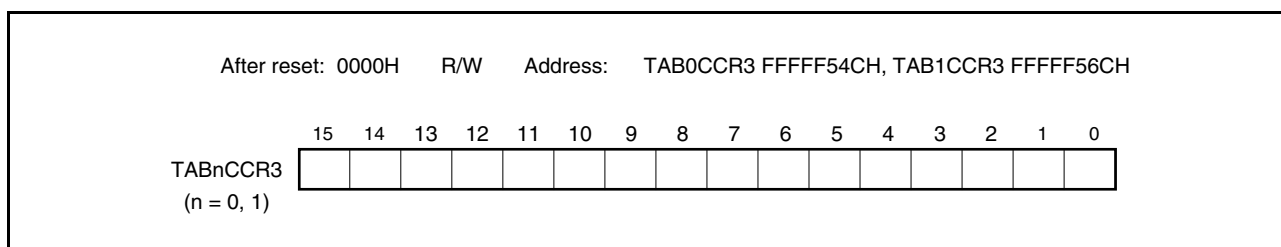
**(a) Function as compare register**

The TABnCCR3 register can be rewritten even when the TABnCTL0.TABnCE bit = 1.

The set value of the TABnCCR3 register is transferred to the CCR3 buffer register.  When the value of the 16-bit counter matches the value of the CCR3 buffer register, a compare match interrupt request signal (INTTABnCC3) is generated.  If TOABn3 pin output is enabled at this time, the output of the TOABn3 pin is inverted.

**(b) Function as capture register**

When the TABnCCR3 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TABnCCR3 register if the valid edge of the capture trigger input pin (TIABn3 pin) is detected.  In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TABnCCR3 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIABn3 pin) is detected.

Even if the capture operation and reading the TABnCCR3 register conflict, the correct value of the TABnCCR3 register can be read.

**Remark**   n = 0, 1

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

**Table 8-5.  Function of Capture/Compare Register in Each Mode and How to Write Compare Register**

| Operation Mode | Capture/Compare Register | How to Write Compare Register |
|---|---|---|
| Interval timer | Compare register | Anytime write |
| External event counter | Compare register | Anytime write |
| External trigger pulse output | Compare register | Batch write |
| One-shot pulse output | Compare register | Anytime write |
| PWM output | Compare register | Batch write |
| Free-running timer | Capture/compare register | Anytime write |
| Pulse width measurement | Capture register | – |
| Triangular wave PWM mode | Compare register | Batch write |

**(12) TABn counter read buffer register (TABnCNT)**

The TABnCNT register is a read buffer register that can read the count value of the 16-bit counter.

If this register is read when the TABnCTL0.TABnCE bit = 1, the count value of the 16-bit timer can be read.

This register is read-only in 16-bit units.

The value of the TABnCNT register is cleared to 0000H when the TABnCE bit = 0. If the TABnCNT register is read at this time, the value of the 16-bit counter (FFFFH) is not read, but 0000H is read.

The value of the TABnCNT register is cleared to 0000H after reset, as the TABnCE bit is cleared to 0.

**Caution  Accessing the TABnCNT register is prohibited in the following statuses.  For details, see 3.4.8 (2) Accessing specific on-chip peripheral I/O registers.**
- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

After reset: 0000H    R/W    Address:    TAB0CCR0 FFFFF546H, TAB1CCR0 FFFFF566H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TABnCCR0 (n = 0, 1) | | | | | | | | | | | | | | | | |
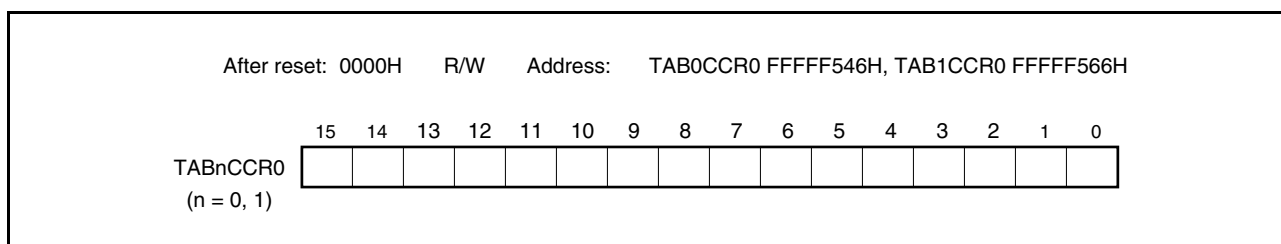
## 8.5　Operation

TABn can perform the following operations.

| Operation | TABnCTL1.TABnEST Bit (Software Trigger Bit) | TIABn0 Pin (External Trigger Input) | Capture/Compare Register Setting | Compare Register Write |
|---|---|---|---|---|
| Interval timer mode | Invalid | Invalid | Compare only | Anytime write |
| External event count mode[Note 1] | Invalid | Invalid | Compare only | Anytime write |
| External trigger pulse output mode[Note 2] | Valid | Valid | Compare only | Batch write |
| One-shot pulse output mode[Note 2] | Valid | Valid | Compare only | Anytime write |
| PWM output mode | Invalid | Invalid | Compare only | Batch write |
| Free-running timer mode | Invalid | Invalid | Switching enabled | Anytime write |
| Pulse width measurement mode[Note 2] | Invalid | Invalid | Capture only | Not applicable |
| Triangular wave PWM mode | Invalid | Invalid | Compare only | Batch write |

**Notes 1.** To use the external event count mode, specify that the valid edge of the TIABn0 pin capture trigger input is not detected (by clearing the TABnIOC1.TABnIS1 and TABnIOC1.TABnIS0 bits to "00").

　　　　**2.** When using the external trigger pulse output mode, one-shot pulse output mode, and pulse width measurement mode, select the internal clock as the count clock (by clearing the TABnCTL1.TABnEEE bit to 0).

**Remark**　n = 0, 1

### 8.5.1 Interval timer mode (TABnMD2 to TABnMD0 bits = 000)

In the interval timer mode, an interrupt request signal (INTTABnCC0) is generated at the specified interval if the TABnCTL0.TABnCE bit is set to 1. A square wave whose half cycle is equal to the interval can be output from the TOABn0 pin.

Usually, the TABnCCR1 to TABnCCR3 registers are not used in the interval timer mode.

**Figure 8-2. Configuration of Interval Timer**



**Remark**    n = 0, 1
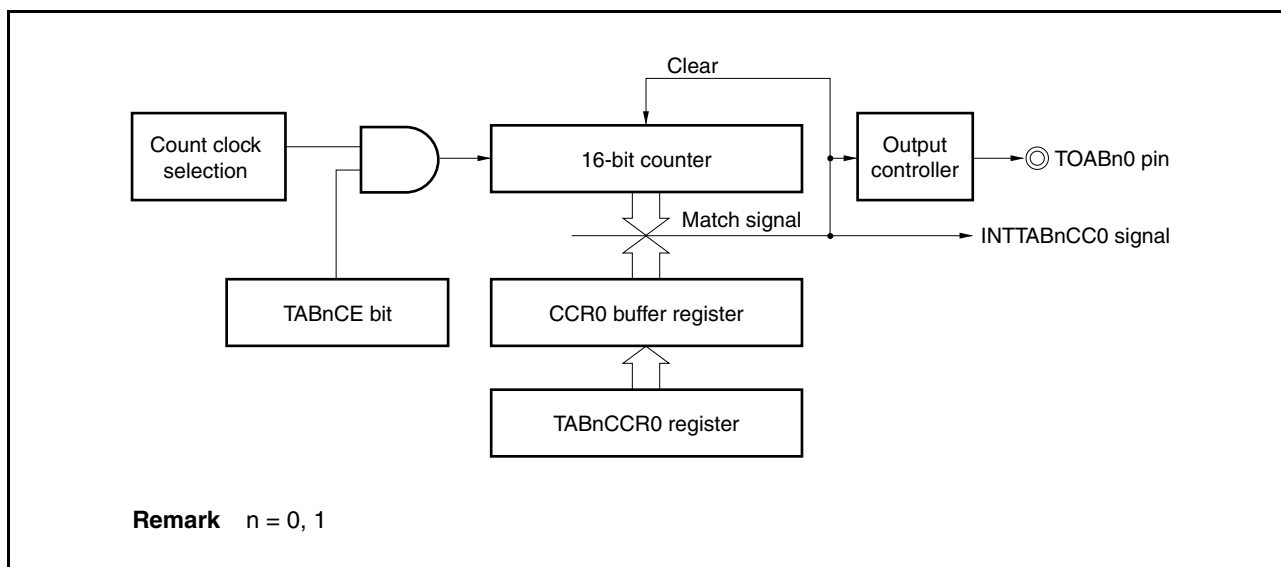
**Figure 8-3. Basic Timing of Operation in Interval Timer Mode**



**Remark**    n = 0, 1

When the TABnCE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H in synchronization with the count clock, and the counter starts counting.  At this time, the output of the TOABn0 pin is inverted.  Additionally, the set value of the TABnCCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, the output of the TOABn0 pin is inverted, and a compare match interrupt request signal (INTTABnCC0) is generated.

The interval can be calculated by the following expression.

Interval = (Set value of TABnCCR0 register + 1) × Count clock cycle

**Figure 8-4.  Register Setting for Interval Timer Mode Operation (1/2)**



**Note**  This bit can be set to 1 only when the interrupt request signals (INTTABnCC0 and INTTABnCCk) are masked by the interrupt mask flags (TABnCCMK0 to TABnCCMKk) and the timer output (TOABnk) is performed at the same time.  However, the TABnCCR0 and TABnCCRk registers must be set to the same value (see **8.5.1 (2) (d)  Operation of TABnCCR1 to TABnCCR3 registers**) (k = 1 to 3).

**Remark**   n = 0, 1

**Figure 8-4. Register Setting for Interval Timer Mode Operation (2/2)**

**(c) TABn I/O control register 0 (TABnIOC0)**



**(d) TABn counter read buffer register (TABnCNT)**

By reading the TABnCNT register, the count value of the 16-bit counter can be read.

**(e) TABn capture/compare register 0 (TABnCCR0)**

If the TABnCCR0 register is set to $D_0$, the interval is as follows.

Interval = $(D_0 + 1) \times$ Count clock cycle

**(f) TABn capture/compare registers 1 to 3 (TABnCCR1 to TABnCCR3)**

Usually, the TABnCCR1 to TABnCCR3 registers are not used in the interval timer mode. However, the set value of the TABnCCR1 to TABnCCR3 registers is transferred to the CCR1 to CCR3 buffer registers. The compare match interrupt request signals (INTTABnCCR1 to INTTABnCCR3) are generated when the count value of the 16-bit counter matches the value of the CCR1 to CCR3 buffer registers. Therefore, mask the interrupt requests by using the corresponding interrupt mask flags (TABnCCMK1 to TABnCCMK3).

**Remarks 1.** TABn I/O control register 1 (TABnIOC1), TABn I/O control register 2 (TABnIOC2), and TABn option register 0 (TABnOPT0) are not used in the interval timer mode.

       **2.** n = 0, 1

**(1) Interval timer mode operation flow**

**Figure 8-5. Software Processing Flow in Interval Timer Mode**



<1> Count operation start flow

START

Register initial setting
TABnCTL0 register
(TABnCKS0 to TABnCKS2 bits)
TABnCTL1 register,
TABnIOC0 register,
TABnCCR0 register

The initial setting of these registers is performed before setting the TABnCE bit to 1.

TABnCE bit = 1

The TABnCKS0 to TABnCKS2 bits can be set at the same time when counting has been started (TABnCE bit = 1).

<2> Count operation stop flow

TABnCE bit = 0

The counter is initialized and counting is stopped by clearing the TABnCE bit to 0.
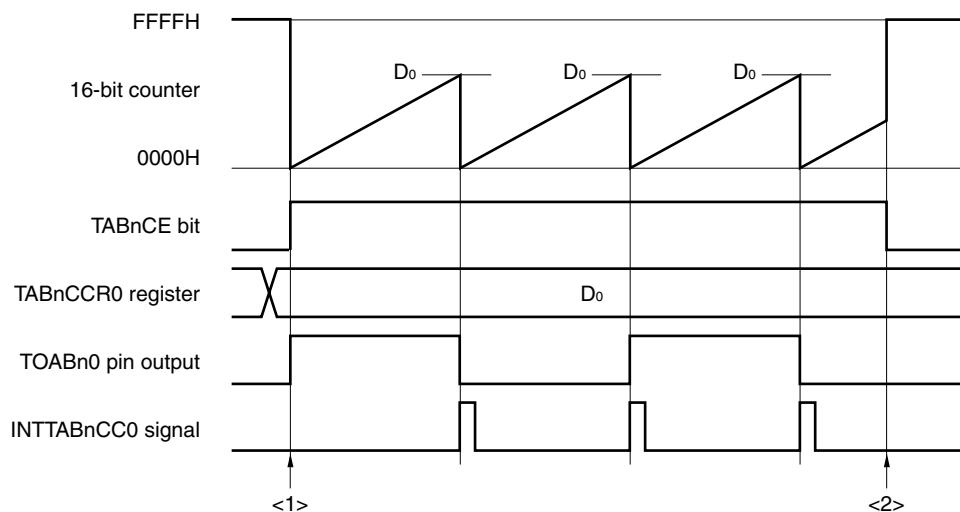
STOP

**Remark** n = 0, 1

**(2) Interval timer mode operation timing**

**(a) Operation if TABnCCR0 register is set to 0000H**

If the TABnCCR0 register is set to 0000H, the INTTABnCC0 signal is generated at each count clock subsequent to the first count clock, and the output of the TOABn0 pin is inverted.

The value of the 16-bit counter is always 0000H.



**Remark**　n = 0, 1
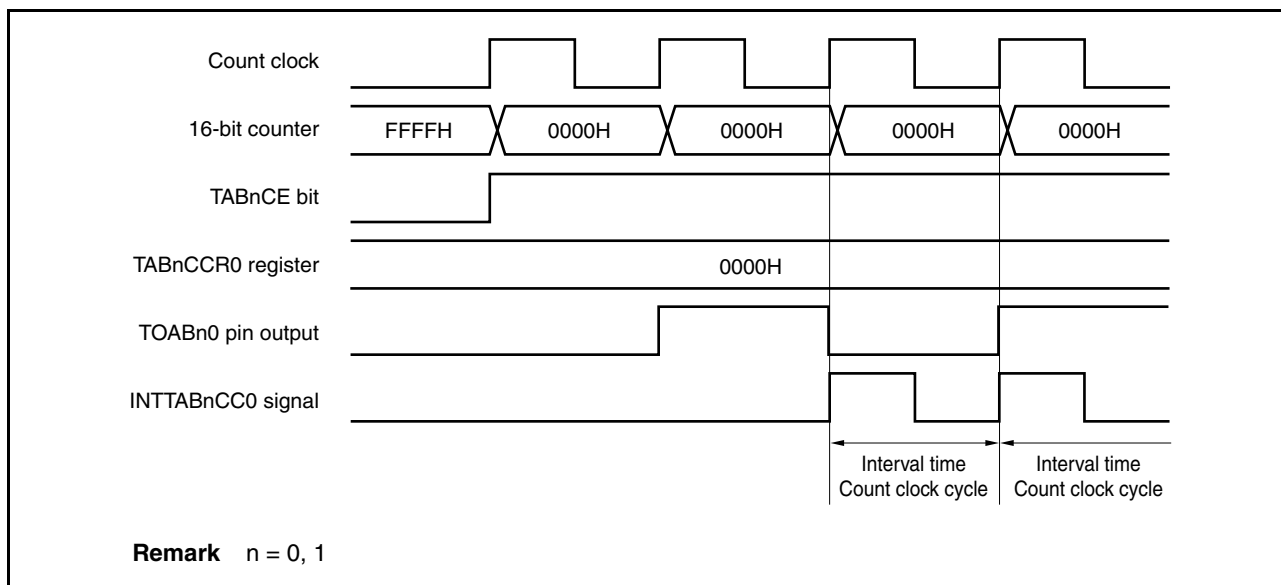
**(b) Operation if TABnCCR0 register is set to FFFFH**

If the TABnCCR0 register is set to FFFFH, the 16-bit counter counts up to FFFFH. The counter is cleared to 0000H in synchronization with the next count-up timing. The INTTABnCC0 signal is generated and the output of the TOABn0 pin is inverted. At this time, an overflow interrupt request signal (INTTABnOV) is not generated, nor is the overflow flag (TABnOPT0.TABnOVF bit) set to 1.



**Remark**　n = 0, 1
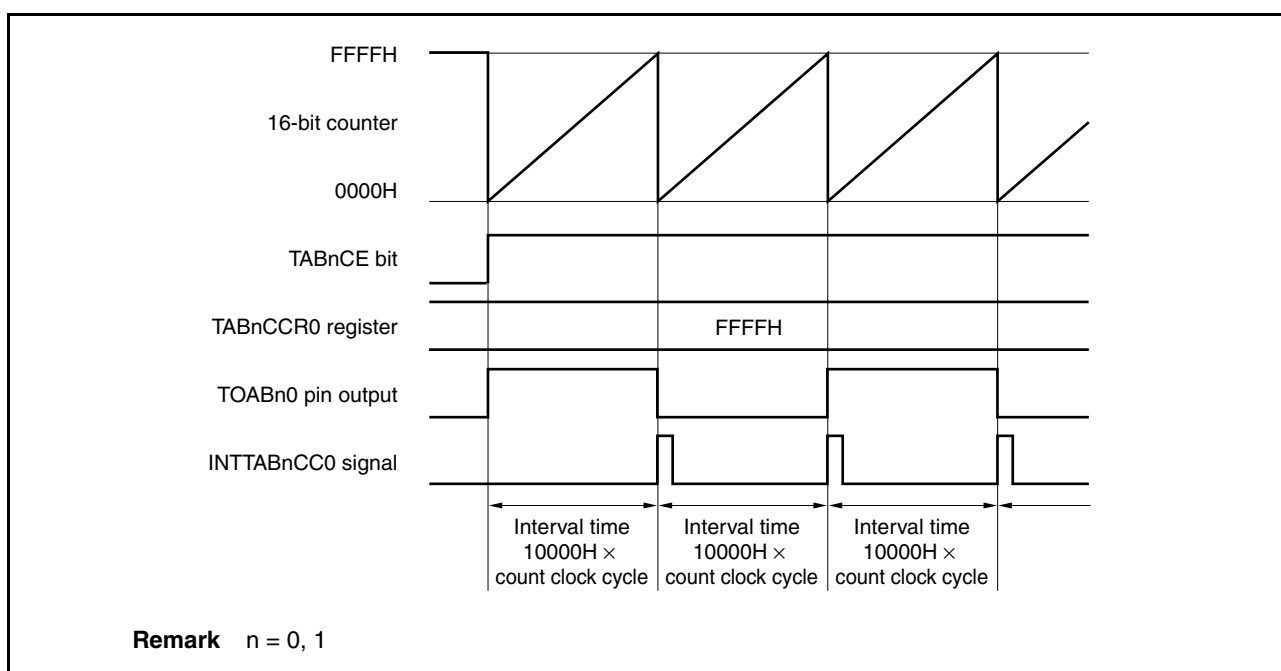
**(c) Notes on rewriting TABnCCR0 register**

To change the value of the TABnCCR0 register to a smaller value, stop counting once and then change the set value.

If the value of the TABnCCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow.
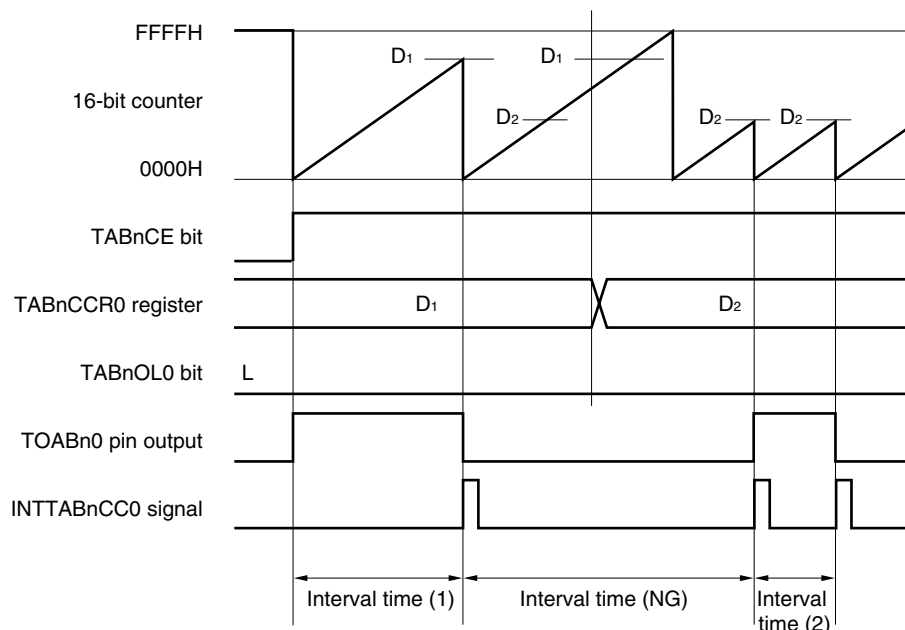


Remarks **1.** Interval time (1):    $(D_1 + 1) \times$ Count clock cycle

            Interval time (NG): $(10000H + D_2 + 1) \times$ Count clock cycle

            Interval time (2):    $(D_2 + 1) \times$ Count clock cycle

      **2.** n = 0, 1

If the value of the TABnCCR0 register is changed from $D_1$ to $D_2$ while the count value is greater than $D_2$ but less than $D_1$, the count value is transferred to the CCR0 buffer register as soon as the TABnCCR0 register has been rewritten. Consequently, the value of the 16-bit counter that is compared is $D_2$.

Because the count value has already exceeded $D_2$, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H. When the count value matches $D_2$, the INTTABnCC0 signal is generated and the output of the TOABn0 pin is inverted.

Therefore, the INTTABnCC0 signal may not be generated at the interval time "$(D_1 + 1) \times$ Count clock cycle" or "$(D_2 + 1) \times$ Count clock cycle" originally expected, but may be generated at an interval of "$(10000H + D_2 + 1) \times$ Count clock period".

**(d)  Operation of TABnCCR1 to TABnCCR3 registers**

**Figure 8-6.  Configuration of TABnCCR1 to TABnCCR3 Registers**



**Remark**    n = 0, 1
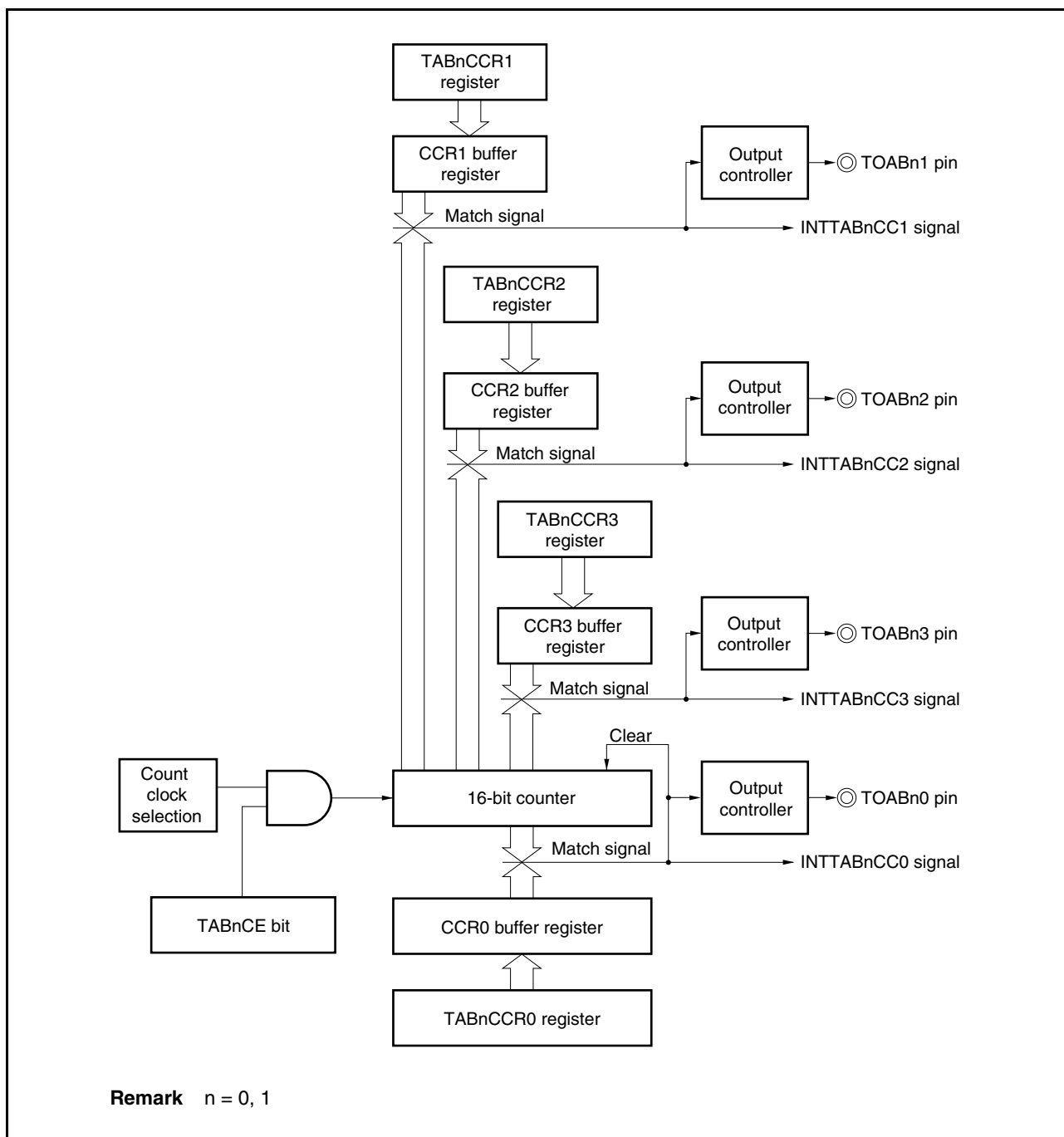
If the set value of the TABnCCRk register is less than the set value of the TABnCCR0 register, the INTTABnCCk signal is generated once per cycle. At the same time, the output of the TOABnk pin is inverted. The TOABnk pin outputs a square wave with the same cycle as that output by the TOABn0 pin.

**Remark**  k = 1 to 3,
           n = 0, 1

**Figure 8-7.  Timing Chart When $D_{01} \geq D_{k1}$**
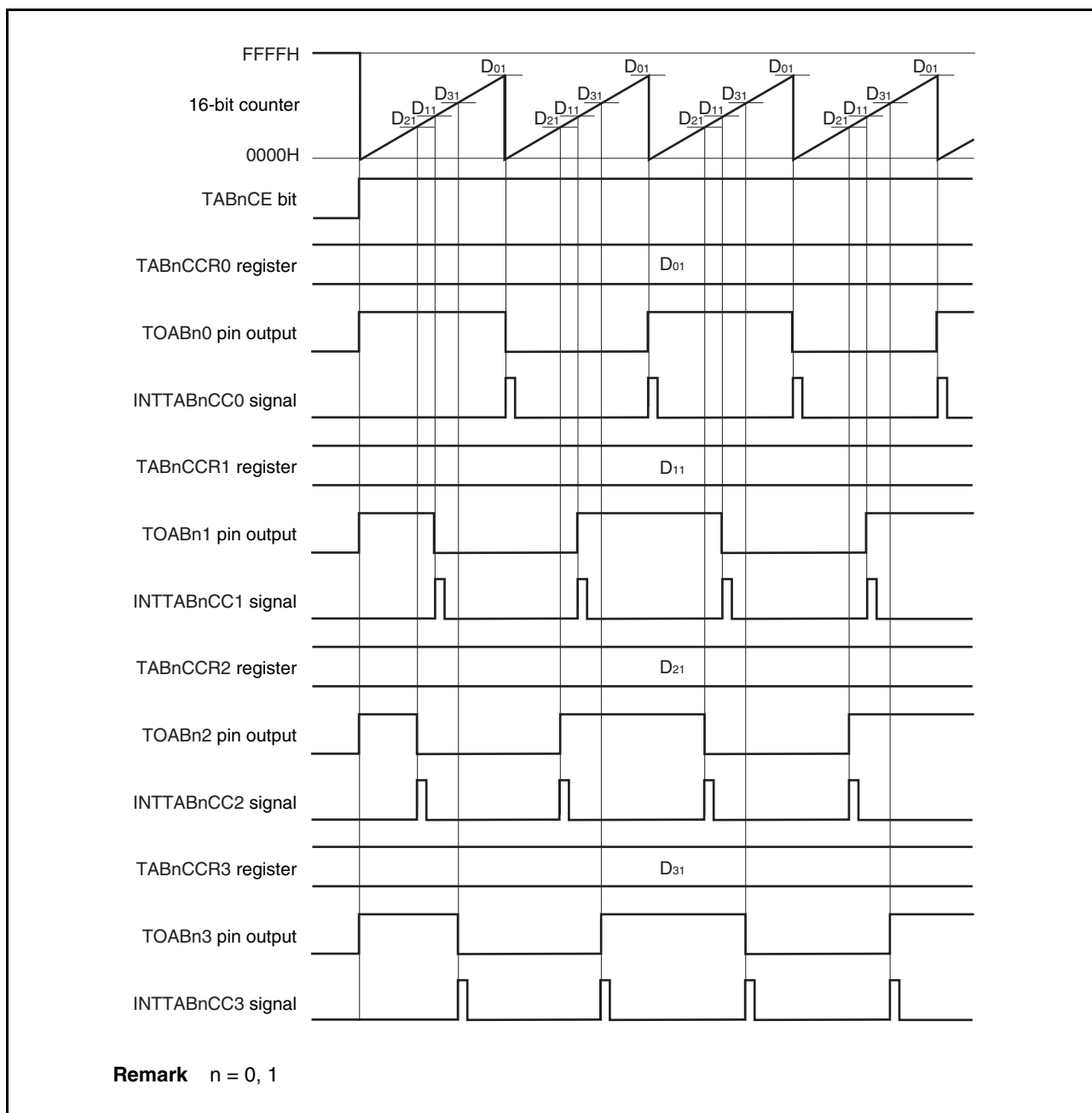


**Remark**  n = 0, 1

If the set value of the TABnCCRk register is greater than the set value of the TABnCCR0 register, the count value of the 16-bit counter does not match the value of the TABnCCRk register. Consequently, the INTTABnCCk signal is not generated, nor is the output of the TOABnk pin changed.

**Remark**     k = 1 to 3,
               n = 0, 1

**Figure 8-8. Timing Chart When $D_{01} < D_{k1}$**



**Remark**    n = 0, 1
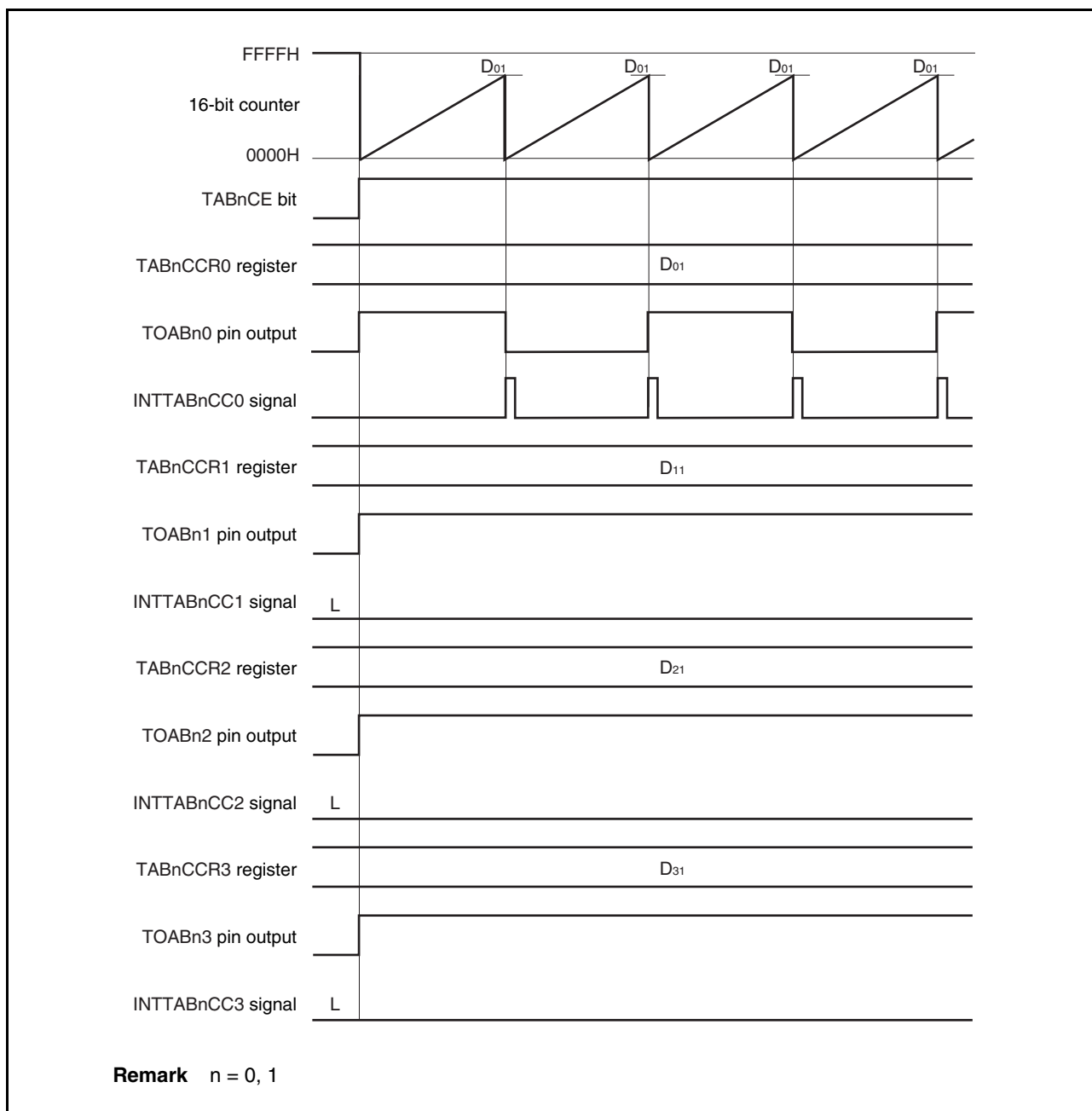
### 8.5.2 External event count mode (TABnMD2 to TABnMD0 bits = 001)

In the external event count mode, the valid edge of the external event count input is counted when the TABnCTL0.TABnCE bit is set to 1, and an interrupt request signal (INTTABnCC0) is generated each time the specified number of edges have been counted. The TOABn0 pin cannot be used.

Usually, the TABnCCR1 to TABnCCR3 registers are not used in the external event count mode.

**Figure 8-9. Configuration in External Event Count Mode**



**Note** TAB1: EVTAB1 pin
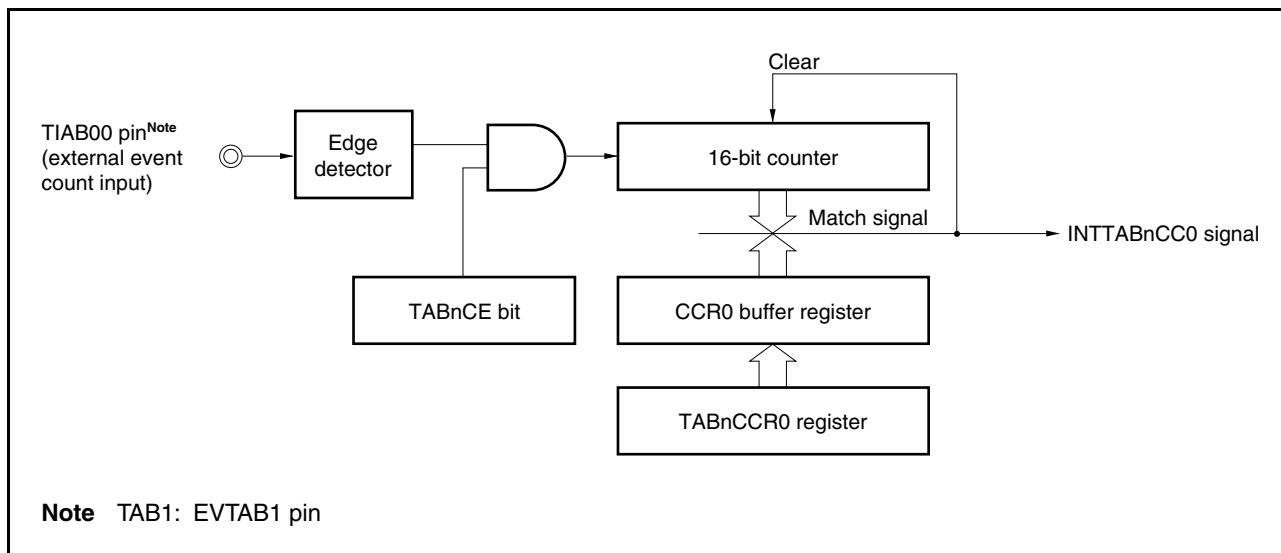
**Figure 8-10. Basic Timing in External Event Count Mode**
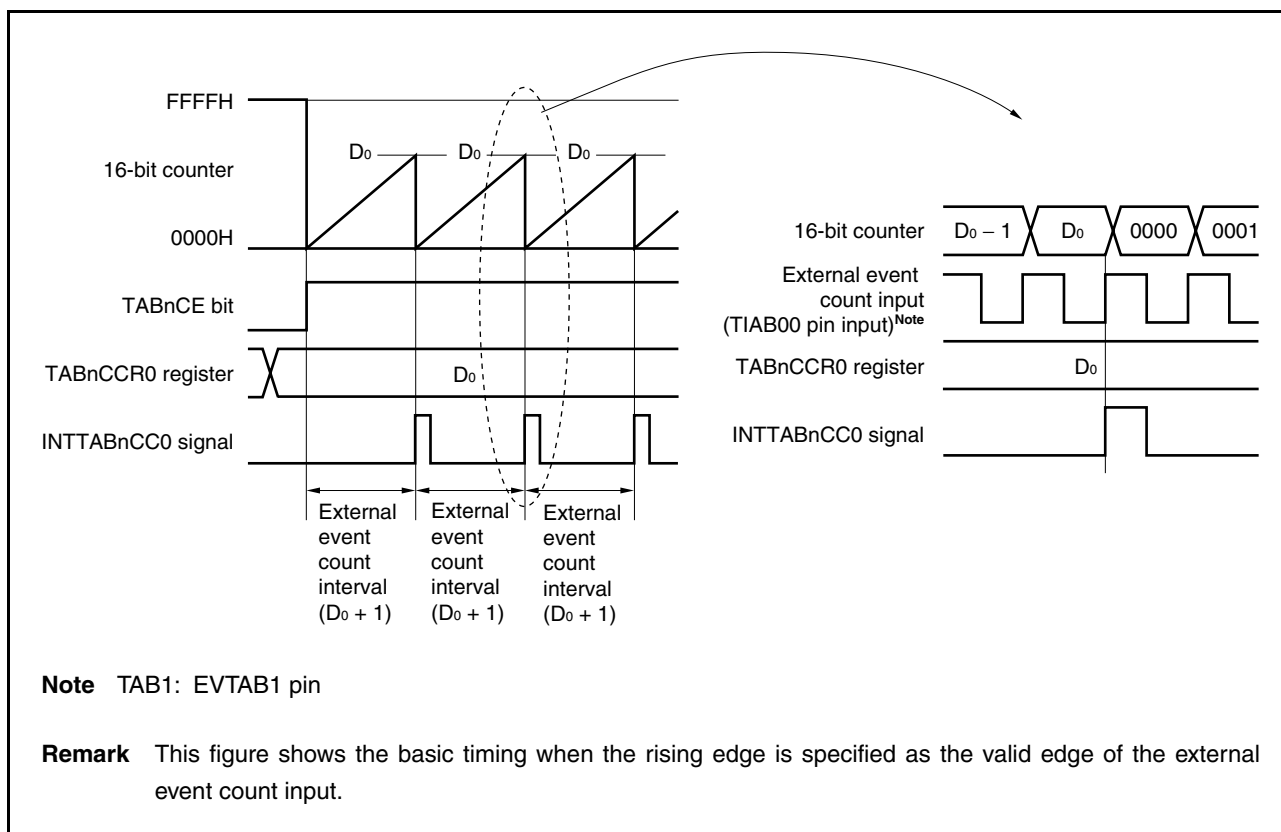


**Note** TAB1: EVTAB1 pin

**Remark** This figure shows the basic timing when the rising edge is specified as the valid edge of the external event count input.

When the TABnCE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H.  The counter counts each time the valid edge of the external event count input is detected.  Additionally, the set value of the TABnCCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, and a compare match interrupt request signal (INTTABnCC0) is generated.

The INTTABnCC0 signal is generated each time the valid edge of the external event count input has been detected (set value of TABnCCR0 register + 1) times.

**Figure 8-11.  Register Setting for Operation in External Event Count Mode (1/2)**
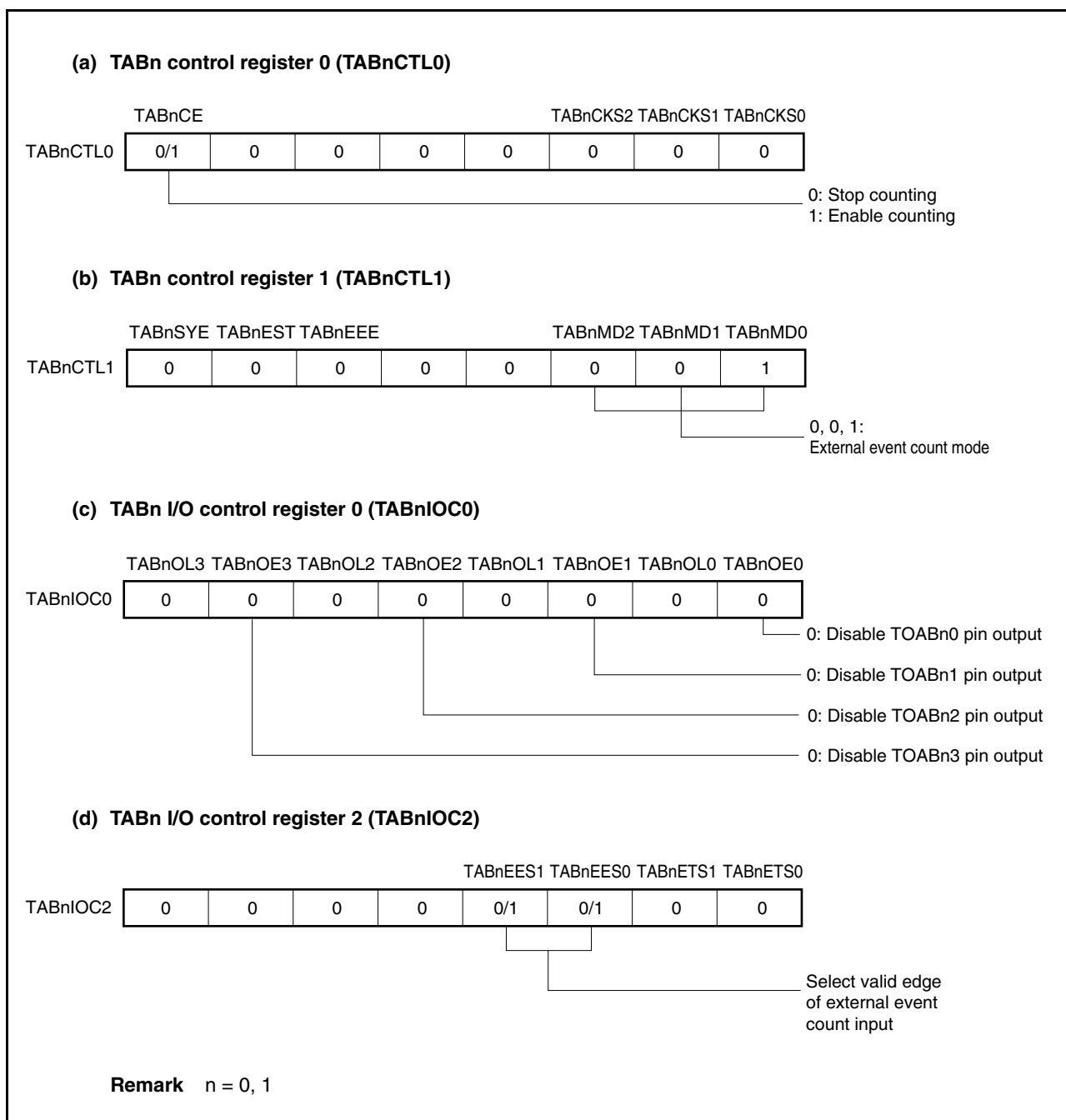
**Figure 8-11.  Register Setting for Operation in External Event Count Mode (2/2)**

---

**(e) TABn counter read buffer register (TABnCNT)**

The count value of the 16-bit counter can be read by reading the TABnCNT register.

**(f) TABn capture/compare register 0 (TABnCCR0)**

If $D_0$ is set to the TABnCCR0 register, the counter is cleared and a compare match interrupt request signal (INTTABnCC0) is generated when the number of external event counts reaches ($D_0 + 1$).

**(g) TABn capture/compare registers 1 to 3 (TABnCCR1 to TABnCCR3)**

Usually, the TABnCCR1 to TABnCCR3 registers are not used in the external event count mode. However, the set value of the TABnCCR1 to TABnCCR3 registers are transferred to the CCR1 to CCR3 buffer registers.  When the count value of the 16-bit counter matches the value of the CCR1 to CCR3 buffer registers, compare match interrupt request signals (INTTABnCC1 to INTTABnCC3) are generated.

Therefore, mask the interrupt signals by using the interrupt mask flags (TABnCCMK1 to TABnCCMK3).

**Caution**  **For TAB0, when an external clock is used as the count clock, the external clock can be input only from the TIAB00 pin.  At this time, set the TAB0IOC1.TAB0IS1 and TAB0IOC1.TAB0IS0 bits to 00 (capture trigger input (TIAB00 pin): no edge detection).**

**Remarks 1.** TABn I/O control register 1 (TABnIOC1) and TABn option register 0 (TABnOPT0) are not used in the external event count mode.

**2.** n = 0, 1

---

**(1) External event count mode operation flow**

**Figure 8-12. Flow of Software Processing in External Event Count Mode**
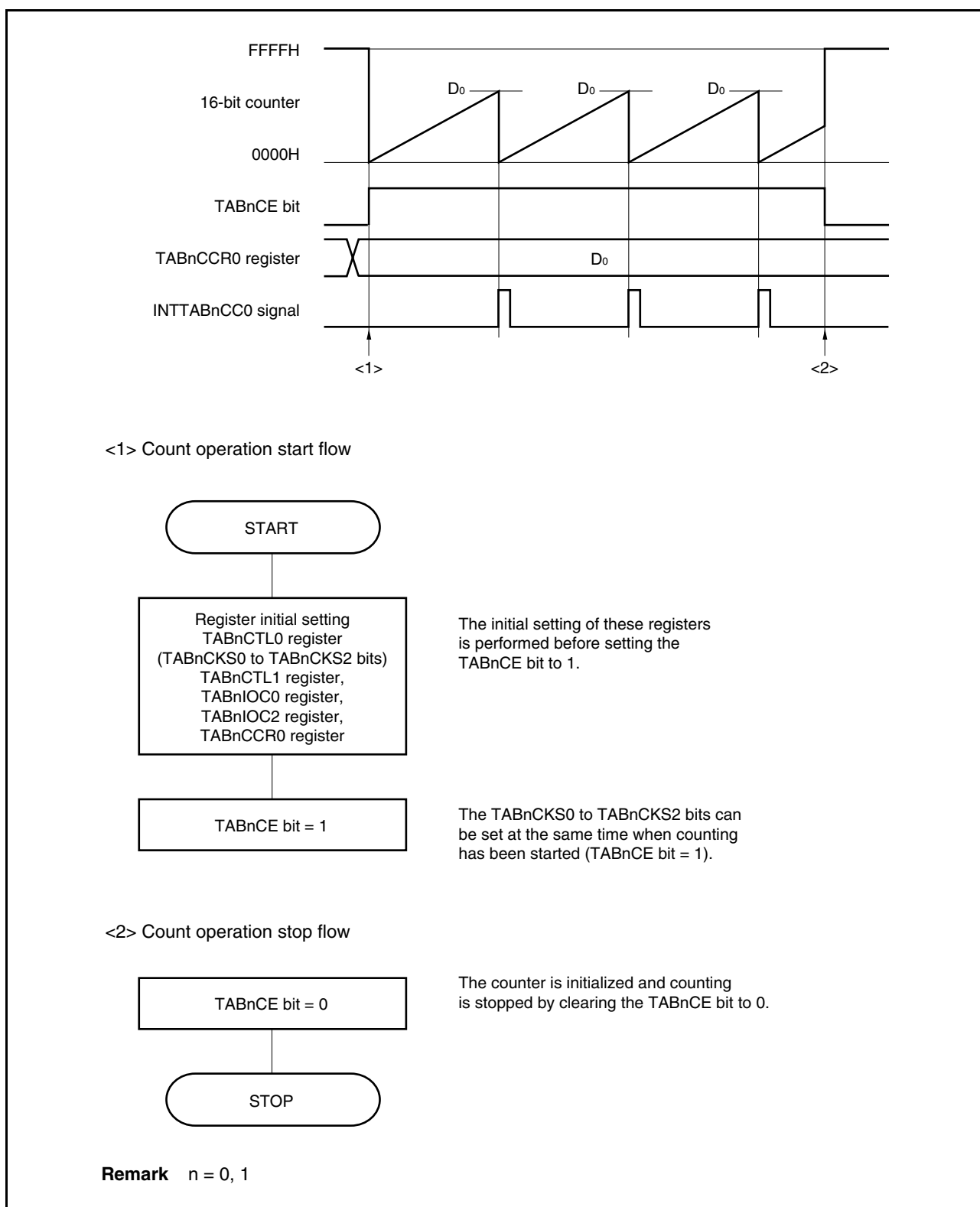


**Remark**    n = 0, 1

**(2)  Operation timing in external event count mode**

**Cautions 1.  In the external event count mode, do not set the TABnCCR0 register to 0000H.**

**2.  In the external event count mode, use of the timer output is disabled.  If performing timer output using external event count input, set the interval timer mode, and select the operation enabled by the external event count input for the count clock (TABnCTL1.TABnMD2 to TABnCTL1.TABnMD0 bits = 000, TABnCTL1.TABnEEE bit = 1).**

**(a)  Operation if TABnCCR0 register is set to FFFFH**

If the TABnCCR0 register is set to FFFFH, the 16-bit counter counts to FFFFH each time the valid edge of the external event count signal has been detected.  The 16-bit counter is cleared to 0000H in synchronization with the next count-up timing, and the INTTABnCC0 signal is generated.  At this time, the TABnOPT0.TABnOVF bit is not set.
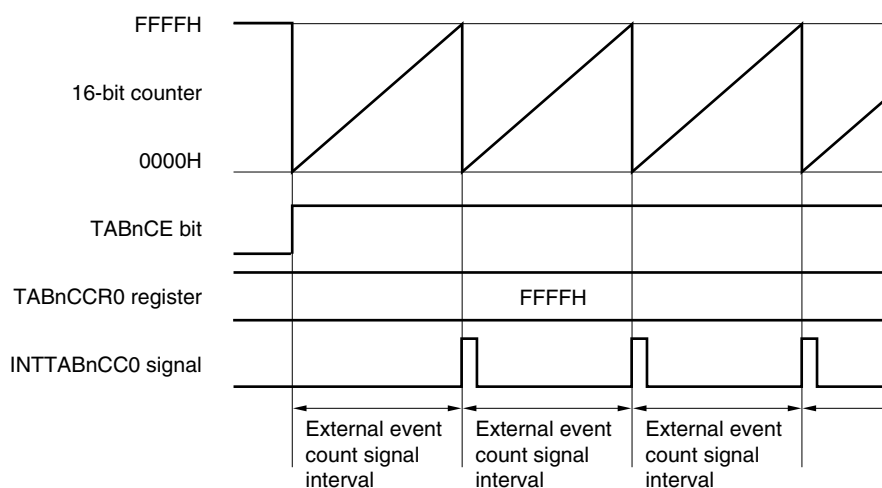


**Remark**   n = 0, 1

**(b) Notes on rewriting the TABnCCR0 register**

To change the value of the TABnCCR0 register to a smaller value, stop counting once and then change the set value.

If the value of the TABnCCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow.



**Remark**  n = 0, 1

If the value of the TABnCCR0 register is changed from $D_1$ to $D_2$ while the count value is greater than $D_2$ but less than $D_1$, the count value is transferred to the CCR0 buffer register as soon as the TABnCCR0 register has been rewritten. Consequently, the value that is compared with the 16-bit counter is $D_2$.

Because the count value has already exceeded $D_2$, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H. When the count value matches $D_2$, the INTTABnCC0 signal is generated.

Therefore, the INTTABnCC0 signal may not be generated at the valid edge count of "$(D_1 + 1)$ times" or "$(D_2 + 1)$ times" originally expected, but may be generated at the valid edge count of "$(10000H + D_2 + 1)$ times".

**(c)  Operation of TABnCCR1 to TABnCCR3 registers**

**Figure 8-13.  Configuration of TABnCCR1 to TABnCCR3 Registers**



**Note**  TAB1:  EVTAB1 pin

**Remark**  n = 0, 1
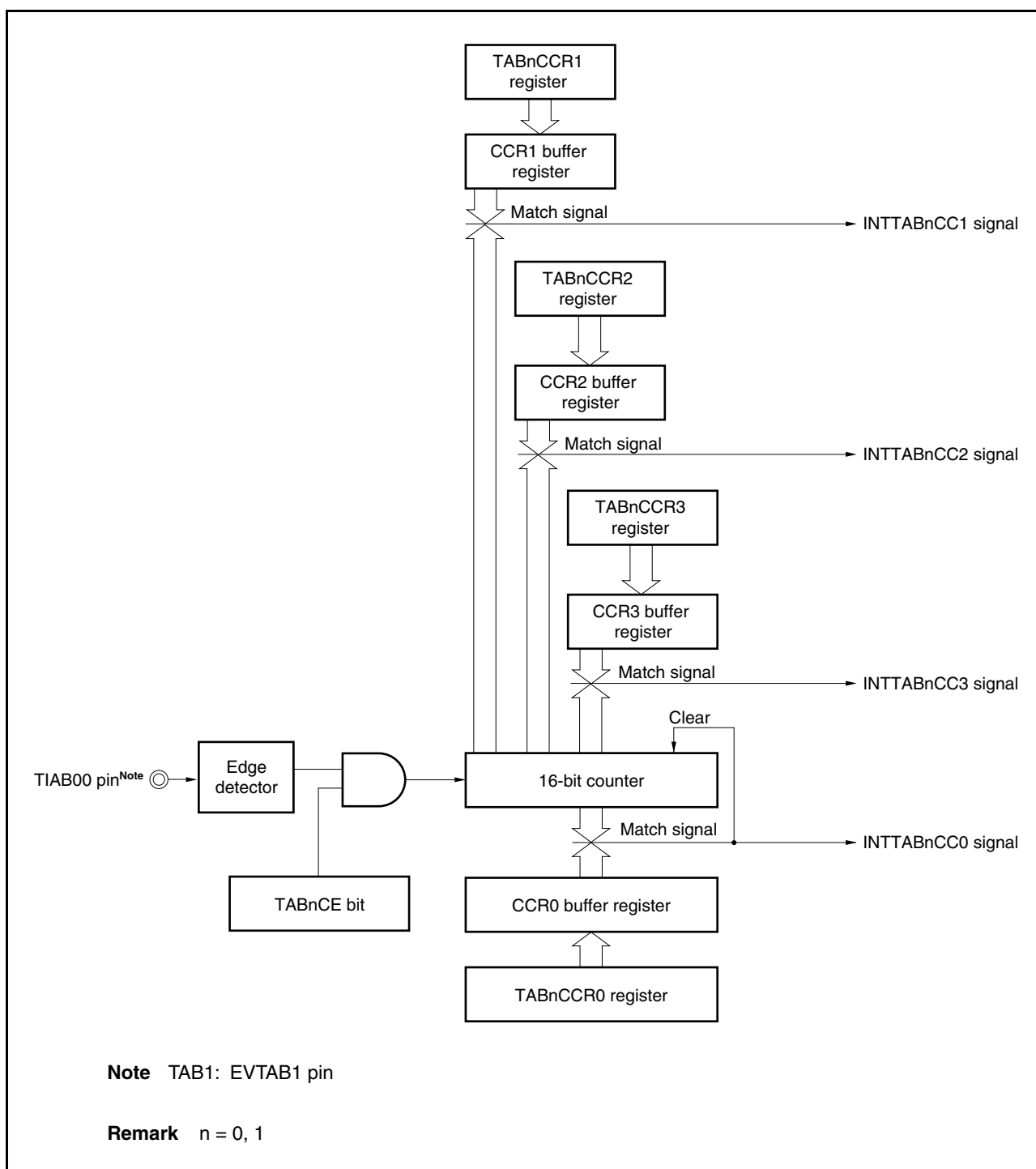
If the set value of the TABnCCRk register is smaller than the set value of the TABnCCR0 register, the INTTABnCCk signal is generated once per cycle.

**Remark**  k = 1 to 3,
            n = 0, 1

**Figure 8-14. Timing Chart When $D_{01} \geq D_{k1}$**
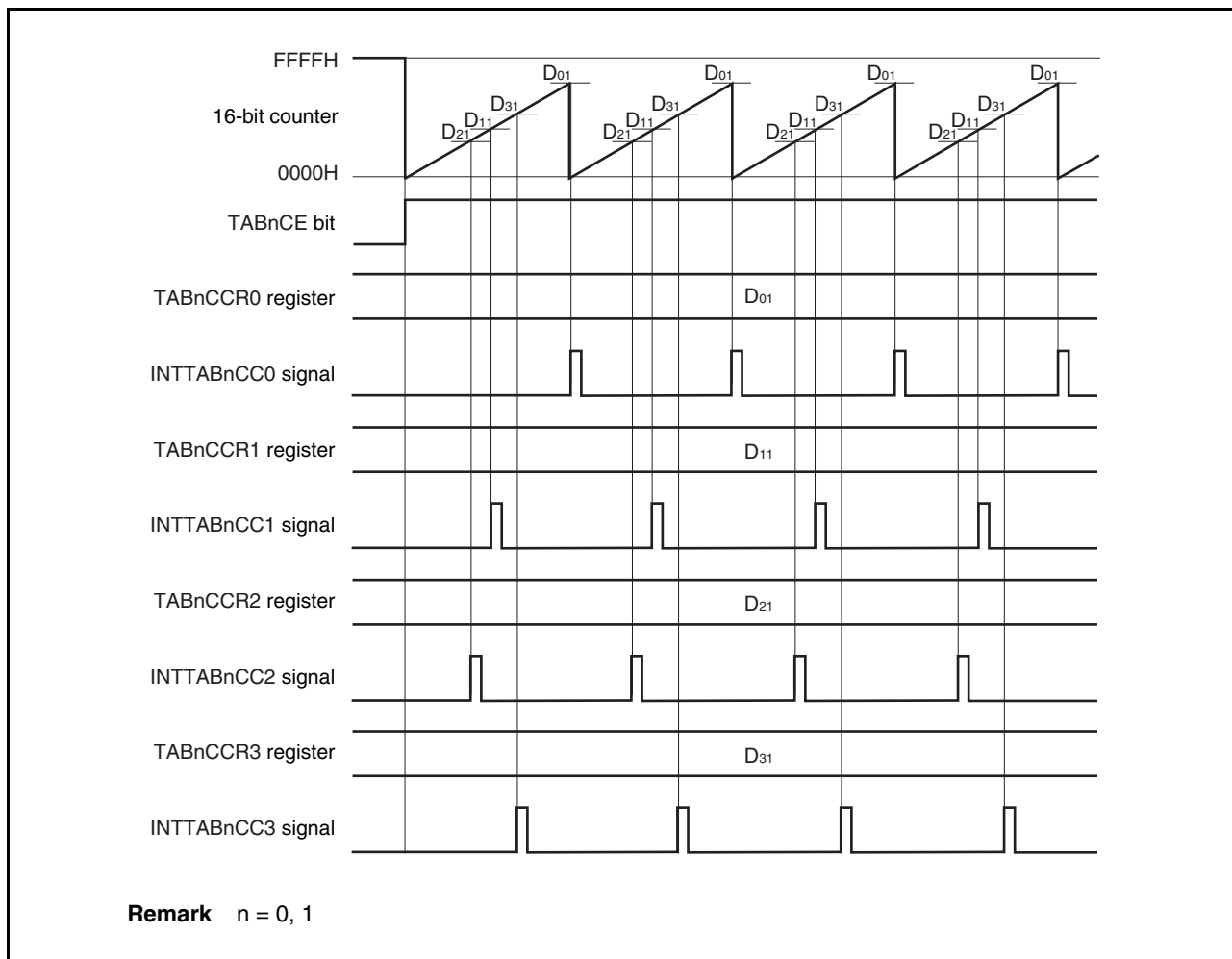


**Remark**  n = 0, 1

If the set value of the TABnCCRk register is greater than the set value of the TABnCCR0 register, the INTTABnCCk signal is not generated because the count value of the 16-bit counter and the value of the TABnCCRk register do not match.

**Remark** k = 1 to 3,
     n = 0, 1

**Figure 8-15. Timing Chart When $D_{01} < D_{k1}$**



**Remark** n = 0, 1
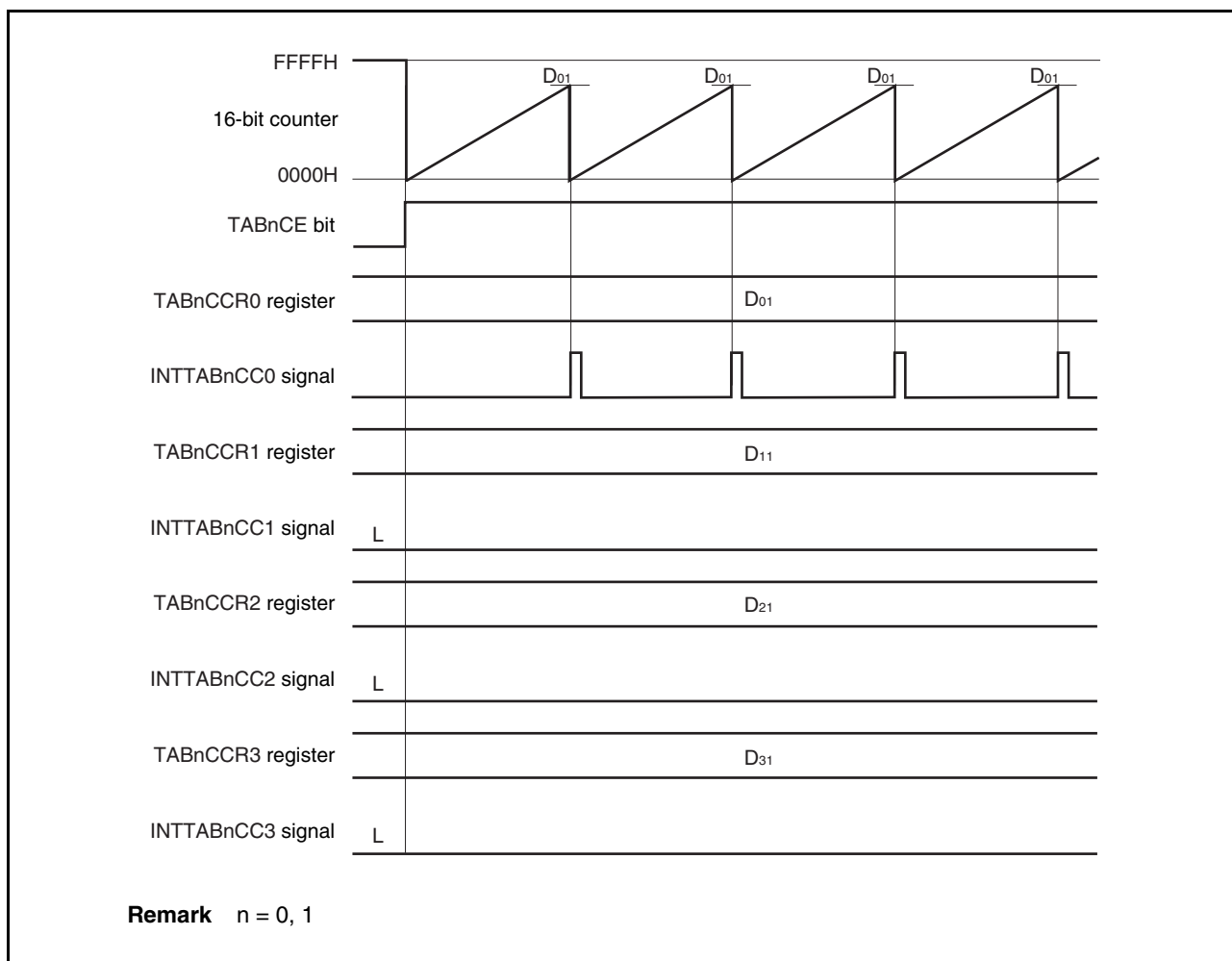
### 8.5.3 External trigger pulse output mode (TABnMD2 to TABnMD0 bits = 010)

In the external trigger pulse output mode, TABn waits for a trigger when the TABnCTL0.TABnCE bit is set to 1. When the valid edge of the external trigger input signal is detected, TABn starts counting, and outputs a PWM waveform from the TOABn1 to TOABn3 pins.

Pulses can also be output by generating a software trigger instead of using the external trigger. When using a software trigger, a square wave that has one cycle of the PWM waveform as half its cycle can also be output from the TOABn0 pin.

**Figure 8-16. Configuration in External Trigger Pulse Output Mode**



**Note**    TAB1: TRGAB1 pin

**Remark**    n = 0, 1
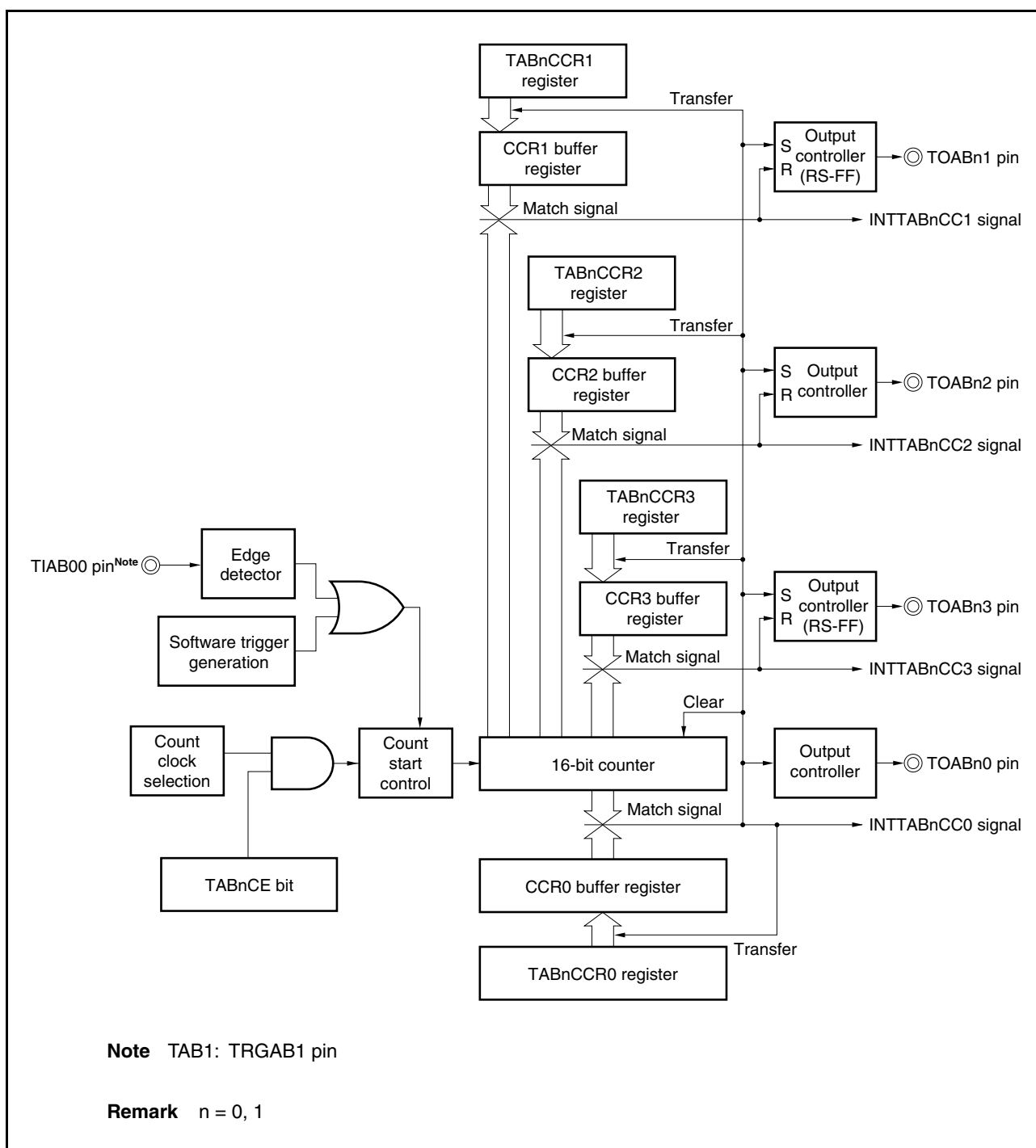
**Figure 8-17.  Basic Timing in External Trigger Pulse Output Mode**



**Note**    TAB1: TRGAB1 pin

**Remark**    n = 0, 1
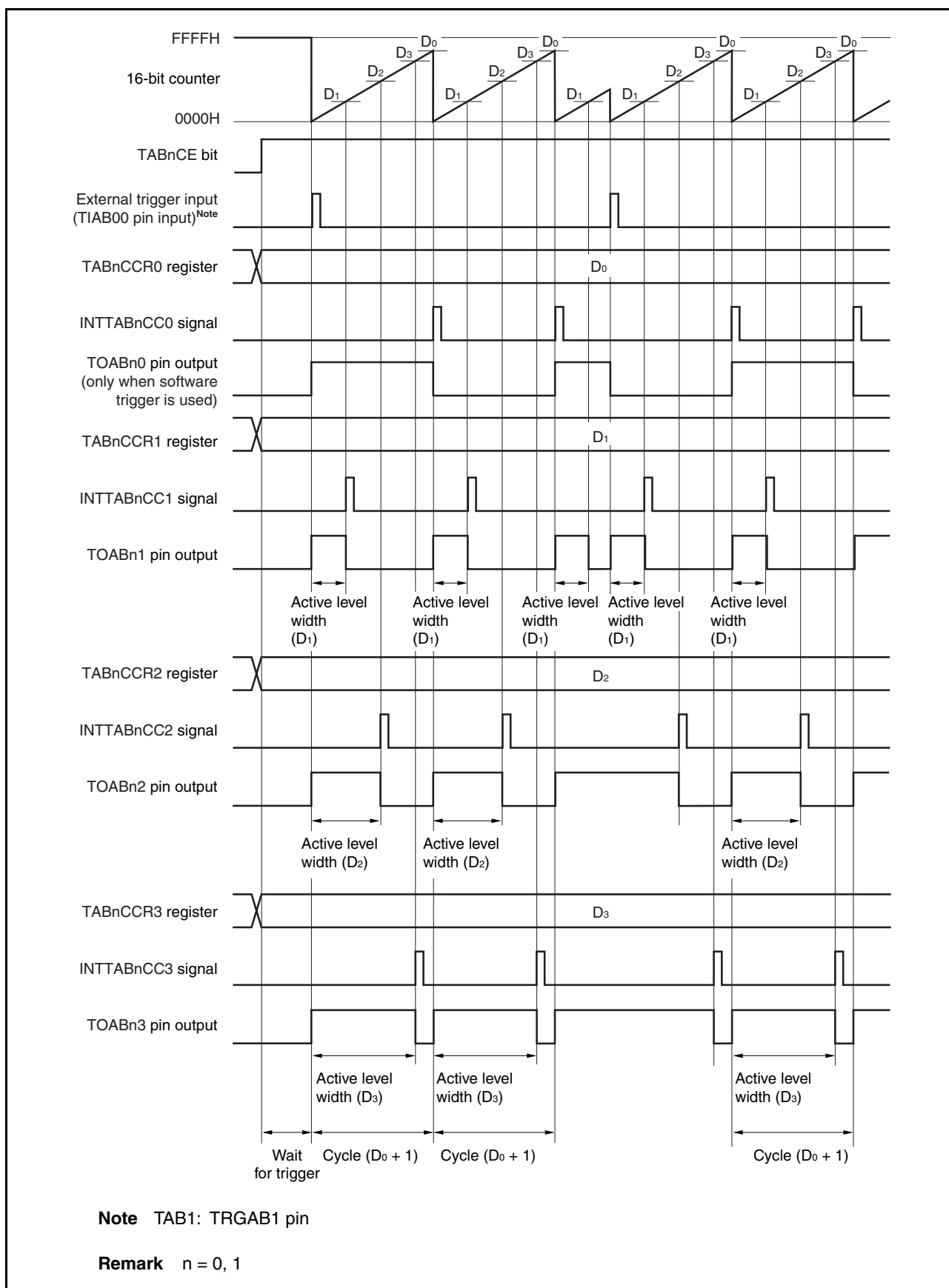
TABn waits for a trigger when the TABnCE bit is set to 1. When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting at the same time, and outputs a PWM waveform from the TOABnk pin. If the trigger is generated again while the counter is operating, the counter is cleared to 0000H and restarted. (The output of the TOABn0 pin is inverted. The TOABnk pin outputs a high level regardless of the status (high/low) when a trigger is generated.)

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

Active level width = (Set value of TABnCCRk register) × Count clock cycle
Cycle = (Set value of TABnCCR0 register + 1) × Count clock cycle
Duty factor = (Set value of TABnCCRk register)/(Set value of TABnCCR0 register + 1)

The compare match request signal (INTTABnCC0) is generated when the 16-bit counter counts up next time after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H. The compare match interrupt request signal (INTTABnCCk) is generated when the count value of the 16-bit counter matches the value of the CCRk buffer register.

The value set to the TABnCCRm register is transferred to the CCRm buffer register when the count value of the 16-bit counter matches the value of the CCR0 buffer register and the 16-bit counter is cleared to 0000H.

The valid edge of the external trigger input signal or setting the software trigger (TABnCTL1.TABnEST bit) to 1 is used as the trigger.

**Remark**  k = 1 to 3,
          m = 0 to 3,
          n = 0, 1

**Figure 8-18.  Setting of Registers in External Trigger Pulse Output Mode (1/3)**



**Note**  The setting is invalid when the TABnCTL1.TABnEEE bit = 1.

**Remark**  n = 0, 1
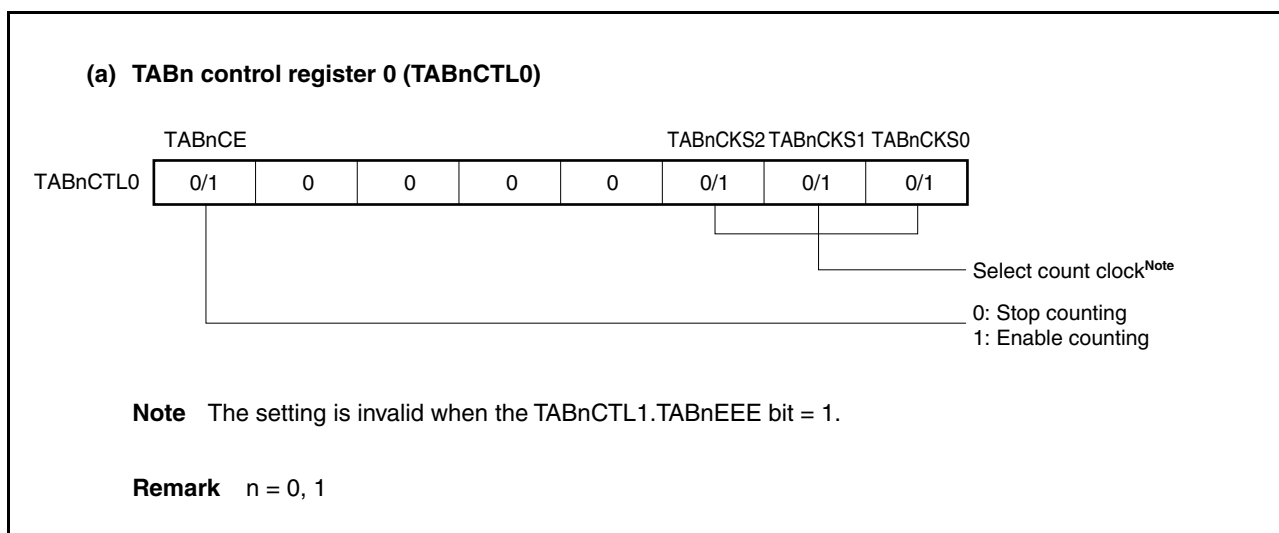
**Figure 8-18. Setting of Registers in External Trigger Pulse Output Mode (2/3)**

**(b) TABn control register 1 (TABnCTL1)**

|  | TABnSYE | TABnEST | TABnEEE |  |  | TABnMD2 | TABnMD1 | TABnMD0 |
|---|---|---|---|---|---|---|---|---|
| TABnCTL1 | 0 | 0/1 | 0/1 | 0 | 0 | 0 | 1 | 0 |

0, 1, 0:
External trigger pulse
output mode

0: Operate on count clock
　selected by TABnCKS0 to
　TABnCKS2 bits
1: Count by external event
　input signal

Generate software trigger
when 1 is written

**(c) TABn I/O control register 0 (TABnIOC0)**

|  | TABnOL3 | TABnOE3 | TABnOL2 | TABnOE2 | TABnOL1 | TABnOE1 | TABnOL0 | TABnOE0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC0 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 Note | 0/1 Note |

0: Disable TOABn0 pin output
1: Enable TOABn0 pin output

Setting of output level while
operation of TOABn0 pin is disabled
0: Low level
1: High level

0: Disable TOABn1 pin output
1: Enable TOABn1 pin output

Specification of active level
of TOABn1 pin output
0: Active-high
1: Active-low

0: Disable TOABn2 pin output
1: Enable TOABn2 pin output

Specification of active level
of TOABn2 pin output
0: Active-high
1: Active-low

0: Disable TOABn3 pin output
1: Enable TOABn3 pin output

Specification of active level
of TOABn3 pin output
0: Active-high
1: Active-low

- When TABnOLk bit = 0

16-bit counter

TOABnk pin output

- When TABnOLk bit = 1

16-bit counter

TOABnk pin output

**Note** Clear this bit to 0 when the TOABn0 pin is not used in the external trigger pulse output mode.

**Remark** n = 0, 1

**Figure 8-18.  Setting of Registers in External Trigger Pulse Output Mode (3/3)**

**(d)  TABn I/O control register 2 (TABnIOC2)**
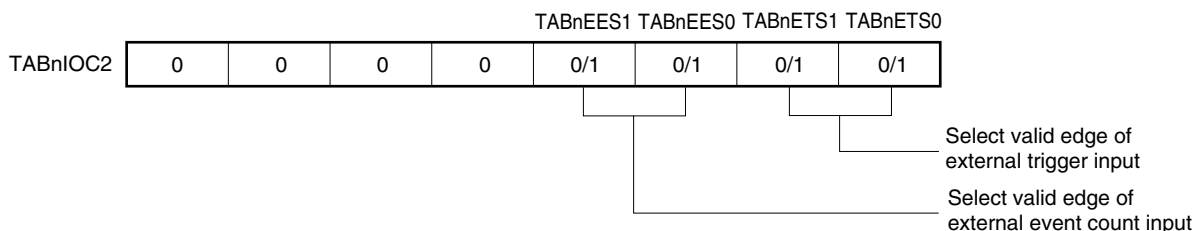
|  | | | | TABnEES1 | TABnEES0 | TABnETS1 | TABnETS0 |
|---|---|---|---|---|---|---|---|
| TABnIOC2 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |

Select valid edge of external trigger input

Select valid edge of external event count input

**(e)  TABn counter read buffer register (TABnCNT)**

The value of the 16-bit counter can be read by reading the TABnCNT register.

**(f)  TABn capture/compare registers 0 to 3 (TABnCCR0 to TABnCCR3)**

If $D_0$ is set to the TABnCCR0 register, $D_1$ to the TABnCCR1 register, $D_2$ to the TABnCCR2 register, and $D_3$ to the TABnCCR3 register, the cycle and active level of the PWM waveform are as follows.

Cycle = $(D_0 + 1) \times$ Count clock cycle

TOABn1 pin PWM waveform active level width = $D_1 \times$ Count clock cycle

TOABn2 pin PWM waveform active level width = $D_2 \times$ Count clock cycle

TOABn3 pin PWM waveform active level width = $D_3 \times$ Count clock cycle

**Remarks  1.** TABn I/O control register 1 (TABnIOC1) and TABn option register 0 (TABnOPT0) are not used in the external trigger pulse output mode.

**2.** Updating TABn capture/compare register 2 (TABnCCR2) and TABn capture/compare register 3 (TABnCCR3) is enabled by writing TABn capture/compare register 1 (TABnCCR1).

**3.** n = 0, 1

**(1) Operation flow in external trigger pulse output mode**

**Figure 8-19. Software Processing Flow in External Trigger Pulse Output Mode (1/2)**



**Note**    TAB1: TRGAB1 pin

**Remark**    n = 0, 1

**Figure 8-19.  Software Processing Flow in External Trigger Pulse Output Mode (2/2)**

<1> Count operation start flow

```
        ┌─────────────────┐
        │      START      │
        └─────────────────┘
                 │
   ┌──────────────────────────┐      The initial setting of these
   │  Register initial setting │      registers is performed
   │   TABnCTL0 register       │      before setting the
   │ (TABnCKS0 to TABnCKS2     │      TABnCE bit to 1.
   │      bits),               │
   │   TABnCTL1 register,      │
   │   TABnIOC0 register,      │
   │   TABnIOC2 register,      │
   │  TABnCCR0 to TABnCCR3     │
   │      registers            │
   └──────────────────────────┘
                 │
   ┌──────────────────────────┐      The TABnCKS0 to
   │     TABnCE bit = 1        │      TABnCKS2 bits can be
   └──────────────────────────┘      set at the same time
                                     when counting is
                                     enabled (TABnCE bit = 1).
                                     Trigger wait status
```

<2> TABnCCR0 to TABnCCR3 register
    setting change flow

```
   ┌──────────────────────────┐      Writing the TABnCCR1
   │ Setting of TABnCCR0, TABnCCR2, │  register must be performed
   │  and TABnCCR3 registers   │      after writing the TABnCCR0,
   └──────────────────────────┘      TABnCCR2, and TABnCCR3
                 │                    registers.
   ┌──────────────────────────┐      When the counter is cleared
   │ Setting of TABnCCR1 register │   after setting, the value
   └──────────────────────────┘      of the TABnCCRm register is
                                     transferred to the CCRm buffer
                                     registers.
```

<3> TABnCCR0 register setting change flow

```
   ┌──────────────────────────┐      Writing the same value to
   │ Setting of TABnCCR0 register │   the TABnCCR1 register is
   └──────────────────────────┘      necessary only when the
                 │                    set cycle is changed.
   ┌──────────────────────────┐
   │ Setting of TABnCCR1 register │   When the counter is
   └──────────────────────────┘      cleared after setting,
                                     the value of the TABnCCRm
                                     register is transferred to
                                     the CCRm buffer register.
```

<4> TABnCCR1 to TABnCCR3 register
    setting change flow

```
   ┌──────────────────────────┐      Writing the TABnCCR1
   │ Setting of TABnCCR2,      │      register must be performed
   │  TABnCCR3 registers       │      only when the set duty factor is
   └──────────────────────────┘      changed after writing the
                 │                    TABnCCR2 and TABnCCR3
   ┌──────────────────────────┐      registers.
   │ Setting of TABnCCR1 register │   When the counter is cleared
   └──────────────────────────┘      after setting, the value of the
                                     TABnCCRm register is transferred
                                     to the CCRm buffer register.
```

<5> TABnCCR2, TABnCCR3 register
    setting change flow

```
   ┌──────────────────────────┐      Writing the same value to the
   │ Setting of TABnCCR2,      │      TABnCCR1 register is necessary
   │  TABnCCR3 registers       │      only when the set duty factor of
   └──────────────────────────┘      the TOABn2 and TOABn3 pin
                 │                    outputs is changed.
   ┌──────────────────────────┐      When the counter is cleared
   │ Setting of TABnCCR1 register │   after setting, the value of the
   └──────────────────────────┘      TABnCCRm register
                                     is transferred to the CCRm buffer
                                     register.
```

<6> TABnCCR1 register setting change flow

```
                                     The TABnCCR1 register only needs
                                     to be written, only when the set duty
   ┌──────────────────────────┐      factor of the TOABn1 pin output is
   │ Setting of TABnCCR1 register │   changed.
   └──────────────────────────┘      When the counter is cleared after
                                     setting, the value of the TABnCCRm
                                     register is transferred to the CCRm
                                     buffer register.
```

<7> Count operation stop flow

```
   ┌──────────────────────────┐
   │     TABnCE bit = 0        │      Counting is stopped.
   └──────────────────────────┘
                 │
        ┌─────────────────┐
        │      STOP       │
        └─────────────────┘
```

**Remark**   m = 0 to 3,
             n = 0, 1

**(2) External trigger pulse output mode operation timing**

**(a) Note on changing pulse width during operation**

To change the PWM waveform while the counter is operating, write the TABnCCR1 register last.

Rewrite the TABnCCRk register after writing the TABnCCR1 register after the INTTABnCC0 signal is detected.



**Note** TAB1: TRGAB1 pin

**Remark** n = 0, 1

To transfer data from the TABnCCRm register to the CCRm buffer register, the TABnCCR1 register must be written.

To change both the cycle and active level width of the PWM waveform at this time, first set the cycle to the TABnCCR0 register, set the active level width to the TABnCCR2 and TABnCCR3 registers, and then set the active level to the TABnCCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TABnCCR0 register, and then write the same value to the TABnCCR1 register.

To change only the active level width (duty factor) of the PWM waveform, first set the active level to the TABnCCR2 and TABnCCR3 registers and then set the active level to the TABnCCR1 register.

To change only the active level width (duty factor) of the PWM waveform output by the TOABn1 pin, only the TABnCCR1 register has to be set.

To change only the active level width (duty factor) of the PWM waveform output by the TOABn2 and TOABn3 pins, first set the active level width to the TABnCCR2 and TABnCCR3 registers, and then write the same value to the TABnCCR1 register.

After data is written to the TABnCCR1 register, the value written to the TABnCCRm register is transferred to the CCRm buffer register in synchronization with clearing of the 16-bit counter, and is used as the value to be compared with the 16-bit counter.

To write the TABnCCR0 to TABnCCR3 registers again after writing the TABnCCR1 register once, do so after the INTTABnCC0 signal is generated. Otherwise, the value of the CCRm buffer register may become undefined because the timing of transferring data from the TABnCCRm register to the CCRm buffer register conflicts with writing the TABnCCRm register.

**Remark**   m = 0 to 3,
n = 0, 1

**(b) 0%/100% output of PWM waveform**

To output a 0% waveform, set the TABnCCRk register to 0000H.  If the set value of the TABnCCR0 register is FFFFH, the INTTABnCCk signal is generated periodically.
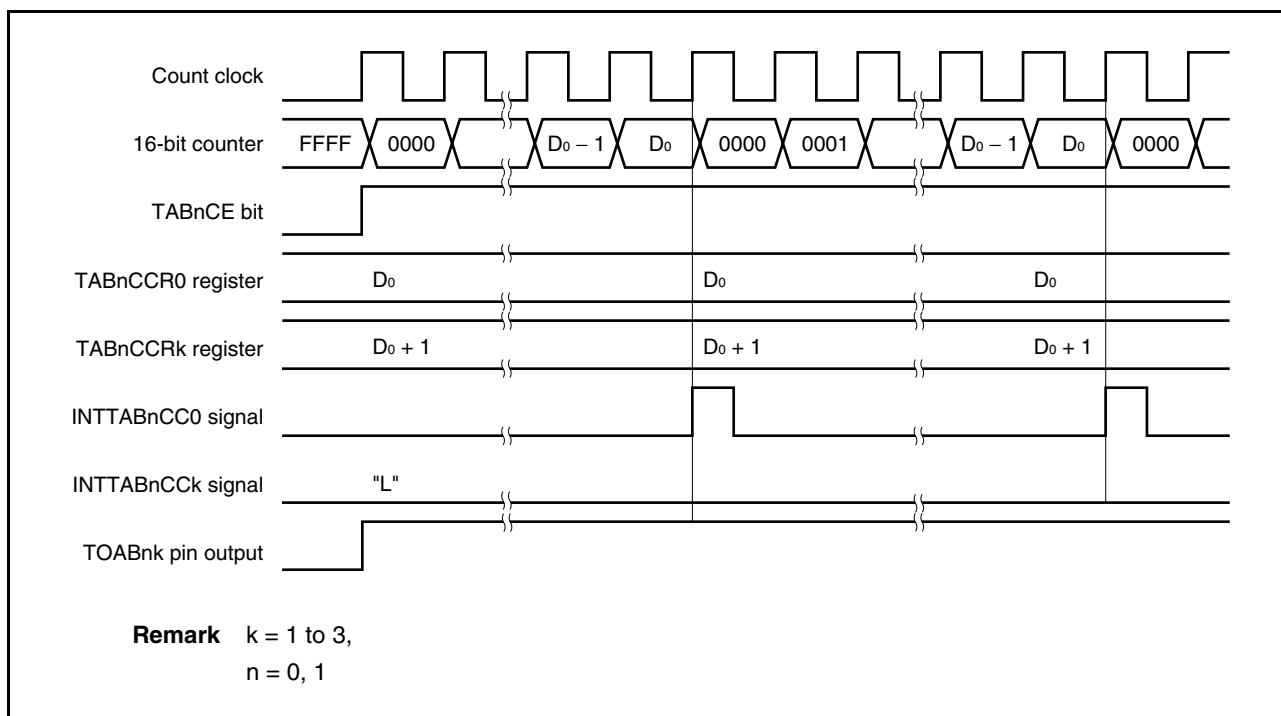


Count clock

16-bit counter   FFFF   0000   $D_0 - 1$   $D_0$   0000   0001   $D_0 - 1$   $D_0$   0000

TABnCE bit

TABnCCR0 register   $D_0$   $D_0$   $D_0$

TABnCCRk register   0000H   0000H   0000H

INTTABnCC0 signal

INTTABnCCk signal

TOABnk pin output   "L"

**Remark**   k = 1 to 3,
n = 0, 1

To output a 100% waveform, set a value of "set value of TABnCCR0 register + 1" to the TABnCCRk register.  If the set value of the TABnCCR0 register is FFFFH, 100% output cannot be produced.



Count clock

16-bit counter   FFFF   0000   $D_0 - 1$   $D_0$   0000   0001   $D_0 - 1$   $D_0$   0000

TABnCE bit

TABnCCR0 register   $D_0$   $D_0$   $D_0$

TABnCCRk register   $D_0 + 1$   $D_0 + 1$   $D_0 + 1$

INTTABnCC0 signal

INTTABnCCk signal   "L"

TOABnk pin output

**Remark**   k = 1 to 3,
n = 0, 1

**(c) Conflict between trigger detection and match with CCRk buffer register**

If the trigger is detected immediately after the INTTABnCCk signal is generated, the 16-bit counter is immediately cleared to 0000H, the output signal of the TOABnk pin is asserted, and the counter continues counting. Consequently, the inactive period of the PWM waveform is shortened.
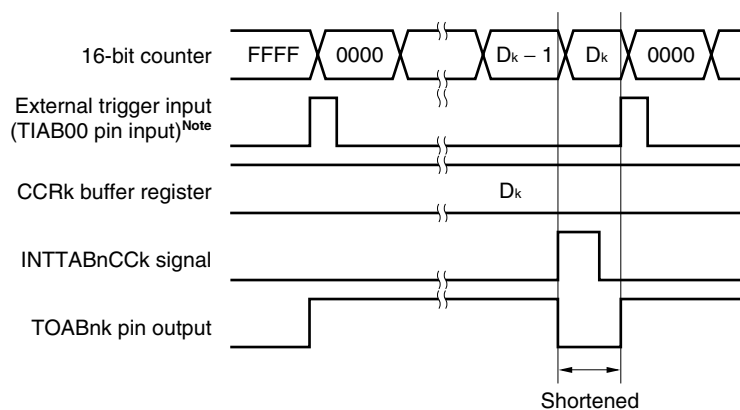


**Note**    TAB1: TRGAB1 pin

**Remark**    k = 1 to 3,
           n = 0, 1

If the trigger is detected immediately before the INTTABnCCk signal is generated, the INTTABnCCk signal is not generated, and the 16-bit counter is cleared to 0000H and continues counting. The output signal of the TOABnk pin remains active. Consequently, the active period of the PWM waveform is extended.



**Note**    TAB1: TRGAB1 pin

**Remark**    k = 1 to 3,
           n = 0, 1

**(d) Conflict between trigger detection and match with CCR0 buffer register**

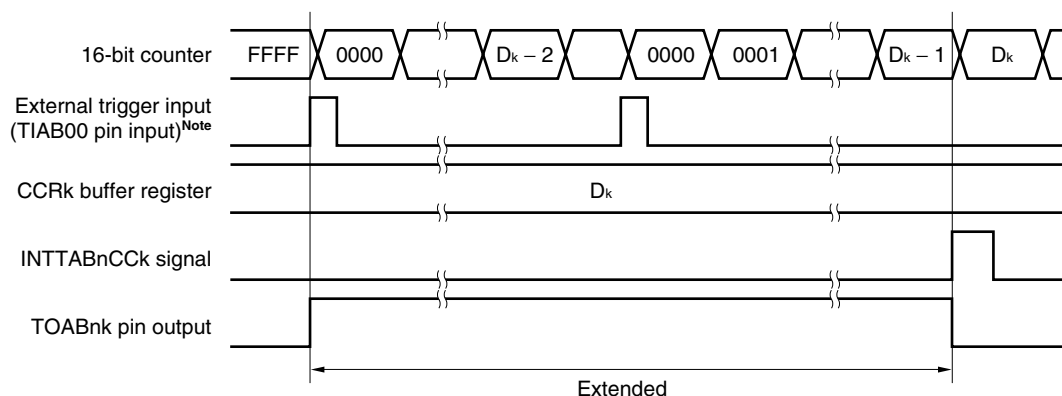If the trigger is detected immediately after the INTTABnCC0 signal is generated, the 16-bit counter is cleared to 0000H and continues counting up. Therefore, the active period of the TOABnk pin is extended by time from generation of the INTTABnCC0 signal to trigger detection.
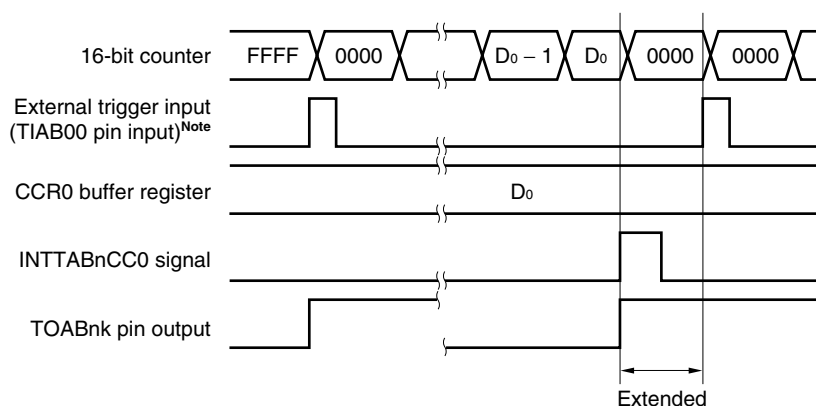


**Note**   TAB1: TRGAB1 pin

**Remark**   k = 1 to 3,
n = 0, 1

If the trigger is detected immediately before the INTTABnCC0 signal is generated, the INTTABnCC0 signal is not generated. The 16-bit counter is cleared to 0000H, the TOABnk pin is asserted, and the counter continues counting. Consequently, the inactive period of the PWM waveform is shortened.
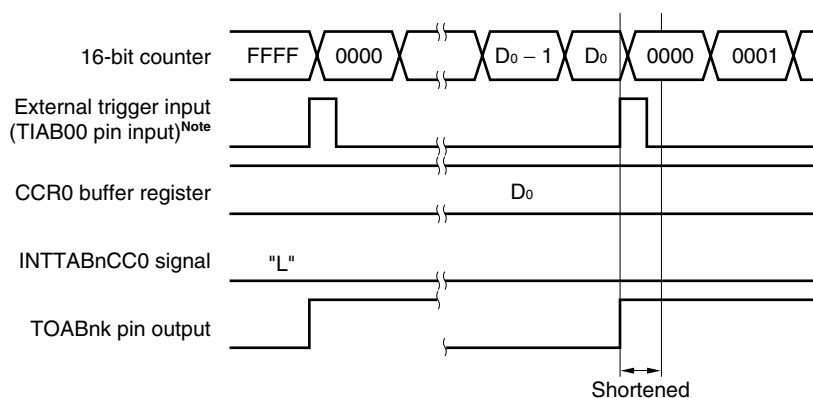


**Note**   TAB1: TRGAB1 pin

**Remark**   k = 1 to 3,
n = 0, 1

**(e) Generation timing of compare match interrupt request signal (INTTABnCCk)**

The timing of generation of the INTTABnCCk signal in the external trigger pulse output mode differs from the timing of other INTTABnCCk signals; the INTTABnCCk signal is generated when the count value of the 16-bit counter matches the value of the CCRk buffer register.



**Remark**  k = 1 to 3,
            n = 0, 1
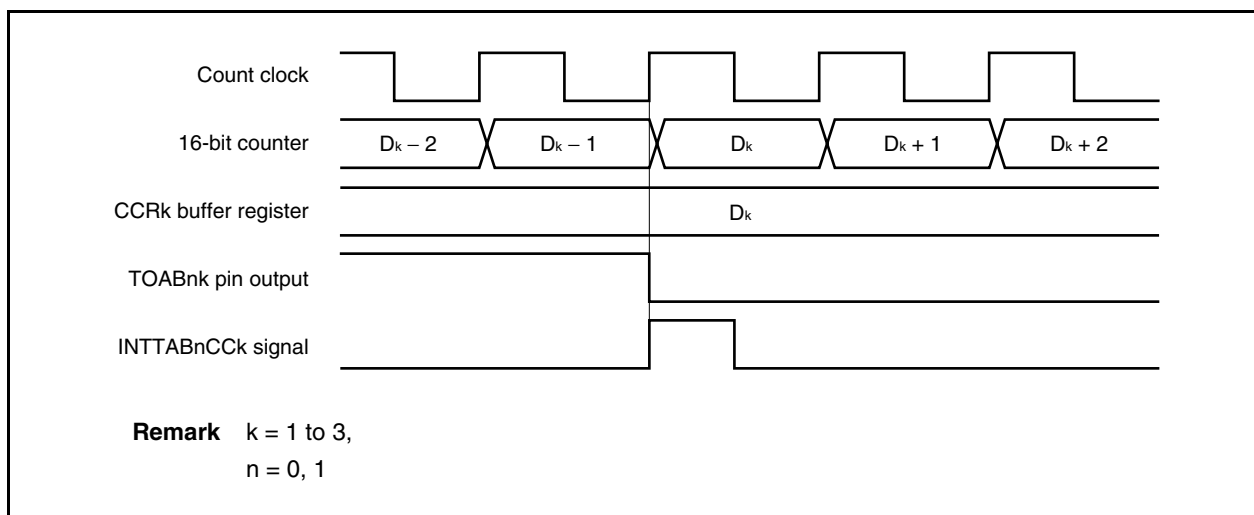
Usually, the INTTABnCCk signal is generated in synchronization with the next count-up after the count value of the 16-bit counter matches the value of the CCRk buffer register.

In the external trigger pulse output mode, however, it is generated one clock earlier.  This is because the timing is changed to match the timing of changing the output signal of the TOABnk pin.

### 8.5.4　One-shot pulse output mode (TABnMD2 to TABnMD0 bits = 011)

In the one-shot pulse output mode, TABn waits for a trigger when the TABnCTL0.TABnCE bit is set to 1. When the valid edge of the external trigger input is detected, TABn starts counting, and outputs a one-shot pulse from the TOABn1 to TOABn3 pins.

Instead of the external trigger, a software trigger can also be generated to output the pulse. When the software trigger is used, the TOABn0 pin outputs the active level while the 16-bit counter is counting, and the inactive level when the counter is stopped (waiting for a trigger).

**Figure 8-20.  Configuration in One-Shot Pulse Output Mode**



**Note**　TAB1:  TRGAB1 pin

**Remark**　n = 0, 1
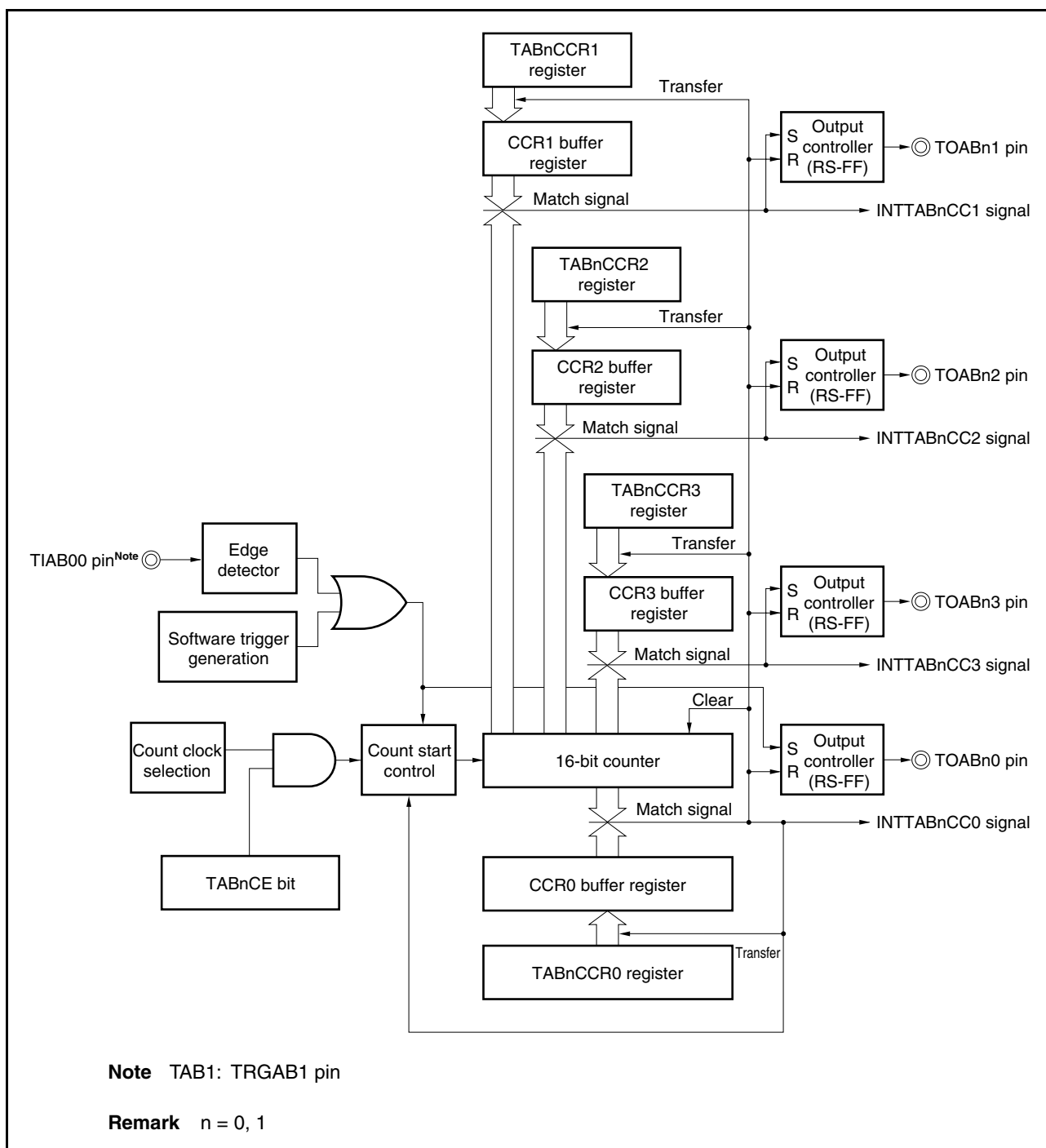
**Figure 8-21.  Basic Timing in One-Shot Pulse Output Mode**



**Note**   TAB1:  TRGAB1 pin

**Remark**   n = 0, 1
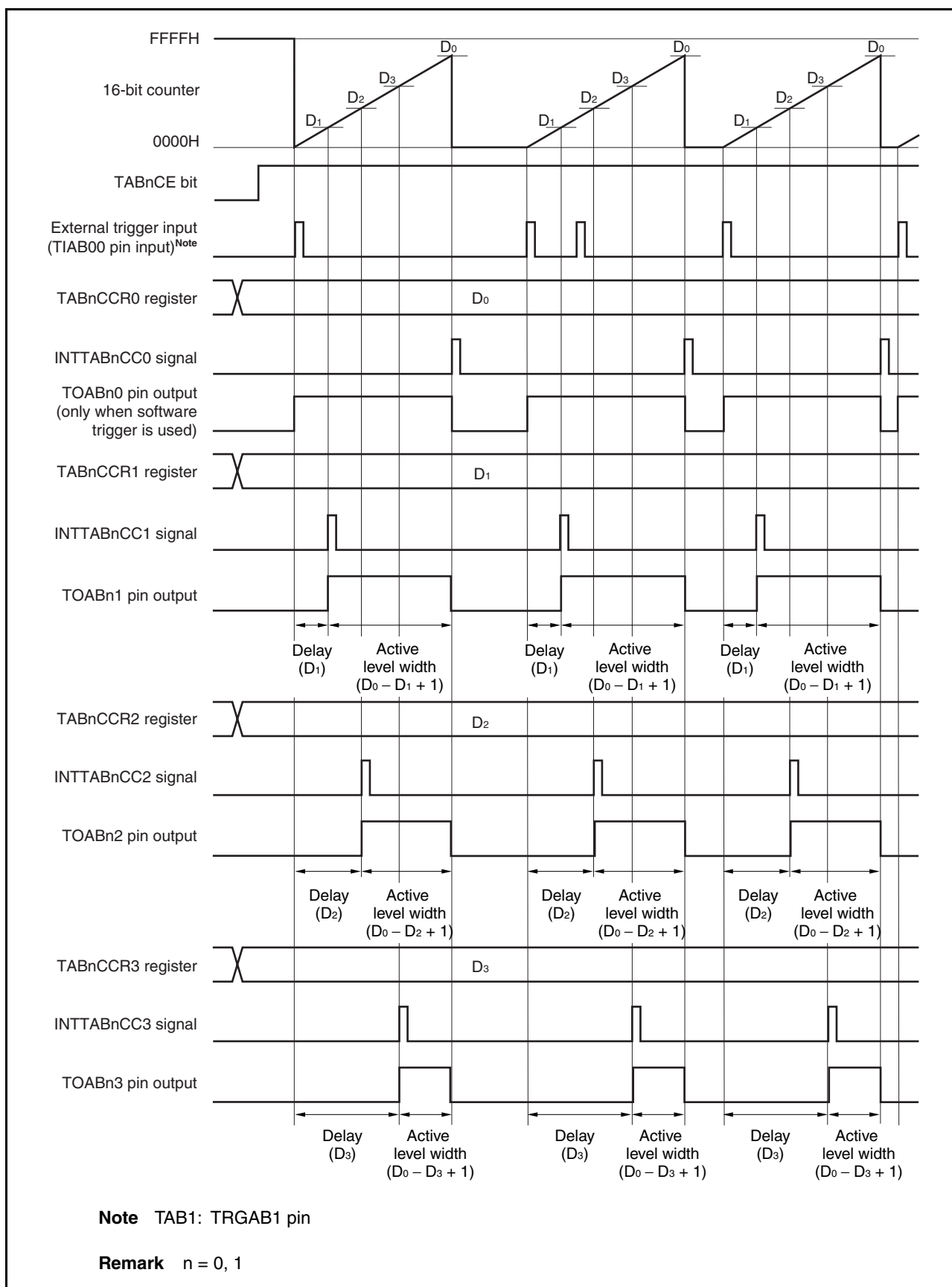
When the TABnCE bit is set to 1, TABn waits for a trigger.  When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a one-shot pulse from the TOABnk pin.  After the one-shot pulse is output, the 16-bit counter is set to FFFFH, stops counting, and waits for a trigger.  If a trigger is generated again while the one-shot pulse is being output, it is ignored.

The output delay period and active level width of the one-shot pulse can be calculated as follows.

Output delay period = (Set value of TABnCCRk register) × Count clock cycle
Active level width = (Set value of TABnCCR0 register − Set value of TABnCCRk register + 1) × Count clock cycle

The compare match interrupt request signal INTTABnCC0 is generated when the 16-bit counter counts up after its count value matches the value of the CCR0 buffer register.  The compare match interrupt request signal (INTTABnCCk) is generated when the count value of the 16-bit counter matches the value of the CCRk buffer register.

The valid edge of the external trigger input or setting the software trigger (TABnCTL1.TABnEST bit) to 1 is used as the trigger.

**Remark**  k = 1 to 3,
         n = 0, 1

**Figure 8-22.  Register Setting for Operation in One-Shot Pulse Output Mode (1/3)**

**Figure 8-22.  Register Setting for Operation in One-Shot Pulse Output Mode (2/3)**

**(c)  TABn I/O control register 0 (TABnIOC0)**

| | TABnOL3 | TABnOE3 | TABnOL2 | TABnOE2 | TABnOL1 | TABnOE1 | TABnOL0 | TABnOE0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC0 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1[Note] | 0/1[Note] |

0: Disable TOABn0 pin output
1: Enable TOABn0 pin output

Setting of output level while
operation of TOABn0 pin is disabled
0: Low level
1: High level

0: Disable TOABn1 pin output
1: Enable TOABn1 pin output

Specification of active level
of TOABn1 pin output
0: Active-high
1: Active-low

0: Disable TOABn2 pin output
1: Enable TOABn2 pin output

Specification of active level
of TOABn2 pin output
0: Active-high
1: Active-low

0: Disable TOABn3 pin output
1: Enable TOABn3 pin output

Specification of active level
of TOABn3 pin output
0: Active-high
1: Active-low

• When TABnOLk bit = 0

16-bit counter

TOABnk pin output

• When TABnOLk bit = 1

16-bit counter

TOABnk pin output

**(d)  TABn I/O control register 2 (TABnIOC2)**

| | | | | | TABnEES1 | TABnEES0 | TABnETS1 | TABnETS0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC2 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |

Select valid edge of
external trigger input

Select valid edge of
external event count input

**(e)  TABn counter read buffer register (TABnCNT)**

The value of the 16-bit counter can be read by reading the TABnCNT register.

**Note**　Clear this bit to 0 when the TOABn0 pin is not used in the one-shot pulse output mode.

**Remark**　n = 0, 1

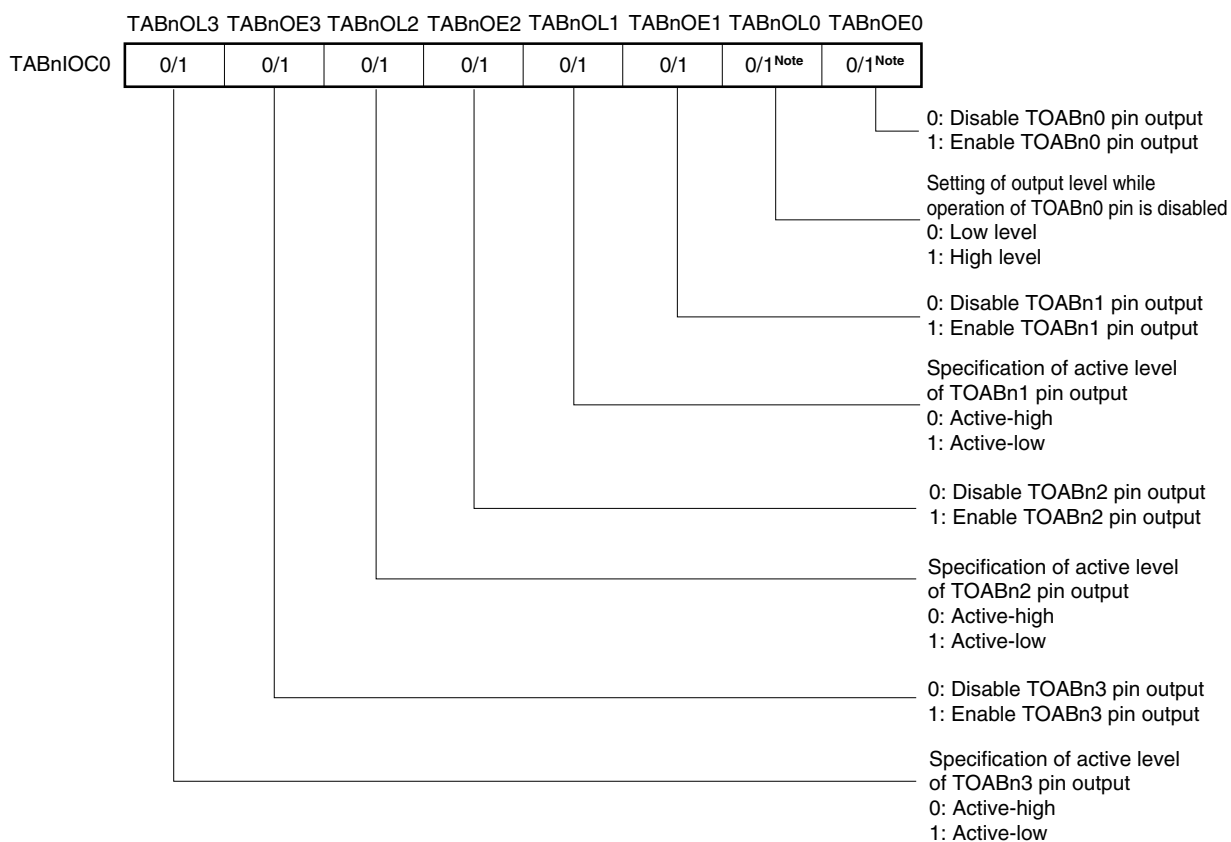**Figure 8-22.  Register Setting for Operation in One-Shot Pulse Output Mode (3/3)**

---

**(f)  TABn capture/compare registers 0 to 3 (TABnCCR0 to TABnCCR3)**

If $D_0$ is set to the TABnCCR0 register and $D_k$ to the TABnCCRk register, the active level width and output delay period of the one-shot pulse are as follows.

Active level width = $(D_0 - D_k + 1) \times$ Count clock cycle

Output delay period = $(D_k) \times$ Count clock cycle

**Caution   One-shot pulses are not output even in the one-shot pulse output mode, if the set value of the TABnCCRk register is greater than that value of the TABnCCR0 register.**

**Remarks  1.** TABn I/O control register 1 (TABnIOC1) and TABn option register 0 (TABnOPT0) are not used in the one-shot pulse output mode.

**2.** $k$ = 1 to 3,
$n$ = 0, 1

---

**(1) Operation flow in one-shot pulse output mode**

**Figure 8-23. Software Processing Flow in One-Shot Pulse Output Mode (1/2)**
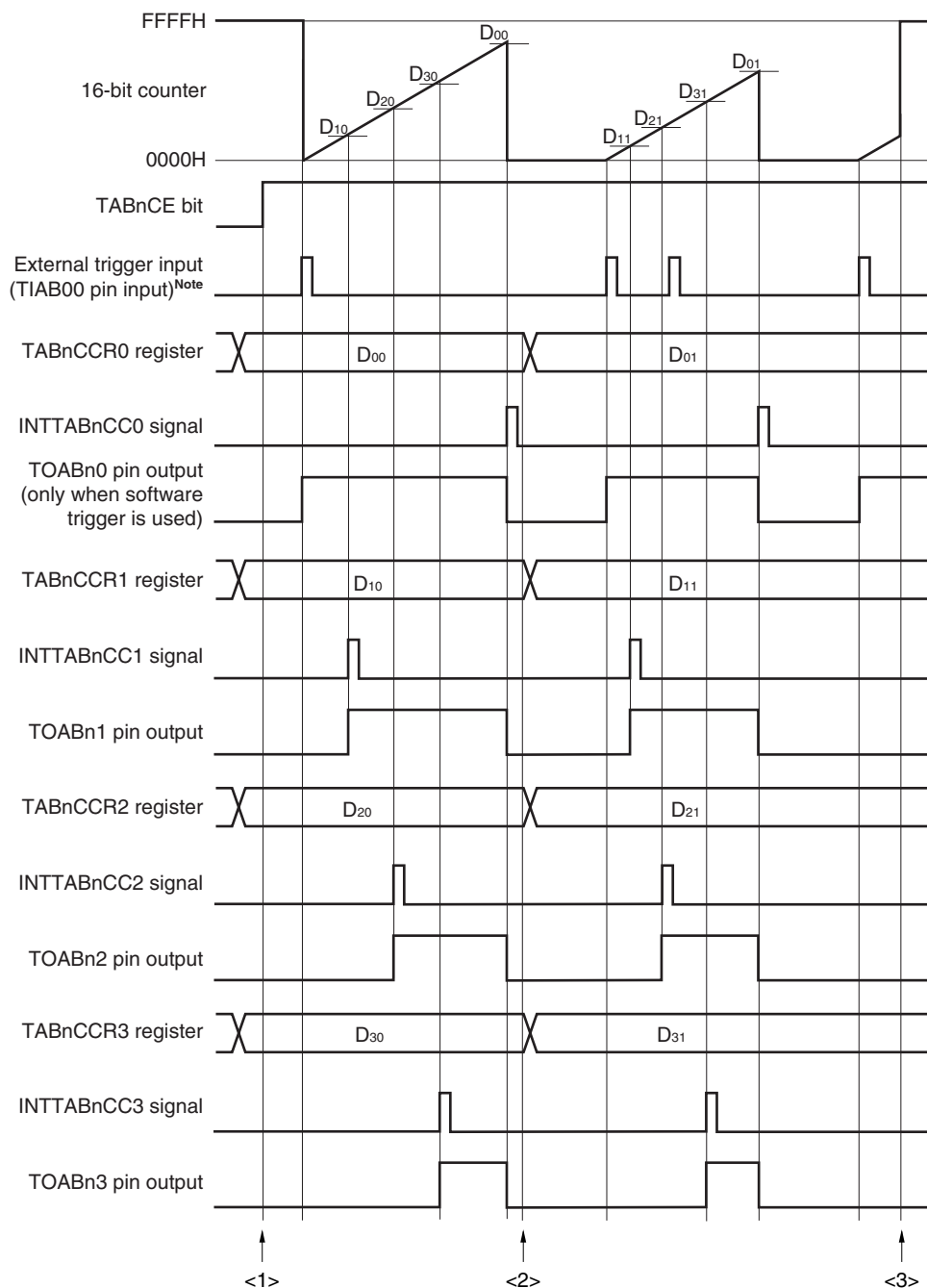


**Note**   TAB1: TRGAB1 pin

**Remark**   n = 0, 1

**Figure 8-23. Software Processing Flow in One-Shot Pulse Output Mode (2/2)**
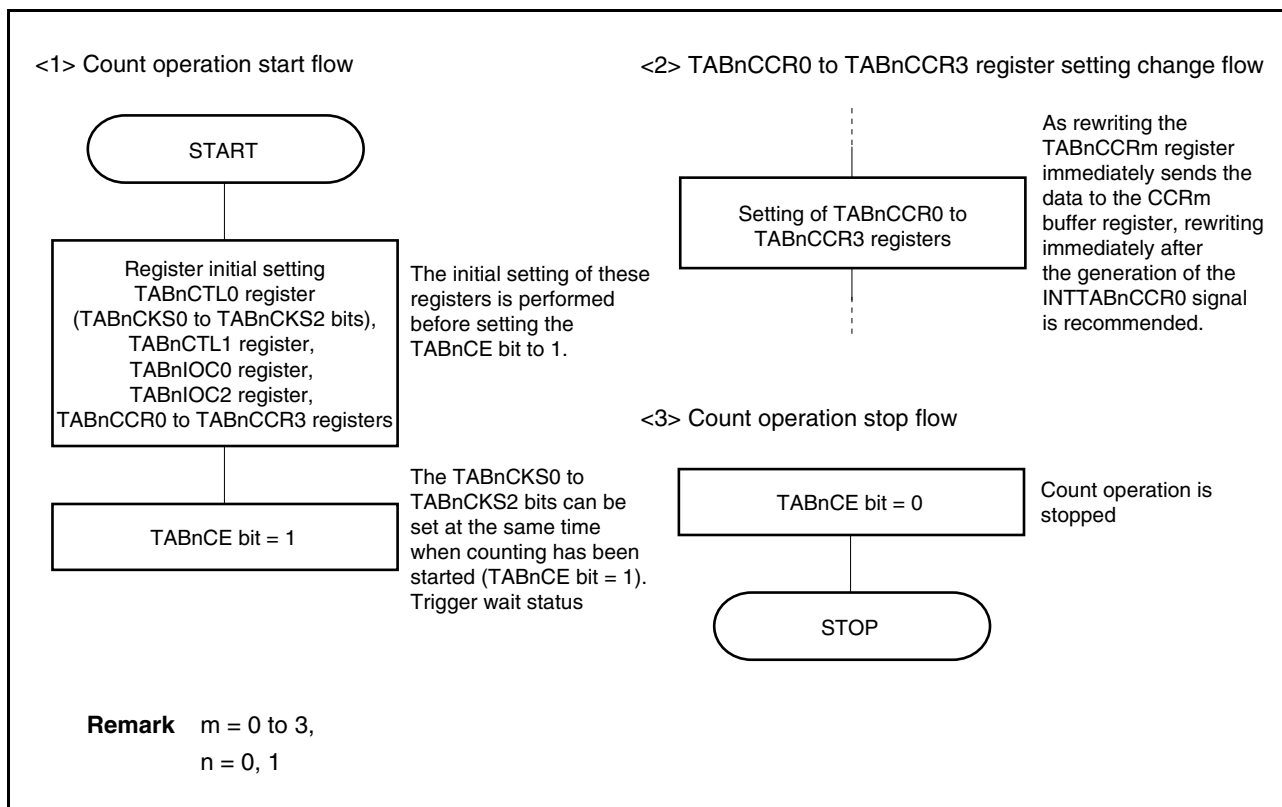
**(2)  Operation timing in one-shot pulse output mode**

**(a)  Notes on rewriting TABnCCRm register**

To change the set value of the TABnCCRm register to a smaller value, stop counting once, and then change the set value.

If the value of the TABnCCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow.



**Note**   TAB1: TRGAB1 pin
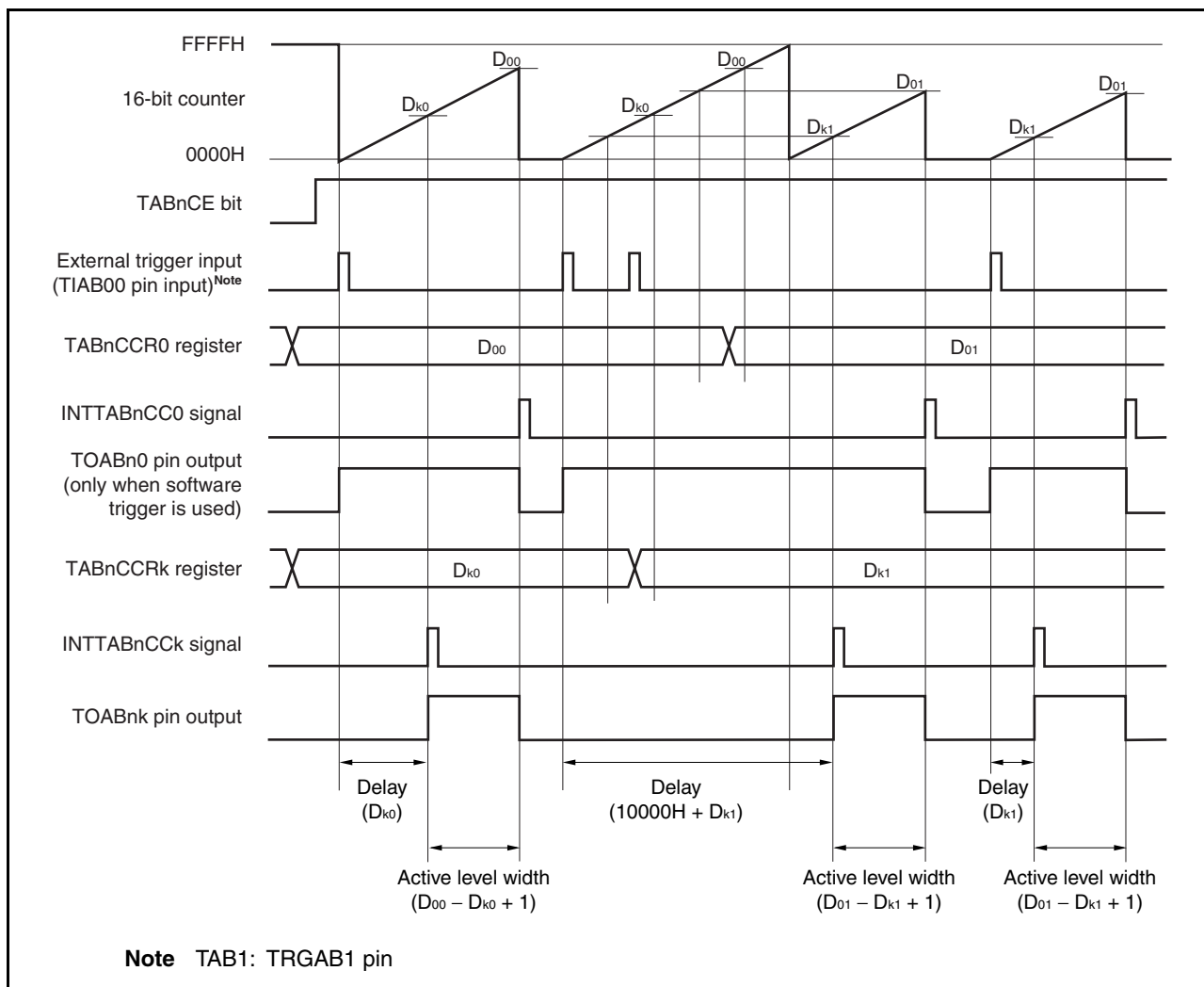
When the TABnCCR0 register is rewritten from $D_{00}$ to $D_{01}$ and the TABnCCRk register from $D_{k0}$ to $D_{k1}$ where $D_{00} > D_{01}$ and $D_{k0} > D_{k1}$, if the TABnCCRk register is rewritten when the count value of the 16-bit counter is greater than $D_{k1}$ and less than $D_{k0}$ and if the TABnCCR0 register is rewritten when the count value is greater than $D_{01}$ and less than $D_{00}$, each set value is reflected as soon as the register has been rewritten and compared with the count value.  The counter counts up to FFFFH and then counts up again from 0000H. When the count value matches $D_{k1}$, the counter generates the INTTABnCCk signal and asserts the TOABnk pin.  When the count value matches $D_{01}$, the counter generates the INTTABnCC0 signal, deasserts the TOABnk pin, and stops counting.

Therefore, the counter may output a pulse with a delay period or active period different from that of the one-shot pulse that is originally expected.

**Remark**   k = 1 to 3,
n = 0, 1

**(b) Generation timing of compare match interrupt request signal (INTTABnCCk)**

The generation timing of the INTTABnCCk signal in the one-shot pulse output mode is different from other INTTABnCCk signals; the INTTABnCCk signal is generated when the count value of the 16-bit counter matches the value of the TABnCCRk register.
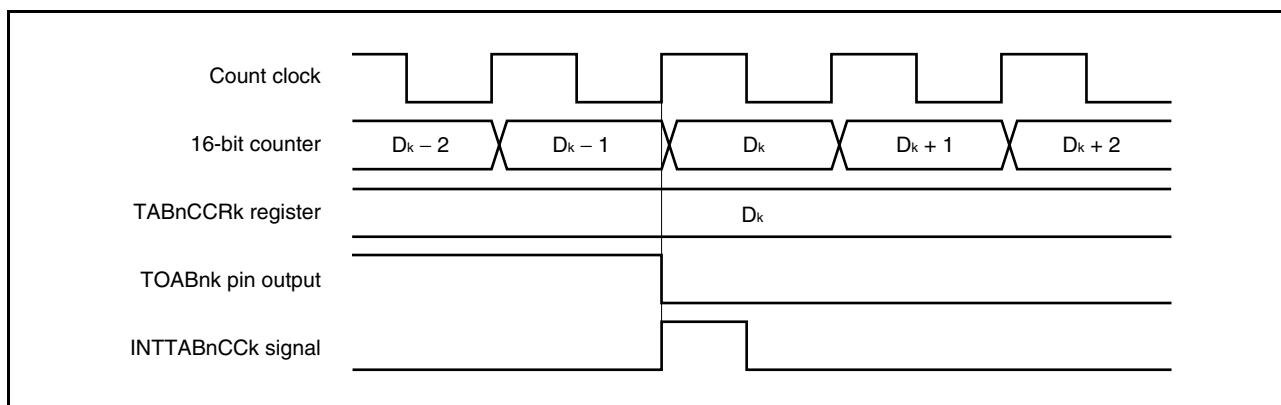


Usually, the INTTABnCCk signal is generated when the 16-bit counter counts up next time after its count value matches the value of the TABnCCRk register.

In the one-shot pulse output mode, however, it is generated one clock earlier.  This is because the timing is changed to match the change timing of the TOABnk pin.

**Remark**   k = 1 to 3,
            n = 0, 1

**8.5.5　PWM output mode (TABnMD2 to TABnMD0 bits = 100)**

In the PWM output mode, a PWM waveform is output from the TOABn1 to TOABn3 pins when the TABnCTL0.TABnCE bit is set to 1.

In addition, a pulse with one cycle of the PWM waveform as half its cycle is output from the TOABn0 pin.

**Figure 8-24.　Configuration in PWM Output Mode**
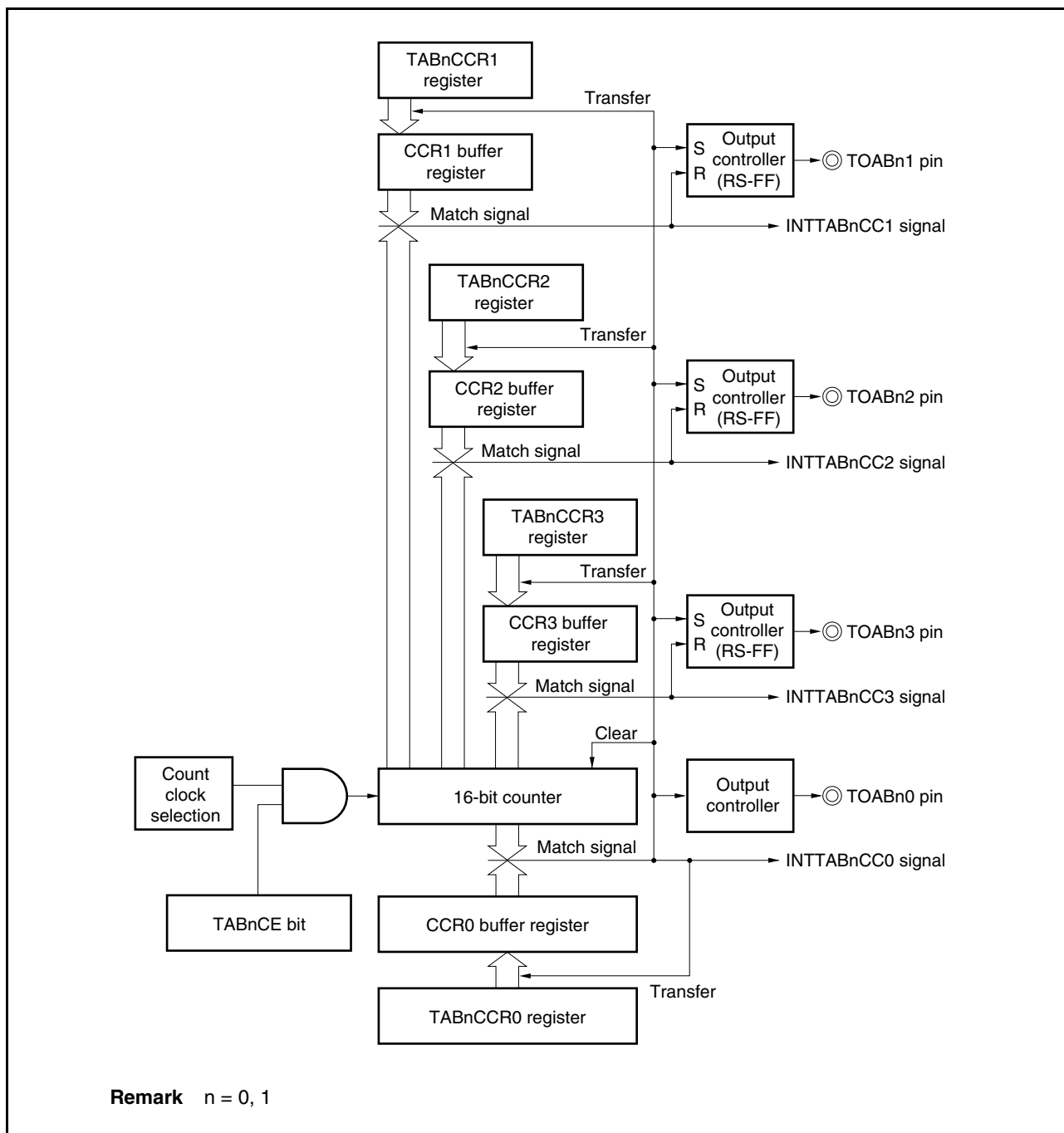
**Figure 8-25. Basic Timing in PWM Output Mode**
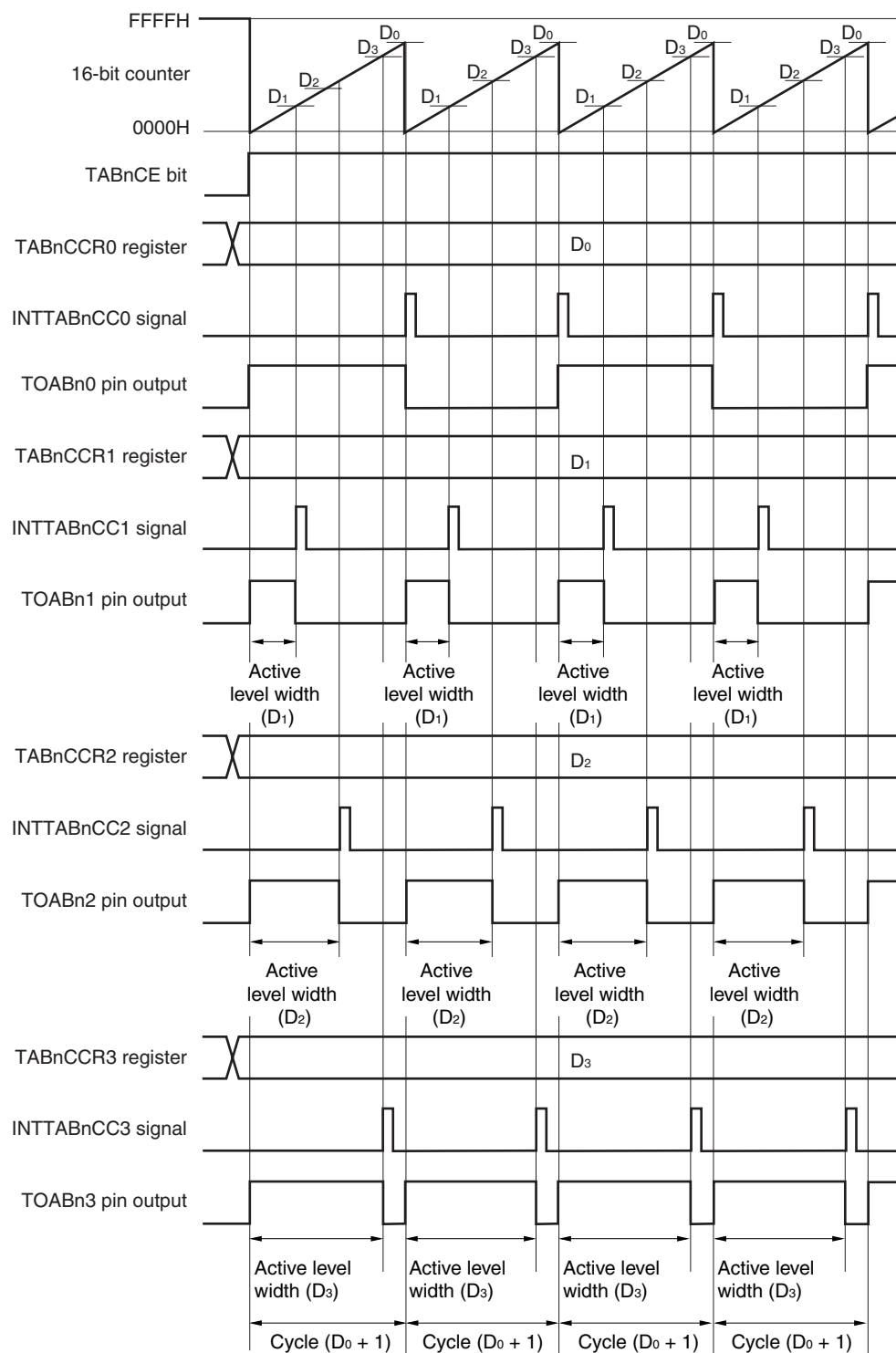


**Remark** n = 0, 1

When the TABnCE bit is set to 1, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a PWM waveform from the TOABnk pin.

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

Active level width = (Set value of TABnCCRk register) × Count clock cycle

Cycle = (Set value of TABnCCR0 register + 1) × Count clock cycle

Duty factor = (Set value of TABnCCRk register)/(Set value of TABnCCR0 register + 1)

The PWM waveform can be changed by rewriting the TABnCCRm register while the counter is operating. The newly written value is reflected when the count value of the 16-bit counter matches the value of the CCR0 buffer register and the 16-bit counter is cleared to 0000H.

The compare match interrupt request signal (INTTABnCC0) is generated when the 16-bit counter counts up next time after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H. The compare match interrupt request signal (INTTABnCCk) is generated when the count value of the 16-bit counter matches the value of the CCRk buffer register.

**Remark**    k = 1 to 3,
            m = 0 to 3,
            n = 0, 1

**Figure 8-26. Setting of Registers in PWM Output Mode (1/3)**



**(a) TABn control register 0 (TABnCTL0)**

**(b) TABn control register 1 (TABnCTL1)**

**Note** The setting is invalid when the TABnCTL1.TABnEEE bit = 1.

**Remark** n = 0, 1

**Figure 8-26.  Setting of Registers in PWM Output Mode (2/3)**



**(c)  TABn I/O control register 0 (TABnIOC0)**

| | TABnOL3 | TABnOE3 | TABnOL2 | TABnOE2 | TABnOL1 | TABnOE1 | TABnOL0 | TABnOE0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC0 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 Note | 0/1 Note |

0: Disable TOABn0 pin output
1: Enable TOABn0 pin output

Setting of output level while operation of TOABn0 pin is disabled
0: Low level
1: High level

0: Disable TOABn1 pin output
1: Enable TOABn1 pin output

Specification of active level of TOABn1 pin output
0: Active-high
1: Active-low

0: Disable TOABn2 pin output
1: Enable TOABn2 pin output

Specification of active level of TOABn2 pin output
0: Active-high
1: Active-low

0: Disable TOABn3 pin output
1: Enable TOABn3 pin output

Specification of active level of TOABn3 pin output
0: Active-high
1: Active-low

- When TABnOLk bit = 0

16-bit counter

TOABnk pin output

- When TABnOLk bit = 1

16-bit counter

TOABnk pin output

**(d)  TABn I/O control register 2 (TABnIOC2)**

| | | | | | TABnEES1 | TABnEES0 | TABnETS1 | TABnETS0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC2 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0 | 0 |

Select valid edge of external event count input.

**(e)  TABn counter read buffer register (TABnCNT)**
   The value of the 16-bit counter can be read by reading the TABnCNT register.

   **Note**   Clear this bit to 0 when the TOABn0 pin is not used in the PWM output mode.

**Remark**   n = 0, 1

**Figure 8-26.  Register Setting in PWM Output Mode (3/3)**

---

**(f) TABn capture/compare registers 0 to 3 (TABnCCR0 to TABnCCR3)**

If $D_0$ is set to the TABnCCR0 register and $D_k$ to the TABnCCRk register, the cycle and active level of the PWM waveform are as follows.

Cycle = $(D_0 + 1) \times$ Count clock cycle

Active level width = $D_k \times$ Count clock cycle

**Remarks  1.** TABn I/O control register 1 (TABnIOC1) and TABn option register 0 (TABnOPT0) are not used in the PWM output mode.

**2.** Updating TABn capture/compare register 2 (TABnCCR2) and TABn capture/compare register 3 (TABnCCR3) is enabled by writing TABn capture/compare register 1 (TABnCCR1).

**3.** n = 0, 1

---

**(1)  Operation flow in PWM output mode**

**Figure 8-27.  Software Processing Flow in PWM Output Mode (1/2)**



**Remark**    n = 0, 1

**Figure 8-27.  Software Processing Flow in PWM Output Mode (2/2)**

<1> Count operation start flow

START

Register initial setting
TABnCTL0 register
(TABnCKS0 to TABnCKS2
bits),
TABnCTL1 register,
TABnIOC0 register,
TABnIOC2 register,
TABnCCR0 to TABnCCR3
registers

The initial setting of these registers is performed before setting the TABnCE bit to 1.

TABnCE bit = 1

The TABnCKS0 to TABnCKS2 bits can be set at the same time when counting is enabled (TABnCE bit = 1).

<2> TABnCCR0 to TABnCCR3 register
setting change flow

Setting of TABnCCR0, TABnCCR2,
and TABnCCR3 registers

Setting of TABnCCR1 register

Writing the TABnCCR1 register must be performed after writing the TABnCCR0, TABnCCR2, and TABnCCR3 registers.
When the counter is cleared after setting, the value of the TABnCCRm register is transferred to the CCRm buffer registers.

<3> TABnCCR0 register setting change flow

Setting of TABnCCR0 register

Setting of TABnCCR1 register

Writing the same value to TABnCCR1 is necessary only when the set cycle is changed.
When the counter is cleared after setting, the value of the TABnCCRm register is transferred to the CCRm buffer register.

<4> TABnCCR1 to TABnCCR3 register
setting change flow

Setting of TABnCCR2,
TABnCCR3 registers

Setting of TABnCCR1 register

Writing the TABnCCR1 register must be performed only when the set duty factor is changed after writing the TABnCCR2 and TABnCCR3 registers.
When the counter is cleared after setting, the value of the TABnCCRm register is transferred to the CCRm buffer register.

<5> TABnCCR2, TABnCCR3 register
setting change flow

Setting of TABnCCR2,
TABnCCR3 registers

Setting of TABnCCR1 register

Writing the same value to the TABnCCR1 register is necessary only when the set duty factor of TOABn2 and TOABn3 pin outputs is changed.
When the counter is cleared after setting, the value of the TABnCCRm register is transferred to the CCRm buffer register.

<6> TABnCCR1 register setting change flow

Setting of TABnCCR1 register

The TABnCCR1 register only needs to be written, only when the set duty factor is changed.
When the counter is cleared after setting, the value of the TABnCCRm register is transferred to the CCRm buffer register.

<7> Count operation stop flow

TABnCE bit = 0

Counting is stopped.

STOP

**Remark** k = 1 to 3,
        m = 0 to 3,
        n = 0, 1

**(2) PWM output mode operation timing**

**(a) Changing pulse width during operation**

To change the PWM waveform while the counter is operating, write the TABnCCR1 register last.

Rewrite the TABnCCRk register after writing the TABnCCR1 register after the INTTABnCC1 signal is detected.
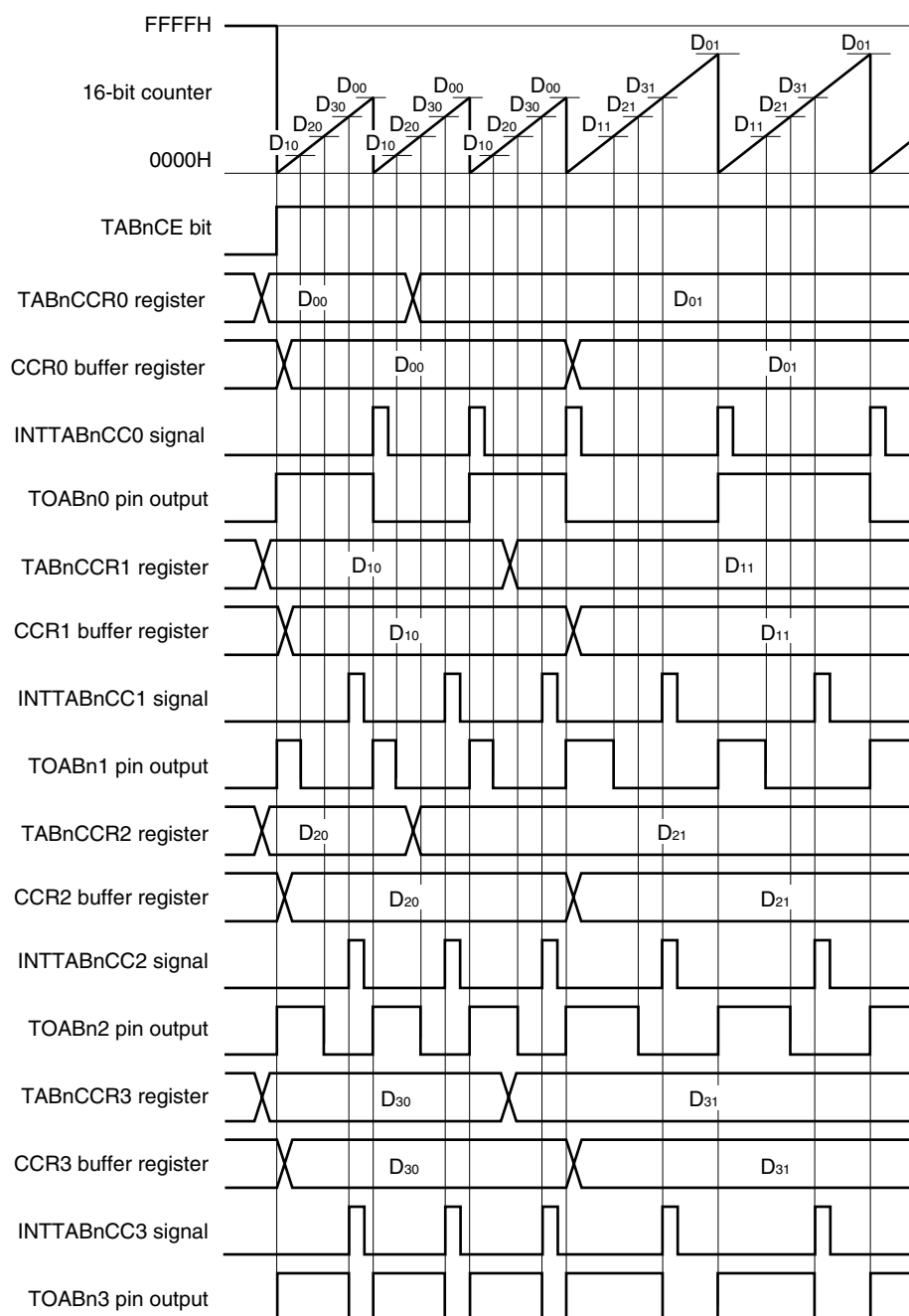


**Remark**   n = 0, 1

To transfer data from the TABnCCRm register to the CCRm buffer register, the TABnCCR1 register must be written.

To change both the cycle and active level of the PWM waveform at this time, first set the cycle to the TABnCCR0 register, set the active level width to the TABnCCR2 and TABnCCR3 registers, and then set the active level width to the TABnCCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TABnCCR0 register, and then write the same value to the TABnCCR1 register.

To change only the active level width (duty factor) of the PWM wave, first set the active level to the TABnCCR2 and TABnCCR3 registers, and then set the active level to the TABnCCR1 register.

To change only the active level width (duty factor) of the PWM waveform output by the TOABn1 pin, only the TABnCCR1 register has to be set.

To change only the active level width (duty factor) of the PWM waveform output by the TOABn2 and TOABn3 pins, first set the active level width to the TABnCCR2 and TABnCCR3 registers, and then write the same value to the TABnCCR1 register.

After the TABnCCR1 register is written, the value written to the TABnCCRm register is transferred to the CCRm buffer register in synchronization with the timing of clearing the 16-bit counter, and is used as the value to be compared with the value of the 16-bit counter.

To write the TABnCCR0 to TABnCCR3 registers again after writing the TABnCCR1 register once, do so after the INTTABnCC0 signal is generated.  Otherwise, the value of the CCRm buffer register may become undefined because the timing of transferring data from the TABnCCRm register to the CCRm buffer register conflicts with writing the TABnCCRm register.

**Remark**    m = 0 to 3,
              n = 0, 1

**(b) 0%/100% output of PWM waveform**

To output a 0% waveform, set the TABnCCRk register to 0000H.  If the set value of the TABnCCR0 register is FFFFH, the INTTABnCCk signal is generated periodically.



**Remark**   k = 1 to 3,
             n = 0, 1

To output a 100% waveform, set a value of "set value of TABnCCR0 register + 1" to the TABnCCRk register.  If the set value of the TABnCCR0 register is FFFFH, 100% output cannot be produced.



**Remark**   k = 1 to 3,
             n = 0, 1

**(c) Generation timing of compare match interrupt request signal (INTTABnCCk)**

The timing of generation of the INTTABnCCk signal in the PWM output mode differs from the timing of other INTTABnCCk signals; the INTTABnCCk signal is generated when the count value of the 16-bit counter matches the value of the TABnCCRk register.



**Remark**    k = 1 to 3,
           n = 0, 1

Usually, the INTTABnCCk signal is generated in synchronization with the next counting up after the count value of the 16-bit counter matches the value of the TABnCCRk register.

In the PWM output mode, however, it is generated one clock earlier. This is because the timing is changed to match the change timing of the output signal of the TOABnk pin.

### 8.5.6   Free-running timer mode (TABnMD2 to TABnMD0 bits = 101)

In the free-running timer mode, TABn starts counting when the TABnCTL0.TABnCE bit is set to 1.  At this time, the TABnCCRm register can be used as a compare register or a capture register, according to the setting of the TABnOPT0.TABnCCS0 and TABnOPT0.TABnCCS1 bits.

**Remark**   m = 0 to 3,
n = 0, 1

**Figure 8-28.  Configuration in Free-Running Timer Mode**



Note   TAB1:  EVTAB1 pin
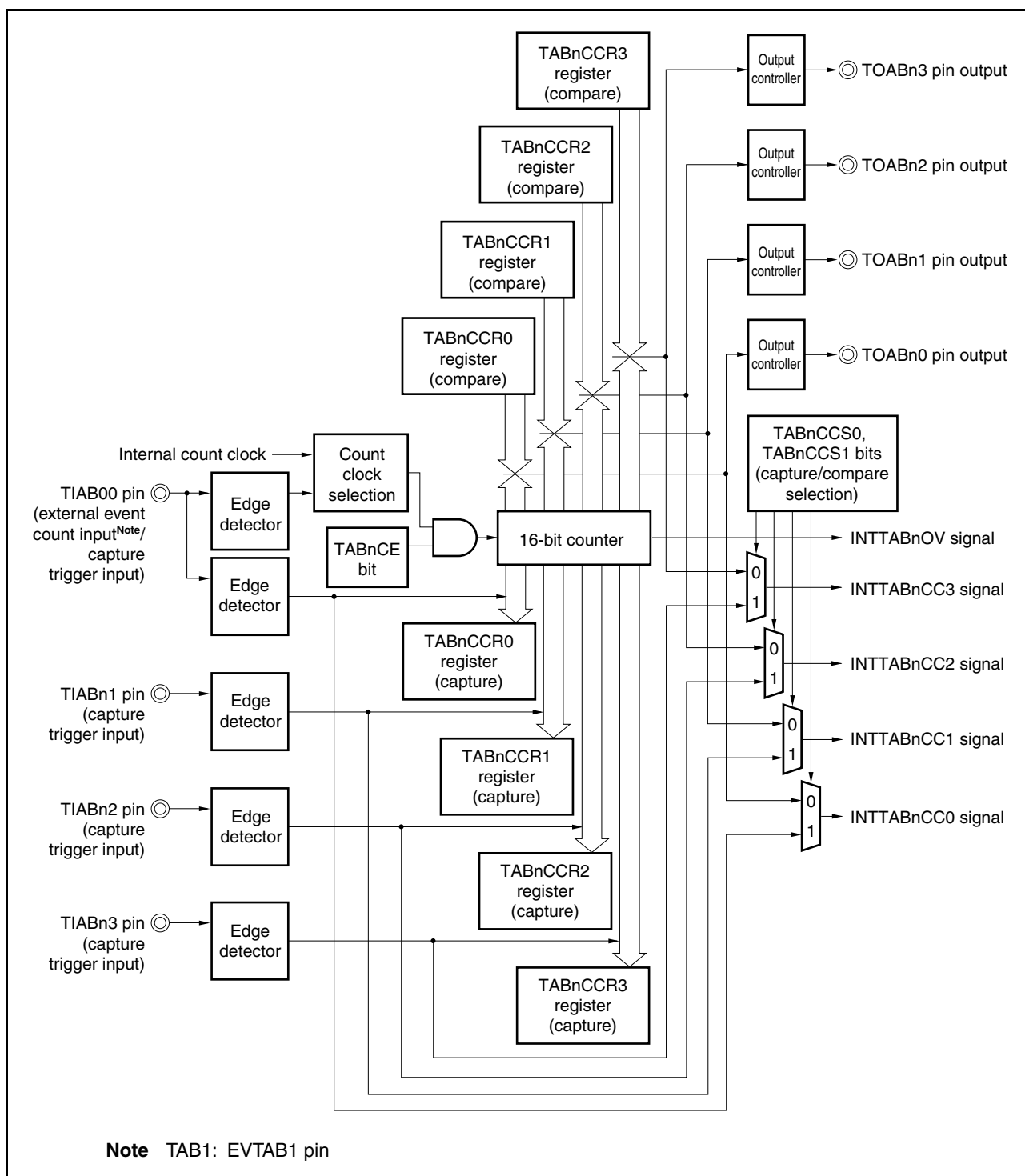
When the TABnCE bit is set to 1, TABn starts counting, and the output signals of the TOABn0 to TOABn3 pins are inverted.  When the count value of the 16-bit counter subsequently matches the set value of the TABnCCRm register, a compare match interrupt request signal (INTTABnCCm) is generated, and the output signal of the TOABnm pin is inverted.

The 16-bit counter continues counting in synchronization with the count clock.  When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTABnOV) at the next clock, is cleared to 0000H, and continues counting. At this time, the overflow flag (TABnOPT0.TABnOVF bit) is also set to 1.  Clear the overflow flag to 0 by executing the CLR instruction by software.

The TABnCCRm register can be rewritten while the counter is operating.  If it is rewritten, the new value is reflected at that time, and compared with the count value.

**Remark**  m = 0 to 3,
n = 0, 1

**Figure 8-29.  Basic Timing in Free-Running Timer Mode (Compare Function)**
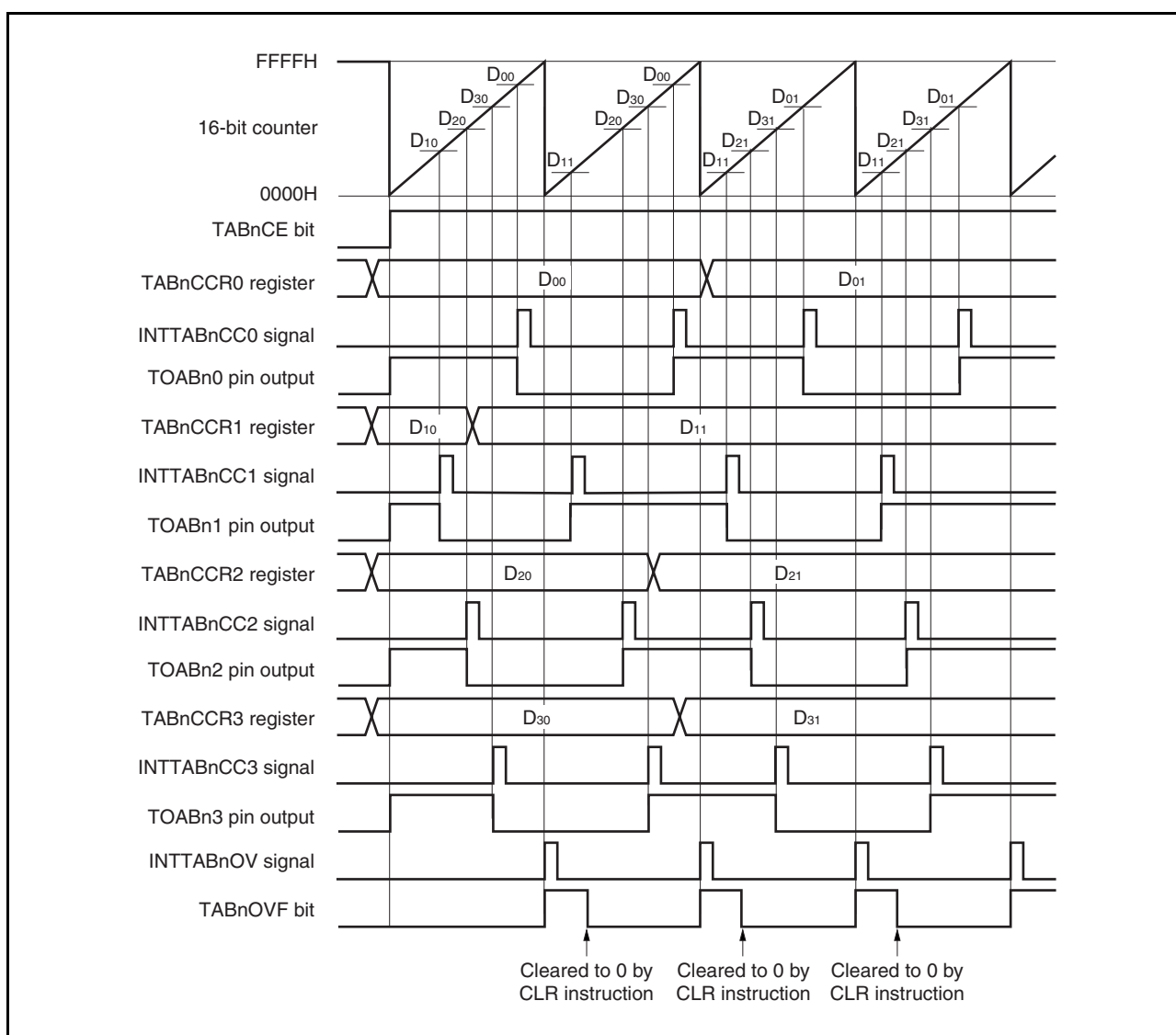
When the TABnCE bit is set to 1, the 16-bit counter starts counting.  When the valid edge input to the TIABnm pin is detected, the count value of the 16-bit counter is stored in the TABnCCRm register, and a capture interrupt request signal (INTTABnCCm) is generated.

The 16-bit counter continues counting in synchronization with the count clock.  When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTABnOV) at the next clock, is cleared to 0000H, and continues counting. At this time, the overflow flag (TABnOVF bit) is also set to 1. Clear the overflow flag to 0 by executing the CLR instruction by software.

**Remark**   m = 0 to 3,
             n = 0, 1

**Figure 8-30.  Basic Timing in Free-Running Timer Mode (Capture Function)**

**Figure 8-31.  Register Setting in Free-Running Timer Mode (1/3)**



**(a)  TABn control register 0 (TABnCTL0)**

TABnCE / TABnCKS2 TABnCKS1 TABnCKS0

TABnCTL0 | 0/1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1

Select count clock**Note**
0: Stop counting
1: Enable counting

**Note**   The setting is invalid when the TABnCTL1.TABnEEE bit = 1

**(b)  TABn control register 1 (TABnCTL1)**

TABnSYE TABnEST TABnEEE / TABnMD2 TABnMD1 TABnMD0

TABnCTL1 | 0 | 0 | 0/1 | 0 | 0 | 1 | 0 | 1

1, 0, 1:
Free-running mode

0: Operate with count
   clock selected by
   TABnCKS0 to TABnCKS2 bits
1: Count by external
   event count input signal

**(c)  TABn I/O control register 0 (TABnIOC0)**

TABnOL3 TABnOE3 TABnOL2 TABnOE2 TABnOL1 TABnOE1 TABnOL0 TABnOE0

TABnIOC0 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1

0: Disable TOABn0 pin output
1: Enable TOABn0 pin output

Setting of output level with
operation of TOABn0 pin disabled
0: Low level
1: High level

0: Disable TOABn1 pin output
1: Enable TOABn1 pin output

Setting of output level with
operation of TOABn1 pin
disabled
0: Low level
1: High level

0: Disable TOABn2 pin output
1: Enable TOABn2 pin output

Setting of output level with
operation of TOABn2 pin
disabled
0: Low level
1: High level

0: Disable TOABn3 pin output
1: Enable TOABn3 pin output

Setting of output level with
operation of TOABn3 pin
disabled
0: Low level
1: High level

**Remark**   n = 0, 1

**Figure 8-31.  Register Setting in Free-Running Timer Mode (2/3)**

**(d) TABn I/O control register 1 (TABnIOC1)**

| | TABnIS7 | TABnIS6 | TABnIS5 | TABnIS4 | TABnIS3 | TABnIS2 | TABnIS1 | TABnIS0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |

Select valid edge
of TIABn0 pin input

Select valid edge
of TIABn1 pin input

Select valid edge
of TIABn2 pin input

Select valid edge
of TIABn3 pin input

**(e) TABn I/O control register 2 (TABnIOC2)**

| | | | | | TABnEES1 | TABnEES0 | TABnETS1 | TABnETS0 |
|---|---|---|---|---|---|---|---|---|
| TABnIOC2 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0 | 0 |

Select valid edge of
external event count input

**(f) TABn option register 0 (TABnOPT0)**

| | TABnCCS3 | TABnCCS2 | TABnCCS1 | TABnCCS0 | | | | TABnOVF |
|---|---|---|---|---|---|---|---|---|
| TABnOPT0 | 0/1 | 0/1 | 0/1 | 0/1 | 0 | 0 | 0 | 0/1 |

Overflow flag

Specifies if TABnCCR0
register functions as
capture or compare register

Specifies if TABnCCR1
register functions as
capture or compare register

Specifies if TABnCCR2
register functions as
capture or compare register

Specifies if TABnCCR3
register functions as
capture or compare register

**(g) TABn counter read buffer register (TABnCNT)**

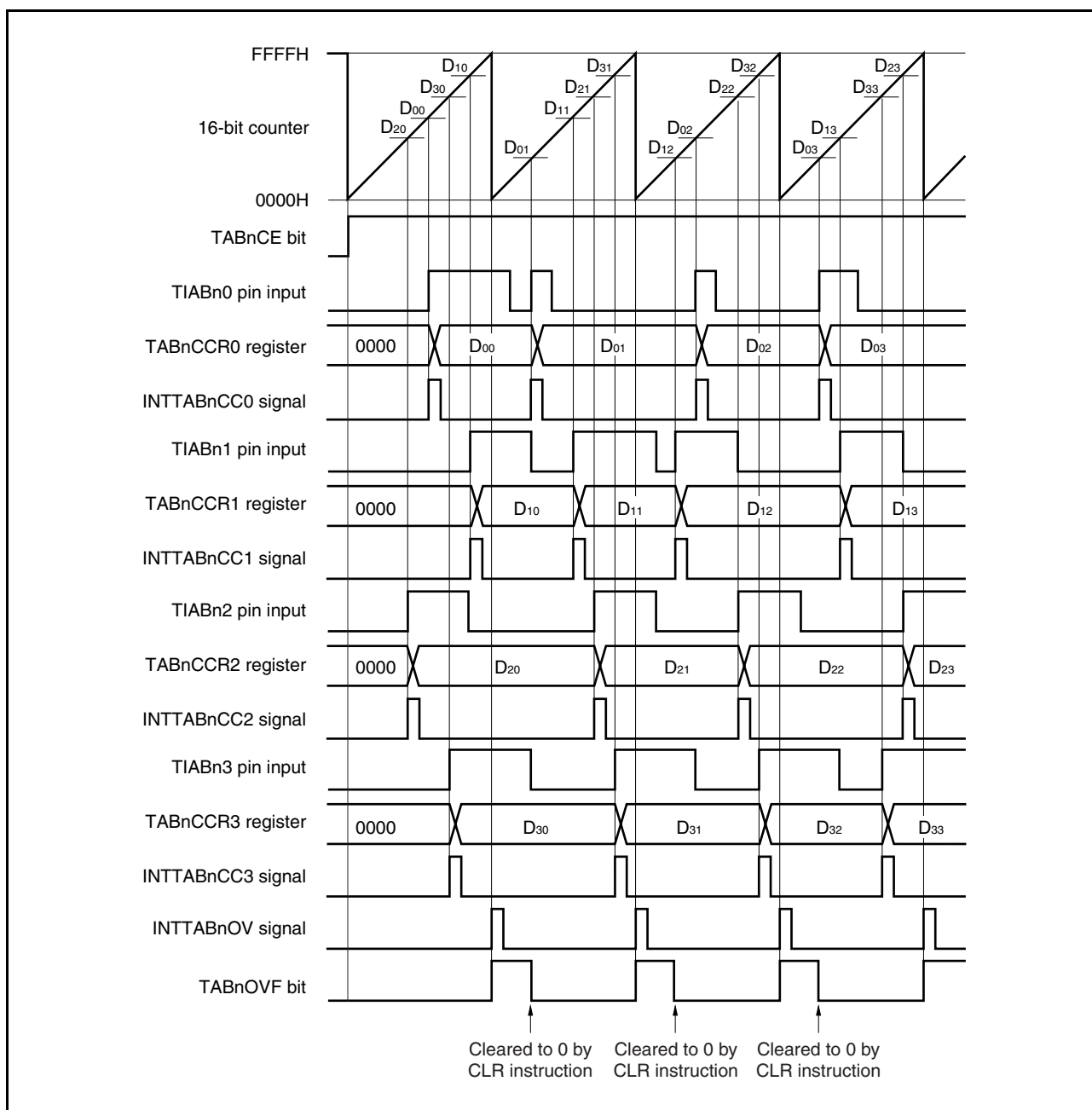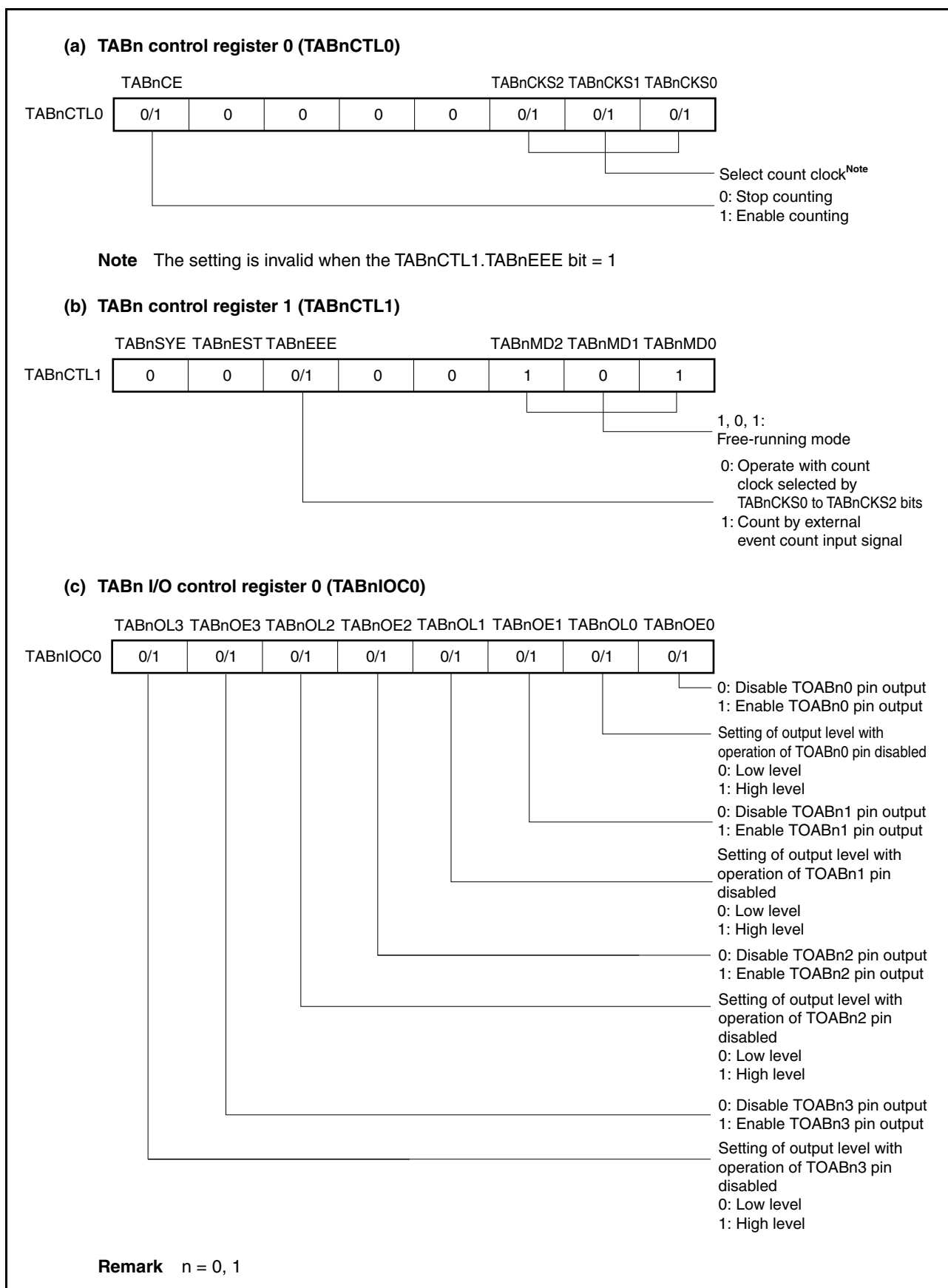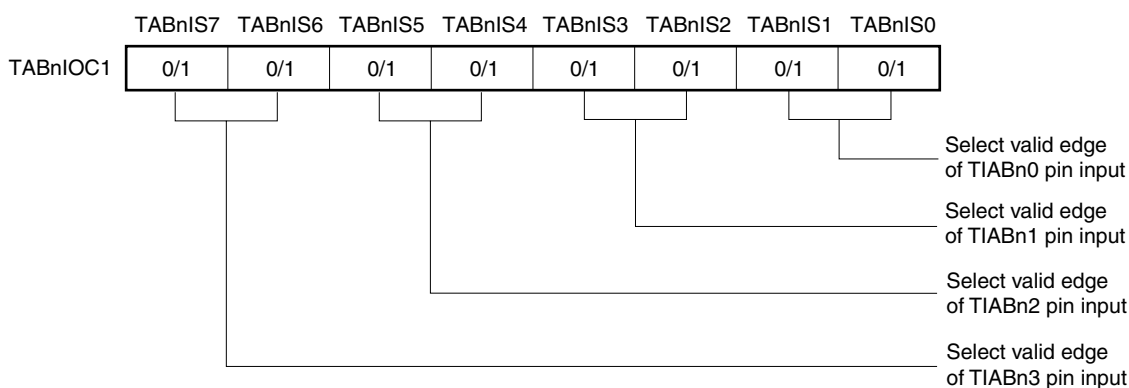The value of the 16-bit counter can be read by reading the TABnCNT register.

**Remark**   n = 0, 1

**Figure 8-31.  Register Setting in Free-Running Timer Mode (3/3)**

---

**(h) TABn capture/compare registers 0 to 3 (TABnCCR0 to TABnCCR3)**

These registers function as capture registers or compare registers according to the setting of the TABnOPT0.TABnCCSm bit.

When the registers function as capture registers, they store the count value of the 16-bit counter when the valid edge input to the TIABnm pin is detected.

When the registers function as compare registers and when $D_m$ is set to the TABnCCRm register, the INTTABnCCm signal is generated when the counter reaches ($D_m$ + 1), and the output signal of the TOABnm pin is inverted.

**Remark**   m = 0 to 3,
n = 0, 1

---

**(1) Operation flow in free-running timer mode**

**(a) When using capture/compare register as compare register**

**Figure 8-32. Software Processing Flow in Free-Running Timer Mode (Compare Function) (1/2)**
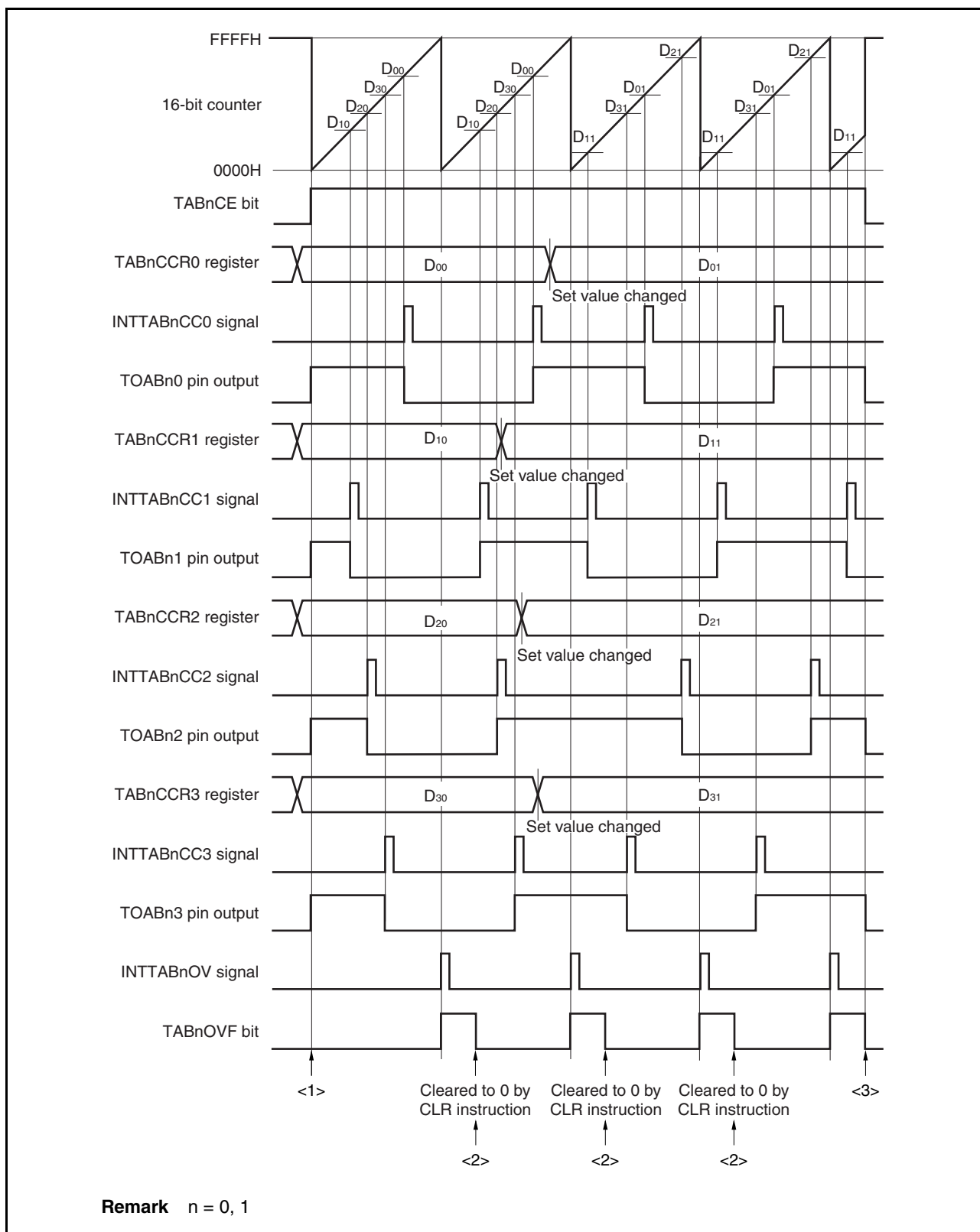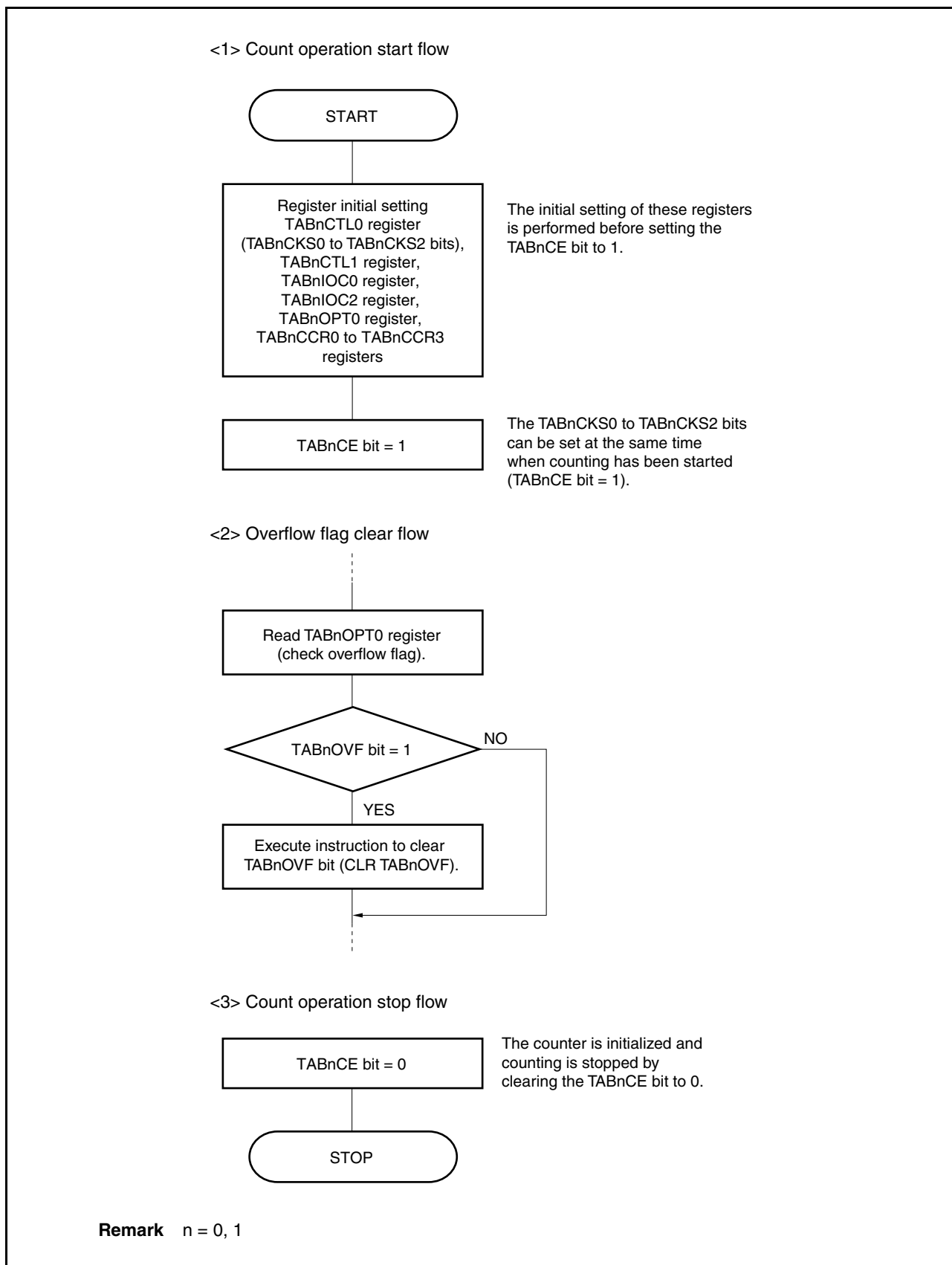


Remark   n = 0, 1

**Figure 8-32.  Software Processing Flow in Free-Running Timer Mode (Compare Function) (2/2)**

<1> Count operation start flow

```
                    ┌──────────────┐
                    │    START     │
                    └──────────────┘
                            │
        ┌───────────────────────────────────┐      The initial setting of these registers
        │     Register initial setting      │      is performed before setting the
        │        TABnCTL0 register          │      TABnCE bit to 1.
        │ (TABnCKS0 to TABnCKS2 bits),      │
        │        TABnCTL1 register,          │
        │        TABnIOC0 register,          │
        │        TABnIOC2 register,          │
        │        TABnOPT0 register,          │
        │     TABnCCR0 to TABnCCR3          │
        │           registers               │
        └───────────────────────────────────┘
                            │
        ┌───────────────────────────────────┐      The TABnCKS0 to TABnCKS2 bits
        │          TABnCE bit = 1           │      can be set at the same time
        └───────────────────────────────────┘      when counting has been started
                                                    (TABnCE bit = 1).
```

<2> Overflow flag clear flow

```
        ┌───────────────────────────────────┐
        │     Read TABnOPT0 register        │
        │      (check overflow flag).       │
        └───────────────────────────────────┘
                            │
                       ╱─────────╲                    NO
                      ╱ TABnOVF    ╲ ──────────────┐
                      ╲  bit = 1   ╱               │
                       ╲─────────╱                 │
                            │ YES                   │
        ┌───────────────────────────────────┐      │
        │    Execute instruction to clear   │      │
        │    TABnOVF bit (CLR TABnOVF).      │      │
        └───────────────────────────────────┘      │
                            │◄─────────────────────┘
```

<3> Count operation stop flow

```
        ┌───────────────────────────────────┐      The counter is initialized and
        │          TABnCE bit = 0           │      counting is stopped by
        └───────────────────────────────────┘      clearing the TABnCE bit to 0.
                            │
                    ┌──────────────┐
                    │    STOP      │
                    └──────────────┘
```

**Remark**   n = 0, 1

**(b) When using capture/compare register as capture register**

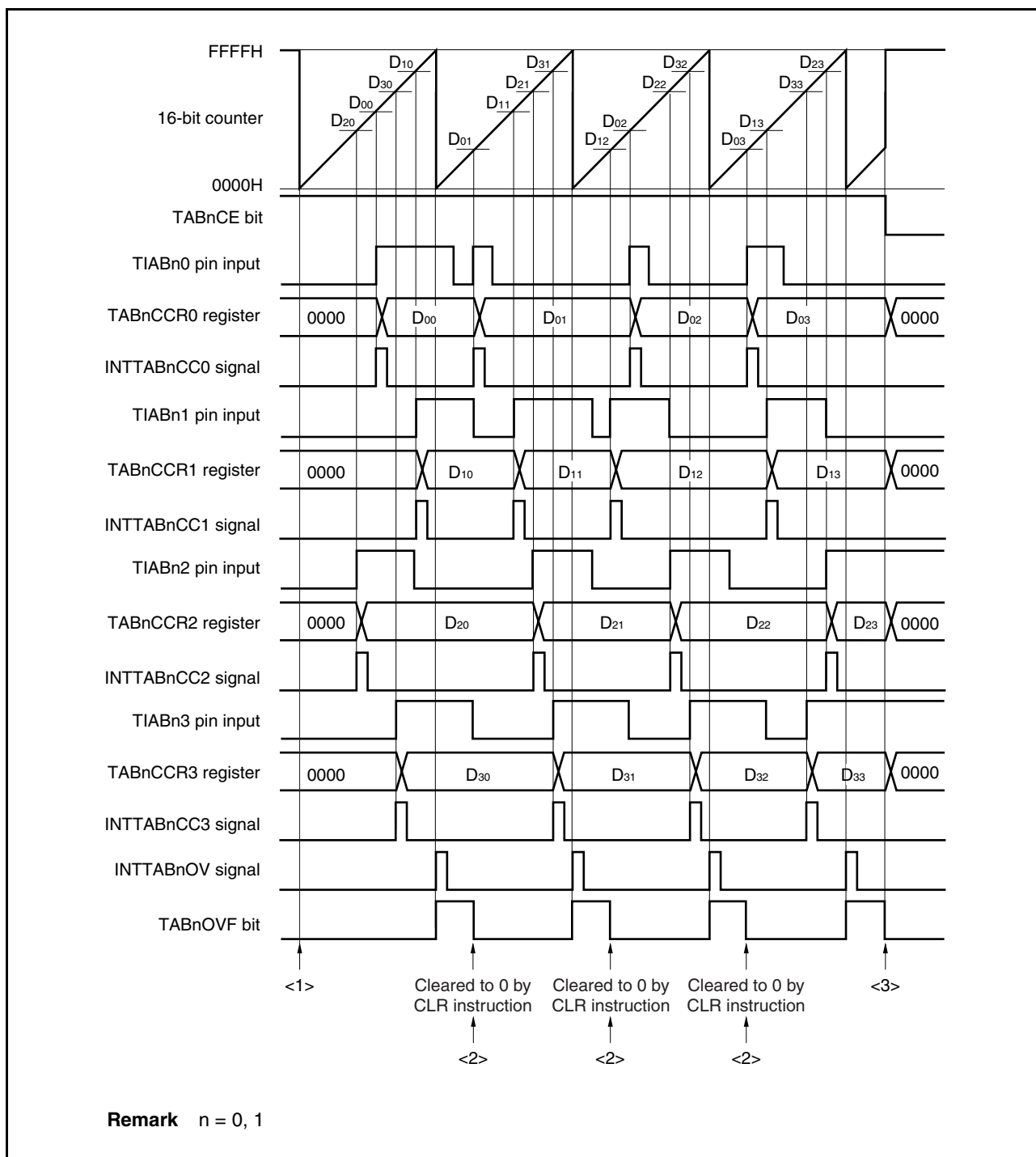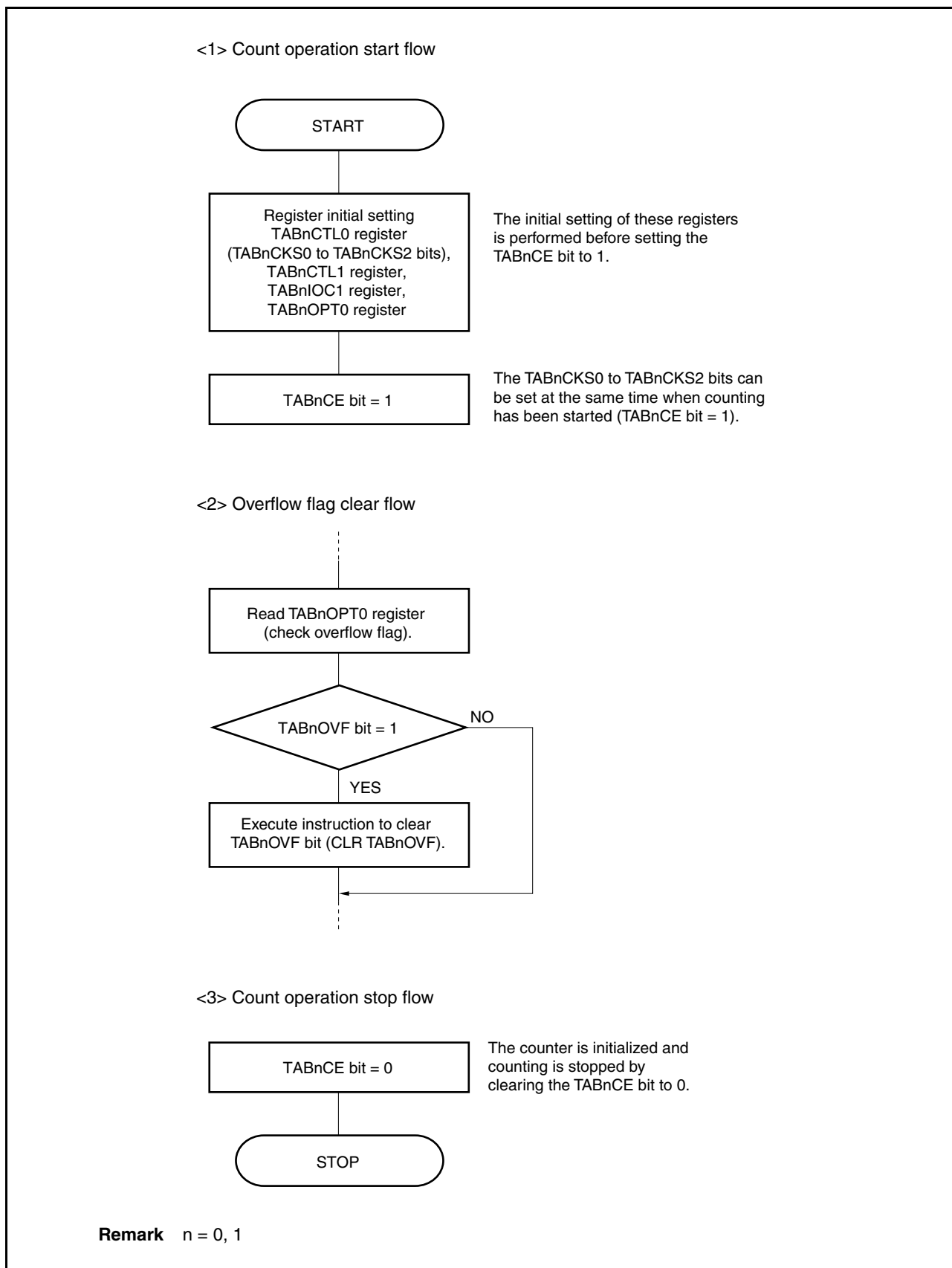**Figure 8-33.  Software Processing Flow in Free-Running Timer Mode (Capture Function) (1/2)**



**Remark**    n = 0, 1

**Figure 8-33. Software Processing Flow in Free-Running Timer Mode (Capture Function) (2/2)**

**(2) Operation timing in free-running timer mode**

**(a) Interval operation with compare register**

When TABn is used as an interval timer with the TABnCCRm register used as a compare register, software processing is necessary for setting a comparison value to generate the next interrupt request signal each time the INTTABnCCm signal has been detected.
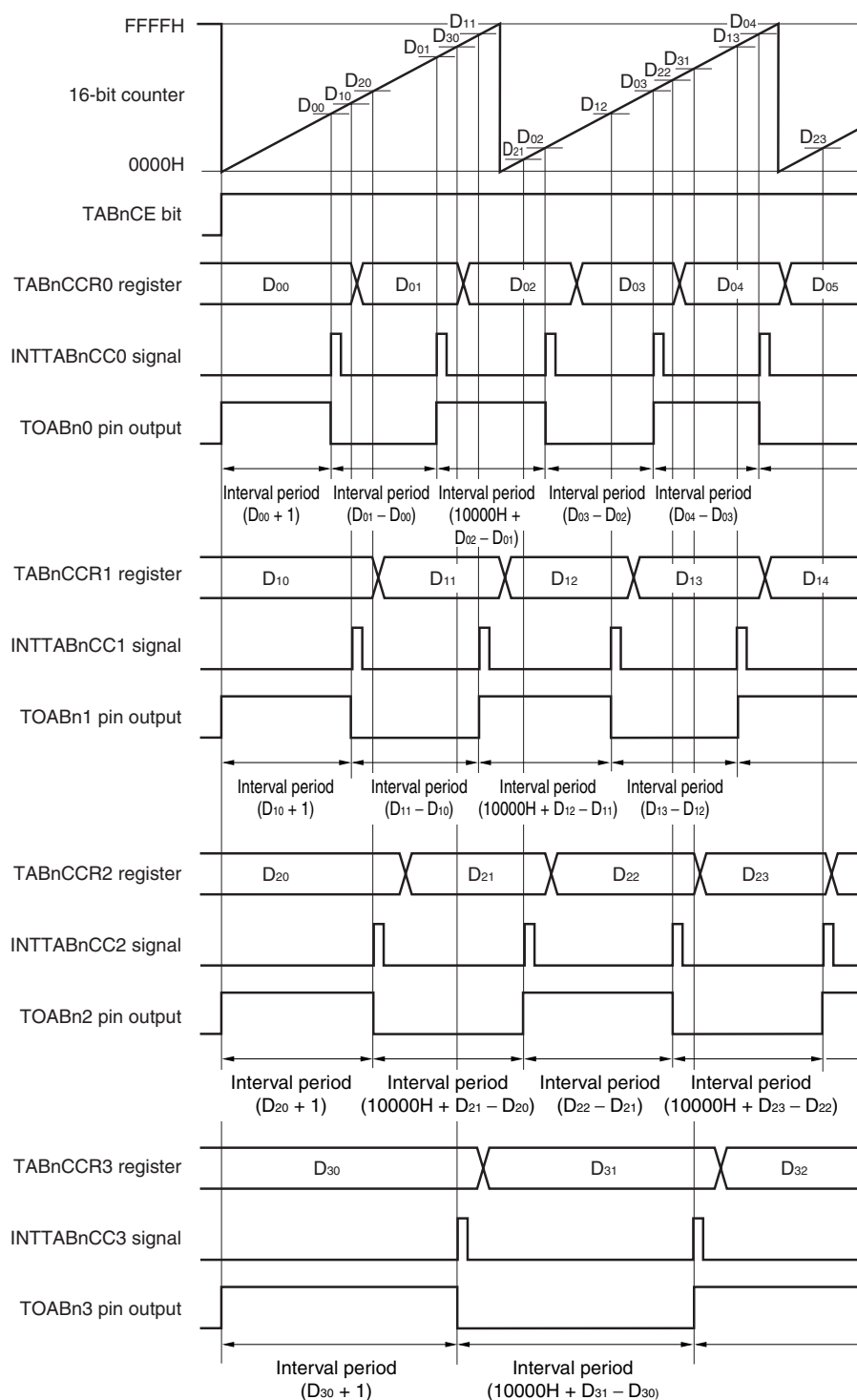


**Remark**  n = 0, 1

When performing an interval operation in the free-running timer mode, four intervals can be set with one channel.

To perform the interval operation, the value of the corresponding TABnCCRm register must be re-set in the interrupt servicing that is executed when the INTTABnCCm signal is detected.

The set value for re-setting the TABnCCRm register can be calculated by the following expression, where "$D_m$" is the interval period.

Compare register default value: $D_m - 1$

Value set to compare register second and subsequent times: Previous set value + $D_m$

(If the calculation result is greater than FFFFH, subtract 10000H from the result and set this value to the register.)

**Remark**    m = 0 to 3,
              n = 0, 1

**(b) Pulse width measurement with capture register**

When pulse width measurement is performed with the TABnCCRm register used as a capture register, software processing is necessary for reading the capture register each time the INTTABnCCm signal has been detected and for calculating an interval.



**Remark** n = 0, 1
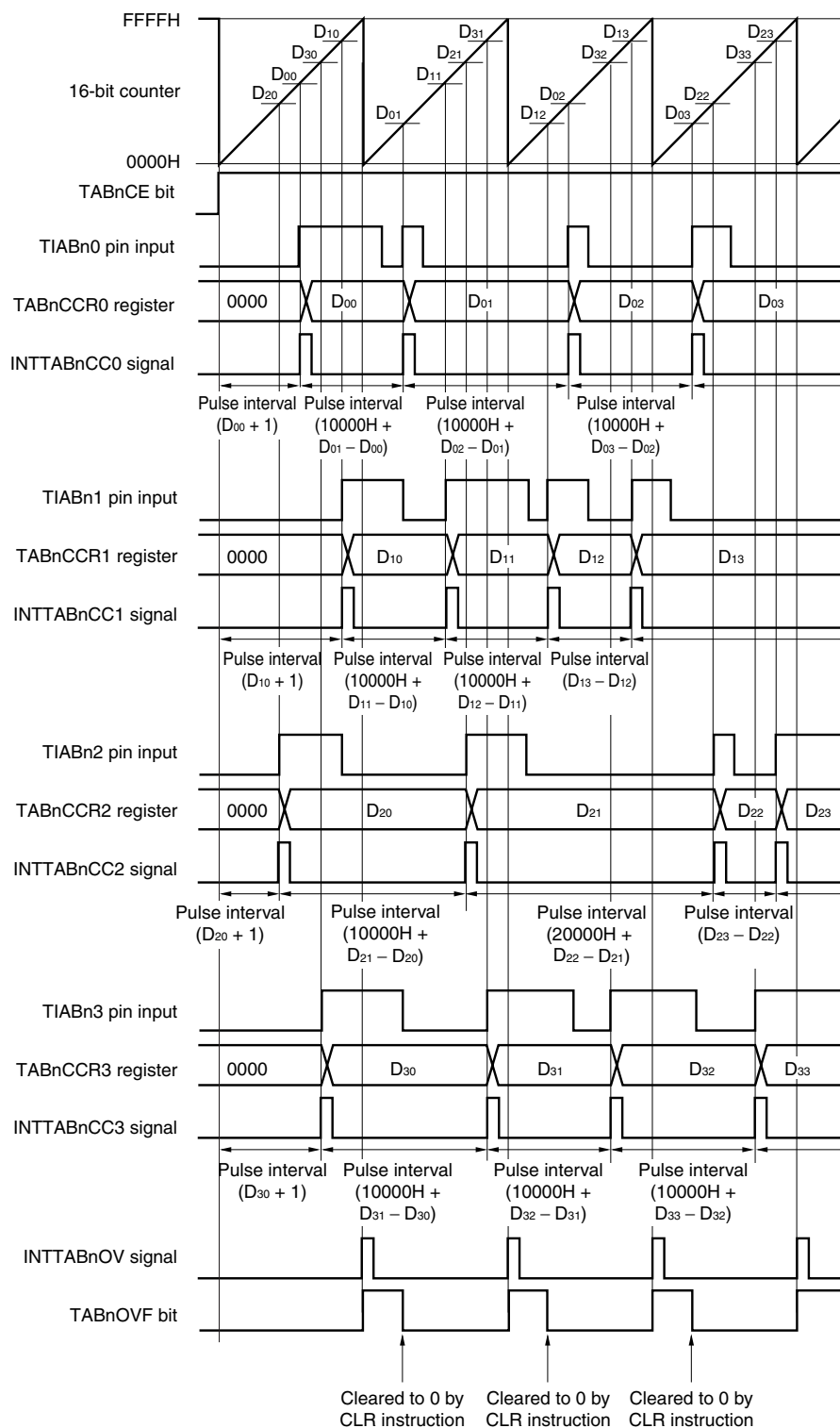
When executing pulse width measurement in the free-running timer mode, four pulse widths can be measured with one channel.

To measure a pulse width, the pulse width can be calculated by reading the value of the TABnCCRm register in synchronization with the INTTABnCCm signal, and calculating the difference between the value read this time and the previously read value.

**Remark**   m = 0 to 3,
              n = 0, 1

**(c) Processing of overflow when two or more capture registers are used**

Care must be exercised in processing the overflow flag when two or more capture registers are used. First, an example of incorrect processing is shown below.

---

**Example of incorrect processing when two or more capture registers are used**
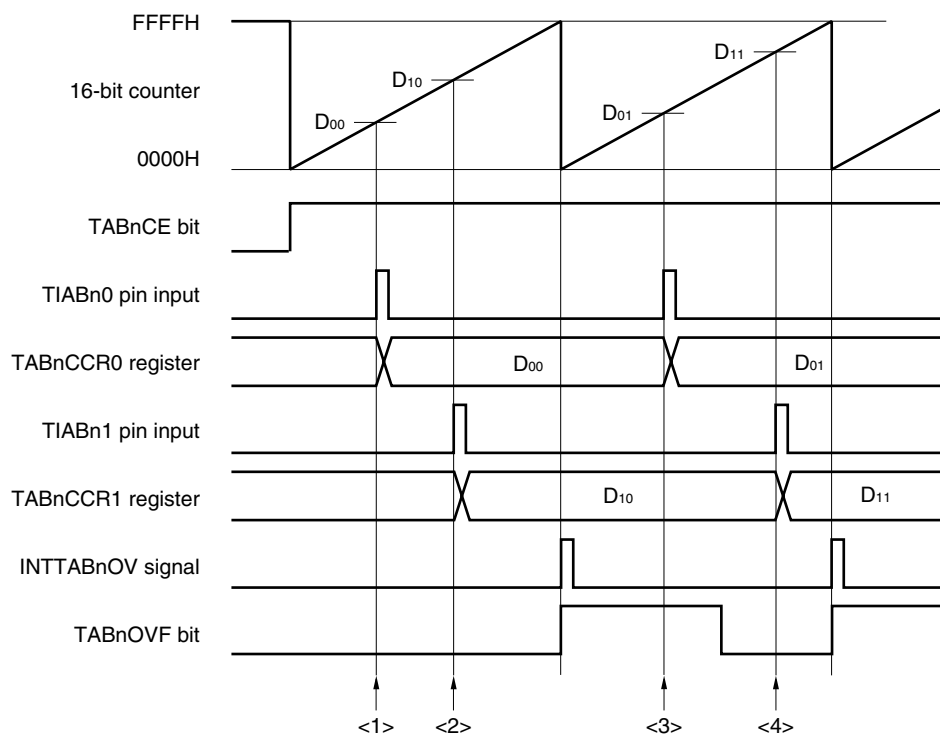


The following problem may occur when two pulse widths are measured in the free-running timer mode.

<1> Read the TABnCCR0 register (setting of the default value of the TIABn0 pin input).
<2> Read the TABnCCR1 register (setting of the default value of the TIABn1 pin input).
<3> Read the TABnCCR0 register.
    Read the overflow flag. If the overflow flag is 1, clear it to 0.
    Because the overflow flag is 1, the pulse width can be calculated by $(10000H + D_{01} - D_{00})$.
<4> Read the TABnCCR1 register.
    Read the overflow flag. Because the flag is cleared in <3>, 0 is read.
    Because the overflow flag is 0, the pulse width can be calculated by $(D_{11} - D_{10})$ (incorrect).

**Remark**   n = 0, 1

---

When two or more capture registers are used, and if the overflow flag is cleared to 0 by one capture register, the other capture register may not obtain the correct pulse width.

Use software when using two or more capture registers. An example of how to use software is shown below.
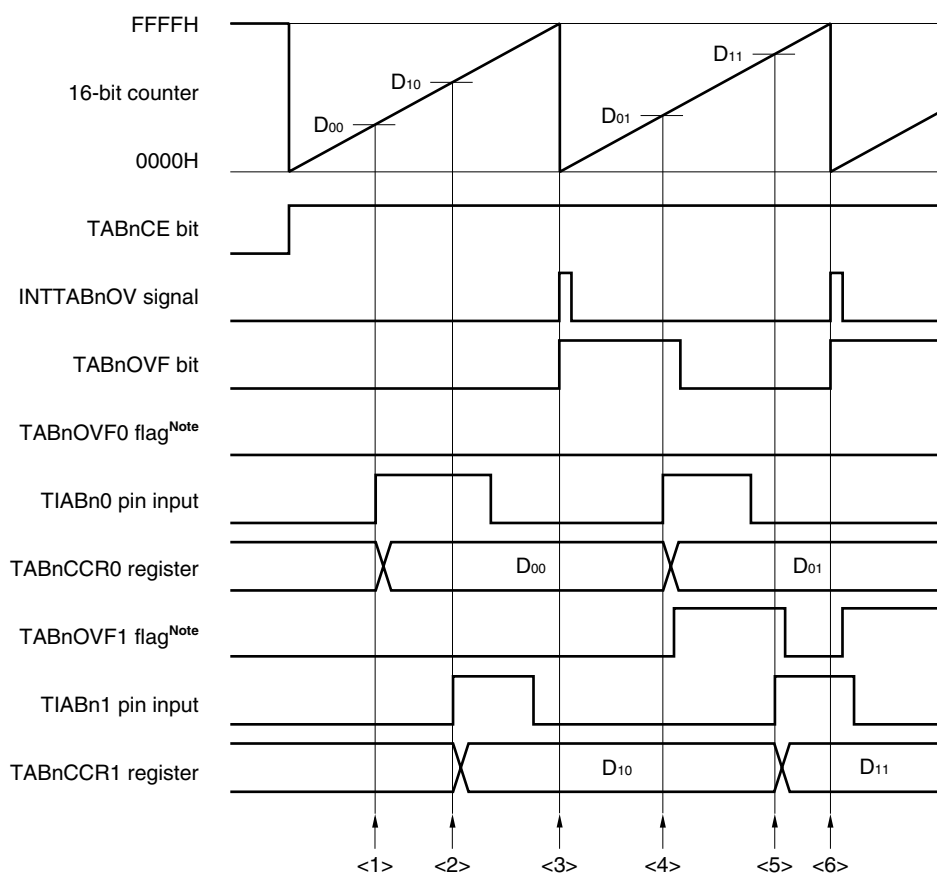
**Example when two capture registers are used (using overflow interrupt)**
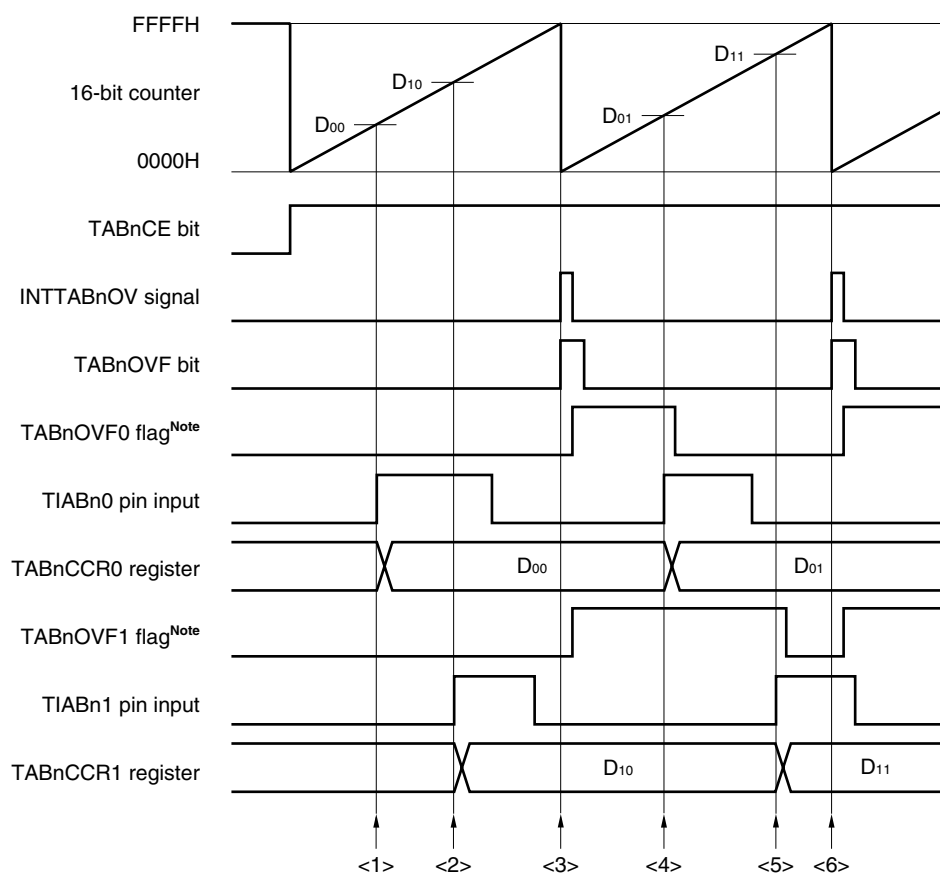


**Note**   The TABnOVF0 and TABnOVF1 flags are set in the internal RAM by software.

<1> Read the TABnCCR0 register (setting of the default value of the TIABn0 pin input).

<2> Read the TABnCCR1 register (setting of the default value of the TIABn1 pin input).

<3> An overflow occurs.  Set the TABnOVF0 and TABnOVF1 flags to 1 in the overflow interrupt servicing, and clear the overflow flag to 0.

<4> Read the TABnCCR0 register.

Read the TABnOVF0 flag.  If the TABnOVF0 flag is 1, clear it to 0.

Because the TABnOVF0 flag is 1, the pulse width can be calculated by ($10000H + D_{01} − D_{00}$).

<5> Read the TABnCCR1 register.

Read the TABnOVF1 flag.  If the TABnOVF1 flag is 1, clear it to 0 (the TABnOVF0 flag is cleared in <4>, and the TABnOVF1 flag remains 1).

Because the TABnOVF1 flag is 1, the pulse width can be calculated by ($10000H + D_{11} − D_{10}$) (correct).

<6> Same as <3>

**Example when two capture registers are used (without using overflow interrupt)**



**Note** The TABnOVF0 and TABnOVF1 flags are set in the internal RAM by software.

<1> Read the TABnCCR0 register (setting of the default value of the TIABn0 pin input).

<2> Read the TABnCCR1 register (setting of the default value of the TIABn1 pin input).

<3> An overflow occurs. Nothing is done by software.

<4> Read the TABnCCR0 register.

Read the overflow flag. If the overflow flag is 1, set only the TABnOVF1 flag to 1, and clear the overflow flag to 0.

Because the overflow flag is 1, the pulse width can be calculated by $(10000H + D_{01} - D_{00})$.

<5> Read the TABnCCR1 register.

Read the overflow flag. Because the overflow flag is cleared in <4>, 0 is read.

Read the TABnOVF1 flag. If the TABnOVF1 flag is 1, clear it to 0.
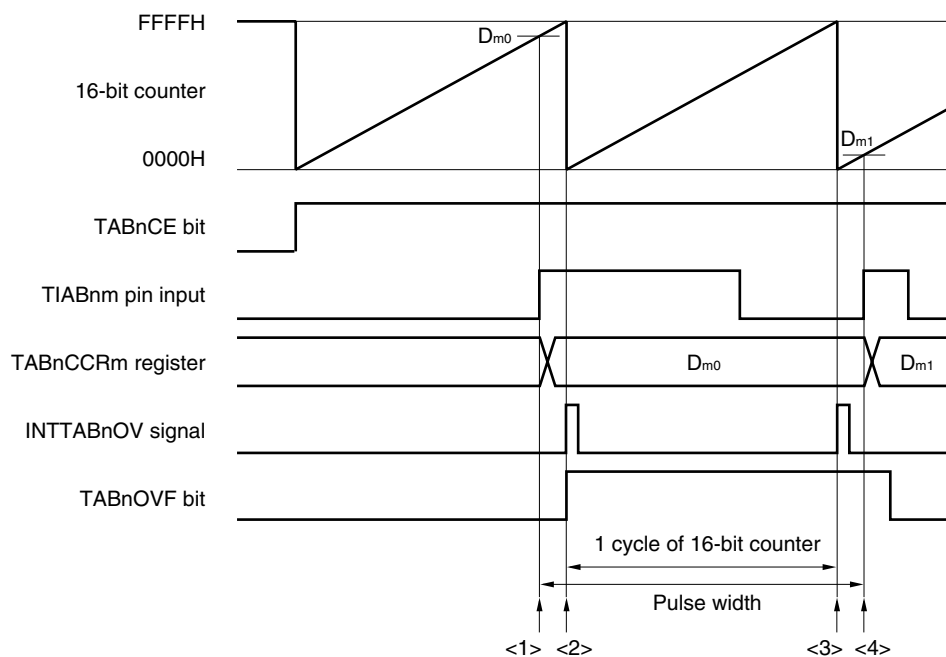
Because the TABnOVF1 flag is 1, the pulse width can be calculated by $(10000H + D_{11} - D_{10})$ (correct).

<6> Same as <3>

**(d) Processing of overflow if capture trigger interval is long**

If the pulse width is greater than one cycle of the 16-bit counter, care must be exercised because an overflow may occur more than once from the first capture trigger to the next. First, an example of incorrect processing is shown below.

---

**Example of incorrect processing when capture trigger interval is long**



The following problem may occur when a long pulse width is measured in the free-running timer mode.
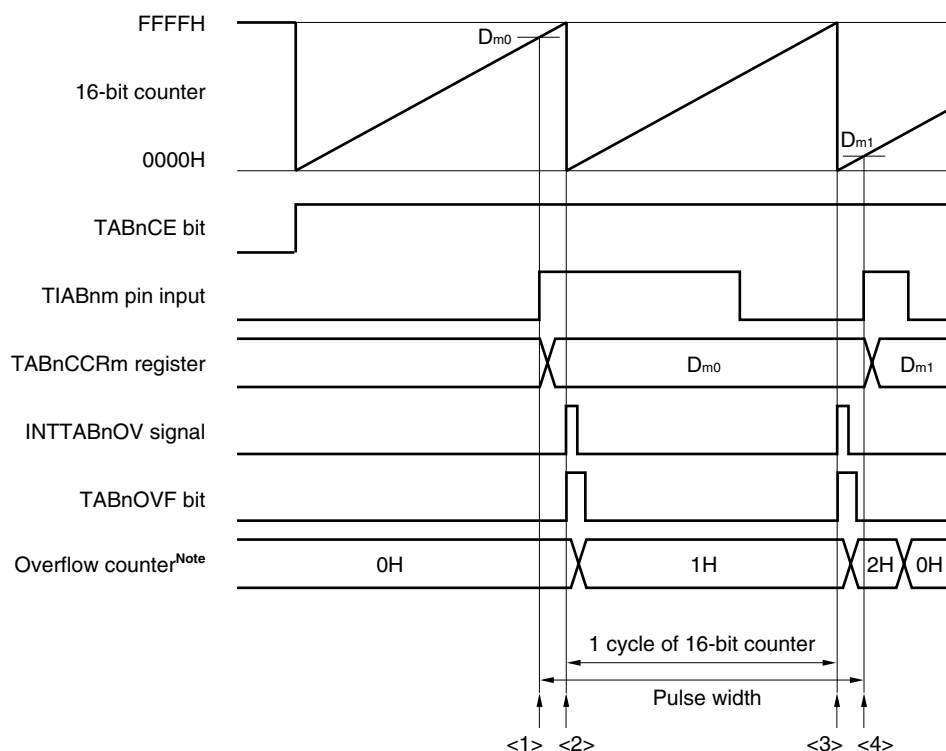
<1> Read the TABnCCRm register (setting of the default value of the TIABnm pin input).

<2> An overflow occurs. Nothing is done by software.

<3> An overflow occurs a second time. Nothing is done by software.

<4> Read the TABnCCRm register.

Read the overflow flag. If the overflow flag is 1, clear it to 0.

Because the overflow flag is 1, the pulse width can be calculated by $(10000H + D_{m1} - D_{m0})$ (incorrect).

Actually, the pulse width must be $(20000H + D_{m1} - D_{m0})$ because an overflow occurs twice.

**Remark**   m = 0 to 3,

              n = 0, 1

---

If an overflow occurs twice or more when the capture trigger interval is long, the correct pulse width may not be obtained.

If the capture trigger interval is long, slow the count clock to lengthen one cycle of the 16-bit counter, or use software. An example of how to use software is shown next.

**Example when capture trigger interval is long**

**Note**  The overflow counter is set arbitrarily by software in the internal RAM.

<1> Read the TABnCCRm register (setting of the default value of the TIABnm pin input).

<2> An overflow occurs.  Increment the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<3> An overflow occurs a second time.  Increment (+1) the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<4> Read the TABnCCRm register.

Read the overflow counter.

When the overflow counter is "N", the pulse width can be calculated by ($N \times 10000H + D_{m1} - D_{m0}$).

In this example, the pulse width is ($20000H + D_{m1} - D_{m0}$) because an overflow occurs twice.

Clear the overflow counter (0H).

**Remark**    m = 0 to 3,
n = 0, 1

**(e) Clearing overflow flag**

The overflow flag can be cleared to 0 by clearing the TABnOVF bit to 0 with the CLR instruction and by writing 8-bit data (bit 0 is 0) to the TABnOPT0 register.  To accurately detect an overflow, read the TABnOVF bit when it is 1, and then clear the overflow flag by using a bit manipulation instruction.



To clear the overflow flag to 0, read the overflow flag to check if it is set to 1, and clear it with the CLR instruction.  If 0 is written to the overflow flag without checking if the flag is 1, the set overflow information may be erased by writing 0 ((ii) in the above chart).  Therefore, software may judge that no overflow has occurred even when an overflow actually has occurred.

If execution of the CLR instruction conflicts with occurrence of an overflow when the overflow flag is cleared to 0 with the CLR instruction, the overflow flag remains set even after execution of the CLR instruction.

### 8.5.7 Pulse width measurement mode (TABnMD2 to TABnMD0 bits = 110)

In the pulse width measurement mode, TABn starts counting when the TABnCTL0.TABnCE bit is set to 1. Each time the valid edge input to the TIABnm pin has been detected, the count value of the 16-bit counter is stored in the TABnCCRm register, and the 16-bit counter is cleared to 0000H.

The interval of the valid edge can be measured by reading the TABnCCRm register after a capture interrupt request signal (INTTABnCCm) occurs.

Select one of the TIABn0 to TIABn3 pins as the capture trigger input pin. Specify "No edge detected" for the unused pins by using the TABnIOC1 register.

When an external clock is used as the count clock, measure the pulse width of the TIAB0k pin because the external clock is fixed to the TIAB00 pin. At this time, clear the TAB0IOC1.TAB0IS1 and TAB0IOC1.TAB0IS0 bits to 00 (capture trigger input (TIAB00 pin): No edge detected).

For TAB1, the external clock is input from the EVTAB1 pin, and the pulse width can be measured by using the TIAB10 to TIAB13 pins.

**Remark**  m = 0 to 3,
n = 0, 1
k = 1 to 3

**Figure 8-34.  Configuration in Pulse Width Measurement Mode**



**Note**  TAB1: EVTAB1 pin

**Remark**  n = 0, 1
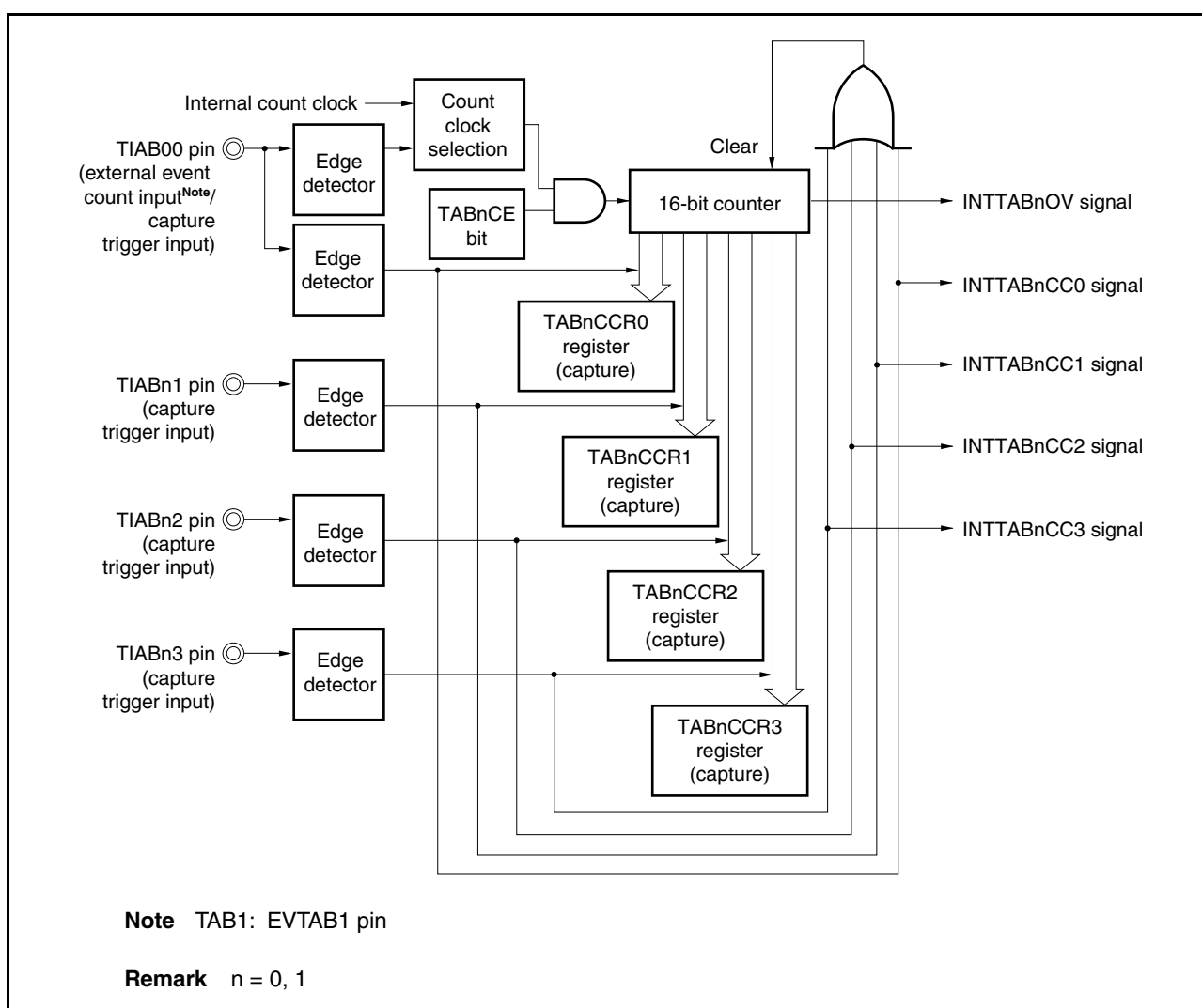
**Figure 8-35. Basic Timing in Pulse Width Measurement Mode**
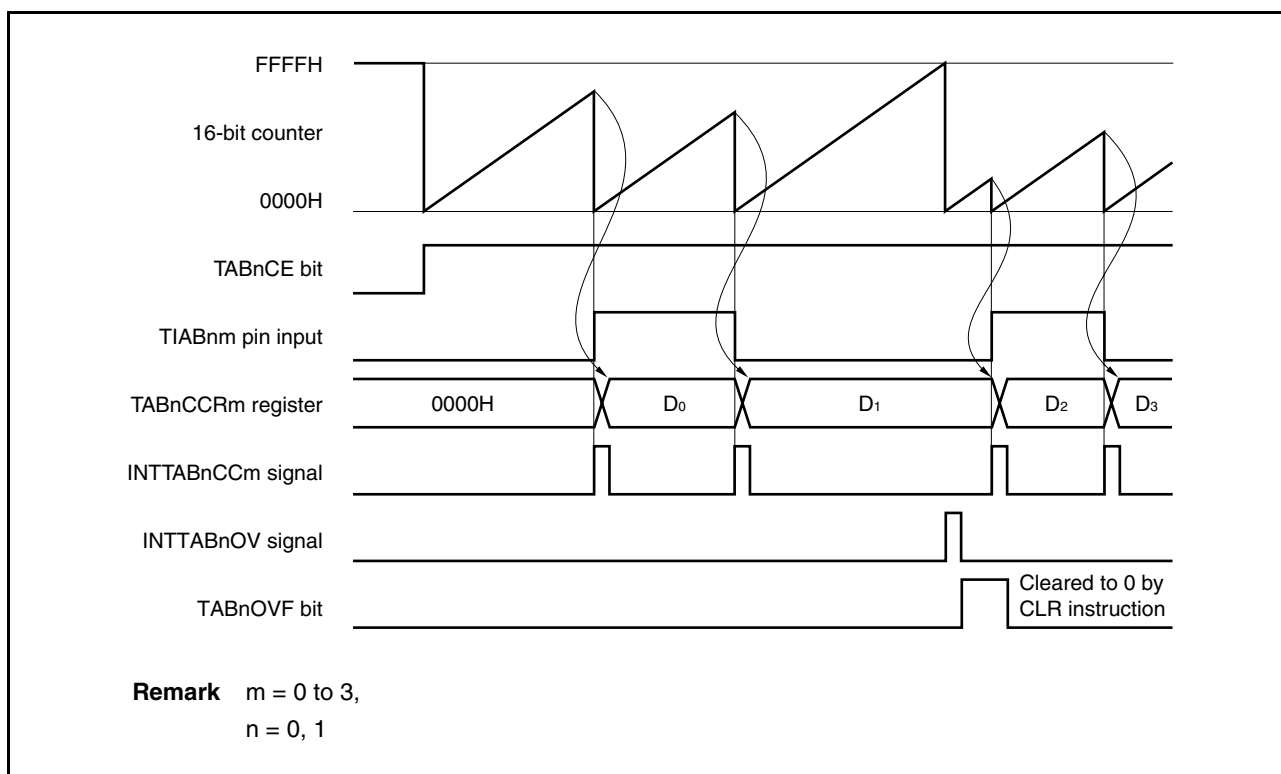


**Remark** m = 0 to 3,
n = 0, 1

When the TABnCE bit is set to 1, the 16-bit counter starts counting. When the valid edge input to the TIABnm pin is later detected, the count value of the 16-bit counter is stored in the TABnCCRm register, the 16-bit counter is cleared to 0000H, and a capture interrupt request signal (INTTABnCCm) is generated.

The pulse width is calculated as follows.

Pulse width = Captured value × Count clock cycle

If the valid edge is not input to the TIABnm pin even when the 16-bit counter has counted up to FFFFH, an overflow interrupt request signal (INTTABnOV) is generated at the next count clock, and the counter is cleared to 0000H and continues counting. At this time, the overflow flag (TABnOPT0.TABnOVF bit) is also set to 1. Clear the overflow flag to 0 by executing the CLR instruction via software.

If the overflow flag is set to 1, the pulse width can be calculated as follows.

Pulse width = (10000H × TABnOVF bit setting (1) count + Captured value) × Count clock cycle

**Remark** m = 0 to 3,
n = 0, 1

**Figure 8-36. Register Setting in Pulse Width Measurement Mode (1/2)**

### (a) TABn control register 0 (TABnCTL0)

TABnCE ... TABnCKS2 TABnCKS1 TABnCKS0

TABnCTL0: 0/1, 0, 0, 0, 0, 0/1, 0/1, 0/1

Select count clock**Note**

0: Stop counting
1: Enable counting
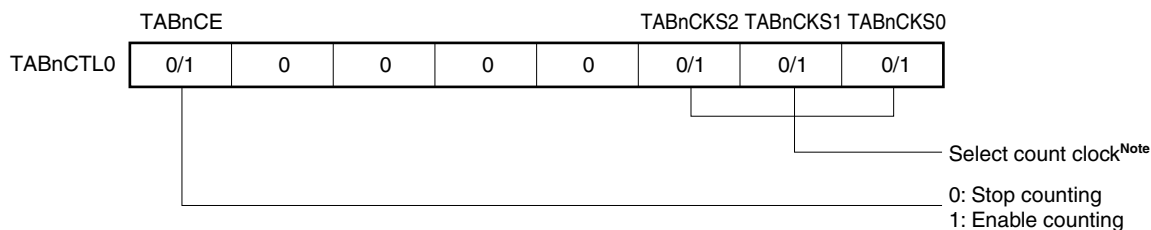
**Note** The setting is invalid when the TABnEEE bit = 1.

### (b) TABn control register 1 (TABnCTL1)

TABnEST TABnEEE ... TABnMD2 TABnMD1 TABnMD0

TABnCTL1: 0, 0, 0/1, 0, 0, 1, 1, 0

1, 1, 0:
Pulse width measurement mode

0: Operate on count clock
selected by TABnCKS0 to
TABnCKS2 bits
1: Count by external event
count input signal

### (c) TABn I/O control register 1 (TABnIOC1)

TABnIS7 TABnIS6 TABnIS5 TABnIS4 TABnIS3 TABnIS2 TABnIS1 TABnIS0

TABnIOC1: 0/1, 0/1, 0/1, 0/1, 0/1, 0/1, 0/1, 0/1

Select valid edge
of TIABn0 pin input

Select valid edge
of TIABn1 pin input

Select valid edge
of TIABn2 pin input

Select valid edge
of TIABn3 pin input

### (d) TABn I/O control register 2 (TABnIOC2)

TABnEES1 TABnEES0 TABnETS1 TABnETS0

TABnIOC2: 0, 0, 0, 0, 0/1, 0/1, 0, 0

Select valid edge of
external event count input

**Remark** n = 0, 1

**Figure 8-36.  Register Setting in Pulse Width Measurement Mode (2/2)**

---

**(e)  TABn option register 0 (TABnOPT0)**

| | TABnCCS3 | TABnCCS2 | TABnCCS1 | TABnCCS0 | | | | TABnOVF |
|---|---|---|---|---|---|---|---|---|
| TABnOPT0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 |

Overflow flag

**(f)  TABn counter read buffer register (TABnCNT)**

The value of the 16-bit counter can be read by reading the TABnCNT register.

**(g)  TABn capture/compare registers 0 to 3 (TABnCCR0 to TABnCCR3)**

These registers store the count value of the 16-bit counter when the valid edge input to the TIABnm pin is detected.

**Remarks  1.** TABn I/O control register 0 (TABnIOC0) is not used in the pulse width measurement mode.

**2.** m = 0 to 3,

n = 0, 1

---

**(1)  Operation flow in pulse width measurement mode**

**Figure 8-37.  Software Processing Flow in Pulse Width Measurement Mode**

**(2) Operation timing in pulse width measurement mode**

**(a) Clearing overflow flag**

The overflow flag can be cleared to 0 by clearing the TABnOVF bit to 0 with the CLR instruction and by writing 8-bit data (bit 0 is 0) to the TABnOPT0 register.  To accurately detect an overflow, read the TABnOVF bit when it is 1, and then clear the overflow flag by using a bit manipulation instruction.



**(i) Operation to write 0 (without conflict with setting)**

Overflow set signal   L

0 write signal

Overflow flag (TABnOVF bit)

**(iii) Operation to clear to 0 (without conflict with setting)**

Overflow set signal   L

0 write signal

Register access signal   Read   Write

Overflow flag (TABnOVF bit)

**(ii) Operation to write 0 (conflict with setting)**

Overflow set signal

0 write signal

Overflow flag (TABnOVF bit)

**(iv) Operation to clear to 0 (conflict with setting)**

Overflow set signal

0 write signal

Register access signal   Read   Write

Overflow flag (TABnOVF bit)   H
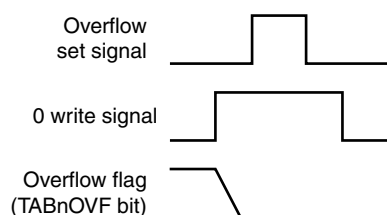
**Remark**   n = 0, 1

To clear the overflow flag to 0, read the overflow flag to check if it is set to 1, and clear it with the CLR instruction.  If 0 is written to the overflow flag without checking if the flag is 1, the set overflow information may be erased by writing 0 ((ii) in the above chart).  Therefore, software may judge that no overflow has occurred even when an overflow actually has occurred.

If execution of the CLR instruction conflicts with occurrence of an overflow when the overflow flag is cleared to 0 with the CLR instruction, the overflow flag remains set even after execution of the CLR instruction.

**8.5.8    Triangular wave PWM mode (TABnMD2 to TABnMD0 bits = 111)**

In the triangular wave PWM mode, TABn capture/compare register k (TABnCCRk) is used to set the duty factor, and TABn capture/compare register 0 (TABnCCR0) is used to set the cycle.

By using these four registers and operating the timer, triangular wave PWM with a variable cycle is output.

The value of the TABnCCRm register can be rewritten when TABnCE = 1.

To stop timer AB, clear TABnCE to 0.  The PWM waveform is output from the TOABnk pin.  The TOABn0 pin produces a toggle output when the value of the 16-bit counter matches the value of the TABnCCR0 register and when the counter underflows.

**Caution   In the PWM mode, the capture function of the TABnCCRm register cannot be used because this register can be used only as a compare register.**

**Remark**    n = 0, 1, m = 0 to 3, k = 1 to 3

**Figure 8-38.  Timing of Basic Operation in Triangular Wave PWM Mode**
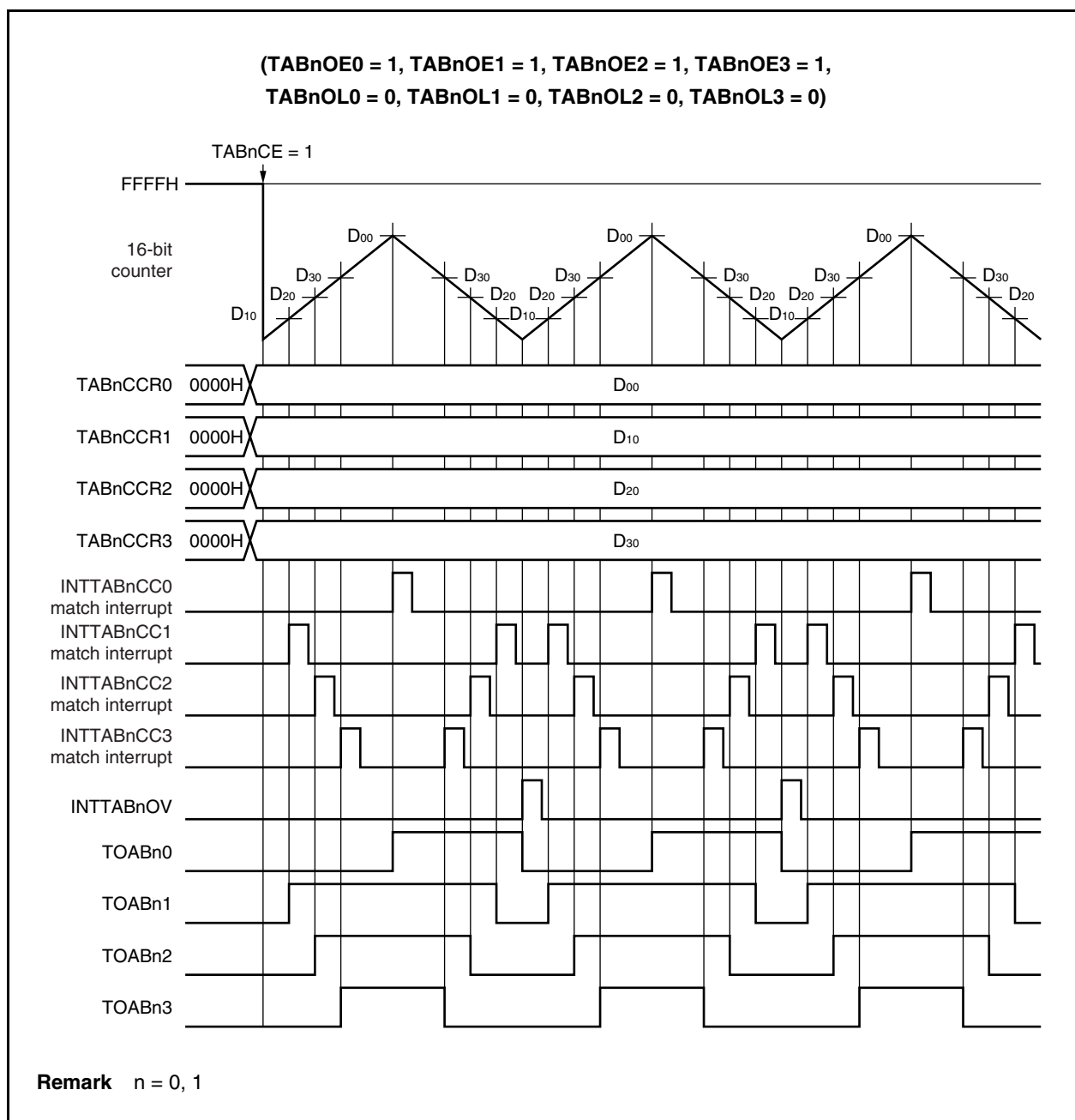
### 8.5.9 Timer output operations

The following table shows the operations and output levels of the TOABn0 to TOABn3 pins.

**Table 8-6. Timer Output Control in Each Mode**

| Operation Mode | TOABn0 Pin | TOABn1 Pin | TOABn2 Pin | TOABn3 Pin |
|---|---|---|---|---|
| Interval timer mode | Square wave output | | | |
| External event count mode | Square wave output | − | | |
| External trigger pulse output mode | Square wave output | External trigger pulse output | External trigger pulse output | External trigger pulse output |
| One-shot pulse output mode | | One-shot pulse output | One-shot pulse output | One-shot pulse output |
| PWM output mode | | PWM output | PWM output | PWM output |
| Free-running timer mode | Square wave output (only when compare function is used) | | | |
| Pulse width measurement mode | − | | | |
| Triangular wave PWM output mode | Square wave output | Triangular PWM output | Triangular PWM output | Triangular PWM output |

**Table 8-7. Truth Table of TOABn0 to TOABn3 Pins Under Control of Timer Output Control Bits**

| TABnIOC0.TABnOLm Bit | TABnIOC0.TABnOEm Bit | TABnCTL0.TABnCE Bit | Level of TOABnm Pin |
|---|---|---|---|
| 0 | 0 | × | Low-level output |
| | 1 | 0 | Low-level output |
| | | 1 | Low level immediately before counting, high level after counting is started |
| 1 | 0 | × | High-level output |
| | 1 | 0 | High-level output |
| | | 1 | High level immediately before counting, low level after counting is started |

**Remark**    m = 0 to 3,

          n = 0, 1

## 8.6  Timer-Tuned Operation Function/Simultaneous-Start Function

Timer AA and timer AB have a timer-tuned operation function/simultaneous-start function.

The timers that can be synchronized are listed in Table 8-8.

**Table 8-8.  Timer-Tuned Operation Mode**

| Master Timer | Slave Timer |
|---|---|
| TAA1 | TAA0 |
| TAA3 | TAA2 |
| TAB0 | TAA5 |

For details of the timer-tuned operation function, see **7.6  Timer-Tuned Operation Function**, and for details of the simultaneous-start function, see **7.7  Simultaneous-Start Function**.

## 8.7 Cautions

### (1) Capture operation

When the capture operation is used and a slow clock is selected as the count clock, FFFFH, not 0000H, may be captured in the TABnCCR0, TABnCCR1, TABnCCR2, and TABnCCR3 registers if the capture trigger is input immediately after the TABnCE bit is set to 1.

**(a) Free-running timer mode**
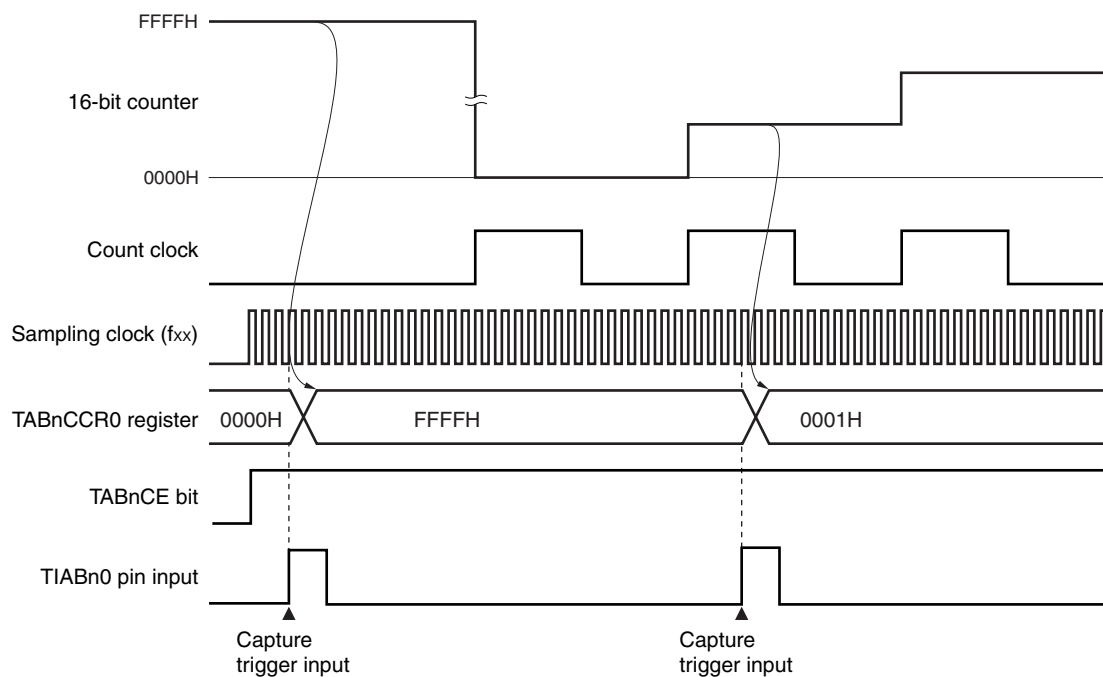
**(b) Pulse width measurement mode**
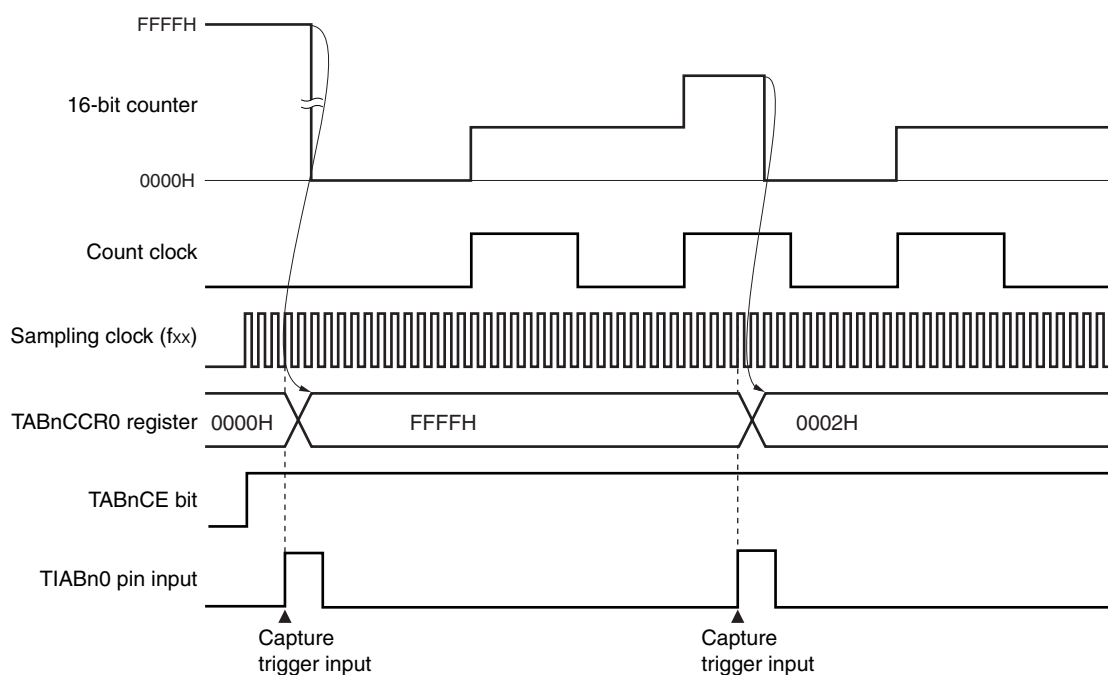
**Remark**   n = 0, 1

## CHAPTER  9  16-BIT  TIMER/EVENT  COUNTER  T  (TMT)

Timer T (TMT) is a 16-bit timer/event counter.

An encoder count function and other functions are added to timer AA (TAA).  However, TMT does not have a function to operate with an external event count input when it operates in the interval timer mode.

The V850ES/JG3-U and V850ES/JH3-U have one TMT channel.

### 9.1   Overview

An overview of TMT0 is given below.

- Clock selection:                                      8 types
- Capture trigger input pins (TIT00, TIT01):            2
- External event count input pin (EVTT0):              1
- Encoder input pins (TENC00, TENC01):                 2
- Encoder clear input pin (TECR0):                     1
- External trigger input pin[Note]:                    1
- Timer counter:                                       1
- Capture/compare registers:                           2
- Capture/compare match interrupt request signals:    2
- Timer output pins:                                   2

**Note**   The external trigger input pin and external event count input pin (EVTT0) are shared with an encoder input pin (TENC00).

### 9.2   Functions

The functions of TMT0 are shown below.

- Interval timer
- External event counter
- External trigger pulse output
- One-shot pulse output
- PWM output
- Free-running timer
- Pulse width measurement
- Triangular-wave PWM output
- Encoder count

## 9.3    Configuration

TMT0 includes the following hardware.

**Table 9-1.  Configuration of TMT0**

| Item | Configuration |
|---|---|
| Registers | 16-bit counter $\times$ 1<br>TMT0 capture/compare registers 0, 1 (TT0CCR0, TT0CCR1)<br>TMT0 counter read buffer register (TT0CNT)<br>TMT0 counter write register (TT0TCW)<br>CCR0, CCR1 buffer registers<br>TMT0 control registers 0, 1 (TT0CTL0, TT0CTL1)<br>TMT0 control registers 2 (TT0CTL2)<br>TMT0 I/O control registers 0 to 3 (TT0IOC0 to TT0IOC3)<br>TMT0 option register 0 (TT0OPT0)<br>TMT0 option register 1 (TT0OPT1)<br>TMT noise elimination control register (TTNFC) |
| Timer input | • TIT00, TIT01 (capture trigger input pins)<br>• EVTT0/TENC00 (external event input/encoder 0 input pin)[Note]<br>• TENC01 (encoder 1 input pin)<br>• TENCR0 (encoder clear input pin) |
| Timer output | TOT00, TOT01 |

**Note**   Shared with the external trigger input function

**Figure 9-1.  Block Diagram of TMT0**
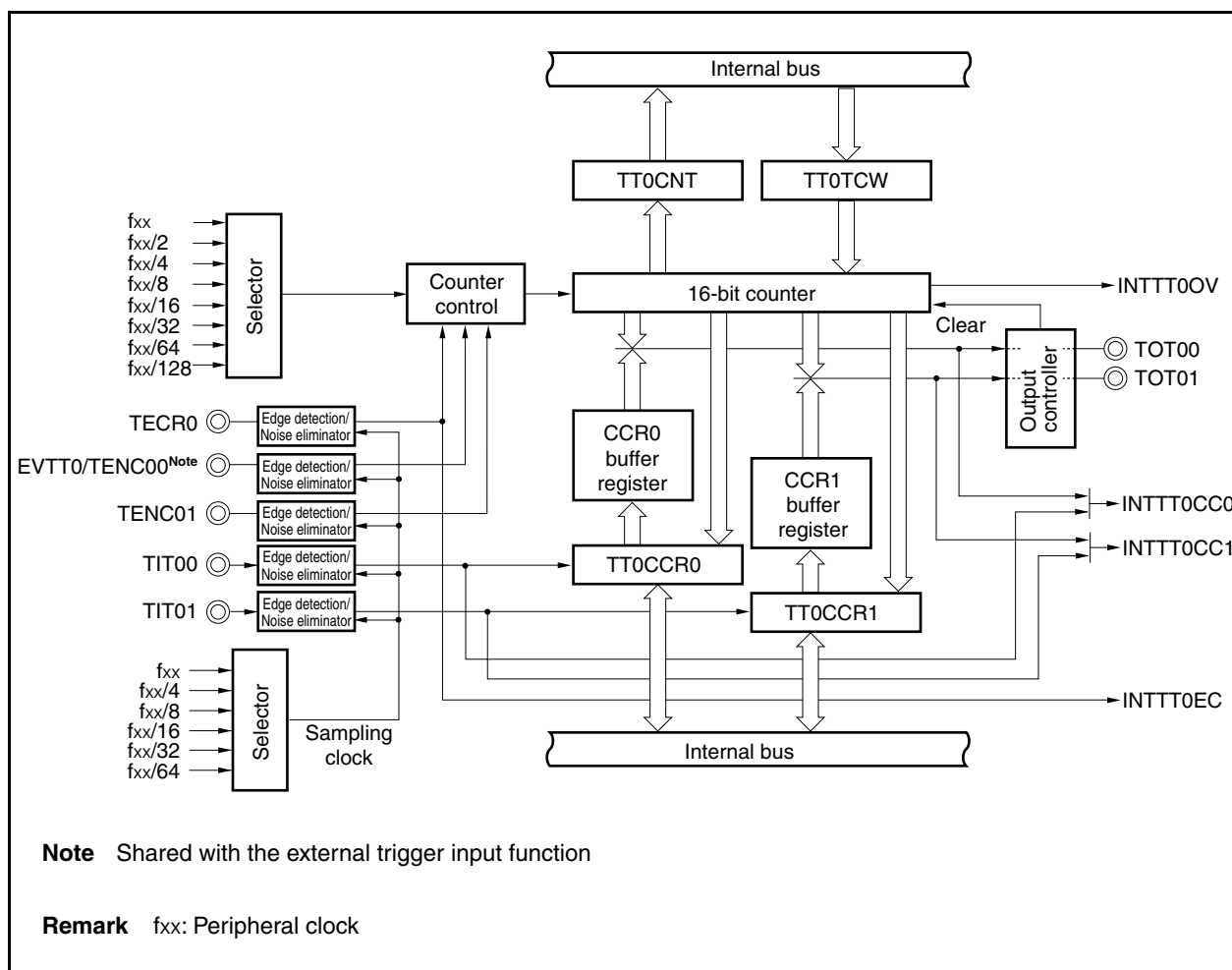


**Note**   Shared with the external trigger input function

**Remark**   fxx: Peripheral clock

**(1) 16-bit counter**

This 16-bit counter can count internal clocks or external events.

The count value of this counter can be read by using the TT0CNT register.

When the TT0CTL0.TT0CE bit = 0, the value of the 16-bit counter is FFFFH. If the TT0CNT register is read at this time, 0000H is read.

Reset sets the TT0CE bit to 0.

**(2) CCR0 buffer register**

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TT0CCR0 register is used as a compare register, the value written to the TT0CCR0 register is transferred to the CCR0 buffer register. When the count value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTTCC00) is generated.

The CCR0 buffer register cannot be read or written directly.

The CCR0 buffer register is set to 0000H after reset, and the TT0CCR0 register is set to 0000H.

**(3) CCR1 buffer register**

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TT0CCR1 register is used as a compare register, the value written to the TT0CCR1 register is transferred to the CCR1 buffer register. When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTTCC01) is generated.

The CCR1 buffer register cannot be read or written directly.

The CCR1 buffer register is set to 0000H after reset, and the TT0CCR1 register is set to 0000H.

**(4) Edge detector**

This circuit detects the valid edges input to the TIT00, TIT01, EVTT0/TENC00, TENC01, and TECR0 pins. No edge, rising edge, falling edge, or both the rising and falling edges can be selected as the valid edge by using the TT0IOC1, TT0IOC2, and TT0IOC3 registers.

**(5) Output controller**

This circuit controls the output of the TOT00 and TOT01 pins via the TT0IOC0 register.

**(6) Selector**

This selector selects the count clock for the 16-bit counter. Eight types of internal clocks or an external event can be selected as the count clock.

**(7) Counter control**

The count operation is controlled by the timer mode selected by the TT0CTL1 register.

### 9.3.1 Pin configuration

The timer inputs and outputs that configure TMT0 are shared with the following ports. The port functions must be set when using each pin (see **Table 4-20 Using Port Pin as Alternate-Function Pin**).

**Table 9-2. Pin Configuration**

| Port | Timer Input Pin | | Timer Output | Other Alternate Function |
|------|------|------|------|------|
| P92 | TIT01 (capture trigger input 1) | TENC01 (encoder input) | TOT01 | A2[Note 1] |
| P93 | TIT00 (capture trigger input 0) | TECR0 (encoder clear input) | TOT00 | A3[Note 1] |
| P94 | EVTT0/TENC00[Note 2] | – | – | – |

**Notes 1.** V850ES/JH3-U only

**2.** The external event count input (EVTT0), encoder input (TENC00), and external trigger input are shared in a state that cannot be controlled by using the port functions. To use each function, set them by using the TT0IOC2 and TT0IOC3 registers after setting their corresponding ports.

## 9.4   Registers

**(1)  TMT0 control register 0 (TT0CTL0)**

The TT0CTL0 register is an 8-bit register that controls the operation of TMT0.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

The same value can always be written to the TT0CTL0 register by software.

After reset: 00H      R/W      Address: FFFF600H

| | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TT0CTL0 | TT0CE | 0 | 0 | 0 | 0 | TT0CKS2 | TT0CKS1 | TT0CKS0 |

| TT0CE | TMT0 operation control |
|---|---|
| 0 | TMT0 operation disabled (TMT0 reset asynchronously[Note]) |
| 1 | TMT0 operation enabled.  TMT0 operation start |

| TT0CKS2 | TT0CKS1 | TT0CKS0 | Internal count clock selection |
|---|---|---|---|
| 0 | 0 | 0 | $f_{xx}$ |
| 0 | 0 | 1 | $f_{xx}/2$ |
| 0 | 1 | 0 | $f_{xx}/4$ |
| 0 | 1 | 1 | $f_{xx}/8$ |
| 1 | 0 | 0 | $f_{xx}/16$ |
| 1 | 0 | 1 | $f_{xx}/32$ |
| 1 | 1 | 0 | $f_{xx}/64$ |
| 1 | 1 | 1 | $f_{xx}/128$ |

**Note**   The TT0OPT0.TT0OVF bit and 16-bit counter are reset simultaneously. Moreover, timer outputs (TOT00 and TOT01) are reset at the same time as the 16-bit counter.

**Cautions  1.  Set the TT0CKS2 to TT0CKS0 bits when the TT0CE bit = 0.**
**When the value of the TT0CE bit is changed from 0 to 1, the TT0CKS2 to TT0CKS0 bits can be set simultaneously.**
**2.  Be sure to set bits 3 to 6 to "0".**

**Remark**   $f_{xx}$: Peripheral clock

**(2) TMT0 control register 1 (TT0CTL1)**

The TT0CTL1 register is an 8-bit register that controls the TMT0 operation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

(1/2)

After reset: 00H    R/W    Address: FFFFF601H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TT0CTL1 | 0 | TT0EST | TT0EEE | 0 | TT0MD3 | TT0MD2 | TT0MD1 | TT0MD0 |

| TT0EST | Software trigger control |
|---|---|
| 0 | – |
| 1 | Generates a valid signal for external trigger input.<br>• In one-shot pulse output mode: A one-shot pulse is output with writing 1 to the TT0EST bit as the trigger.<br>• In external trigger pulse output mode: A PWM waveform is output with writing 1 to the TT0EST bit as the trigger. |

The read value of the TT0EST bit is always 0.

| TT0EEE | Count clock selection |
|---|---|
| 0 | Disables operation with external event count input (EVTT0 pin).<br>(Performs counting with the count clock selected by the TT0CTL0.TT0CKS0 to TT0CTL0.TT0CKS2 bits.) |
| 1 | Enables operation with external event count input (EVTT0 pin).<br>(Performs counting at every valid edge of the external event count input signal (EVTT0 pin).) |

The TT0EEE bit selects whether counting is performed with the internal count clock or the valid edge of the external event count input.

| TT0MD3 | TT0MD2 | TT0MD1 | TT0MD0 | Timer mode selection |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Interval timer mode |
| 0 | 0 | 0 | 1 | External event count mode |
| 0 | 0 | 1 | 0 | External trigger pulse output mode |
| 0 | 0 | 1 | 1 | One-shot pulse output mode |
| 0 | 1 | 0 | 0 | PWM output mode |
| 0 | 1 | 0 | 1 | Free-running timer mode |
| 0 | 1 | 1 | 0 | Pulse width measurement mode |
| 0 | 1 | 1 | 1 | Triangular-wave PWM output mode |
| 1 | 0 | 0 | 0 | Encoder compare mode |
| Other than above | | | | Setting prohibited |

**Cautions 1.** The TT0EST bit is valid only in the external trigger pulse output mode or one-shot pulse output mode.  In any other mode, writing 1 to this bit is ignored.

**2.** The TT0EEE bit is valid only in the interval timer mode, external trigger pulse output mode, one-shot pulse output mode, PWM output mode, free-running timer mode, pulse width measurement mode, or triangular-wave PWM output mode.  In any other mode, writing 1 to this bit is ignored.

**3.** External event count input (EVTT0) or encoder inputs (TENC00, TENC01) is selected in the external event count mode or encoder compare mode regardless of the value of the TT0EEE bit.

**4.** Set the TT0EEE and TT0MD3 to TT0MD0 bits when the TT0CTL0.TT0CE bit = 0.  (The same value can be written when the TT0CE bit = 1.)  The operation is not guaranteed when rewriting is performed with the TT0CE bit = 1.  If rewriting was mistakenly performed, clear the TT0CE bit to 0 and then set the bits again.

**5.** Be sure to set bits 4 and 7 to "0".

**(3) TMT0 control register 2 (TT0CTL2)**

The TT0CTL2 register is an 8-bit register that controls the encoder count function operation.

The TT0CTL2 register is valid only in the encoder compare mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

**Caution   For details of each bit of the TT0CTL2 register, see 9.6.9 (5) Controlling bits of TT0CTL2 register.**

(1/2)

After reset: 00H    R/W    Address: FFFFF602H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TT0CTL2 | TT0ECC | 0 | 0 | TT0LDE | TT0ECM1 | TT0ECM0 | TT0UDS1 | TT0UDS0 |

| TT0ECC | Encoder counter control |
|---|---|
| 0 | Normal operation |
| 1 | Holds count value of 16-bit counter when TT0CTL0.TT0CE bit = 0. |

| TT0LDE | Transfer setting to 16-bit counter |
|---|---|
| 0 | Disables transfer of set value of TT0CCR0 to 16-bit counter in case of underflow. |
| 1 | Enables transfer of set value of TT0CCR0 to 16-bit counter in case of underflow. |

| TT0ECM1 | Control of encoder clear operation 1 |
|---|---|
| 0 | The 16-bit counter is not cleared to 0000H when its count value matches value of CCR1 register. |
| 1 | The 16-bit counter is cleared to 0000H when the count after a match between the 16-bit counter count value and CCR1 register value is a down-count |

| TT0ECM0 | Control of encoder clear operation 0 |
|---|---|
| 0 | The 16-bit counter is not cleared to 0000H when its count value matches value of CCR0 register. |
| 1 | The 16-bit counter is cleared to 0000H when the count after a match between the 16-bit counter count value and CCR0 register value is an up-count |

| TT0UDS1 | TT0UDS0 | Up/down count selection |
|---------|---------|-------------------------|
| 0 | 0 | When valid edge of TENC00 input is detected<br>Counts down when TENC01 = high level.<br>Counts up when TENC01 = low level. |
| 0 | 1 | Counts up when valid edge of TENC00 input is detected.<br>Counts down when valid edge of TENC01 input is detected. |
| 1 | 0 | Counts down when rising edge of TENC00 input is detected.<br>Counts up when falling edge of TENC00 input is detected.<br>However, count operation is performed only when<br>TENC01 = low level. |
| 1 | 1 | Both rising and falling edges of TENC00 and TENC01 are<br>detected.  Count operation is automatically identified by<br>combination of edge detection and level detection. |

**Cautions 1.** The TT0ECC bit is valid only in the encoder compare mode.  In any other mode, writing "1" to this bit is ignored.

If the TT0CTL0.TT0CE bit is cleared to 0 while the TT0ECC bit = 1, the values of the timer/counter and capture registers (TT0CCR0 and TT0CCR1), and the TT0OPT1, TT0EUF, TT0EOF, and TT0ESF flags are retained.

If the TT0CE bit is set from 0 to 1 when the TT0ECC bit = 1, the value of the TT0TCW register is not transferred to the 16-bit counter.

**2.** The TT0LDE bit is valid only when the TT0ECM1 and TT0ECM0 bits = 00, 01.  Writing "1" to this bit is ignored when the TT0ECM1 and TT0ECM0 bits = 10, 11.

**3.** The edge detection of the TENC00 and TENC01 inputs specified by the TT0IOC3.TT0EIS1 and TT0IOC3.TT0EIS0 bits is invalid and fixed to both the rising and falling edges when the TT0UDS1 and TT0UDS0 bits = 10, 11.

**4.** Set the TT0LDE, TT0ECM1, TT0ECM0, TT0UDS1, and TT0UDS0 bits when the TT0CTL0.TT0CE bit = 0 (the same value can be written to these bits when the TT0CE bit = 1).  If the value of these bits is changed when the TT0CE bit = 1, the operation cannot be guaranteed.  If it is changed by mistake, clear the TT0CE bit and then set the correct value.

**5.** Be sure to set bits 5 and 6 to "0".

**(4) TMT0 I/O control register 0 (TT0IOC0)**

The TT0IOC0 register is an 8-bit register that controls the timer output (TOT00, TOT01 pins).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H　　R/W　　Address: FFFFF603H

| | 7 | 6 | 5 | 4 | 3 | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| TT0IOC0 | 0 | 0 | 0 | 0 | TT0OL1 | TT0OE1 | TT0OL0 | TT0OE0 |

| TT0OL1 | TOT01 pin output level setting[Note] |
|---|---|
| 0 | TOT01 pin starts output at high level. |
| 1 | TOT01 pin starts output at low level. |

| TT0OE1 | TOT01 pin output setting |
|---|---|
| 0 | Timer output prohibited<br>• Low level is output from the TOT01 pin when the TT0OL1 bit = 0.<br>• High level is output from the TOT01 pin when the TT0OL1 bit = 1. |
| 1 | Timer output enabled (A pulse is output from the TOT01 pin.) |

| TT0OL0 | TOT00 pin output level setting[Note] |
|---|---|
| 0 | TOT00 pin starts output at high level. |
| 1 | TOT00 pin starts output at low level. |

| TT0OE0 | TOT00 pin output setting |
|---|---|
| 0 | Timer output prohibited<br>• Low level is output from the TOT00 pin when the TT0OL0 bit = 0.<br>• High level is output from the TOT00 pin when the TT0OL0 bit = 1. |
| 1 | Timer output enabled (A pulse is output from the TOT00 pin.) |

**Note** The output level of the timer output pins (TOT00 and TOT01) specified by the TT0OLn bit is shown below (n = 0, 1).

• When TT0OLn bit = 0　　　　　　　　• When TT0OLn bit = 1

16-bit counter　　　　　　　　　　　　16-bit counter

TT0CE bit　　　　　　　　　　　　　　TT0CE bit

TOT0n pin output　　　　　　　　　　TOT0n pin output
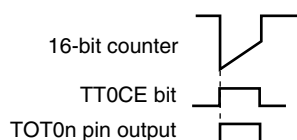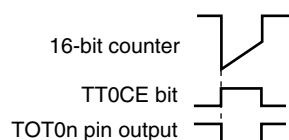
**Cautions 1.** **If the setting of the TT0IOC0 register is changed when TOT00 and TOT01 outputs are set for the port mode, the output of the pins change. Set the port in the input mode and make the port go into a high-impedance state, noting changes in the pin status.**

　　　　　**2.** **Rewrite the TT0OL1, TT0OE1, TT0OL0, and TT0OE0 bits when the TT0CTL0.TT0CE bit = 0. (The same value can be written when the TT0CE bit = 1.) If rewriting was mistakenly performed, clear the TT0CE bit to 0 and then set the bits again.**

　　　　　**3.** **Even if the TT0OL0 or TT0OL1 bit is manipulated when the TT0CE, TT0OE0, and TT0OE1 bits are 0, the output level of the TOT00 and TOT01 pins changes.**

**(5)  TMT0 I/O control register 1 (TT0IOC1)**

The TT0IOC1 register is an 8-bit register that controls the valid edge for the capture trigger input signals (TIT00, TIT01 pins).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address: FFFFF604H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TT0IOC1 | 0 | 0 | 0 | 0 | TT0IS3 | TT0IS2 | TT0IS1 | TT0IS0 |

| TT0IS3 | TT0IS2 | Capture trigger input signal (TIT01 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (capture operation invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

| TT0IS1 | TT0IS0 | Capture trigger input signal (TIT00 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (capture operation invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

**Cautions  1.  Rewrite the TT0IS3 to TT0IS0 bits when the TT0CTL0.TT0CE bit = 0. (The same value can be written when the TT0CE bit = 1.)  If rewriting was mistakenly performed, clear the TT0CE bit to 0 and then set the bits again.**

**2.  The TT0IS3 and TT0IS2 bits are valid only in the free-running timer mode (only when the TT0OPT0.TT0CCS1 bit = 1) and the pulse width measurement mode.  In all other modes, a capture operation is not performed.**

**The TT0IS1 and TT0IS0 bits are valid only in the free-running timer mode (only when the TT0OPT0. TT0CCS0 bit = 1) and the pulse width measurement mode.  In all other modes, a capture operation is not performed.**

**(6) TMT0 I/O control register 2 (TT0IOC2)**

The TT0IOC2 register is an 8-bit register that controls the valid edge for the external event count input signal (EVTT0 pin) and external trigger input signal (EVTT0 pin).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address: FFFFF605H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TT0IOC2 | 0 | 0 | 0 | 0 | TT0EES1 | TT0EES0 | TT0ETS1 | TT0ETS0 |

| TT0EES1 | TT0EES0 | External event count input signal (EVTT0 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (external event count invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

| TT0ETS1 | TT0ETS0 | External trigger input signal (EVTT0 pin) valid edge setting |
|---|---|---|
| 0 | 0 | No edge detection (external trigger invalid) |
| 0 | 1 | Detection of rising edge |
| 1 | 0 | Detection of falling edge |
| 1 | 1 | Detection of both edges |

**Cautions 1. Rewrite the TT0EES1, TT0EES0, TT0ETS1, and TT0ETS0 bits when the TT0CTL0.TT0CE bit = 0. (The same value can be written when the TT0CE bit = 1.) If rewriting was mistakenly performed, clear the TT0CE bit to 0 and then set the bits again.**

**2. The TT0EES1 and TT0EES0 bits are valid only when the TT0CTL1.TT0EEE bit = 1 or when the external event count mode (the TT0CTL1.TT0MD3 to TT0CTL1.TT0MD0 bits = 0001) has been set.**

**3. The TT0ETS1 and TT0ETS0 bits are valid only in the external trigger pulse mode or one-shot pulse output mode.**

**(7) TMT0 I/O control register 3 (TT0IOC3)**

The TT0IOC3 register is an 8-bit register that controls the encoder clear function operation.

The TT0IOC3 register is valid only in the encoder compare mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

(1/2)

After reset: 00H　　R/W　　Address: FFFFF606H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TT0IOC3 | TT0SCE | TT0ZCL | TT0BCL | TT0ACL | TT0ECS1 | TT0ECS0 | TT0EIS1 | TT0EIS0 |

| TT0SCE | Encoder clear selection |
|---|---|
| 0 | Clears 16-bit counter on detection of edge of encoder clear signal (TECR0 pin). |
| 1 | Clears 16-bit counter on detection of clear level condition of the TENC00, TENC01, and TECR0 pins. |

- Clears the 16-bit counter to 0000H when the valid edge of TECR0 pin specified by the TT0ECS1 and TT0ECS0 bits is detected when the TT0SCE bit = 0.
- Clears the 16-bit counter to 0000H when the clear level conditions of the TT0ZCL, TT0BCL, and TT0ACL bits match the input levels of the TECR0, TENC01, and TENC00 pins when TT0SCE bit = 1.
- Setting of the TT0ZCL, TT0BCL, and TT0ACL bits is valid and that of the TT0ECS1 and TT0ECS0 bits is invalid when the TT0SCE bit = 1.
  An encoder clear interrupt request signal (INTTT0EC) is not generated.
- Setting of the TT0ZCL, TT0BCL, and TT0ACL bits is invalid and setting of the TT0ECS1 and TT0ECS0 bits is valid when the TT0SCE bit = 0.
  The INTTT0EC signal is generated when the valid edge specified by the TT0ECS1 and TT0ECS0 bits is detected.
- Be sure to set the TT0CTL2.TT0UDS1 and TT0CTL2.TT0UDS0 bits to 10 or 11 when the TT0SCE bit = 1.
  Operation is not guaranteed if the TT0UDS1 and TT0UDS0 bits = 00 or 01 and the TT0SCE bit = 1.

| TT0ZCL | Clear level selection of encoder clear signal (TECR0 pin) |
|---|---|
| 0 | Clears low level of the TECR0 pin. |
| 1 | Clears high level of the TECR0 pin. |
| Setting of the TT0ZCL bit is valid only when the TT0SCE bit = 1. | |

| TT0BCL | Clear level selection of encoder input signal (TENC01 pin) |
|---|---|
| 0 | Clears low level of the TENC01 pin. |
| 1 | Clears high level of the TENC01 pin. |
| Setting of the TT0BCL bit is valid only when the TT0SCE bit = 1. | |

| TT0ACL | Clear level selection of encoder input signal (TENC00 pin) |
|---|---|
| 0 | Clears low level of the TENC00 pin. |
| 1 | Clears high level of the TENC00 pin. |
| Setting of the TT0ACL bit is valid only when the TT0SCE bit = 1. | |

| TT0ECS1 | TT0ECS0 | Valid edge setting of encoder clear signal (TECR0 pin) |
|---------|---------|---------------------------------------------------------|
| 0 | 0 | Detects no edge (clearing encoder is invalid). |
| 0 | 1 | Detects rising edge. |
| 1 | 0 | Detects falling edge. |
| 1 | 1 | Detects both edges. |

| TT0EIS1 | TT0EIS0 | Valid edge setting of encoder input signals (TENC00, TENC01 pins) |
|---------|---------|-------------------------------------------------------------------|
| 0 | 0 | Detects no edge (inputting encoder is invalid). |
| 0 | 1 | Detects rising edge. |
| 1 | 0 | Detects falling edge. |
| 1 | 1 | Detects both edges. |

**Cautions 1. Rewrite the TT0SCE, TT0ZCL, TT0BCL, TT0ACL, TT0ECS1, TT0ECS0, TT0EIS1, and TT0EIS0 bits when the TT0CTL0.TT0CE bit = 0. (The same value can be written to these bits when the TT0CE bit = 1.) If rewriting was mistakenly performed, clear the TT0CE bit to 0 and then set these bits again.**

**2. The TT0ECS1 and TT0ECS0 bits are valid only when the TT0SCE bit = 0 and the encoder compare mode is set.**

**3. The TT0EIS1 and TT0EIS0 bits are valid only when the TT0CTL2.TT0UDS1 and TT0CTL2.TT0UDS0 bits = 00 or 01.**

**(8)  TMT0 option register 0 (TT0OPT0)**

The TT0OPT0 register is an 8-bit register that sets the capture/compare operation and detects overflows.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address: FFFFF607H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| TT0OPT0 | 0 | 0 | TT0CCS1 | TT0CCS0 | 0 | 0 | 0 | TT0OVF |

| TT0CCS1 | TT0CCR1 register capture/compare selection |
|---|---|
| 0 | Selected as compare register |
| 1 | Selected as capture register (cleared by the TT0CTL0.TT0CE bit = 0) |
| The TT0CCS1 bit setting is valid only in the free-running timer mode. | |

| TT0CCS0 | TT0CCR0 register capture/compare selection |
|---|---|
| 0 | Selected as compare register |
| 1 | Selected as capture register (cleared by the TT0CTL0.TT0CE bit = 0) |
| The TT0CCS0 bit setting is valid only in the free-running timer mode. | |

| TT0OVF | TMT0 overflow detection flag |
|---|---|
| Set (1) | Overflow occurred |
| Reset (0) | 0 written to TT0OVF bit or TT0CTL0.TT0CE bit = 0 |

- The TT0OVF bit is set to 1 when the 16-bit counter value overflows from FFFFH to 0000H in the free-running timer mode or the pulse width measurement mode.
- An overflow interrupt request signal (INTTT0OV) is generated when the TT0OVF bit is set to 1.  The INTTT0OV signal is not generated in modes other than the free-running timer mode and the pulse width measurement mode.
- The TT0OVF bit is not cleared to 0 even when the TT0OVF bit or the TT0OPT0 register are read when the TT0OVF bit = 1.
- Before clearing the TT0OVF bit to 0 after generation of the INTTT0OV signal, be sure to confirm (by reading) that the TT0OVF bit is set to 1.
- The TT0OVF bit can be both read and written, but the TT0OVF bit cannot be set to 1 by software.  Writing 1 has no effect on the operation of TMT0.

**Cautions  1.  Rewrite the TT0CCS1 and TT0CCS0 bits when the TT0CE bit = 0.  (The same value can be written when the TT0CE bit = 1.)  If rewriting was mistakenly performed, clear the TT0CE bit to 0 and then set these bits again.**

**2.  Be sure to set bits 1 to 3, 6, and 7 to "0".**

**(9) TMT0 option register 1 (TT0OPT1)**

The TT0OPT1 register is an 8-bit register that detects overflows, underflows, and count-up/down operations of the encoder count function.

The TT0OPT1 register is valid only in the encoder compare mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

This register can be rewritten even when the TT0CTL0.TT0CE bit = 1.

(1/2)

After reset: 00H    R/W    Address: FFFFF608H

| | 7 | 6 | 5 | 4 | 3 | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| TT0OPT1 | 0 | 0 | 0 | 0 | 0 | TT0EUF | TT0EOF | TT0ESF |

| TT0EUF | TMT0 underflow detection flag |
|---|---|
| Set (1) | Underflow occurs. |
| Reset (0) | Cleared by writing to TT0EUF bit or when TT0CTL0.TT0CE bit = 0 |

- The TT0EUF bit is set to 1 when the 16-bit counter underflows from 0000H to FFFFH in the encoder compare mode.

- When the TT0CTL2.TT0LDE bit = 1, the TT0EUF bit is set to 1 when the value of the 16-bit counter is changed from 0000H to the set value of the TT0CCR0 register.

- An overflow interrupt request signal (INTTTOV0) is generated as soon as the TT0EUF bit is set to 1.

- The TT0EUF bit is not cleared to 0 even if the TT0EUF bit or TT0OPT1 register is read when the TT0EUF bit = 1.

- The status of the TT0EUF bit is retained even if the TT0CTL0.TT0CE bit is cleared to 0 when the TT0CTL2.TT0ECC bit = 1.

- Before clearing the TT0EUF bit to 0 after the INTTTOV0 signal is generated, be sure to confirm (read) that the TT0EUF bit is set to 1.

- The TT0EUF bit can be read or written, but it cannot be set to 1 by software. Setting this bit to 1 does not affect the operation of TMT0.

| TT0EOF | Overflow detection flag for TMT0 encoder function |
|---|---|
| Set (1) | Overflow occurs. |
| Reset (0) | Cleared by writing 0 to the TT0EOF bit or when the TT0CTL0.TT0CE bit = 0 |

- The TT0EOF bit is set to 1 when the 16-bit counter overflows from FFFFH to 0000H in the encoder compare mode.
- As soon as the TT0EOF bit has been set to 1, an overflow interrupt request signal (INTTTOV0) is generated.  At this time, the TT0OPT0.TT0OVF bit is not set to 1.
- The TT0EOF bit is not cleared to 0 even if the TT0EOF bit or TT0OPT1 register is read when the TT0EOF bit = 1.
- The status of the TT0EOF bit is retained even if the TT0CTL0.TT0CE bit is cleared to 0 when the TT0CTL2.TT0ECC bit = 1.
- Before clearing the TT0EOF bit to 0 after the INTTTOV0 signal is generated, be sure to confirm (read) that the TT0EOF bit is set to 1.
- The TT0EOF bit can be read or written, but it cannot be set to 1 by software. Writing 1 to this bit does not affect the operation of TMT0.

| TT0ESF | TMT0 count-up/-down operation status detection flag |
|---|---|
| 0 | TMT0 is counting up. |
| 1 | TMT0 is counting down. |

- This bit is cleared to 0 if  the TT0CTL0.TT0CE bit = 0 when the TT0CTL2.TT0ECC bit = 0.
- The status of the TT0ESF bit is retained even if the TT0CE bit = 0 when the TT0ECC bit = 1.

**Caution    Be sure to set bits 3 to 7 to "0".**

**(10) TMT0 capture/compare register 0 (TT0CCR0)**

The TT0CCR0 register is a 16-bit register that can be used as a capture register or compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, depending on the setting of the TT0OPT0.TT0CCS0 bit. In the pulse width measurement mode, the TT0CCR0 register can be used only as a capture register. In any other mode, this register can be used only as a compare register.

The TT0CCR0 register can be read or written during operation.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

| After reset: 0000H | R/W | Address: FFFFF60AH |
| --- | --- | --- |

|  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TT0CCR0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**(a) Function as compare register**

The TT0CCR0 register can be rewritten even when the TT0CTL0.TT0CE bit = 1.

The set value of the TT0CCR0 register is transferred to the CCR0 buffer register.  When the value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTT0CC0) is generated.  If TOT00 pin output is enabled at this time, the output of the TOT00 pin is inverted.

When the TT0CCR0 register is used as a cycle register in the interval timer mode, or when the TT0CCR0 register is used as a cycle register in the external event count mode, external trigger pulse output mode, one-shot pulse output mode, PWM output mode, triangular-wave PWM output mode, or encoder compare mode, the value of the 16-bit counter is cleared (0000H) if its count value matches the value of the CCR0 buffer register.

The compare register is not cleared by setting the TT0CTL0.TT0CE bit to 0.

**(b) Function as capture register**

In the free-running timer mode (when the TT0CCR0 register is used as a capture register), the count value of the 16-bit counter is stored in the TT0CCR0 register if the valid edge of the capture trigger input pin (TIT00 pin) is detected.  In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TT0CCR0 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIT00 pin) is detected.

Even if the capture operation and reading the TT0CCR0 register conflict, the correct value of the TT0CCR0 register can be read.

The capture register is cleared by setting the TT0CTL0.TT0CE bit to 0.

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

**Table 9-3.  Function of Capture/Compare Register in Each Mode and How to Write Compare Register**

| Operation Mode | TT0CCR0 Register | How to Write Compare Register |
|---|---|---|
| Interval timer | Compare register | Anytime write |
| External event counter | Compare register | Anytime write |
| External trigger pulse output | Compare register | Batch write[Note] |
| One-shot pulse output | Compare register | Anytime write |
| PWM output | Compare register | Batch write[Note] |
| Free-running timer | Capture/compare register | Anytime write |
| Pulse width measurement | Capture register | None |
| Triangular-wave WPM output | Compare register | Batch write[Note] |
| Encoder compare | Compare register | Anytime write |

**Note**   Writing to the TT0CCR1 register is the trigger.

**Remark**   For anytime write and batch write, see **9.6 (2)  Anytime write and batch write**.

**(11)  TMT0 capture/compare register 1 (TT0CCR1)**

The TT0CCR1 register is a 16-bit register that can be used as a capture register or compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, depending on the setting of the TT0OPT0.TT0CCS1 bit.  In the pulse width measurement mode, the TT0CCR1 register can be used only as a capture register.  In any other mode, this register can be used only as a compare register.

The TT0CCR1 register can be read or written during operation.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

After reset: 0000H    R/W    Address: FFFFFF60CH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| TT0CCR1 | | | | | | | | | | | | | | | | |

**(a) Function as compare register**

The TT0CCR1 register can be rewritten even when the TT0CTL0.TT0CE bit = 1.

The set value of the TT0CCR1 register is transferred to the CCR1 buffer register. When the value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTT0CC01) is generated. If TOT01 pin output is enabled at this time, the output of the TOT01 pin is inverted. The compare register is not cleared by setting the TT0CTL0.TT0CE bit to 0.

**(b) Function as capture register**

In the free-running timer mode (when the TT0CCR1 register is used as a capture register), the count value of the 16-bit counter is stored in the TT0CCR1 register if the valid edge of the capture trigger input pin (TIT01 pin) is detected. In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TT0CCR1 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIT01 pin) is detected.

Even if the capture operation and reading the TT0CCR1 register conflict, the correct value of the TT0CCR1 register can be read.

The capture register is cleared by setting the TT0CTL0.TT0CE bit to 0.

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

**Table 9-4.  Function of Capture/Compare Register in Each Mode and How to Write Compare Register**

| Operation Mode | TT0CCR1 Register | How to Write Compare Register |
|---|---|---|
| Interval timer | Compare register | Anytime write |
| External event counter | Compare register | Anytime write |
| External trigger pulse output | Compare register | Batch write[Note] |
| One-shot pulse output | Compare register | Anytime write |
| PWM output | Compare register | Batch write[Note] |
| Free-running timer | Capture/compare register | Anytime write |
| Pulse width measurement | Capture register | None |
| Triangular-wave PWM output | Compare register | Batch write[Note] |
| Encoder compare | Compare register | Anytime write |

**Note**   Writing to the TT0CCR1 register is the trigger.

**Remark**   For anytime write and batch write, see **9.6 (2)  Anytime write and batch write**.

**(12) TMT0 counter write register (TT0TCW)**

The TT0TCW register is used to set the initial value of the 16-bit counter.

The TT0TCW register is valid only in the encoder compare mode.

This register can be read or written in 16-bit units.

Rewrite the TT0TCW register when the TT0CTL0.TT0CE bit = 0.

The value of the TT0TCW register is transferred to the 16-bit counter when the TT0CE bit is set (1).

Reset sets this register to 0000H.

After reset: 0000H    R/W    Address: FFFFF610H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TT0TCW | | | | | | | | | | | | | | | | |
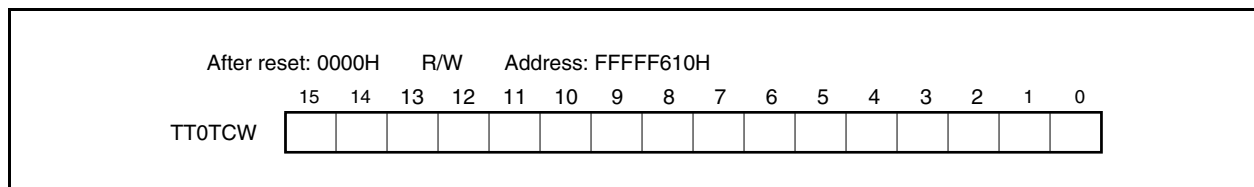
**(13) TMT0 counter read buffer register (TT0CNT)**

The TT0CNT register is a read buffer register that can read the count value of the 16-bit counter.

If this register is read when the TT0CTL0.TT0CE bit = 1, the count value of the 16-bit timer can be read.

This register is read-only, in 16-bit units.

The value of the TT0CNT register is set to 0000H when the TT0CTL2.TT0ECC and TT0CE bits = 0.  If the TT0CNT register is read at this time, the value of the 16-bit counter (FFFFH) is not read, but 0000H is read.  The TT0CNT register is not set to 0000H but the previous value is read when the TT0ECC bit = 1 and TT0CE bit = 0.

The TT0ECC and TT0CE bits are set to 0 after reset, and the value of the TT0CNT register is set to 0000H.

After reset: 0000H    R    Address: FFFFFF60EH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TT0CNT | | | | | | | | | | | | | | | | |
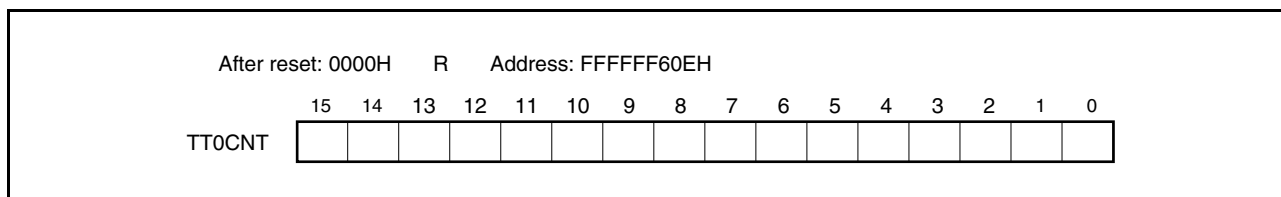
**(14) Noise elimination control register (TTNFC)**

Digital noise elimination can be selected for the TIT00, TIT01, TENC01, TECR0, and EVTT00 pins.  The noise elimination settings are performed using the TTNFC register.

When digital noise elimination is selected, the sampling clock for digital sampling can be selected from among $f_{XX}$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, and $f_{XX}/64$.  Sampling is performed 3 times.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**Caution   Time equal to the sampling clock × 3 clocks is required until the digital noise eliminator is initialized after the sampling clock has been changed.  If the valid edge of the TIT00, TIT01, TENC01, TECR0, and EVTT pins is input after the sampling clock has been changed and before the time of the sampling clock × 3 clocks passes, therefore, an interrupt request signal may be generated.  Therefore, when using the external trigger function, the external event function, the capture trigger function, and the encoder function of TMT, enable TMT operation after the sampling clock × 3 clocks have elapsed.**

After reset:  00H    R/W    Address:  FFFFF726H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TTNFC | TTNFEN | 0 | 0 | 0 | 0 | TTNFC2 | TTNFC1 | TTNFC0 |

| TTNFEN | Settings of digital noise elimination |
|---|---|
| 0 | Digital noise elimination not executed |
| 1 | Digital noise elimination executed |

| TTNFC2 | TTNFC1 | TTNFC0 | Digital sampling clock |
|---|---|---|---|
| 0 | 0 | 0 | $f_{XX}$ |
| 0 | 0 | 1 | $f_{XX}/4$ |
| 0 | 1 | 0 | $f_{XX}/8$ |
| 0 | 1 | 1 | $f_{XX}/16$ |
| 1 | 0 | 0 | $f_{XX}/32$ |
| 1 | 0 | 1 | $f_{XX}/64$ |
| Other than above | | | Setting prohibited |

**Remarks 1.** Since sampling is performed three times, the noise width for reliably eliminating noise is 2 sampling clocks.

**2.** In the case of noise with a width smaller than 2 sampling clocks, an interrupt request signal is generated if noise synchronized with the sampling clock is input.

A timing example of noise elimination performed by the timer T input pin digital filter is shown Figure 9-2.

**Figure 9-2.  Example of Digital Noise Elimination Timing**



**Remark**    If there are two or fewer noise elimination clocks while the TIT00, TIT01, TENC01, TECR0, and EVTT00 input signals are high level (or low level), the input signal is eliminated as noise.  If it is sampled three times or more, the edge is detected as a valid input.
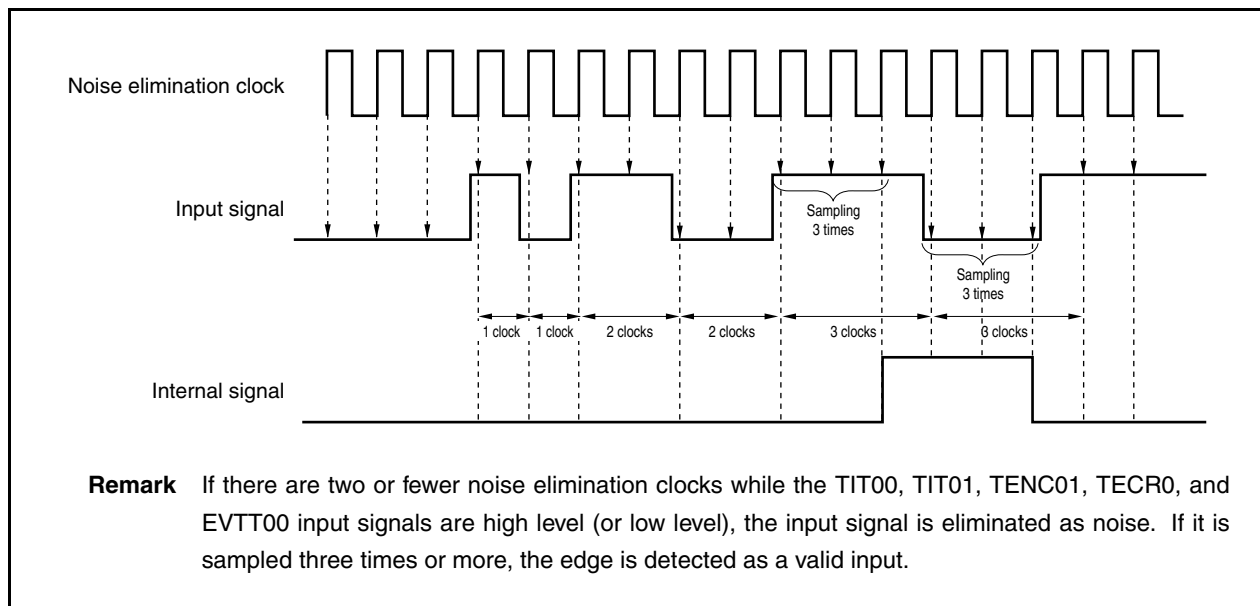
## 9.5  Timer Output Operations

The following table shows the operations and output levels of the TOT00 and TOT01 pins.

**Table 9-5.  Timer Output Control in Each Mode**

| Operation Mode | TOT01 Pin | TOT00 Pin |
|---|---|---|
| Interval timer mode | Square wave output | |
| External event count mode | None | |
| External trigger pulse output mode | External trigger pulse output | Square wave output |
| One-shot pulse output mode | One-shot pulse output | |
| PWM output mode | PWM output | |
| Free-running timer mode | Square wave output (only when compare function is used) | |
| Pulse width measurement mode | None | |
| Triangular-wave PWM output mode | Triangular-wave PWM output | |
| Encoder compare mode | None | |

**Table 9-6.  Truth Table of TOT00 and TOT01 Pins Under Control of Timer Output Control Bits**

| TT0IOC0.TT0OLn Bit | TT0IOC0.TT0OEn Bit | TT0CTL0.TT0CE Bit | Level of TOT0n Pin |
|---|---|---|---|
| 0 | 0 | × | Low-level output |
| | 1 | 0 | Low-level output |
| | | 1 | Low level immediately before counting, high level after counting is started |
| 1 | 0 | × | High-level output |
| | 1 | 0 | High-level output |
| | | 1 | High level immediately before counting, low level after counting is started |

**Remark**    n = 0, 1

## 9.6  Operation

The functions of TMT0 that can be implemented differ from one channel to another.  The functions of each channel are shown below.

**Table 9-7.  TMT0 Specifications in Each Mode**

| Operation | TT0CTL1.TT0EST Bit (Software Trigger Bit) | EVTT0 Pin (External Trigger Input) | Capture/Compare Register Setting | Compare Register Write Method |
|---|---|---|---|---|
| Interval timer mode | Invalid | Invalid | Compare only | Anytime write |
| External event count mode | Invalid | Invalid | Compare only | Anytime write |
| External trigger pulse output mode | Valid | Valid | Compare only | Batch write |
| One-shot pulse output mode | Valid | Valid | Compare only | Anytime write |
| PWM output mode | Invalid | Invalid | Compare only | Batch write |
| Free-running timer mode | Invalid | Invalid | Switchable | Anytime write |
| Pulse width measurement mode | Invalid | Invalid | Capture only | Not applicable |
| Triangular-wave PWM output mode | Invalid | Invalid | Compare only | Batch write |
| Encoder compare mode | Invalid | Invalid | Compare only | Anytime write |

**(1) Basic counter operation**

This section explains the basic operation of the 16-bit counter.  For details, refer to the description of the operation in each mode.

**(a) Count start operation**

- **Encoder compare mode**

  A count operation is controlled by TENC00 and TENC01 phases.

  The 16-bit counter initial setting is performed by transferring the set value of the TT0TCW register to the 16-bit counter and the count operation is started. (When the TT0CTL2.TT0ECC bit = 0, the TT0TCW register set value is transferred to the 16-bit counter at the timing when the TT0CTL0.TT0CE bit changes from 0 to 1.)

- **Triangular-wave PWM mode**

  The 16-bit counter starts counting from the initial value FFFFH.

  It counts up FFFFH, 0000H, 0001H, 0002H, 0003H, and so on.

  Following the count-up operation, the counter counts down upon a match between the 16-bit count value and the CCR0 buffer register.

- **Mode other than above**

  The 16-bit counter starts counting from the initial value FFFFH.

  It counts up FFFFH, 0000H, 0001H, 0002H, 0003H, and so on.

**(b) Clear operation**

The 16-bit counter is cleared to 0000H when its value matches the value of the compare register, when its value is captured, when the edge of the encoder clear signal is detected, and when the clear level condition of the TENC00, TENC01, and TECR0 pins is detected.  The count operation from FFFFH to 0000H that takes place immediately after the counter has started counting or when the counter overflows is not a clear operation. Therefore, the INTTT0CC0 and INTTT0CC1 interrupt signals are not generated.

### (c) Overflow operation

The 16-bit counter overflows when it counts up from FFFFH to 0000H in the free-running mode, pulse width measurement mode, and encoder compare mode. If the counter overflows in the free-running mode and pulse width measurement mode, the TT0OPT0.TT0OVF bit is set to 1 and an interrupt request signal (INTTT0OV) is generated.

If the counter overflows in the encoder compare mode, the TT0OPT1.TT0EOF bit is set to 1 and an interrupt request signal (INTTT0OV) is generated.

Note that the INTTT0OV signal is not generated under the following conditions.

- Immediately after a count operation has been started
- If the counter value matches the compare value FFFFH and is cleared
- When FFFFH is captured and cleared to 0000H in the pulse width measurement mode

**Caution After the overflow interrupt request signal (INTTT0OV) has been generated, be sure to check that the overflow flag (TT0OVF, TT0EOF bits) is set to 1.**

### (d) Count value hold operation

The value of the 16-bit counter is held by the TT0CTL2.TT0ECC bit in the encoder compare mode. The value of the 16-bit counter is reset to FFFFH when the TT0ECC bit = 0 and TT0CTL0.TT0CE bit = 0. When the TT0CE bit is next set to 1, the set value of the TT0TCW register is transferred to the 16-bit counter and a count operation is performed.

If the TT0ECC bit = 1 and TT0CE bit = 0, the value of the 16-bit counter is held. When the TT0CE bit is next set to 1, the counter resumes the count operation from the held value.

### (e) Counter read operation during count operation

The value of the 16-bit counter of TMT0 can be read by using the TT0CNT register during the count operation. When the TT0CTL0.TT0CE bit = 1, the value of the 16-bit counter can be read by reading the TT0CNT register. If the TT0CNT register is read when the TT0CTL2.TT0ECC bit = 0 and TT0CE bit = 0, however, it is 0000H. The held value of the TT0CNT register is read if the register is read when the TT0ECC bit = 1 and TT0CE bit = 0.

### (f) Underflow operation

A 16-bit counter underflow occurs at the timing when the 16-bit counter value changes from 0000H to FFFFH in the encoder compare mode. When an underflow occurs, the TT0OPT1.TT0EUF bit is set to 1 and an interrupt request signal (INTTT0OV) is generated.

**(g) Interrupt operation**

TMT0 generates the following four types of interrupt request signals.

- INTTT0CC0 interrupt:   This signal functions as a match interrupt request signal of the CCR0 buffer register and as a capture interrupt request signal to the TT0CCR0 register.
- INTTT0CC1 interrupt:   This signal functions as a match interrupt request signal of the CCR1 buffer register and as a capture interrupt request signal to the TT0CCR1 register.
- INTTT0OV interrupt:   This signal functions as an overflow interrupt request signal.
- INTTT0EC interrupt:   This signal functions as a valid edge detection interrupt request signal of the encoder clear input (TECR0 pin).

**(2) Anytime write and batch write**

The TT0CCR0 and TT0CCR1 registers in TMT0 can be rewritten during timer operation (TT0CTL0.TT0CE bit = 1), but the write method (anytime write, batch write) of the CCR0 and CCR1 buffer registers differs depending on the mode.

**(a) Anytime write**

In this mode, data is transferred at any time from the TT0CCR0 and TT0CCR1 registers to the CCR0 and CCR1 buffer registers during timer operation (n = 0, 1).

**Figure 9-3. Flowchart of Basic Operation for Anytime Write**



**Note**   The 16-bit counter is not cleared upon a match between the 16-bit counter value and the CCR1 buffer register value. It is cleared upon a match between the 16-bit counter value and the CCR0 buffer register value.

**Remarks 1.** The above flowchart illustrates an example of the operation in the interval timer mode.
        **2.** n = 0, 1
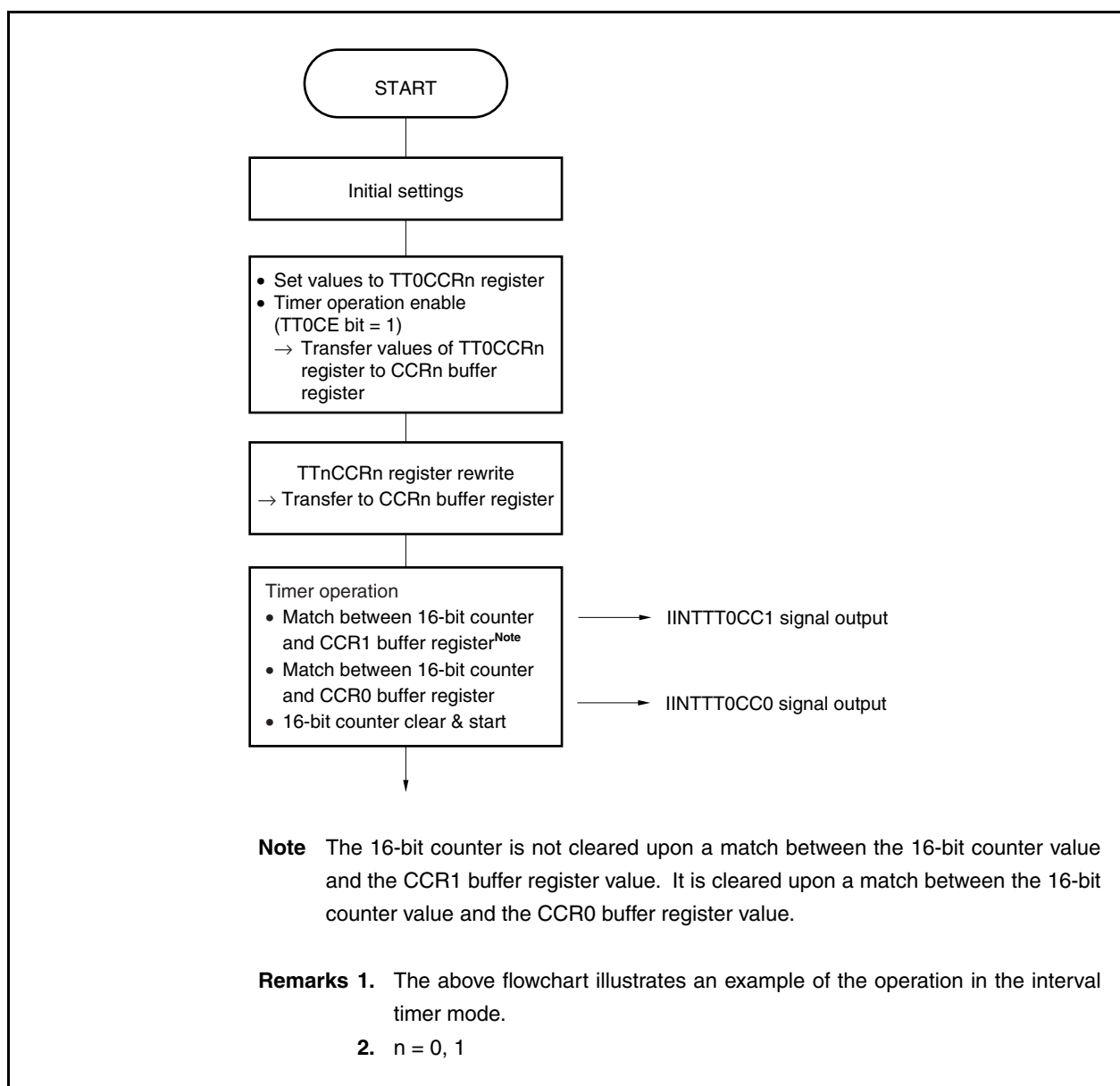
**Figure 9-4. Timing of Anytime Write**



Remarks 1. $D_{01}$, $D_{02}$: Set values of TT0CCR0 register

$D_{11}$, $D_{12}$: Set values of TT0CCR1 register

2. The above timing chart illustrates an example of the operation in the interval timer mode.

3. n = 0, 1
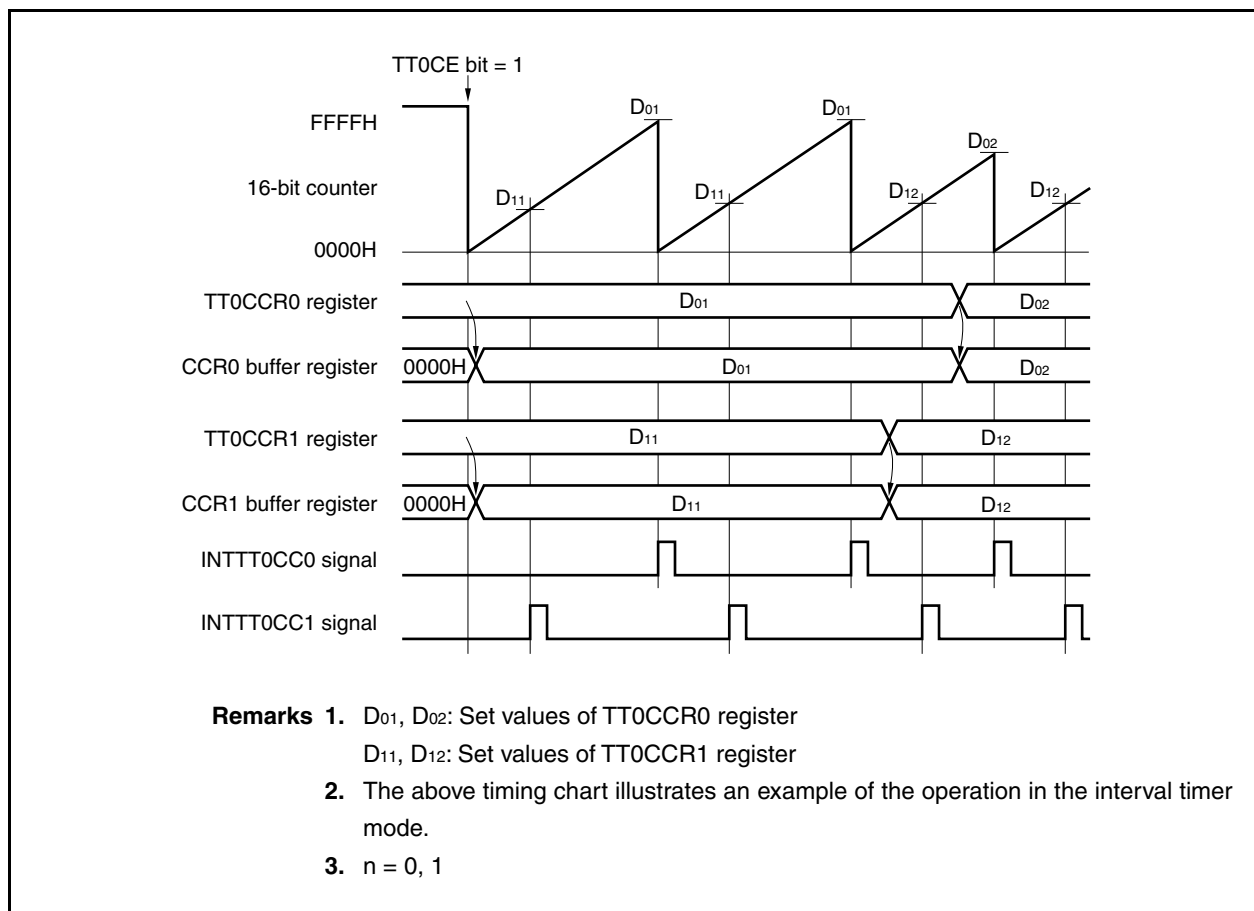
**(b)  Batch write**

In this mode, data is transferred all at once from the TT0CCR0 and TT0CCR1 registers to the CCR0 and CCR1 buffer registers during timer operation.  This data is transferred upon a match between the value of the CCR0 buffer register and the value of the 16-bit counter.  Transfer is enabled by writing to the TT0CCR1 register. Whether to enable or disable the next transfer timing is controlled by writing or not writing to the TT0CCR1 register.

In order for the set value when the TT0CCR0 and TT0CCR1 registers are rewritten to become the 16-bit counter comparison value (in other words, in order for this value to be transferred to the CCR0 and CCR1 buffer registers), it is necessary to rewrite the TT0CCR0 register and then write to the TT0CCR1 register before the 16-bit counter value and the CCR0 buffer register value match.  Therefore, the values of the TT0CCR0 and TT0CCR1 registers are transferred to the CCR0 and CCR1 buffer registers upon a match between the count value of the 16-bit counter and the value of the CCR0 buffer register.  Thus even when wishing only to rewrite the value of the TT0CCR0 register, also write the same value (same as preset value of the TT0CCR1 register) to the TT0CCR1 register.

**Figure 9-5.  Flowchart of Basic Operation for Batch Write**



**Note** The 16-bit counter is not cleared upon a match between the 16-bit counter value and the CCR1 buffer register value.  It is cleared upon a match between the 16-bit counter value and the CCR0 buffer register value.

**Caution** **Writing to the TT0CCR1 register includes enabling of batch write.  Thus, rewrite the TT0CCR1 register after rewriting the TT0CCR0 register.**

**Remark** The above flowchart illustrates an example of the operation in the PWM output mode.

**Figure 9-6.  Timing of Batch Write**
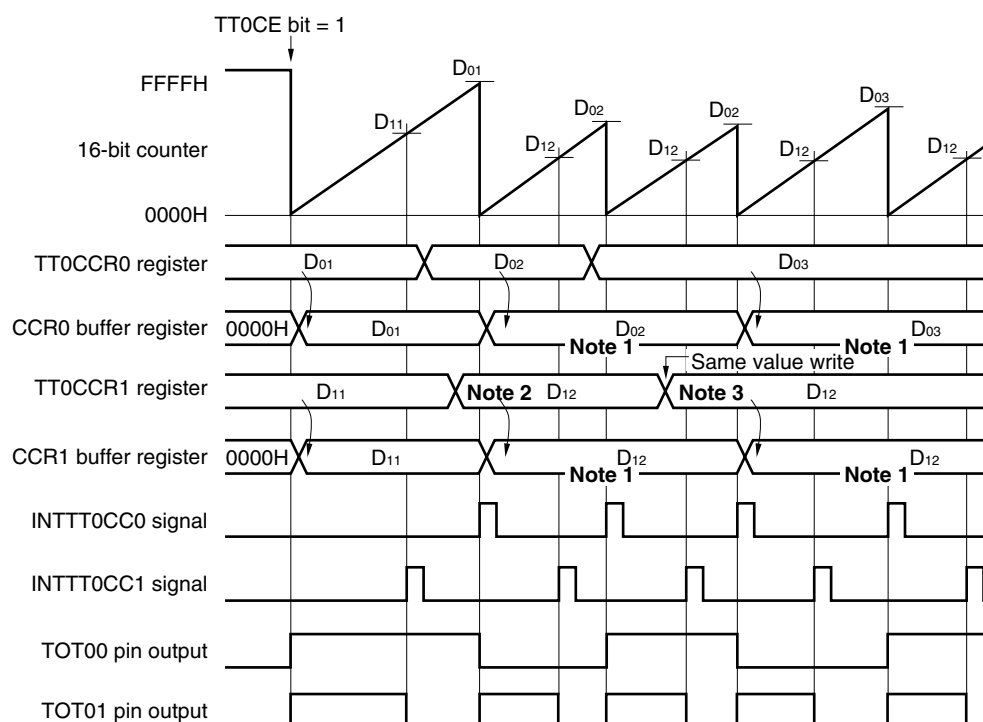


Notes 1.  Because the TT0CCR1 register was not rewritten, $D_{03}$ is not transferred.

2.  Because the TT0CCR1 register has been written ($D_{12}$), data is transferred to the CCR1 buffer register upon a match between the value of the 16-bit counter and the value of the TT0CCR0 register ($D_{01}$).

3.  Because the TT0CCR1 register has been written ($D_{12}$), data is transferred to the CCR1 buffer register upon a match between the value of the 16-bit counter and the value of the TT0CCR0 register ($D_{02}$).

Remarks 1.  $D_{01}$, $D_{02}$, $D_{03}$:   Set values of TT0CCR0 register

$D_{11}$, $D_{12}$:      Set values of TT0CCR1 register

2.  The above timing chart illustrates the operation in the PWM output mode as an example.

### 9.6.1  Interval timer mode (TT0MD3 to TT0MD0 bits = 0000)

In the interval timer mode, an interrupt request signal (INTTT0CC0) is generated at the interval set by the TT0CCR0 register if the TT0CTL0.TT0CE bit is set to 1.  A square wave whose half cycle is equal to the interval can be output from the TOT00 pin.

The TT0CCR1 register is not used in the interval timer mode.  However, the set value of the TT0CCR1 register is transferred to the CCR1 buffer register, and when the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTT0CC1) is generated.  In addition, a square wave, which is inverted when the INTTT0CC1 signal is generated, can be output from the TOT01 pin.

The value of the TT0CCR0 and TT0CCR1 registers can be rewritten even while the timer is operating.

**Figure 9-7.  Configuration of Interval Timer**
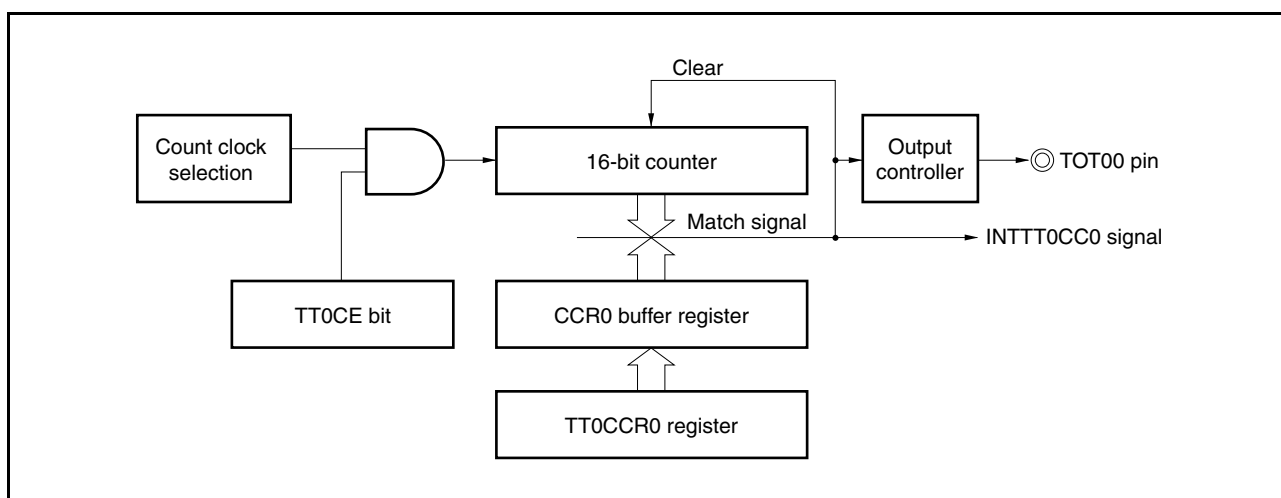


**Figure 9-8.  Basic Timing of Operation in Interval Timer Mode**

When the TT0CE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H in synchronization with the count clock, and the counter starts counting.  At this time, the output of the TOT00 pin is inverted.  Additionally, the set value of the TT0CCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, the output of the TOT00 pin is inverted, and a compare match interrupt request signal (INTTT0CC0) is generated.

The interval can be calculated by the following expression.

Interval = (Set value of TT0CCR0 register + 1) $\times$ Count clock cycle

**Figure 9-9.  Register Setting for Interval Timer Mode Operation (1/2)**
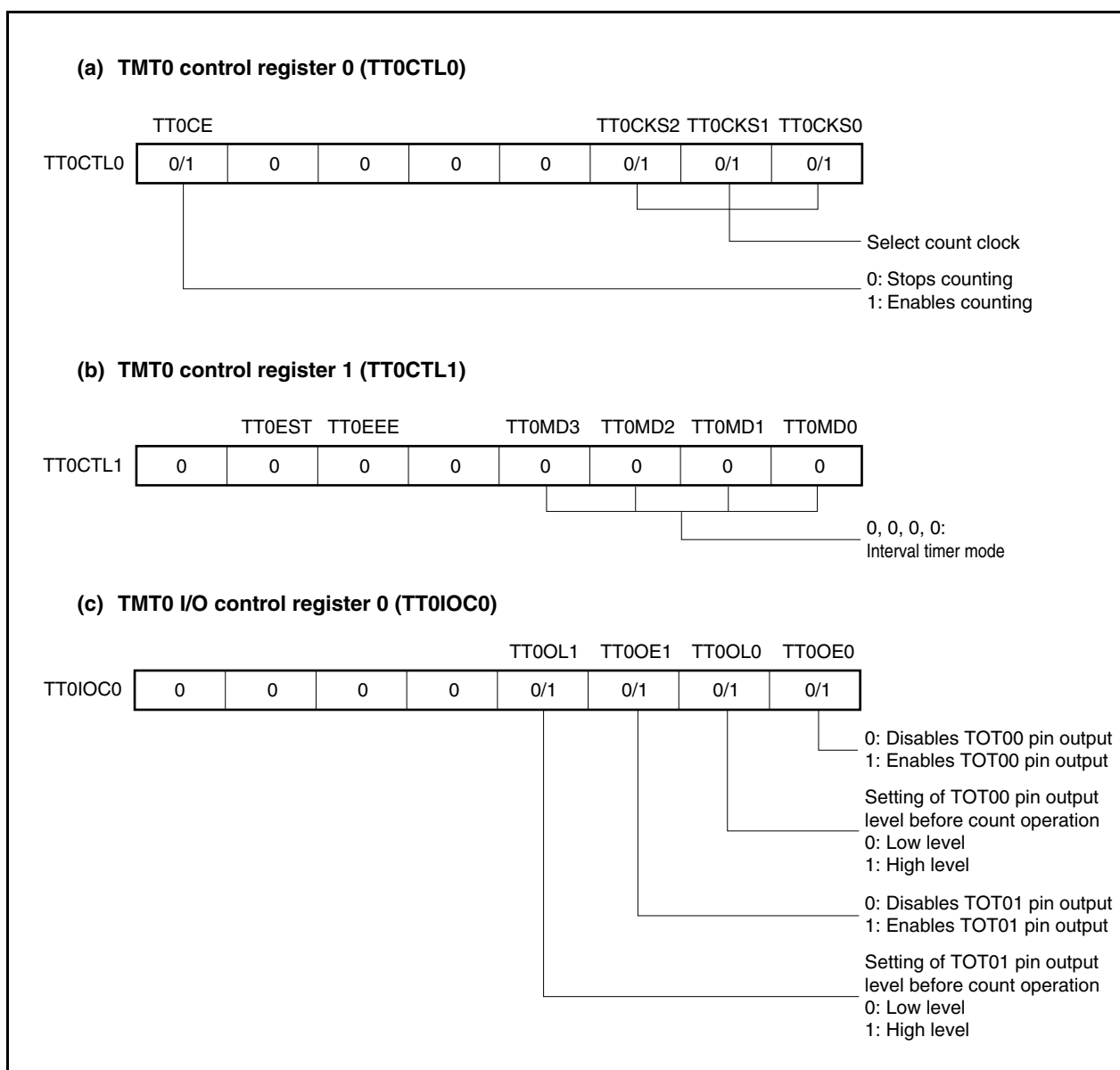
**Figure 9-9. Register Setting for Interval Timer Mode Operation (2/2)**

---

**(d) TMT0 counter read buffer register (TT0CNT)**

By reading the TT0CNT register, the count value of the 16-bit counter can be read.

**(e) TMT0 capture/compare register 0 (TT0CCR0)**

If the TT0CCR0 register is set to $D_0$, the interval is as follows.

Interval = $(D_0 + 1) \times$ Count clock cycle

**(f) TMT0 capture/compare register 1 (TT0CCR1)**

The TT0CCR1 register is not used in the interval timer mode. However, the set value of the TT0CCR1 register is transferred to the CCR1 buffer register. When the count value of the 16-bit counter matches the value of the CCR1 buffer register, the TOT01 pin output is inverted and a compare match interrupt request signal (INTTT0CC1) is generated.

By setting this register to the same value as the value set in the TT0CCR0 register, a square wave can be output from the TOT01 pin.

When the TT0CCR1 register is not used, it is recommended to set its value to FFFFH. Also mask the register by the interrupt mask flag (TT0CCIC1.TT0CCMK1).

**Remark** TMT0 control register 2 (TT0CTL2), TMT0 I/O control register 1 (TT0IOC1), TMT0 I/O control register 2 (TT0IOC2), TMT0 I/O control register 3 (TT0IOC3), TMT0 option register 0 (TT0OPT0), TMT0 option register 1 (TT0OPT1), and TMT0 counter write register (TT0TCW) are not used in the interval timer mode.

---

**(1) Interval timer mode operation flow**

**Figure 9-10.  Software Processing Flow in Interval Timer Mode**
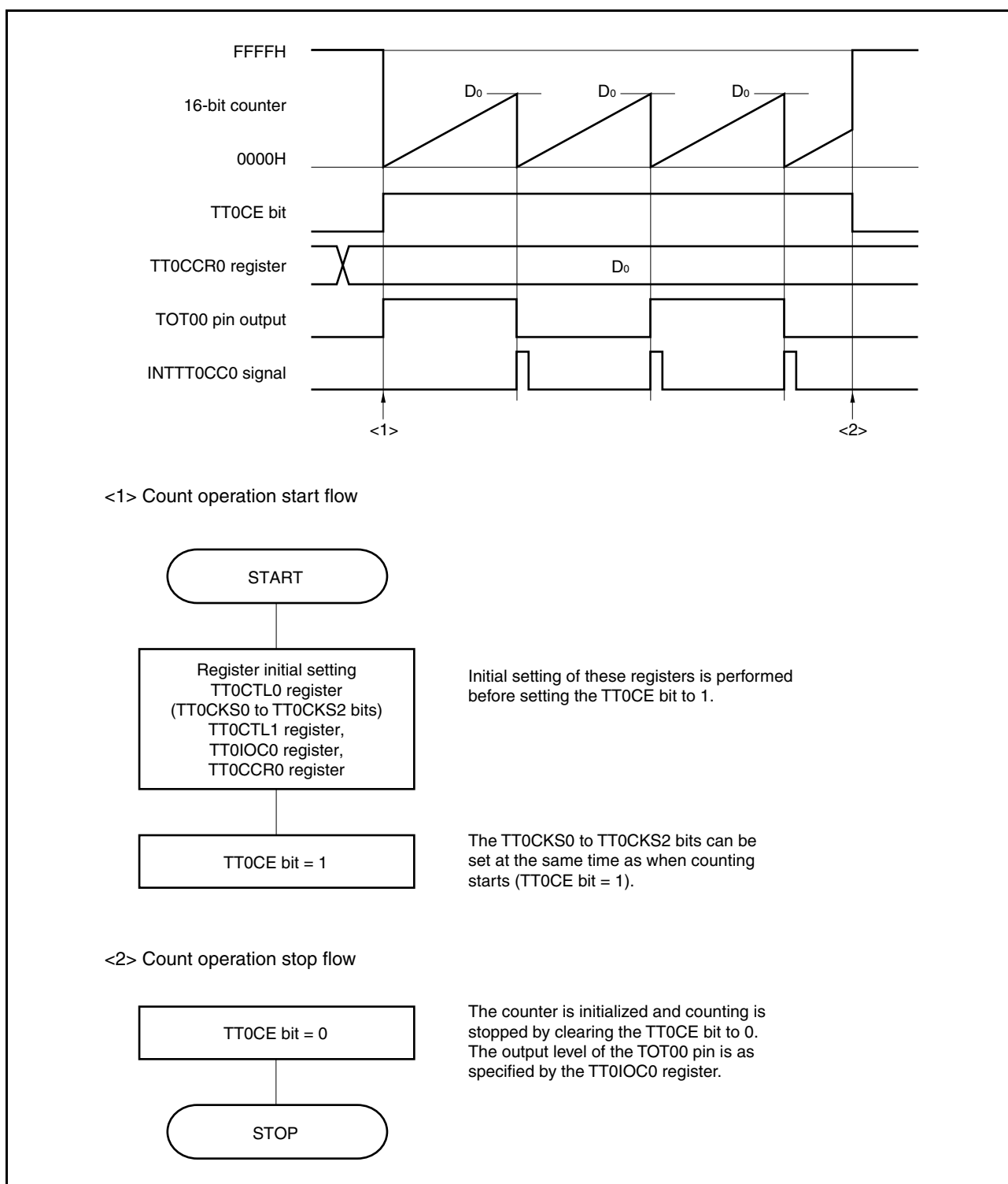
**(2) Interval timer mode operation timing**

**(a) Operation if TT0CCR0 register is set to 0000H**

If the TT0CCR0 register is set to 0000H, the INTTT0CC0 signal is generated at each count clock, and the output of the TOT00 pin is inverted.

The value of the 16-bit counter is always 0000H.
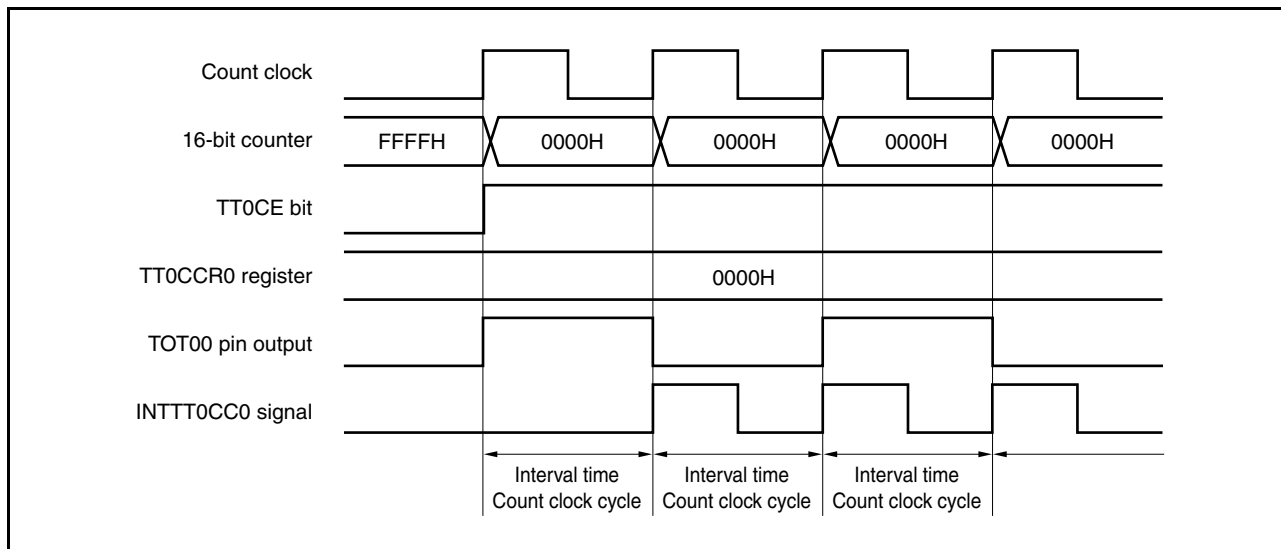
**(b) Operation if TT0CCR0 register is set to FFFFH**

If the TT0CCR0 register is set to FFFFH, the 16-bit counter counts up to FFFFH.  The counter is cleared to 0000H in synchronization with the next count-up timing.  The INTTT0CC0 signal is generated and the output of the TOT00 pin is inverted.  At this time, an overflow interrupt request signal (INTTT0OV) is not generated, nor is the overflow flag (TT0OPT0.TT0OVF bit) set to 1.
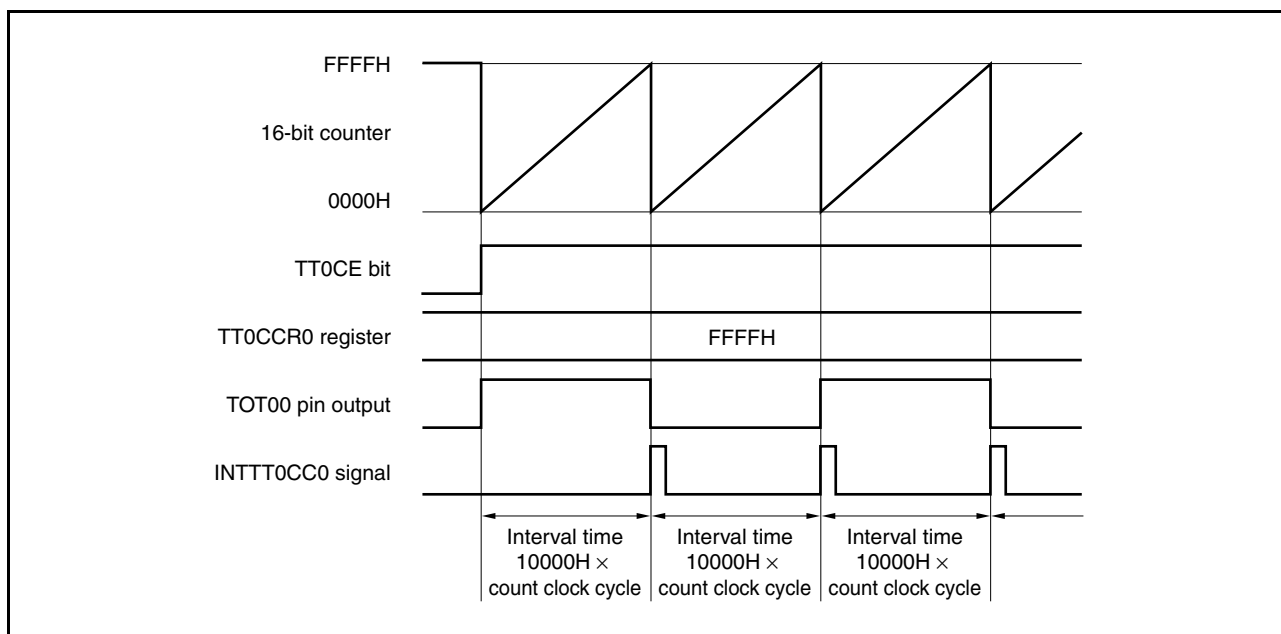
**(c) Notes on rewriting TT0CCR0 register**

If the value of the TT0CCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow. When an overflow may occur, stop counting and then change the set value.
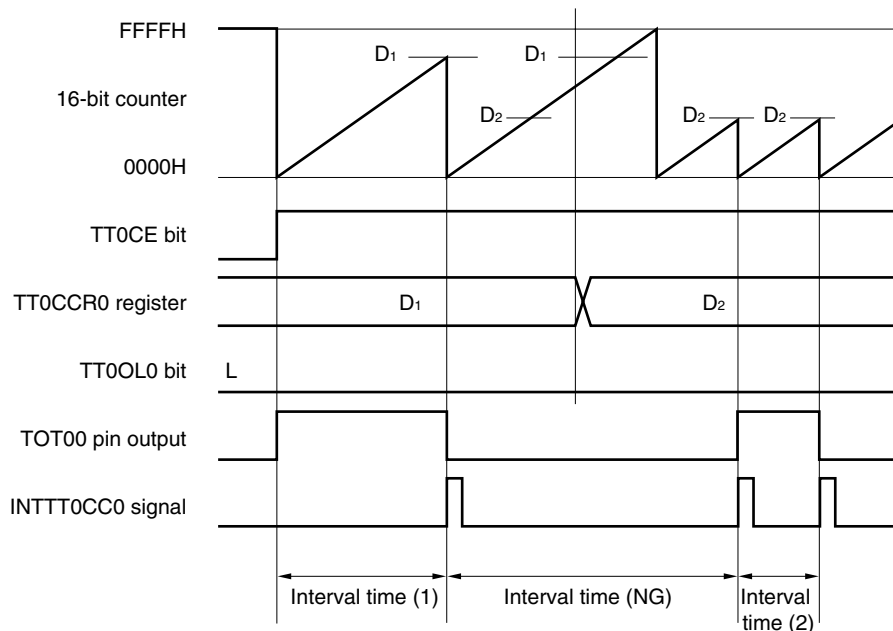


**Remark**    Interval time (1):     ($D_1$ + 1) × Count clock cycle

                 Interval time (NG):  (10000H + $D_2$ + 1) × Count clock cycle

                 Interval time (2):     ($D_2$ + 1) × Count clock cycle

If the value of the TT0CCR0 register is changed from $D_1$ to $D_2$ while the count value is greater than $D_2$ but less than $D_1$, the count value is transferred to the CCR0 buffer register as soon as the TT0CCR0 register has been rewritten. Consequently, the value of the 16-bit counter that is compared is $D_2$.

Because the count value has already exceeded $D_2$, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H. When the count value matches $D_2$, the INTTT0CC0 signal is generated and the output of the TOT00 pin is inverted.

Therefore, the INTTT0CC0 signal may not be generated at the interval time "($D_1$ + 1) × Count clock cycle" or "($D_2$ + 1) × Count clock cycle" as originally expected, but may be generated at an interval of "(10000H + $D_2$ + 1) × Count clock cycle".

**(d) Operation of TT0CCR1 register**

**Figure 9-11. Configuration of TT0CCR1 Register**
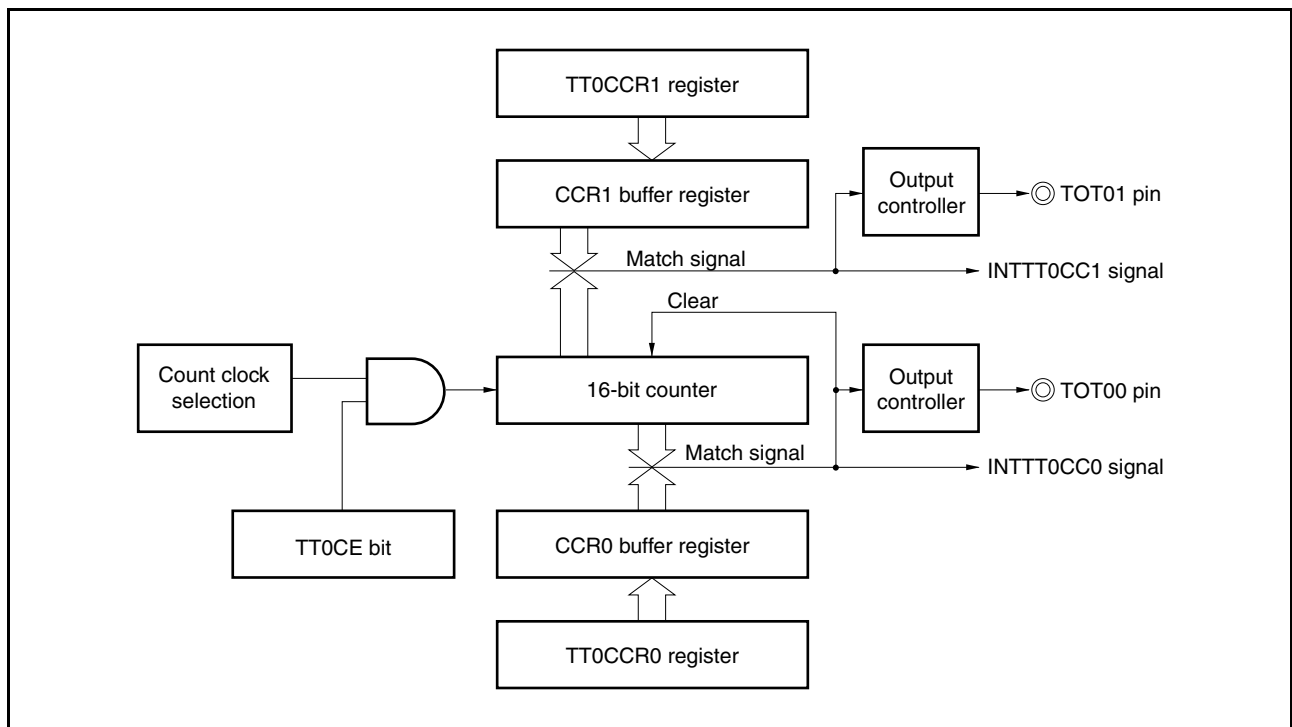
When the TT0CCR1 register is set to the same value as the TT0CCR0 register, the INTTT0CC0 signal is generated at the same timing as the INTTT0CC1 signal and the TOT01 pin output is inverted. In other words, a square wave can be output from the TOT01 pin.

The following shows the operation when the TT0CCR1 register is set to other than the value set in the TT0CCR0 register.

If the set value of the TT0CCR1 register is less than the set value of the TT0CCR0 register, the INTTT0CC1 signal is generated once per cycle. At the same time, the output of the TOT01 pin is inverted.

The TOT01 pin outputs a square wave after outputting a short-width pulse.

**Figure 9-12.  Timing Chart When $D_{01} \geq D_{11}$**
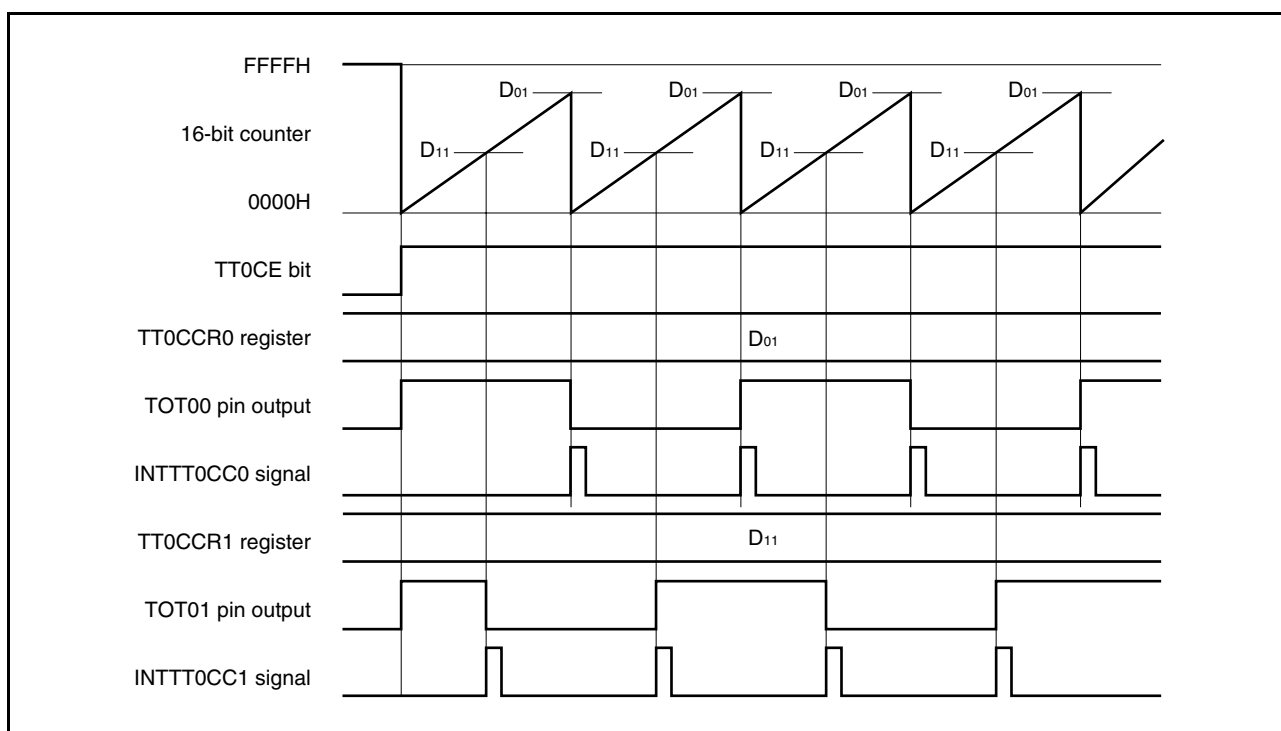
If the set value of the TT0CCR1 register is greater than the set value of the TT0CCR0 register, the count value of the 16-bit counter does not match the value of the TT0CCR1 register.  Consequently, the INTTT0CC1 signal is not generated, nor is the output of the TOT01 pin changed.

When the TT0CCR1 register is not used, it is recommended to set its value to FFFFH.

**Figure 9-13.  Timing Chart When $D_{01} < D_{11}$**
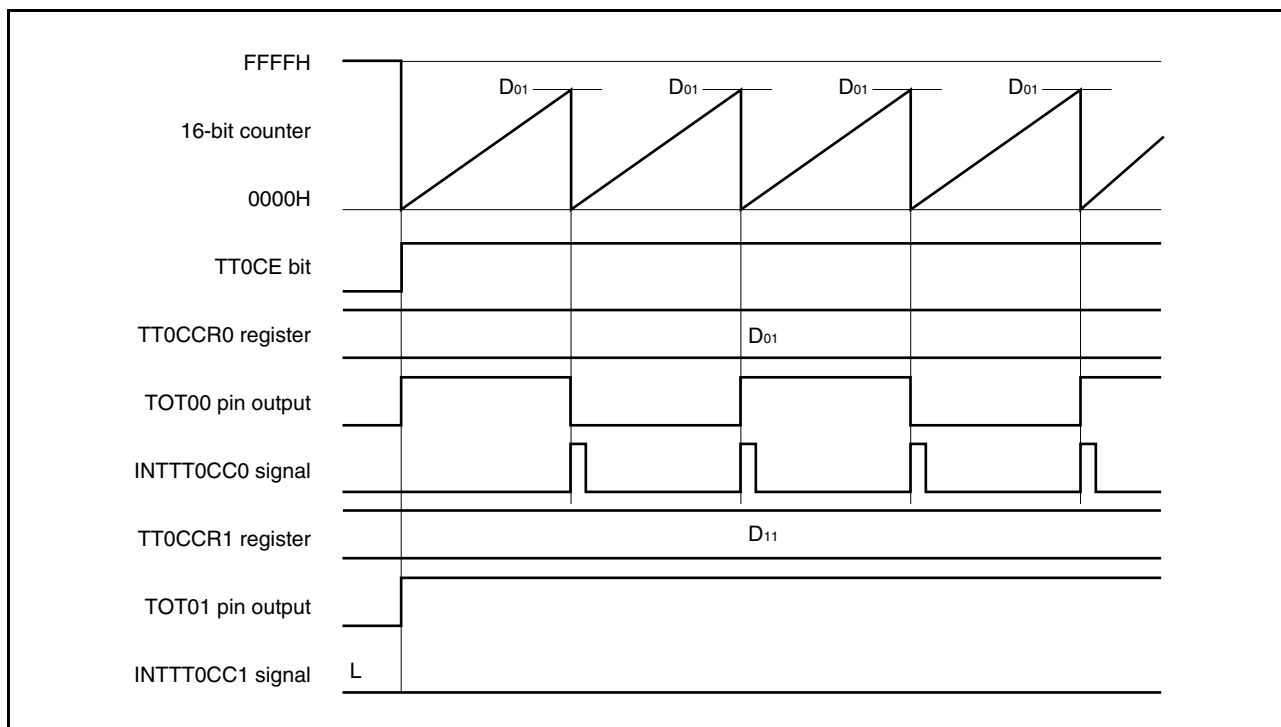
**9.6.2    External event count mode (TT0MD3 to TT0MD0 bits = 0001)**

In the external event count mode, the valid edge of the external event count input (EVTT0) is counted when the TT0CTL0.TT0CE bit is set to 1, and an interrupt request signal (INTTT0CC0) is generated each time the number of edges set by the TT0CCR0 register have been counted.  The TOT00 and TOT01 pins cannot be used.

The TT0CCR1 register is not used in the external event count mode.

**Figure 9-14.  Configuration in External Event Count Mode**



**Note**   Set by the TT0IOC2.TT0EES1 and TT0IOC2.TT0EES0 bits.
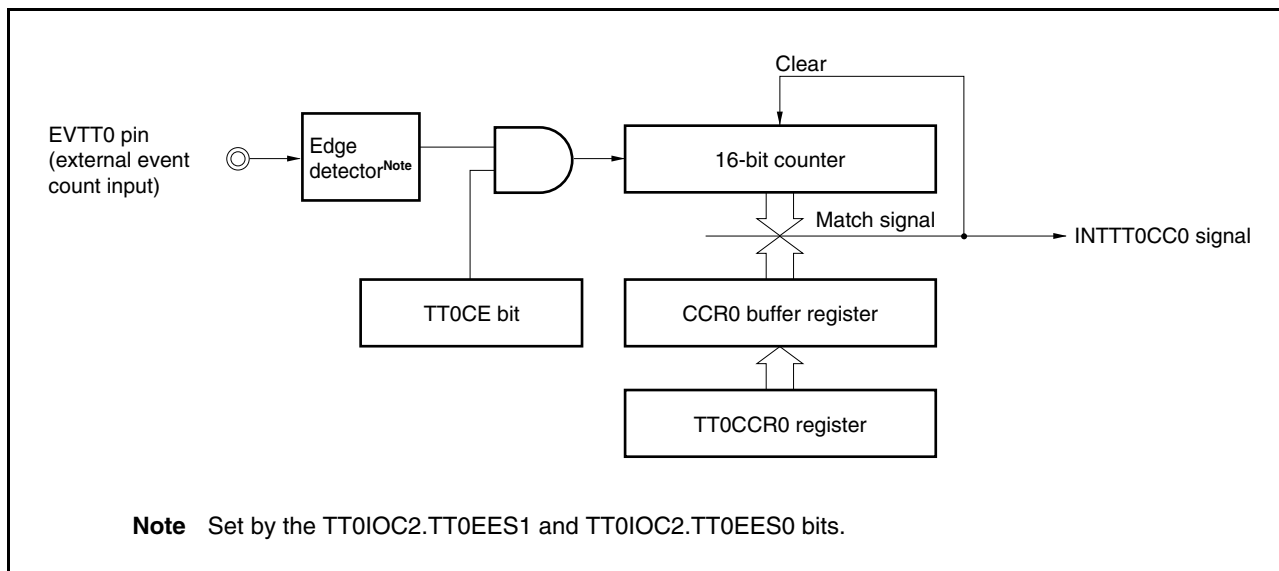
**Figure 9-15. Basic Timing in External Event Count Mode**
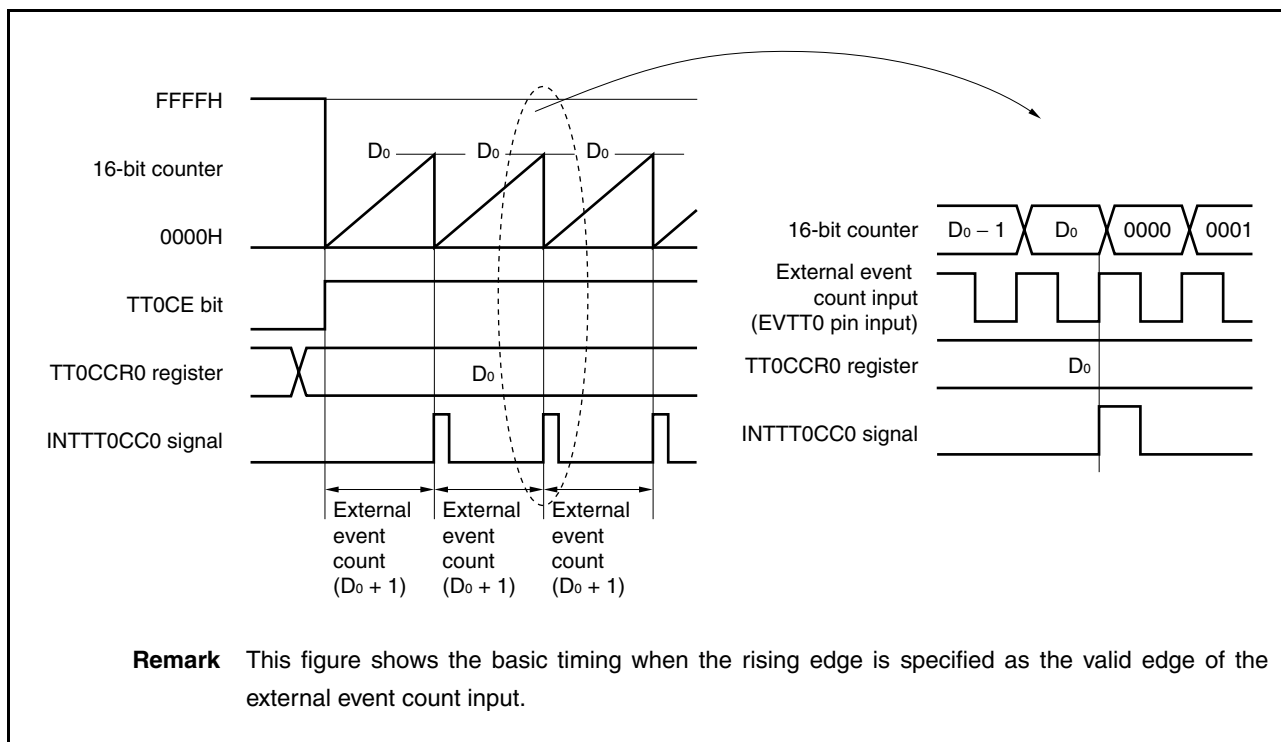


**Remark**    This figure shows the basic timing when the rising edge is specified as the valid edge of the external event count input.

When the TT0CE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H.  The counter counts each time the valid edge of external event count input is detected.  Additionally, the set value of the TT0CCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, and a compare match interrupt request signal (INTTT0CC0) is generated.

The INTTT0CC0 signal is generated each time the valid edge of the external event count input has been detected "value set to TT0CCR0 register + 1" times.

**Figure 9-16.  Register Setting for Operation in External Event Count Mode (1/2)**



(a)  **TMT0 control register 0 (TT0CTL0)**

(b)  **TMT0 control register 1 (TT0CTL1)**

(c)  **TMT0 I/O control register 2 (TT0IOC2)**

(d)  **TMT0 counter read buffer register (TT0CNT)**
     The count value of the 16-bit counter can be read by reading the TT0CNT register.

(e)  **TMT0 capture/compare register 0 (TT0CCR0)**
     If the TT0CCR0 register is set to $D_0$, the count is cleared when the number of external events has reached $(D_0 + 1)$ and the compare match interrupt request signal (INTTT0CC0) is generated.
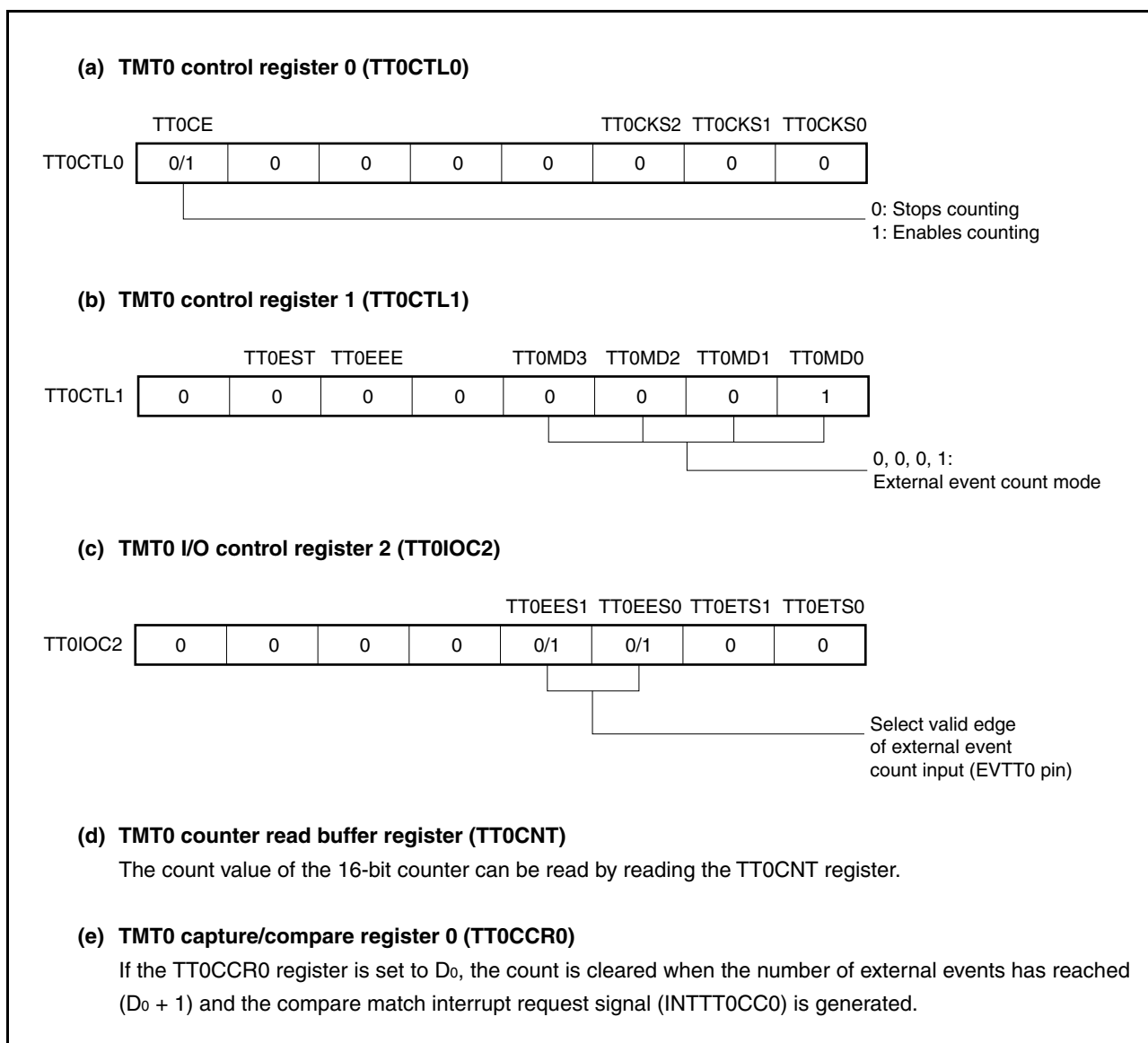
**Figure 9-16.  Register Setting for Operation in External Event Count Mode (2/2)**

**(f)  TMT0 capture/compare register 1 (TT0CCR1)**

The TT0CCR1 register is not used in the external event count mode.  However, the set value of the TT0CCR1 register is transferred to the CCR1 buffer register.  When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTT0CC1) is generated.

When the TT0CCR1 register is not used, it is recommended to set its value to FFFFH.  Also mask the register by the interrupt mask flag (TT0CCIC1.TT0CCMK1).

**Remark**   TMT0 control register 2 (TT0CTL2), TMT0 I/O control register 0 (TT0IOC0), TMT0 I/O control register 1 (TT0IOC1), TMT0 I/O control register 3 (TT0IOC3), TMT0 option register 0 (TT0OPT0), TMT0 option register 1 (TT0OPT1), and TMT0 counter write register (TT0TCW) are not used in the external event count mode.

**(1)  External event count mode operation flow**

**Figure 9-17.  Software Processing Flow in External Event Count Mode**
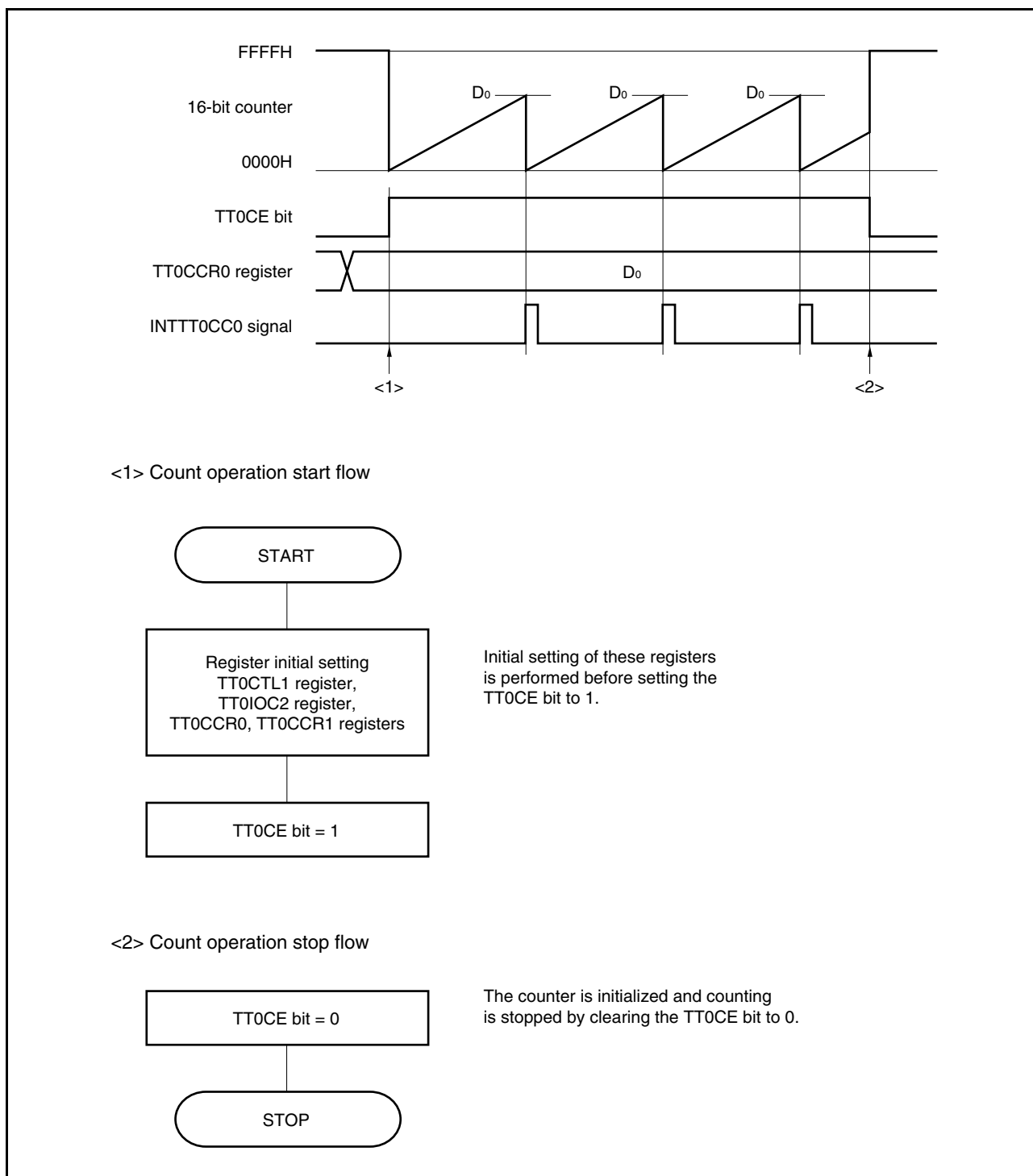
**(2) Operation timing in external event count mode**

**(a) Operation if TT0CCR0 register is set to 0000H**

When the TT0CCR0 register is set to 0000H, the 16-bit counter is repeatedly cleared to 0000H and generates the INTTT0CC0 signal each time it has detected the valid edge of the external event count signal and its value has matched that of the CCR0 buffer register.
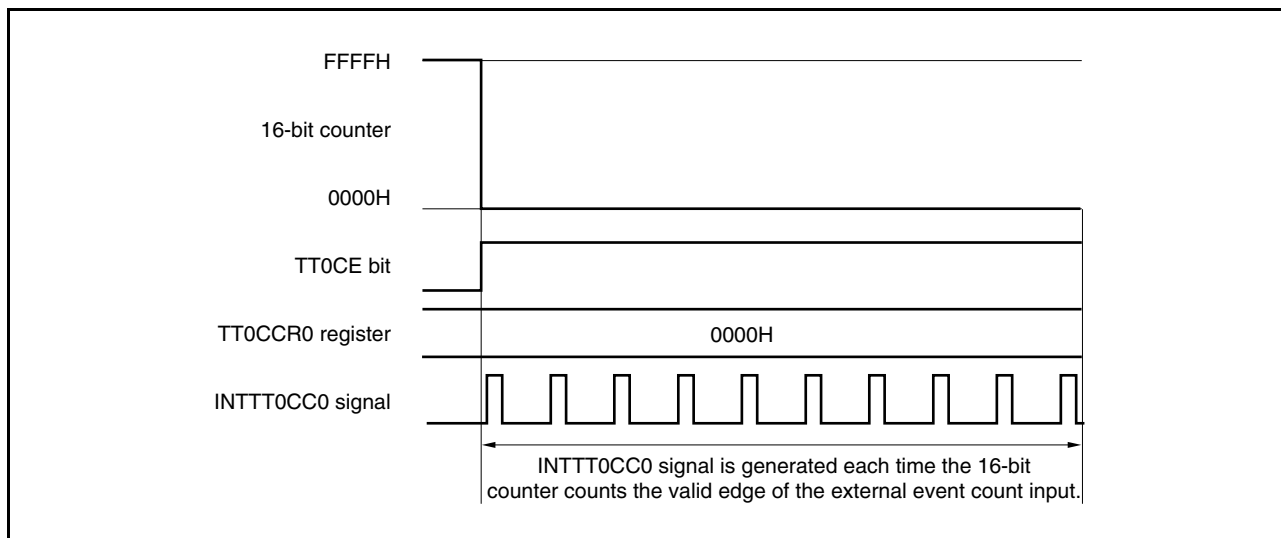
The value of the 16-bit counter is always 0000H.



**(b) Operation if TT0CCR0 register is set to FFFFH**

If the TT0CCR0 register is set to FFFFH, the 16-bit counter counts up to FFFFH each time the valid edge of the external event count signal has been detected. The 16-bit counter is cleared to 0000H in synchronization with the next count-up timing, and the INTTT0CC0 signal is generated. At this time, the TT0OPT0.TT0OVF bit is not set.
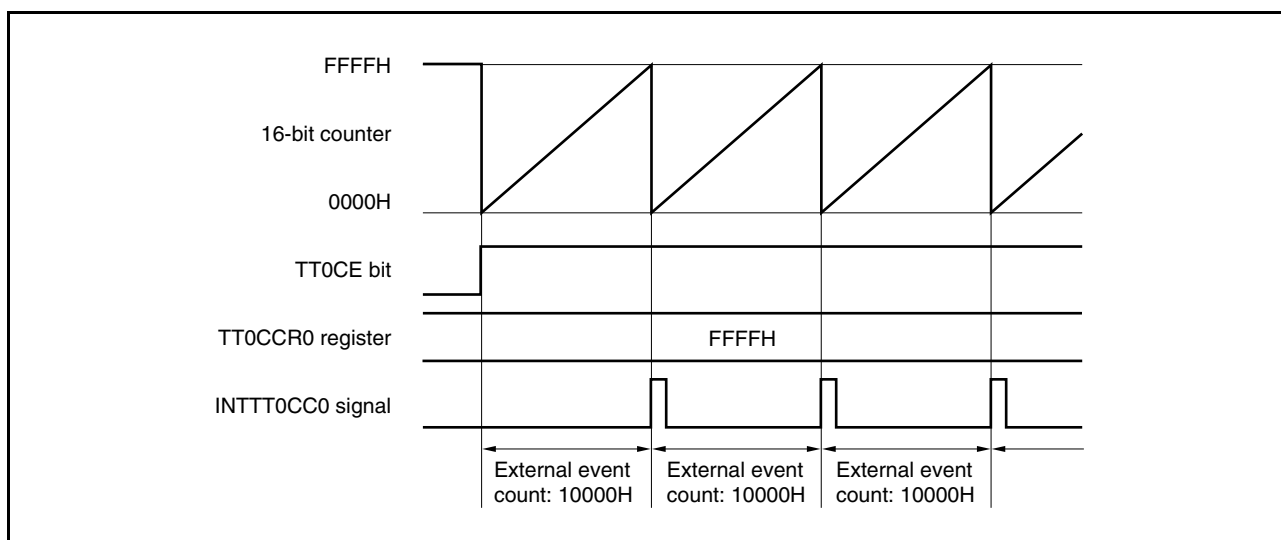
**(c) Operation with TT0CCR0 set to FFFFH and TT0CCR1 register to 0000H**

When the TT0CCR0 register is set to FFFFH, the 16-bit counter counts up to FFFFH each time it has detected the valid edge of the external event count signal.  The counter is then cleared to 0000H in synchronization with the next count-up timing and the INTTT0CC0 signal is generated.  At this time, the TT0OPT0.TT0OVF bit is not set.

If the TT0CCR1 register is set to 0000H, the INTTT0CC1 signal is generated when the 16-bit counter is cleared to 0000H.

**(d) Notes on rewriting TT0CCR0 register**

If the value of the TT0CCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow.  When an overflow may occur, stop counting once and then change the set value.



If the value of the TT0CCR0 register is changed from $D_1$ to $D_2$ while the count value is greater than $D_2$ but less than $D_1$, the count value is transferred to the CCR0 buffer register as soon as the TT0CCR0 register has been rewritten.  Consequently, the value that is compared with the 16-bit counter is $D_2$.

Because the count value has already exceeded $D_2$, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H.  When the count value matches $D_2$, the INTTT0CC0 signal is generated.  Therefore, the INTTT0CC0 signal may not be generated at the valid edge count of "$(D_1 + 1)$ times" or "$(D_2 + 1)$ times" as originally expected, but may be generated at the valid edge count of "$(10000H + D_2 + 1)$ times".

**(e) Operation of TT0CCR1 register**

**Figure 9-18. Configuration of TT0CCR1 Register**
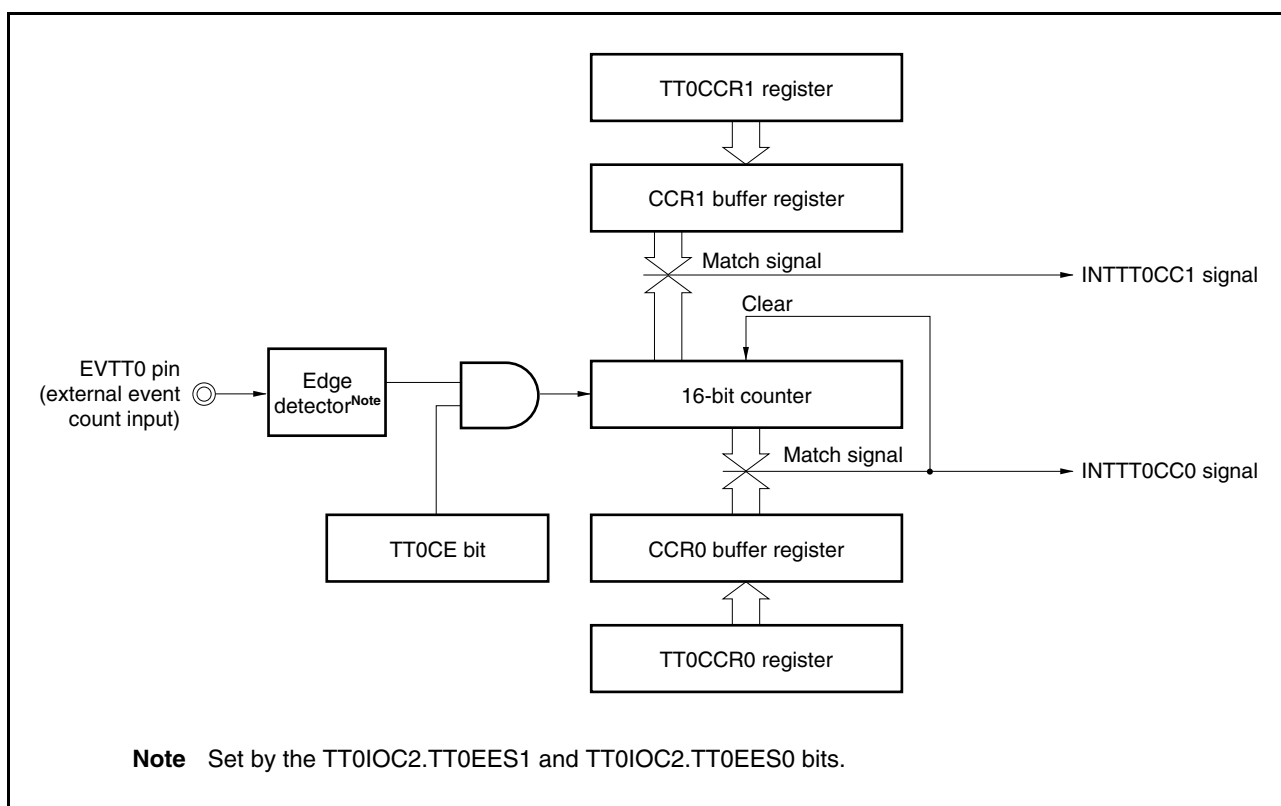


**Note**   Set by the TT0IOC2.TT0EES1 and TT0IOC2.TT0EES0 bits.

If the set value of the TT0CCR1 register is smaller than the set value of the TT0CCR0 register, the INTTT0CC1 signal is generated once per cycle.

**Figure 9-19. Timing Chart When $D_{01} \geq D_{11}$**
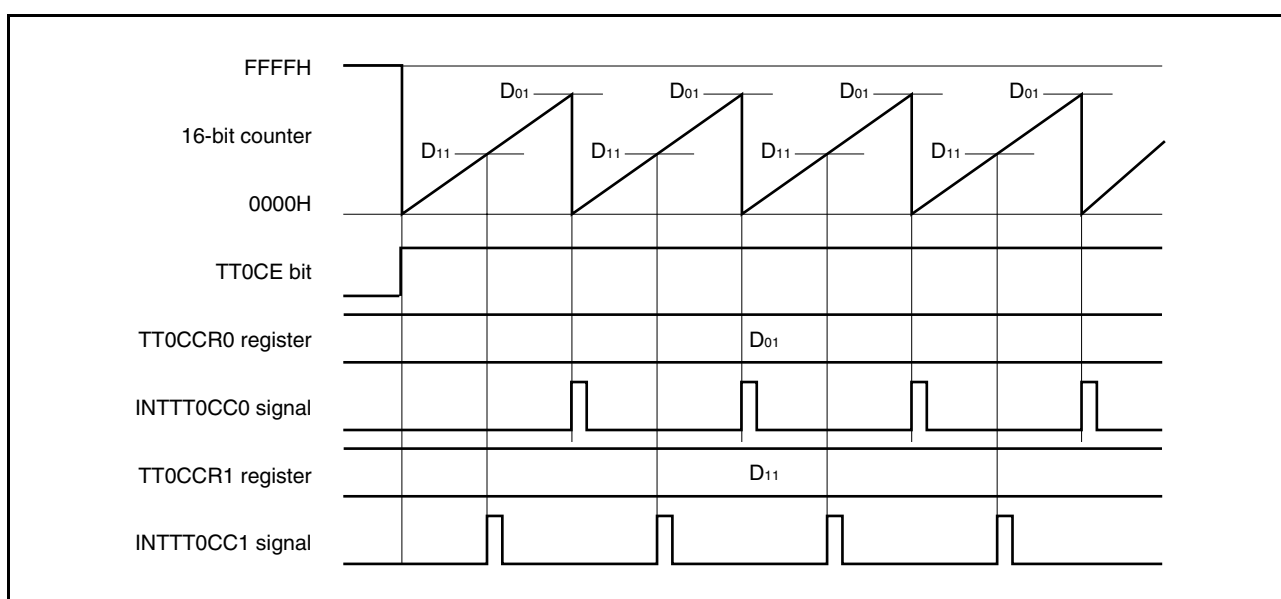
If the set value of the TT0CCR1 register is greater than the set value of the TT0CCR0 register, the INTTT0CC1 signal is not generated because the count value of the 16-bit counter and the value of the TT0CCR1 register do not match.

When the TT0CCR1 register is not used, it is recommended to set its value to FFFFH.

**Figure 9-20.  Timing Chart When $D_{01} < D_{11}$**
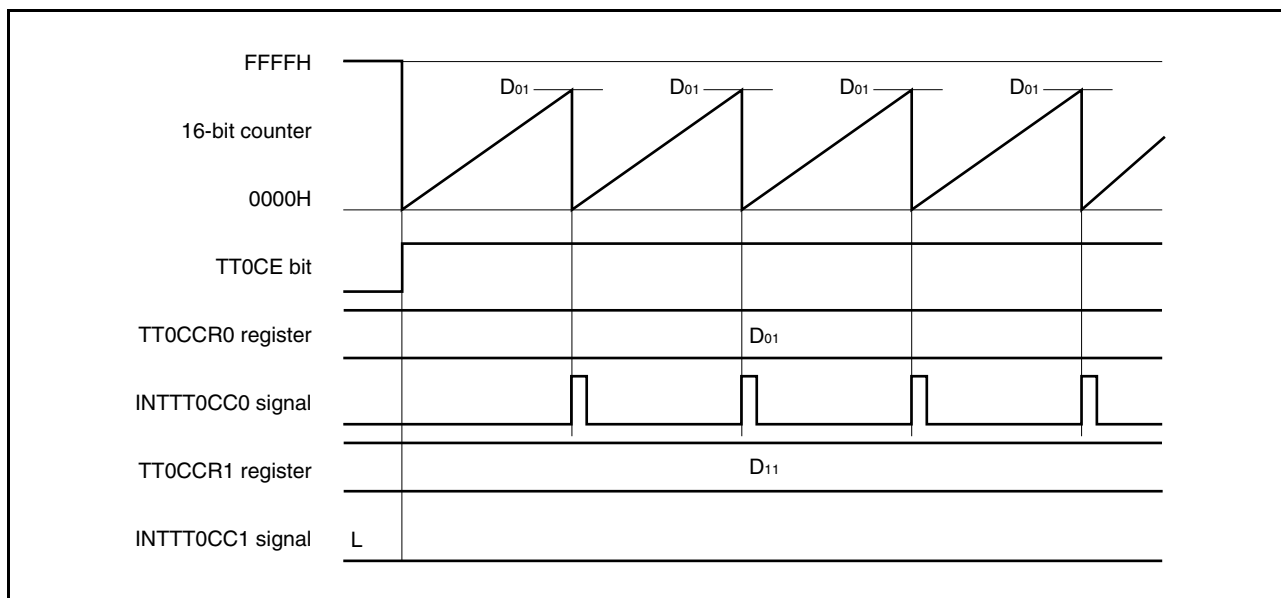
### 9.6.3 External trigger pulse output mode (TT0MD3 to TT0MD0 bits = 0010)

In the external trigger pulse output mode, 16-bit timer/event counter T waits for a trigger when the TT0CTL0.TT0CE bit is set to 1. When the valid edge of an external trigger input (EVTT0) is detected, 16-bit timer/event counter T starts counting, and outputs a PWM waveform from the TOT01 pin.

Pulses can also be output by generating a software trigger instead of using the external trigger. When using a software trigger, a square wave that has the set value of the TT0CCR0 register + 1 as half its cycle can also be output from the TOT00 pin.

**Figure 9-21. Configuration in External Trigger Pulse Output Mode**



**Notes 1.** Since the external trigger input pin and external event count input pin are the same alternate-function pin, the external event count input function cannot be used.

**2.** Edge detector for external trigger input.
Set by the TT0IOC2.TT0ETS1 and TT0IOC2.TT0ETS0 bits.

**3.** Edge detector for external event count input.
Set by the TT0IOC2.TT0EES1 and TT0IOC2.TT0EES0 bits.
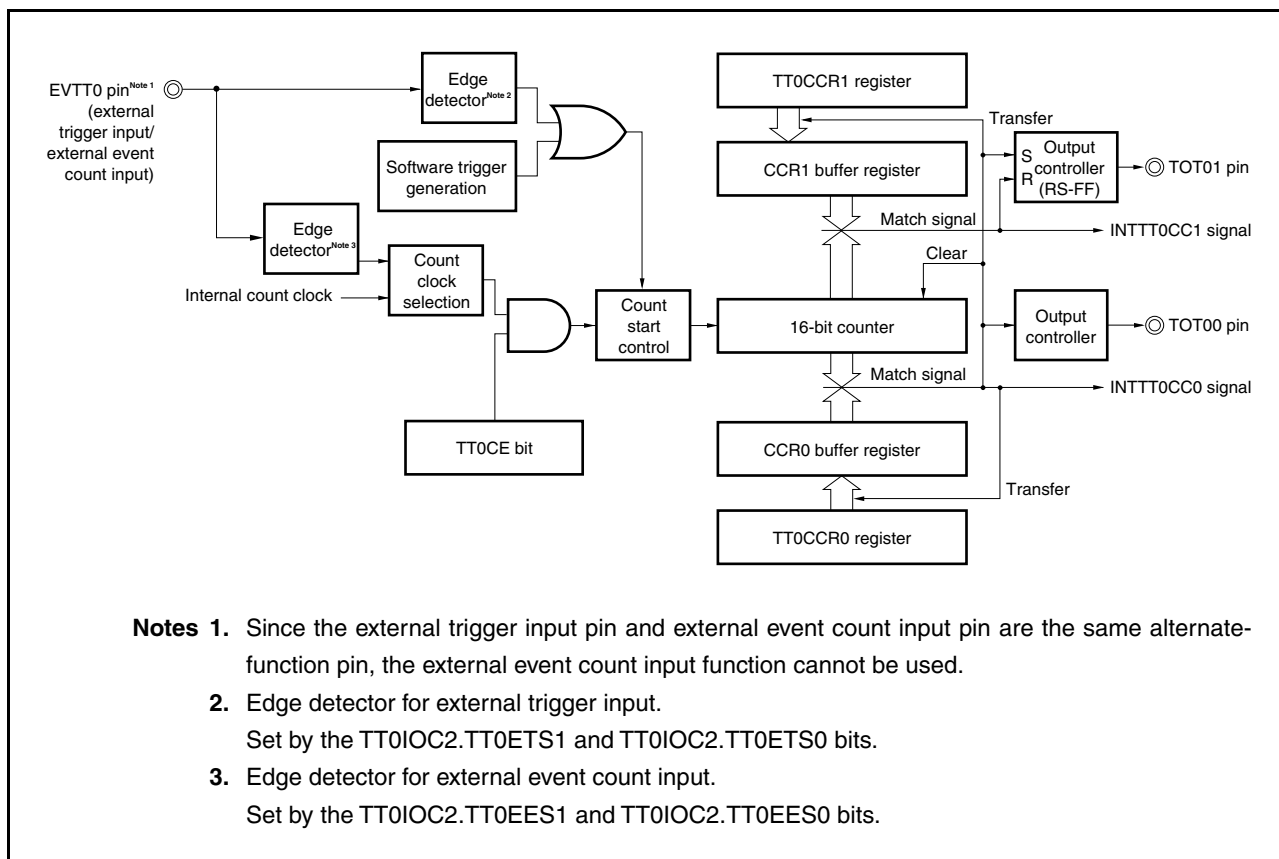
**Figure 9-22.  Basic Timing in External Trigger Pulse Output Mode**
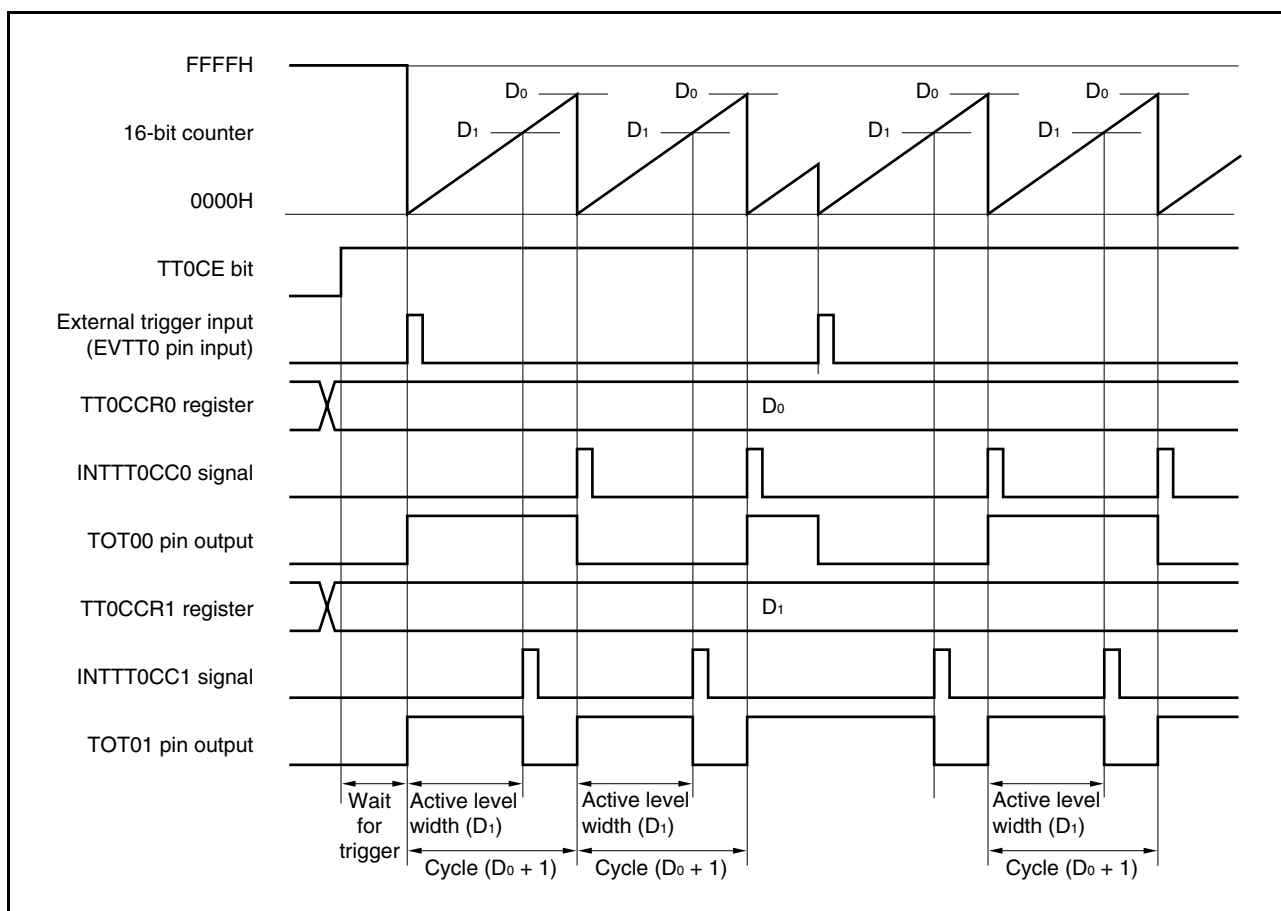


16-bit timer/event counter T waits for a trigger when the TT0CE bit is set to 1.  When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting at the same time, and outputs a PWM waveform from the TOT01 pin.  If the trigger is generated again while the counter is operating, the counter is cleared to 0000H and restarted.  (The output of the TOT00 pin is inverted.  The TOT01 pin outputs a high level regardless of the status (high/low) when a trigger occurs.)

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

    Active level width = (Set value of TT0CCR1 register) $\times$ Count clock cycle
    Cycle = (Set value of TT0CCR0 register + 1) $\times$ Count clock cycle
    Duty factor = (Set value of TT0CCR1 register)/(Set value of TT0CCR0 register + 1)

The compare match request signal (INTTT0CC0) is generated the next time the 16-bit counter counts after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H.  The compare match interrupt request signal (INTTT0CC1) is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The value set to the TT0CCRn register is transferred to the CCRn buffer register when the count value of the 16-bit counter matches the value of the CCRn buffer register and the 16-bit counter is cleared to 0000H.

The valid edge of an external trigger input (EVTT0), or setting the software trigger (TT0CTL1.TT0EST bit) to 1 is used as the trigger (n = 0, 1).

**Figure 9-23.  Setting of Registers in External Trigger Pulse Output Mode (1/2)**



**(a) TMT0 control register 0 (TT0CTL0)**

TT0CTL0: TT0CE [0/1] [0] [0] [0] [0] TT0CKS2 [0/1] TT0CKS1 [0/1] TT0CKS0 [0/1]

- Select count clock
- 0: Stops counting
- 1: Enables counting

**(b) TMT0 control register 1 (TT0CTL1)**

TT0CTL1: [0] TT0EST [0/1] TT0EEE [0] [0] TT0MD3 [0] TT0MD2 [0] TT0MD1 [1] TT0MD0 [0]

- 0, 0, 1, 0:
  External trigger pulse output mode
- Generates software trigger when 1 is written

**(c) TMT0 I/O control register 0 (TT0IOC0)**

TT0IOC0: [0] [0] [0] [0] TT0OL1 [0/1] TT0OE1 [0/1] TT0OL0 [0/1] TT0OE0 [0/1]

- 0: Disables TOT00 pin output
  1: Enables TOT00 pin output
- Setting of TOT00 pin output level while waiting for external trigger
  0: Low level
  1: High level
- 0: Disables TOT01 pin output
  1: Enables TOT01 pin output
- Setting of TOT01 pin output level while waiting for external trigger
  0: Low level
  1: High level

- When TT0OL1 bit = 0

  16-bit counter
  TOT01 pin output

- When TT0OL1 bit = 1

  16-bit counter
  TOT01 pin output

**Figure 9-23. Setting of Registers in External Trigger Pulse Output Mode (2/2)**

**(d) TMT0 I/O control register 2 (TT0IOC2)**

|  | | | | | TT0EES1 | TT0EES0 | TT0ETS1 | TT0ETS0 |
|---|---|---|---|---|---|---|---|---|
| TT0IOC2 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |

Select valid edge of external trigger input (EVTT0 pin)**Note**

**Note**   Set the valid edge selection of the unused alternate external input signals to "No edge detection".

**(e) TMT0 counter read buffer register (TT0CNT)**
The value of the 16-bit counter can be read by reading the TT0CNT register.

**(f) TMT0 capture/compare registers 0 and 1 (TT0CCR0 and TT0CCR1)**
If $D_0$ is set to the TT0CCR0 register and $D_1$ to the TT0CCR1 register, the cycle and active level of the PWM waveform are as follows.

Cycle = $(D_0 + 1) \times$ Count clock cycle
Active level width = $D_1 \times$ Count clock cycle

**Remark**   TMT0 control register 2 (TT0CTL2), TMT0 I/O control register 1 (TT0IOC1), TMT0 I/O control register 3 (TT0IOC3), TMT0 option register 0 (TT0OPT0), TMT0 option register 1 (TT0OPT1), and TMT0 counter write register (TT0TCW) are not used in the external trigger pulse output mode.

**(1) Operation flow in external trigger pulse output mode**

**Figure 9-24. Software Processing Flow in External Trigger Pulse Output Mode (1/2)**
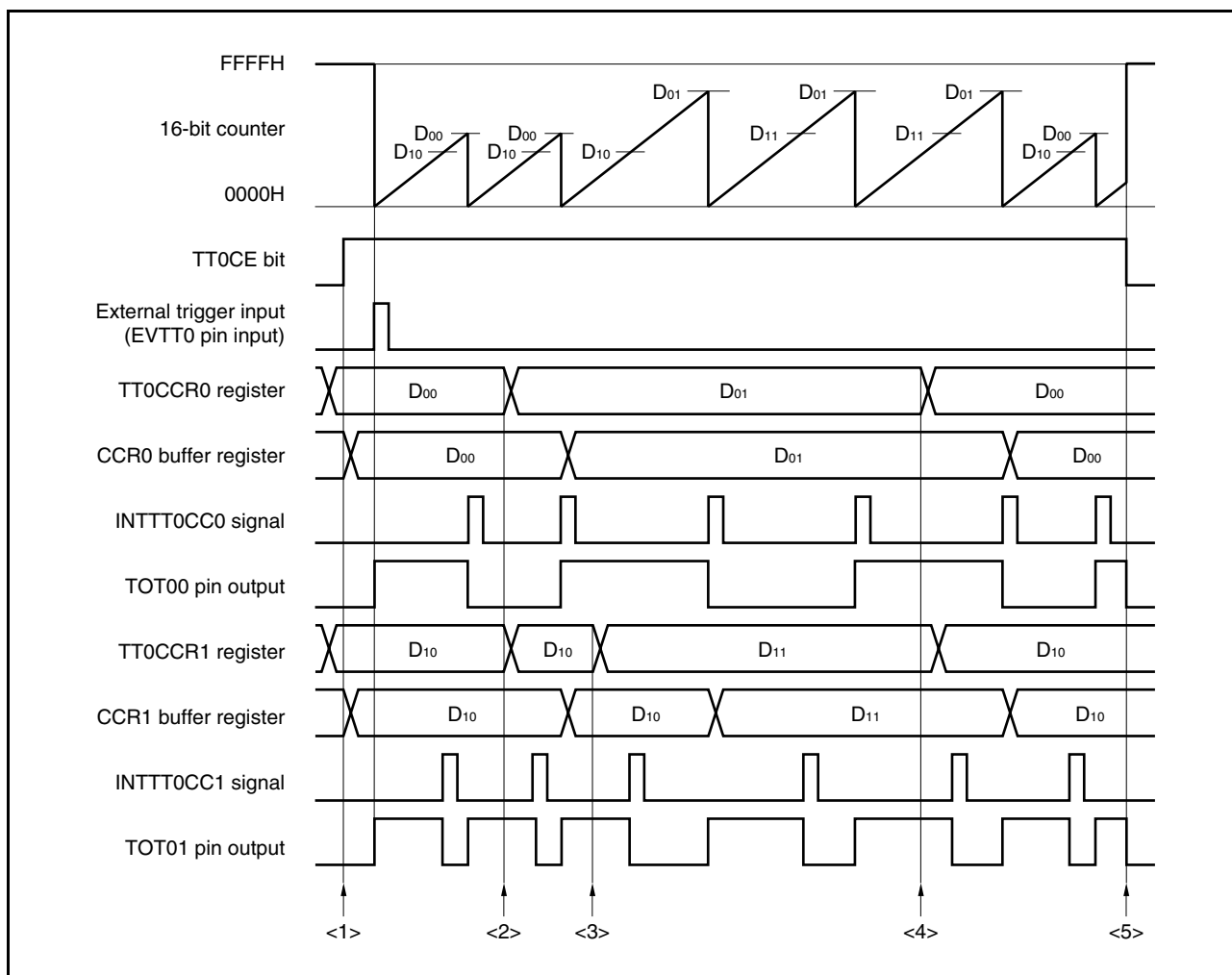
**Figure 9-24.  Software Processing Flow in External Trigger Pulse Output Mode (2/2)**

**(2)  External trigger pulse output mode operation timing**

**(a)  Note on changing pulse width during operation**

To change the PWM waveform while the counter is operating, write the TT0CCR1 register last.

Rewrite the TT0CCRn register after writing the TT0CCR1 register after the INTTT0CC0 signal is detected.

In order to transfer data from the TT0CCRn register to the CCRn buffer register, the TT0CCR1 register must be written.

To change both the cycle and active level width of the PWM waveform at this time, first set the cycle to the TT0CCR0 register and then set the active level width to the TT0CCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TT0CCR0 register, and then write the same value (same as preset value of the TT0CCR1 register) to the TT0CCR1 register.

To change only the active level width (duty factor) of the PWM waveform, only the TT0CCR1 register has to be set.

After data is written to the TT0CCR1 register, the value written to the TT0CCRn register is transferred to the CCRn buffer register in synchronization with clearing of the 16-bit counter, and is used as the value compared with the 16-bit counter.

To write the TT0CCR0 or TT0CCR1 register again after writing the TT0CCR1 register once, do so after the INTTT0CC0 signal is generated. Otherwise, the value of the CCRn buffer register may become undefined because the timing of transferring data from the TT0CCRn register to the CCRn buffer register conflicts with writing the TT0CCRn register.

**Remark**    n = 0, 1

**(b)  0%/100% output of PWM waveform**

To output a 0% waveform, set the TT0CCR1 register to 0000H.  The 16-bit counter is cleared to 0000H and the INTTT0CC0 and INTTT0CC1 signals are generated after a match between the count value of the 16-bit counter and the value of the CCR0 buffer register.



**Note**  The timing is actually delayed by one operating clock ($f_{XX}$).

To output a 100% waveform, set a value of (set value of TT0CCR0 register + 1) to the TT0CCR1 register.  If the set value of the TT0CCR0 register is FFFFH, 100% output cannot be produced.



**Note**   The timing is actually delayed by one operating clock ($f_{xx}$).

**(c)  Conflict between trigger detection and match with CCR1 buffer register**

If the trigger is detected immediately after the INTTT0CC1 signal is generated, the 16-bit counter is immediately cleared to 0000H, the TOT01 pin is asserted, and the counter continues counting.  Consequently, the inactive period of the PWM waveform is shortened.
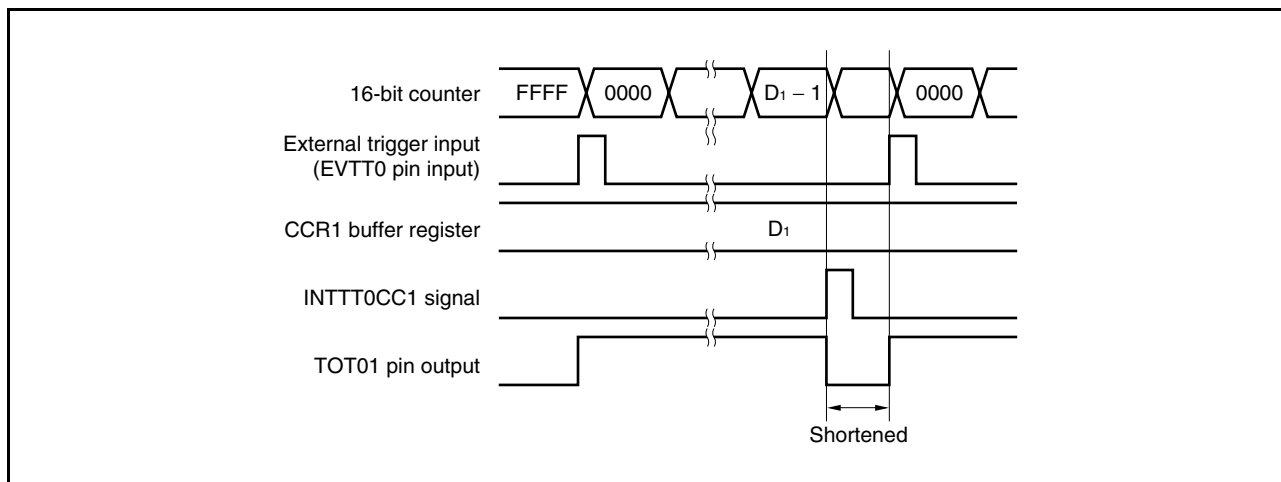


If the trigger is detected immediately before the INTTT0CC1 signal is generated, the INTTT0CC1 signal is not generated, and the 16-bit counter is cleared to 0000H and continues counting.  The output signal of the TOT01 pin remains active.  Consequently, the active period of the PWM waveform is extended.
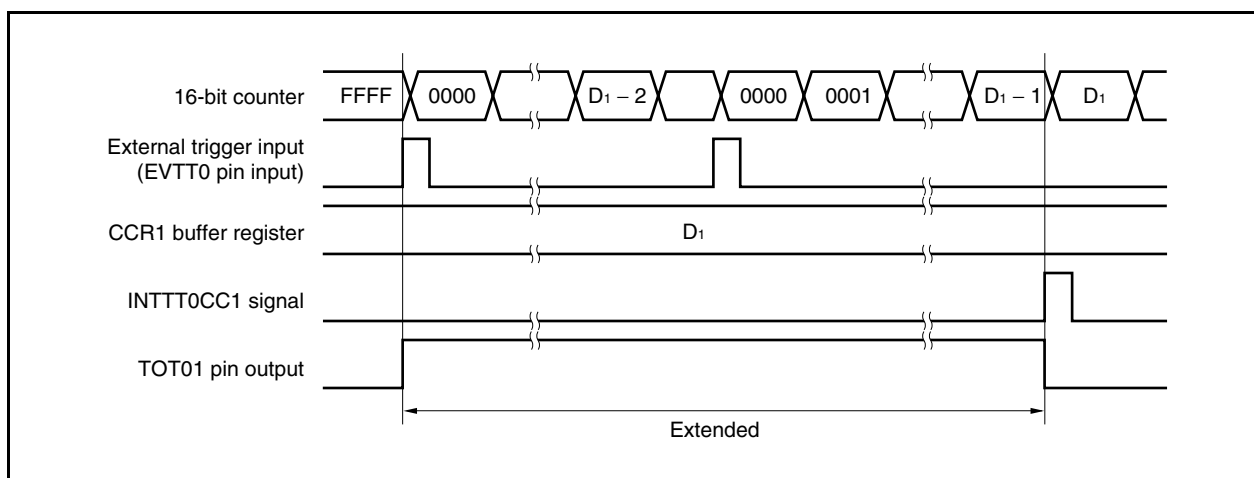
**(d) Conflict between trigger detection and match with CCR0 buffer register**

If the trigger is detected immediately after the INTTT0CC0 signal is generated, the 16-bit counter is cleared to 0000H and continues counting up again from that point. Therefore, the active period of the TOT01 pin is extended by the time from generation of the INTTT0CC0 signal to trigger detection.
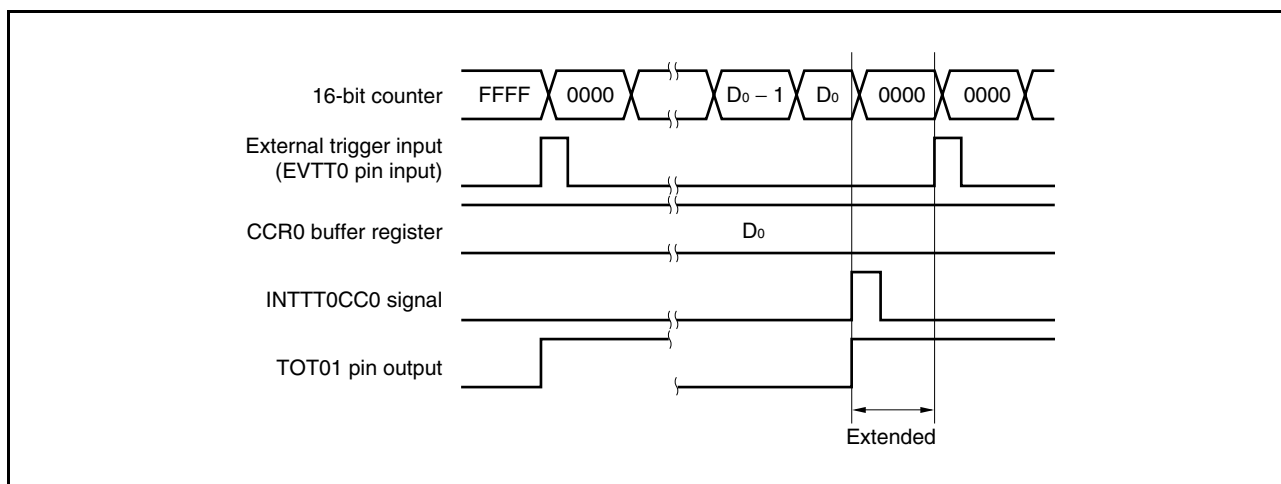


If the trigger is detected immediately before the INTTT0CC0 signal is generated, the INTTT0CC0 signal is not generated. The 16-bit counter is cleared to 0000H, the TOT01 pin is asserted, and the counter continues counting. Consequently, the inactive period of the PWM waveform is shortened.
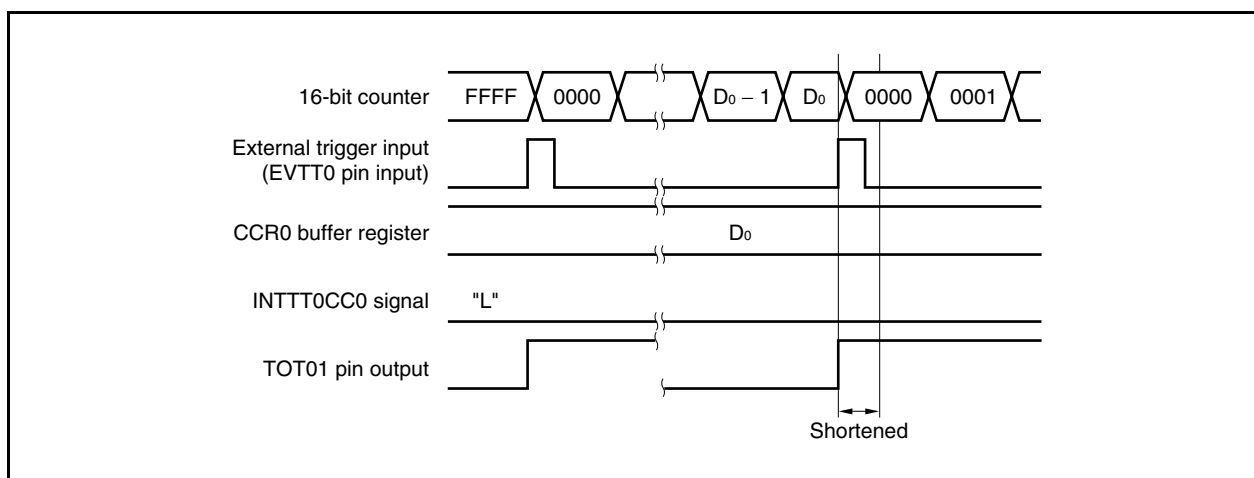
**(e) Generation timing of compare match interrupt request signal (INTTT0CC1)**

The timing of generating the INTTT0CC1 signal in the external trigger pulse output mode differs from the timing of generating INTTT0CC1 signals in other modes; the INTTT0CC1 signal is generated when the count value of the 16-bit counter matches the value of the TT0CCR1 register.



| Count clock | | | | | |
| 16-bit counter | $D_1 - 2$ | $D_1 - 1$ | $D_1$ | $D_1 + 1$ | $D_1 + 2$ |
| TT0CCR1 register | | | $D_1$ | | |
| TOT01 pin output | | | **Note** | | |
| INTTT0CC1 signal | | | **Note** | | |

**Note**   The timing is actually delayed by one operating clock ($f_{XX}$).
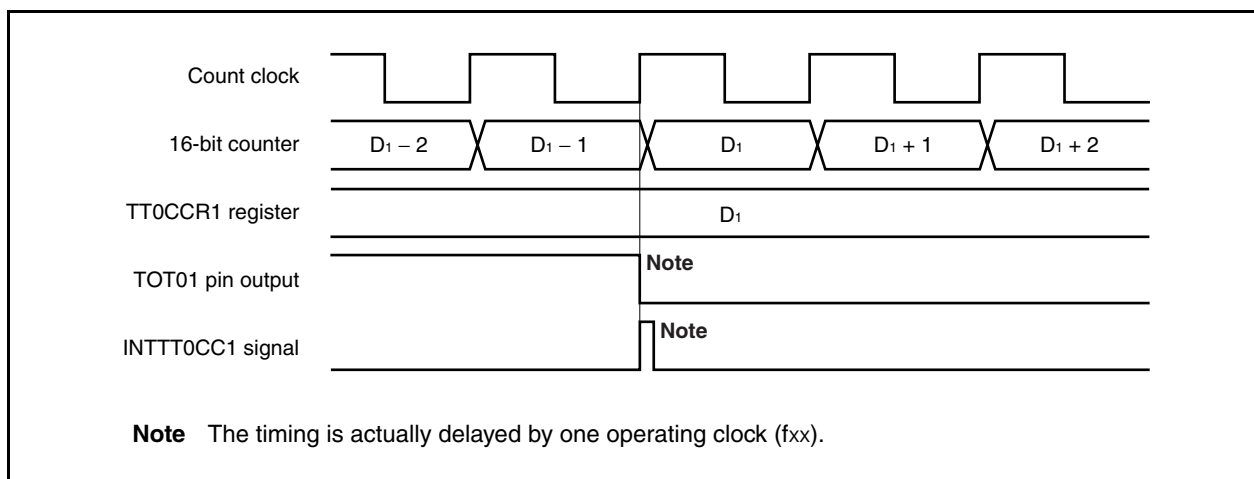
Usually, the INTTT0CC1 signal is generated in synchronization with the next count-up, after the count value of the 16-bit counter matches the value of the TT0CCR1 register.

In the external trigger pulse output mode, however, it is generated one clock earlier.  This is because the timing is changed to match the timing when the output signal of the TOT01 pin changes.

### 9.6.4  One-shot pulse output mode (TT0MD3 to TT0MD0 bits = 0011)

In the one-shot pulse output mode, 16-bit timer/event counter T waits for a trigger when the TT0CTL0.TT0CE bit is set to 1.  When the valid edge of an external trigger input (EVTT0) is detected, 16-bit timer/event counter T starts counting, and outputs a one-shot pulse from the TOT01 pin.

Instead of the external trigger input (EVTT0), a software trigger can also be generated to output the pulse.  When the software trigger is used, the TOT00 pin outputs the active level while the 16-bit counter is counting, and the inactive level when the counter is stopped (waiting for a trigger).

**Figure 9-25.  Configuration in One-Shot Pulse Output Mode**



> **Notes 1.** Because the external trigger input pin (EVTT0) and external event count input pin (EVTT0) are the same alternate-function pin, the external trigger input pin (EVTT0) cannot be used.
> **2.** Edge detector for external trigger input.
> Set by the TT0IOC2.TT0ETS1 and TT0IOC2.TT0ETS0 bits.
> **3.** Edge detector for external event count input.
> Set by the TT0IOC2.TT0EES1 and TT0IOC2.TT0EES0 bits.
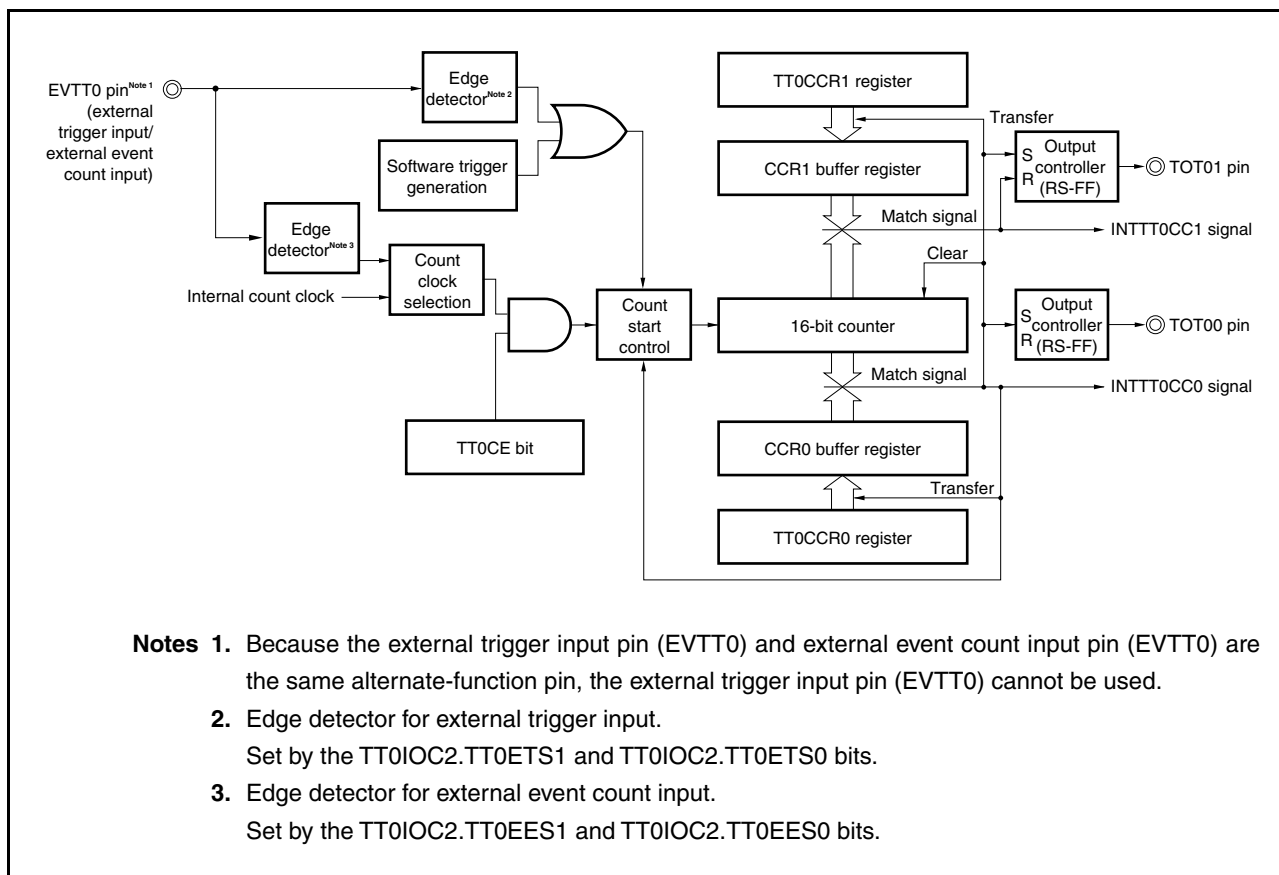
**Figure 9-26.  Basic Timing in One-Shot Pulse Output Mode**
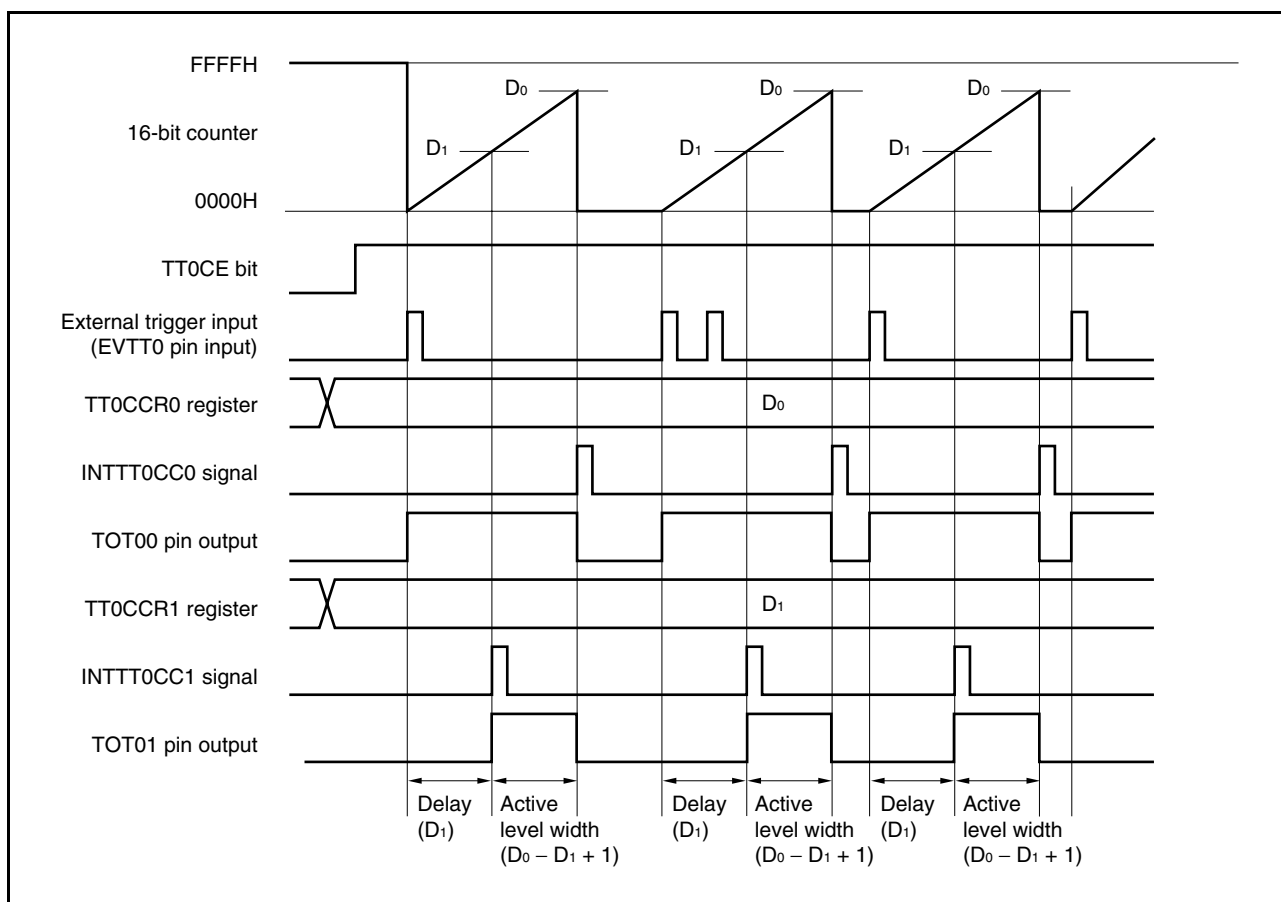


When the TT0CE bit is set to 1, 16-bit timer/event counter T waits for a trigger.  When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a one-shot pulse from the TOT01 pin.  After the one-shot pulse is output, the 16-bit counter is cleared to 0000H, stops counting, and waits for a trigger.  When the trigger is generated again, the 16-bit counter starts counting from 0000H.  If a trigger is generated again while the one-shot pulse is being output, it is ignored.

The output delay period and active level width of the one-shot pulse can be calculated as follows.

> Output delay period = (Set value of TT0CCR1 register) $\times$ Count clock cycle
> Active level width = (Set value of TT0CCR0 register $-$ Set value of TT0CCR1 register $+$ 1) $\times$ Count clock cycle

The compare match interrupt request signal (INTTT0CC0) is generated the next time the 16-bit counter counts after its count value matches the value of the CCR0 buffer register.  The compare match interrupt request signal (INTTT0CC1) is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The valid edge of an external trigger input (EVTT0 pin) or setting the software trigger (TT0CTL1.TT0EST bit) to 1 is used as the trigger.

**Figure 9-27.  Setting of Registers in One-Shot Pulse Output Mode (1/2)**



**(a)  TMT0 control register 0 (TT0CTL0)**

|  | TT0CE |  |  |  |  | TT0CKS2 | TT0CKS1 | TT0CKS0 |
|---|---|---|---|---|---|---|---|---|
| TT0CTL0 | 0/1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 |

Select count clock

0: Stops counting
1: Enables counting

**(b)  TMT0 control register 1 (TT0CTL1)**

|  |  | TT0EST | TT0EEE |  | TT0MD3 | TT0MD2 | TT0MD1 | TT0MD0 |
|---|---|---|---|---|---|---|---|---|
| TT0CTL1 | 0 | 0/1 | 0 | 0 | 0 | 0 | 1 | 1 |

0, 0, 1, 1:
One-shot pulse output mode

Generates software trigger
when 1 is written

**(c)  TMT0 I/O control register 0 (TT0IOC0)**

|  |  |  |  |  | TT0OL1 | TT0OE1 | TT0OL0 | TT0OE0 |
|---|---|---|---|---|---|---|---|---|
| TT0IOC0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |

0: Disables TOT00 pin output
1: Enables TOT00 pin output

Setting of TOT00 pin output level
while waiting for external trigger
0: Low level
1: High level

0: Disables TOT01 pin output
1: Enables TOT01 pin output

Setting of TOT01 pin output level
while waiting for external trigger
0: Low level
1: High level

• When TT0OL1 bit = 0

16-bit counter

TOT01 pin output

• When TT0OL1 bit = 1

16-bit counter

TOT01 pin output

**Figure 9-27. Setting of Registers in One-Shot Pulse Output Mode (2/2)**

**(d) TMT0 I/O control register 2 (TT0IOC2)**

| | | | | | | TT0EES1 | TT0EES0 | TT0ETS1 | TT0ETS0 |
|---|---|---|---|---|---|---|---|---|---|

| TT0IOC2 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
|---|---|---|---|---|---|---|---|---|

Select valid edge of external
trigger input (EVTT0 pin)**Note**

**Note**   Set the valid edge selection of the unused alternate external input signals to "No edge detection".

**(e) TMT0 counter read buffer register (TT0CNT)**

The value of the 16-bit counter can be read by reading the TT0CNT register.

**(f) TMT0 capture/compare registers 0 and 1 (TT0CCR0 and TT0CCR1)**

If $D_0$ is set to the TT0CCR0 register and $D_1$ to the TT0CCR1 register, the active level width and output delay period of the one-shot pulse are as follows.

Active level width $= (D_0 - D_1 + 1) \times$ Count clock cycle

Output delay period $= D_1 \times$ Count clock cycle

**Remark**   TMT0 control register 2 (TT0CTL2), TMT0 I/O control register 1 (TT0IOC1), TMT0 I/O control register 3 (TT0IOC3), TMT0 option register 0 (TT0OPT0), TMT0 option register 1 (TT0OPT1), and TMT0 counter write register (TT0TCW) are not used in the one-shot pulse output mode.

**(1)  Operation flow in one-shot pulse output mode**

**Figure 9-28.  Software Processing Flow in One-Shot Pulse Output Mode**
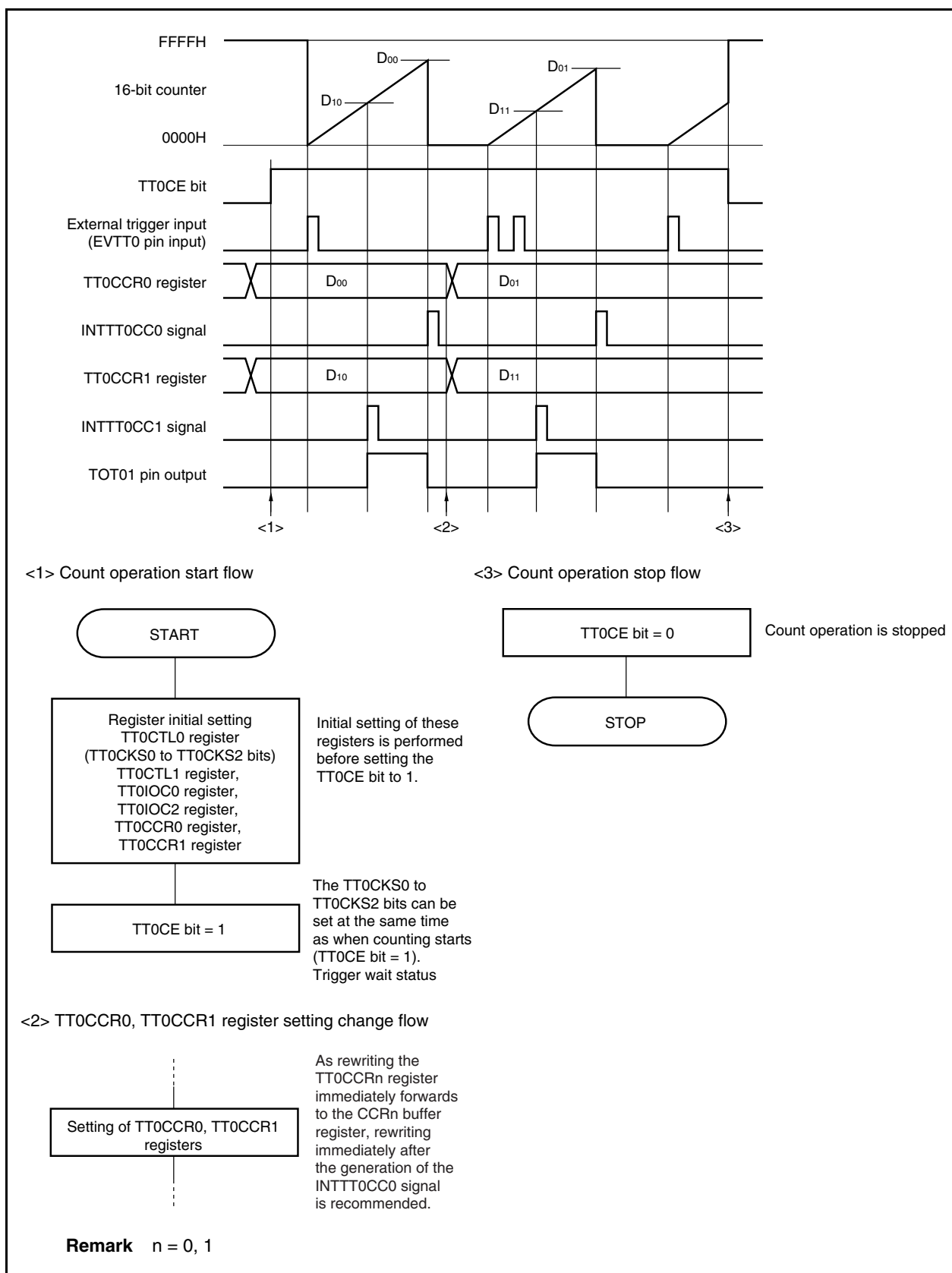
**(2) Operation timing in one-shot pulse output mode**

**(a) Note on rewriting TT0CCRn register**

If the value of the TT0CCRn register is rewritten to a smaller value during counting, the 16-bit counter may overflow. When an overflow may occur, stop counting and then change the set value.

**Remark**   n = 0, 1
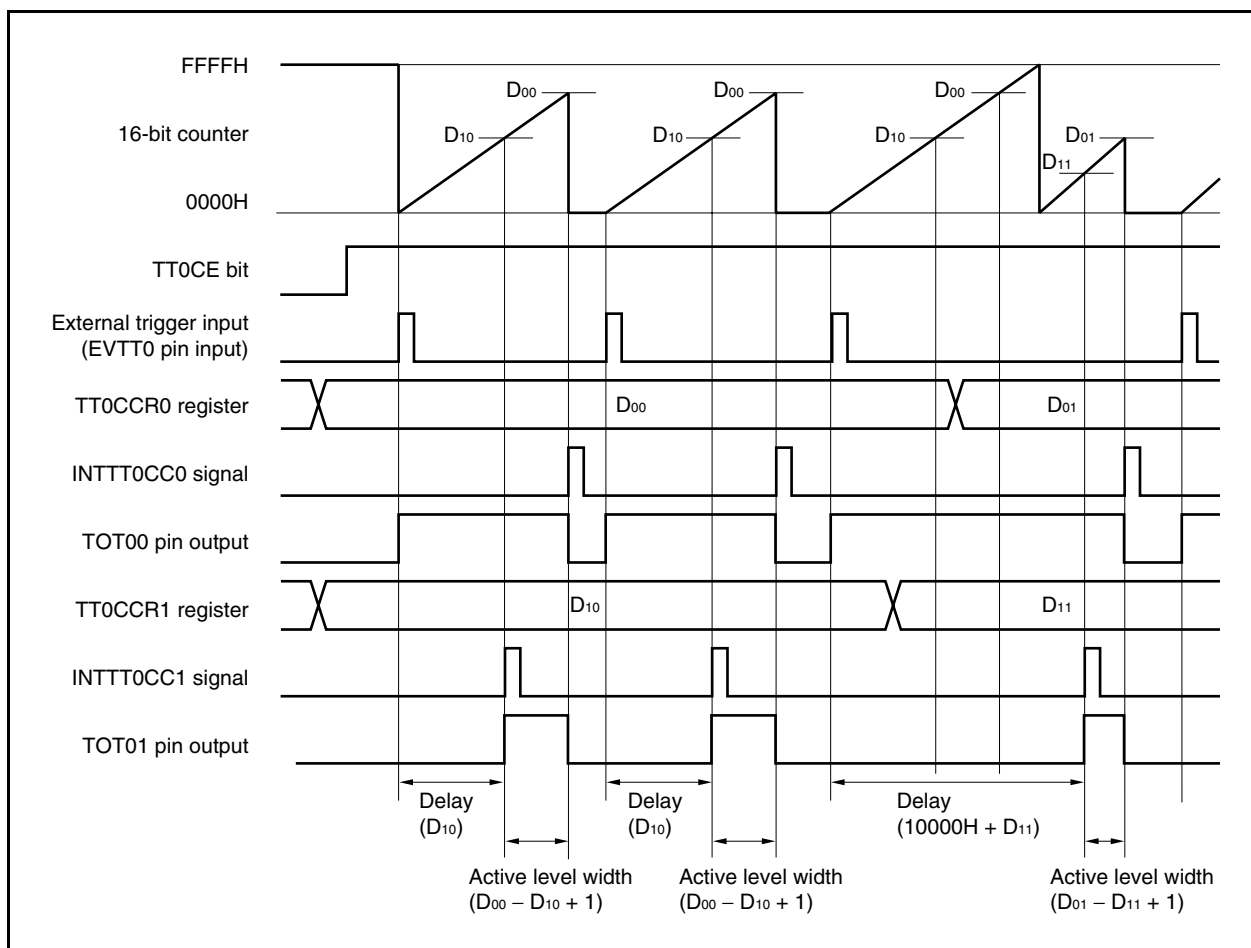


When the TT0CCR0 register is rewritten from $D_{00}$ to $D_{01}$ and the TT0CCR1 register from $D_{10}$ to $D_{11}$ where $D_{00} > D_{01}$ and $D_{10} > D_{11}$, if the TT0CCR1 register is rewritten when the count value of the 16-bit counter is greater than $D_{11}$ and less than $D_{10}$ and if the TT0CCR0 register is rewritten when the count value is greater than $D_{01}$ and less than $D_{00}$, each set value is reflected as soon as the register has been rewritten and compared with the count value. The counter counts up to FFFFH and then counts up again from 0000H. When the count value matches $D_{11}$, the counter generates the INTTT0CC1 signal and asserts the TOT01 pin. When the count value matches $D_{01}$, the counter generates the INTTT0CC0 signal, deasserts the TOT01 pin, and stops counting. Therefore, the counter may output a pulse with a delay period or active period different from that of the one-shot pulse that is originally expected.

**(b) Generation timing of compare match interrupt request signal (INTTT0CC1)**

The generation timing of the INTTT0CC1 signal in the one-shot pulse output mode is different from INTTT0CC1 signals in other modes; the INTTT0CC1 signal is generated when the count value of the 16-bit counter matches the value of the TT0CCR1 register.



| | Timing diagram |
|---|---|
| Count clock | |
| 16-bit counter | $D_1 - 2$   $D_1 - 1$   $D_1$   $D_1 + 1$   $D_1 + 2$ |
| TT0CCR1 register | $D_1$ |
| TOT01 pin output | **Note** |
| INTTT0CC1 signal | **Note** |

**Note**   The timing is actually delayed by one operating clock ($f_{XX}$).
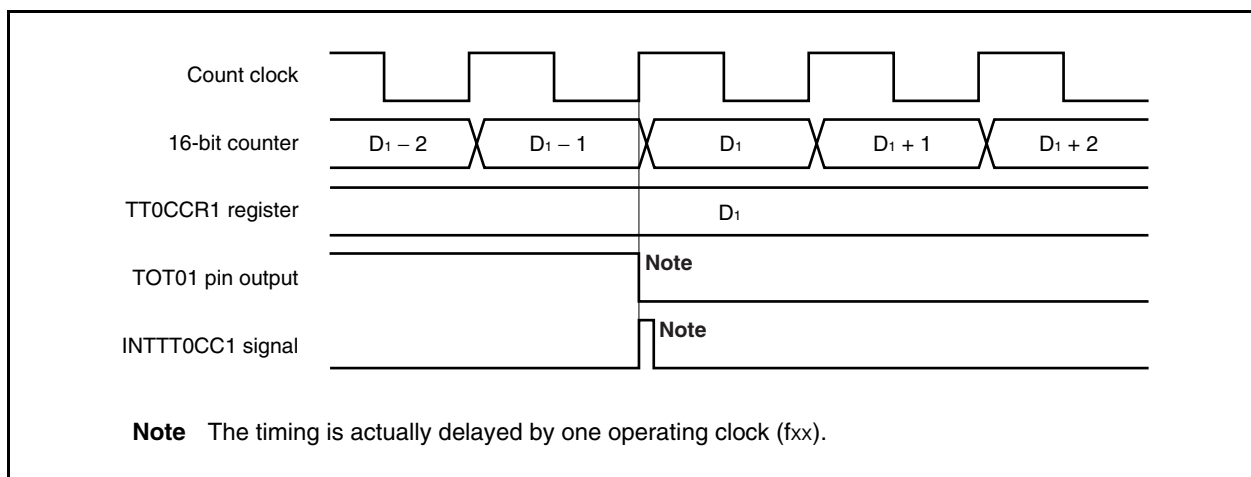
Usually, the INTTT0CC1 signal is generated the next time the 16-bit counter counts up after its count value matches the value of the TT0CCR1 register.

In the one-shot pulse output mode, however, it is generated one clock earlier.  This is because the timing is changed to match the timing the output signal of the TOT01 pin changes.

### 9.6.5  PWM output mode (TT0MD3 to TT0MD0 bits = 0100)

In the PWM output mode, a PWM waveform is output from the TOT01 pin when the TT0CTL0.TT0CE bit is set to 1.

In addition, a square wave with the set value of the TT0CCR0 register + 1 as half its cycle is output from the TOT00 pin.

**Figure 9-29.  Configuration in PWM Output Mode**



Note  Set by the TT0IOC2.TT0EES1 and TT0IOC2.TT0EES0 bits.
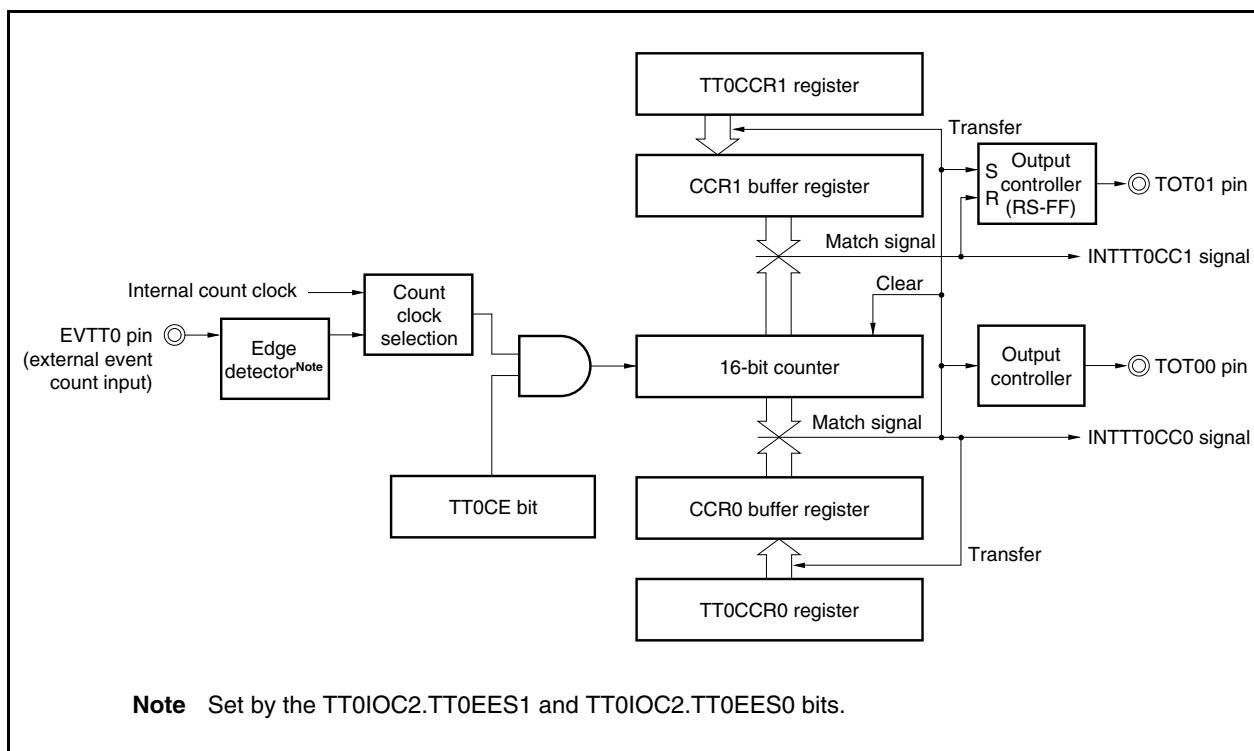
**Figure 9-30.  Basic Timing in PWM Output Mode**
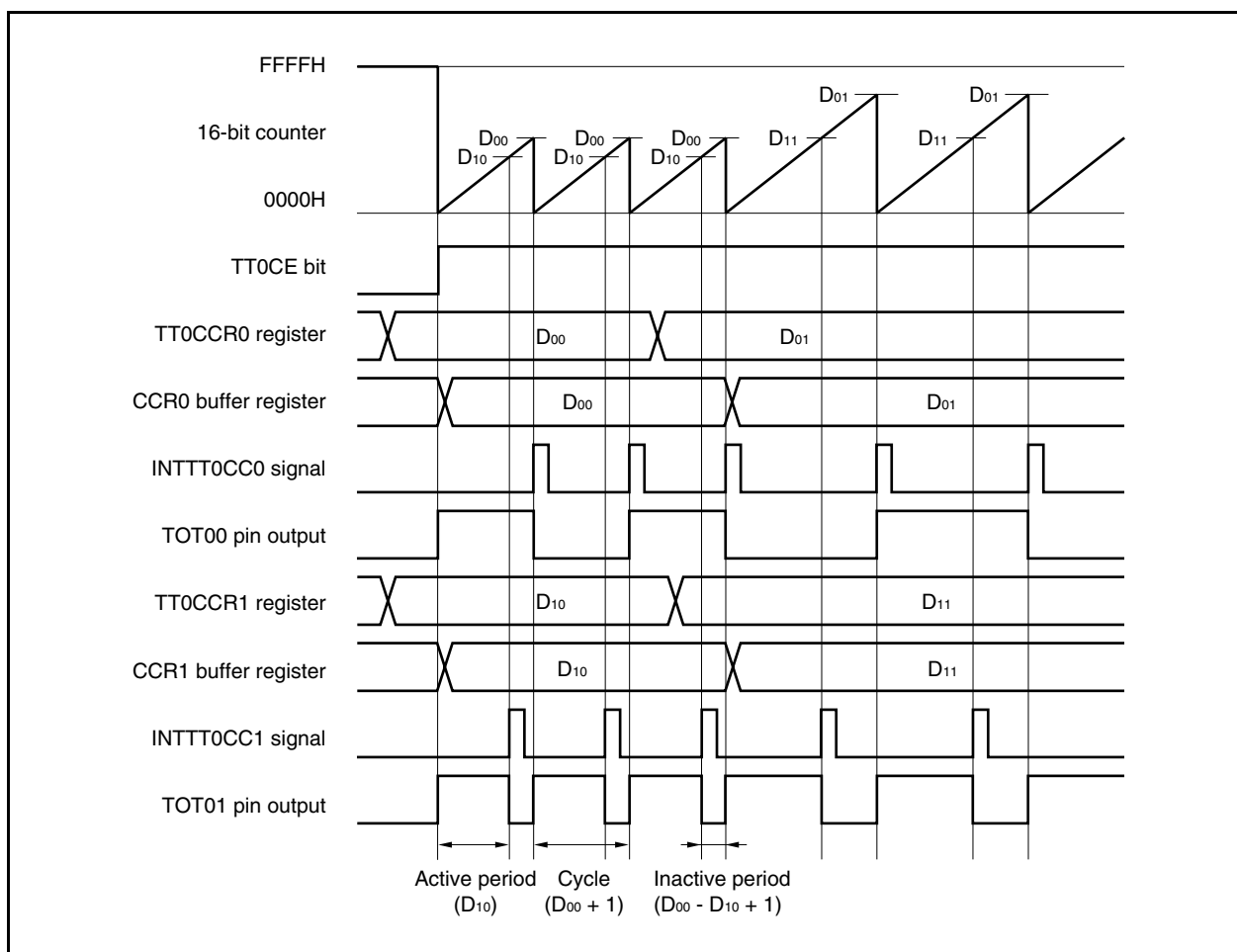


When the TT0CE bit is set to 1, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a PWM waveform from the TOT01 pin.

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

Active level width = (Set value of TT0CCR1 register) $\times$ Count clock cycle

Cycle = (Set value of TT0CCR0 register + 1) $\times$ Count clock cycle

Duty factor = (Set value of TT0CCR1 register)/(Set value of TT0CCR0 register + 1)

The PWM waveform can be changed by rewriting the TT0CCRn register while the counter is operating.  The newly written value is reflected when the count value of the 16-bit counter matches the value of the CCR0 buffer register and the 16-bit counter is cleared to 0000H.

The compare match interrupt request signal (INTTT0CC0) is generated the next time the 16-bit counter counts after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H.  The compare match interrupt request signal (INTTT0CC1) is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The value set to the TT0CCRn register is transferred to the CCRn buffer register when the count value of the 16-bit counter matches the value of the CCRn buffer register and the 16-bit counter is cleared to 0000H.

**Remark**   n = 0, 1

**Figure 9-31.  Setting of Registers in PWM Output Mode (1/2)**

**(a)  TMT0 control register 0 (TT0CTL0)**

|  | TT0CE |  |  |  |  | TT0CKS2 | TT0CKS1 | TT0CKS0 |
|---|---|---|---|---|---|---|---|---|
| TT0CTL0 | 0/1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 |

Select count clock**Note**

0: Stops counting
1: Enables counting

**Note**   The setting is invalid when the TT0CTL1.TT0EEE bit = 1.

**(b)  TMT0 control register 1 (TT0CTL1)**

|  |  | TT0EST | TT0EEE |  | TT0MD3 | TT0MD2 | TT0MD1 | TT0MD0 |
|---|---|---|---|---|---|---|---|---|
| TT0CTL1 | 0 | 0 | 0/1 | 0 | 0 | 1 | 0 | 0 |

0, 1, 0, 0:
PWM output mode

0:  Operates on count clock
    selected by TT0CKS0 to
    TT0CKS2 bits
1:  Counts with external event
    count input signal

**(c)  TMT0 I/O control register 0 (TT0IOC0)**

|  |  |  |  | TT0OL1 | TT0OE1 | TT0OL0 | TT0OE0 |
|---|---|---|---|---|---|---|---|
| TT0IOC0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |

0: Disables TOT00 pin output
1: Enables TOT00 pin output

Setting of TOT00 pin output
level before count operation
0: Low level
1: High level

0: Disables TOT01 pin output
1: Enables TOT01 pin output

Setting of TOT01 pin output
level before count operation
0: Low level
1: High level

- When TT0OL1 bit = 0

16-bit counter

TOT01 pin output

- When TT0OL1 bit = 1

16-bit counter

TOT01 pin output

**Figure 9-31.  Register Setting in PWM Output Mode (2/2)**

**(d)  TMT0 I/O control register 2 (TT0IOC2)**

|  | | | | TT0EES1 | TT0EES0 | TT0ETS1 | TT0ETS0 |
|---|---|---|---|---|---|---|---|
| TT0IOC2 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0 | 0 |

Select valid edge
of external event
count input (EVTT0 pin).

**(e)  TMT0 counter read buffer register (TT0CNT)**

The value of the 16-bit counter can be read by reading the TT0CNT register.

**(f)  TMT0 capture/compare registers 0 and 1 (TT0CCR0 and TT0CCR1)**

If $D_0$ is set to the TT0CCR0 register and $D_1$ to the TT0CCR1 register, the cycle and active level of the PWM waveform are as follows.

Cycle = $(D_0 + 1) \times$ Count clock cycle
Active level width = $D_1 \times$ Count clock cycle

**Remark**   TMT0 control register 2 (TT0CTL2), TMT0 I/O control register 1 (TT0IOC1), TMT0 I/O control register 3 (TT0CTL3), TMT0 option register 0 (TT0OPT0), TMT0 option register 1 (TT0OPT1), and TMT0 counter write register (TT0TCW) are not used in the PWM output mode.

**(1) Operation flow in PWM output mode**

**Figure 9-32. Software Processing Flow in PWM Output Mode (1/2)**

**Figure 9-32.  Software Processing Flow in PWM Output Mode (2/2)**

<1> Count operation start flow

START

Register initial setting
TT0CTL0 register
(TT0CKS0 to TT0CKS2 bits)
TT0CTL1 register,
TT0IOC0 register,
TT0IOC2 register,
TT0CCR0 register,
TT0CCR1 register

Initial setting of these
registers is performed
before setting the
TT0CE bit to 1.

TT0CE bit = 1

The TT0CKS0 to
TT0CKS2 bits can be
set at the same time
as when counting is
enabled (TT0CE bit = 1).

<2> TT0CCR0, TT0CCR1 register
setting change flow (cycle only)

Setting of TT0CCR0 register

Writing the same value
(same as preset value of
the TT0CCR1 register)
to the TT0CCR1 register
is necessary only when
the set cycle is changed.

Setting of TT0CCR1 register

When the counter is
cleared after setting,
the value of the TT0CCRn
register is transferred to the
CCRn buffer register.

**Remark**   n = 0, 1

<3> TT0CCR0, TT0CCR1 register
setting change flow (duty only)

Setting of TT0CCR1 register

Only writing of the TT0CCR1
register must be performed
when the set duty factor is
changed.  When the counter is
cleared after setting, the
value of compare register n
is transferred to the CCRn
buffer register.

<4> TT0CCR0, TT0CCR1 register
setting change flow (cycle and duty)

Setting of TT0CCR0 register

When the counter is
cleared after setting,
the values of compare
register n are transferred
to the CCRn buffer register
in a batch.

Setting of TT0CCR1 register

<5> Count operation stop flow

TT0CE bit = 0

Counting is stopped.

STOP

**(2) PWM output mode operation timing**

**(a) Changing pulse width during operation**

To change the PWM waveform while the counter is operating, write the TT0CCR1 register last.

Rewrite the TT0CCRn register after writing the TT0CCR1 register after the INTTT0CC1 signal is detected.



To transfer data from the TT0CCRn register to the CCRn buffer register, the TT0CCR1 register must be written. To change both the cycle and active level of the PWM waveform at this time, first set the cycle to the TT0CCR0 register and then set the active level width to the TT0CCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TT0CCR0 register, and then write the same value (same as preset value of the TT0CCR1 register) to the TT0CCR1 register.

To change only the active level width (duty factor) of the PWM waveform, only the TT0CCR1 register has to be set.

After data is written to the TT0CCR1 register, the value written to the TT0CCRn register is transferred to the CCRn buffer register in synchronization with clearing of the 16-bit counter, and is used as the value compared with the 16-bit counter.

To write the TT0CCR0 or TT0CCR1 register again after writing the TT0CCR1 register once, do so after the INTTT0CC0 signal is generated. Otherwise, the value of the CCRn buffer register may become undefined because the timing of transferring data from the TT0CCRn register to the CCRn buffer register conflicts with writing the TT0CCRn register.

**Remark**   n = 0, 1

**(b) 0%/100% output of PWM waveform**

To output a 0% waveform, set the TT0CCR1 register to 0000H. The 16-bit counter is cleared to 0000H and the INTTT0CC0 and INTTT0CC1 signals are generated after a match between the count value of the 16-bit counter and the value of the CCR0 buffer register.



**Note** The timing is actually delayed by one operating clock ($f_{XX}$).

To output a 100% waveform, set a value of (set value of TT0CCR0 register + 1) to the TT0CCR1 register. If the set value of the TT0CCR0 register is FFFFH, 100% output cannot be produced.



**Note** The timing is actually delayed by one operating clock ($f_{XX}$).

**(c) Generation timing of compare match interrupt request signal (INTTT0CC1)**

The timing of generation of the INTTT0CC1 signal in the PWM output mode differs from the timing of INTTT0CC1 signals in other modes; the INTTT0CC1 signal is generated when the count value of the 16-bit counter matches the value of the TT0CCR1 register.



| | | |
|---|---|---|
| Count clock | | |
| 16-bit counter | $D_1 - 2$   $D_1 - 1$   $D_1$   $D_1 + 1$   $D_1 + 2$ | |
| TT0CCR1 register | $D_1$ | |
| TOT01 pin output | **Note** | |
| INTTT0CC1 signal | **Note** | |

**Note**  The timing is actually delayed by one operating clock ($f_{XX}$).
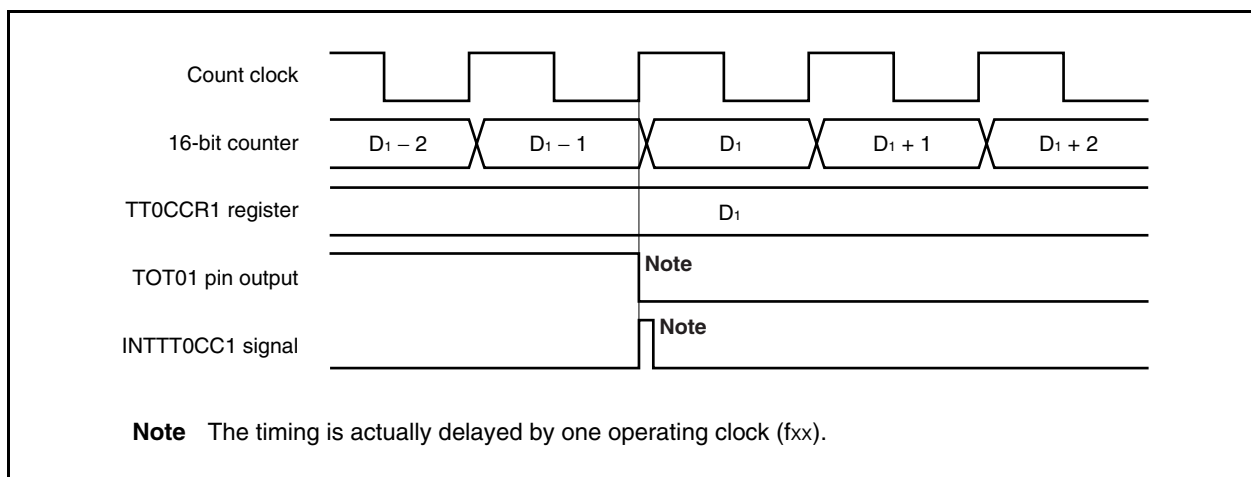
Usually, the INTTT0CC1 signal is generated in synchronization with the next count-up after the count value of the 16-bit counter matches the value of the TT0CCR1 register.

In the PWM output mode, however, it is generated one clock earlier. This is because the timing is changed to match the timing at which the output signal of the TOT01 pin changes.

### 9.6.6  Free-running timer mode (TT0MD3 to TT0MD0 bits = 0101)

In the free-running timer mode, 16-bit timer/event counter T starts counting when the TT0CTL0.TT0CE bit is set to 1. At this time, the TT0CCR0 and TT0CCR1 registers can be used as compare registers or capture registers, depending on the setting of the TT0OPT0.TT0CCS0 and TT0OPT0.TT0CCS1 bits.

**Figure 9-33.  Configuration in Free-Running Timer Mode**



Notes **1.** Because the capture trigger input pins (TIT00, TIT01) and timer output pins (TOT00, TOT01) are the same alternate-function pins, two functions cannot be used at the same time.
    **2.** Set by the TT0IOC2.TT0EES1 and TT0IOC2.TT0EES0 bits.
    **3.** Set by the TT0IOC1.TT0IS1 and TT0IOC1.TT0IS0 bits.
    **4.** Set by the TT0IOC1.TT0IS3 and TT0IOC1.TT0IS2 bits.
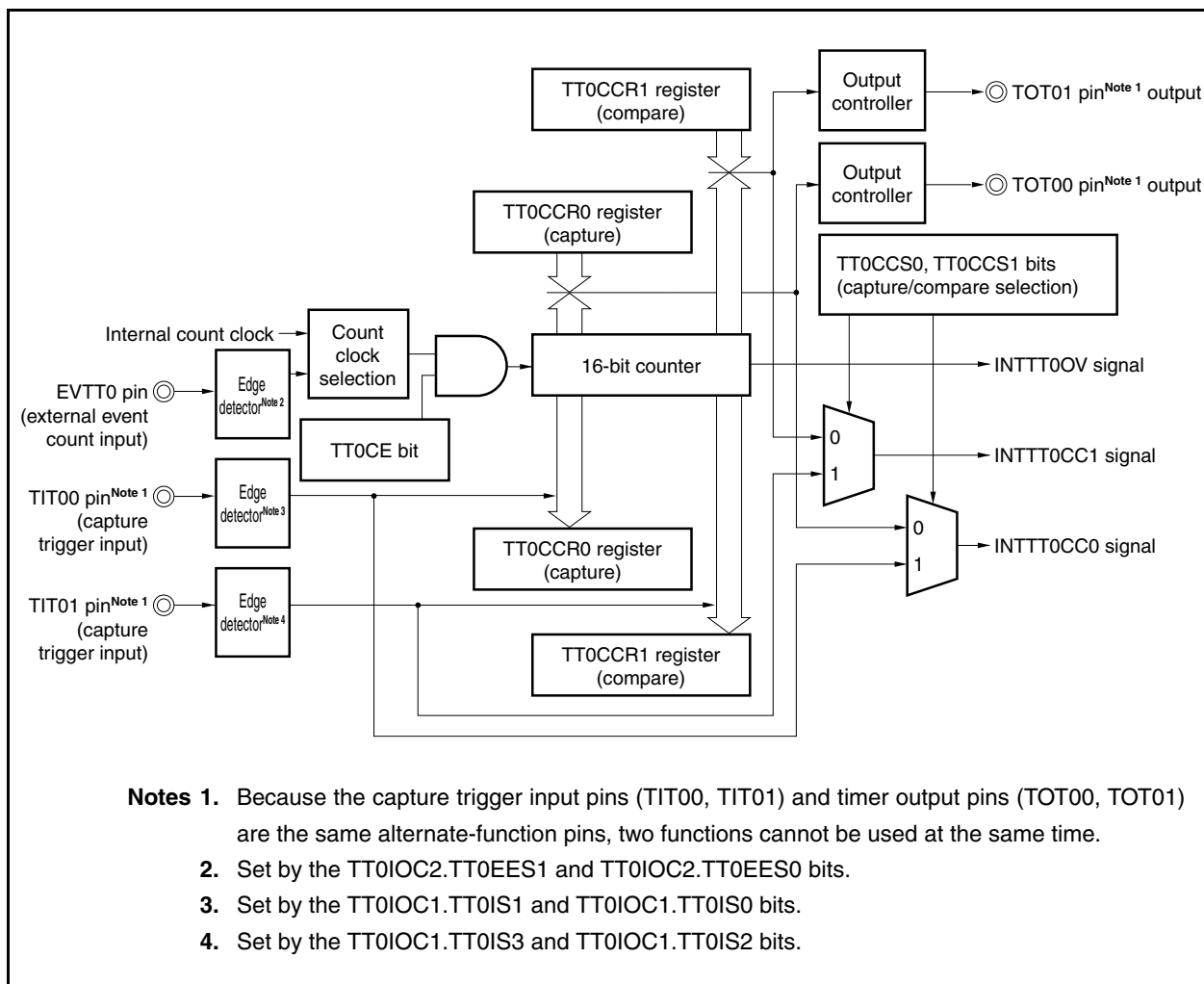
- Compare operation

When the TT0CE bit is set to 1, 16-bit timer/event counter T starts counting, and the output signal of the TOT0n pin is inverted. When the count value of the 16-bit counter later matches the set value of the TT0CCRn register, a compare match interrupt request signal (INTTT0CCn) is generated, and the output signal of the TOT0n pin is inverted.

The 16-bit counter continues counting in synchronization with the count clock. When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTT0OV) at the next clock, is cleared to 0000H, and continues counting. At this time, the overflow flag (TT0OPT0.TT0OVF bit) is also set to 1. Confirm that the overflow flag is set to 1 and then clear it to 0 by executing the CLR instruction via software.

The TT0CCRn register can be rewritten while the counter is operating. If it is rewritten, the new value is reflected at that time by anytime write, and compared with the count value.

**Figure 9-34.  Basic Timing in Free-Running Timer Mode (Compare Function)**
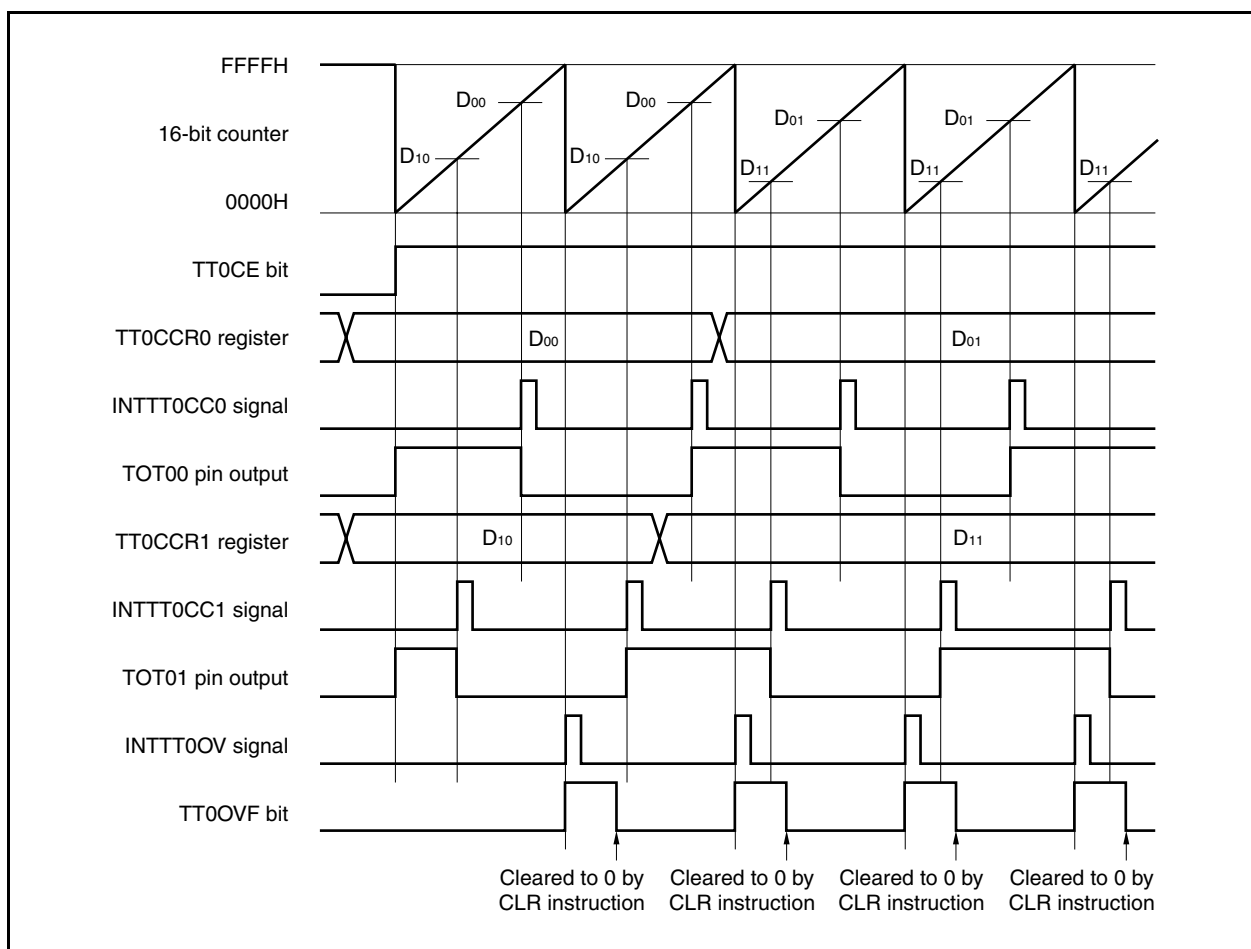
- Capture operation

When the TT0CE bit is set to 1, the 16-bit counter starts counting.  When the valid edge input to the TIT0n pin is detected, the count value of the 16-bit counter is stored in the TT0CCRn register, and a capture interrupt request signal (INTTT0CCn) is generated.

The 16-bit counter continues counting in synchronization with the count clock.  When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTT0OV) at the next clock, is cleared to 0000H, and continues counting.  At this time, the overflow flag (TT0OPT0.TT0OVF bit) is also set to 1.  Confirm that the overflow flag is set to 1 and then clear it to 0 by executing the CLR instruction via software.

**Figure 9-35.  Basic Timing in Free-Running Timer Mode (Capture Function)**
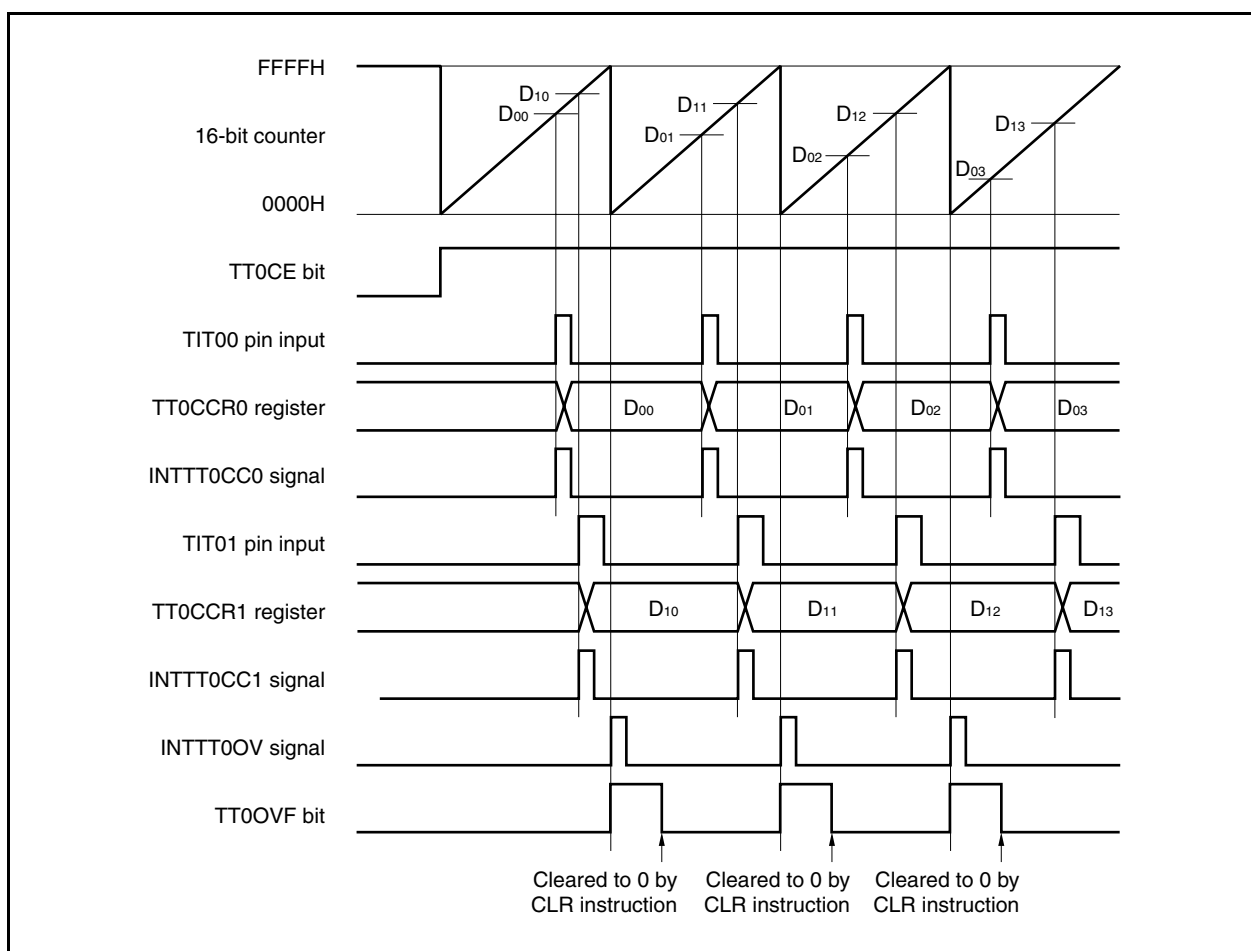
**Figure 9-36.  Register Setting in Free-Running Timer Mode (1/3)**



**(a)  TMT0 control register 0 (TT0CTL0)**

| | | | | | TT0CKS2 | TT0CKS1 | TT0CKS0 |

TT0CTL0

| TT0CE | | | | | | | |
| 0/1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 |

Select count clock**Note**

0: Stops counting
1: Enables counting

**Note**   The setting is invalid when the TT0CTL1.TT0EEE bit = 1

**(b)  TMT0 control register 1 (TT0CTL1)**

| TT0EST | TT0EEE | | TT0MD3 | TT0MD2 | TT0MD1 | TT0MD0 |

TT0CTL1

| 0 | 0 | 0/1 | 0 | 0 | 1 | 0 | 1 |

0, 1, 0, 1:
Free-running timer mode

0: Operates with count
   clock selected by
   TT0CKS0 to TT0CKS2 bits
1: Counts on external
   event count input signal

**(c)  TMT0 I/O control register 0 (TT0IOC0)**

| | | | | TT0OL1 | TT0OE1 | TT0OL0 | TT0OE0 |

TT0IOC0

| 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |

0: Disables TOT00 pin output
1: Enables TOT00 pin output

Setting of TOT00 pin output
level before count operation
0: Low level
1: High level

0: Disables TOT01 pin output
1: Enables TOT01 pin output

Setting of TOT01 pin output
level before count operation
0: Low level
1: High level

**Figure 9-36.  Register Setting in Free-Running Timer Mode (2/3)**



**(d)  TMT0 I/O control register 1 (TT0IOC1)**

|  | | | | TT0IS3 | TT0IS2 | TT0IS1 | TT0IS0 |
|---|---|---|---|---|---|---|---|
| TT0IOC1 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |

Select valid edge of TIT00 pin input

Select valid edge of TIT01 pin input

**(e)  TMT0 I/O control register 2 (TT0IOC2)**

|  | | | | TT0EES1 | TT0EES0 | TT0ETS1 | TT0ETS0 |
|---|---|---|---|---|---|---|---|
| TT0IOC2 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0 | 0 |

Select valid edge of external event count input (EVTT0 pin)

**(f)  TMT0 option register 0 (TT0OPT0)**

|  | | TT0CCS1 | TT0CCS0 | | | | TT0OVF |
|---|---|---|---|---|---|---|---|
| TT0OPT0 | 0 | 0 | 0/1 | 0/1 | 0 | 0 | 0 | 0/1 |

Overflow flag

Specifies if TT0CCR0 register functions as capture or compare register
0: Compare register
1: Capture register

Specifies if TT0CCR1 register functions as capture or compare register
0: Compare register
1: Capture register

**(g)  TMT0 counter read buffer register (TT0CNT)**

The value of the 16-bit counter can be read by reading the TT0CNT register.

**Figure 9-36.　Register Setting in Free-Running Timer Mode (3/3)**

**(h)　TMT0 capture/compare registers 0 and 1 (TT0CCR0 and TT0CCR1)**

These registers function as capture registers or compare registers depending on the setting of the TT0OPT0.TT0CCSn bit.

When the registers function as capture registers, they store the count value of the 16-bit counter when the valid edge input to the TIT0n pin is detected.

When the registers function as compare registers and when $D_a$ is set to the TT0CCRn register, the INTTT0CCn signal is generated when the counter reaches ($D_a$ + 1), and the output signals of the TOT00 and TOT01 pins are inverted.

**Remark**　n = 0, 1

**(1)  Operation flow in free-running timer mode**

**(a)  When using capture/compare register as compare register**

**Figure 9-37.  Software Processing Flow in Free-Running Timer Mode (Compare Function) (1/2)**
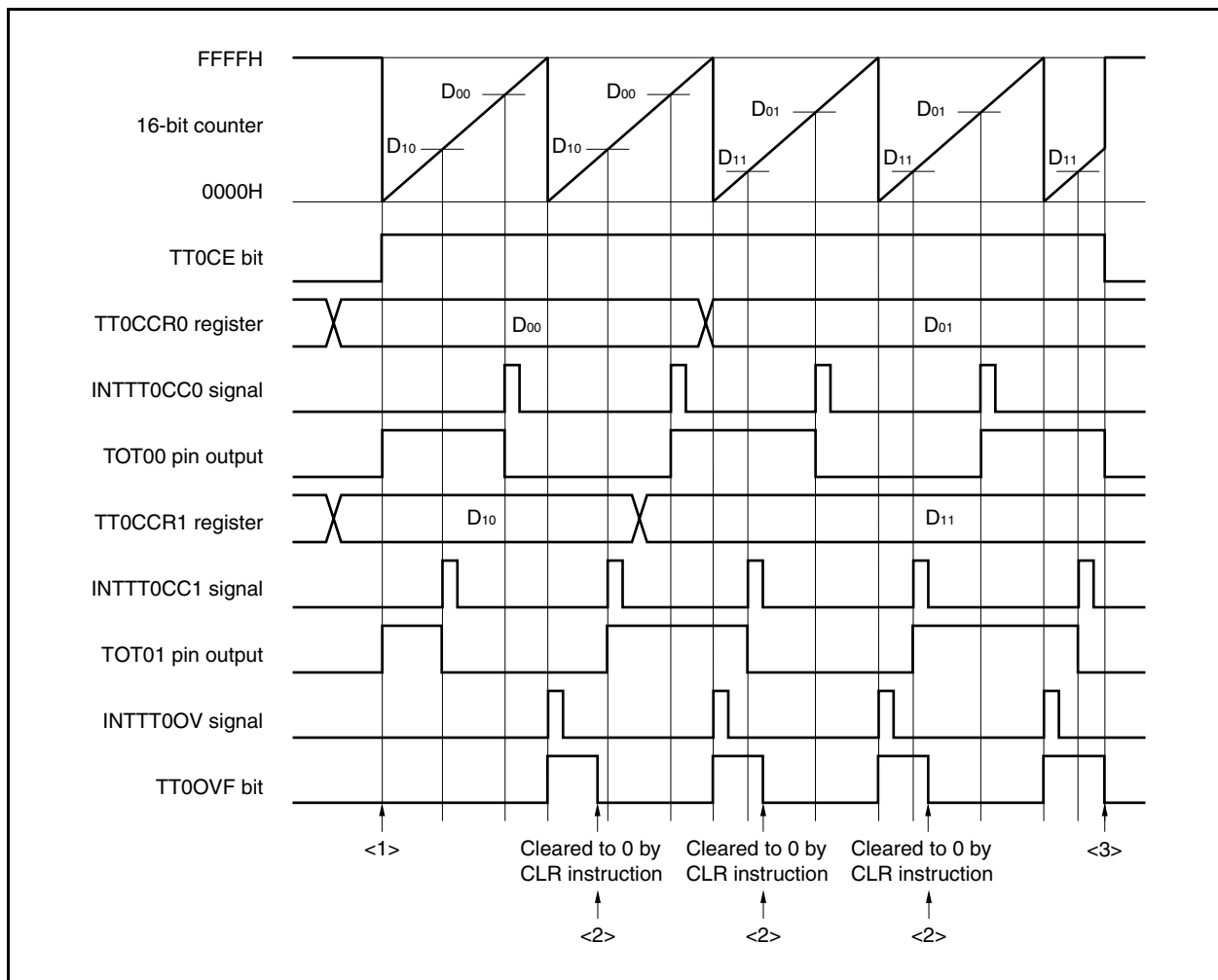
**Figure 9-37.  Software Processing Flow in Free-Running Timer Mode (Compare Function) (2/2)**
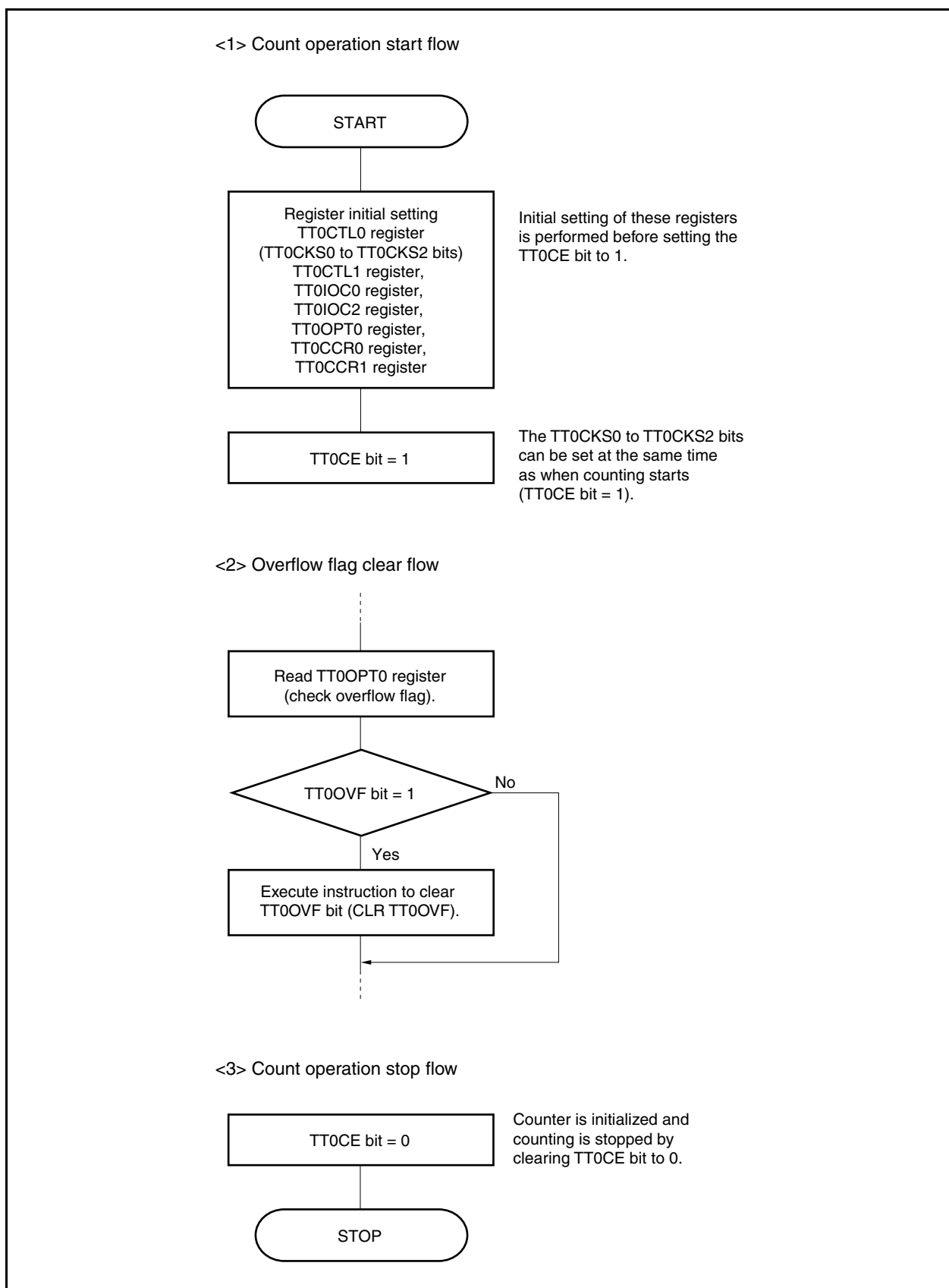
**(b) When using capture/compare register as capture register**

**Figure 9-38. Software Processing Flow in Free-Running Timer Mode (Capture Function) (1/2)**
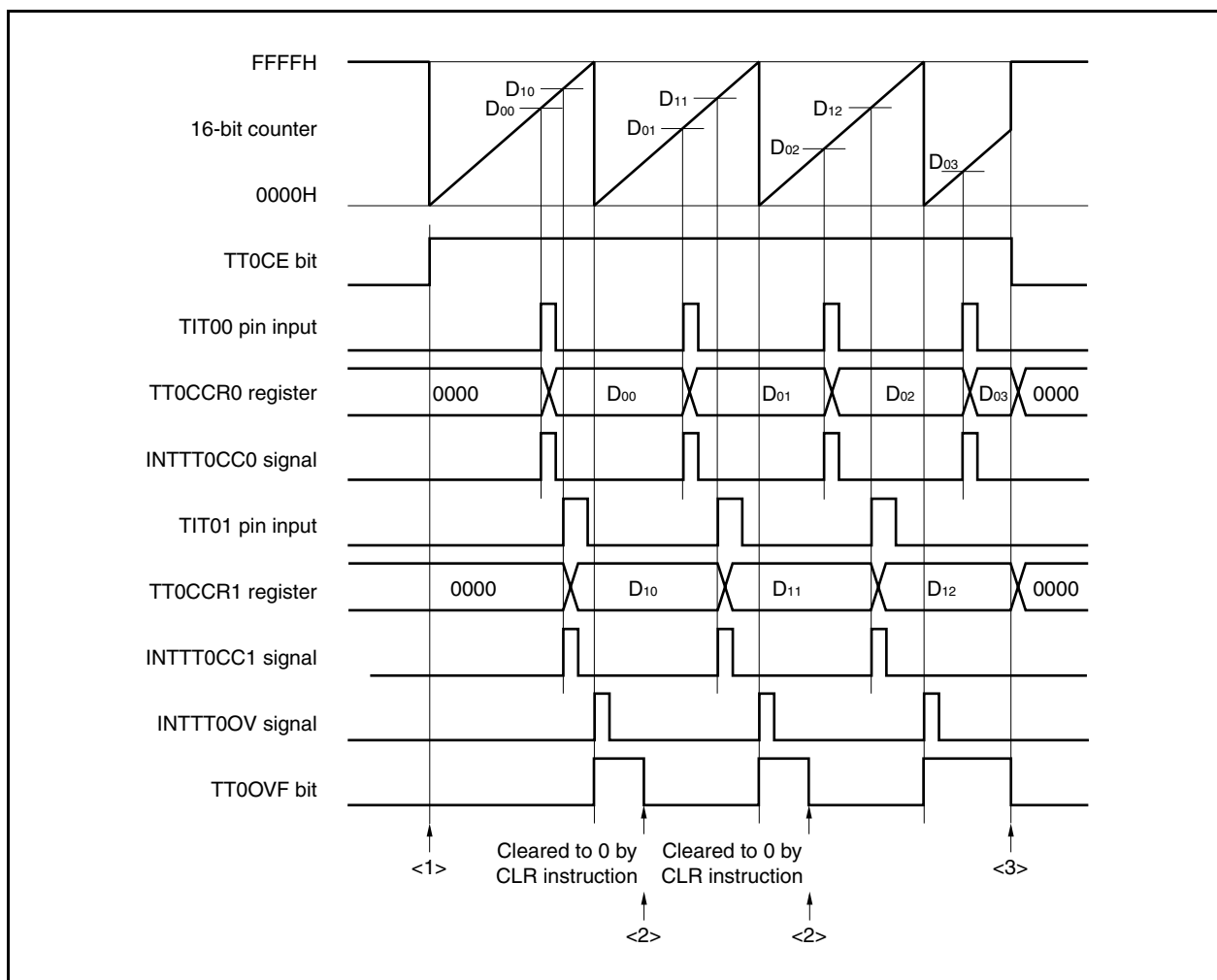
**Figure 9-38.  Software Processing Flow in Free-Running Timer Mode (Capture Function) (2/2)**

<1> Count operation start flow

```
          ┌──────────────┐
          │    START     │
          └──────────────┘
                 │
    ┌─────────────────────────────┐     Initial setting of these registers
    │  Register initial setting    │     is performed before setting the
    │     TT0CTL0 register         │     TT0CE bit to 1.
    │  (TT0CKS0 to TT0CKS2 bits)   │
    │     TT0CTL1 register,        │
    │     TT0IOC1 register,        │
    │     TT0OPT0 register         │
    └─────────────────────────────┘
                 │
    ┌─────────────────────────────┐     The TT0CKS0 to TT0CKS2 bits can
    │       TT0CE bit = 1          │     be set at the same time as when counting
    └─────────────────────────────┘     starts (TT0CE bit = 1).
```

<2> Overflow flag clear flow

```
    ┌─────────────────────────────┐
    │   Read TT0OPT0 register      │
    │   (check overflow flag).     │
    └─────────────────────────────┘
                 │
            ◇─────────────◇          No
            │ TT0OVF bit = 1 │ ─────────────┐
            ◇─────────────◇              │
                 │ Yes                   │
    ┌─────────────────────────────┐      │
    │  Execute instruction to clear │      │
    │  TT0OVF bit (CLR TT0OVF).    │      │
    └─────────────────────────────┘      │
                 │◄─────────────────────┘
```

<3> Count operation stop flow

```
    ┌─────────────────────────────┐     Counter is initialized and
    │       TT0CE bit = 0          │     counting is stopped by
    └─────────────────────────────┘     clearing TT0CE bit to 0.
                 │
          ┌──────────────┐
          │    STOP      │
          └──────────────┘
```

**(2) Operation timing in free-running timer mode**

**(a) Interval operation with compare register**

When 16-bit timer/event counter T is used as an interval timer with the TT0CCRn register used as a compare register, software processing is necessary for setting a comparison value to generate the next interrupt request signal each time the INTTT0CCn signal has been detected.
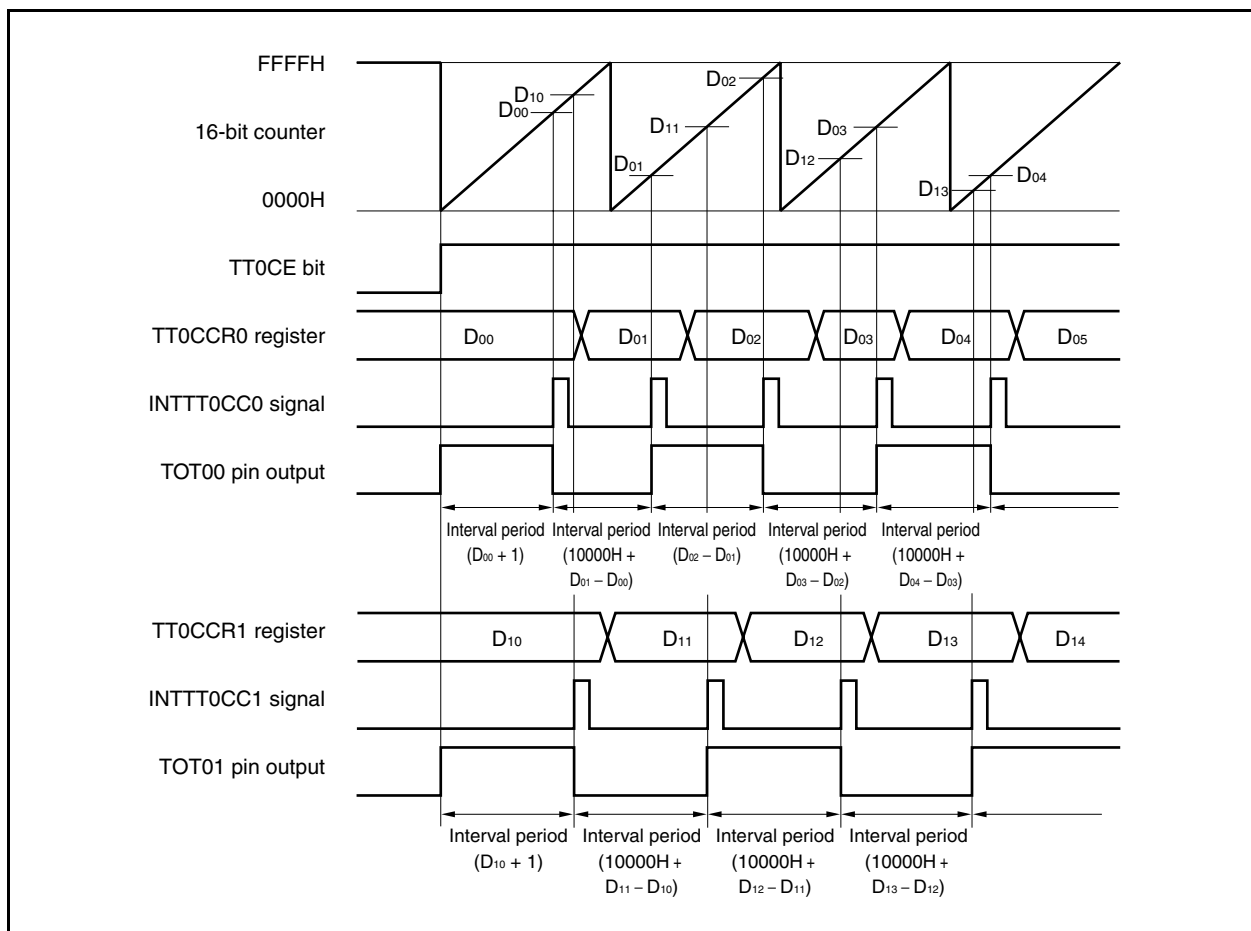


When performing an interval operation in the free-running timer mode, two intervals can be set with one channel.

To perform the interval operation, the value of the corresponding TT0CCRn register must be re-set in the interrupt servicing that is executed when the INTTT0CCn signal is detected.

The set value for re-setting the TT0CCRn register can be calculated by the following expression, where "$D_a$" is the interval period.

Compare register default value: $D_a - 1$

Value set to compare register second and subsequent time: Previous set value + $D_a$

(If the calculation result is greater than FFFFH, subtract 10000H from the result and set this value to the register.)

**(b) Pulse width measurement with capture register**

When pulse width measurement is performed with the TT0CCRn register used as a capture register, software processing is necessary for reading the capture register each time the INTTT0CCn signal has been detected and for calculating an interval.
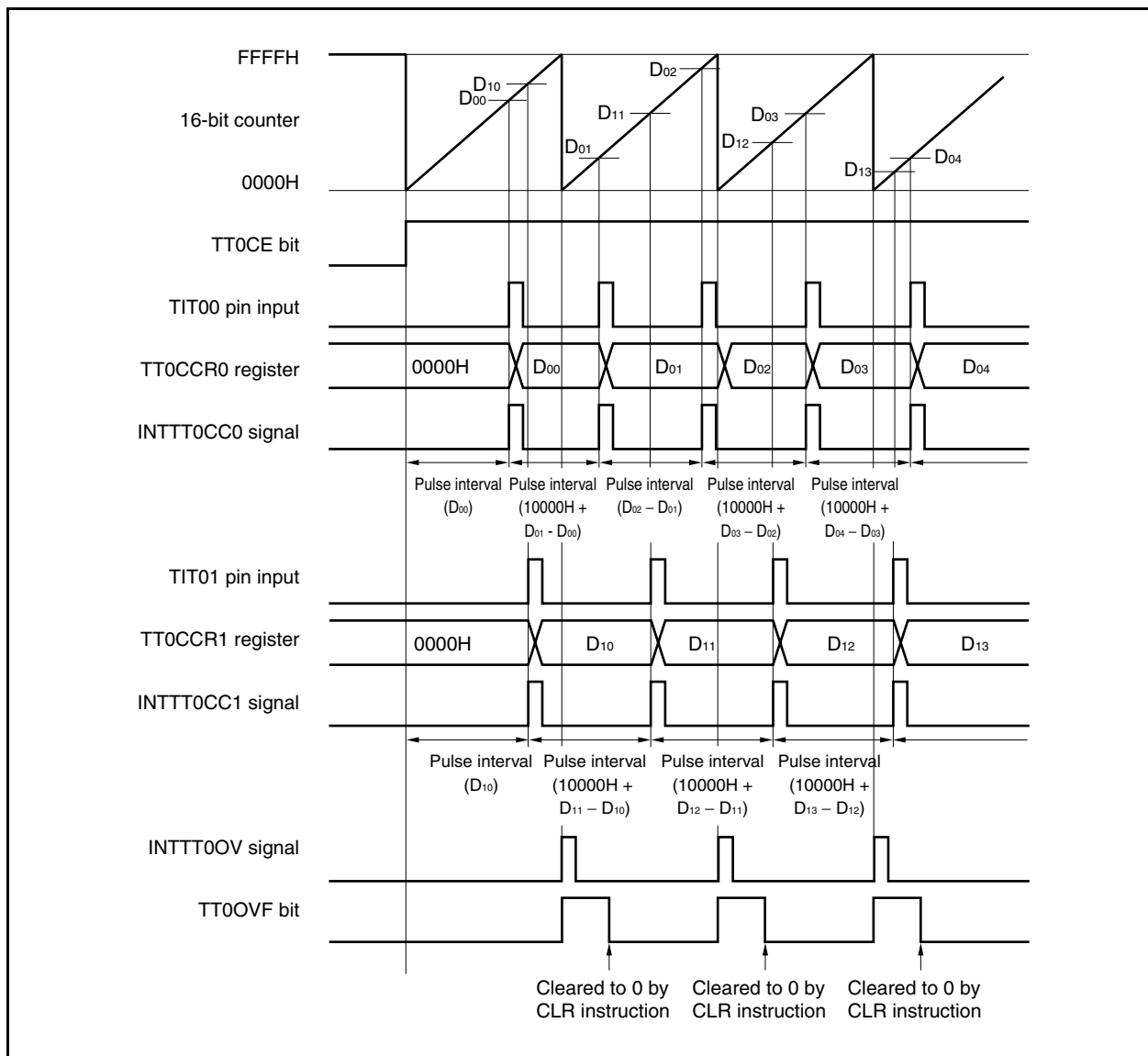


When executing pulse width measurement in the free-running timer mode, two pulse widths can be measured with one channel.

To measure a pulse width, the pulse width can be calculated by reading the value of the TT0CCRn register in synchronization with the INTTT0CCn signal, and calculating the difference between the read value and the previously read value.

**(c) Processing of overflow when two capture registers are used**

Care must be exercised in processing the overflow flag when two capture registers are used.  First, an example of incorrect processing is shown below.

**Example of incorrect processing when two capture registers are used**
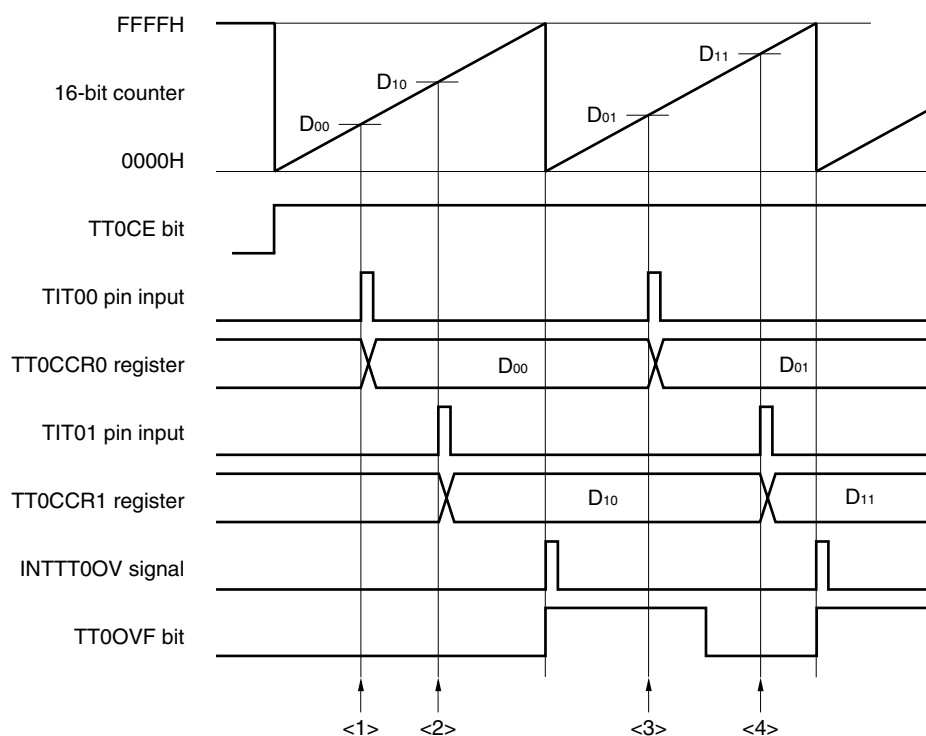


The following problem may occur when two pulse widths are measured in the free-running timer mode.

<1>  Read the TT0CCR0 register (setting of the default value of the TIT00 pin input).

<2>  Read the TT0CCR1 register (setting of the default value of the TIT01 pin input).

<3>  Read the TT0CCR0 register.

Read the overflow flag.  If the overflow flag is 1, clear it to 0.

Because the overflow flag is 1, the pulse width can be calculated by $(10000H + D_{01} - D_{00})$.

<4>  Read the TT0CCR1 register.

Read the overflow flag.  Because the flag is cleared in <3>, 0 is read.

Because the overflow flag is 0, the pulse width can be calculated by $(D_{11} - D_{10})$ (incorrect).

When two capture registers are used, and if the overflow flag is cleared to 0 by one capture register, the other capture register may not obtain the correct pulse width.

Use software when using two capture registers.  An example of how to use software is shown below.

**Example when two capture registers are used (using overflow interrupt)**



**Note** The TT0OVF0 and TT0OVF1 flags are set on the internal RAM by software.

<1> Read the TT0CCR0 register (setting of the default value of the TIT00 pin input).

<2> Read the TT0CCR1 register (setting of the default value of the TIT01 pin input).

<3> An overflow occurs. Set the TT0OVF0 and TT0OVF1 flags to 1 in the overflow interrupt servicing, and clear the overflow flag to 0.

<4> Read the TT0CCR0 register.

Read the TT0OVF0 flag. If the TT0OVF0 flag is 1, clear it to 0.

Because the TT0OVF0 flag is 1, the pulse width can be calculated by $(10000H + D_{01} - D_{00})$.

<5> Read the TT0CCR1 register.

Read the TT0OVF1 flag. If the TT0OVF1 flag is 1, clear it to 0 (the TT0OVF0 flag is cleared in <4>, and the TT0OVF1 flag remains 1).

Because the TT0OVF1 flag is 1, the pulse width can be calculated by $(10000H + D_{11} - D_{10})$ (correct).

<6> Same as <3>

**Example when two capture registers are used (without using overflow interrupt)**
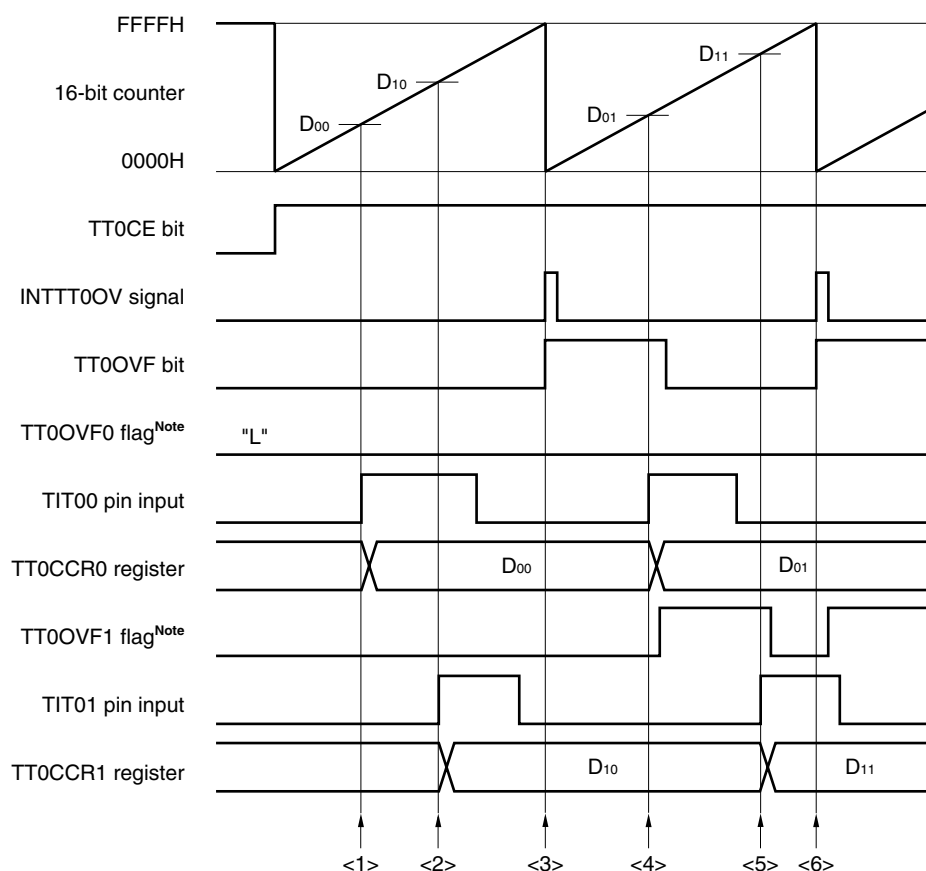


**Note**   The TT0OVF0 and TT0OVF1 flags are set on the internal RAM by software.

<1>  Read the TT0CCR0 register (setting of the default value of the TIT00 pin input).

<2>  Read the TT0CCR1 register (setting of the default value of the TIT01 pin input).

<3>  An overflow occurs.  Nothing is done by software.

<4>  Read the TT0CCR0 register.

Read the overflow flag.  If the overflow flag is 1, set only the TT0OVF1 flag to 1, and clear the overflow flag to 0.

Because the overflow flag is 1, the pulse width can be calculated by ($10000H + D_{01} - D_{00}$).

<5>  Read the TT0CCR1 register.

Read the overflow flag.  Because the overflow flag is cleared in <4>, 0 is read.

Read the TT0OVF1 flag.  If the TT0OVF1 flag is 1, clear it to 0.

Because the TT0OVF1 flag is 1, the pulse width can be calculated by ($10000H + D_{11} - D_{10}$) (correct).

<6>  Same as <3>

**(d) Processing of overflow if capture trigger interval is long**

If the pulse width is greater than one cycle of the 16-bit counter, care must be exercised because an overflow may occur more than once from the first capture trigger to the next.  First, an example of incorrect processing is shown below.

---

**Example of incorrect processing when capture trigger interval is long**
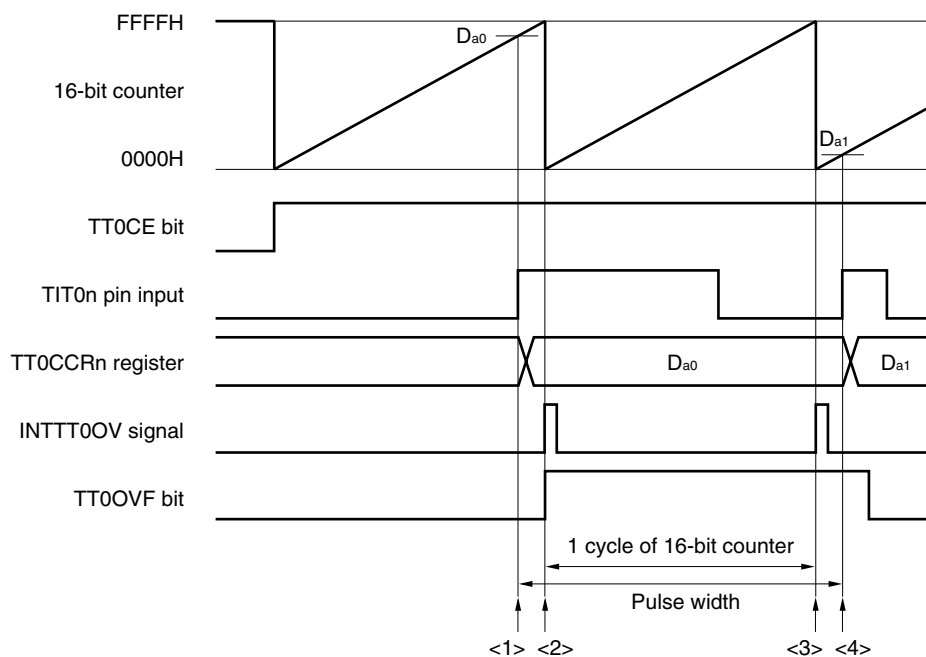


The following problem may occur when long pulse width is measured in the free-running timer mode.

<1> Read the TT0CCRn register (setting of the default value of the TIT0n pin input).

<2> An overflow occurs.  Nothing is done by software.

<3> An overflow occurs a second time.  Nothing is done by software.

<4> Read the TT0CCRn register.

Read the overflow flag.  If the overflow flag is 1, clear it to 0.

Because the overflow flag is 1, the pulse width can be calculated by $(10000H + D_{a1} - D_{a0})$ (incorrect).

Actually, the pulse width must be $(20000H + D_{a1} - D_{a0})$ because an overflow occurs twice.

**Remark**   $n = 0, 1$

---

If an overflow occurs twice or more when the capture trigger interval is long, the correct pulse width may not be obtained.

If the capture trigger interval is long, slow the count clock to lengthen one cycle of the 16-bit counter, or use software.  An example of how to use software is shown next.

**Example when capture trigger interval is long**
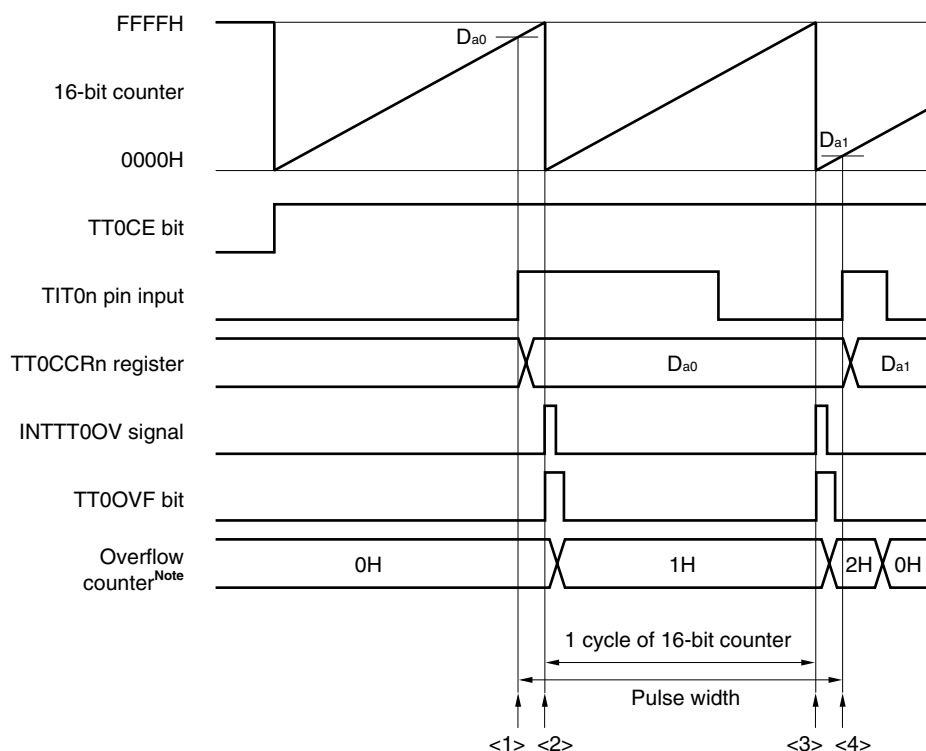


**Note** The overflow counter is set arbitrarily by software on the internal RAM.

<1> Read the TT0CCRn register (setting of the default value of the TIT0n pin input).

<2> An overflow occurs. Increment the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<3> An overflow occurs a second time. Increment the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<4> Read the TT0CCRn register.
Read the overflow counter.
$\rightarrow$ When the overflow counter is "N", the pulse width can be calculated by ($N \times 10000H + D_{a1} - D_{a0}$).
In this example, the pulse width is ($20000H + D_{a1} - D_{a0}$) because an overflow occurs twice.
Clear the overflow counter (0H).

**Remark** n = 0, 1

**(e) Clearing overflow flag**

The overflow flag can be cleared to 0 by clearing the TT0OVF bit to 0 with the CLR instruction after reading the TT0OVF bit when it is 1 and by writing 8-bit data (bit 0 is 0) to the TT0OPT0 register after reading the TT0OVF bit when it is 1.

### 9.6.7   Pulse width measurement mode (TT0MD3 to TT0MD0 bits = 0110)

In the pulse width measurement mode, 16-bit timer/event counter T starts counting when the TT0CTL0.TT0CE bit is set to 1.  Each time the valid edge input to the TIT0n pin has been detected, the count value of the 16-bit counter is stored in the TT0CCRn register, and the 16-bit counter is cleared to 0000H.

The interval of the valid edge can be measured by reading the TT0CCRn register after a capture interrupt request signal (INTTT0CCn) occurs.

As shown in Figure 9-39, select either the TIT00 or TIT01 pin as the capture trigger input pin and set the unused pins to "No edge detection" by using the TT0IOC1 register.

**Figure 9-39.  Configuration in Pulse Width Measurement Mode**



Notes **1.**  Set by the TT0IOC2.TT0EES1 and TT0IOC2.TT0EES0 bits.
　　　**2.**  Set by the TT0IOC1.TT0IS1 and TT0IOC1.TT0IS0 bits.
　　　**3.**  Set by the TT0IOC1.TT0IS3 and TT0IOC1.TT0IS2 bits.
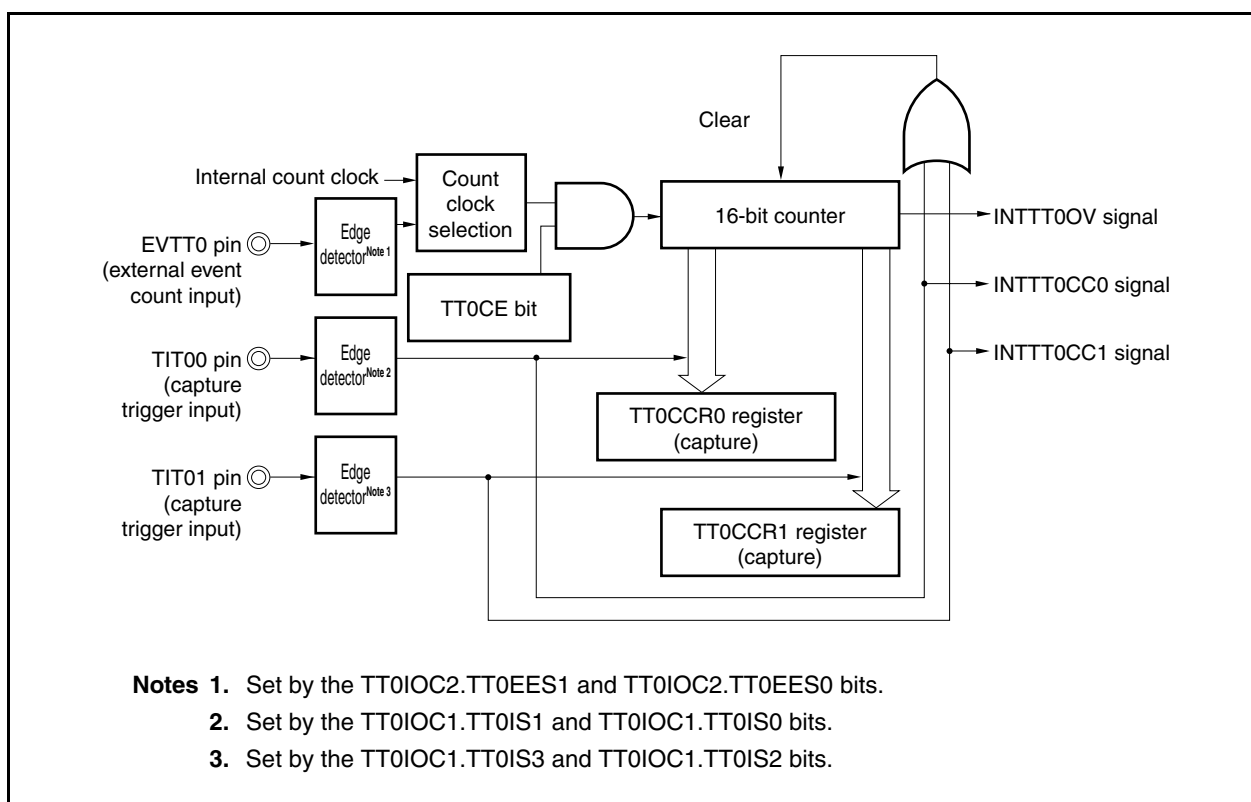
**Figure 9-40.  Basic Timing in Pulse Width Measurement Mode**
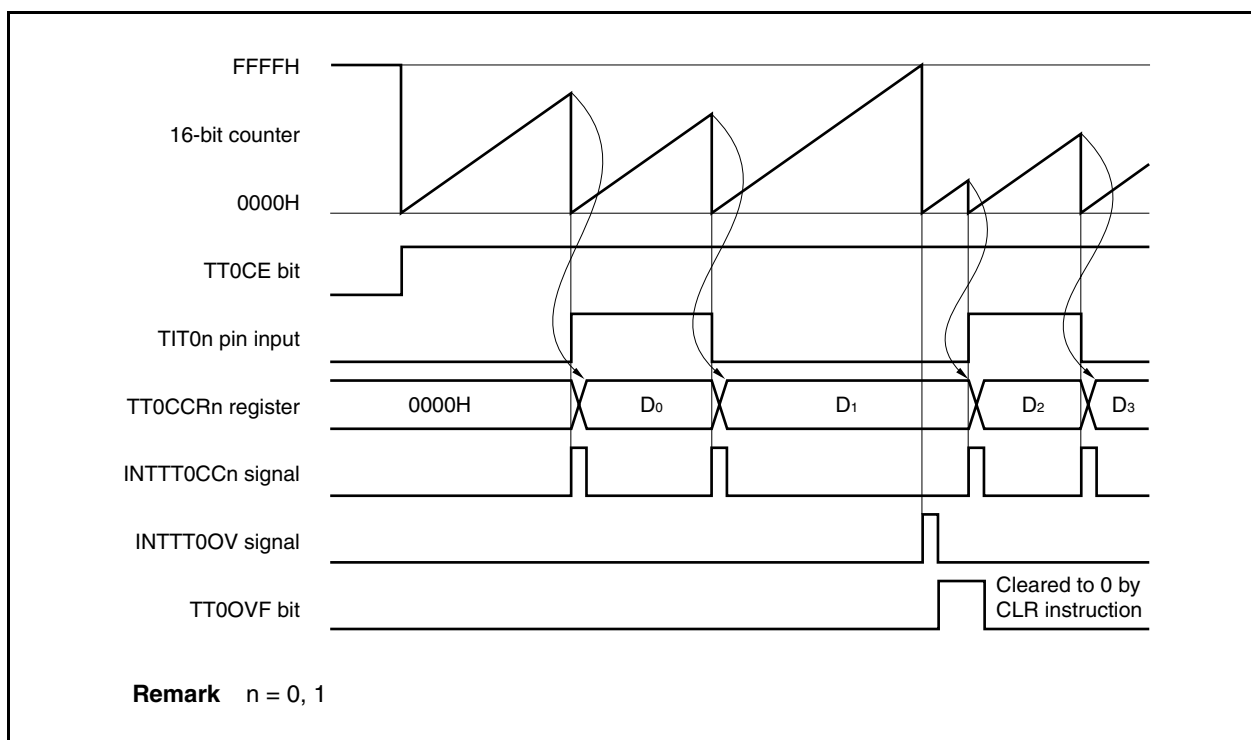


When the TT0CE bit is set to 1, the 16-bit counter starts counting.  When the valid edge input to the TIT0n pin is later detected, the count value of the 16-bit counter is stored in the TT0CCRn register, the 16-bit counter is cleared to 0000H, and a capture interrupt request signal (INTTT0CCn) is generated.

The pulse width is calculated as follows.

Pulse width = Captured value $\times$ Count clock cycle

If the valid edge is not input to the TIT0n pin even when the 16-bit counter counted up to FFFFH, an overflow interrupt request signal (INTTT0OV) is generated at the next count clock, and the counter is cleared to 0000H and continues counting.  At this time, the overflow flag (TT0OPT0.TT0OVF bit) is also set to 1.  Clear the overflow flag to 0 by executing the CLR instruction via software.

If the overflow flag is set to 1, the pulse width can be calculated as follows.

Pulse width = (10000H $\times$ TT0OVF bit set (1) count + Captured value) $\times$ Count clock cycle

**Remark**    n = 0, 1

**Figure 9-41. Register Setting in Pulse Width Measurement Mode (1/2)**



**(a) TMT0 control register 0 (TT0CTL0)**
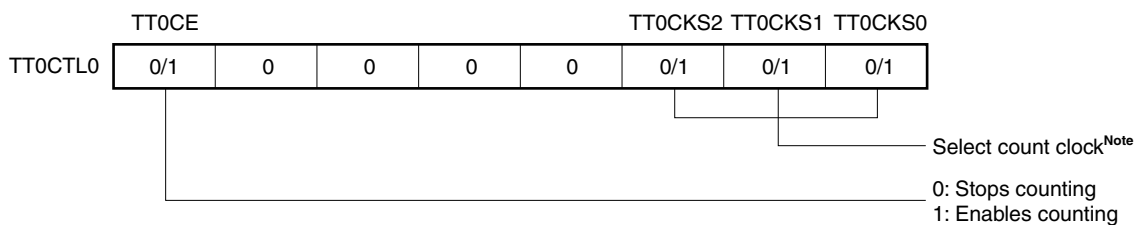
TT0CTL0 | TT0CE 0/1 | 0 | 0 | 0 | 0 | TT0CKS2 0/1 | TT0CKS1 0/1 | TT0CKS0 0/1

Select count clock**Note**

0: Stops counting
1: Enables counting

**Note** Setting is invalid when the TT0CTL1.TT0EEE bit = 1.

**(b) TMT0 control register 1 (TT0CTL1)**

TT0CTL1 | 0 | 0 | TT0EST TT0EEE 0/1 | 0 | TT0MD3 0 | TT0MD2 1 | TT0MD1 1 | TT0MD0 0

0, 1, 1, 0:
Pulse width measurement mode

0: Operates with count
clock selected by
TT0CKS0 to TT0CKS2 bits
1: Counts on external
event count input signal

**(c) TMT0 I/O control register 1 (TT0IOC1)**

TT0IOC1 | 0 | 0 | 0 | 0 | TT0IS3 0/1 | TT0IS2 0/1 | TT0IS1 0/1 | TT0IS0 0/1

Select valid edge
of TIT00 pin input

Select valid edge
of TIT01 pin input

**(d) TMT0 I/O control register 2 (TT0IOC2)**

TT0IOC2 | 0 | 0 | 0 | 0 | TT0EES1 0/1 | TT0EES0 0/1 | TT0ETS1 0 | TT0ETS0 0

Select valid edge of
external event count
input (EVTT0 pin)
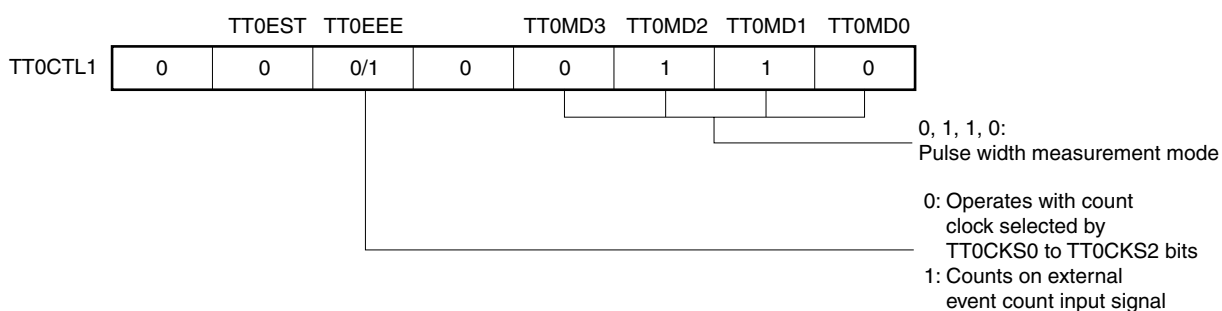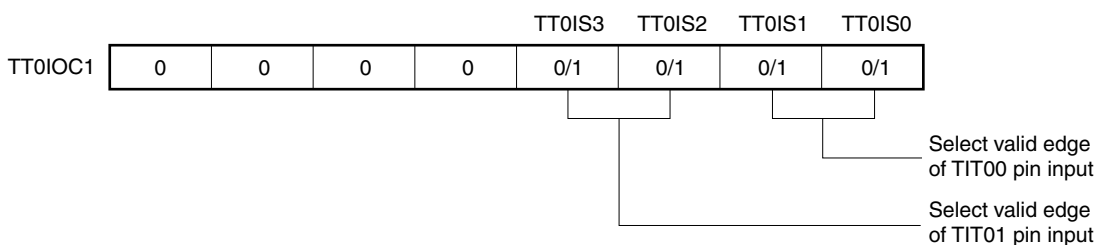
**Figure 9-41. Register Setting in Pulse Width Measurement Mode (2/2)**

**(e) TMT0 option register 0 (TT0OPT0)**

| | | TT0CCS1 | TT0CCS0 | | | | TT0OVF |
|---|---|---|---|---|---|---|---|

| TT0OPT0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 |
|---|---|---|---|---|---|---|---|---|

└─── Overflow flag
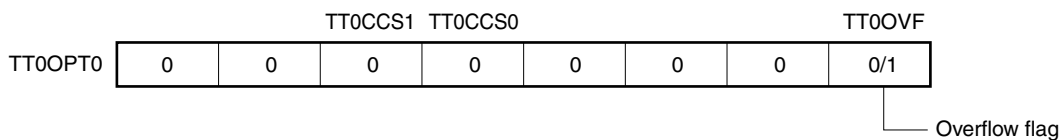
**(f) TMT0 counter read buffer register (TT0CNT)**

The value of the 16-bit counter can be read by reading the TT0CNT register.

**(g) TMT0 capture/compare registers 0 and 1 (TT0CCR0 and TT0CCR1)**

These registers store the count value of the 16-bit counter when the valid edge input to the TIT00 and TIT01 pins is detected.

**Remark**   TMT0 control register 2 (TT0CTL2), TMT0 I/O control register 0 (TT0IOC0), TMT0 I/O control register 3 (TT0IOC3), TMT0 option register 1 (TT0OPT1), and TMT0 counter write register (TT0TCW) are not used in the pulse width measurement mode.

**(1)  Operation flow in pulse width measurement mode**

**Figure 9-42.  Software Processing Flow in Pulse Width Measurement Mode**
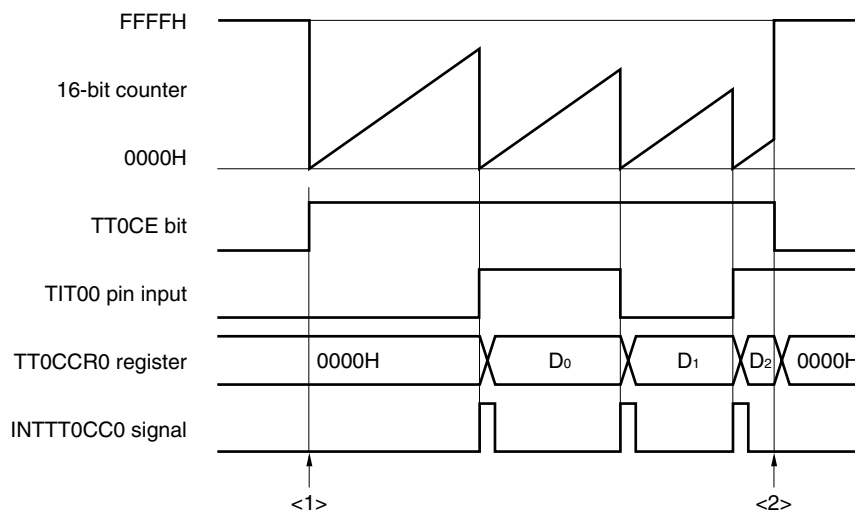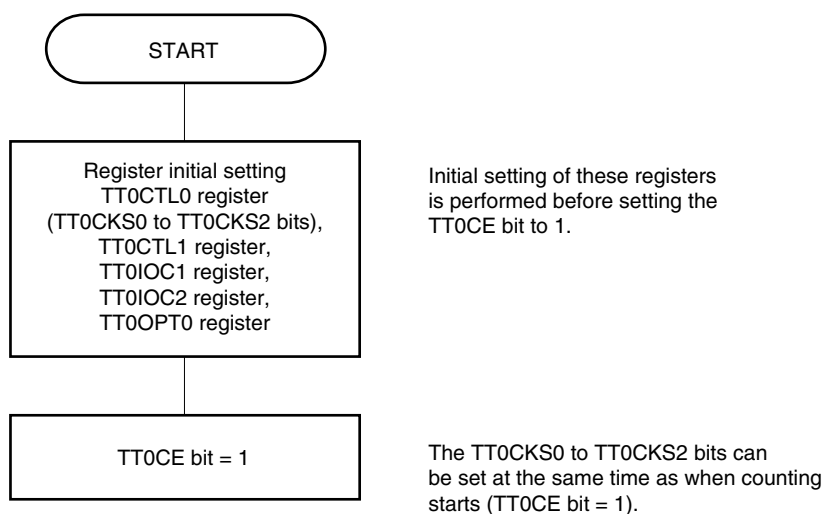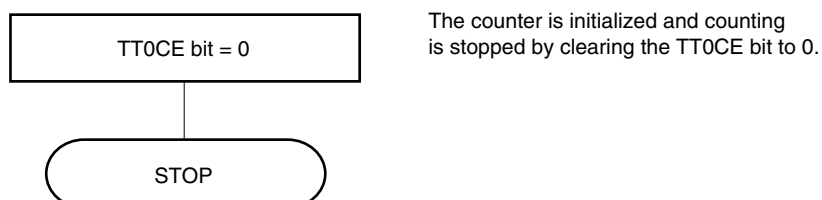
**(2) Operation timing in pulse width measurement mode**

**(a) Clearing overflow flag**

The overflow flag can be cleared to 0 by clearing the TT0OVF bit to 0 with the CLR instruction after reading the TT0OVF bit when it is 1 and by writing 8-bit data (bit 0 is 0) to the TT0OPT0 register after reading the TT0OVF bit when it is 1.

### 9.6.8  Triangular-wave PWM output mode (TT0MD3 to TT0MD0 bits = 0111)

In the triangular-wave PWM output mode, a triangular-wave PWM waveform is output from the TOT01 pin when the TT0CTL0.TT0CE bit is set to 1.

A PWM waveform that is inverted when the count value of the 16-bit counter matches the value of the CCR0 buffer register and when the 16-bit counter is set to 0000H is output from the TOT00 pin.

**Figure 9-43.  Configuration in Triangular-Wave PWM Output Mode**



**Note**  Set by the TT0IOC2.TT0EES1 and TT0IOC2.TT0EES0 bits.
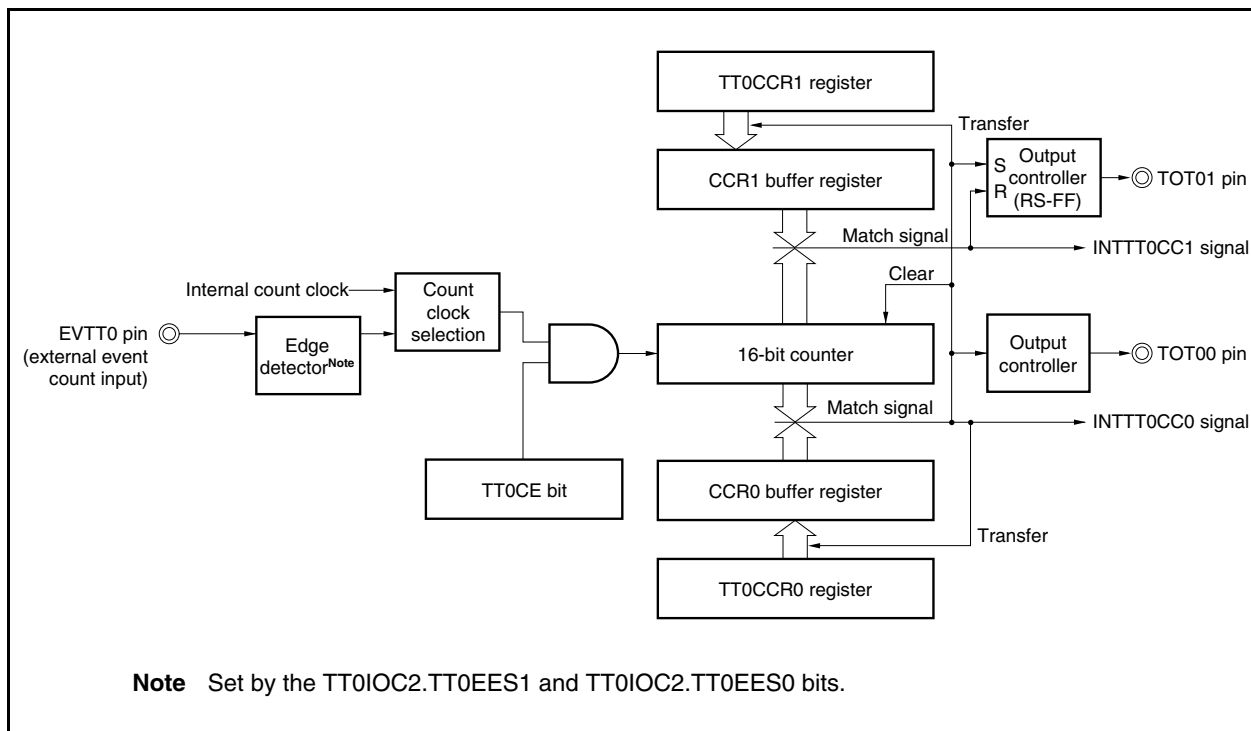
**Figure 9-44.  Basic Timing in Triangular-Wave PWM Output Mode**
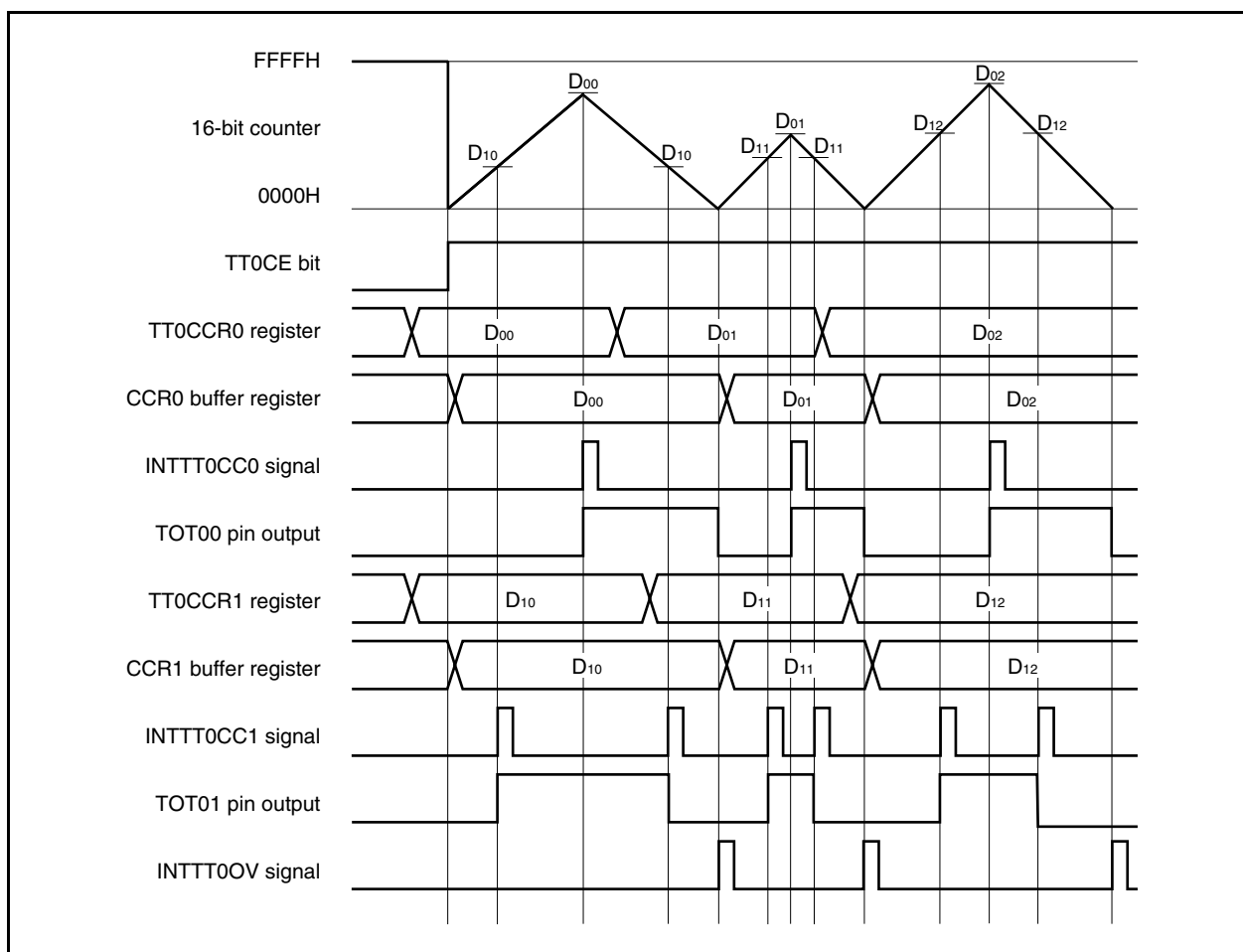


The 16-bit counter is cleared from FFFFH and 0000H and starts counting when the TT0CE bit is set to 1.  The triangular PWM waveform is output from the TOT01 pin.

In the triangular-wave PWM output mode, the counter counts up or down.  When the 16-bit counter reaches 0000H while it is counting down, an overflow interrupt request signal (INTTT0OV) is generated.  At this time, the TT0OPT0.TT0OVF bit is not set to 1.  If the count value of the 16-bit counter matches the value of the CCR0 buffer register while the counter is counting up, a compare match interrupt request signal (INTTT0CC0) is generated.

The counting direction is changed from up to down when the value of the 16-bit counter matches that of the CCR0 buffer register, and from down to up when the counter is cleared to 0000H.

The PWM waveform can be changed by rewriting the TT0CCRn register during operation.  To change the PWM waveform during operation, write the TT0CCR1 register last.

The cycle of the triangular PWM waveform is set by the TT0CCR0 register and its duty factor is set by the TT0CCR1 register.  Set a value to the TT0CCR0 register in a range of "0 ≤ TT0CCR0 ≤ FFFEH".  The rewritten value is reflected when the 16-bit counter reaches 0000H while it is counting down.

Even when changing only the cycle of the PWM waveform, first set a period to the TT0CCR0 register, and then write the same value (value same as that set to the TT0CCR1 register) to the TT0CCR1 register.

To transfer data from the TT0CCRn register to the CCRn buffer register, the data must be written to the TT0CCR1 register (n = 0, 1).

### 9.6.9  Encoder count function

The encoder count function includes an encoder compare mode (see **9.6.10  Encoder compare mode (TT0MD3 to TT0MD0 bits = 1000)**).

| Mode | TT0CCR0 Register | TT0CCR1 Register |
|---|---|---|
| Encoder compare mode | Compare only | Compare only |

**(1) Count-up/-down control**

Counting up or down by the 16-bit counter is controlled by the phase of input encoder signals (TENC00 and TENC01) and settings of the TT0CTL2.TT0UDS1 and TT0CTL2.TT0UDS0 bits.

When the encoder count function is used, the internal count clock and external event count input (EVTT0) cannot be used.  Set the TT0CTL0.TT0CKS2 to TT0CTL0.TT0CKS0 bits to 000 and the TT0CTL1.TT0EEE bit to 0.

**(2) Setting initial value of 16-bit counter**

The initial count value set to the TT0TCW register when the TT0CTL2.TT0ECC bit = 0 is transferred to the 16-bit counter immediately after the counter starts its operation (TT0CTL0.TT0CE bit = 0 $\rightarrow$ 1), and the counter starts the operation after it detects the valid edge of the encoder input signal (TENC00 or TENC01).

**(3) Basic operation**

The TT0CCRn register generates a compare match interrupt request signal (INTTT0CCn) when the count value of the 16-bit counter matches the value of the CCRn buffer register.

**(4) Clear operation**

The 16-bit counter is cleared when the following conditions are satisfied in the encoder compare mode.

- When the value of the 16-bit counter matches the value of the compare register (the TT0CTL2.TT0ECM1 and TT0CTL2.TT0ECM0 bits are set)
- When the edge of the encoder clear input signal (TECR0) is detected (the TT0ECS1 and TT0ECS0 bits are set when the TT0IOC3.TT0SCE bit = 0)
- When the clear level condition of the TENC00, TENC01, and TECR0 pins is detected (the TT0ZCL, TT0BCL, and TT0ACL bits are set when the TT0SCE bit = 1)

**Remark**   n = 0, 1

**(5)  Controlling bits of TT0CTL2 register**

The setting of the TT0CTL2 register in the encoder compare mode is shown below.

**Table 9-8.  Setting of TT0CTL2 Register**

| Mode | TT0UDS1, TT0UDS0 Bits (<1>) | TT0ECM1 Bit (<2>) | TT0ECM0 Bit (<2>) | TT0LDE Bit (<3>) | Counter Clear (Target Compare Register) | Transfer to Counter |
|---|---|---|---|---|---|---|
| Encoder compare mode | Can be set to 00, 01, 10, or 11. | 0 | 0 | 0 | – | – |
| | | | | 1 | | Possible |
| | | | 1 | 0 | TT0CCR0 | – |
| | | | | 1 | | Possible[Note] |
| | | 1 | 0 | Invalid | TT0CCR1 | – |
| | | | 1 | Invalid | TT0CCR0, TT0CCR1 | – |

**Note**   The counter can operate in a range from 0000H to the set value of the TT0CCR0 register.

**(a)  Outline of each bit**

<1>  The TT0UDS1 and TT0UDS0 bits identify the counting direction (up or down) of the 16-bit counter by the phase input from the encoder input pin (TENC00 or TENC01).

<2>  The TT0ECM1 and TT0ECM0 bits control clearing of the 16-bit counter when its count value matches the value of the CCR0 or CCR1 buffer register.

<3>  The TT0LDE bit controls a function to transfer the set value of the TT0CCR0 register to the 16-bit counter when the counter underflows.  The TT0LDE bit is valid only when the TT0ECM1 and TT0ECM0 bits are 00 and 01, respectively.  It is invalid when these bits are set to any other values.

**(b)  Detailed explanation of each bit**

<1>  TT0UDS1 and TT0UDS0 bits: Count-up-/-down selection

Whether the 16-bit counter is counting up or down is identified by the phase input from the TENC00 or TENC01 pin and depending on the settings of the TT0UDS1 and TT0UDS0 bits.  These bits are valid only in the encoder compare mode.

- When TT0UDS1 and TT0UDS0 bits = 00

| TENC00 Pin | TENC01 Pin | Count Operation |
|---|---|---|
| Rising edge | High level | Count down |
| Falling edge | | |
| Both edges | | |
| Rising edge | Low level | Count up |
| Falling edge | | |
| Both edges | | |

**Remark**   Detecting the edge of the TENC00 pin is specified by the TT0IOC3.TT0EIS1 and TT0EIS0 bits.

**Figure 9-45.  Operation Example (When Valid Edge of TENC00 Pin Is Specified to Be Rising Edge and No Edge Is Specified as Valid Edge of TENC01 Pin)**
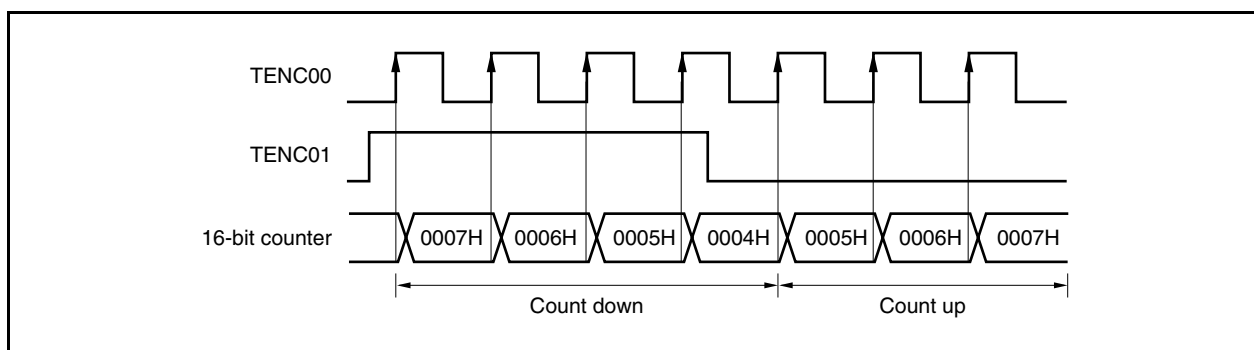
• When TT0UDS1 and TT0UDS0 bits = 01

| TENC00 Pin | TENC01 Pin | Count Operation |
|---|---|---|
| Low level | Rising edge | Count down |
| | Falling edge | |
| | Both edges | |
| High level | Rising edge | |
| | Falling edge | |
| | Both edges | |
| Rising edge | High level | Count up |
| Falling edge | | |
| Both edges | | |
| Rising edge | Low level | |
| Falling edge | | |
| Both edges | | |
| Simultaneous input to TENC00 and TENC01 pins | | Counter does not perform count operation but holds value immediately before. |

Remark    Detecting the edges of the TENC00 and TENC01 pins is specified by the TT0IOC3.TT0EIS1 and TT0IOC3.TT0EIS0 bits.

**Figure 9-46. Operation Example (When Rising Edge Is Specified as Valid Edges of TENC00 and TENC01 Pins)**
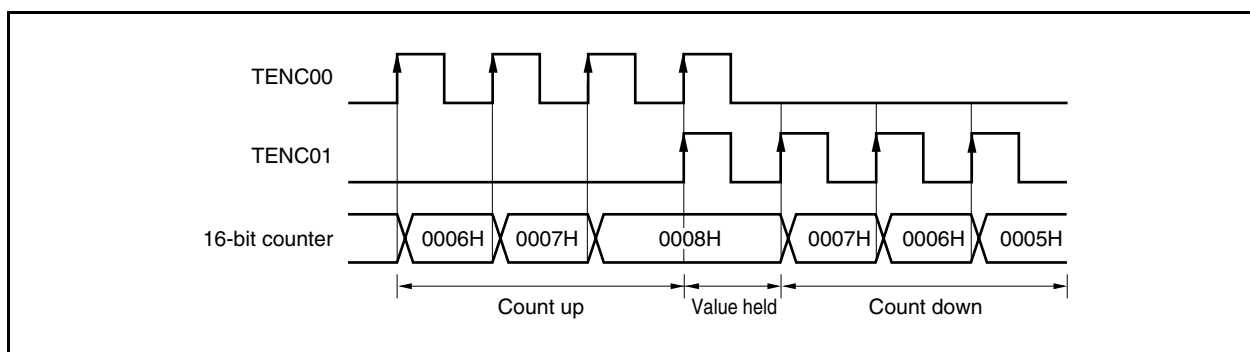
- When TT0UDS1 and TT0UDS0 bits = 10

| TENC00 Pin | TENC01 Pin | Count Operation |
|---|---|---|
| Low level | Falling edge | Counter does not perform count operation but holds value immediately before. |
| Rising edge | Low level | Count down |
| High level | Rising edge | Counter does not perform count operation but holds value immediately before. |
| Falling edge | High level | |
| Rising edge | | |
| High level | Falling edge | |
| Falling edge | Low level | Count up |
| Low level | Rising edge | Counter does not perform count operation but holds value immediately before. |
| Rising edge | | |
| Falling edge | | |
| Rising edge | Falling edge | Count down |
| Falling edge | | Count up |

**Caution**    **Specification of the valid edges of the TENC00 and TENC01 pins is invalid.**

**Figure 9-47. Operation Example (Count Operation When Valid Edges of TENC00 and TENC01 Pins do not Overlap)**
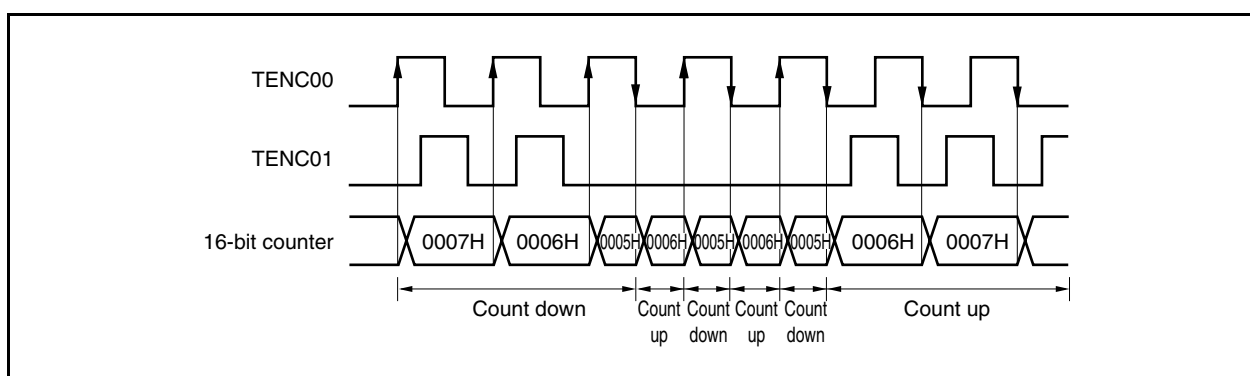


**Figure 9-48. Operation Example (Count Operation When Valid Edges of TENC00 and TENC01 Pins Overlap)**
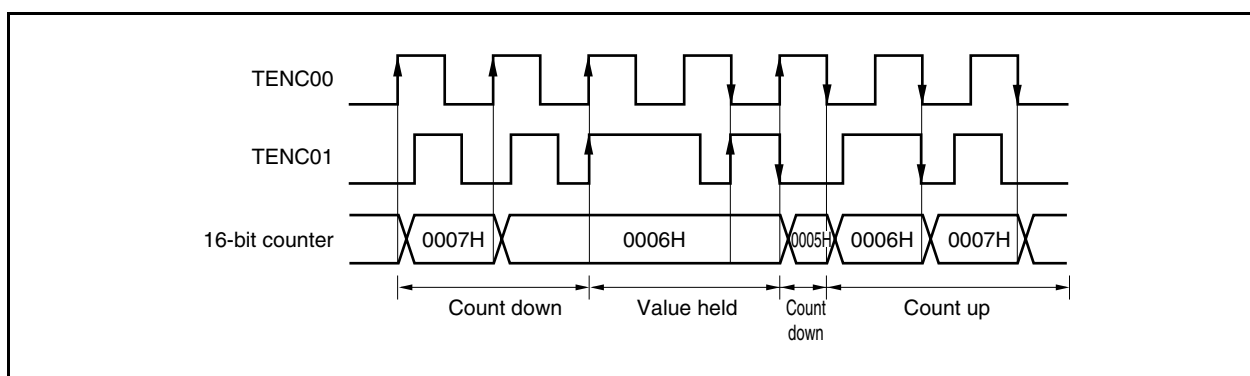
- When TT0UDS1 and TT0UDS0 bits = 11

| TENC00 Pin | TENC01 Pin | Count Operation |
|---|---|---|
| Low level | Falling edge | Count down |
| Rising edge | Low level | |
| High level | Rising edge | |
| Falling edge | High level | |
| Rising edge | | Count up |
| High level | Falling edge | |
| Falling edge | Low level | |
| Low level | Rising edge | |
| Simultaneous input to TENC00 and TENC01 pins | | Counter does not perform count operation but holds value immediately before. |

**Caution   Specification of the valid edges of the TENC00 and TENC01 pins is invalid.**

**Figure 9-49.  Operation Example (Count Operation When Valid Edges of TENC00 and TENC01 Pins do not Overlap)**
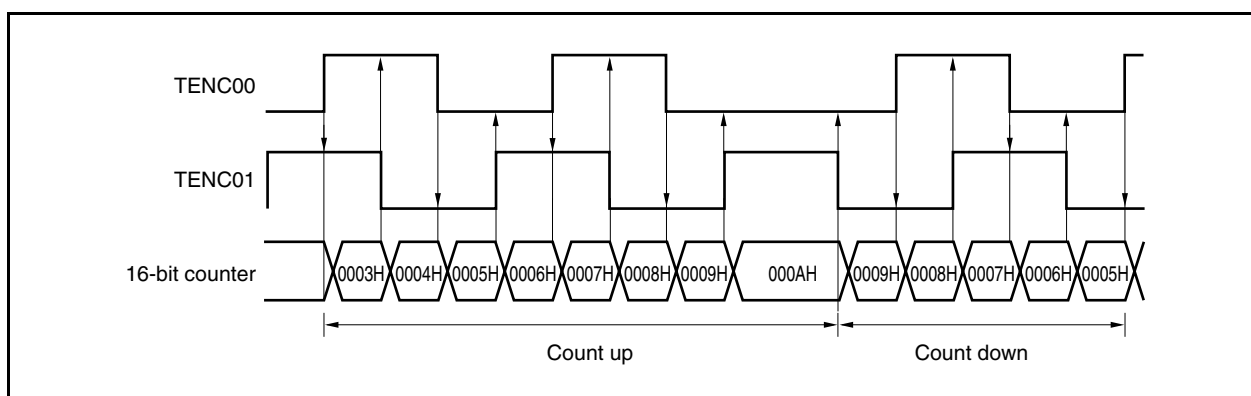


**Figure 9-50.  Operation Example (Count Operation When Valid Edges of TENC00 and TENC01 Pins Overlap)**
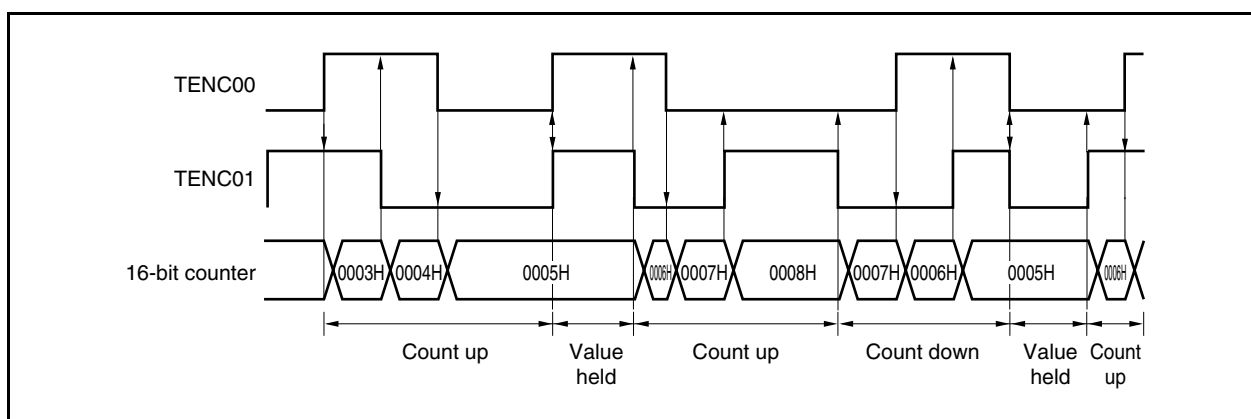
<2>  TT0ECM1 and TT0ECM0 bits: Timer/counter clear function upon match of the compare register
     The 16-bit counter performs its count operation in accordance with the set value of the TT0ECM1 and
     TT0ECM0 bits when the count value of the counter matches the value of the CCRn buffer register.

- When TT0ECM1 and TT0ECM0 bits = 00
  The 16-bit counter is not cleared when its count value matches the value of the CCRn buffer register.

- When TT0ECM1 and TT0ECM0 bits = 01
  The 16-bit counter performs a count operation under the following condition when its count value
  matches the value of the CCR0 buffer register.

| Next Count Operation | Description |
|---|---|
| Count up | 16-bit counter is cleared to 0000H. |
| Count down | Count value of 16-bit counter is counted down. |

- When TT0ECM1 and TT0ECM0 bits = 10
  The 16-bit counter performs a count operation under the following condition when its count value
  matches the value of the CCR1 buffer register.

| Next Count Operation | Description |
|---|---|
| Count up | Count value of 16-bit counter is counted up. |
| Count down | 16-bit counter is cleared to 0000H. |

- When TT0ECM1 and TT0ECM0 bits = 11
  The 16-bit counter performs a count operation under the following condition when its count value
  matches the value of the CCR0 buffer register.

| Next Count Operation | Description |
|---|---|
| Count up | 16-bit counter is cleared to 0000H. |
| Count down | Count value of 16-bit counter is counted down. |

The 16-bit counter performs a count operation under the following condition when its count value
matches the value of the CCR1 buffer register.

| Next Count Operation | Description |
|---|---|
| Count up | Count value of 16-bit counter is counted up. |
| Count down | 16-bit counter is cleared to 0000H. |

<3> TT0LDE bit: Transfer function of the set value of the TT0CCR0 register to the 16-bit counter when the counter underflows

When the TT0LDE bit = 1, the set value of the TT0CCR0 register can be transferred to the 16-bit counter when the counter underflows.

The TT0LDE bit is valid only in the encoder compare mode.

- Count operation in range from 0000H to set value of the TT0CCR0 register

  If the 16-bit counter performs a count operation when the TT0LDE bit = 1 and TT0ECM1 and TT0ECM0 bits = 01, and when the count value of the counter matches the set value of the CCR0 buffer register when the TT0ECM0 bit = 1, the 16-bit counter is cleared to 0000H if the next count operation is counting up.

  If the 16-bit counter underflows when the TT0LDE bit = 1, the set value of the TT0CCR0 register is transferred to the counter.

  Therefore, the counter can operate in a range from 0000H to the set value of the TT0CCR0 register in which the upper-limit count value is the set value of the TT0CCR0 register and the lower-limit value is 0000H.

**Figure 9-51.  Operation Example (Count Operation in Range from 0000H to Set Value of TT0CCR0 Register)**
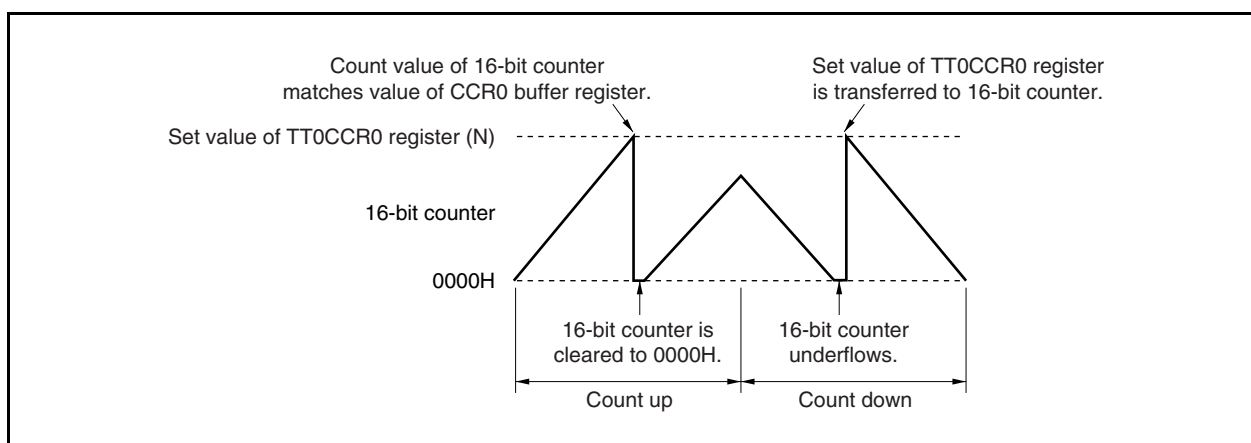
**Figure 9-52.  Operation Timing (Count Operation in Range from 0000H to Set Value of TT0CCR0 Register)**



Remark  TT0ESF bit:  Bit 0 of TMT0 option register 1 (TT0OPT1)
TT0EOF bit:  Bit 1 of TMT0 option register 1 (TT0OPT1)
TT0EUF bit:  Bit 2 of TMT0 option register 1 (TT0OPT1)
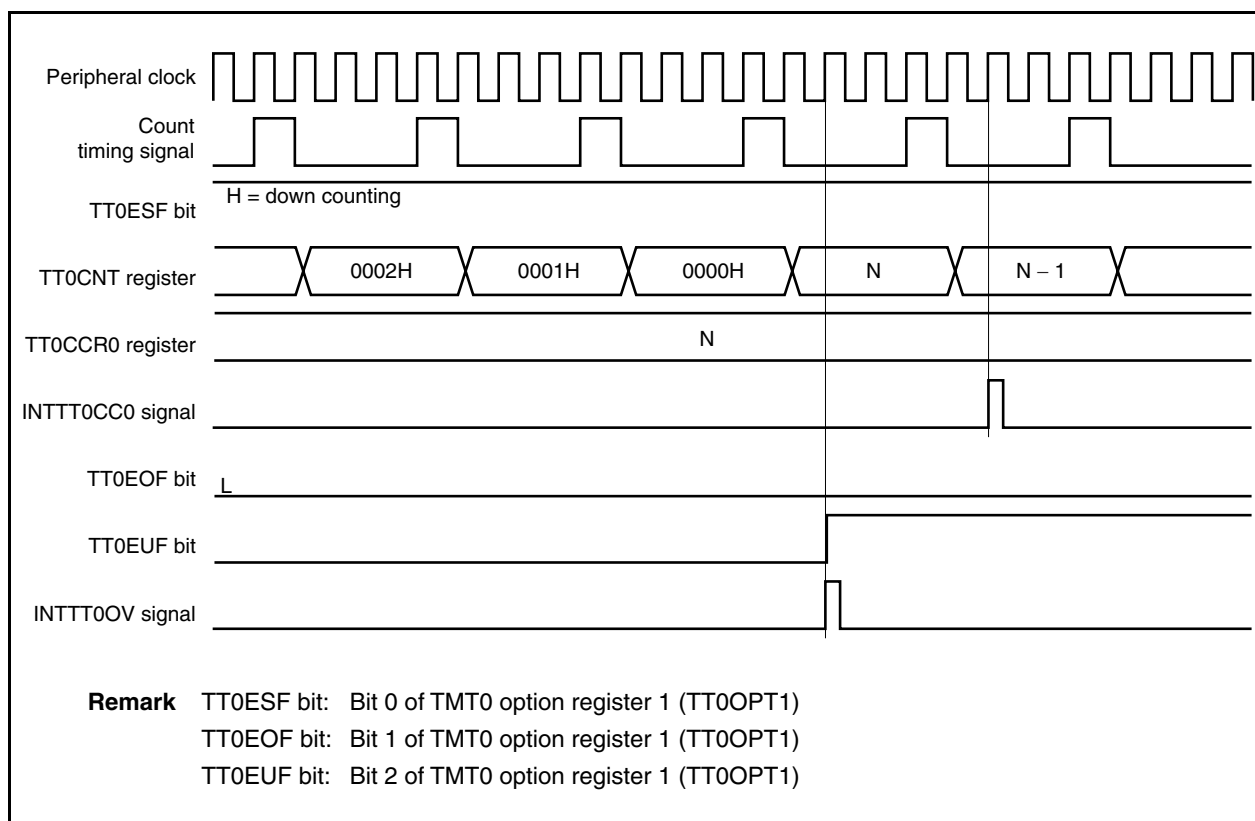
**(6) Function to clear counter to 0000H by encoder clear signal (TECR0 pin)**

The 16-bit counter can be cleared to 0000H by the input signal of the TECR0 pin in two ways which are selected by the TT0IOC3.TT0SCE bit.  The TT0SCE bit also controls, depending on its setting, the TT0IOC3.TT0ZCL, TT0IOC3.TT0BCL, TT0IOC3.TT0ACL, TT0IOC3.TT0ESC1, and TT0IOC3.TT0ECS0 bits.

The counter can be cleared by the methods described below only in the encoder compare mode.

**Table 9-9.  Relationship Between TT0SCE Bit and TT0ZCL, TT0BCL, TT0ACL, TT0ECS1, and TT0ECS0 Bits**

| Clearing Method | TT0SCE Bit | TT0ZCL Bit | TT0BCL Bit | TT0ACL Bit | TT0ECS1, TT0ECS0 Bits |
|---|---|---|---|---|---|
| <1> | 0 | Invalid | Invalid | Invalid | Valid |
| <2> | 1 | Valid | Valid | Valid | Invalid |

**(a) Clearing method <1>: By detecting edge of encoder clear signal (TECR0 pin) (TT0SCE bit = 0)**

When the TT0SCE bit = 0, the 16-bit counter is cleared to 0000H in synchronization with the peripheral clock if the valid edge of the TECR0 pin specified by the TT0ECS1 and TT0ECS0 bits is detected.  At this time, an encoder clear interrupt request signal (INTTT0EC) is generated.  When the TT0SCE bit = 0, the settings of the TT0ZCL, TT0BCL, and TT0ACL bits is invalid.

**Figure 9-53.  Operation Example (When TT0SCE Bit = 0, TT0ECS1 and TT0ECS0 Bits = 01, and TT0UDS1 and TT0UDS0 Bits = 11)**
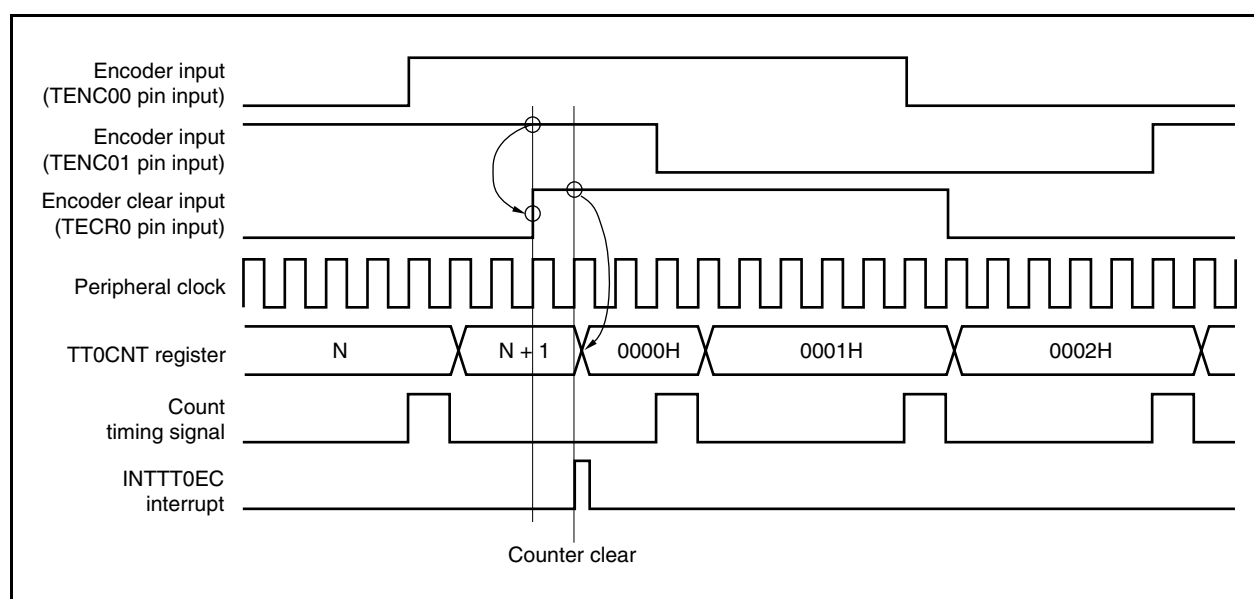
**(b) Clearing method <2>: By detecting clear level condition of the TENC00, TENC01, and TECR0 pins (TT0SCE bit = 1)**

When the TT0SCE bit = 1, the 16-bit counter is cleared to 0000H if the clear level condition of the TECR0, TENC00, or TENC01 pin specified by the TT0ZCL, TT0BCL, and TT0ACL bits is detected.  At this time, the encoder clear interrupt request signal (INTTT0EC) is not generated.  The settings of the TT0ECS1 and TT0ECS0 bits is invalid when the TT0SCE bit = 1.

**Table 9-10.  16-bit Counter Clearing Condition When TT0SCE Bit = 1**

| Clear Level Condition Setting | | | Input Level of Encoder Pin | | |
|---|---|---|---|---|---|
| TT0ZCL Bit | TT0BCL Bit | TT0ACL Bit | TECR0 Pin | TENC01 Pin | TENC00 Pin |
| 0 | 0 | 0 | L | L | L |
| 0 | 0 | 1 | L | L | H |
| 0 | 1 | 0 | L | H | L |
| 0 | 1 | 1 | L | H | H |
| 1 | 0 | 0 | H | L | L |
| 1 | 0 | 1 | H | L | H |
| 1 | 1 | 0 | H | H | L |
| 1 | 1 | 1 | H | H | H |

**Caution   The 16-bit counter is cleared to 0000H when the clear level condition of the TT0ZCL, TT0BCL, and TT0ACL bits match the input level of the TECR0, TENC01, or TENC00 pin.**

**Figure 9-54. Operation Example (When TT0SCE Bit = 1, TT0ZCL Bit = 1, TT0BCL Bit = 0, TT0ACL Bit = 1, TT0UDS1 and TT0UDS0 Bits = 11, TECR0 = High Level, TENC01 = Low Level, and TENC00 = High Level) (1/3)**
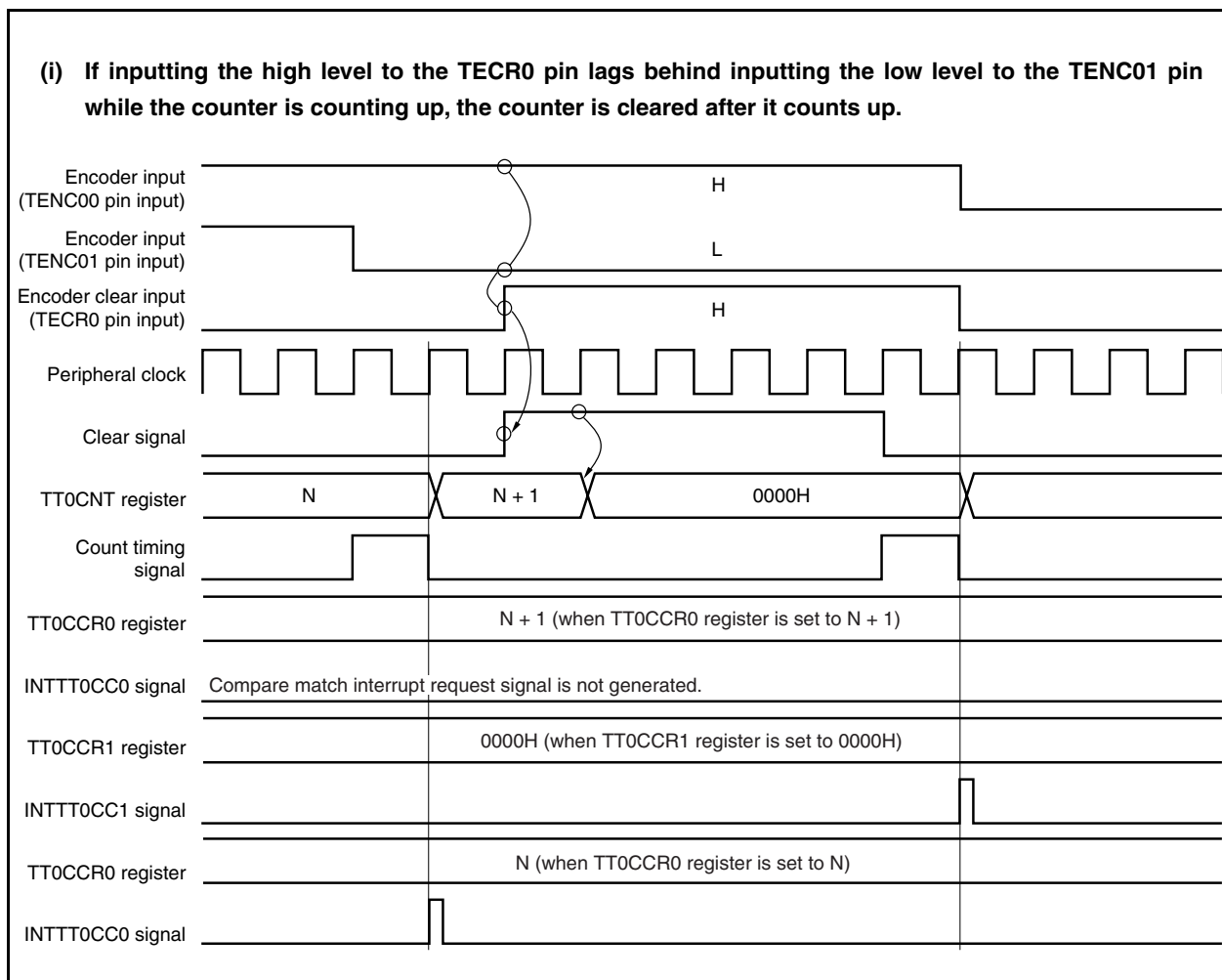
**(i)** **If inputting the high level to the TECR0 pin lags behind inputting the low level to the TENC01 pin while the counter is counting up, the counter is cleared after it counts up.**

**Figure 9-54.  Operation Example (When TT0SCE Bit = 1, TT0ZCL Bit = 1, TT0BCL Bit = 0, TT0ACL Bit = 1, TT0UDS1 and**
**TT0UDS0 Bits = 11, TECR0 = High Level, TENC01 = Low Level, and TENC00 = High Level) (2/3)**



(ii)  **If the high level is input to the TECR0 pin at the same time as the low level is input to the TECN01 pin**
      **while the counter is counting up, the counter is cleared without counting up.**

(iii) **If the high level is input to the TECR0 pin earlier than the low level is input to the TENC01 pin while**
      **the counter is counting up, the counter is cleared without counting up.**
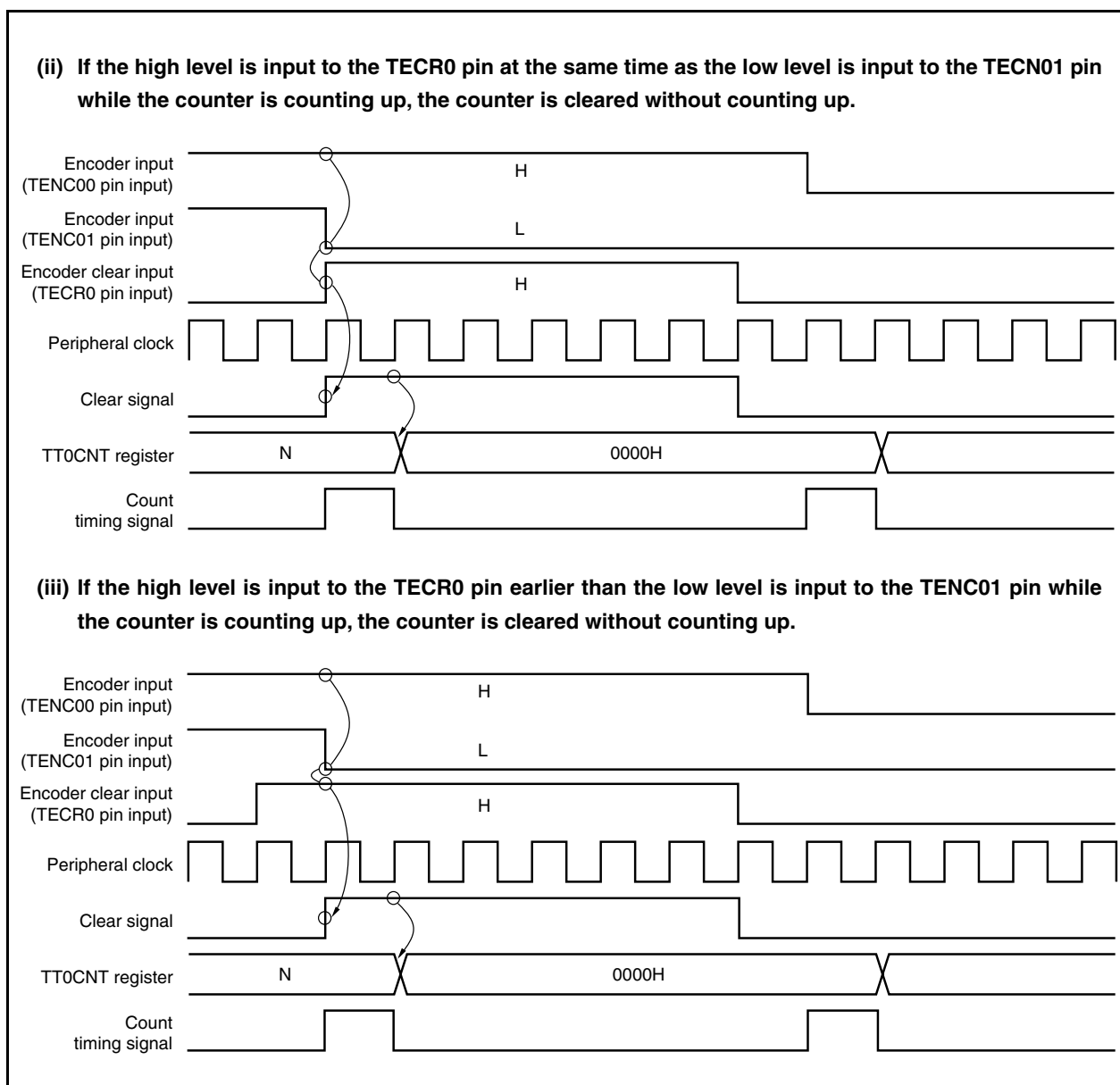
**Figure 9-54. Operation Example (When TT0SCE Bit = 1, TT0ZCL Bit = 1, TT0BCL Bit = 0, TT0ACL Bit = 1, TT0UDS1 and TT0UDS0 Bits = 11, TECR0 = High Level, TENC01 = Low Level, and TENC00 = High Level) (3/3)**
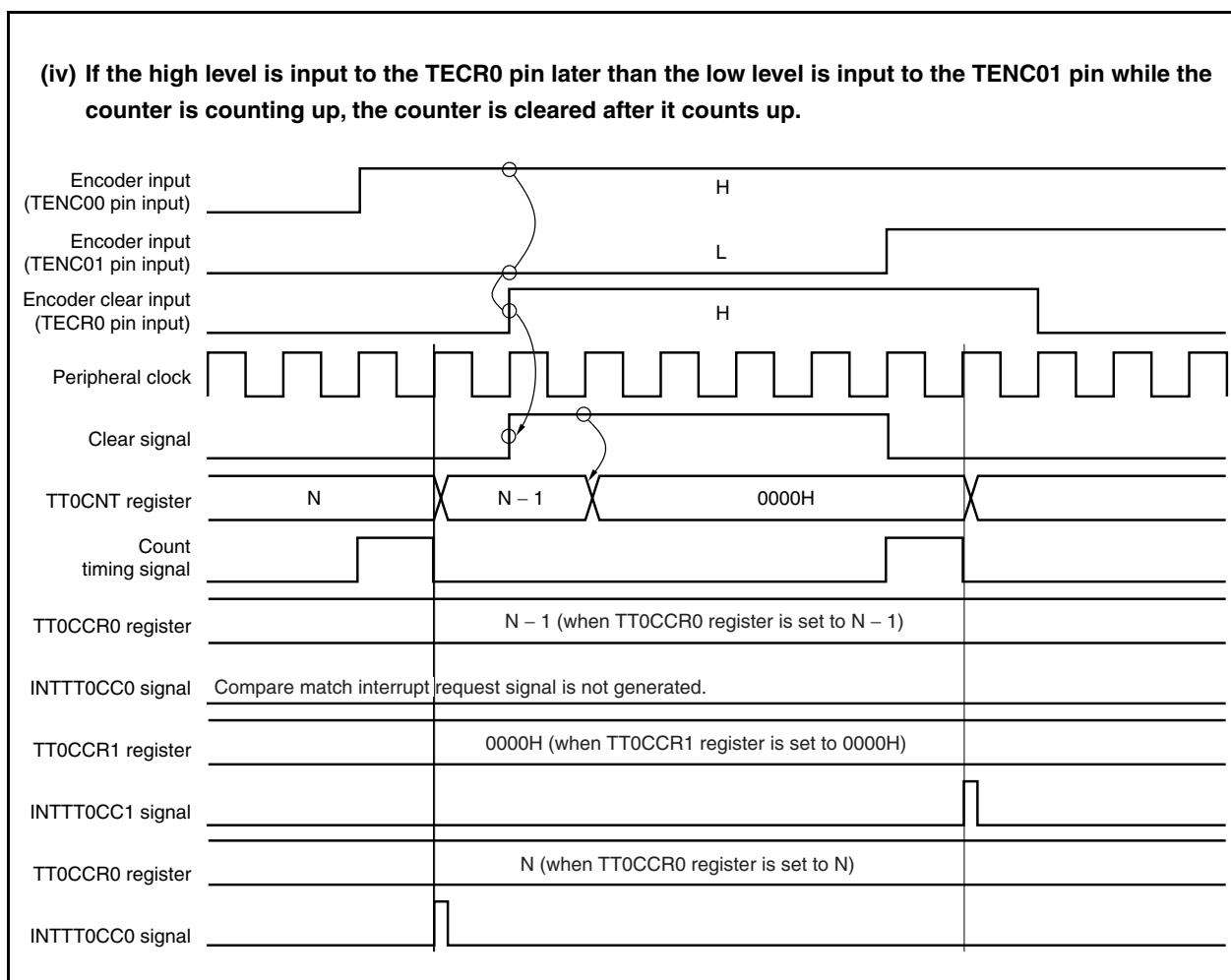


If the counter is cleared in this way, a miscount does not occur even if inputting the signal to the TECR0 pin is late, because the clear level condition of the TECR0, TENC01, and TENC00 pins is set and the 16-bit counter is cleared to 0000H when the clear level condition is detected.

**(7) Notes on using encoder count function**

**(a) If compare match interrupt is not generated immediately after operation is started**

If a value which is the same as that of the TT0TCW register is set to the TT0CCR0 or TT0CCR1 register and the counter operation is started when the TT0CTL2.TT0ECC bit = 0, and if the count value (TT0TCW) of the 16-bit counter matches the value of the CCRn buffer register immediately after the start of the operation, the match is masked and the compare match interrupt request signal (INTTT0CCn) is not generated (n = 0, 1). In addition, the 16-bit counter is not cleared to 0000H by setting the TT0CTL2.TT0ECM1 and TT0CTL2.TT0ECM0 bits.
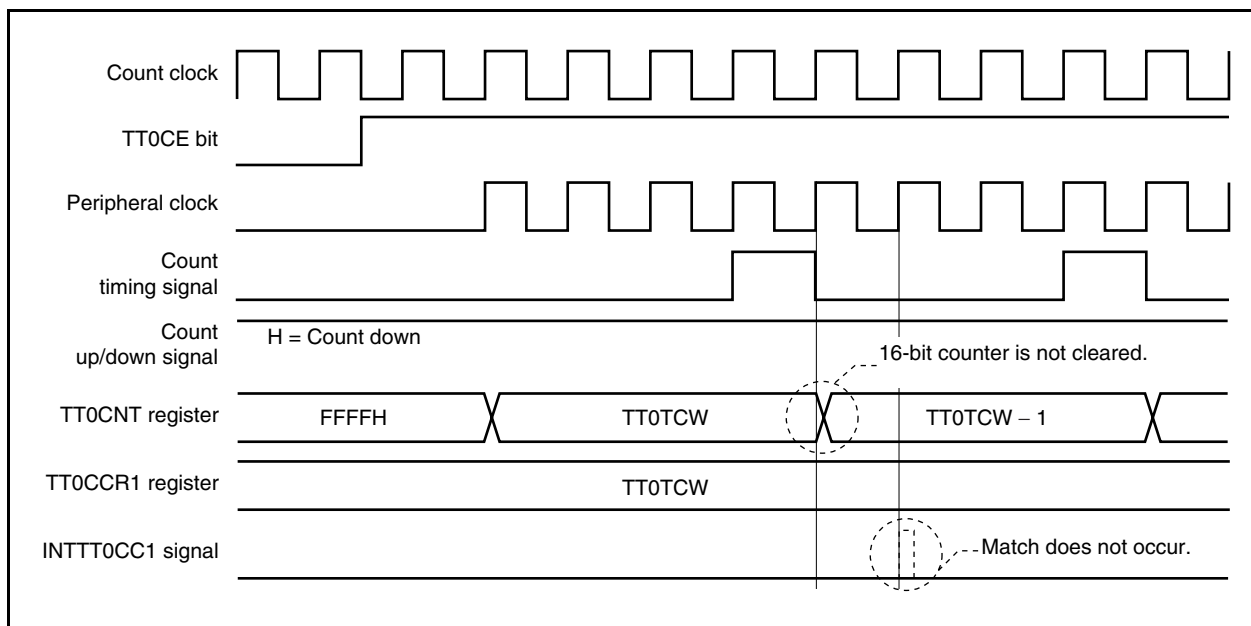
**(b) If overflow does not occur immediately after start of operation**

If the count operation is resumed when the TT0CTL2.TT0ECC bit = 1, the 16-bit counter does not overflow if its count value that has been held is FFFFH and if the next count operation is counting up.

After the counter starts operating and counts up from a count value (value of TT0TCW register = FFFFH), the counter overflows from FFFFH to 0000H.  However, detection of the overflow is masked, the overflow flag (TT0EOF) is not set, and the overflow interrupt request signal (INTTT0OV) is not generated.
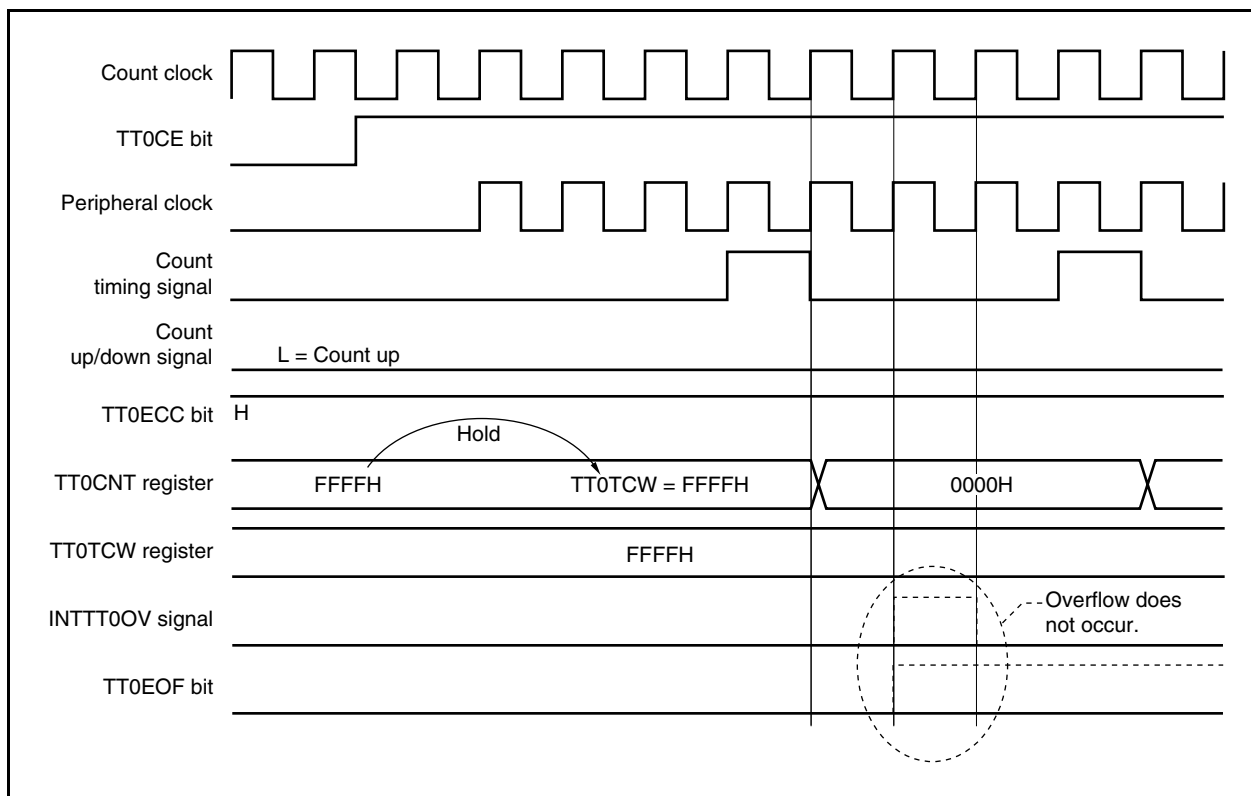
### 9.6.10 Encoder compare mode (TT0MD3 to TT0MD0 bits = 1000)

In the encoder compare mode, the encoder is controlled by using both the TT0CCR0 and TT0CCR1 registers as compare registers and the input pins for encoder count function (TENC00, TENC01, and TECR0).

In this mode, the 16-bit counter can be cleared to 0000H in three ways: when the count value of the counter matches the value of the CCRn buffer register (compare match interrupt request signal (INTTT0CCn) is generated), when the edge of the encoder clear input (TECR0 pin) is detected, and when the clear level condition of TENC00, TENC01, and TECR0 pins is detected.

When the 16-bit counter underflows, the set value of the TT0CCR0 register can be transferred to the counter.

### (1) Encoder compare mode operation flow

**Figure 9-55. Encoder Compare Mode Operation Flow**
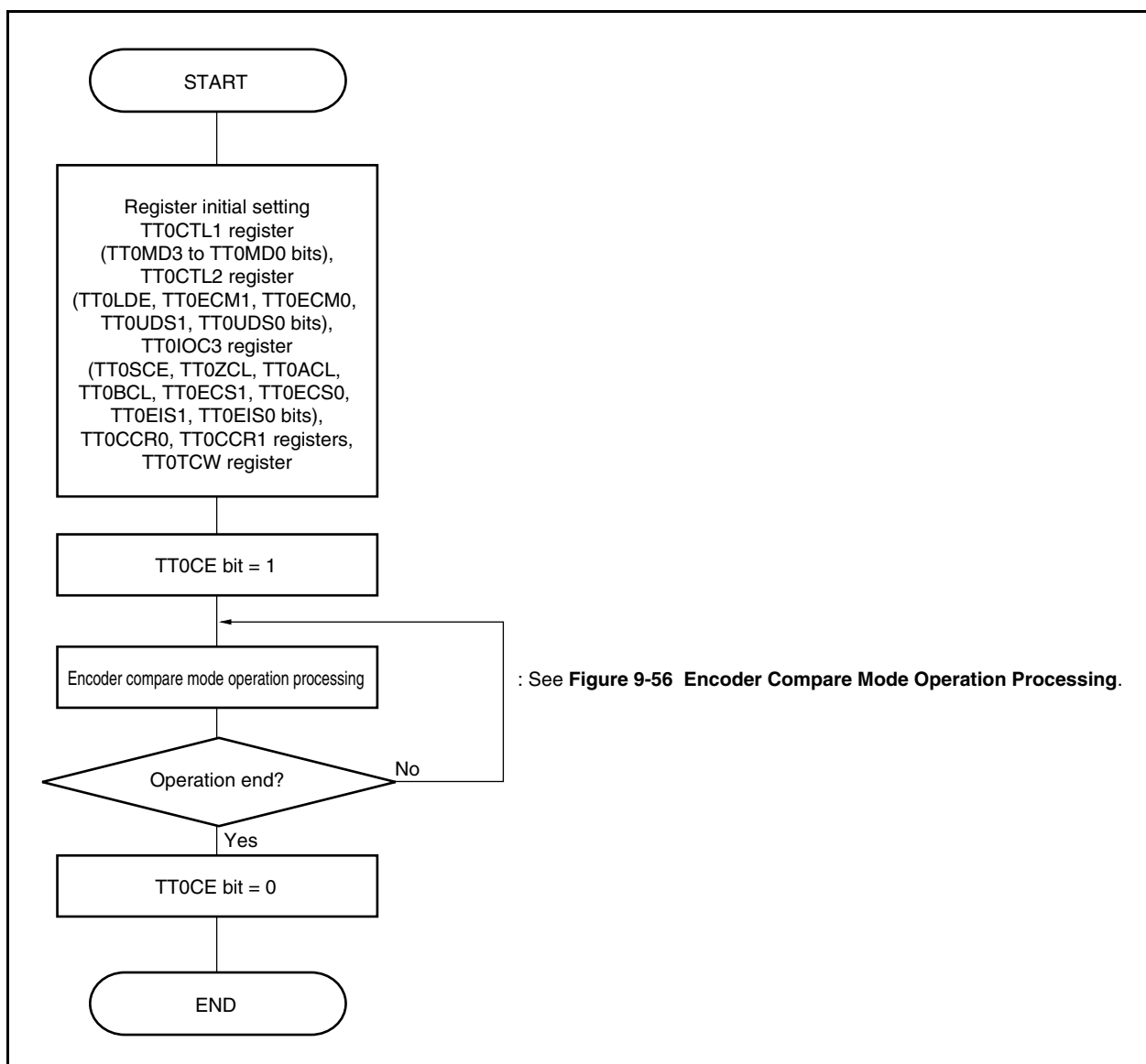
**Figure 9-56.  Encoder Compare Mode Operation Processing**
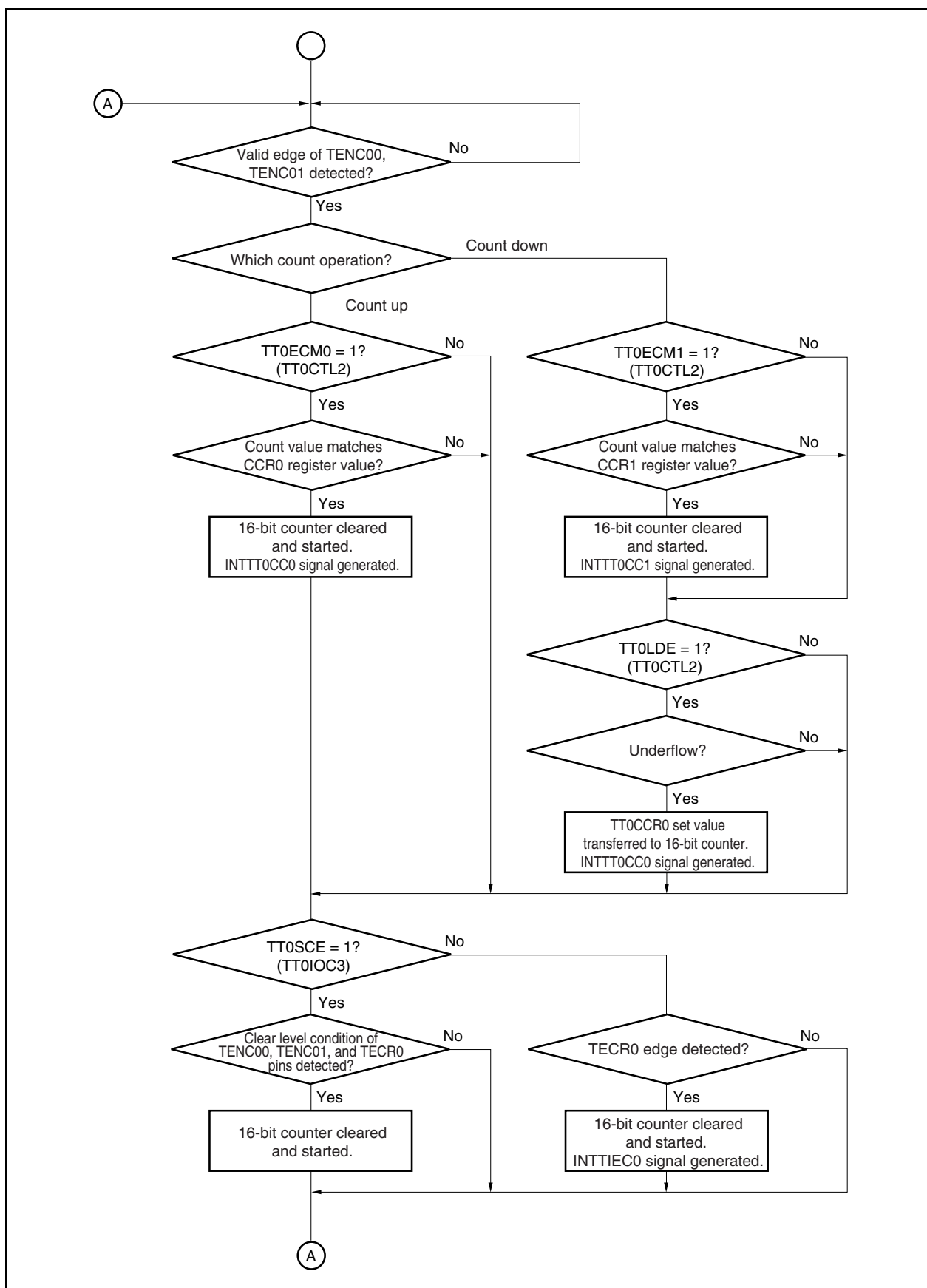
**(2)  Encoder compare mode operation timing**

**(a)  Basic timing 1**

**[Register setting conditions]**
- TT0CTL2.TT0ECM1 and TT0CTL2.TT0ECM0 bits = 01
  The 16-bit counter is cleared to 0000H when its count value matches the value of the CCR0 buffer register.
- TT0CTL2.TT0LDE bit = 1
  The set value of the TT0CCR0 register is transferred to the 16-bit counter when it overflows.
- TT0IOC3.TT0SCE bit = 0, and TT0IOC3.TT0ECS1 and TT0IOC3.TT0ECS0 bits = 00
  Specification of clearing the 16-bit counter when the edge of the encoder clear input signal (TECR0 pin) is detected (no edge specified)
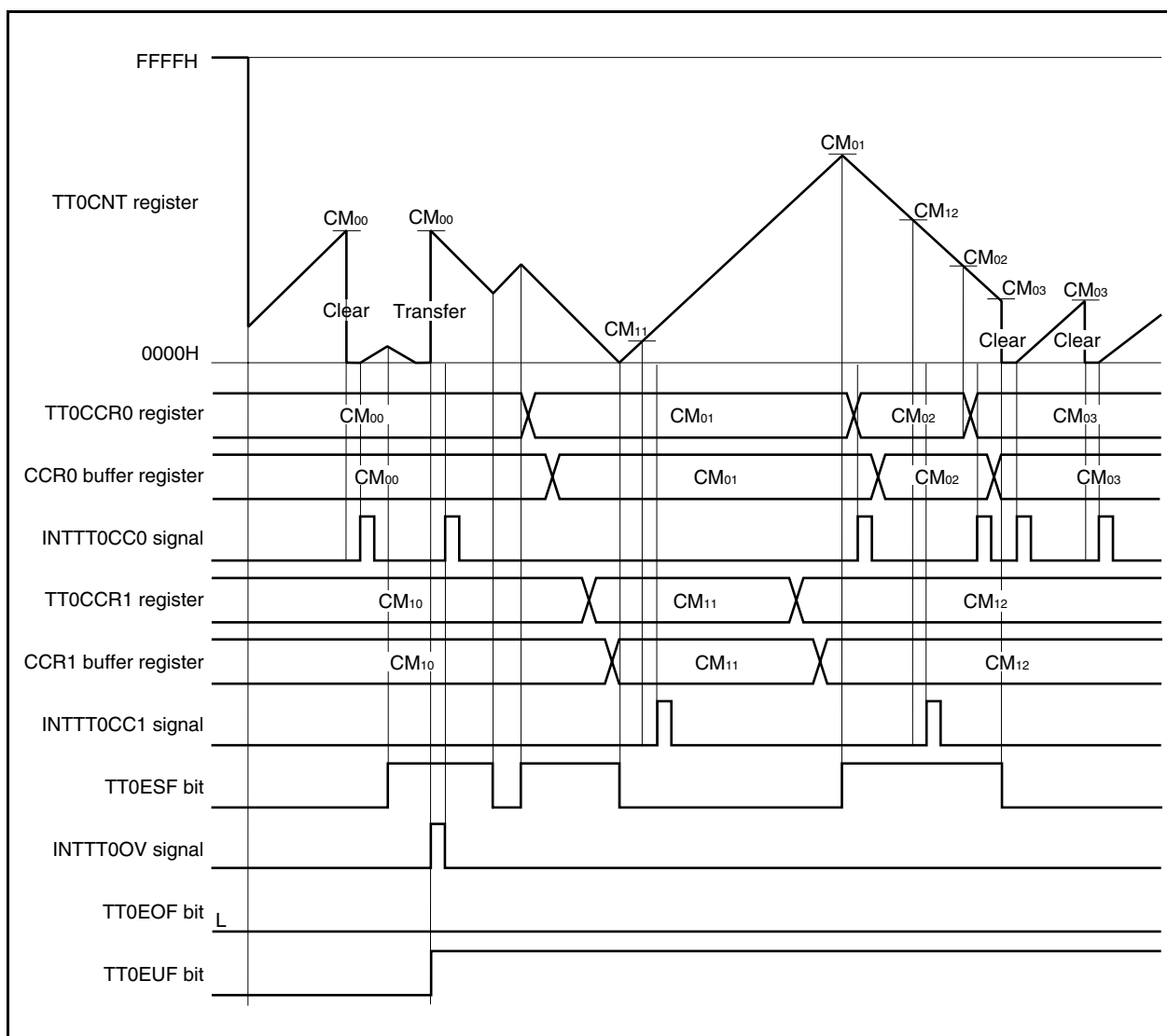
When the 16-bit counter starts operating (TT0CE bit = 0 → 1), the set value of the TT0TCW register is transferred to the counter and the 16-bit counter starts operating.

When the count value of the counter matches the value of the CCR0 buffer register, the compare match interrupt request signal (INTTT0CC0) is generated. Because the TT0ECM0 bit = 1, the 16-bit counter is cleared to 0000H if the next count operation is counting up.

When the count value of the 16-bit counter matches the value of the CCR1 buffer register, the compare match interrupt request signal (INTTT0CC1) is generated. Because the TT0ECM1 bit = 0, the 16-bit counter is not cleared to 0000H when its value matches that of the CCR1 buffer register.

When the TT0LDE bit = 1 and TT0ECM0 bit = 1, the counter can operate in a range from 0000H to the set value of the TT0CCR0 register.

### (b) Basic timing 2

**[Register setting condition]**

- TT0CTL2.TT0ECM1 and TT0CTL2.TT0ECM0 bits = 00

  The 16-bit counter is not cleared even when its count value matches the value of the CCRn buffer register (a = 0, 1).

- TT0CTL2.TT0LDE bit = 0

  The set value of the TT0CCR0 register is not transferred to the 16-bit counter after the counter underflows.

- TT0IOC3.TT0SCE bit = 0, and TT0IOC3.TT0ECS1 and TT0IOC3.TT0ECS0 bits = 00

  Specification of clearing the 16-bit counter when the edge of the encoder clear input signal (TECR0 pin) is detected (no edge specified)
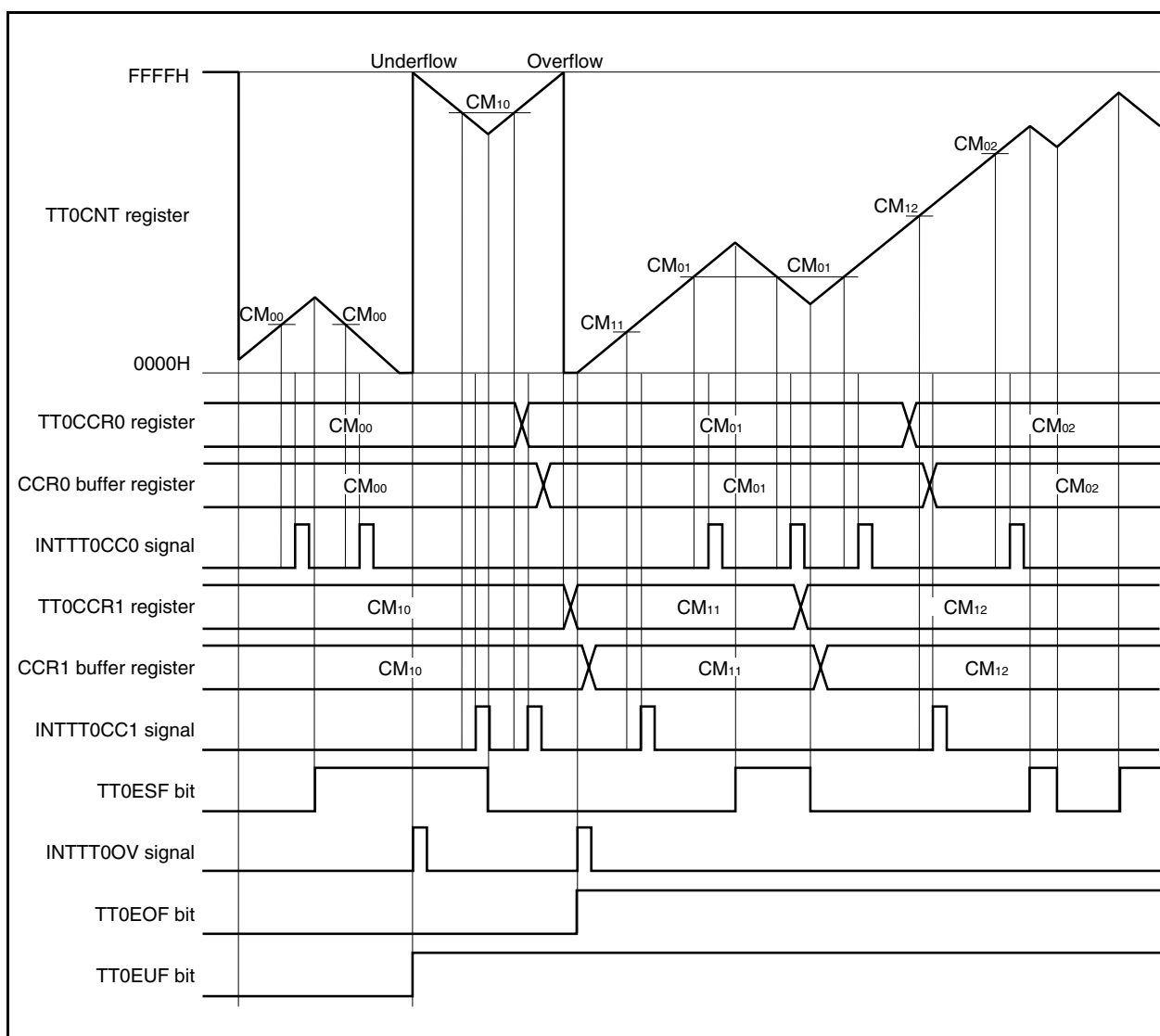
When the 16-bit counter starts operating (TT0CE bit = 0 → 1), the set value of the TT0TCW register is transferred to the 16-bit counter and the counter starts operating.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTT0CC0) is generated.

When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTT0CC1) is generated.

The 16-bit counter is not cleared to 0000H even when its count value matches the value of the CCRn buffer register because the TT0ECM1 and TT0ECM0 bits = 00 (n = 0, 1).

**(c) Basic timing 3**

**[Register setting condition]**

- TT0CTL2.TT0ECM1 and TT0CTL2.TT0ECM0 bits = 11

  The count value of the 16-bit counter is cleared to 0000H when its value matches the value of the CCR0 buffer register.

  The count value of the 16-bit counter is cleared to 0000H when its value matches the value of the CCR1 buffer register.

- Setting of the TT0CTL2.TT0LDE bit is invalid.

- TT0IOC3.TT0SCE bit = 0, and TT0IOC3.TT0ECS1 and TT0IOC3.TT0ECS0 bits = 00

  Specification of clearing the 16-bit counter when the edge of the encoder clear input signal (TECR0 pin) is detected (no edge specified)

When the 16-bit counter starts operating (TT0CE bit = 0 → 1), the set value of the TT0TCW register is transferred to the 16-bit counter and the counter starts operating.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTT0CC0) is generated. At this time, the 16-bit counter is cleared to 0000H if the next count operation is counting up.

When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTT0CC1) is generated. At this time, the 16-bit counter is cleared to 0000H if the next count operation is counting down.

## CHAPTER 10  16-BIT INTERVAL TIMER M (TMM)

The V850ES/JG3-U and V850ES/JH3-U have four TMM channels (TMMn).

### 10.1  Overview

TMMn has the following features.

- Interval function
- 8 clocks selectable
- 16-bit counter $\times$ 1
  (The 16-bit counter cannot be read during timer count operation.)
- Compare register $\times$ 1
  (The compare register cannot be written during timer counter operation.)
- Compare match interrupt $\times$ 1

TMMn supports only the clear & start mode.  The free-running timer mode is not supported.

**Remark**   n = 0 to 3

## 10.2  Configuration

TMMn includes the following hardware.

**Table 10-1.  Configuration of TMMn**

| Item | Configuration |
| --- | --- |
| Timer register | 16-bit counter |
| Register | TMMn compare register 0 (TMnCMP0) |
| Control register | TMMn control register 0 (TMnCTL0) |

**Figure 10-1.  Block Diagram of TMMn**



**Note**  In TMM0, $f_{XX}$, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/64$, $f_{XX}/512$, $f_{XX}/1024$, $f_R$, $f_{XT}$

**Remark**   $f_{XX}$:        Main clock frequency
             $f_R$:         Internal oscillation clock frequency
             $f_{XT}$:       Subclock frequency
             n = 0 to 3
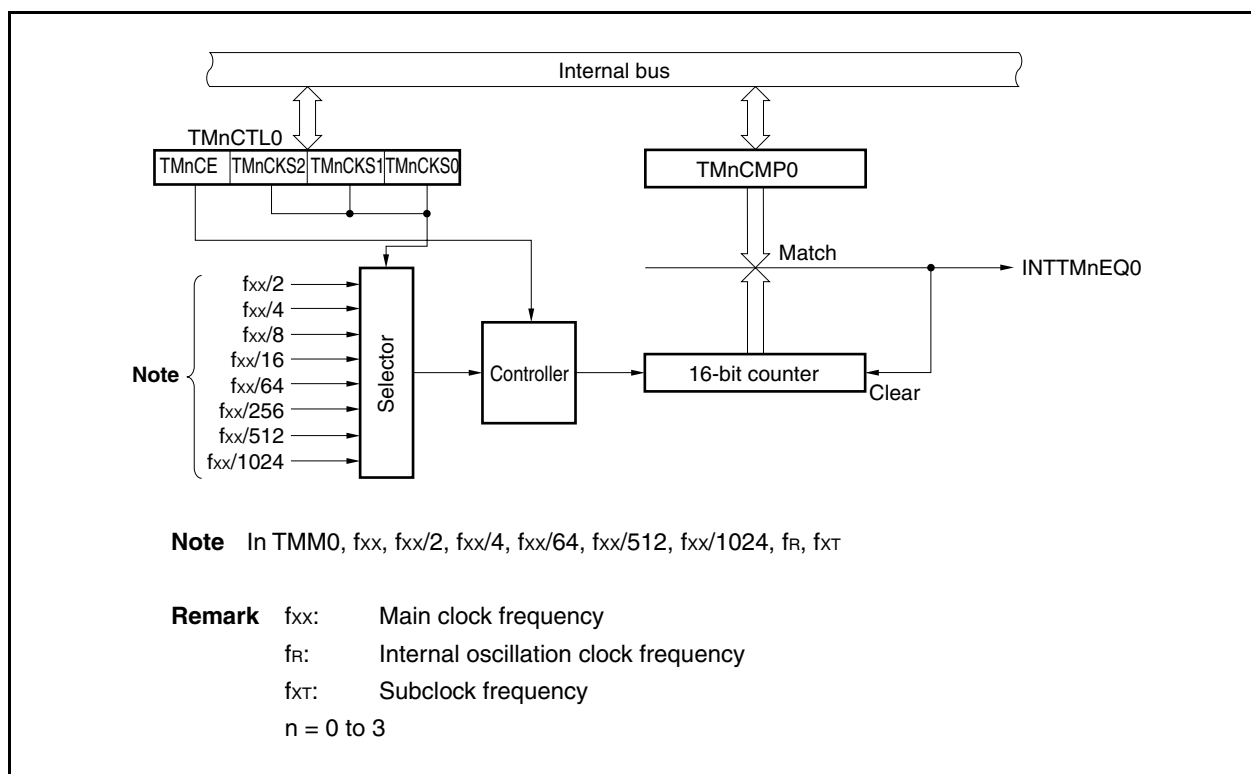
**(1) 16-bit counter**

This is a 16-bit counter that counts the internal clock.

The 16-bit counter cannot be read or written.

**(2) TMMn compare register 0 (TMnCMP0)**

The TMnCMP0 register is a 16-bit compare register.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

Software can be used to always write the same value to the TMnCMP0 register.

Rewriting the TMnCMP0 register is prohibited when the TMnCTL0.TMnCE bit = 1.

After reset:  0000H     R/W     Address:  TM0CMP0 FFFFFA84H, TM1CMP0 FFFFFA94H,

TM2CMP0 FFFFFAA4H, TM3CMP0 FFFFFAB4H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMnCMP0 | | | | | | | | | | | | | | | | |

(n = 0 to 3)
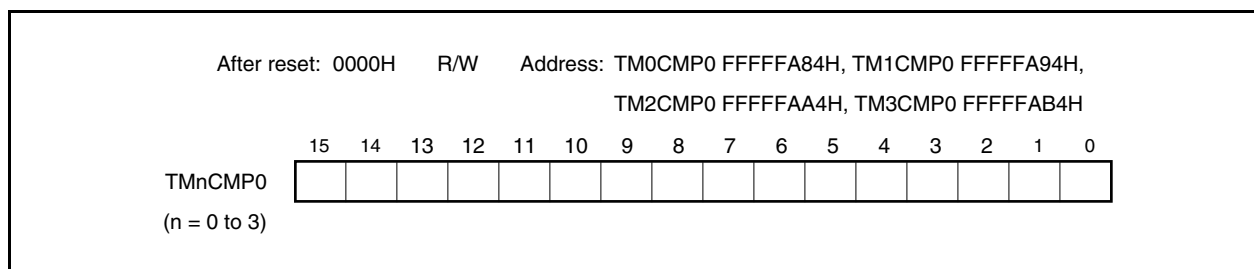
## 10.3 Registers

**(1) TMMn control register (TMnCTL0)**

The TMnCTL0 register is an 8-bit register that controls the TMMn operation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

Software can be used to always write the same value to the TMnCTL0 register.

**Remark**    n = 0 to 3

After reset: 00H　　R/W　　Address:　TM0CTL0 FFFFFA80H, TM1CTL0 FFFFFA90H,
　　　　　　　　　　　　　　　　　　　　TM2CTL0 FFFFFAA0H, TM3CTL0 FFFFFAB0H

| | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TMnCTL0 | TMnCE | 0 | 0 | 0 | 0 | TMnCKS2 | TMnCKS1 | TMnCKS0 |

(n = 0 to 3)

| TMnCE | Internal clock operation enable/disable specification |
|---|---|
| 0 | TMMn operation disabled (16-bit counter reset asynchronously). Operation clock application stopped. |
| 1 | TMMn operation enabled. Operation clock application started. TMMn operation started. |
| | Internal clock control and internal circuit reset for TMMn are performed asynchronously using the TMnCE bit. When the TMnCE bit is cleared to 0, the internal clock of TMMn is disabled (fixed to low level) and 16-bit counter is reset asynchronously. |

(m = 0)

| TMmCKS2 | TMmCKS1 | TMmCKS0 | Count clock selection | | | |
|---|---|---|---|---|---|---|
| | | | | fxx = 48 MHz | fxx = 32 MHz | fxx = 24 MHz |
| 0 | 0 | 0 | fxx | 20.8 ns | 31.3 ns | 41.7 ns |
| 0 | 0 | 1 | fxx/2 | 41.7 ns | 62.5 ns | 83.3 ns |
| 0 | 1 | 0 | fxx/4 | 83.3 ns | 125 ns | 167 ns |
| 0 | 1 | 1 | fxx/64 | 1.33 ms | 2.00 ms | 2.67 ms |
| 1 | 0 | 0 | fxx/512 | 10.7 ms | 16.0 ms | 21.3 ms |
| 1 | 0 | 1 | fxx/1024 | 21.3 ms | 32.0 ms | 42.7 ms |
| 1 | 1 | 0 | fR/8 | 36.4 $\mu$s | 36.4 $\mu$s | 36.4 $\mu$s |
| 1 | 1 | 1 | fXT | 30.5 $\mu$s | 30.5 $\mu$s | 30.5 $\mu$s |

(m = 1 to 3)

| TMmCKS2 | TMmCKS1 | TMmCKS0 | Count clock selection | | | |
|---|---|---|---|---|---|---|
| | | | | fxx = 48 MHz | fxx = 32 MHz | fxx = 24 MHz |
| 0 | 0 | 0 | fxx/2 | 41.7 ns | 62.5 ns | 83.3 ns |
| 0 | 0 | 1 | fxx/4 | 83.3 ns | 125 ns | 167 ns |
| 0 | 1 | 0 | fxx/8 | 167 ns | 250 ns | 333 ns |
| 0 | 1 | 1 | fxx/16 | 333 ns | 500 ns | 667 ns |
| 1 | 0 | 0 | fxx/64 | 1.33 ms | 2.00 ms | 2.67 ms |
| 1 | 0 | 1 | fxx/256 | 5.33 ms | 8.00 ms | 10.7 ms |
| 1 | 1 | 0 | fxx/512 | 10.7 ms | 16.0 ms | 21.3 ms |
| 1 | 1 | 1 | fxx/1024 | 21.3 ms | 32.0 ms | 42.7 ms |

**Cautions 1. Set the TMnCKS2 to TMnCKS0 bits when the TMnCE bit = 0.**
**When changing the value of TMnCE from 0 to 1, it is not possible to set the value of the TMnCKS2 to TMnCKS0 bits simultaneously.**
**2. Be sure to clear bits 3 to 6 to "0".**

**Remark**　fxx: Main clock frequency
　　　　　　fR:　Internal oscillation clock frequency
　　　　　　fXT: Subclock frequency

## 10.4  Operation

**Caution   Do not set the TMnCMP0 register to FFFFH.**

### 10.4.1  Interval timer mode

In the interval timer mode, an interrupt request signal (INTTMnEQ0) is generated at the specified interval if the TMnCTL0.TMnCE bit is set to 1.

**Figure 10-2.  Configuration of Interval Timer**
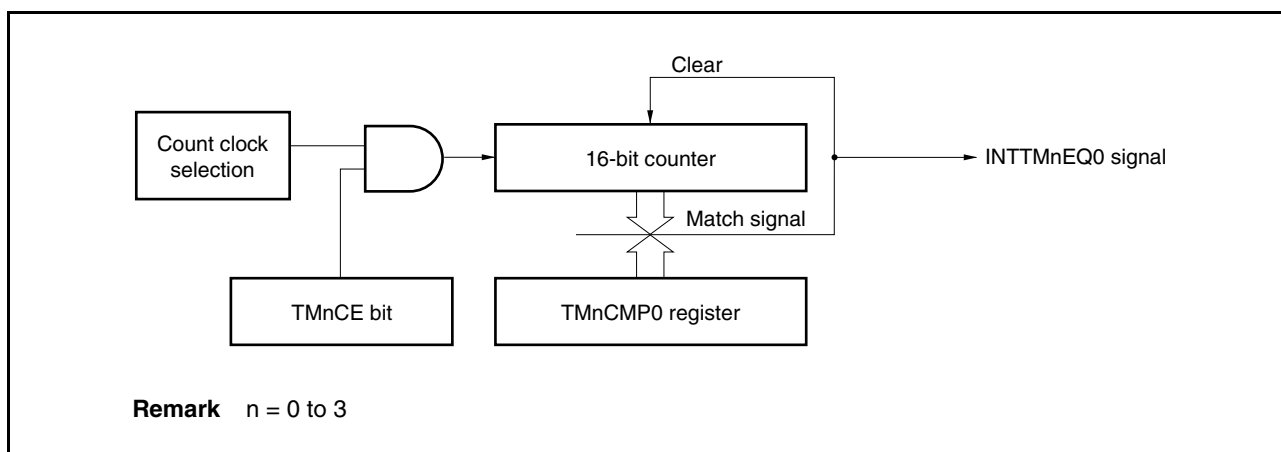


**Figure 10-3.  Basic Timing of Operation in Interval Timer Mode**
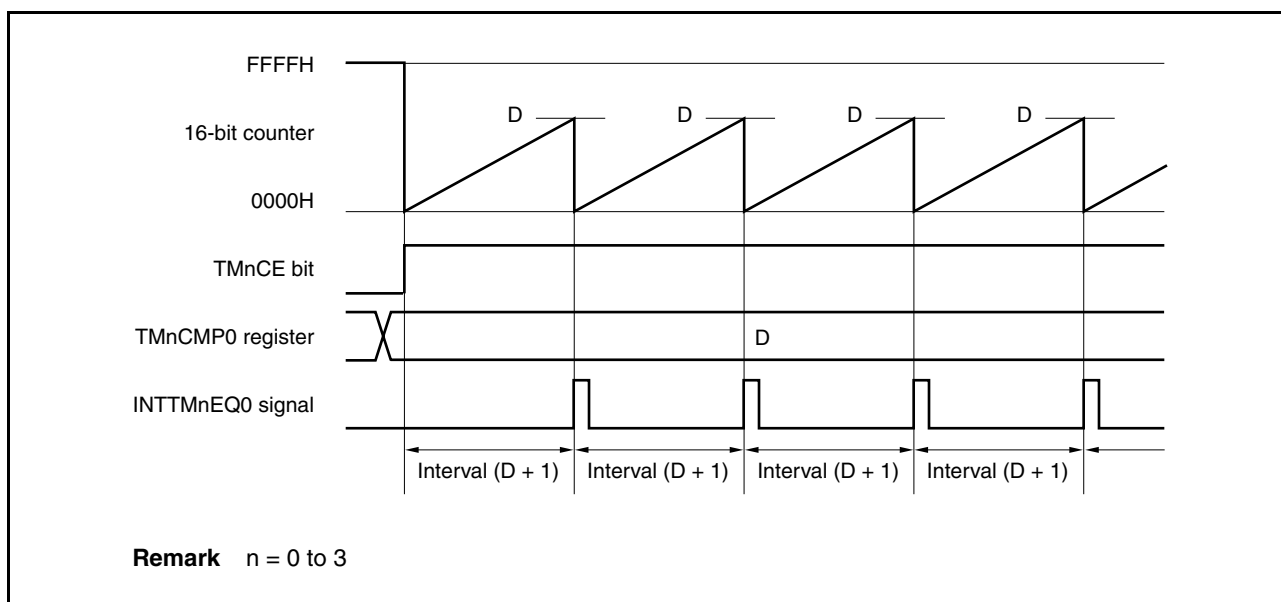
When the TMnCE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H in synchronization with the count clock, and the counter starts counting.

When the count value of the 16-bit counter matches the value of the TMnCMP0 register, the 16-bit counter is cleared to 0000H and a compare match interrupt request signal (INTTMnEQ0) is generated.

The interval can be calculated by the following expression.

Interval = (Set value of TMnCMP0 register + 1) × Count clock cycle

**Figure 10-4.  Register Setting for Interval Timer Mode Operation**



**(a) TMMn control register 0 (TMnCTL0)**

**(b) TMMn compare register 0 (TMnCMP0)**

If the TMnCMP0 register is set to D, the interval is as follows.

Interval = (D + 1) × Count clock cycle

**Remark**   n = 0 to 3
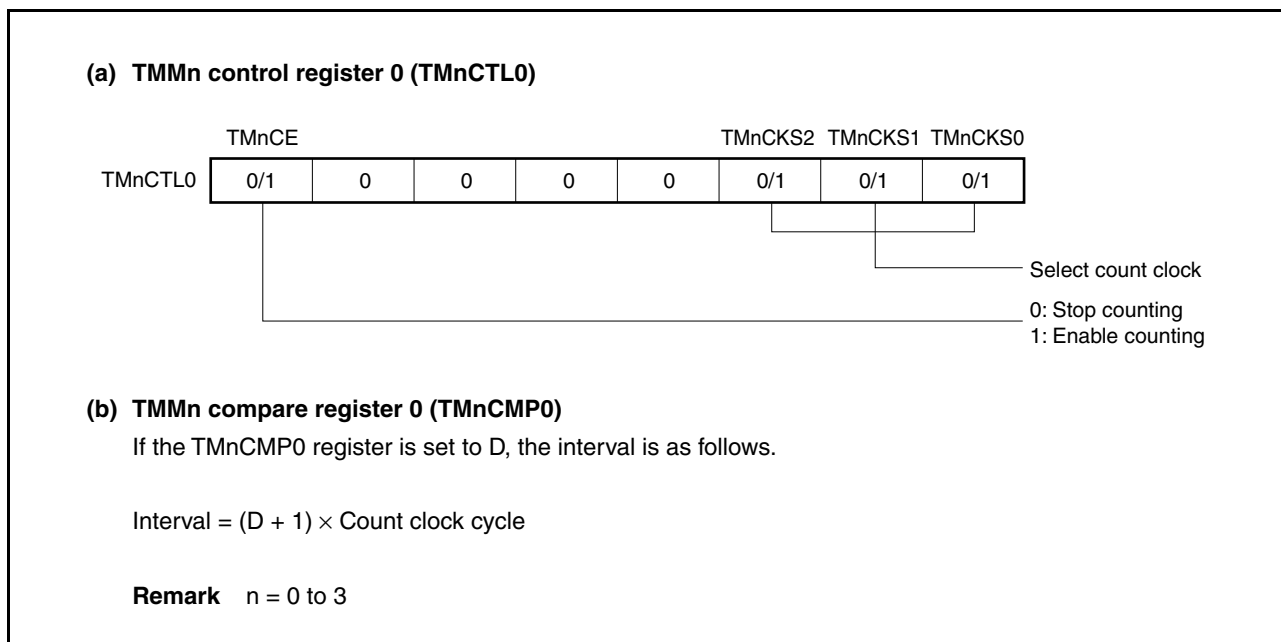
**(1)  Interval timer mode operation flow**

**Figure 10-5.  Software Processing Flow in Interval Timer Mode**



**Remark**   n = 0 to 3
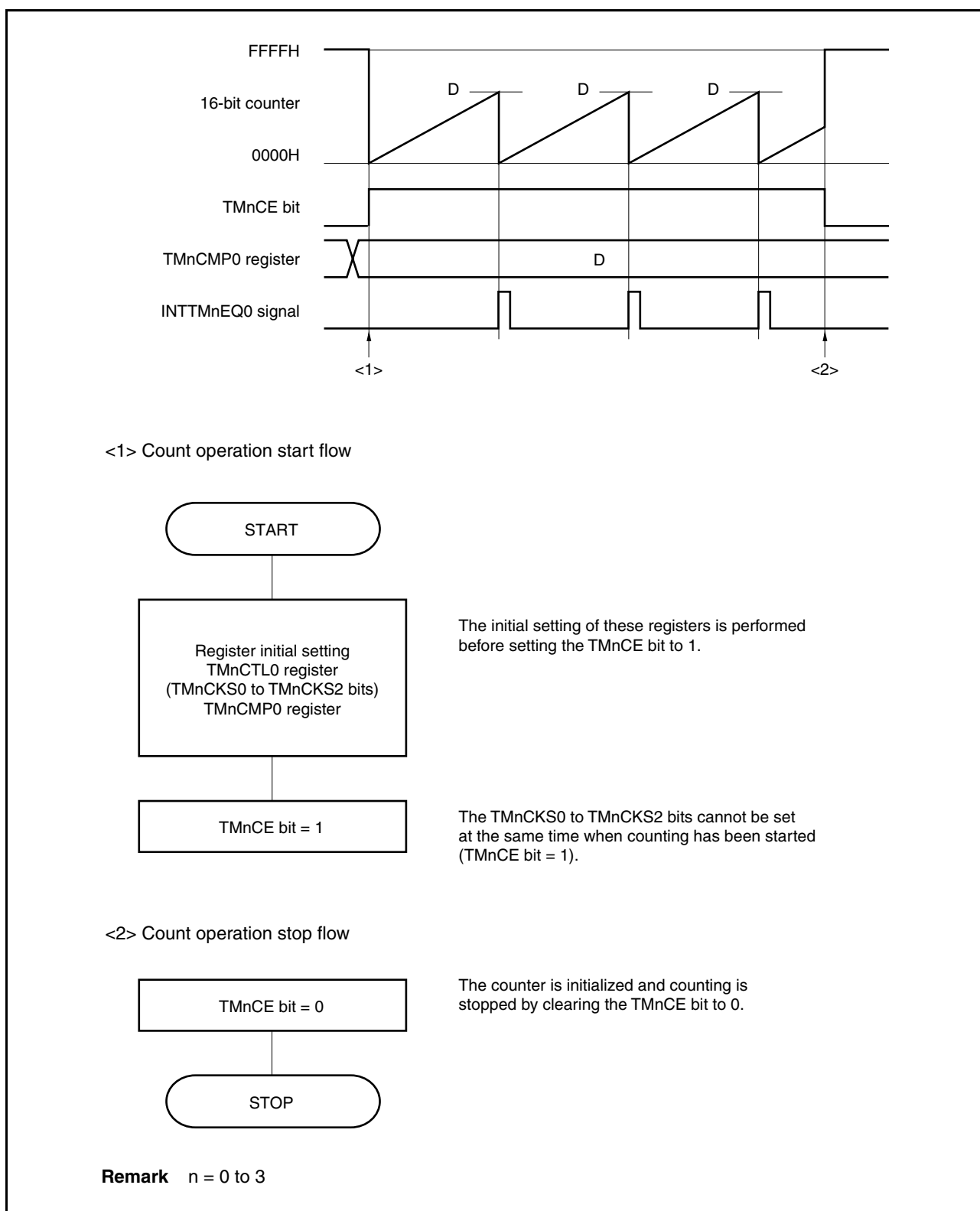
**(2) Interval timer mode operation timing**

　　**Caution　Do not set the TMnCMP0 register to FFFFH.**

**(a) Operation if TMnCMP0 register is set to 0000H**
If the TMnCMP0 register is set to 0000H, the INTTMnEQ0 signal is generated at each count clock.
The value of the 16-bit counter is always 0000H.



**Remark**　n = 0 to 3
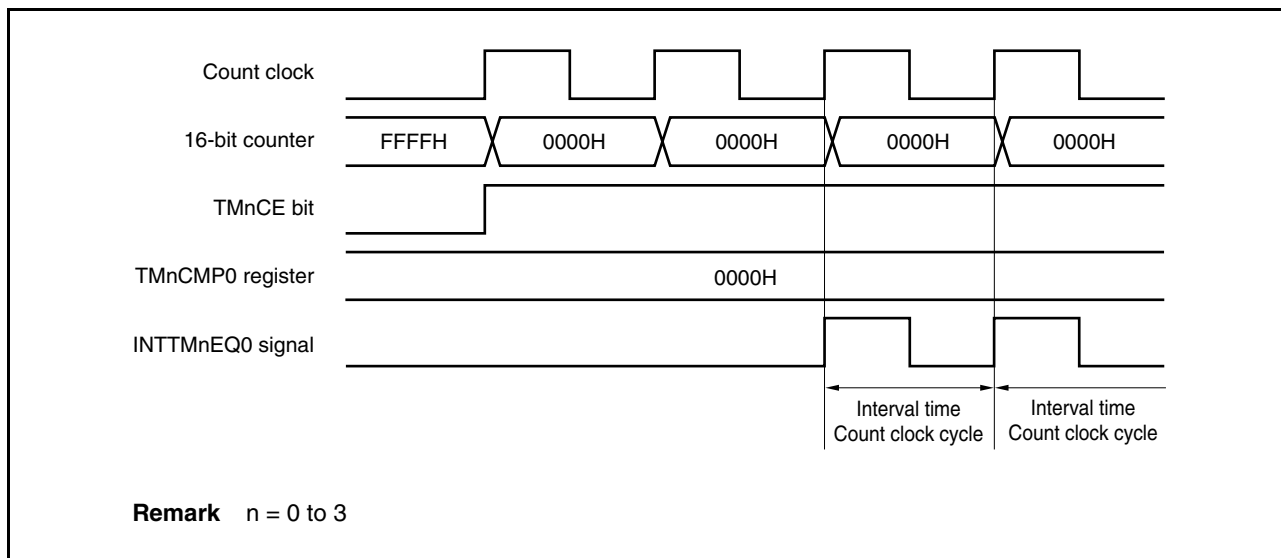
**(b) Operation if TMnCMP0 register is set to N**
If the TMnCMP0 register is set to N, the 16-bit counter counts up to N. The counter is cleared to 0000H in synchronization with the next count-up timing and the INTTMnEQ0 signal is generated.



**Remark**　n = 0 to 3
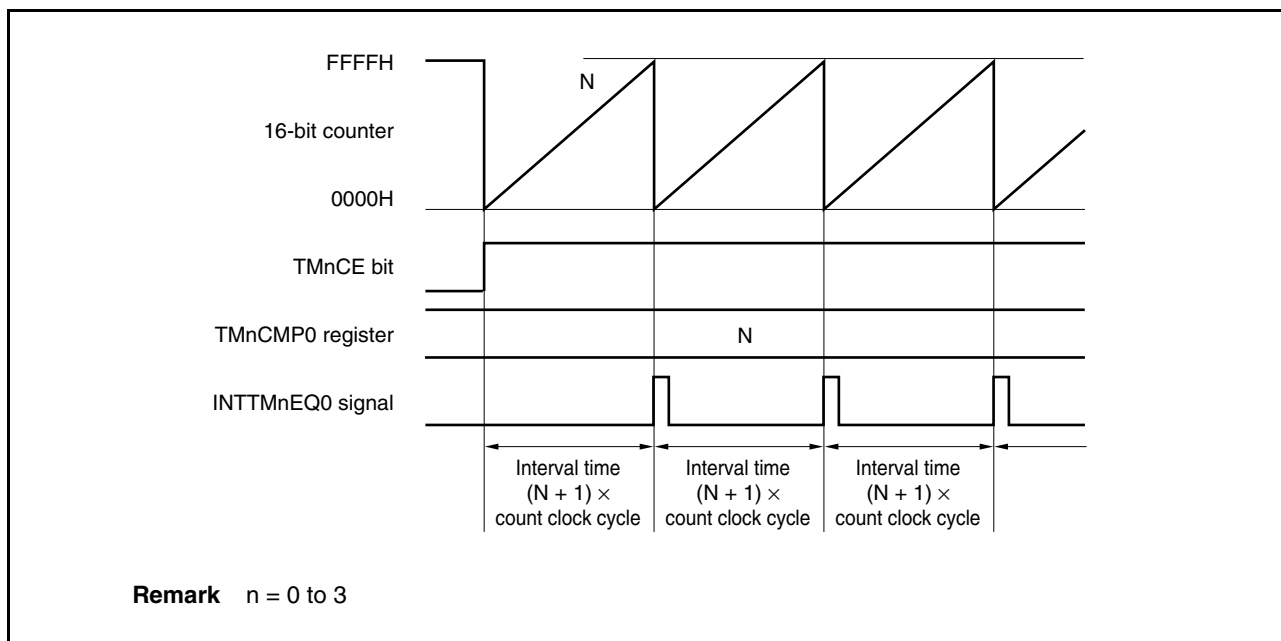
#### 10.4.2 Cautions

(1) It takes the 16-bit counter up to the following time to start counting after the TMnCTL0.TMnCE bit is set to 1, depending on the count clock selected.

(n = 0)

| Selected Count Clock | Maximum Time Before Counting Start |
|---|---|
| $f_{XX}$ | $2/f_{XX}$ |
| $f_{XX}/2$ | $3/f_{XX}$ |
| $f_{XX}/4$ | $6/f_{XX}$ |
| $f_{XX}/64$ | $128/f_{XX}$ |
| $f_{XX}/512$ | $1024/f_{XX}$ |
| $f_{XX}/1024$ | $2048/f_{XX}$ |
| $f_R/8$ | $16/f_R$ |
| $f_{XT}$ | $2/f_{XT}$ |

(n = 1 to 3)

| Selected Count Clock | Maximum Time Before Counting Start |
|---|---|
| $f_{XX}/2$ | $4/f_{XX}$ |
| $f_{XX}/4$ | $6/f_{XX}$ |
| $f_{XX}/8$ | $12/f_{XX}$ |
| $f_{XX}16$ | $32/f_{XX}$ |
| $f_{XX}/64$ | $128/f_{XX}$ |
| $f_{XX}/256$ | $512/f_{XX}$ |
| $f_{XX}/512$ | $1024/f_{XX}$ |
| $f_{XX}/1024$ | $2048/f_{XX}$ |

(2) Rewriting the TMnCMP0 and TMnCTL0 registers is prohibited while TMMn is operating.
If these registers are rewritten while the TMnCE bit is 1, the operation cannot be guaranteed.
If they are rewritten by mistake, clear the TMnCTL0.TMnCE bit to 0, and re-set the registers.

**Remark** n = 0 to 3

## CHAPTER 11  MOTOR CONTROL FUNCTION

### 11.1  Functional Overview

Timer AB1 (TAB1) and the TMQ0 option (TMQOP0) can be used as an inverter function that controls a motor.  It performs a tuning operation with timer AA4 (TAA4) and A/D conversion of the A/D converter can be started when the value of TAB1 matches the value of TAA4.  The following operations can be performed as motor control functions.

- 6-phase PWM output function with 16-bit accuracy
- Timer tuning operation function (tunable with TAA4)
- Cycle setting function (cycle can be changed during operation of crest or valley interrupt)
- Compare register rewriting: Anytime rewrite, batch rewrite, or intermittent rewrite (selectable during TAB1 operation)
- Interrupt and transfer culling functions
- Dead-time setting function
- A/D trigger timing function of the A/D converter
- 0% output and 100% output available
- 0% output and 100% output selectable by crest interrupt and valley interrupt
- Forcible output stop function
  - When valid edge detected by external pin input (TOAB1OFF, TOAA1OFF)
  - When main clock oscillation stop detected by clock monitor function

## 11.2 Configuration

The motor control function consists of the following hardware.

| Item | Configuration |
|---|---|
| Timer register | Dead-time counters |
| Compare register | TAB1 dead-time compare register (TAB1DTC register) |
| Control registers | TAB1 option register 1 (TAB1OPT1)<br>TAB1 option register 2 (TAB1OPT2)<br>TAB1 I/O control register 3 (TAB1IOC3)<br>High-impedance output control register 0 (HZA0CTL0)<br>High-impedance output control register 1 (HZA0CTL1) |

- 6-phase PWM output can be produced with dead time by using the output of TAB1 (TOAB11, TOAB12, TOAB13).
- The output level of the 6-phase PWM output can be set individually.
- The 16-bit timer/counter of TAB1 counts up/down triangular waves. When the timer/counter underflows and when a cycle match occurs, an interrupt is generated. Interrupt generation, however, can be suppressed up to 31 times.
- TAA4 can execute counting at the same time as TAB1 (timer tuning operation function). TAA4 can be set in three ways as it can generate an A/D trigger source (TABTADT0) and two types of interrupts: a TAB1 underflow interrupt (INTTAB1OV) and a cycle match interrupt (INTTAB1CC0).

**Figure 11-1. Block Diagram of Motor Control**

**Figure 11-2. TMQ1 Option**



Note TOAB11, TOAB12, and TOAB13 function alternately as output pins.
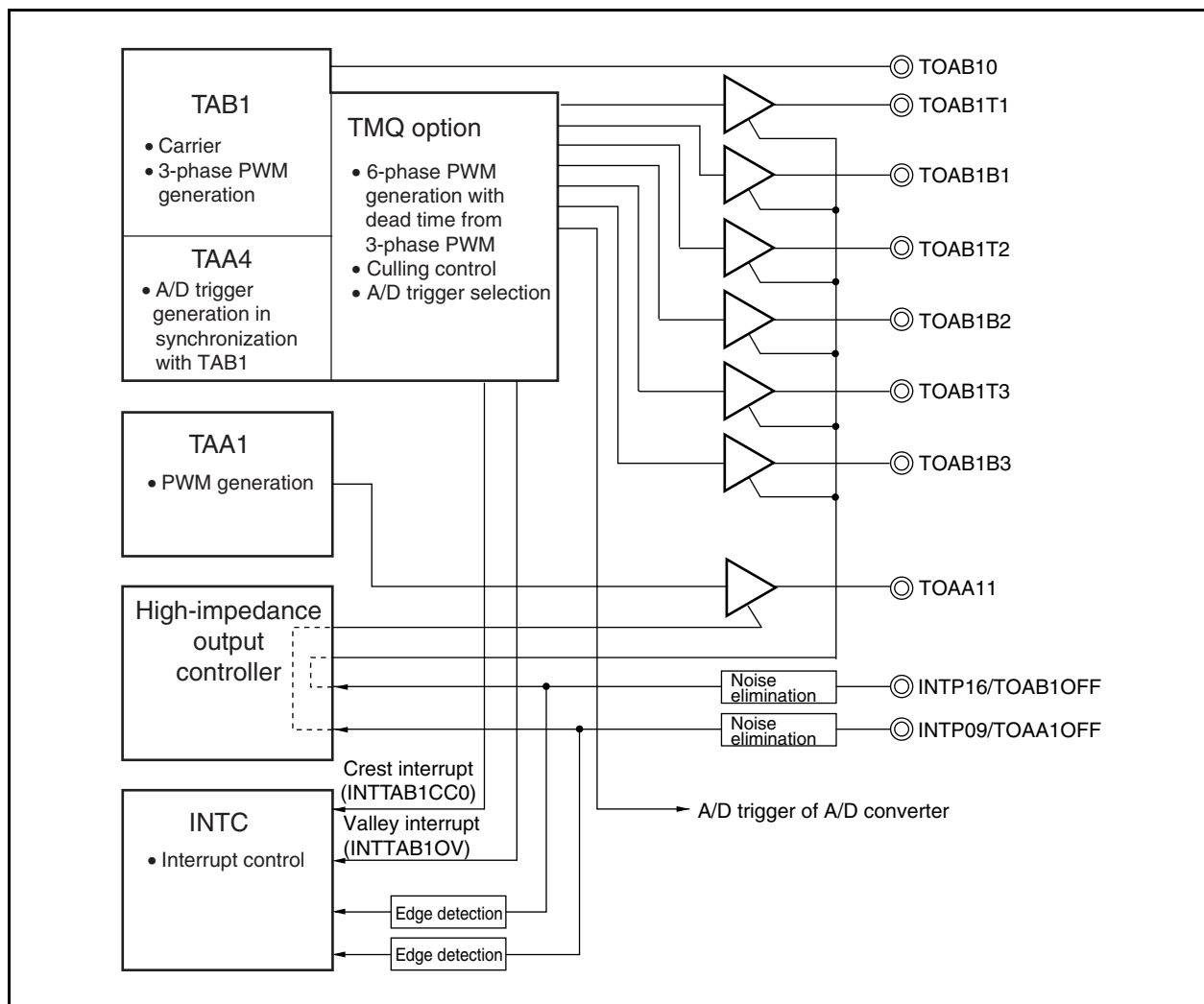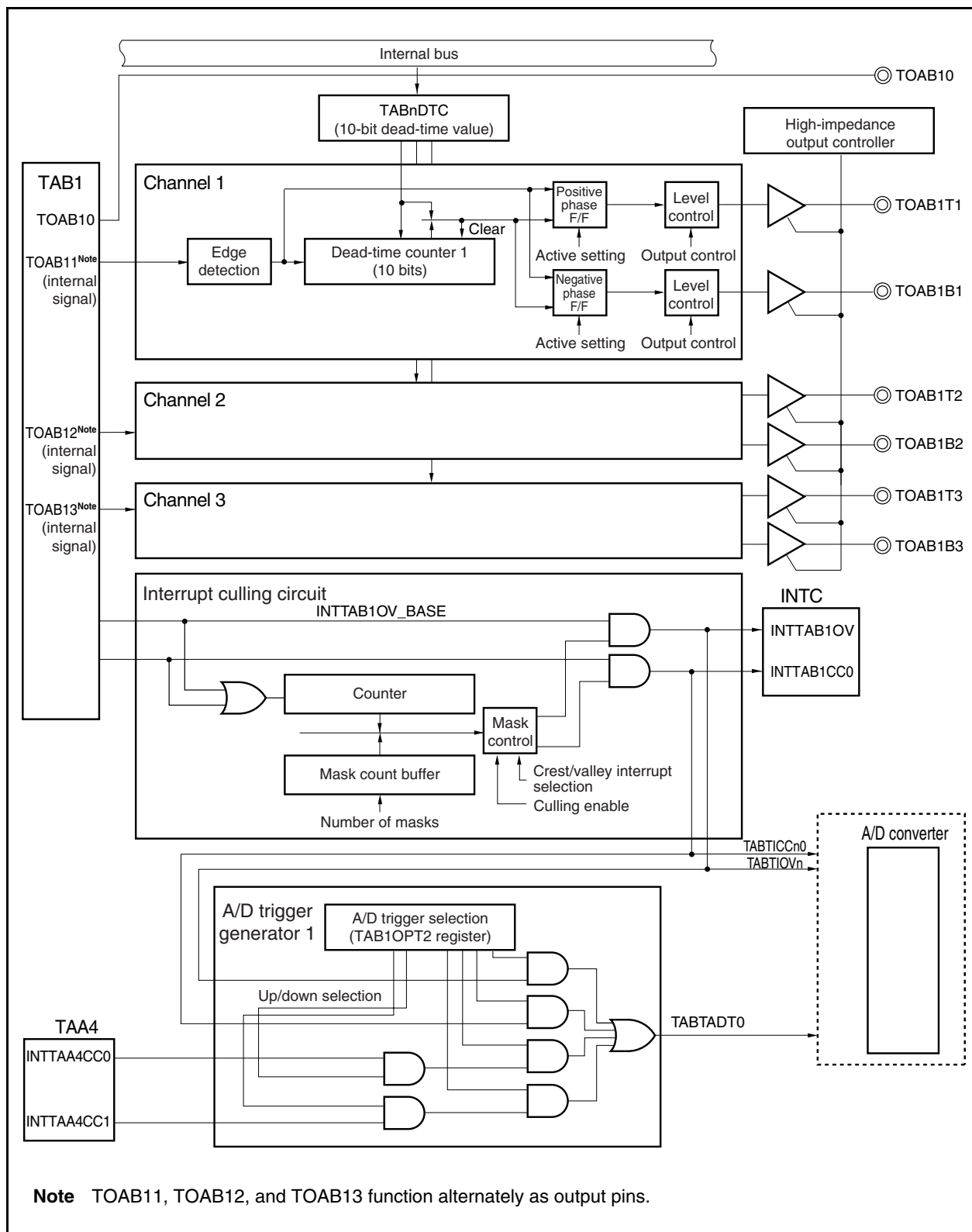
**(1) TAB1 dead-time compare register (TAB1DTC)**

The TAB1DTC register is a 10-bit compare register that specifies the dead-time value.

Rewriting this register is prohibited when the TAB1CTL0.TAB1CE bit = 1.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

**Caution    When generating a dead-time period, set the TAB1DTC register to 1 or higher.**
**Note, when the operation is stopped (TAB1CTL0.TAB1CE bit = 0), a dead-time period is not**
**generated, so the output levels of the TOAB1T1 to TOAB1T3 and TOAB1B1 to TOAB1B3 pins are**
**in their default states.  Therefore, for the protection of the system, take measures such as making**
**the TOAB1T1 to TOAB1T3 and TOAB1B1 to TOAB1B3 pins go into a high-impedance state before**
**stopping operation, or setting the output levels of the pins before switching port modes.**
**When a dead-time period is not needed, set the TAB1DTC register to 0.**

| | After reset: 0000H | R/W | Address:  FFFFF584H | |
|---|---|---|---|---|
| | 15 | 10 9 | | 0 |
| TAB1DTC | 000000 | | TAB1DTC9 to TAB1DTC0 | |
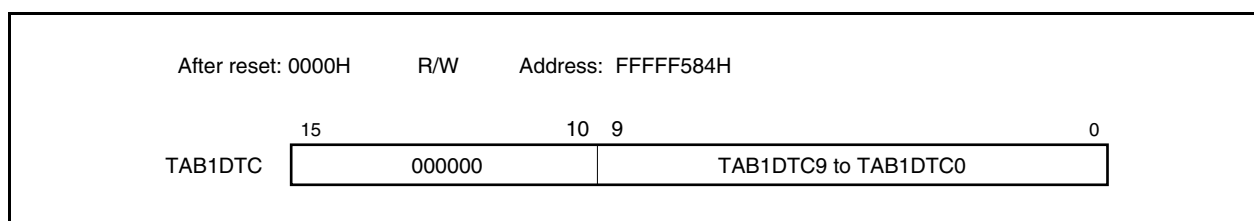
**(2)  Dead-time counters 1 to 3**

The dead-time counters are 10-bit counters that count dead time.

These counters are cleared or count up at the rising or falling edge of the TOAB1m output signal of TAB1, and are cleared or stopped when their count value matches the value of the TAB1DTC register.  The count clock of these counters is the same as that set by the TAB1CTL0.TAB1CKS2 to TAB1CTL0.TAB1CKS0 bits of TAB1.

**Remarks 1.**  The operation differs when the TAB1OPT2.TAB1DTM bit = 1.  For details, see **11.4.2 (4)  Automatic dead-time width narrowing function (TAB1OPT2.TAB1DTM bit = 1)**.

**2.**  $m = 1$ to 3

## 11.3  Control Registers

**(1)  TAB1 option register 1 (TAB1OPT1)**

The TAB1OPT1 register is an 8-bit register that controls the interrupt request signal generated by the timer Q option function.

This register can be rewritten when the TAB1CTL0.TAB1CE bit is 1.

Two rewrite modes (batch write mode and anytime write mode) can be selected, depending on the setting of the TAB1OPT0.TAB1CMS bit.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H        R/W        Address:  FFFFF580H

| | <7> | <6> | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TAB1OPT1 | TAB1ICE | TAB1IOE | 0 | TAB1ID4 | TAB1ID3 | TAB1ID2 | TAB1ID1 | TAB1ID0 |

| TAB1ICE | Crest interrupt (INTTAB1CC0 signal) enable |
|---|---|
| 0 | Do not use INTTAB1CC0 signal (do not use it as count signal for interrupt culling). |
| 1 | Use INTTAB1CC0 signal (use it as count signal for interrupt culling). |

| TAB1IOE | Valley interrupt (INTTAB1OV signal) enable |
|---|---|
| 0 | Do not use INTTAB1OV signal (do not use it as count signal for interrupt culling). |
| 1 | Use INTTAB1OV signal (use it as count signal for interrupt culling). |

| TAB1ID4 | TAB1ID3 | TAB1ID2 | TAB1ID1 | TAB1ID0 | Number of times of interrupt |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Not culled (all interrupts are output) |
| 0 | 0 | 0 | 0 | 1 | 1 masked (one of two interrupts is output) |
| 0 | 0 | 0 | 1 | 0 | 2 masked (one of three interrupts is output) |
| 0 | 0 | 0 | 1 | 1 | 3 masked (one of four interrupts is output) |
| : | : | : | : | : | : |
| 1 | 1 | 1 | 0 | 0 | 28 masked (one of 29 interrupts is output) |
| 1 | 1 | 1 | 0 | 1 | 29 masked (one of 30 interrupts is output) |
| 1 | 1 | 1 | 1 | 0 | 30 masked (one of 31 interrupts is output) |
| 1 | 1 | 1 | 1 | 1 | 31 masked (one of 32 interrupts is output) |

**(2) TAB1 option register 2 (TAB1OPT2)**

The TAB1OPT2 register is an 8-bit register that controls the timer Q option function.

This register can be rewritten when the TAB1CTL0.TAB1CE bit is 1.  However, rewriting the TAB1DTM bit is prohibited when the TAB1CE bit is 1.  The same value can be rewritten.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

(1/2)

After reset: 00H    R/W    Address:  FFFFF581H

| | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| TAB1OPT2 | TAB1RDE | TAB1DTM | TAB1ATM3 | TAB1ATM2 | TAB1AT3 | TAB1AT2 | TAB1AT1 | TAB1AT0 |

| TAB1RDE | Transfer culling enable |
|---|---|
| 0 | Do not cull transfer (transfer timing is generated every time at crest and valley). |
| 1 | Cull transfer at the same interval as interrupt culling set by the TAB1OPT1 register. |

| TAB1DTM | Dead-time counter operation mode selection (m = 1 to 3) |
|---|---|
| 0 | The dead-time counter counts up normally and, if TOAB1m output of TAB1 is at a narrow interval (TOAB1m output width < dead-time width), the dead-time counter is cleared and counts up again. |
| 1 | The dead-time counter counts up normally and, if TOAB1m output of TAB1 is at a narrow interval (TOAB1m output width < dead-time width), the dead-time counter counts down and the dead-time control width is automatically narrowed. |
| | Rewriting the TAB1DTM bit is disabled during timer operation.  If it is rewritten by mistake, stop the timer operation by clearing the TAB1CE bit to 0, and re-set the TAB1DTM bit. |

**Cautions 1.  When using interrupt culling (the TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 bits are set to other than 00000), be sure to set the TAB1RDE bit to 1.**
   **This means that interrupts and transfers are generated at the same timing.  Interrupts and transfers, cannot be set separately.  If interrupts and transfers are set separately (TAB1RDE bit = 0), transfers are not performed normally.**
 **2.  When generating a dead-time period, set the TAB1DTC register to 1 or higher.**
   **Note, when the operation is stopped (TAB1CTL0.TAB1CE bit = 0), a dead-time period is not generated, so the output levels of the TOAB1T1 to TOAB1T3 and TOAB1B1 to TOAB1B3 pins are in their default states.  Therefore, for the protection of the system, take measures such as making the TOAB1T1 to TOAB1T3 and TOAB1B1 to TOAB1B3 pins go into a high-impedance state before stopping operation, or setting the output levels of the pins before switching port modes.**
   **When a dead-time period is not needed, set the TAB1DTC register to 0.**

(2/2)

| TAB1ATM3 | TAB1ATM3 mode selection |
|---|---|
| 0 | Output A/D trigger signal (TABTADT0) for INTTAA4CC1 interrupt while dead-time counter is counting up. |
| 1 | Output A/D trigger signal (TABTADT0) for INTTAA4CC1 interrupt while dead-time counter is counting down. |

| TAB1ATM2 | TAB1ATM2 mode selection |
|---|---|
| 0 | Output A/D trigger signal (TABTADT0) for INTTAA4CC0 interrupt while dead-time counter is counting up. |
| 1 | Output A/D trigger signal (TABTADT0) for INTTAA4CC0 interrupt while dead-time counter is counting down. |

| TAB1AT3[Note] | A/D trigger output control 3 |
|---|---|
| 0 | Disable output of A/D trigger signal (TABTADT0) for INTTAA4CC1 interrupt. |
| 1 | Enable output of A/D trigger signal (TABTADT0) for INTRAA4CC1 interrupt. |

| TAB1AT2[Note] | A/D trigger output control 2 |
|---|---|
| 0 | Disable output of A/D trigger signal (TABTADT0) for INTTAA4CC0 interrupt. |
| 1 | Enable output of A/D trigger signal (TABTADT0) for INTTAA4CC0 interrupt. |

| TAB1AT1[Note] | A/D trigger output control 1 |
|---|---|
| 0 | Disable output of A/D trigger signal (TABTADT0) for INTTAB1CC0 (crest interrupt). |
| 1 | Enable output of A/D trigger signal (TABTADT0) for INTTAB1CC0 (crest interrupt). |

| TAB1AT0[Note] | A/D trigger output control 0 |
|---|---|
| 0 | Disable output of A/D trigger signal (TABTADT0) for INTTAB1OV (valley interrupt). |
| 1 | Enable output of A/D trigger signal (TABTADT0) for INTTAB1OV (valley interrupt). |

**Note**    For the setting of the TAB1AT3 to TAB1AT0 bits, see **CHAPTER 15  A/D CONVERTER**.

**(3) TAB1 I/O control register 3 (TAB1IOC3)**

The TAB1IOC3 register is an 8-bit register that controls the output of the timer Q option function.

To output from the TOAB1Tm pin, set the TAB1IOC0.TAB1OEm bit to 1 and then set the TAB1IOC3 register.

The TAB1IOC3 register can be rewritten only when the TAB1CTL0.TAB1CE bit is 0.

Rewriting each bit of the TAB1IOC3 register is prohibited when the TAB1CTL0.TAB1CE bit is 1; however the same value can be rewritten to each bit of the TAB1IOC3 register when the TAB1CTL0.TAB1CE bit is 1.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to A8H.

**Caution**   **Set the TAB1IOC3 register to the reset value (A8H) when the timer is used in a mode other than the 6-phase PWM output mode.**

**Remarks 1.** Set the output level of the TOAB1Tm pin by using the TAB1IOC0 register.
**2.** m = 1 to 3

After reset: A8H      R/W      Address:  FFFFF582H

| | <7> | <6> | <5> | <4> | <3> | <2> | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TAB1IOC3 | TAB1OLB3 | TAB1OEB3 | TAB1OLB2 | TAB1OEB2 | TAB1OLB1 | TAB1OEB1 | 0 | 0 |

| TAB1OLBm | Setting of TOAB1Bm pin output level (m = 1 to 3) |
|---|---|
| 0 | Disable inversion of output of TOAB1Bm pin |
| 1 | Enable inversion of output of TOAB1Bm pin |

| TAB1OEBm | TOAB1Bm pin output (m = 1 to 3) |
|---|---|
| 0 | Disable TOAB1Bm pin output.<br>• When TAB1OLBm bit = 0, low level is output from TOAB1Bm pin.<br>• When TAB1OLBm bit = 1, high level is output from TOAB1Bm pin. |
| 1 | Enable TOAB1Bm pin output. |

**(a)  Output from TOAB1Tm and TOAB1Bm pins**

The  TOAB1Tm  pin  output  is  controlled  by  the  TAB1IOC0.TAB1OLm  and  TAB1IOC0.TAB1OEm  bits.   The TOAB1Bm pin output is controlled by the TAB1IOC3.TAB1OLBm and TAB1IOC3.TAB1OEBm bits.

The timer output with each setting in the 6-phase PWM output mode is shown below.

**Figure 11-3.  Output Control of TOAB1Tm and TOAB1Bm Pins (Without Dead Time)**
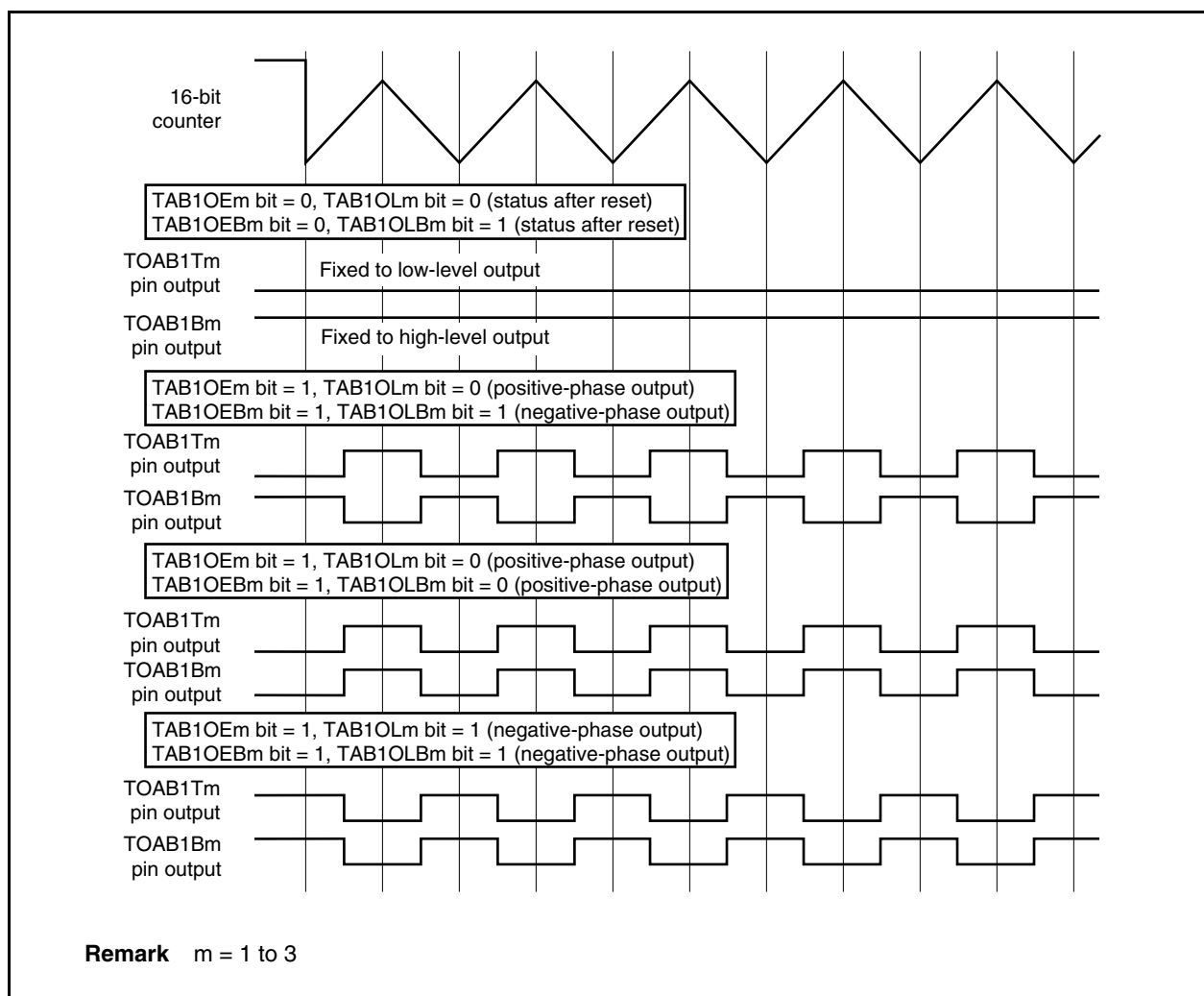


**Remark**   m = 1 to 3

**Table 11-1. TOAB1Tm Pin Output**

| TAB1OLm Bit | TAB1OEm Bit | TAB1CE Bit | TOAB1Tm Pin Output |
|---|---|---|---|
| 0 | 0 | x | Low-level output |
| | 1 | 0 | Low-level output |
| | | 1 | TOAB1Tm positive-phase output |
| 1 | 0 | x | High-level output |
| | 1 | 0 | High-level output |
| | | 1 | TOAB1Tm negative-phase output |

**Remark**　m = 1 to 3

**Table 11-2. TOAB1Bm Pin Output**

| TAB1OLBm Bit | TAB1OEBm Bit | TAB1CE Bit | TOAB1Bm Pin Output |
|---|---|---|---|
| 0 | 0 | x | Low-level output |
| | 1 | 0 | Low-level output |
| | | 1 | TOAB1Bm positive-phase output |
| 1 | 0 | x | High-level output |
| | 1 | 0 | High-level output |
| | | 1 | TOAB1Bm negative-phase output |

**Remark**　m = 1 to 3

**(6)　High-impedance output control registers 0, 1 (HZA0CTL0, HZA0CTL1)**

The HZA0CTL0 and HZA0CTL1 registers are 8-bit registers that control the high-impedance state of the output buffer.

These registers can be read or written in 8-bit or 1-bit units.  However, the HZA0DCFn bit is a read-only bit and cannot be written.

16-bit access is not possible.

Reset sets these registers to 00H.

Software can be used to always write the same value to the HZA0CTLn register.

The relationship between detection factor and the control registers is shown below.

| Pins Subject to High-Impedance Control | High-Impedance Control Factor | Control Register |
|---|---|---|
| | External Pin | |
| When TOAB1T1 to TOAB1T3 are output<br>When TOAB1B1 to TOAB1B3 are output | TOAB1OFF/INTP16 | HZA0CTL0 |
| When TOAA11 is output | TOAA1OFF/INTP09 | HZA0CTL1 |

**Caution**　**High impedance control is performed only when the target port is specified as a target pin in the above table.**

**Remark**　n = 0, 1

(1/2)

After reset: 00H     R/W     Address: HZA0CTL0 FFFFF590H, HZA0CTL1 FFFFF591H

| | <7> | <6> | 5 | 4 | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| HZA0CTLn | HZA0DCEn | HZA0DCMn | HZA0DCNn | HZA0DCPn | HZA0DCTn | HZA0DCCn | 0 | HZA0DCFn |

(n = 0, 1)

| HZA0DCEn | High-impedance output control |
|---|---|
| 0 | Disable high-impedance output control operation. Pins can function as output pins. |
| 1 | Enable high-impedance output control operation. |

| HZA0DCMn | Condition of clearing high-impedance state by HZA0DCCn bit |
|---|---|
| 0 | Setting of the HZA0DCCn bit is valid regardless of the external pin input. |
| 1 | Setting of the HZA0DCCn bit is invalid while the external pin input holds a level detected as abnormal (active level). |
| Rewrite the HZA0DCMn bit when the HZA0DCEn bit = 0. | |

| HZA0DCNn | HZA0DCPn | External pin input edge specification |
|---|---|---|
| 0 | 0 | No valid edge (setting the HZA0DCFn bit by external pin input is prohibited). |
| 0 | 1 | Rising edge of the external pin is valid (abnormality is detected by rising edge input). |
| 1 | 0 | Falling edge of the external pin is valid (abnormality is detected by falling edge input). |
| 1 | 1 | Setting prohibited |

- Rewrite the HZA0DCNn and HZA0DCPn bits when the HZA0DCEn bit is 0.
- For the valid edge specification of the interrupts of the INTP09 and INTP16 pins, see **23.6.2 (3) External interrupt falling, rising edge specification register 3 (INTR3, INTF3)** and **(6) External interrupt falling, rising edge specification register 9H (INTR9, INTF9)**.
- For the edge specification of the external pins, begin with the TOAB1OFF and TOAA1OFF pins. Then, perform edge specification for the external pins other than the TOAB1OFF and TOAA1OFF pins. Otherwise, an undefined edge may be detected when the edge for the TOAB1OFF and TOAA1OFF pins is specified.
- High-impedance output control is performed when the valid edge is input after the operation is enabled (by setting HZA0DCEn bit to 1). If the external pin is at the active level when the operation is enabled, therefore, high-impedance output control is not performed.

| HZA0DCTn | High-impedance output trigger bit |
|---|---|
| 0 | No operation |
| 1 | Pins are made to go into a high-impedance state by software and the HZA0DCFn bit is set to 1. |

- If an edge indicating abnormality is input to the external pin (which is detected according to the setting of the HZA0DCNn and HZA0DCPn bits), the HZA0DCTn bit is invalid even if it is set to 1.
- The HZA0DCTn bit is always 0 when it is read because it is a software-triggered bit.
- The HZA0DCTn bit is invalid even if it is set to 1 when the HZA0DCEn bit = 0.
- Simultaneously setting the HZA0DCTn and HZA0DCCn bits to 1 is prohibited.

| HZA0DCCn | High-impedance output control clear bit |
|---|---|
| 0 | No operation |
| 1 | Pins that have gone into a high-impedance state are output-enabled by software and the HZA0DCFn bit is cleared to 0. |

- Pins can function as output pins when the HZA0DCM bit = 0, regardless of the status of the external pin.
- If an edge indicating abnormality is input to the external pin (which is set by the HZA0DCNn and HZA0DCPn bits) when the HZA0DCM bit = 1, the HZA0DCCn bit is invalid even if it is set to 1.
- The HZA0DCCn bit is always 0 when it is read.
- The HZA0DCCn bit is invalid even if it is set to 1 when the HZA0DCEn bit = 0.
- Simultaneously setting the HZA0DCTn and HZA0DCCn bits to 1 is prohibited.

| HZA0DCFn | High-impedance output status flag |
|---|---|
| Clear (0) | Indicates that output of the pin is enabled.<br>• This bit is cleared to 0 when the HZA0DCEn bit = 0.<br>• This bit is cleared to 0 when the HZA0DCCn bit = 1. |
| Set (1) | Indicates that the pin goes into a high-impedance state.<br>• This bit is set to 1 when the HZA0DCTn bit = 1.<br>• This bit is set to 1 when an edge indicating abnormality is input to the external pin (which is detected according to the setting of the HZA0DCNn and HZA0DCPn bits). |

**Figure 11-4.  High-Impedance Output Controller Configuration**



Remark   Also see Figures 11-1 and 11-2.
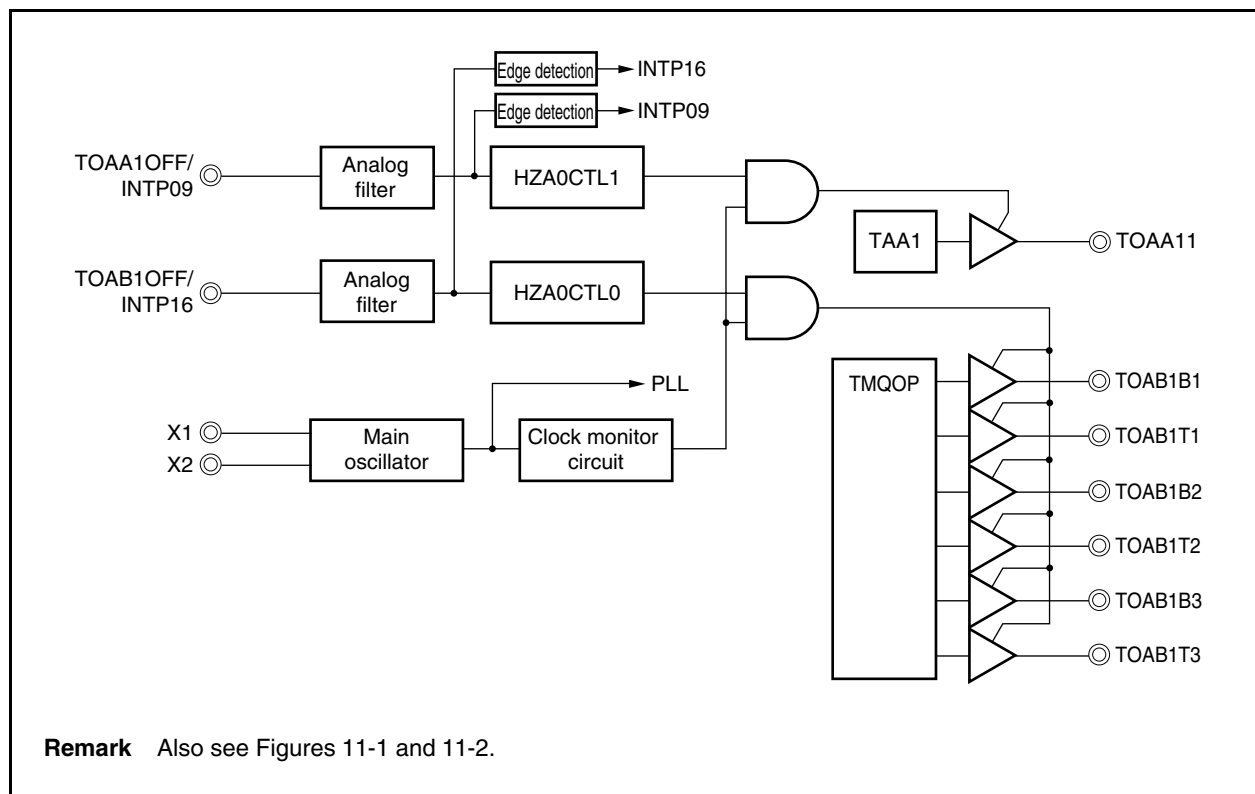
### (a) Setting procedure

#### (i) Setting of high-impedance control operation
<1>  Set the HZA0DCMn, HZA0DCNn, and HZA0DCPn bits.
<2>  Set the HZA0DCEn bit to 1 (enable high-impedance control).

#### (ii) Changing setting after enabling high-impedance control operation
<1>  Clear the HZA0DCEn bit to 0 (to stop the high-impedance control operation).
<2>  Change the setting of the HZA0DCMn, HZA0DCNn, and HZA0DCPn bits.
<3>  Set the HZA0DCEn bit to 1 (to enable the high-impedance control operation again).

#### (iii) Resuming output when pins are in high-impedance state
If the HZA0DCMn bit is 1, set the HZA0DCCn bit to 1 to clear the high-impedance state after the valid edge of the external pin is detected.  However, the high-impedance state cannot be cleared unless this bit is set while the input level of the external pin is inactive.
<1>  Set the HZA0DCCn bit to 1 (command signal to clear the high-impedance state).
<2>  Read the HZA0DCFn bit and check the flag status.
<3>  Return to <1> if the HZA0DCFn bit is 1.  The input level of the external pin must be checked.
      The pin can function as an output pin if the HZA0DCFn bit is 0.

#### (iv) Making pin go into high-impedance state by software
The HZA0DCTn bit must be set to 1 by software to make the pin go into a high-impedance state while the input level of the external pin is inactive.  The following procedure is an example in which the setting is not dependent upon the setting of the HZA0DCMn bit.
<1>  Set the HZA0DCTn bit to 1 (high-impedance output command).
<2>  Read the HZA0DCFn bit to check the flag status.
<3>  Return to <1> if the HZA0DCFn bit is 0.  The input level of the external pin must be checked.
      The pin is in a high-impedance state if the HZA0DCFn bit is 1.

However, if the external pin is not used with the HZA0DCPn bit and HZA0DCNn bit cleared to 0, the pin goes into a high-impedance state when the HZA0DCTn bit is set to 1.

**Remark**   n = 0, 1

## 11.4 Operation

### 11.4.1 System outline

**(1) Outline of 6-phase PWM output**

The 6-phase PWM output mode is used to generate a 6-phase PWM output wave, by using the timer AB1 (TAB1) and the TMQ option (TMQOPA) in combination.

The 6-phase PWM output mode is enabled by setting the TAB1CTL1.TAB1MD2 to TAB1CTL1.TAB1MD0 bits of TAB1 to "111".

One 16-bit counter and four 16-bit compare registers of TAB1 are used to generate a basic 3-phase wave.

The functions of the compare registers are as follows.

TAA4 can perform a tuning operation with TAB1 to generate a conversion trigger source for the A/D converter.

| Compare Register | Function | Settable Range |
|---|---|---|
| TAB1CCR0 register | Setting of cycle | $0002H \leq m \leq FFFEH$ |
| TAB1CCR1 register | Specifying output width of phase U | $0000H \leq i \leq m + 1$ |
| TAB1CCR2 register | Specifying output width of phase V | $0000H \leq j \leq m + 1$ |
| TAB1CCR3 register | Specifying output width of phase W | $0000H \leq k \leq m + 1$ |

**Remark**　m = Set value of TAB1CCR0 register
　　　　　　　i = Set value of TAB1CCR1 register
　　　　　　　j = Set value of TAB1CCR2 register
　　　　　　　k = Set value of TAB1CCR3 register

A dead-time interval is generated from the basic 3-phase wave generated by using three 10-bit dead-time counters and one compare register to create a wave with a reverse phase to that of the basic 3-phase wave. Then a 6-phase PWM output wave (U, $\overline{U}$, V, $\overline{V}$, W, and $\overline{W}$) is generated.

The 16-bit counter for generating the basic 3-phase wave counts up or down. After the operation has been started, this counter counts up. When its count value matches the cycle set to the TAB1CCR0 register, the counter starts counting down. When the count value matches 0001H, the counter counts up again. This means that a value two times higher than the value set to the TAB1CCR0 register + 1 is the carrier cycle.

10-bit dead-time counters 1 to 3, which generate the dead-time interval, count up. Therefore, the value set to the TAB1 dead-time compare register (TAB1DTC) is used as a dead-time value as is. Because three counters are used, dead time can be generated independently in phases U, V, and W. However, because there is only one register that specifies a dead-time value (TAB1DTC), the same dead-time value is used in all three phases.

**Figure 11-5.  Outline of 6-Phase PWM Output Mode**



**Note**  TOAB11, TOAB12, and TOAB13 function alternately as output pins.
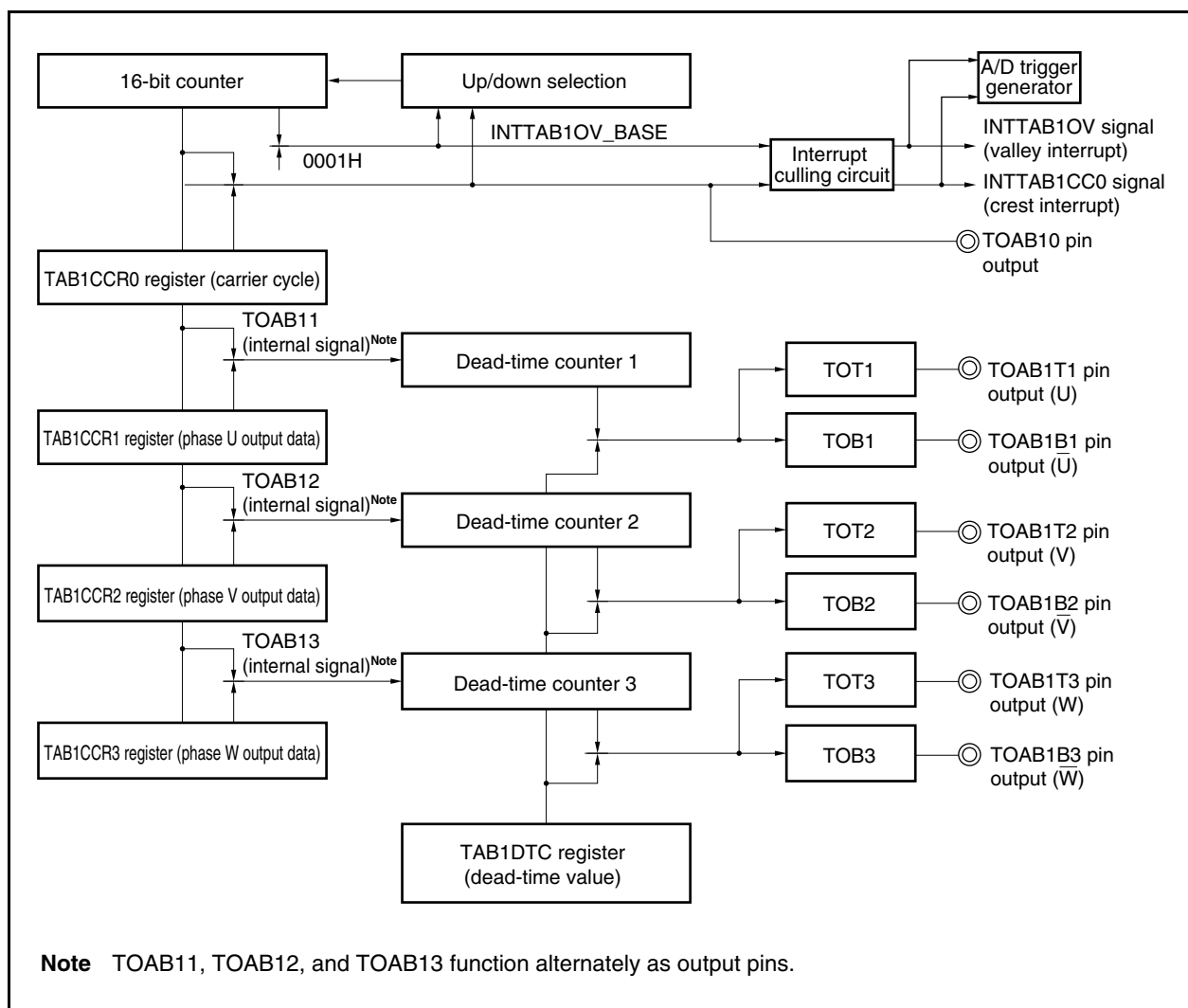
**Figure 11-6. Timing Chart of 6-Phase PWM Output Mode**



Carrier cycle = (M + 1) × 2

Basic phase U output width = (M + 1 - i) × 2

Basic phase V output width = (M + 1 − j) × 2

Basic phase W output width = (M + 1 − k) × 2

Phase U output width = (M + 1 − i) × 2 − N

Phase $\overline{U}$ output width = (M + 1 − i) × 2 + N

Phase V output width = (M + 1 − j) × 2 − N

Phase $\overline{V}$ output width = (M + 1 − j) × 2 + N

Phase W output width = (M + 1 − k) × 2 − N

Phase $\overline{W}$ output width = (M + 1 − k) × 2 + N

Dead-time width = N
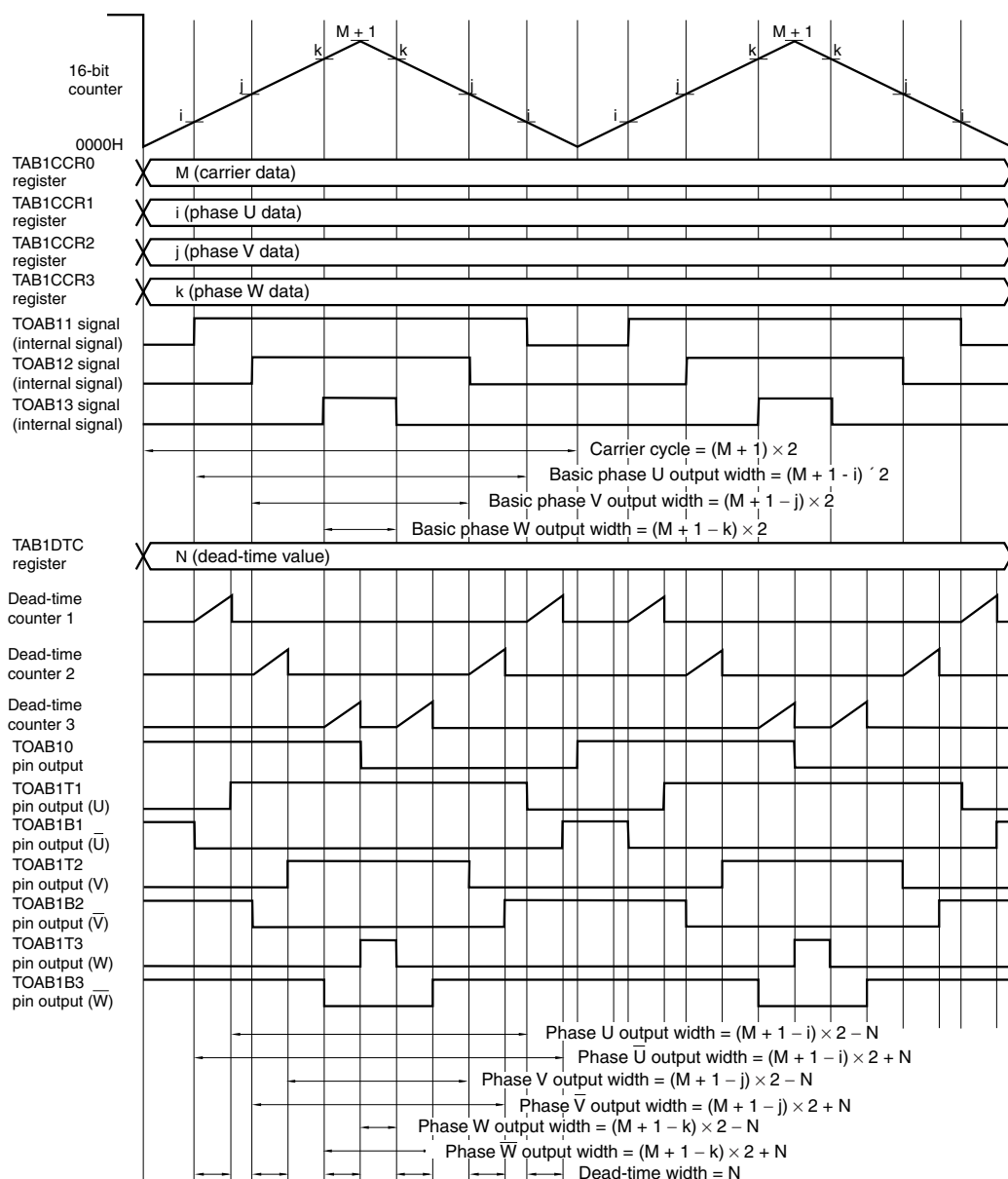
**Cautions 1.** Set the value "M" of the TAB1CCR0 register in a range of 0002H ≤ M ≤ FFFEH in the 6-phase PWM output mode.

**2.** Only a value of up to "M + 1" can be set to the TAB1CCR1, TAB1CCR2, and TAB1CCR3 registers.

**3.** The output is 100% if "0000H" is set to the TAB1CCR1, TAB1CCR2, and TAB1CCR3 registers. The output is 0% if "M + 1" is set to the TAB1CCR1, TAB1CCR2, and TAB1CCR3 registers.

The output (duty 50%) rises at the crest (M + 1) of the 16-bit counter and falls at the valley (0000H) if "M + 2" or higher is set to the TAB1CCR1, TAB1CCR2, and TAB1CCR3 registers.

**4.** If the operation value of an equation (such as (M + 1 − i) × 2 − N) of the output width of phases U, V, and W is 0 or lower, it is converged to 0 (100% output). If the operation value is higher than "(M + 1) × 2", it is converged to (M + 1) × 2 (0% output).

**(2) Interrupt requests**

Two types of interrupt requests are available: the INTTAB1CC0 (crest interrupt) signal and INTTAB1OV (valley interrupt) signal.

The INTTAB1CC0 and INTTAB1OV signals can be culled by using the TAB1OPT1 register.

For details of culling interrupts, see **11.4.3 Interrupt culling function**.

- INTTAB1CC0 (crest interrupt) signal: An interrupt signal indicating a match between the value of the 16-bit counter that counts up and the value of the TAB1CCR0 register
- INTTAB1OV (valley interrupt) signal: An interrupt signal indicating a match between the value of the 16-bit counter that counts down and the value 0001H

**(3) Rewriting registers during timer operation**

The following registers have a buffer register and can be rewritten in the anytime rewrite mode, batch rewrite mode, or intermittent batch rewrite mode.

| Related Unit | Register |
|---|---|
| Timer AA1 | TAA1 capture/compare register 0 (TAA1CCR0) |
| | TAA1 capture/compare register 1 (TAA1CCR1) |
| Timer AB1 | TAB1 capture/compare register 0 (TAB1CCR0) |
| | TAB1 capture/compare register 1 (TAB1CCR1) |
| | TAB1 capture/compare register 2 (TAB1CCR2) |
| | TAB1 capture/compare register 3 (TAB1CCR3) |
| Timer Q1 option | TAB1 option register 1 (TAB1OPT1) |

For details of the transfer function of the compare register, see **11.4.4 Operation to rewrite register with transfer function**.

**(4) Counting-up/down operation of 16-bit counter**

The operation status of the 16-bit counter can be checked by using the TAB1CUF bit of TAB1 option register 0 (TAB1OPT0).

| Status of TAB1CUF Bit | Status of 16-Bit Counter | Range of 16-Bit Counter Value |
|---|---|---|
| TAB1CUF bit = 0 | Counting up | 0000H – m |
| TAB1CUF bit = 1 | Counting down | (m + 1) – 0001H |

**Remark** m = Set value of TAB1CCR0 register

**Figure 11-7.  Interrupt and Up/Down Flag**
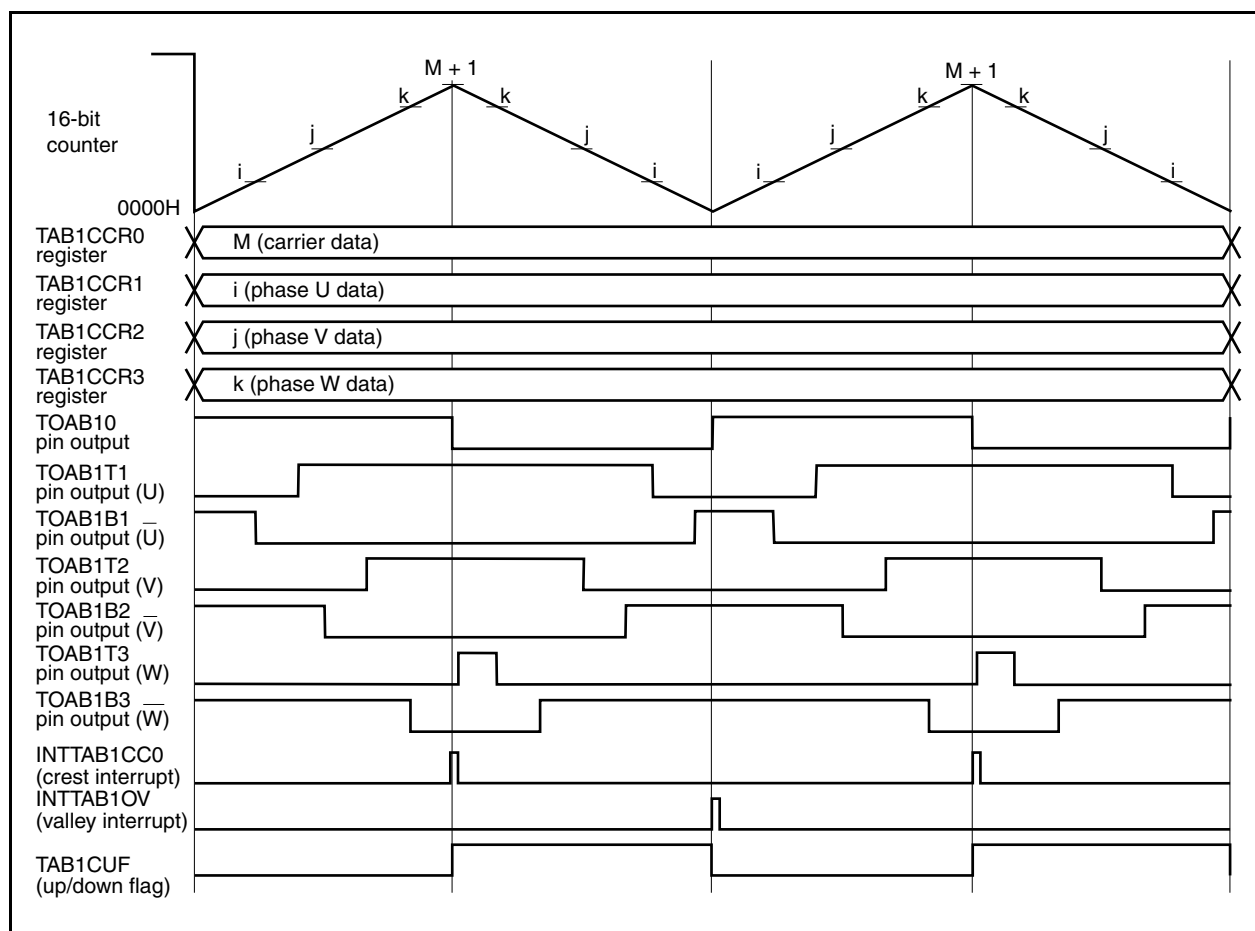
### 11.4.2  Dead-time control (generation of negative-phase wave signal)

**(1) Dead-time control mechanism**

In the 6-phase PWM output mode, compare registers 1 to 3 (TAB1CCR1, TAB1CCR2, and TAB1CCR3) are used to set the duty factor, and compare register 0 (TAB1CCR0) is used to set the cycle. By setting these four registers and by starting the operation of TAB1, three types of PWM output waves (basic 3-phase waves) with a variable duty factor are generated. These three PWM output waves are input to the timer Q option unit (TMQOP) and their inverted signal with dead-time is created to generate three sets of (six) PWM waves.

The TMQOP unit consists of three 10-bit counters (dead-time counters 1 to 3) that operate in synchronization with the count clock of TAB1, and a TAB1 dead-time compare register (TAB1DTC) that specifies dead time. If "a" is set to the TAB1DTC register, the dead-time value is "a", and interval "a" is created between a positive-phase wave and a negative-phase wave.

**Figure 11-8.  PWM Output Wave with Dead Time (1)**



**(a)  When dead time is inserted (TAB1DTC register = a)**

**(b)  No dead time (TAB1DTC register = 000H)**

**Remark**    m = 1 to 3
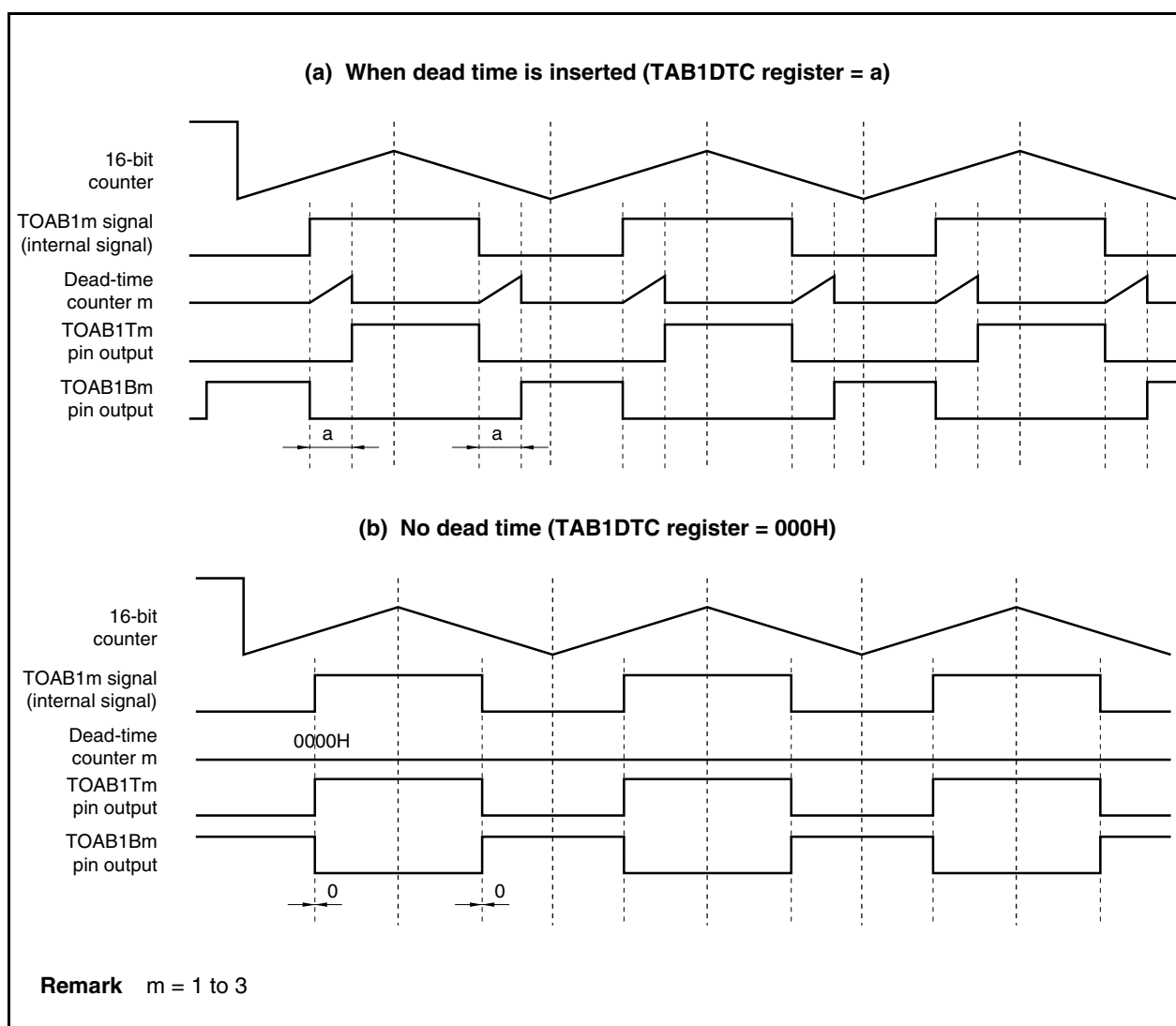
**(2) PWM output of 0%/100%**

The V850ES/V850ES/JG3-U and V850ES/JH3-U are capable of 0% wave output and 100% wave output for PWM output.

A low level is continuously output from the TOAB1Tm pin as the 0% wave output. A high level is continuously output from the TOAB1Tm pin as the 100% wave output.

A 0% wave is output by setting the TAB1CCRm register to "M + 1" when the TAB1CCR0 register = M.

A 100% wave is output by setting the TAB1CCRm register to "0000H".

Rewriting the TAB1CCRm register is enabled while the timer is operating, and 0% wave output or 100% wave output can be selected at the point of the crest interrupt (INTTAB1CC0) and valley interrupt (INTTAB1OV).

**Remark**  m = 1 to 3

**Figure 11-9.  0% PWM Output Waveform (With Dead Time)**
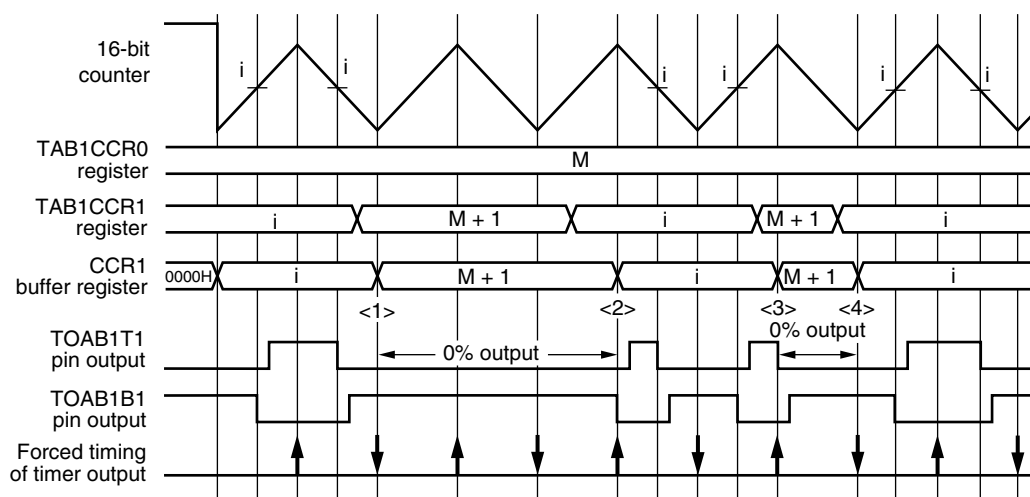


<1>  0% output is selected by the valley interrupt (without a match with the 16-bit counter).

The valley interrupt forcibly lowers the timer output. This produces the 0% output.

<2>  0% output is canceled by the crest interrupt (without a match with the 16-bit counter).

The crest interrupt forcibly raises the timer output. This cancels the 0% output.

<3>  0% output is selected by the crest interrupt (with a match with the 16-bit counter).

The crest interrupt forcibly raises the timer output, but lowering the timer output takes precedence when the value of the TAB1CCRm register matches the value of the 16-bit counter. As a result, the 0% wave is output.

<4>  0% output is canceled by the valley interrupt (without a match with the 16-bit counter).

The valley interrupt forcibly lowers the timer output. This cancels the 0% output.

**Remarks 1.**  ↑ means forcible raising and ↓ means forcible lowering.
**2.**  m = 1 to 3

**Figure 11-10.  100% PWM Output Waveform (With Dead Time)**
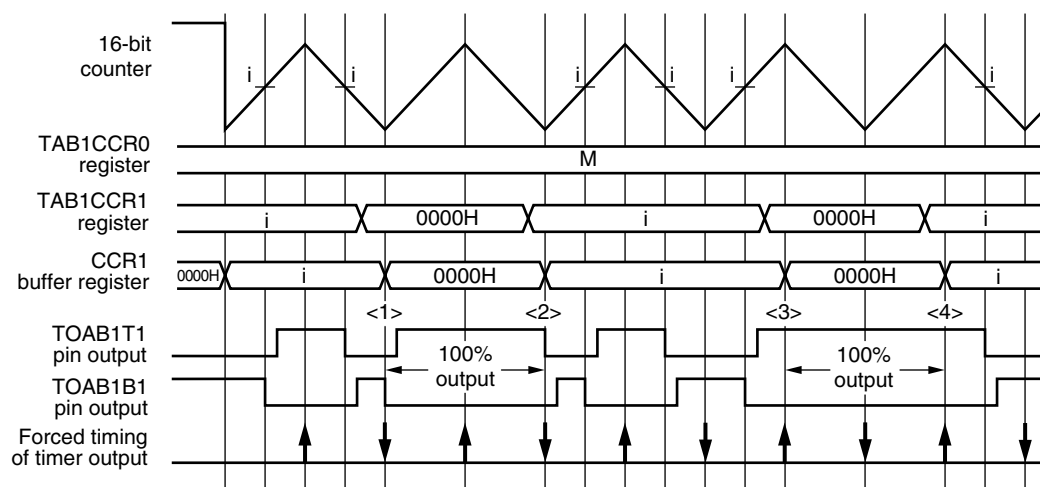


<1>  100% output is selected by the valley interrupt (with a match with the 16-bit counter).

     The valley interrupt forcibly lowers the timer output, but raising the timer output takes precedence when the value of the TAB1CCRm register matches the value of the 16-bit counter.  As a result, the 100% output is produced.

<2>  100% output is canceled by the valley interrupt (without a match with the 16-bit counter).

     The valley interrupt forcibly lowers the timer output.  This cancels the 100% output.

<3>  100% output is selected by the crest interrupt (without a match with the 16-bit counter).

     The crest interrupt forcibly raises the timer output.  This produces the 100% output.

<4>  100% output is canceled by the crest interrupt (without a match with the 16-bit counter).

     The crest interrupt forcibly raises the timer output.  This cancels the 100% output.

**Remarks 1.** ↑ means forcible raising and ↓ means forcible lowering.

       **2.** m = 1 to 3

**Figure 11-11.  PWM Output Waveform from 0% to 100% and from 100% to 0% (With Dead Time)**



<1>  The valley interrupt selects 100% ←→ 0% or 0% ←→ 100% output.

Output can be selected from 100% ←→ 0% or 0% ←→ 100% immediately after the timer has been started.

<2>  The crest interrupt selects 100% ←→ 0% output.

The crest interrupt selects 100% → 0% output by using the timer output forcible raising function and by a match between the 16-bit counter value and the TAB1CCR0 register value.
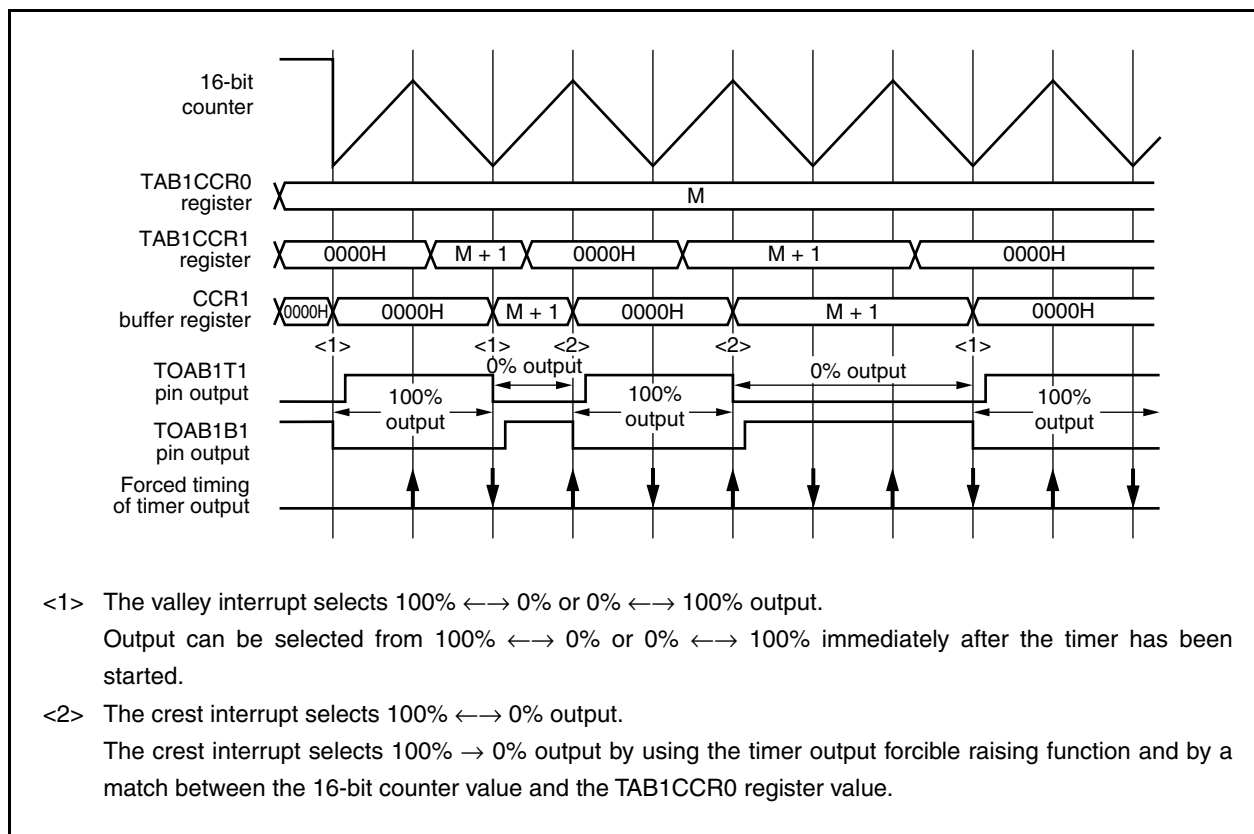
**(3)  Output waveform in vicinity of 0% and 100% output**

If an interrupt is generated because the value of the 16-bit counter matches the value of the compare register while dead time is being counted, the dead-time counter is cleared and starts its count operation again.

The output waveform of dead-time control in the vicinity of 0% and 100% output is shown below.

**Figure 11-12.  PWM Output Waveform with Dead Time (2)**



(a) 0% output (TAB1CCRm register = M + 1, TAB1CCR0 register = M, TAB1DTC register = a)

(b) In vicinity of 0% output (TAB1CCRm register = i ≥ M + 1 − a/2, TAB1CCR0 register = M, TAB1DTC register = a)

Negative-phase output width: $(M + 1 - i) \times 2 + a$
(e.g., output width is 2 + a where TAB1CCRm register = M)

(c) In vicinity of 100% output (TAB1CCRm register = i ≤ a/2, TAB1CCR0 register = M, TAB1DTC register = a)

Positive-phase output width: $(M + 1 - i) \times 2 - a$
(e.g., output width is 2 − a where TAB1CCRm register = 0001H.)

(d) 100% output (TAB1CCRm register = 0000H, TAB1CCR0 register = M, TAB1DTC register = a)

**Remark**　m = 1 to 3

**(4) Automatic dead-time width narrowing function (TAB1OPT2.TAB1DTM bit = 1)**

The dead-time width can be automatically narrowed in the vicinity of 0% output or 100% output by setting the TAB1OPT2.TAB1DTM bit to 1.

By setting the TAB1DTM bit to 1, the dead-time counter is not cleared, but starts down counting if the TOAB1m (internal signal) output of timer AB changes during dead-time counting.

The following timing chart shows the operation of the dead-time counter when the TAB1DTM bit is set to 1.

**Figure 11-13.  Operation of Dead-Time Counter m (1)**



**(a) In vicinity of 0% output**
**(TAB1CCRm register = i ≥ M + 1 − a/2, TAB1CCR0 register = M, TAB1DTC register = a)**

Negative-phase wave output width: $(M + 1 - i) \times 4$
(e.g., output width is 4 where TAB1CCRm = M)

**(b)  In vicinity of 100% output (TAB1CCRm register = i ≤ a/2, TAB1CCR0 register = M, TAB1DTC register = a)**

Positive-phase wave output width: $(M + 1 - i) \times 2 - (i \times 2)$
(e.g., output width is $M \times 2 - 2$ where TAB1CCRm = 0001H)

**Note** The output width of the first wave differs from that of the second and subsequent waves immediately after the TAB1CTL0.TAB1CE bit has been set.  The first wave is shorter than the second wave because the dead time is fully counted.

**Remark**  m = 1 to 3

**(5) Dead-time control in case of incorrect setting**

Usually, the TOAB1m (internal signal) output of TAB1 changes only once during dead-time counting, only in the vicinity of 0% and 100% output. This section shows an example where the TAB1CCR0 register (carrier cycle) and TAB1DTC register (dead-time value) are incorrectly set. If these registers are incorrectly set, the TOAB1m (internal signal) output of TAB1 changes twice or three times during dead-time counting. The following flowchart shows the 6-phase PWM output wave in this case.

**Figure 11-14.  Operation of Dead-Time Counter m (2)**



**(a) When TAB1OPT2.TAB1DTM bit = 0, TAB1CCR0 register = 0006H, TAB1DTC register = 000FH, TAB1CCRm register = 0004H**

**(b) When TAB1OPT2.TAB1DTM bit = 1, TAB1CCR0 register = 0006H, TAB1DTC register = 000FH, TAB1CCRm register = 0002H**

**Remark**   m = 1 to 3
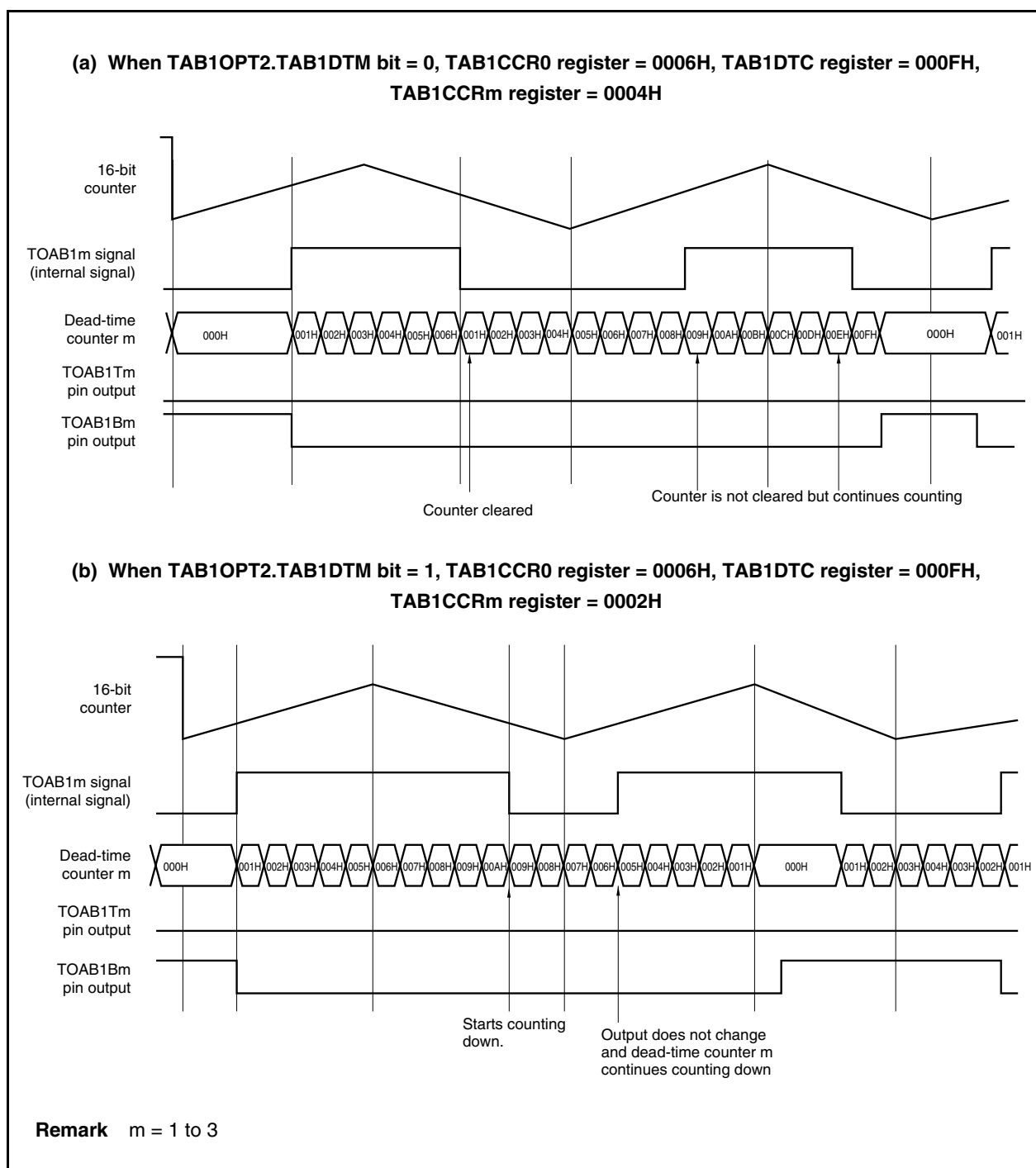
**11.4.3  Interrupt culling function**

- The interrupts to be culled are INTTAB1CC0 (crest interrupt) and INTTAB1OV (valley interrupt).
- The TAB1OPT1.TAB1ICE bit is used to enable output of the INTTAB1CC0 interrupt and the number of times the interrupt is to be culled.
- The TAB1OPT1.TAB1IOE bit is used to enable output of the INTTAB1OV interrupt and the number of times the interrupt is to be culled.
- The TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 bits are used to specify the number of counts by which a specified interrupt is to be culled.  The interrupt is masked for the duration of the specified number of counts and is generated at the next interrupt timing.
- The TAB1OPT2.TAB1RDE bit is used to specify whether transfer is to be culled or not.
  If it is specified that transfer is to be culled, transfer is executed at the same timing as the interrupt output after culling.
  If it is specified that transfer is not to be culled, transfer is executed at the transfer timing after the TAB1CCR1 register has been written.
- The TAB1OPT0.TAB1CMS bit is used to specify whether the registers with a transfer function are batch rewritten or anytime rewritten.
  The values of the registers are updated in synchronization with transfer when the TAB1CMS bit is 0.  When the TAB1CMS bit is 1, the values of the registers are immediately updated when a new value is written to the registers.
  Transfer is performed from the TAB1CCRm register to the CCRm buffer register in synchronization with the interrupt culling timing.

**Cautions 1.  When using the interrupt culling function in the batch rewrite mode (transfer mode), execute the function in the intermittent batch rewrite mode (transfer culling mode).**

**2.  The interrupt is generated at the timing after culling.**

**Remark**  m = 1 to 3

**(1)  Interrupt culling operation**

**Figure 11-15.  Interrupt Culling Operation When TAB1OPT1.TAB1ICE Bit = 1, TAB1OPT1.TAB1IOE Bit = 1, TAB1OPT2.TAB1RDE Bit = 1 (Crest/Valley Interrupt Output)**



**Remark** ⌐¦⌐: Culled interrupt
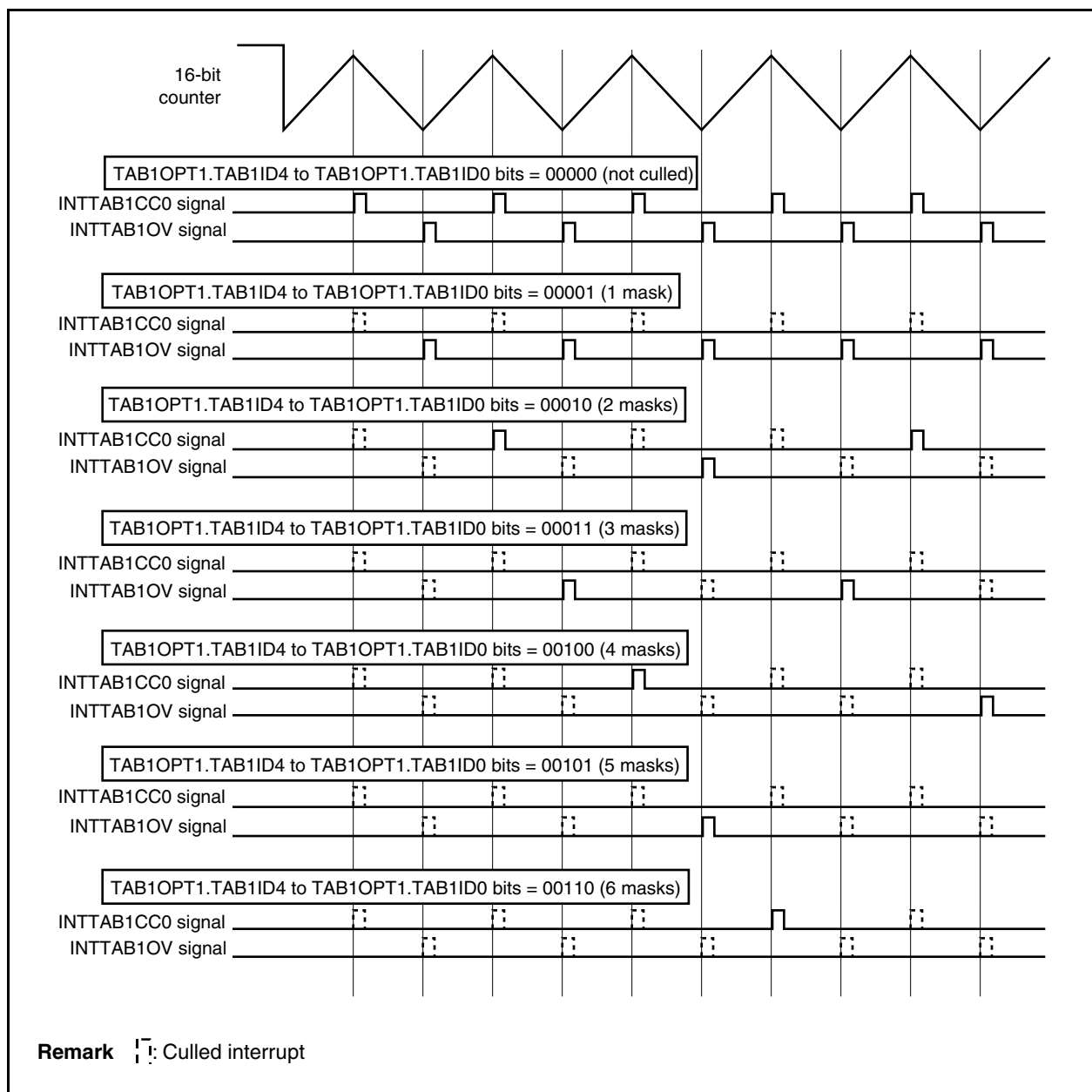
**Figure 11-16.  Interrupt Culling Operation When TAB1OPT1.TAB1ICE Bit = 1, TAB1OPT1.TAB1IOE Bit = 0,**
**TAB1OPT2.TAB1RDE Bit = 1 (Crest Interrupt Output)**
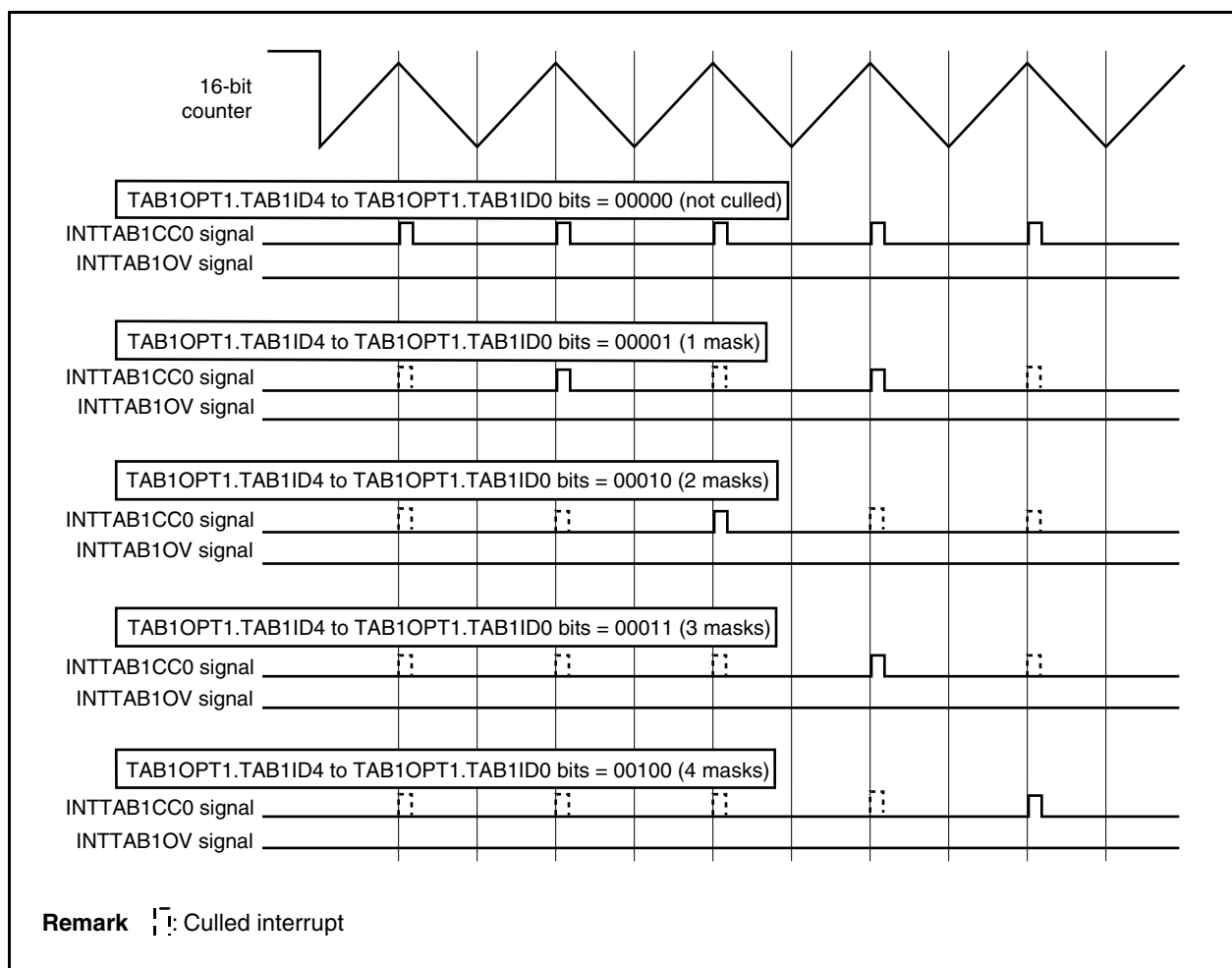
**Figure 11-17. Interrupt Culling Operation When TAB1OPT1.TAB1ICE Bit = 0, TAB1OPT1.TAB1IOE Bit = 1, TAB1OPT2.TAB1RDE Bit = 1 (Valley Interrupt Output)**
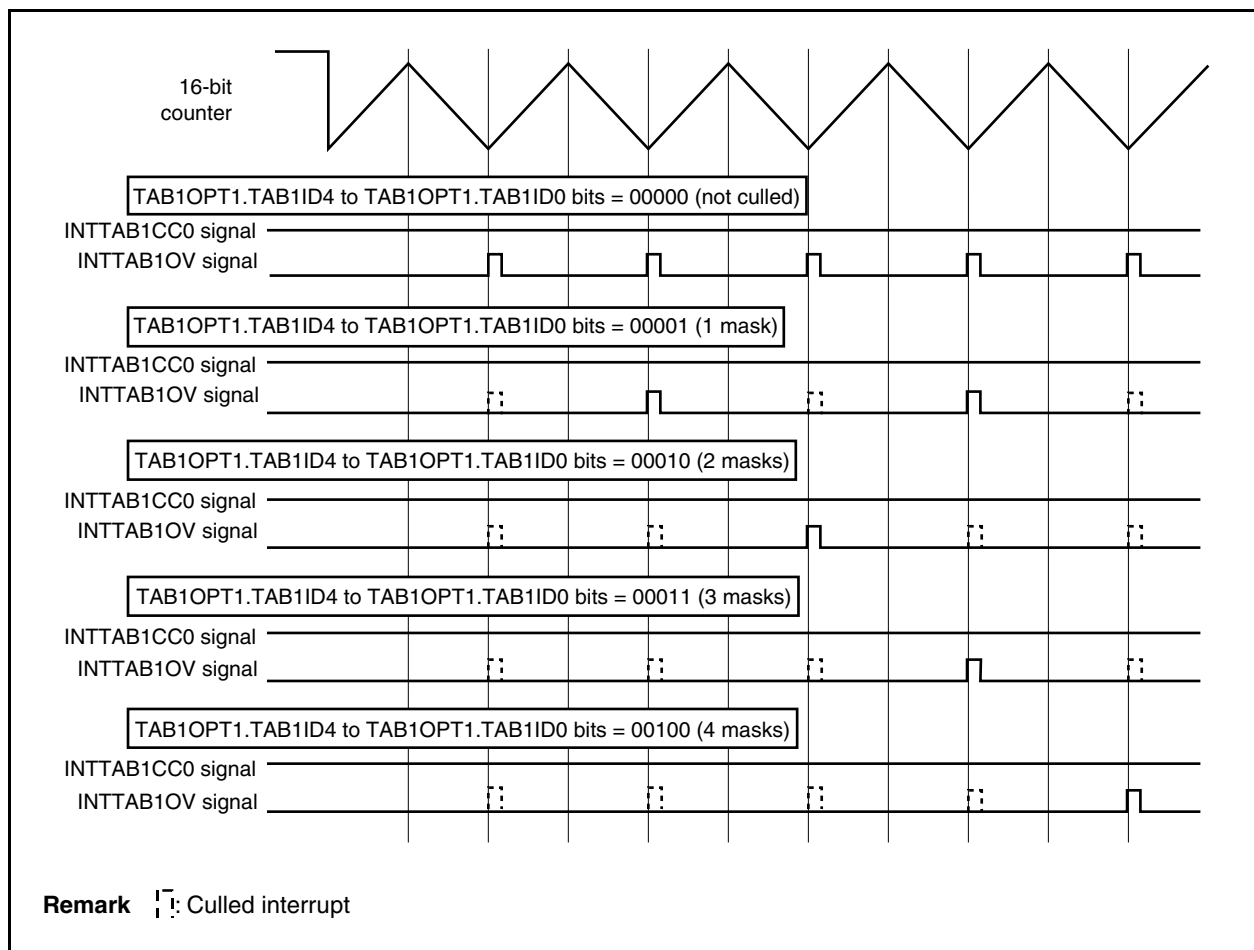
**(2) To alternately output crest interrupt (INTTAB1CC0) and valley interrupt (INTTAB1OV)**

To alternately output the crest and valley interrupts, set both the TAB1OPT1.TAB1ICE and TAB1OPT1.TAB1IOE bits to 1.

**Figure 11-18. Crest/Valley Interrupt Output**



**(a) TAB1OPT0.TAB1CMS bit = 0, TAB1OPT2.TAB1RDE bit = 1 (with transfer culling control)**

**Remarks 1.** Transfer is performed at the culled interrupt output timing.  The other transfer timing is ignored.

**2.** ⸢⸥: Culled interrupt

**(b) TAB1CMS bit = 1, TAB1RDE bit = 0 or 1 (without transfer control)**

**Remarks 1.** Rewriting is reflected immediately.  The transfer timing is ignored.

**2.** ⸢⸥: Culled interrupt
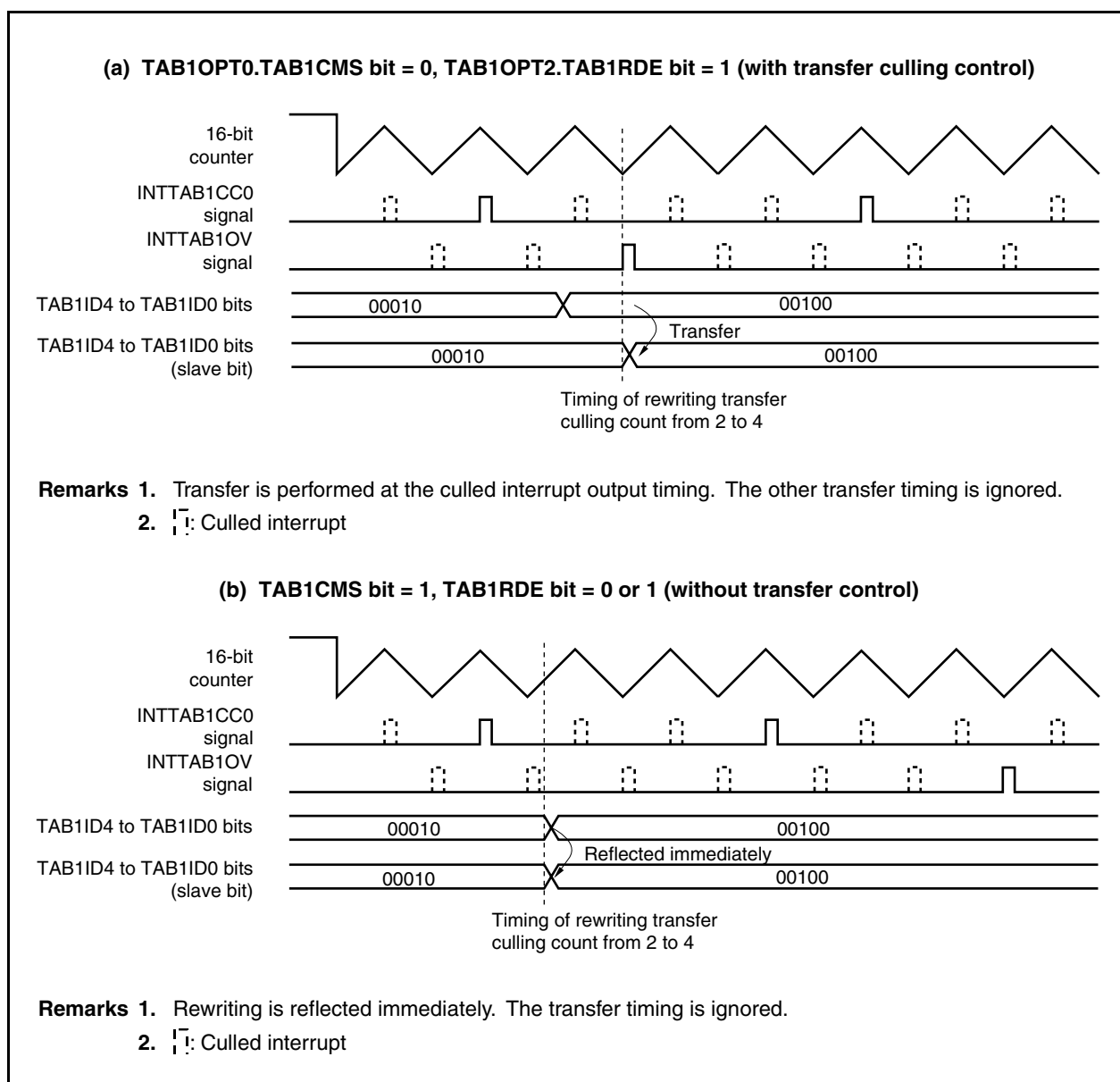
**(3) To output only crest interrupt (INTTAB1CC0)**

Set the TAB1OPT1.TAB1ICE bit to 1 and clear the TAB1OPT1.TAB1IOE bit to 0.

**Figure 11-19.  Crest Interrupt Output**



**(a) TAB1OPT0.TAB1CMS bit = 0, TAB1OPT2.TAB1RDE bit = 1 (with transfer culling control)**

**Remarks 1.** Transfer is performed at the culled interrupt output timing.  The other transfer timing is ignored.

**2.** ⌐ᵢ: Culled interrupt

**(b) TAB1CMS bit = 1, TAB1RDE bit = 0 or 1 (without transfer control)**

**Remarks 1.** Rewriting is reflected immediately.  The transfer timing is ignored.

**2.** ⌐ᵢ: Culled interrupt
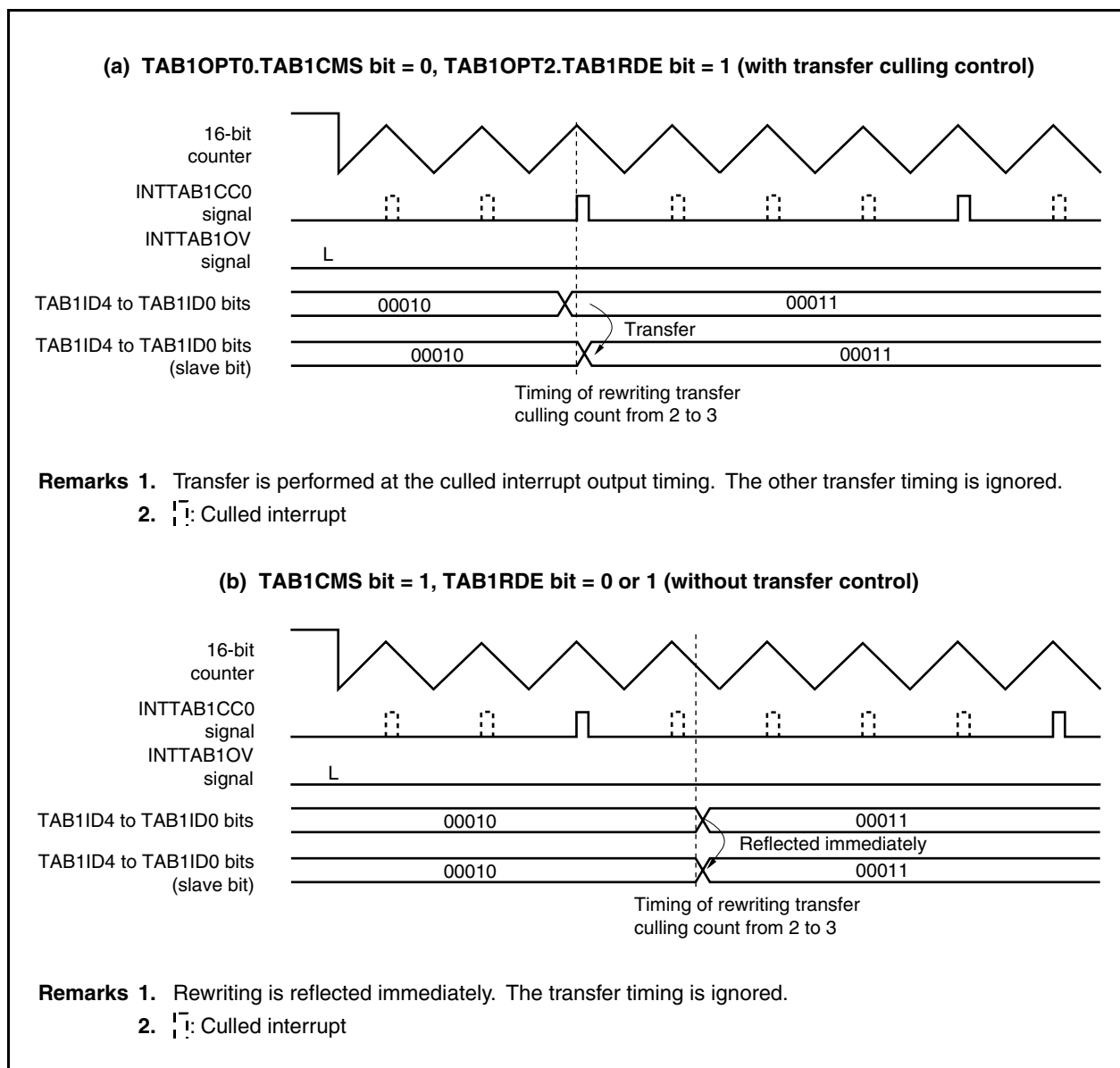
**(4) To output only valley interrupt (INTTAB1OV)**

Clear the TAB1OPT1.TAB1ICE bit to 0 and set the TAB1IOE bit to 1.

**Figure 11-20.  Valley Interrupt Output**



**(a)  TAB1OPT0.TAB1CMS bit = 0, TAB1OPT2.TAB1RDE bit = 1 (with transfer culling control)**

**Remarks 1.**  Transfer is performed at the culled interrupt output timing.  The other transfer timing is ignored.

**2.**  ⌐¦¬: Culled interrupt

**(b)  TAB1CMS bit = 1, TAB1RDE bit = 0 or 1 (without transfer control)**

**Remarks 1.**  Rewriting is reflected immediately.  The transfer timing is ignored.

**2.**  ⌐¦¬: Culled interrupt
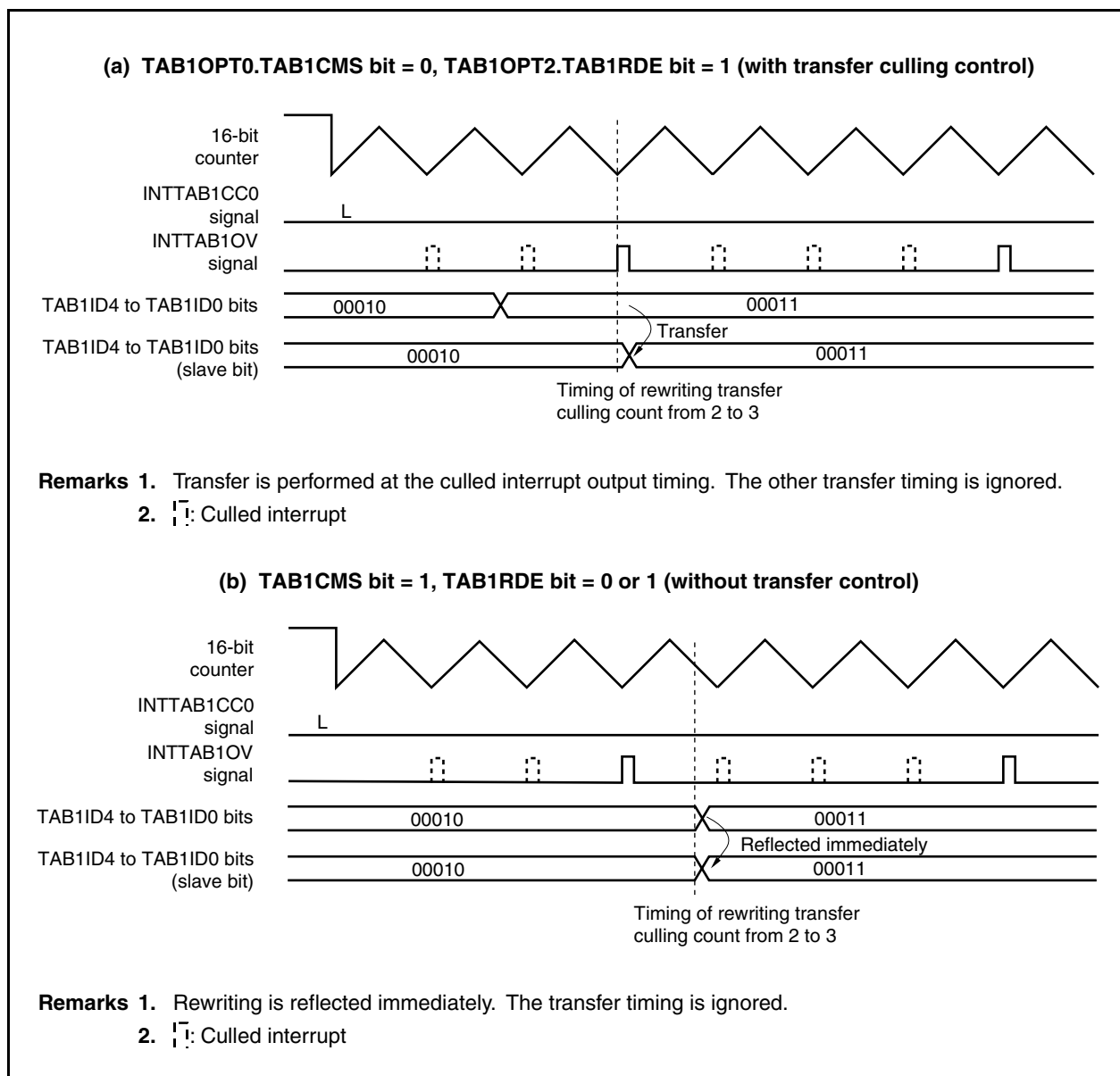
**11.4.4  Operation to rewrite register with transfer function**

The following seven registers are provided with a transfer function and are used to control a motor.  Each of the registers has a buffer register.

- TAB1CCR0: Register that specifies the cycle of the 16-bit counter (TAB)
- TAB1CCR1: Register that specifies the duty factor of TOAB1T1 (U) and TOAB1B1 ($\overline{\text{U}}$)
- TAB1CCR2: Register that specifies the duty factor of TOAB1T2 (V) and TOAB1B2 ($\overline{\text{V}}$)
- TAB1CCR3: Register that specifies the duty factor of TOAB1T3 (W) and TOAB1B3 ($\overline{\text{W}}$)
- TAB1OPT1: Register that specifies the culling of interrupts
- TAA4CCR0: Register that specifies the A/D conversion start trigger generation timing (TAA4 during tuning operation)
- TAA4CCR1: Register that specifies the A/D conversion start trigger generation timing (TAA4 during tuning operation)

The following three rewrite modes are provided in the registers with a transfer function.

- Anytime rewrite mode

  This mode is set by setting the TAB1OPT0.TAB1CMS bit to 1.  The specification of the TAB1OPT2.TAB1RDE bit is ignored.

  In this mode, each compare register is updated independently, and the value of the compare register is updated as soon as a new value is written to it.

- Batch rewrite mode (transfer mode)

  This mode is set by clearing the TAB1OPT0.TAB1CMS bit to 0, the TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 bits to 00000, and the TAB1OPT2.TAB1RDE bit to 0.

  When data is written to the TAB1CCR1 register, data in the seven registers are transferred to the buffer register all at once at the next transfer timing.  Unless the TAB1CCR1 register is rewritten, the transfer operation is not performed even if the other six registers are rewritten.

  The transfer timing is the timing of each crest (match between the 16-bit counter value and TAB1CCR0 register value) and valley (match between the 16-bit counter value and 0001H) regardless of the interrupt.

- Intermittent batch rewrite mode (transfer culling mode)

  This mode is set by clearing the TAB1OPT0.TAB1CMS bit to 0 and setting the TAB1OPT2.TAB1RDE bit to 1.

  When data is written to the TAB1CCR1 register, data from the seven registers is transferred to the buffer register all at once at the next transfer timing.  Unless the TAB1CCR1 register is rewritten, the transfer operation is not performed even if the other six registers are rewritten.

  If interrupt culling is specified by the TAB1OPT1 register, the transfer timing is also culled as the interrupts are culled, and data from the seven registers is transferred all at once at the culled timing of the crest interrupt (match between the 16-bit counter value and TAB1CCR0 register value) or valley interrupt (match between the 16-bit counter value and 0001H).

  For details of the interrupt culling function, see **11.4.3  Interrupt culling function**.

**(1) Anytime rewrite mode**

This mode is set by setting the TAB1OPT0.TAB1CMS bit to 1.  The setting of the TAB1OPT2.TAB1RDE bit is ignored.

In this mode, the value written to each register with a transfer function is immediately transferred to an internal buffer register and compared with the value of the counter.  If a register with transfer function is rewritten in this mode after the count value of the 16-bit counter matches the value of the TAB1CCRm register, the rewritten value is not reflected because the next match is ignored after the first match has occurred.  If the register is rewritten during counting up, the new register value becomes valid after the counter has started counting down.

**Figure 11-21.  Timing of Reflecting Rewritten Value**



**Note**  After the register (TAB1CCR0, TAB1CCR2, TAB1CCR3, TAB1OPT1, TAA4CCR0, or TAA4CCR1) has been written, the written value is transferred to an internal buffer register after four clocks of the operating clock.  However, the value of the TAB1CCR1 register is transferred after 5 clocks.
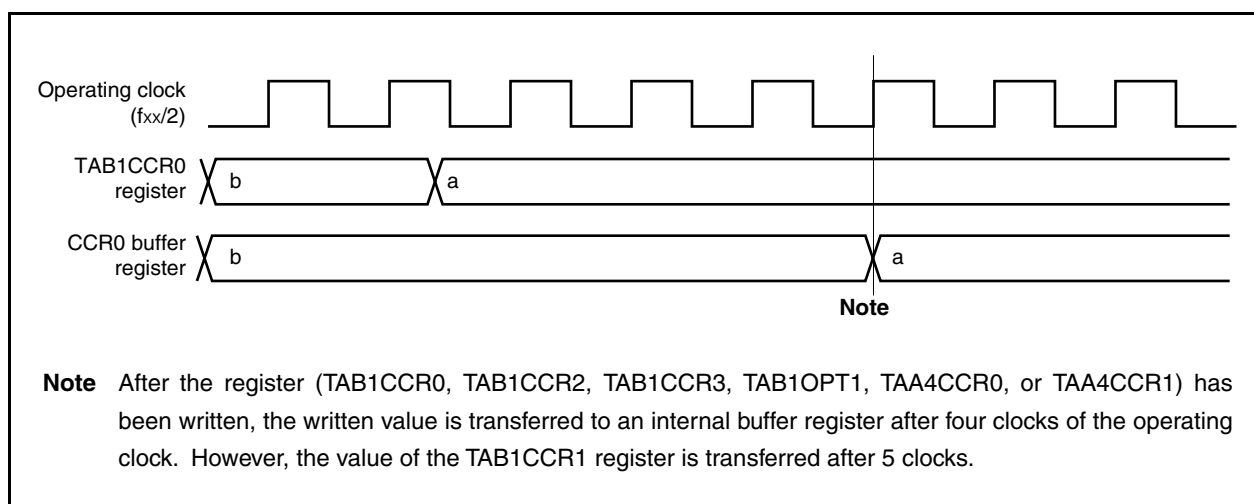
**(a) Rewriting TAB1CCR0 register**

Even if the TAB1CCR0 register is rewritten in the anytime rewrite mode, the new value may not be reflected in some cases.

**Figure 11-22.  Example of Rewriting TAB1CCR0 Register**



**Rewriting during period <1> (rewriting during counting up)**

If the newly rewritten value is greater than the value of the 16-bit counter, there is no problem because it will match the value of the 16-bit counter.  If the new value is less than the value of the 16-bit counter, it will not match the value of the counter.  As a result, the 16-bit counter overflows and continues counting up from 0000H until it matches the register value again, and the correct PWM waveform is not output.

**Rewriting during period <2> (rewriting during counting down)**

A match with the value of the 16-bit counter is ignored during counting down.  Therefore, the rewritten period value is reflected as the match point starting from counting up in the next cycle.

**(b)  Rewriting TAB1CCRm register**

Figure 11-24 shows the timing of rewriting before the value of the 16-bit counter matches the value of the TAB1CCRm register (<1> in Figure 11-23), and Figure 11-25 shows the timing of rewriting after the value of the 16-bit counter matches the value of the TAB1CCRm register (<2> in Figure 11-23).

**Figure 11-23.  Basic Operation of 16-Bit Counter and TAB1CCRm Register**



**Remarks  1.**  i = Set value of TAB1CCRm register
        **2.**  m = 1 to 3
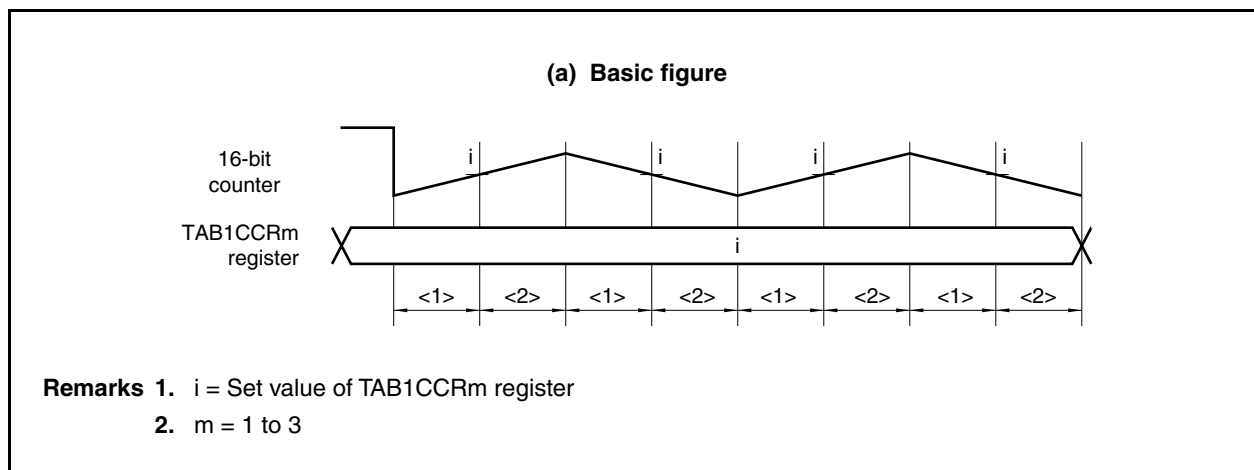
**Figure 11-24. Example of Rewriting TAB1CCR1 to TAB1CCR3 Registers (Rewriting Before Match Occurs)**

**(a)**

If the TAB1CCRm register is rewritten before its value matches the value of the 16-bit counter, the register value will match the value of the 16-bit counter after the register has been rewritten. Consequently, the new register value is immediately reflected.
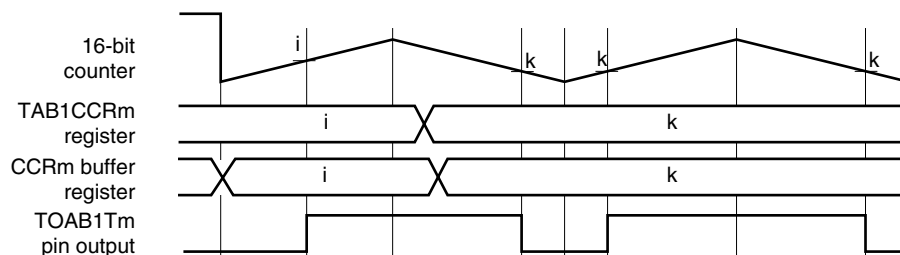


**(b)**

If a value less than the value of the 16-bit counter (greater if the counter is counting down) is written to the TAB1CCRm register, the output waveform is as follows because the register value does not match the counter value.
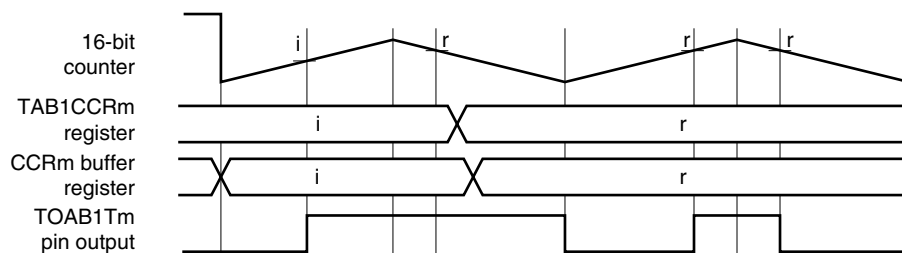


If the register value does not match the counter value, the TOAB1Tm pin output does not change. Even if the value of the 16-bit counter does not match the value of the TAB1CCRm register, the TOAB1Tm pin output always changes to the high level if the crest interrupt occurs and to the low level if the valley interrupt occurs.

This is a function provided for 0% output and 100% output.

For details, see **11.4.2 (2) PWM output of 0%/100%**.

**Remarks 1.** i, r, k = Set values of TAB1CCRm register
      **2.** m = 1 to 3

**Figure 11-25.  Example of Rewriting TAB1CCR1 to TAB1CCR3 Registers (Rewriting After Match Occurs)**
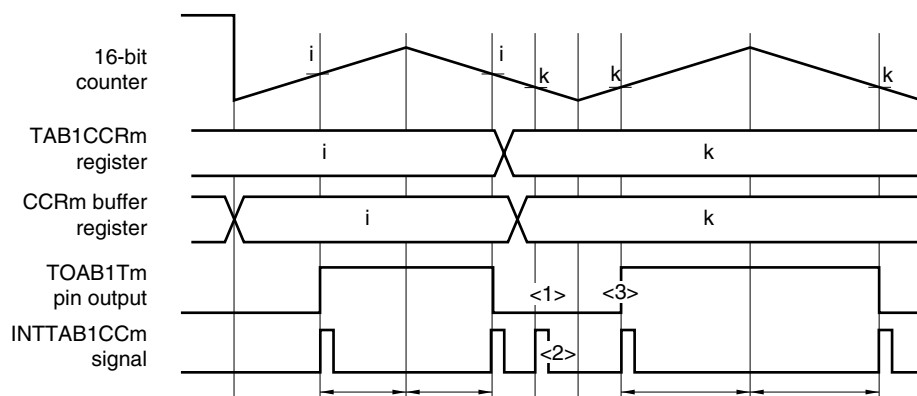


<1>  Matching of the count value of the 16-bit counter and the value of the TAB1CCRm register as a result of rewriting the register is ignored after a match signal has been generated, and the PWM output does not change.

<2>  Even if the PWM output does not change, the interrupt generated upon a match between the 16-bit counter value and the TAB1CCRm register value (INTTAB1CCm) is output.

<3>  The next match between the 16-bit counter and TAB1CCRm register is valid after the counter has changed its counting direction to up or down, and the PWM output changes.

If the TAB1CCRm register is rewritten after its value matches the value of the 16-bit counter, the next match is ignored after the first match occurs and the rewritten value is not reflected in the TOAB1Tm pin output.  If the register is rewritten while the counter is counting down, the match that occurs after the counter starts counting down is valid (the match that occurs after the counter has started counting up is valid if the register is rewritten while the counter is counting up).

**Remarks 1.**  i, r, k = Set value of TAB1CCRm register
           **2.**  m = 1 to 3

**(c)  Rewriting TAB1OPT1 register**

The interrupt culling counter is cleared when the TAB1OPT1 register is written.  When the interrupt culling counter has been cleared, the measured number of times the interrupt has occurred is discarded.  Consequently, the interrupt generation interval is temporarily extended.

To avoid this operation, rewrite the TAB1OPT1 register in the intermittent batch rewrite mode (transfer culling mode).

For details of rewriting the TAB1OPT1 register, see **11.4.3  Interrupt culling function**.

**(2) Batch rewrite mode (transfer mode)**

This mode is set by clearing the TAB1OPT0.TAB1CMS bit to 0, the TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 bits to 00000, and the TAB1OPT2.TAB1RDE bit to 0.

In this mode, the values written to each compare register are transferred to the internal buffer register all at once at the transfer timing and compared with the counter value.

**(a) Rewriting procedure**

If data is written to the TAB1CCR1 register, the values set to the TAB1CCR0 to TAB1CCR3, TAB1OPT1, TAA4CCR0, and TAA4CCR1 registers are transferred all at once to the internal buffer register at the next transfer timing. Therefore, write to the TAB1CCR1 register last. Writing to the register is prohibited after the TAB1CCR1 register has been written and before the transfer timing is generated (until the crest (match between the 16-bit counter value and TAB1CCR0 register value) or the valley (match between the 16-bit counter value and 0001H)). The operation procedure is as follows.

<1> Rewriting the TAB1CCR0, TAB1CCR2, TAB1CCR3, TAB1OPT1, TAA4CCR0, and TAA4CCR1 registers
Do not rewrite registers that do not have to be rewritten.
<2> Rewriting the TAB1CCR1 register
Rewrite the same value to the register even when it is not necessary to rewrite the TAB1CCR1 register.
<3> Holding the next rewriting pending until the transfer timing is generated
Rewrite the register next time after the INTTAB1OV or INTTAB1CC0 interrupt has occurred.
<4> Return to <1>.

**Figure 11-26. Basic Operation in Batch Mode**
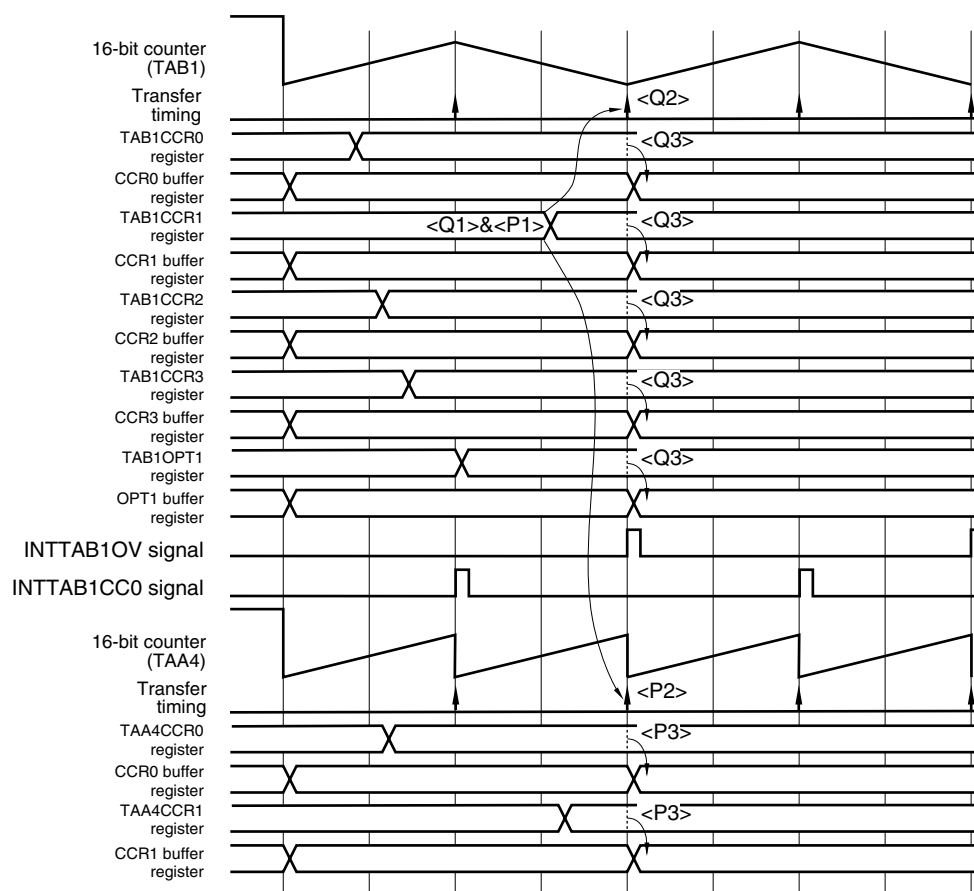


[Operation of TAB1]

<Q1> Write the TAB1CCR1 register

<Q2> The target timing is the first transfer timing after a write to the TAB1CCR1 register.

<Q3> The values are transferred all at once at the transfer timing.

[Operation of TAA4]

<P1> Write the TAB1CCR1 register

<P2> The target timing is the first transfer timing after a write to the TAB1CCR1 register.

<P3> The values are transferred all at once at the transfer timing.

**(b) Rewriting TAB1CCR0 register**

When rewriting the TAB1CCR0 register in the batch rewrite mode, the output waveform differs depending on whether transfer occurs at the crest (match between the 16-bit counter value and TAB1CCR0 register value) or at the valley (match between the 16-bit counter value and 0001H).  Usually, it is recommended to rewrite the TAB1CCR0 register while the 16-bit counter is counting down, and transfer the register value at the transfer timing of the crest timing.

Figure 11-28 shows an example of rewriting the TAB1CCR0 register while the 16-bit counter is counting up (during period <1> in Figure 11-27).  Figure 11-29 shows an example of rewriting the TAB1CCR0 register while the counter is counting down (during period <2> in Figure 11-27).

**Figure 11-27.  Basic Operation of 16-Bit Counter**
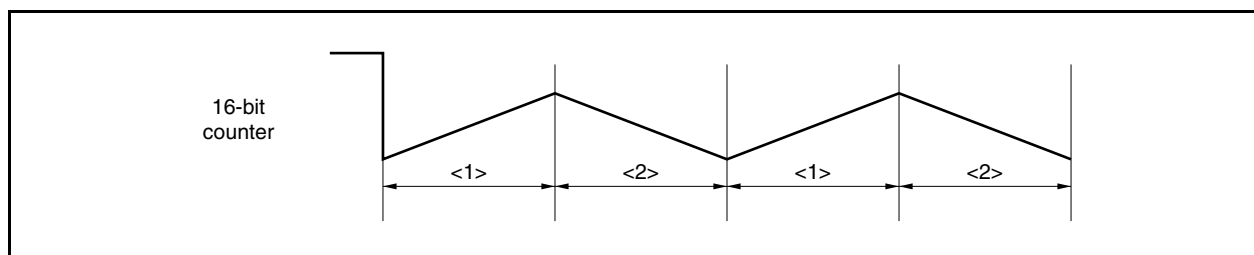
The transfer timing in Figure 11-28 is at the point where the crest timing occurs.  While the 16-bit counter is counting down, the cycle changes and an asymmetrical triangular wave is output.  Because the cycle changes, rewrite the duty factor (voltage data value).

**Figure 11-28.  Example of Rewriting TAB1CCR0 Register (During Counting Up)**
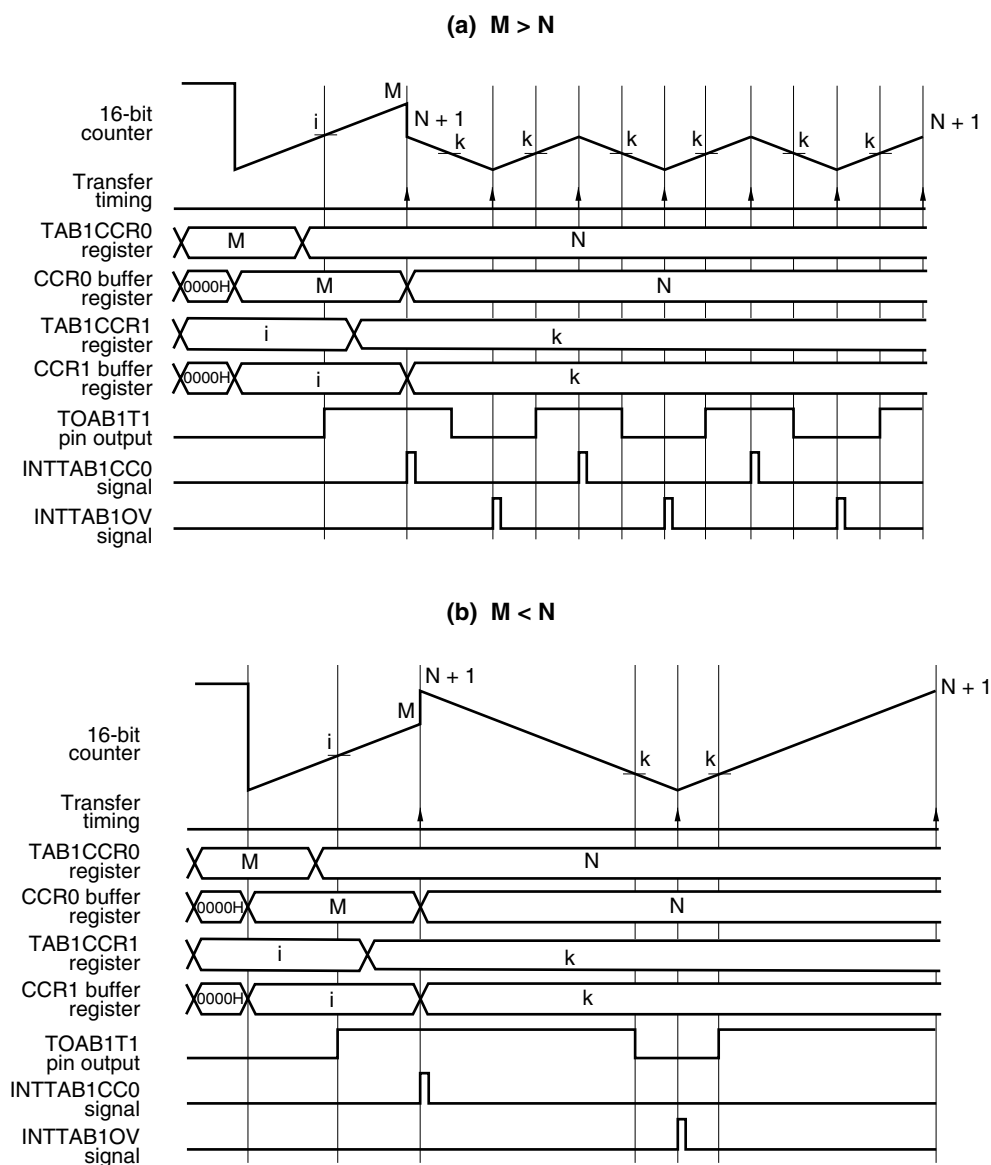


**Remarks 1.** If transfer (match between the value of the 16-bit counter and the value of the CCR0 buffer register) occurs in the 6-phase PWM output mode, the value of the TAB1CCR0 register plus 1 is loaded to the 16-bit counter.  In this way, the expected wave can be output even if the cycle value is changed at the transfer timing of the crest (match between the 16-bit counter value and the TAB1CCR0 register value) timing.

**2.** M:  Value of CCR0 buffer register before rewriting

N:  Value of CCR0 buffer register after rewriting

**Figure 11-29.  Example of Rewriting TAB1CCR0 Register (During Counting Down)**



Because the next transfer timing is at the point of the valley (match between the 16-bit counter value and 0001H), the cycle value changes from the next cycle and output of a symmetrical triangular wave is maintained.  Because the cycle changes, rewrite the duty value (voltage data value) as required.
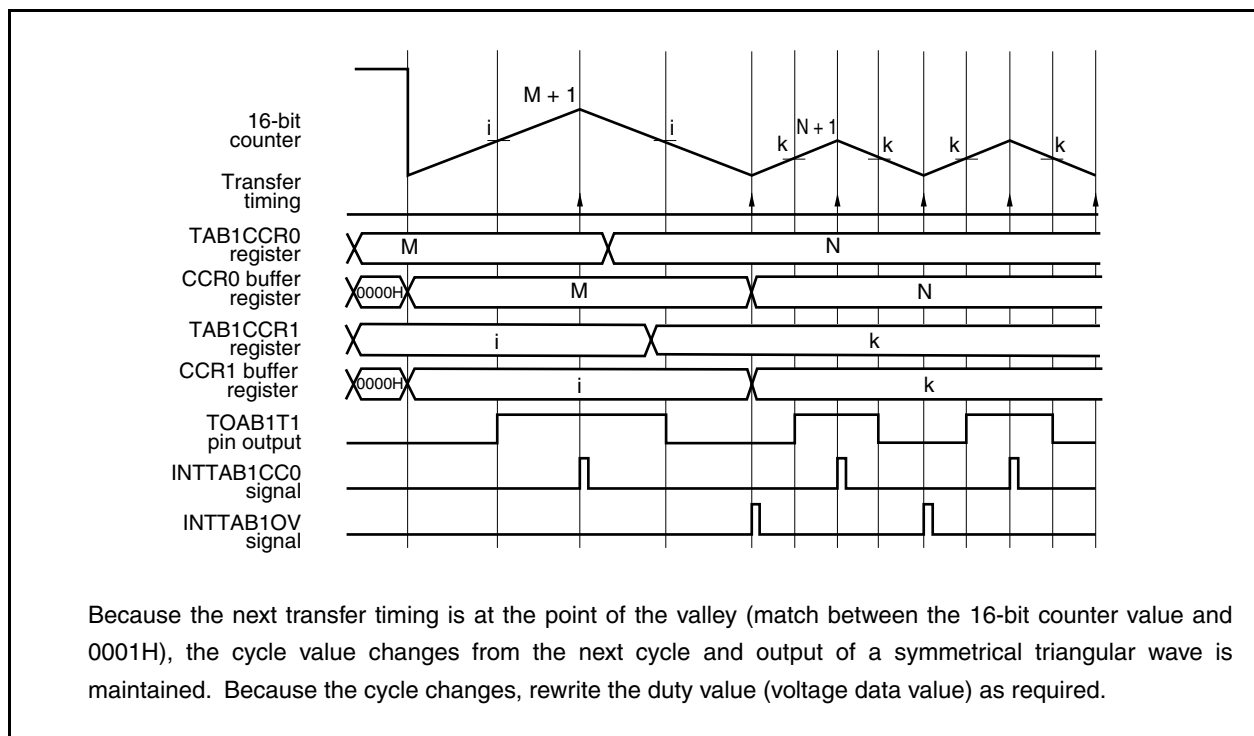
**(c)  Rewriting TAB1CCRm register**

**Figure 11-30.  Example of Rewriting TAB1CCRm Register**
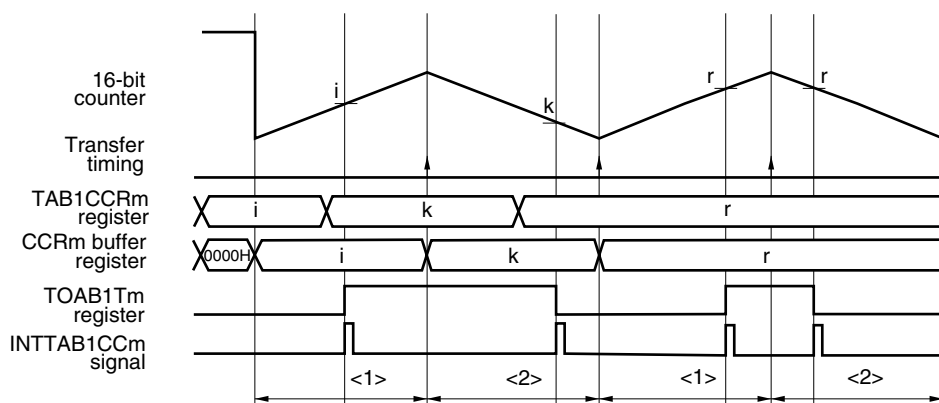


**Rewriting during period <1> (rewriting during counting up)**

   Because the TAB1CCRm register value is transferred at the transfer timing of the crest (match between the
   16-bit counter value and TAB1CCRm register value), an asymmetrical triangular wave is output.

**Rewriting during period <2> (rewriting during counting down)**

   Because the TAB1CCRm register value is transferred at the transfer timing of the valley (match between the
   16-bit counter value and 0001H), a symmetrical triangular wave is output.

   **Remark**   m = 1 to 3

**(d)  Transferring TAB1OPT1 register value**

   Do not set the TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 bits to other than 00000.  When using the interrupt
   culling function, rewrite the TAB1OPT1 register in the intermittent batch rewrite mode (transfer culling mode).
   For details of rewriting the TAB1OPT1 register, see **11.4.3  Interrupt culling function**.

**(3) Intermittent batch rewrite mode (transfer culling mode)**

This mode is set by clearing the TAB1OPT0.TAB1CMS bit to 0 and setting the TAB1OPT2.TAB1RDE bit to 1.

In this mode, the values written to each compare register are transferred to the internal buffer register all at once after the culled transfer timing and compared with the counter value.  The transfer timing is the timing at which an interrupt is generated (INTTAB1CC0, INTTAB1OV) by interrupt culling.

For details of the interrupt culling function, see **11.4.3  Interrupt culling function**.

**(a) Rewriting procedure**

If data is written to the TAB1CCR1 register, the data of the TAB1CCR0 to TAB1CCR3, TAB1OPT1, TAA4CCR0, and TAA4CCR1 registers are transferred all at once to the internal buffer register at the next transfer timing.  Therefore, write to the TAB1CCR1 register last.  Writing to the register is prohibited after the TAB1CCR1 register has been written until the transfer timing is generated (until the INTTAB1OV or INTTAB1CC0 interrupt occurs).  The operation procedure is as follows.

<1>  Rewrite the TAB1CCR0, TAB1CCR2, TAB1CCR3, TAB1OPT1, TAA4CCR0, and TAA4CCR1 registers.
Do not rewrite registers that do not have to be rewritten.

<2>  Rewrite the TAB1CCR1 register.
Rewrite the same value to the register even when it is not necessary to rewrite the TAB1CCR1 register.

<3>  Hold the next rewriting pending until the transfer timing is generated.
Perform the next rewrite after the INTTAB1OV or INTTAB1CC0 interrupt has occurred.

<4>  Return to <1>.

**Figure 11-31. Basic Operation in Intermittent Batch Rewrite Mode**
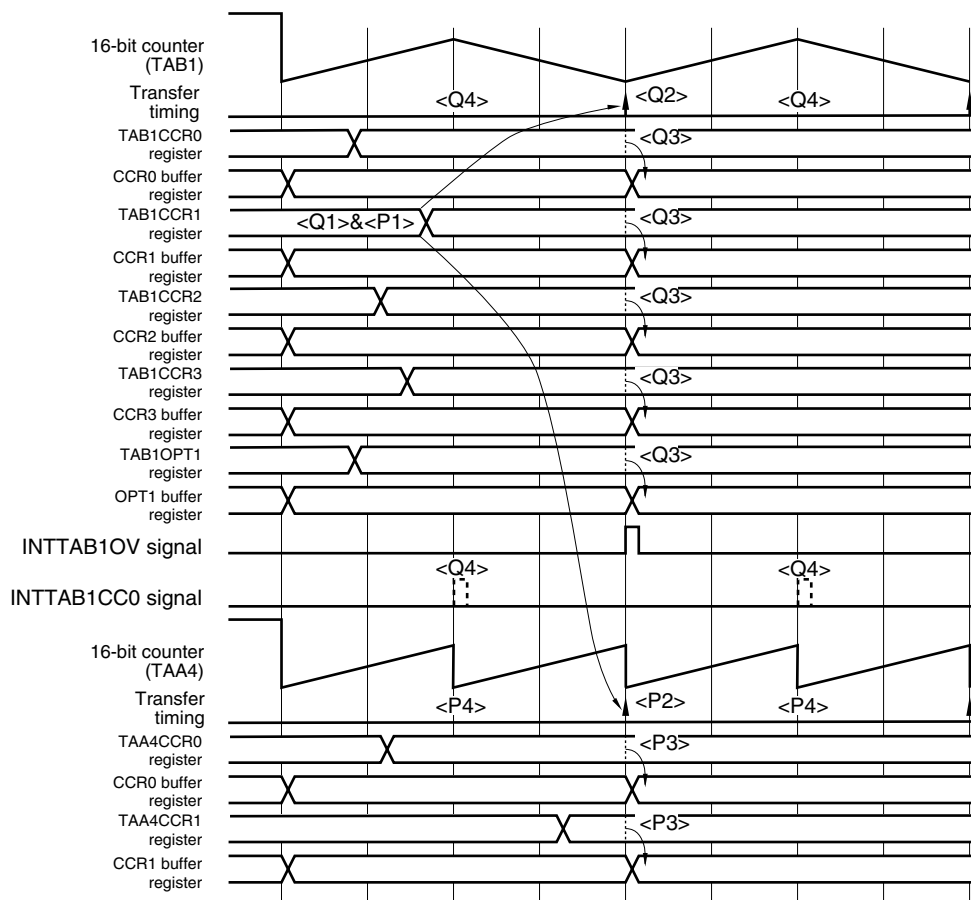


**[TAB1 operation]**

<Q1> Write the TAB1CCR1 register.

<Q2> Rewrite the register at the transfer timing that is generated after the TAB1CCR1 register has been rewritten.

<Q3> The registers are transferred all at once at the transfer timing.

<Q4> The transfer timing is also culled as the interrupts are culled.

**[TAA4 operation]**

<P1> Write the TAB1CCR1 register.

<P2> Rewrite the register at the transfer timing that is generated after the TAB1CCR1 register has been rewritten.

<P3> The registers are transferred all at once at the transfer timing.

<P4> The transfer timing is also culled as the interrupts are culled.

**Remark**  This is an example of the operation when the TAB1OPT1.TAB1ICE bit = 1, TAB1OPT1.TAB1IOE bit = 1, and TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 bits = 00001.

**(b) Rewriting TAB1CCR0 register**

When rewriting the TAB1CCR0 register in the intermittent batch mode, the output waveform differs depending on where the occurrence of the crest or valley interrupt is specified by the interrupt culling setting. The following figure illustrates the change of the output waveform when interrupts are culled.

**Figure 11-32.  Rewriting TAB1CCR0 Register (When Crest Interrupt Is Set)**



The transfer timing is generated when the crest interrupt occurs, the cycle of counting up and counting down changes, and an asymmetrical triangular wave is output.

**Remarks 1.**  This is an example of the operation when the TAB1OPT1.TAB1ICE bit = 1, TAB1OPT1.TAB1IOE bit = 0, and TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 bits = 00001.

**2.**  ⌐¦: Culled interrupt
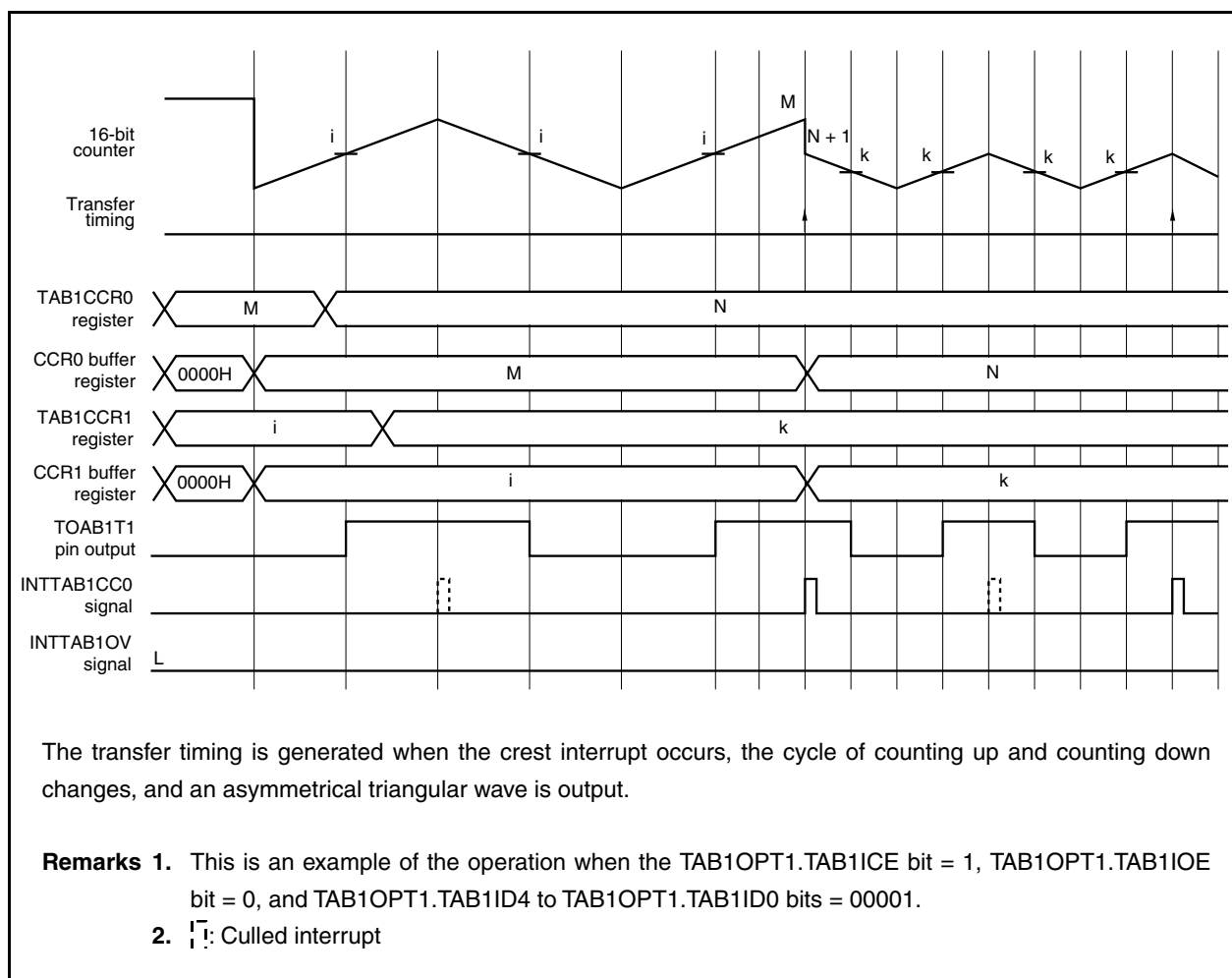
**Figure 11-33.  Rewriting TAB1CCR0 Register (When Valley Interrupt Is Set)**



The transfer timing is generated when the valley interrupt occurs, the cycle of counting up becomes same as cycle of counting down, and a symmetrical triangular wave is output.

**Remarks 1.** This is an example of the operation when the TAB1OPT1.TAB1ICE bit = 0, TAB1OPT1.TAB1IOE bit = 1, and TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 bits = 00001.

**2.** ⌐¦: Culled interrupt
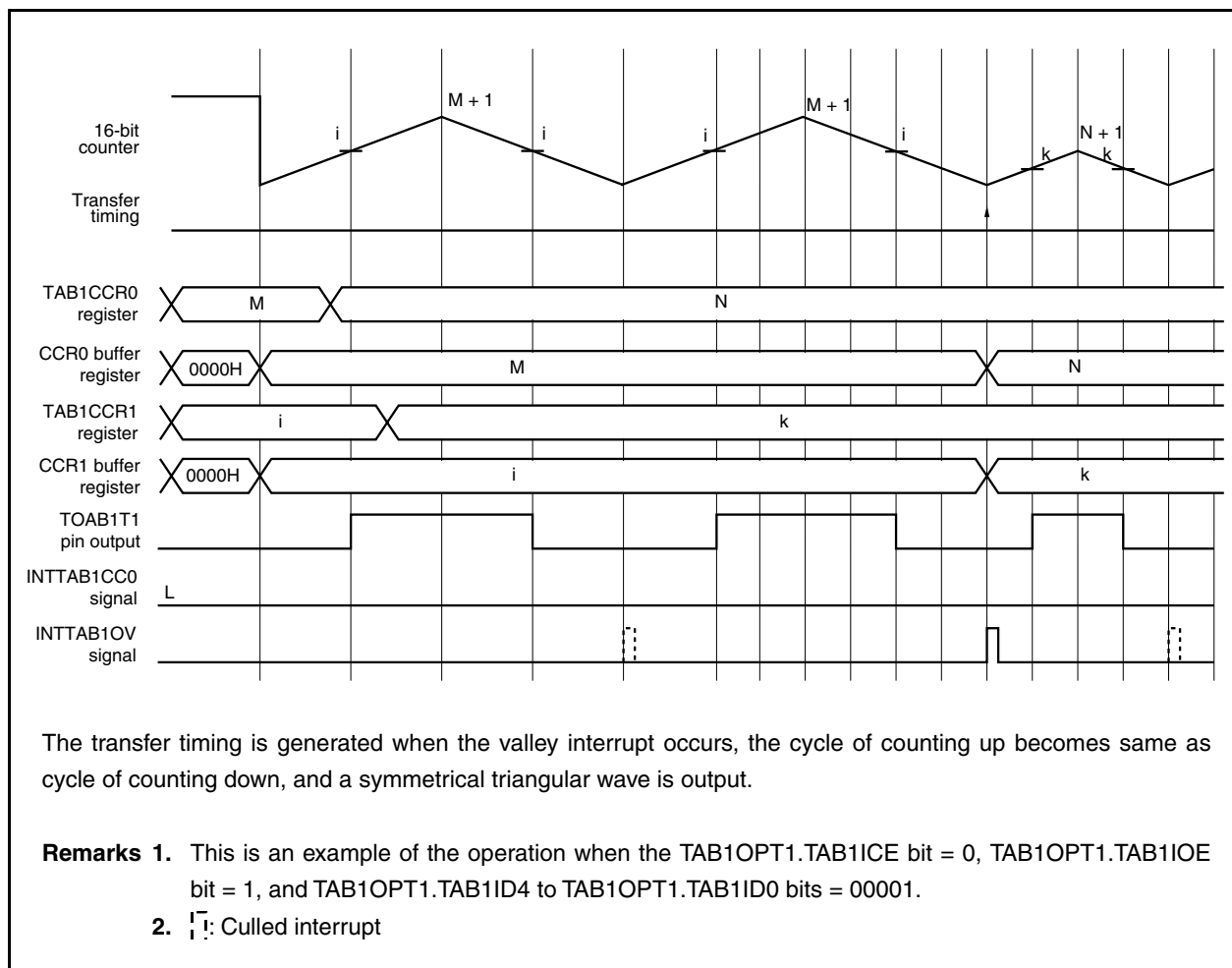
**(c) Rewriting TAB1CCR1 to TAB1CCR3 registers**

- Transfer at crest when crest interrupt is set
  Because the register is transferred at the transfer timing of the crest interrupt, an asymmetrical triangular wave is output.

**Figure 11-34.  Rewriting TAB1CCR1 Register (TAB1OPT1.TAB1ICE Bit = 1, TAB1OPT1.TAB1IOE Bit = 0, TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 Bits = 00001)**



**Remark** ⌐: Culled interrupt
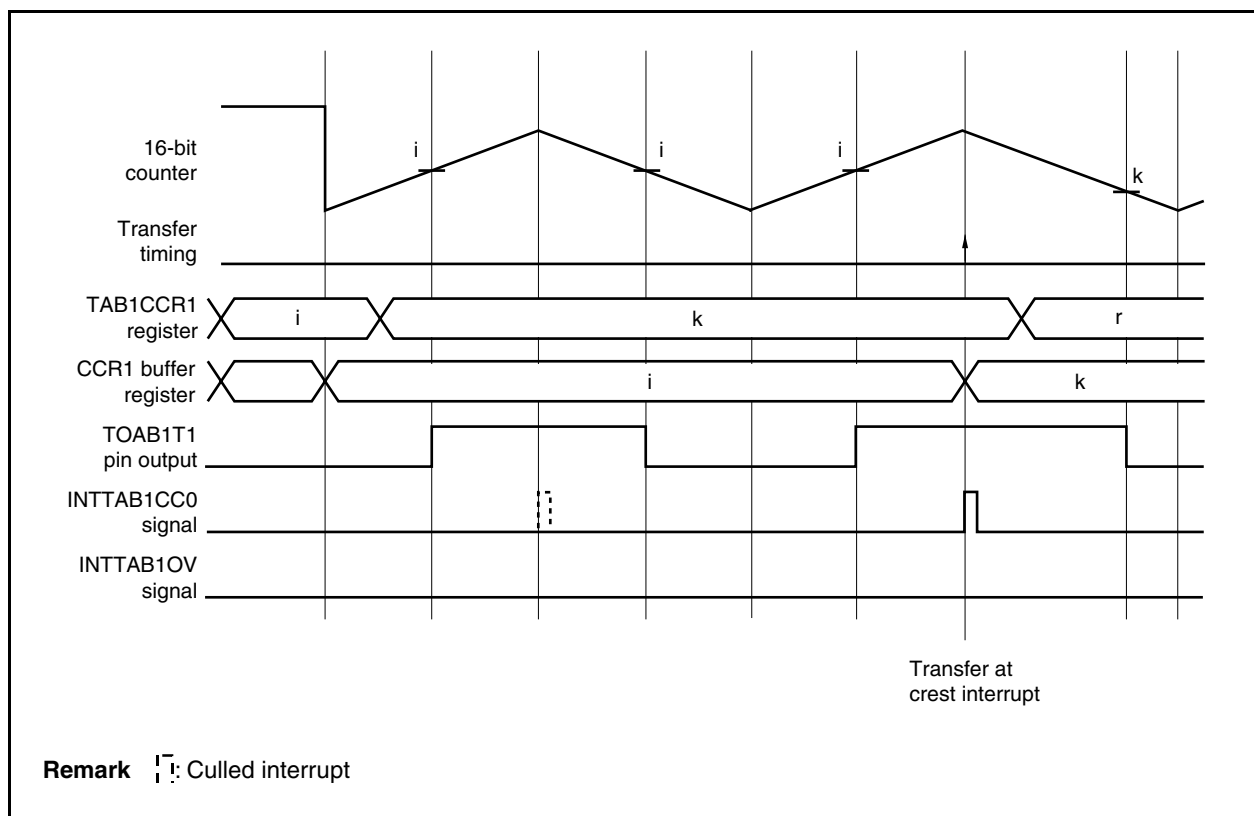
- Transfer at valley when valley interrupt is set

  Because the register is transferred at the transfer timing of the valley interrupt, a symmetrical triangular wave is output.

**Figure 11-35.  Rewriting TAB1CCR1 Register (TAB1OPT1.TAB1ICE Bit = 1, TAB1OPT1.TAB1IOE Bit = 1, TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 Bits = 00001)**



Remark  ⌐i: Culled interrupt
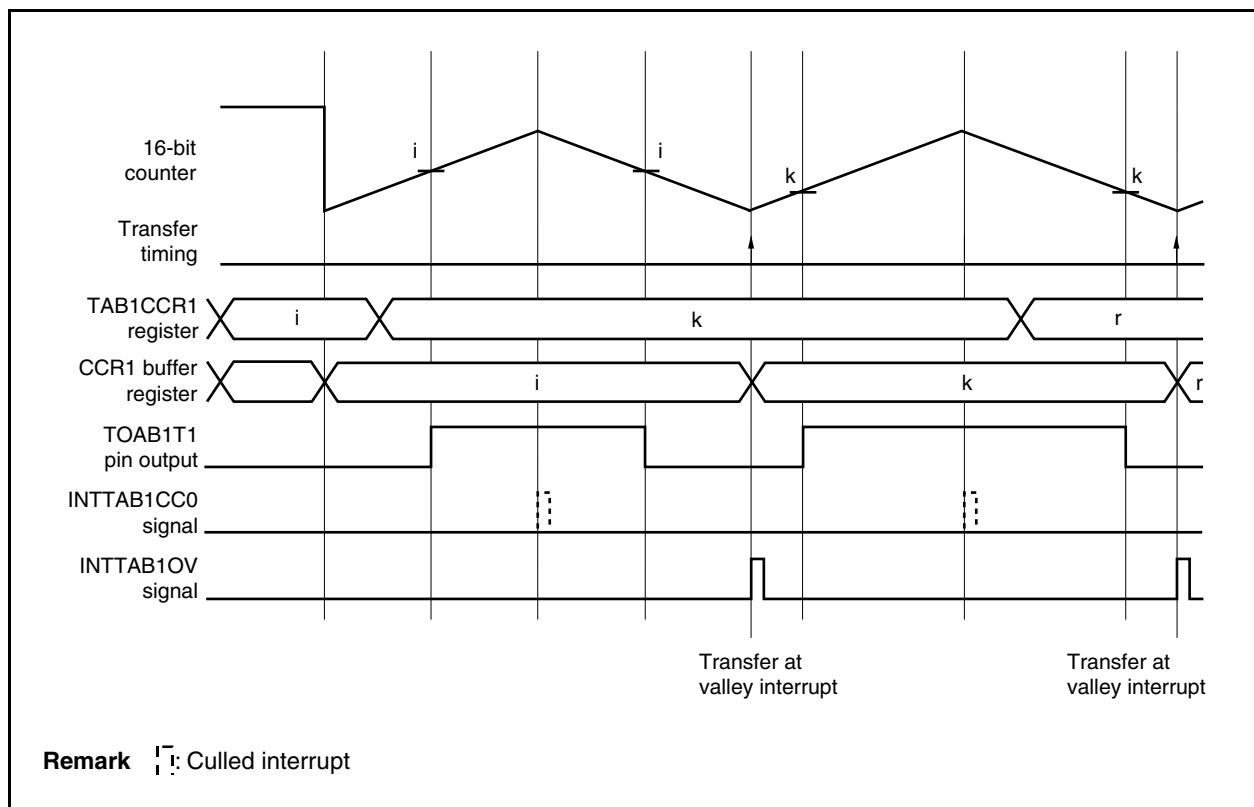
**(d) Rewriting TAB1OPT1 register**

Because a new interrupt culling value is transferred when the value of the interrupt culling counter matches the value of the 16-bit counter, the next interrupt and those that follow occur at the set interval.

For details of rewriting the TAB1OPT1 register, see **11.4.3  Interrupt culling function**.

**(4) Rewriting TAB1OPT0.TAB1CMS bit**

The TAB1CMS bit can select the anytime rewrite mode and batch rewrite mode.  This bit can be rewritten during timer operation (when TAB1CTL0.TAB1CE bit = 1).  However, the operation and caution illustrated in Figure 11-36 are necessary.

If the TAB1CCR1 register is written when the TAB1CMS bit is cleared to 0, a transfer request signal (internal signal) is set.

When the transfer request signal is set, the register is transferred at the next transfer timing, and the transfer request signal is cleared.  This transfer request signal is also cleared when the TAB1CMS bit is set to 1.

**Figure 11-36.  Rewriting TAB1CMS Bit**



<1> If the TAB1CCR1 register is rewritten when the TAB1CMS bit is 0, the transfer request signal is set.
    If the TAB1CMS bit is set to 1 in this status, the transfer request signal is cleared.

<2> The register is not transferred because the TAB1CMS bit is set to 1 and the transfer request signal is cleared.

<3> The transfer request signal is not set even if the TAB1CCR1 register is written when the TAB1CMS bit is 1.

<4> The transfer request signal is not set even if the TAB1CCR1 register is written when the TAB1CMS bit is 1, so even if the TAB1CMS bit is cleared to 0, transfer does not occur at the subsequent transfer timing.

<5> The transfer request signal is set if the TAB1CCR1 register is written when the TAB1CMS bit is 0.
    Transfer is performed at the subsequent transfer timing and the transfer request signal is cleared.

<6> Once transfer has been performed, the transfer request signal is cleared.  Therefore, transfer is not performed at the next transfer timing.

**11.4.5  TAA4 tuning operation for A/D conversion start trigger signal output**

This section explains the tuning operation of TAA4 and TAB1 in the 6-phase PWM output mode.

In the 6-phase PWM output mode, the tuning operation is performed with TAB1 serving as the master and TAA4 as a slave.  The conversion start trigger signal of the A/D converter can be set as the A/D conversion start trigger source by the INTTAA4CC0 and INTTAA4CC1 signals of TAA4 and the INTTAB1OV and INTTAB1CC0 signals of TAB1.

**(1)  Tuning operation starting procedure**

The TAA4 and TAB1 registers should be set using the following procedure to perform the tuning operation.

**(a)  Setting of TAA4 register (stop the operations of TAB1 and TAA4 (by clearing the TAB1CTL0.TAB1CE bit and TAA4CTL0.TAA4CE bit to 0)).**
- Set the TAA4CTL1 register to 85H (set the tuning operation slave mode and free-running timer mode).
- Clear the TAA4OPT0 register to 00H (select the compare register).
- Set an appropriate value to the TAA4CCR0 and TAA4CCR1 registers (set the default value for comparison for starting the operation).

**(b)  Setting of TAB1 register**
- Set the TAB1CTL1 register to 07H (master mode and 6-phase PWM output mode).
- Set an appropriate value to the TAB1IOC0 register (set the output mode of TOAB1T1 to TOAB1T3).
  However, clear the TAB1OL0 bit to 0 and set the TAB1OE0 bit to 1 (enable positive phase output).  Unless this setting is made, the crest interrupt (INTTAB1CC0) and valley interrupt (INTTAB1OV) do not occur.  Consequently, the conversion start trigger signal of the A/D converter is not correctly generated.
- Set the TAB1IOC1 and TAB1IOC2 registers to 00H (the TIAB10 to TIAB13, EVTB1, and TRGB1 pins of TAB1 are not used).
- Clear the TAB1OPT0 register to 00H (select the compare register).
- Set an appropriate value to the TAB1CCR0 to TAB1CCR3 registers (set the default value for comparison for starting the operation).
- Set the TAB1CTL0 register to 0xH (clear the TAB1CE bit to 0 and set the operating clock of TAB1).
- The operating clock of TAB1 set by the TAB1CTL0 register is also supplied to TAA4, and the count operation is performed at the same timing.  The operating clock of TAA4 set by the TAA4CTL0 register is ignored.

**(c)  Setting of TMQOP (TMQ option) register**
- Set an appropriate value to the TAB1OPT1 and TAB1OPT2 registers.
- Set an appropriate value to the TAB1IOC3 register (set TOAB1B1 to TOAB1B3 in the output mode).
- Set an appropriate value to the TAB1DTC register (set the default value for comparison for starting the operation).

**(d)  Setting of alternate function**
- Set the port to alternate function mode using the port control mode setting.

**(e) Set the TAA4CE bit to 1 and set the TAB1CE bit to 1 immediately after that to start the 6-phase PWM output operation**

Rewriting the TAB1CTL0, TAB1CTL1, TAB1IOC1, TAB1IOC2, TAA4CTL0, and TAA4CTL1 registers is prohibited during operation.  The operation and the PWM output waveform are not guaranteed if any of these registers is rewritten during operation.  However, rewriting the TAB1CTL0.TAB1CE bit to clear it is permitted.  Manipulating (reading/writing) the other TAB1, TAA4, and TMQ option registers is prohibited until the TAA4CTL0.TAA4CE bit is set to 1 and then the TAB1CE bit is set to 1.

**(2) Tuning operation clearing procedure**

To clear the tuning operation and exit the 6-phase PWM output mode, set the TAA4 and TAB1 registers using the following procedure.

<1>  Clear the TAB1CTL0.TAB1CE bit to 0 and stop the timer operation.

<2>  Clear the TAA4CTL0.TAA4CE bit to 0 so that TAA4 can be separated.

<3>  Stop the timer output by using the TAB1IOC0 register.

<4>  Clear the TAA4CTL1.TAA4SYE bit to 0 to clear the tuning operation.

**Caution    Manipulating (reading/writing) the other TAB1, TAA4, and TMQ option registers is prohibited until the TAB1CE bit is set to 1 and then the TAA4CE bit is set to 1.**

**(3) When not tuning TAA4**

When the match interrupt signal of TAA4 is not necessary as the conversion trigger source that starts the A/D converter, TAA4 can be used independently as a separate timer without being tuned.  In this case, the match interrupt signal of TAA4 cannot be used as a trigger source to start A/D conversion in the 6-phase PWM output mode.  Therefore, fix the TAB1OPT2.TAB1AT0 to TAB1OPT2.TAB1AT3 bits to 0.

The other control bits can be used in the same manner as when TAA4 is tuned.

If TAA4 is not tuned, the compare registers (TAA4CCR0 and TAA4CCR1) of TAA4 are not affected by the setting of the TAB1OPT0.TAB1CMS and TAB1OPT2.TAB1RDE bit.  For the initialization procedure when TAA4 is not tuned, see (b) to (e) in **11.4.5 (1)  Tuning operation starting procedure**.  (a) is not necessary because it is a step used to set TAA4 for the tuning operation.

**(4) Basic operation of TAA4 during tuning operation**

The 16-bit counter of TAA4 only counts up.  The 16-bit counter is cleared by the set cycle value of the TAB1CCR0 register and starts counting from 0000H again.  The count value of this counter is the same as the value of the 16-bit counter of TAB1 when it counts up.  However, it is not the same when the 16-bit counter of TAA4 counts down.

- When TAB1 counts up (same value)
  16-bit counter of TAB1: 0000H $\rightarrow$ M (counting up)
  16-bit counter of TAA4: 0000H $\rightarrow$ M (counting up)
- When TAB1 counts down (not same value)
  16-bit counter of TAB1: M + 1 $\rightarrow$ 0001H (counting down)
  16-bit counter of TAA4: 0000H $\rightarrow$ M (counting up)

**Figure 11-37. TAA4 During Tuning Operation**



**Note** The TABTADT0 signal is masked by the TAB1OPT2.TAB1ATM2 and TAB1OPT2.TAB1ATM3 bits.
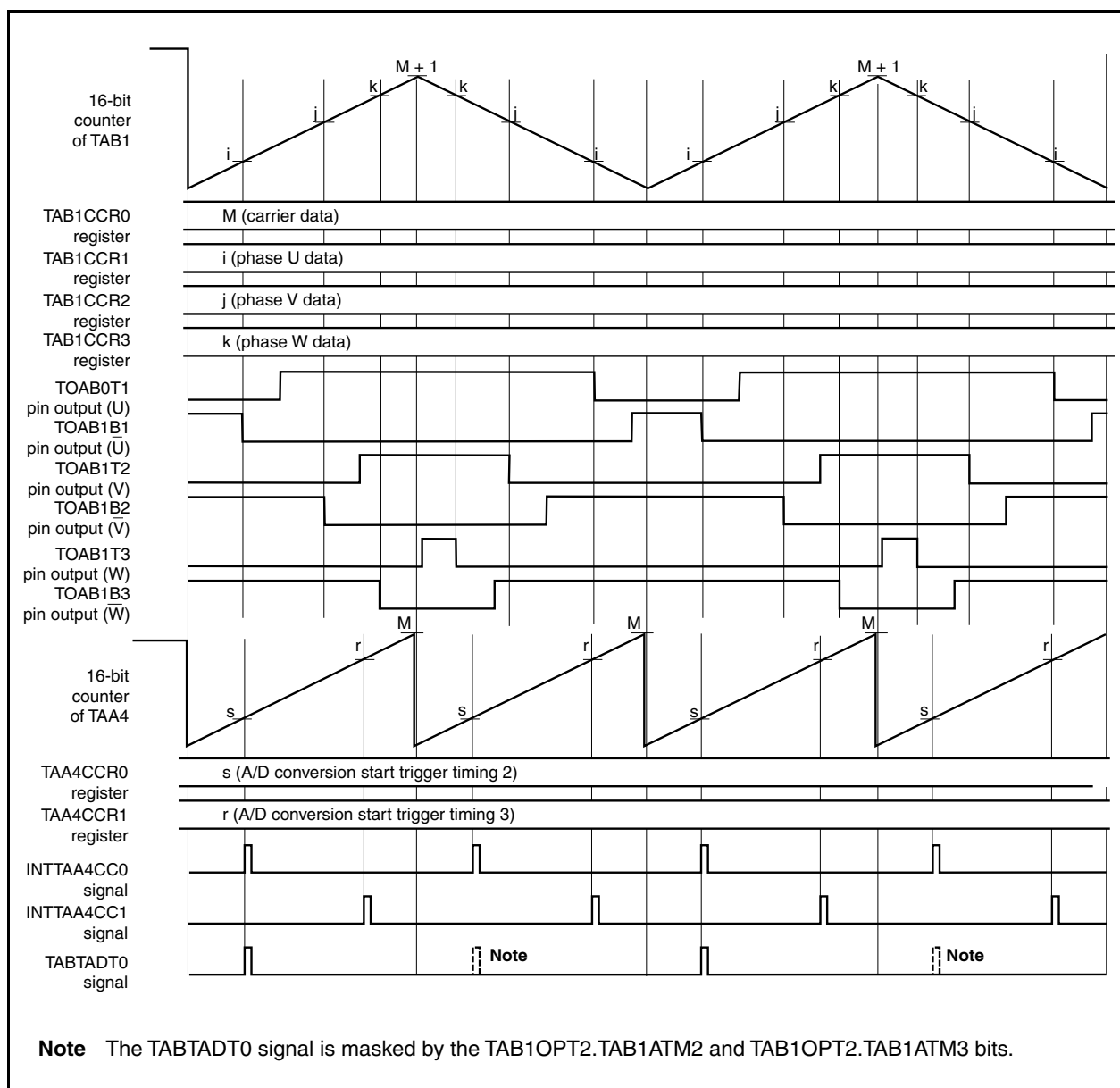
**11.4.6  A/D conversion start trigger output function**

The V850ES/JG3-U and V850ES/JH3-U have a function to select four trigger sources (INTTAB1OV, INTTAB1CC0, INTTAA4CC0, INTTAA4CC1) to generate the A/D conversion start trigger signal (TABTADT0).

The trigger sources are specified by the TAB1OPT2.TAB1AT0 to TAB1OPT2.TAB1AT3 bits.

- TAB1AT0 bit = 1:

    A/D conversion start trigger signal generated when INTTAB1OV (counter underflow) occurs.

- TAB1AT1 bit = 1:

    A/D conversion start trigger signal generated when INTTAB1CC0 (cycle match) occurs.

- TAB1AT2 bit = 1:

    A/D conversion start trigger signal generated when INTTAA4CC0 (match of TAA4CCR0 register of TAA4 during tuning operation) occurs.

- TAB1AT3 bit = 1:

    A/D conversion start trigger signal generated when INTTAA4CC1 (match of TAA4CCR1 register of TAA4 during tuning operation) occurs.

The A/D conversion start trigger signals selected by the TAB1AT0 to TAB1AT3 bits are ORed and output.  Therefore, two or more trigger sources can be specified at the same time.

The INTTAB1OV and INTTAB1CC0 signals selected by the TAB1AT0 and TAB1AT1 bits are culled interrupt signals.

Therefore, these signals are output after the interrupts have been culled and, unless interrupt output is enabled (by the TAB1OPT1.TAB1ICE and TAB1OPT1.TAB1IOE bits), the A/D conversion start trigger signal is not output.

The trigger sources (INTTAA4CC0 and INTTAA4CC1) from TAA4 have a function to mask the A/D conversion start trigger signal depending on the count-up/count-down status of the 16-bit counter, if so set by the TAB1AT2 and TAB1AT3 bits.

- TAB1ATM2 bit: Corresponds to the TAB1AT2 bit and controls INTTAA4CC0 (match interrupt signal) of TAA4.
    - TAB1ATM2 bit = 0: The A/D conversion start trigger signal is output when the 16-bit counter counts up (TAB1OPT0.TAB1CUF bit = 0), and the A/D conversion start trigger signal is not output when the 16-bit counter counts down (TAB1OPT0.TAB1CUF bit = 1).
    - TAB1ATM2 bit = 1: The A/D conversion start trigger signal is output when the 16-bit counter counts up (TAB1OPT0.TAB1CUF bit = 1), and the A/D conversion start trigger signal is not output when the 16-bit counter counts down (TAB1OPT0.TAB1CUF bit = 0).

- TAB1ATM3 bit: Corresponds to the TAB1AT3 bit and controls INTTAA4CC1 (match interrupt signal) of TAA4.
    - TAB1ATM3 bit = 0: The A/D conversion start trigger signal is output when the 16-bit counter counts up (TAB1OPT0.TAB1CUF bit = 0), and the A/D conversion start trigger signal is not output when the 16-bit counter counts down (TAB1OPT0.TAB1CUF bit = 1).
    - TAB1ATM3 bit = 1: The A/D conversion start trigger signal is output when the 16-bit counter counts up (TAB1OPT0.TAB1CUF bit = 1), and the A/D conversion start trigger signal is not output when the 16-bit counter counts down (TAB1OPT0.TAB1CUF bit = 0).

The TAB1ATM3, TAB1ATM2, and TAB1AT3 to TAB1AT0 bits can be rewritten while the timer is operating.  If the bit that sets the A/D conversion start trigger signal is rewritten while the timer is operating, the new setting is immediately reflected in the output status of the A/D conversion start trigger signal.  These control bits do not have a transfer function and can be used only in the anytime rewrite mode.

**Cautions 1.** **The A/D conversion start trigger signal output that is set by the TAB1AT2 and TAB1AT3 bits can be used only when TAA4 is performing a tuning operation as the slave timer of TAB1. If TAB1 and TAA4 are not performing a tuning operation, or if a mode other than the 6-phase PWM output mode is used, the output cannot be guaranteed.**

**2.** **The TAB1 signal output is internally used to identify whether the 16-bit counter is counting up or down. Therefore, enable TOAB10 pin output by clearing the TAB1IOC0.TAB1OL0 bit to 0 and setting the TAB1IOC0.TAB1OE0 bit to 1.**

**Figure 11-38.  Example of A/D Conversion Start Trigger (TABTADT0) Signal Output
(TAB1OPT1.TAB1ICE Bit = 1, TAB1OPT1.TAB1IOE Bit = 1,
TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 Bits = 00000: Without Interrupt Culling)**
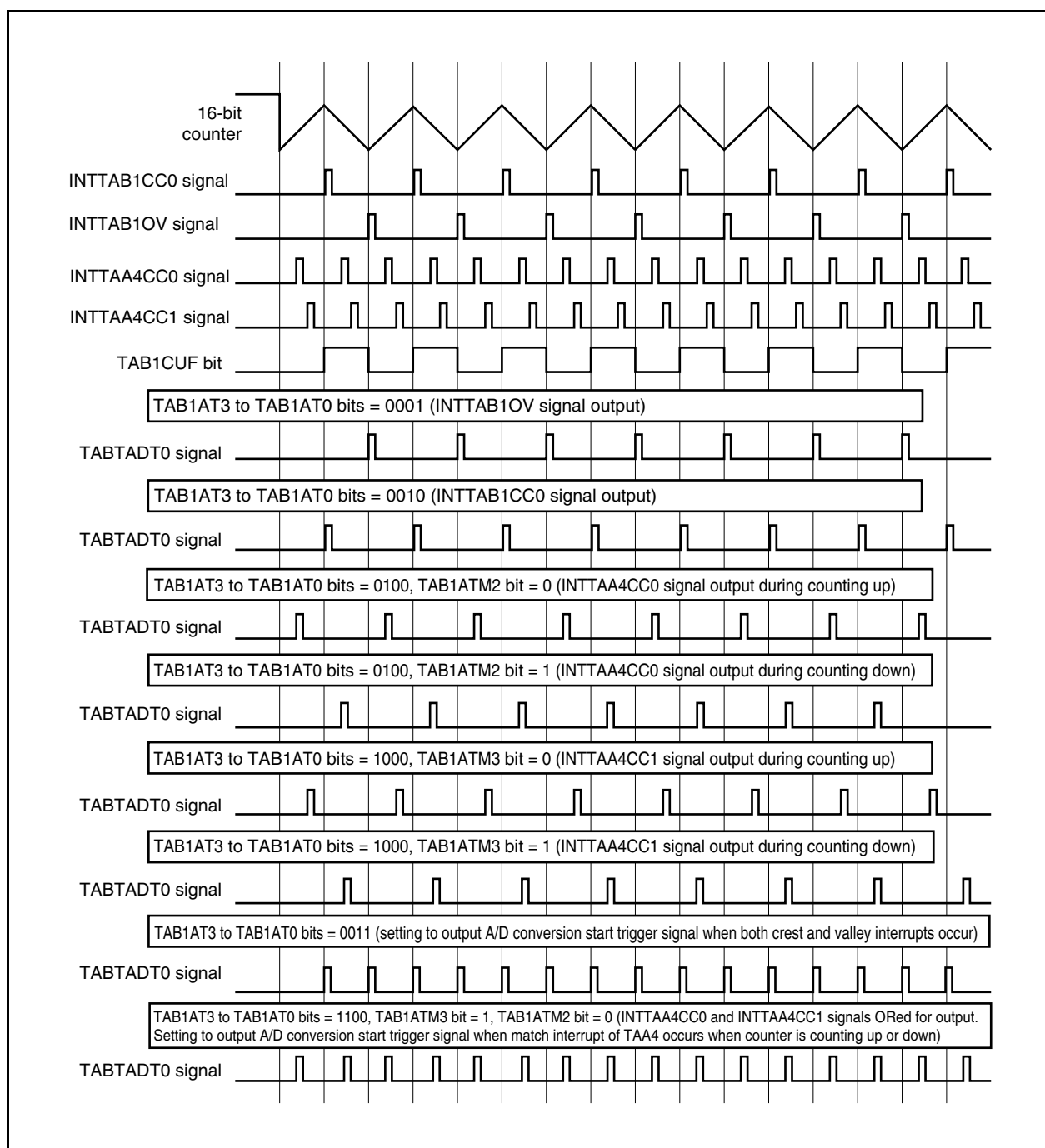
**Figure 11-39.  Example of A/D Conversion Start Trigger (TABTADT0) Signal Output
(TAB1OPT1.TAB1ICE Bit = 0, TAB1OPT1.TAB1IOE Bit = 1,
TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 Bits = 00010: With Interrupt Culling) (1)**
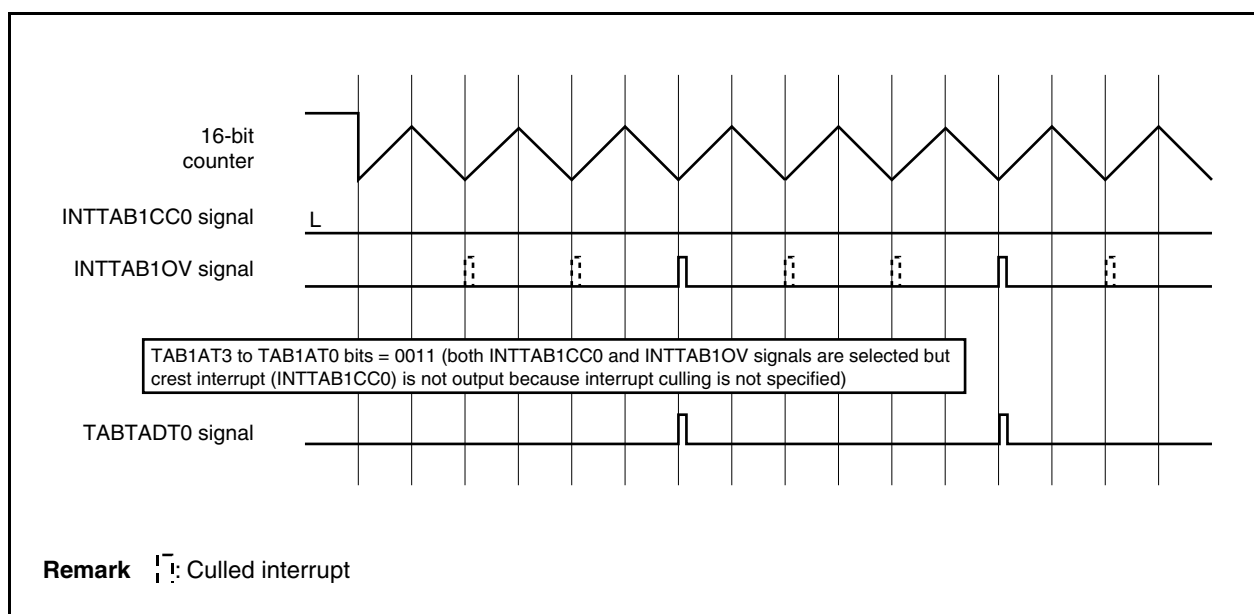


**Figure 11-40.  Example of A/D Conversion Start Trigger (TABTADT0) Signal Output
(TAB1OPT1.TAB1ICE Bit = 0, TAB1OPT1.TAB1IOE Bit = 1,
TAB1OPT1.TAB1ID4 to TAB1OPT1.TAB1ID0 Bits = 00010: With Interrupt Culling) (2)**



**Caution   The INTTAB1CC0 signal is culled but the INTTAA4CC0 signal is not.**

**Remark**  ⌐¦: Culled interrupt
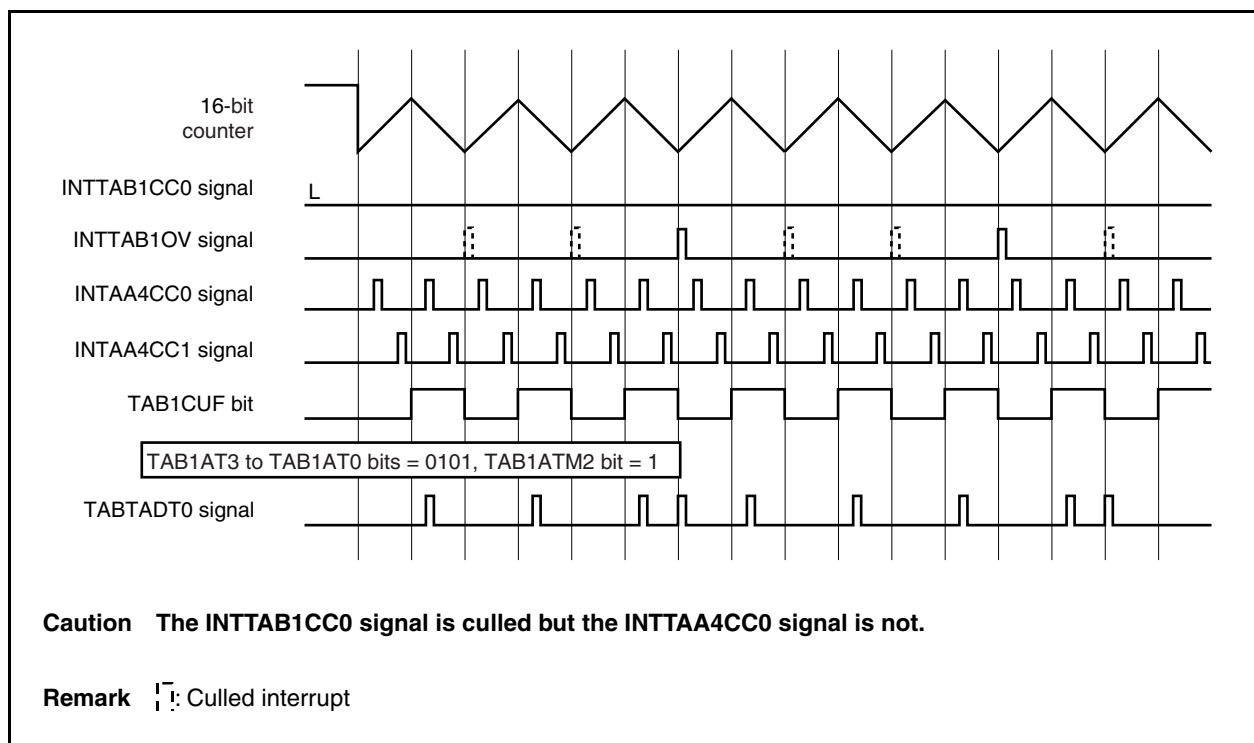
**(1)  Operation under boundary condition (operation when 16-bit counter matches INTTAA4CC0 signal)**

**Table 11-3.  Operation When TAB1CCR0 Register = M, TAB1AT2 Bit = 1, TAB1ATM2 Bit = 0**
**(Counting Up Period Selected)**

| Value of TAA4CCR0 Register | Value of 16-Bit Counter of TAB1 | Value of 16-Bit Counter of TAA4 | Status of 16-Bit Counter of TAB1 | TABTADT0 Signal Output by INTTAA4CC0 Signal |
|---|---|---|---|---|
| 0000H | 0000H | 0000H | – | Output |
| 0000H | M + 1 | 0000H | – | Not output |
| 0001H | 0001H | 0001H | Count-up | Output |
| 0001H | M | 0001H | Count-down | Not output |
| M | M | M | Count-up | Output |
| M | 0001H | M | Count-down | Not output |

**Table 11-4.  Operation When TAB1CCR0 Register = M, TAB1AT2 Bit = 1, TAB1ATM2 Bit = 1**
**(Counting Down Period Selected)**

| Value of TAA4CCR0 Register | Value of 16-Bit Counter of TAB1 | Value of 16-Bit Counter of TAA4 | Status of 16-Bit Counter of TAB1 | TABTADT0 Signal Output by INTTAA4CC0 Signal |
|---|---|---|---|---|
| 0000H | 0000H | 0000H | – | Not output |
| 0000H | M + 1 | 0000H | – | Output |
| 0001H | 0001H | 0001H | Count-up | Not output |
| 0001H | M | 0001H | Count-down | Output |
| M | M | M | Count-up | Not output |
| M | 0001H | M | Count-down | Output |

**Caution**  **The TAA4CCRm register enables the setting of "0" to "M" when the TAB1CCR0 register = M.  Setting a value of "M + 1" or higher is prohibited.**
**If a value of "M + 1" or higher is set, the 16-bit counter of TAA4 is cleared by "M".  Therefore, the TABTADT0 signal is not output.**

**Remark**  m = 0, 1

**CHAPTER 12 REAL-TIME COUNTER**

## 12.1 Functions

The real-time counter (RTC) has the following features.

- Counting up to 99 years using year, month, day-of-week, day, hour, minute, and second sub-counters provided
- Year, month, day-of-week, day, hour, minute, and second counter display using BCD codes[Note 1]
- Alarm interrupt function
- Constant-period interrupt function (period: 1 month to 0.5 second)
- Interval interrupt function (period: 1.95 ms to 125 ms)
- Pin output function of 1 Hz
- Pin output function of 32.768 kHz
- Pin output function of 512 Hz or 16.384 kHz
- Watch error correction function
- Subclock operation or main clock operation[Note 2] selectable

**Notes 1.** A BCD (binary coded decimal) code expresses each digit of a decimal number in 4-bit binary format.
  **2.** Use the baud rate generator dedicated to the real-time counter to divide the main clock frequency to 32.768 kHz for use.

## 12.2 Configuration

The real-time counter includes the following hardware.

**Table 12-1. Configuration of Real-Time Counter**

| Item | Configuration |
|---|---|
| Control registers | Real-time counter control register 0 (RC1CC0) |
| | Real-time counter control register 1 (RC1CC1) |
| | Real-time counter control register 2 (RC1CC2) |
| | Real-time counter control register 3 (RC1CC3) |
| | Sub-count register (RC1SUBC) |
| | Second count register (RC1SEC) |
| | Minute count register (RC1MIN) |
| | Hour count register (RC1HOUR) |
| | Day count register (RC1DAY) |
| | Day-of-week count register (RC1WEEK) |
| | Month count register (RC1MONTH) |
| | Year count register (RC1YEAR) |
| | Watch error correction register (RC1SUBU) |
| | Alarm minute register (RC1ALM) |
| | Alarm hour register (RC1ALH) |
| | Alarm week register (RC1ALW) |
| | Prescaler mode register 0 (PRSM0) |
| | Prescaler compare register 0 (PRSCM0) |

**Figure 12-1. Block Diagram of Real-Time Counter**
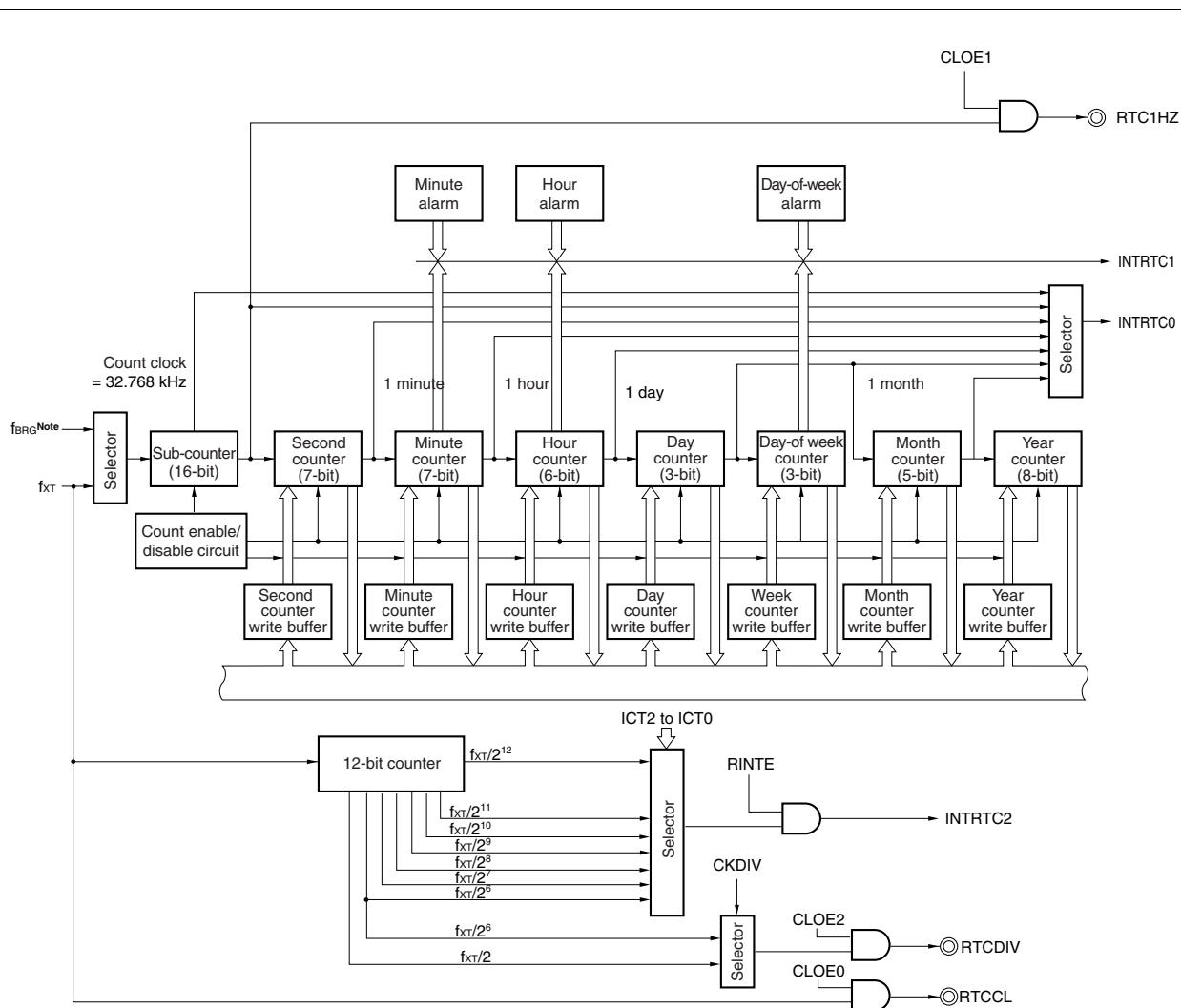


**Note** For detail of $f_{BRG}$, refer to **12.3 (17) Prescaler mode register 0 (PRSM0)** and **12.3 (18) Prescaler compare register 0 (PRSCM0)**.

**Remark** $f_{BRG}$: Real-time counter count clock frequency

$f_{XT}$: Subclock frequency

INTRTC0: Real-time counter fixed-cycle interrupt signal

INTRTC1: Real-time counter alarm match interrupt signal

INTRTC2: Real-time counter interval interrupt signal

### 12.2.1  Pin configuration

The RTC outputs included in the real-time counter are alternatively used as shown in Table 12-2.  The port function must be set when using each pin (see **Table 4-20 Using Port Pin as Alternate-Function Pin**).

**Table 12-2.  Pin Configuration**

| Pin Number | | Port | RTC Output | Other Alternate Function |
|---|---|---|---|---|
| V850ES/JG3-U | V850ES/JH3-U | | | |
| 30 | 42 | P35 | RTC1HZ | TIAA11/TOAA11/$\overline{\text{OCI}}$ |
| 28 | 40 | P33 | RTCDIV | TIAA01/TOAA01/RTCCL |
| 28 | 40 | P33 | RTCCL | TIAA01/TOAA01/RTCDIV |

### 12.2.2  Interrupt functions

The RTC includes the following three types of interrupt signals.

#### (1)  INTRTC0

A fixed-cycle interrupt signal is generated every 0.5 second, second, minute, hour, day, or month.

#### (2)  INTRTC1

Alarm interrupt signal

#### (3)  INTRTC2

An interval interrupt signal of a cycle of $f_{XT}/2^6$, $f_{XT}/2^7$, $f_{XT}/2^8$, $f_{XT}/2^9$, $f_{XT}/2^{10}$, $f_{XT}/2^{1}1$, or $f_{XT}/2^{12}$ is generated.

## 12.3  Registers

The real-time counter is controlled by the following 16 registers.

### (1)  Real-time counter control register 0 (RC1CC0)

The RC1CC0 register selects the real-time counter input clock.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset:  00H     R/W     Address:  FFFFFADDH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1CC0 | RC1PWR | RC1CKS | 0 | 0 | 0 | 0 | 0 | 0 |

| RC1PWR | Real-time counter operation control |
|---|---|
| 0 | Stops real-time counter operation. |
| 1 | Enables real-time counter operation. |

| RC1CKS | Operation clock selection |
|---|---|
| 0 | Selects $f_{XT}$ as operation clock. |
| 1 | Selects $f_{BRG}$ as operation clock. |

**Cautions 1.  Follow the description in 12.4.8 Initializing real-time counter when stopping (RC1PWR = 1 → 0) the real-time counter while it is operating.**

**2.  The RC1CKS bit can be rewritten only when the real-time counter is stopped (RC1PWR bit = 0).  Furthermore, rewriting the RC1CKS bit at the same time as setting the RC1PWR bit from 0 to 1 is prohibited.**

### (2)  Real-time counter control register 1 (RC1CC1)

The RC1CC1 register is an 8-bit register that starts or stops the real-time counter, controls the RTCCL and RTC1HZ pins, selects the 12-hour or 24-hour system, and sets the fixed-cycle interrupt function.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset:  00H      R/W      Address:  FFFFFADEH

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1CC1 | RTCE | 0 | CLOE1 | CLOE0 | AMPM | CT2 | CT1 | CT0 |

| RCTE | Control of operation of each counter |
|---|---|
| 0 | Stops counter operation. |
| 1 | Enables counter operation. |

| CLOE1 | RTC1HZ pin output control |
|---|---|
| 0 | Disables RTC1HZ pin output (1 Hz) |
| 1 | Enables RTC1HZ pin output (1 Hz) |

| CLOE0 | RTCCL pin output control |
|---|---|
| 0 | Disables RTCCL pin output (32.768 kHz) |
| 1 | Enables RTCCL pin output (32.768 kHz) |

| AMPM | 12-hour system/24-hour system selection |
|---|---|
| 0 | 12-hour system (a.m. and p.m. are displayed.) |
| 1 | 24-hour system |

| CT2 | CT1 | CT0 | Fixed-cycle interrupt (INTRTC0) selection |
|---|---|---|---|
| 0 | 0 | 0 | Does not use fixed-cycle interrupts |
| 0 | 0 | 1 | Once in 0.5 second (synchronous with second count-up) |
| 0 | 1 | 0 | Once in 1 second (simultaneous with second count-up) |
| 0 | 1 | 1 | Once in 1 minute (every minute at 00 seconds) |
| 1 | 0 | 0 | Once in 1 hour (every hour at 00 minutes 00 seconds) |
| 1 | 0 | 1 | Once in 1 day (every day at 00 hours 00 minutes 00 seconds) |
| 1 | 1 | × | Once in 1 month (one day every month at 00 hours 00 minutes 00 seconds a.m.) |

**Cautions 1. Writing 0 to the RTCE bit while the RTCE bit is 1 is prohibited. Clear the RTCE bit by clearing the RC1PWR bit according to 12.4.8 Initializing real-time counter.**

**2. The RTC1HZ output operates as follows when the CLOE1 bit setting is changed.**
   - **When changed from 0 to 1: The RTC1HZ output outputs a 1 Hz pulse after two clocks (2 x 32.768 kHz) or less.**
   - **When changed from 1 to 0: The RTC1HZ output is stopped (fixed to low level) after two clocks (2 x 32.768 kHz) or less.**

**3. See 12.4.1 Initial settings and 12.4.2 Rewriting each counter during the real-time counter operation for setting or changing the AMPM bit. Furthermore, re-set the RC1HOUR register when the AMPM bit is rewritten.**

**4. See 12.4.4 Changing INTRTC0 interrupt setting during the real-time counter operation when rewriting the CT2 to CT0 bits while the real-time counter operates (RC1PWR bit = 1).**

**(3) Real-time counter control register 2 (RC1CC2)**

The RC1CC2 register is an 8-bit register that controls the alarm interrupt function and waiting of counters.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset:  00H     R/W     Address:  FFFFFADFH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1CC2 | WALE | 0 | 0 | 0 | 0 | 0 | RWST | RWAIT |

| WALE | Alarm interrupt (INTRTC1) operation control |
|---|---|
| 0 | Does not generate interrupt upon alarm match. |
| 1 | Generates interrupt upon alarm match. |

| RWST | Real-time counter wait state |
|---|---|
| 0 | Counter operating |
| 1 | Counting up of second to year counters stopped (Reading and writing of counter values enabled) |

This is a status flag indicating whether the RWAIT bit setting is valid.
Read or write counter values after confirming that the RWST bit is 1.

| RWAIT | Real-time counter wait control |
|---|---|
| 0 | Sets counter operation. |
| 1 | Stops count operation of second to year counters. (Counter value read/write mode) |

This bit controls the operation of the counters.
Be sure to write 1 to this bit when reading or writing counter values.
If the RC1SUBC register overflows while the RWAIT bit is 1, the overflow information is retained internally and the RC1SEC register is counted up after two clocks or less after 0 is written to the RWAIT bit.
However, if the second counter value is rewritten while the RWAIT bit is 1, the retained overflow information is discarded.

**Cautions 1.  See 12.4.5 Changing INTRTC1 interrupt setting during the real-time counter operation when rewriting the WALE bit while the real-time counter operates (RC1PWR bit = 1).**

**2.  Confirm that the RWST bit is set to 1 when reading or writing each counter value.**

**3.  The RWST bit does not become 0 while each counter is being written, even if the RWAIT bit is set to 0.  It becomes 0 when writing to each counter is completed.**

**(4) Real-time counter control register 3 (RC1CC3)**

The RC1CC3 register is an 8-bit register that controls the interval interrupt function and RTCDIV pin.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H　　　R/W　　　Address: FFFFFAE0H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1CC3 | RINTE | CLOE2 | CKDIV | 0 | 0 | ICT2 | ICT1 | ICT0 |

| RINTE | Interval interrupt (INTRTC2) control |
|---|---|
| 0 | Does not generate interval interrupt. |
| 1 | Generates interval interrupt. |

| CLOE2 | RTCDIV pin output control |
|---|---|
| 0 | Disables RTCDIV pin output. |
| 1 | Enables RTCDIV pin output. |

| CKDIV | RTCCL pin output control |
|---|---|
| 0 | Outputs 512 Hz (1.95 ms) from RTCDIV pin. |
| 1 | Outputs 16.384 kHz (0.061 ms) from RTCDIV pin. |

| ICT2 | ICT1 | ICT0 | Interval interrupt (INTRTC2) selection |
|---|---|---|---|
| 0 | 0 | 0 | $2^6/f_{XT}$ (1.953125 ms) |
| 0 | 0 | 1 | $2^7/f_{XT}$ (3.90625 ms) |
| 0 | 1 | 0 | $2^8/f_{XT}$ (7.8125 ms) |
| 0 | 1 | 1 | $2^9/f_{XT}$ (15.625 ms) |
| 1 | 0 | 0 | $2^{10}/f_{XT}$ (31.25 ms) |
| 1 | 0 | 1 | $2^{11}/f_{XT}$ (62.5 ms) |
| 1 | 1 | × | $2^{12}/f_{XT}$ (125 ms) |

**Cautions 1. See 12.4.7 Changing INTRTC2 interrupt setting during the real-time counter operation when rewriting the RINTE bit during real-time counter operation (RC1PWR bit = 1).**

**2. The RTCDIV output operates as follows when the CLOE2 bit setting is changed.**
- **When changed from 0 to 1:  A pulse set by the CKDIV bit is output after two clocks (2 x 32.768 kHz) or less.**
- **When changed from 1 to 0:   Output of the RTCDIV output is stopped after two clocks (2 x 32.768 kHz)or less (fixed to low level).**

**3. See 12.4.7 Changing INTRTC2 interrupt setting during the real-time counter operation when rewriting the ICT2 to ICT0 bits while the real-time counter operates (RC1PWR bit = 1).**

**(5) Sub-count register (RC1SUBC)**

The RC1SUBC register is a 16-bit register that counts the reference time of 1 second of the real-time counter.

It takes a value of 0000H to 7FFFH and counts one second with a clock of 32.768 kHz.

This register is read-only, in 16-bit units.

Reset sets this register to 0000H.

**Cautions 1  When a correction is made by using the RC1SUBU register, the value may become 8000H or more.**

**2. This register is also cleared by writing to the second count register.**

**3. The value read from this register is not guaranteed if it is read during operation, because a changing value is read.**

<table>
<tr><td colspan="17">After reset:  0000H    R    Address:    FFFFFAD0H</td></tr>
<tr><td></td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>RC1SUBC</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

**(6) Second count register (RC1SEC)**

The RC1SEC register is an 8-bit register that takes a value of 0 to 59 (decimal) and indicates the count value of seconds.

It counts up when the sub-counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (2 x 32.768 kHz) later.  Set a decimal value of 00 to 59 to this register in BCD code.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**Caution   Setting the RC1SEC register to values other than 00 to 59 is prohibited.**

**Remark   See 12.4.1 Initial settings, 12.4.2 Rewriting each counter during the real-time counter operation, and 12.4.3 Reading each counter during the real-time counter operation when reading or writing the RC1SEC register.**

<table>
<tr><td colspan="9">After reset:  00H    R/W    Address:  FFFFFAD2H</td></tr>
<tr><td></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>RC1SEC</td><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
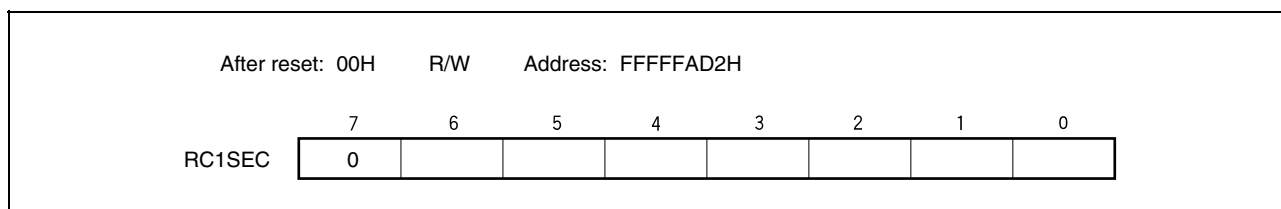
**(7) Minute count register (RC1MIN)**

The RC1MIN register is an 8-bit register that takes a value of 0 to 59 (decimal) and indicates the count value of minutes.

It counts up when the second counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (32.768 kHz) later.

Set a decimal value of 00 to 59 to this register in BCD code.

This register can be read or written 8-bit units.

Reset sets this register to 00H.

**Caution    Setting a value other than 00 to 59 to the RC1MIN register is prohibited.**

**Remark**    See **12.4.1 Initial settings**, **12.4.2 Rewriting each counter during the real-time counter operation**, and **12.4.3 Reading each counter during the real-time counter operation** when reading or writing the RC1MIN register.

After reset:  00H      R/W      Address:  FFFFFAD3H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1MIN | 0 | | | | | | | |
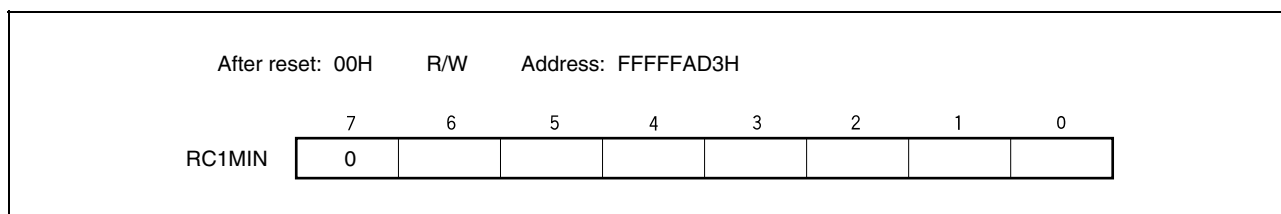
**(8) Hour count register (RC1HOUR)**

The RC1HOUR register is an 8-bit register that takes a value of 0 to 23 or 1 to 12 (decimal) and indicates the count value of hours.

It counts up when the minute counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (32.768 kHz) later.

Set a decimal value of 00 to 23, 01 to 12, or 21 to 32 to this register in BCD code.

This register can be read or written 8-bit units.

Reset sets this register to 12H.

However, the value of this register is 00H if the AMPM bit is set to 1 after reset.

**Cautions 1. Bit 5 of the RC1HOUR register indicates a.m. (0) or p.m. (1) if AMPM = 0 (if the 12-hour system is selected).**

**2. Setting a value other than 01 to 12, 21 to 32 (AMPM bit= 0), or 00 to 23 (AMPM bit = 1) to the RC1HOUR register is prohibited.**

**Remark**    See **12.4.1 Initial settings**, **12.4.2 Rewriting each counter during the real-time counter operation**, and **12.4.3 Reading each counter during the real-time counter operation** when reading or writing the RC1HOUR register.

After reset:  12H      R/W      Address:  FFFFFAD4H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1HOUR | 0 | 0 | | | | | | |
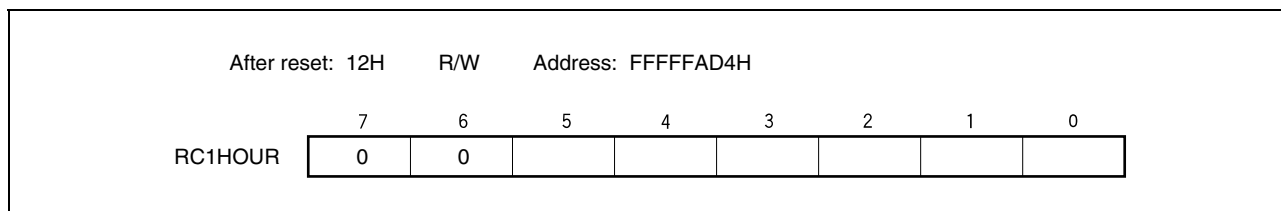
Table 12-3 shows the relationship among the AMPM bit setting value, RC1HOUR register value, and time.

**Table 12-3. Time Digit Display**

| 12-Hour Display (AMPM Bit = 0) | | 24-Hour Display (AMPM Bit = 1) | |
|---|---|---|---|
| Time | RC1HOUR Register Value | Time | RC1HOUR Register Value |
| 0:00 a.m. | 12 H | 0:00 | 00H |
| 1:00 a.m. | 01 H | 1:00 | 01 H |
| 2:00 a.m. | 02 H | 2:00 | 02 H |
| 3:00 a.m. | 03 H | 3:00 | 03 H |
| 4:00 a.m. | 04 H | 4:00 | 04 H |
| 5:00 a.m. | 05 H | 5:00 | 05 H |
| 6:00 a.m. | 06 H | 6:00 | 06 H |
| 7:00 a.m. | 07 H | 7:00 | 07 H |
| 8:00 a.m. | 08 H | 8:00 | 08 H |
| 9:00 a.m. | 09 H | 9:00 | 09 H |
| 10:00 a.m. | 10 H | 10:00 | 10 H |
| 11:00 a.m. | 11 H | 11:00 | 11 H |
| 0:00 p.m. | 32 H | 12:00 | 12 H |
| 1:00 p.m. | 21 H | 13:00 | 13 H |
| 2:00 p.m. | 22 H | 14:00 | 14 H |
| 3 :00 p.m. | 23 H | 15:00 | 15 H |
| 4:00 p.m. | 24 H | 16:00 | 16 H |
| 5:00 p.m. | 25 H | 17:00 | 17 H |
| 6:00 p.m. | 26 H | 18:00 | 18 H |
| 7:00 p.m. | 27 H | 19:00 | 19 H |
| 8:00 p.m. | 28 H | 20:00 | 20 H |
| 9:00 p.m. | 29 H | 21:00 | 21 H |
| 10:00 p.m. | 30 H | 22:00 | 22 H |
| 11:00 p.m. | 31 H | 23:00 | 23 H |

The RC1HOUR register value is displayed in 12 hour-format if the AMPM bit is 0 and in 24-hour format when the AMPM bit is 1.

In 12-hour display, a.m. or p.m. is indicated by the fifth bit of RCHOUR: 0 indicating before noon (a.m.) and 1 indicating noon or afternoon (p.m.).

**(9) Day count register (RC1DAY)**

The RC1DAY register is an 8-bit register that takes a value of 1 to 31 (decimal) and indicates the count value of days.

It counts up when the hour counter overflows.

This counter counts as follows.

- 01 to 31 (January, March, May, July, August, October, December)
- 01 to 30 (April, June, September, November)
- 01 to 29 (February in leap year)
- 01 to 28 (February in normal year)

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (2 x 32.768 kHz) later.  Set a decimal value of 00 to 31 to this register in BCD code.

This register can be read or written in 8-bit units.

Reset sets this register to 01H.

**Caution**   **Setting a value other than 01 to 31 to the RC1DAY register is prohibited.  Setting a value outside the above-mentioned count range, such as "February 30" is also prohibited.**

**Remark**   See **12.4.1 Initial settings**, **12.4.2 Rewriting each counter during the real-time counter operation**, and **12.4.3 Reading each counter during the real-time counter operation** when reading or writing the RC1DAY register.

After reset:  01H        R/W        Address:  FFFFFAD6H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1DAY | 0 | 0 | | | | | | |

**(10) Day-of-week count register (RC1WEEK)**

The RC1WEEK register is an 8-bit register that takes a value of 0 to 6 (decimal) and indicates the day-of-week count value.

It counts up in synchronization with the day counter.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (2 x 32.768 kHz) later. Set a decimal value of 00 to 06 to this register in BCD code. If a value outside this range is set, the register value returns to the normal value after 1 period.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address: FFFFFAD5H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1WEEK | 0 | 0 | 0 | 0 | 0 | | | |

**Cautions 1. Setting a value other than 00 to 06 to the RC1WEEK register is prohibited.**

**2. Values corresponding to the month count register and day count register are not automatically stored to the day-of-week register.**

**Be sure to set as follows after rest release.**

| Day of Week | RC1WEEK |
|---|---|
| Sunday | 00H |
| Monday | 01H |
| Tuesday | 02H |
| Wednesday | 03H |
| Thursday | 04H |
| Friday | 05H |
| Saturday | 06H |

**Remark** See **12.4.1 Initial settings**, **12.4.2 Rewriting each counter during the real-time counter operation**, and **12.4.3 Reading each counter during the real-time counter operation** when reading or writing the RC1WEEK register.

**(11) Month count register (RC1MONTH)**

The RC1MONTH register is an 8-bit register that takes a value of 1 to 12 (decimal) and indicates the count value of months.

It counts up when the day counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (2 x 32.768 kHz) later.  Set a decimal value of 01 to 12 to this register in BCD code.

This register can be read or written in 8-bit units.

Reset sets this register to 01H.

**Caution    Setting a value other than 01 to 12 to the RC1MONTH register is prohibited.**

**Remark**    See **12.4.1 Initial settings**, **12.4.2 Rewriting each counter during the real-time counter operation**, and **12.4.3 Reading each counter during the real-time counter operation** when reading or writing the RC1MONTH register.

| | After reset:  01H | | R/W | | Address:  FFFFFAD7H | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RC1MONTH | 0 | 0 | 0 | | | | | |

**(12) Year count register (RC1YEAR)**

The RC1YEAR register is an 8-bit register that takes a value of 0 to 99 (decimal) and indicates the count value of years.

It counts up when the month counter overflows.

Values 00, 04, 08, …, 92, and 96 indicate a leap year.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (2 x 32.768 kHz) later.  Set a decimal value of 00 to 99 to this register in BCD code.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**Caution    Setting a value other than 00 to 99 to the RC1YEAR register is prohibited.**

**Remark**    See **12.4.1 Initial settings**, **12.4.2 Rewriting each counter during the real-time counter operation**, and **12.4.3 Reading each counter during the real-time counter operation** when reading or writing the RC1YEAR register.

| | After reset:  00H | | R/W | | Address:  FFFFFAD8H | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RC1YEAR | | | | | | | | |

**(13)  Watch error correction register (RC1SUBU)**

The RC1SUBU register can be used to correct the watch with high accuracy when the watch is early or late, by changing the value (reference value: 7FFFH) overflowing from the sub-count register (RSUBC) to the second counter register.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

**Remarks 1.** The RC1SUBU register can be rewritten only when the real-time counter is set to its initial values. Be sure to see **12.4.1 Initial settings**.

**2.** See **12.4.9 Watch error correction example of real-time counter** for details of watch error correction.

---

After reset:  00H        R/W        Address:  FFFFFAD9H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1SUBU | DEV | F6 | F5 | F4 | F3 | F2 | F1 | F0 |

| DEV | Setting of watch error correction timing |
|---|---|
| 0 | Corrects watch errors when RC1SEC (second counter) is at 00, 20, or 40 seconds (every 20 seconds). |
| 1 | Corrects watch errors when RC1SEC (second counter) is at 00 seconds (every 60 seconds). |

| F6 | Setting of watch error correction value |
|---|---|
| 0 | Increments the RC1SUBC count value by the value set using the F5 to F0 bits (positive correction).<br>Expression for calculating increment value:<br>$\qquad$ (Setting value of F5 to F0 bits $- 1) \times 2$ |
| 1 | Decrements the RC1SUBC count value by the value set using the F5 to F0 bits (negative correction).<br>Expression for calculating decrement value:<br>$\qquad$ (Inverted value of setting value of F5 to F0 bits $+ 1) \times 2$ |
| | If the F6 to F0 bit values are {1/0, 0, 0, 0, 0, 0, 1/0}, watch error correction is not performed. |

**(14) Alarm minute setting register (RC1ALM)**

The RC1ALM register (8-bit) is used to set minutes of alarm.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**Caution   Set a decimal value of 00 to 59 to this register in BCD code.  If a value outside the range is set, the alarm is not detected.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|

After reset:  00H      R/W      Address:  FFFFFADAH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1ALM | 0 | | | | | | | |

**(15) Alarm hour setting register (RC1ALH)**

The RC1ALH register (8-bit) is used to set hours of alarm.

This register can be read or written in 8-bit units.

Reset sets this register to 12H.

**Cautions 1.   Set a decimal value of 00 to 23, 01 to 12, or 21 to 32 to this register in BCD code.  If a value outside the range is set, the alarm is not detected.**

**2.   Bit 5 of the RC1ALH register indicates a.m. (0) or p.m. (1) if the AMPM bit = 0 (12-hour system) is selected.**

After reset:  12H      R/W      Address:  FFFFFADBH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1ALH | 0 | 0 | | | | | | |

**(16) Alarm day-of-week setting register (RC1ALW)**

The RC1ALW register is used to set the day-of-week of the alarm.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**Caution    See 12.4.5 Changing INTRTC1 interrupt setting during the real-time counter operation when rewriting the RC1ALW register while the real-time counter operates (RC1PWR bit = 1).**
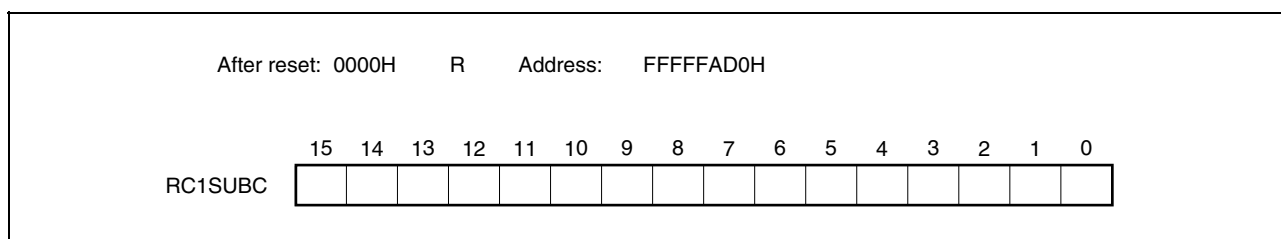
After apply power to RV$_{DD}$: 00H    R/W    Address: FFFFFADCH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RC1ALW | 0 | RC1ALW6 | RC1ALW5 | RC1ALW4 | RC1ALW3 | RC1ALW2 | RC1ALW1 | RC1ALW0 |

| RC1ALWn | Alarm interrupt day-of-week bit (n = 0 to 6) |
|---|---|
| 0 | Does not generate alarm interrupt if RC1WEEK = nH. |
| 1 | Generates an alarm interrupt if the time specified by using the RC1ALM and RC1ALH registers is reached while RC1WEEK is set to nH. |

**Remark**    The relationship between the day-of-week and the RC1WEEK register is described below.

| Day of Week | RC1WEEK |
|---|---|
| Sunday | 00H |
| Monday | 01H |
| Tuesday | 02H |
| Wednesday | 03H |
| Thursday | 04H |
| Friday | 05H |
| Saturday | 06H |

**(a) Alarm interrupt setting examples (RC1ALM, RC1ALH, and RC1ALW setting examples)**

Tables 12-4 and 12-5 show setting examples if Sunday is RC1WEEK = 00, Monday is RC1WEEK = 01, Tuesday is RC1WEEK = 02, …, and Saturday is RC1WEEK = 06.

**Table 12-4.  Alarm Setting Example if AMPM = 0 (RC1HOUR Register 12-Hour Display)**

| Register \ Alarm Setting Time | RC1ALW | RC1ALH | RC1ALM |
|---|---|---|---|
| Sunday, 7:00 a.m. | 01H | 07H | 00H |
| Sunday/Monday, 00:15 p.m. | 03H | 32H | 15H |
| Monday/Tuesday/Friday, 5:30 p.m. | 26H | 25H | 30H |
| Everyday, 10:45 p.m. | 7FH | 30H | 45H |

**Table 12-5.  Alarm Setting Example if AMPM = 1 (RC1HOUR Register 24-Hour Display)**

| Register \ Alarm Setting Time | RC1ALW | RC1ALH | RC1ALM |
|---|---|---|---|
| Sunday, 7:00 | 01H | 07H | 00H |
| Sunday/Monday, 12:15 | 03H | 12H | 15H |
| Monday/Tuesday/Friday, 17:30 | 26H | 17H | 30H |
| Everyday, 22:45 | 7FH | 22H | 45H |

**(17) Prescaler mode register 0 (PRSM0)**

The PRSM0 register controls the generation of the real time counter count clock ($f_{BRG}$).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address: FFFFF8B0H

| | 7 | 6 | 5 | <4> | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PRSM0 | 0 | 0 | 0 | BGCE0 | 0 | 0 | BGCS01 | BGCS00 |

| BGCE0 | Main clock operation enable |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

| BGCS01 | BGCS00 | Selection of real-time counter source clock ($f_{BGCS}$) | | |
|---|---|---|---|---|
| | | | 5 MHz | 4 MHz |
| 0 | 0 | $f_X$ | 200 ns | 250 ns |
| 0 | 1 | $f_X/2$ | 400 ns | 500 ns |
| 1 | 0 | $f_X/4$ | 800 ns | 1 $\mu$s |
| 1 | 1 | $f_X/8$ | 1.6 $\mu$s | 2 $\mu$s |

**Cautions 1. Do not change the values of the BGCS00 and BGCS01 bits during real-time counter operation.**

**2. Set the PRSM0 register before setting the BGCE0 bit to 1.**

**3. Set the PRSM0 and PRSCM0 registers according to the main clock frequency that is used so as to obtain an $f_{BRG}$ frequency of 32.768 kHz.**

**(18) Prescaler compare register 0 (PRSCM0)**

The PRSCM0 register is an 8-bit compare register.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

---

After reset:  00H     R/W     Address:  FFFFF8B1H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PRSCM0 | PRSCM07 | PRSCM06 | PRSCM05 | PRSCM04 | PRSCM03 | PRSCM02 | PRSCM01 | PRSCM00 |
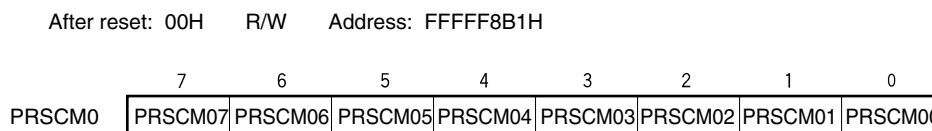
**Cautions 1.  Do not rewrite the PRSCM0 register during real time counter operation.**

**2.  Set the PRSCM0 register before setting the PRSM0.BGCE0 bit to 1.**

**3.  Set the PRSM0 and PRSCM0 registers according to the main clock frequency that is used so as to obtain an $f_{BRG}$ frequency of 32.768 kHz.**

---

The calculation for $f_{BRG}$ is shown below.

$$f_{BRG} = f_{BGCS}/2N$$

**Remark**  $f_{BGCS}$: Watch timer source clock set by the PRSM0 register

N:     Set value of the PRSCM0 register = 1 to 256

However, N = 256 when the PRSCM0 register is set to 00H.

## 12.4 Operation

### 12.4.1 Initial settings

The initial settings are set when operating the watch function and performing a fixed-cycle interrupt operation.

**Figure 12-2. Initial Setting Procedure**

```
                    ┌─────────────────┐
                    │      Start      │
                    └─────────────────┘
                             │
          ┌─────────────────────────────────┐
          │  RC1CC0.RC1PWR bit = 0           │  Stops counter operation.
          └─────────────────────────────────┘
                             │
          ┌─────────────────────────────────┐
          │       Setting RC1CKS            │  Selects real-time counter (RTC) operation clock.
          └─────────────────────────────────┘
                             │
          ┌─────────────────────────────────┐
          │  RC1CC0.RC1PWR bit = 1           │  Enables real-time counter (RTC) internal clock operation.
          └─────────────────────────────────┘
                             │
          ┌─────────────────────────────────┐
          │ Setting AMPM and CT2 to CT0      │  Selects 12-hour system or 24-hour system and interrupt (INTRTC0).
          └─────────────────────────────────┘
                             │
          ┌─────────────────────────────────┐
          │       Setting RC1SUBU           │  Sets watch error correction.
          └─────────────────────────────────┘
                             │
          ┌─────────────────────────────────┐
          │   Setting RC1SEC                 │  Sets each count register.
          │   (Clearing RC1SUBC)             │
          │   Setting RC1MIN                 │
          │   Setting RC1HOUR                │
          │   Setting RC1WEEK                │
          │   Setting RC1DAY                 │
          │   Setting RC1MONTH               │
          │   Setting RC1YEAR                │
          └─────────────────────────────────┘
                             │
          ┌─────────────────────────────────┐
          │   Clearing interrupt IF flag     │  Clears interrupt request flag (RTC0IF)
          └─────────────────────────────────┘
                             │
          ┌─────────────────────────────────┐
          │   Clearing interrupt MK flag     │  Clears interrupt mask flag (RTC0MK)
          └─────────────────────────────────┘
                             │
          ┌─────────────────────────────────┐
          │  RC1CC1.RTCE bit = 1             │  Starts counter operation.
          └─────────────────────────────────┘
                             │
                    ┌────────◄────────┐
              No     ◇                 │
          ◄──────────  INTRTC = 1?  ───┘
                      ◇
                       │ Yes
                    ┌─────────────────┐
                    │ Reading counter │
                    └─────────────────┘
```

### 12.4.2  Rewriting each counter during the real-time counter operation

Set as follows when rewriting each counter (RC1SEC, RC1MIN, RC1HOUR, RC1WEEK, RC1DAY, RC1MONTH, RC1YEAR) during the real-time counter operation (RC1PWR = 1, RTCE = 1).

**Figure 12-3.  Rewriting Each Counter During The real-time counter Operation**



**Note**   Be sure to confirm that RWST = 0 before setting STOP mode.

**Caution**   **Complete the series of operations for setting RWAIT to 1 to clearing RWAIT to 0 within 1 second.**
**If RWAIT = 1 is set, the operation of RC1SEC to RC1YEAR is stopped.  If a carry occurs from RC1SUBC while RWAIT = 1, one carry can be internally retained. However, if two or more carries occur, the number of carries cannot be retained.**

**Remark**   RC1SEC, RC1MIN, RC1HOUR, RC1WEEK, RC1DAY, RC1MONTH, and RC1YEAR may be rewritten in any sequence.
All the registers do not have to be set and only some registers may be read.

### 12.4.3  Reading each counter during the real-time counter operation

Set as follows when reading each counter (RC1SEC, RC1MIN, RC1HOUR, RC1WEEK, RC1DAY, RC1MONTH, RC1YEAR) during the real-time counter operation (RC1PWR = 1, RTCE = 1).

**Figure 12-4.  Reading Each Counter During The real-time counter Operation**
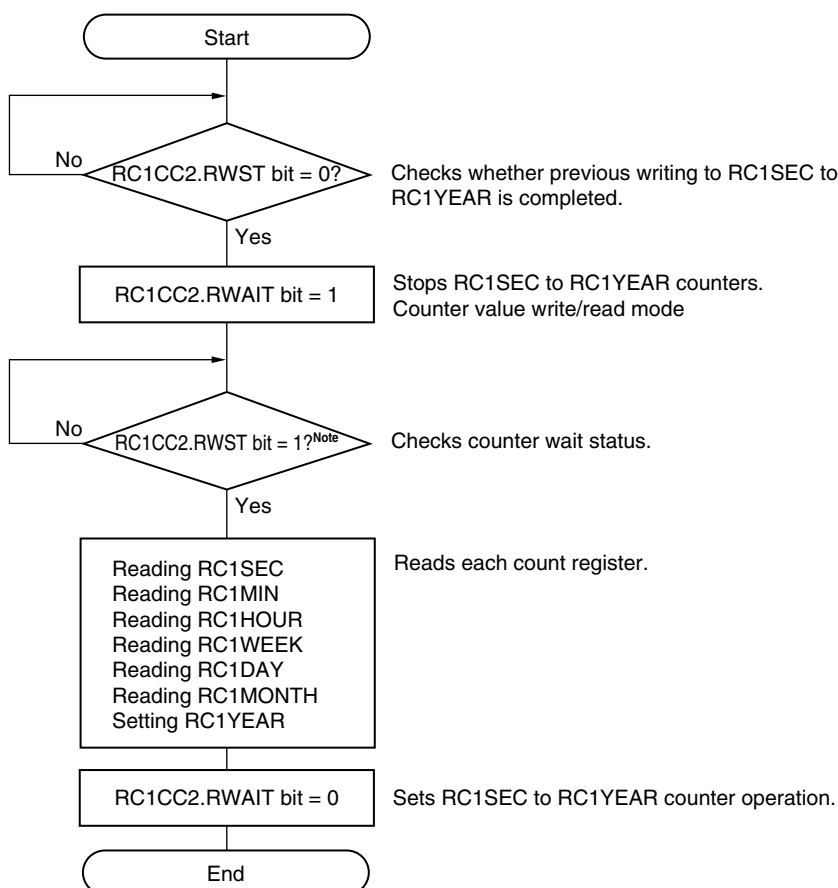


Note   Be sure to confirm that RWST = 0 before setting STOP mode.

Caution   **Complete the series of operations for setting RWAIT to 1 to clearing RWAIT to 0 within 1 second.**
**If RWAIT = 1 is set, the operation of RC1SEC to RC1YEAR is stopped.  If a carry occurs from RC1SUBC while RWAIT = 1, one carry can be internally retained. However, if two or more carries occur, the number of carries cannot be retained.**

Remark   RC1SEC, RC1MIN, RC1HOUR, RC1WEEK, RC1DAY, RC1MONTH, and RC1YEAR may be read in any sequence.
All the registers do not have to be set and only some registers may be read.

**12.4.4  Changing INTRTC0 interrupt setting during the real-time counter operation**

If the setting of the INTRTC0 interrupt (fixed-cycle interrupt) signal is changed while the real-time counter clock operates (PC1PWR = 1), the INTRCT0 interrupt waveform may include whiskers and unintended signals may be output. Set as follows when changing the setting of the INTRTC0 interrupt signal during the real-time counter operation (RC1PWR = 1, RTCE = 1), in order to mask the whiskers.

**Figure 12-5.  Changing INTRTC0 Interrupt Setting During The real-time counter Operation**
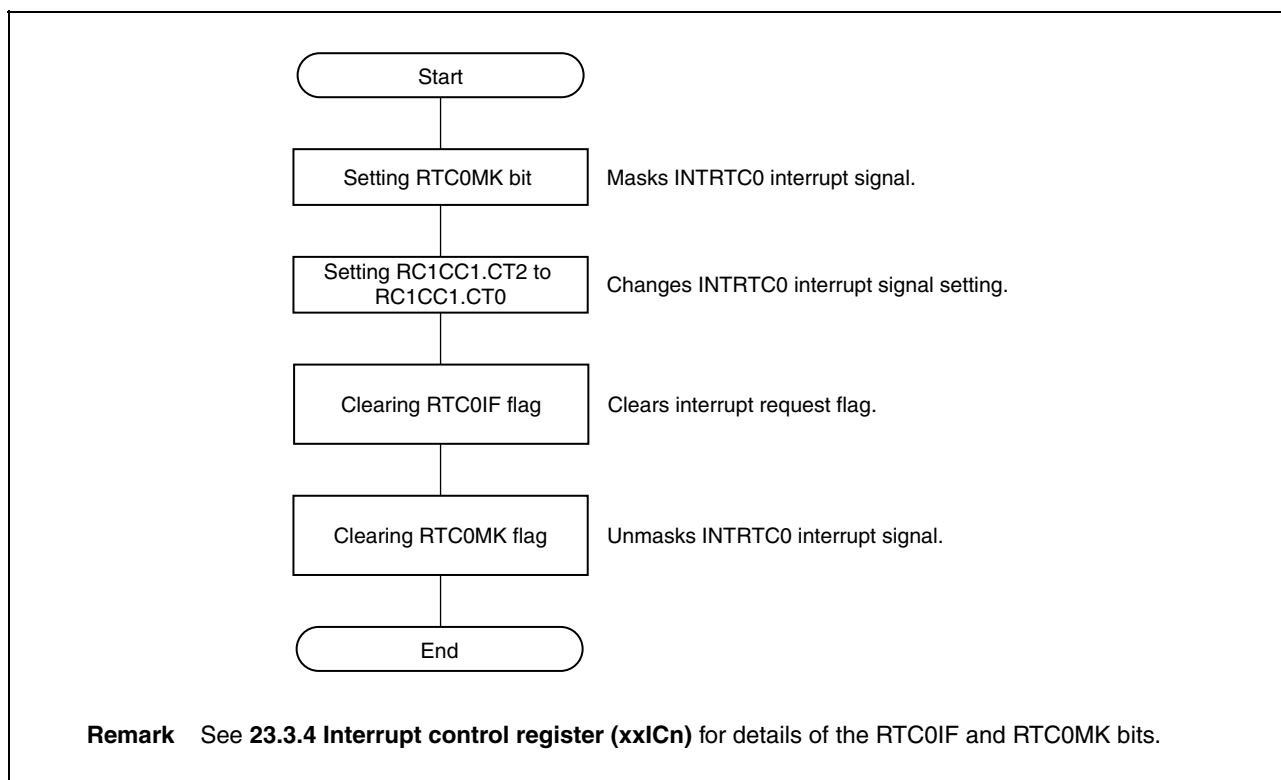


**Remark**   See **23.3.4 Interrupt control register (xxICn)** for details of the RTC0IF and RTC0MK bits.

### 12.4.5  Changing INTRTC1 interrupt setting during the real-time counter operation

If the setting of the INTRTC1 interrupt (alarm interrupt) signal is changed while the real-time counter clock operates (RC1PWR = 1), the INTRCT1 interrupt waveform may include whiskers and unintended signals may be output.  Set as follows when changing the setting of the INTRTC1 interrupt signal during the real-time counter operation (PC1PWR = 1, RTCE = 1), in order to mask the whiskers.

**Figure 12-6.  Changing INTRTC1 Interrupt Setting During The real-time counter Operation**

```
                        ┌──────────────────┐
                        │      Start        │
                        └──────────────────┘
                                 │
                    ┌────────────────────────┐
                    │   Setting RTC1MK bit    │   Masks interrupt signal (INTRTC1).
                    └────────────────────────┘
                                 │
                    ┌────────────────────────┐
                    │  RC1CC2.WALE bit = 0    │   Disables alarm interrupt.
                    └────────────────────────┘
                                 │
                    ┌────────────────────────┐
                    │    Setting RC1ALM       │   Sets alarm minute register.
                    │    Setting RC1ALH       │   Sets alarm hour register
                    │    Setting RC1ALW       │   Sets alarm day-of-week register
                    └────────────────────────┘
                                 │
                    ┌────────────────────────┐
                    │   Clearing RTC1IF flag  │   Clears interrupt pending bit.
                    └────────────────────────┘
                                 │
                    ┌────────────────────────┐
                    │  Clearing RTC1MK flag   │   Unmasks interrupt signal (INTRTC1).
                    └────────────────────────┘
                                 │
                    ┌────────────────────────┐
                    │  RC1CC2.WALE bit = 1    │   Enables alarm interrupt.
                    └────────────────────────┘
                                 │
                        ┌──────────────────┐
                        │       End         │
                        └──────────────────┘
```

**Remark**   See **23.3.4 Interrupt control register (xxICn)** for details of the RTC1IF and RTC1MK bits.

**12.4.6  Initial INTRTC2 interrupt settings**

Set as follows to set the INTRTC1 interrupt (interval interrupt).

**Figure 12-7.  INTRTC2 Interrupt Setting**
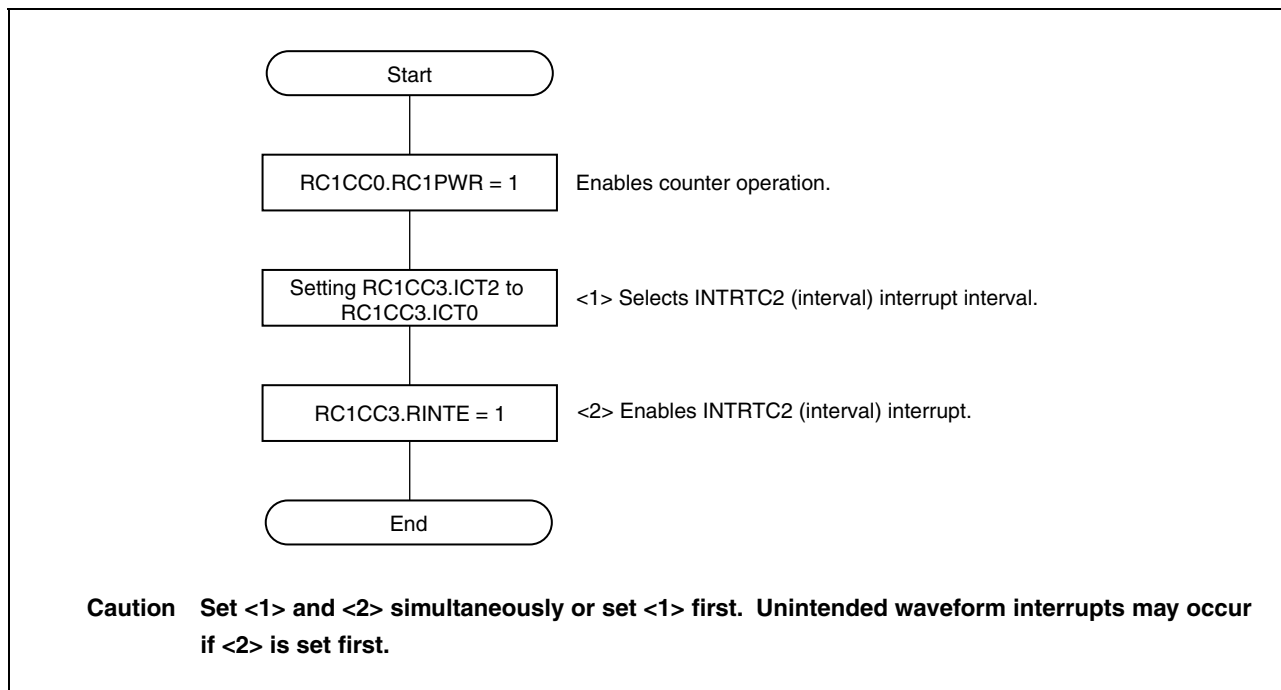


Caution   **Set <1> and <2> simultaneously or set <1> first.  Unintended waveform interrupts may occur if <2> is set first.**

### 12.4.7  Changing INTRTC2 interrupt setting during the real-time counter operation

If the setting of the INTRTC2 interrupt (interval interrupt) is changed while the real-time counter clock operates (PC1PWR = 1), the INTRCT2 interrupt waveform may include whiskers and unintended signals may be output.  Set as follows when changing the setting of the INTRTC2 interrupt signal during the real-time counter operation (PC1PWR = 1, RTCE = 1), in order to mask the whiskers.

**Figure 12-8.  Changing INTRTC2 Interrupt Setting During The real-time counter Operation**

| | |
|---|---|
| Start | |
| Setting RTC2MK bit | Masks interrupt signal (INTRTC2). |
| RC1CC3.RINTE = 1 | Enables INTRTC2 (interval) interrupt. |
| Setting RC1CC3.ICT2 to RC1CC3.ICT0 | Selects INTRTC2 (interval) interrupt interval. |
| Clearing RTC2IF flag | Clears interrupt pending bit. |
| Clearing RTC2MK flag | Unmasks interrupt signal (INTRTC2). |
| End | |

**Remark**   See **23.3.4 Interrupt control register (xxICn)** for details of the RTC2IF and RTC2MK bits.
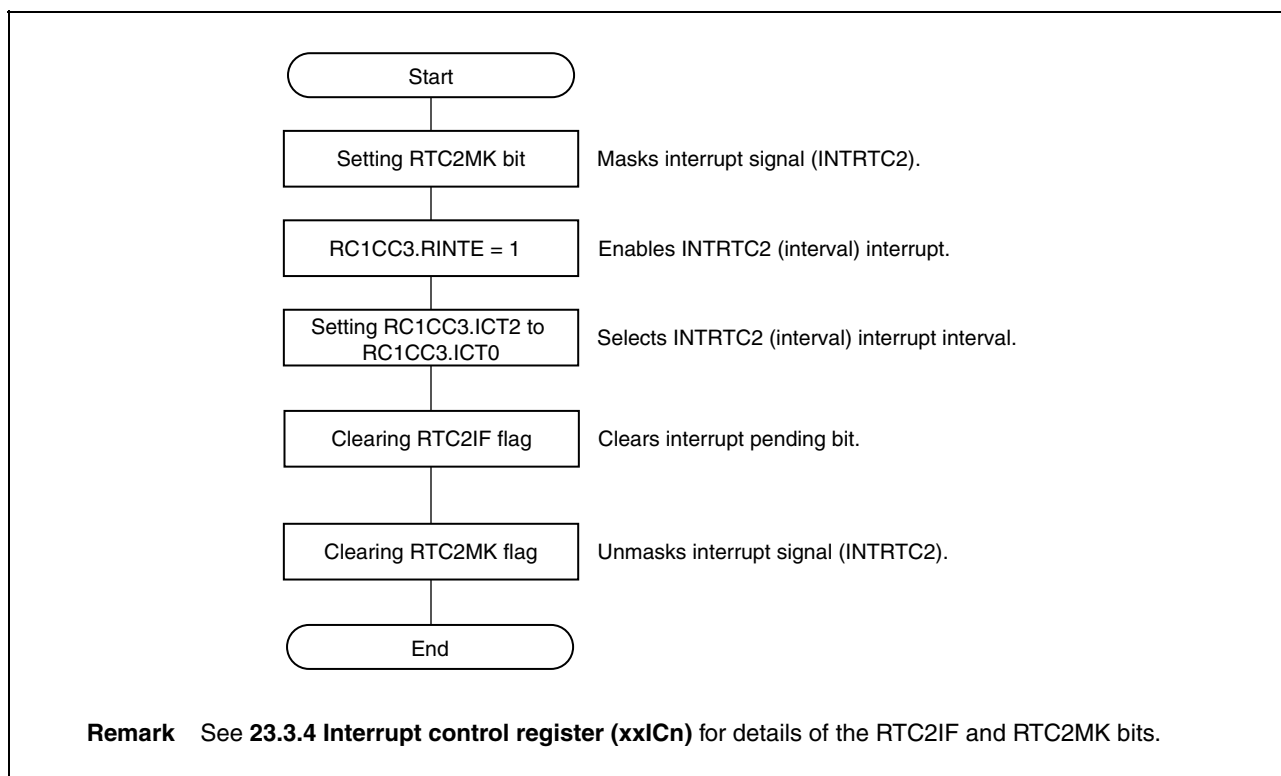
### 12.4.8  Initializing real-time counter

The procedure for initializing the real-time counter is shown below.

**Figure 12-9.  Initializing Real-Time Counter**



**Remarks 1.** See **23.3.4 Interrupt control register (xxICn)** for details of the RTCnIF and RTCnMK bits.

       **2.** n = 0 to 2
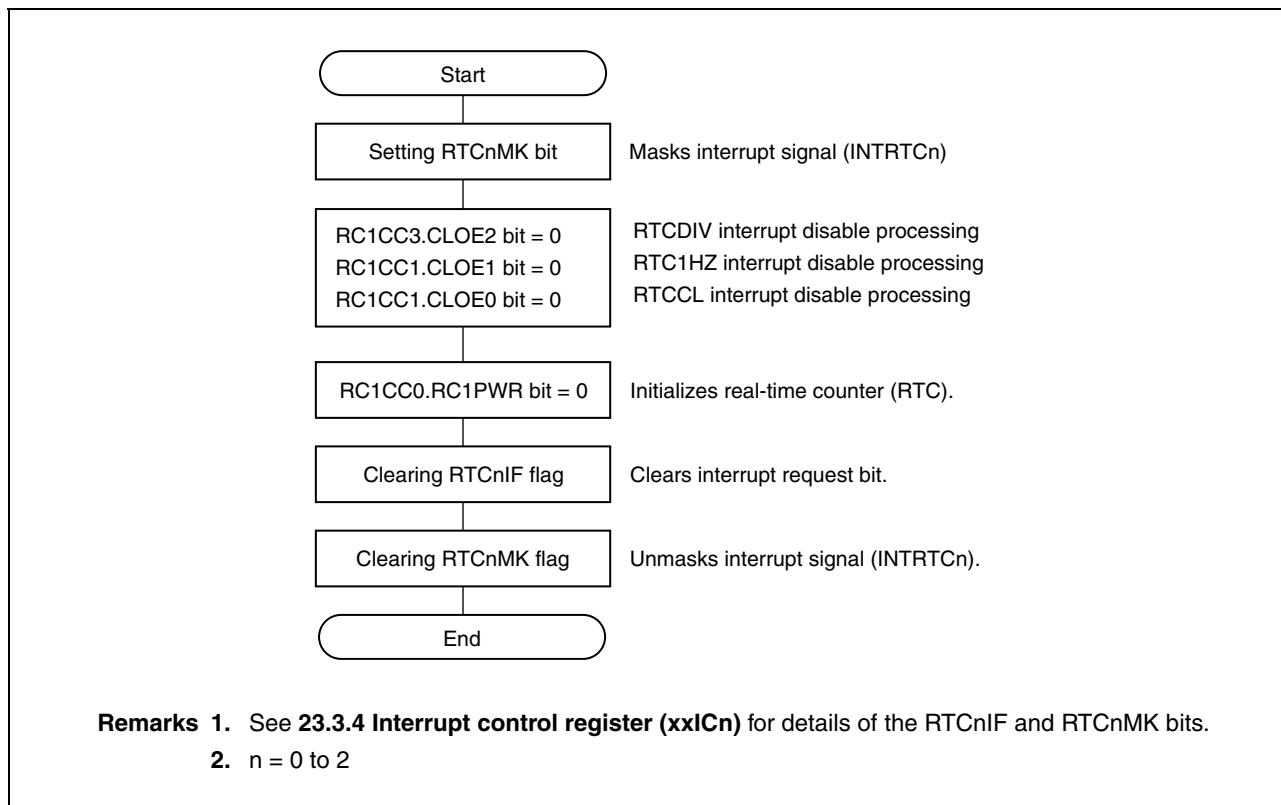
### 12.4.9  Watch error correction example of real-time counter

The watch error correction function corrects deviation in the oscillation frequency of a resonator connected to the V850ES/Jx3-U.

Deviation, here, refers to steady-state deviation, which is deviation in the frequency when the resonator is designed.

Next, the timing chart when an error has occurred in the input clock intended to be 32.768 kHz but a 32.7681 kHz resonator has been connected when designing the system, and the RC1SUBC and RC1SEC count operations to correct the error are shown below.

**Figure 12-10.  Watch Error Correction Example**
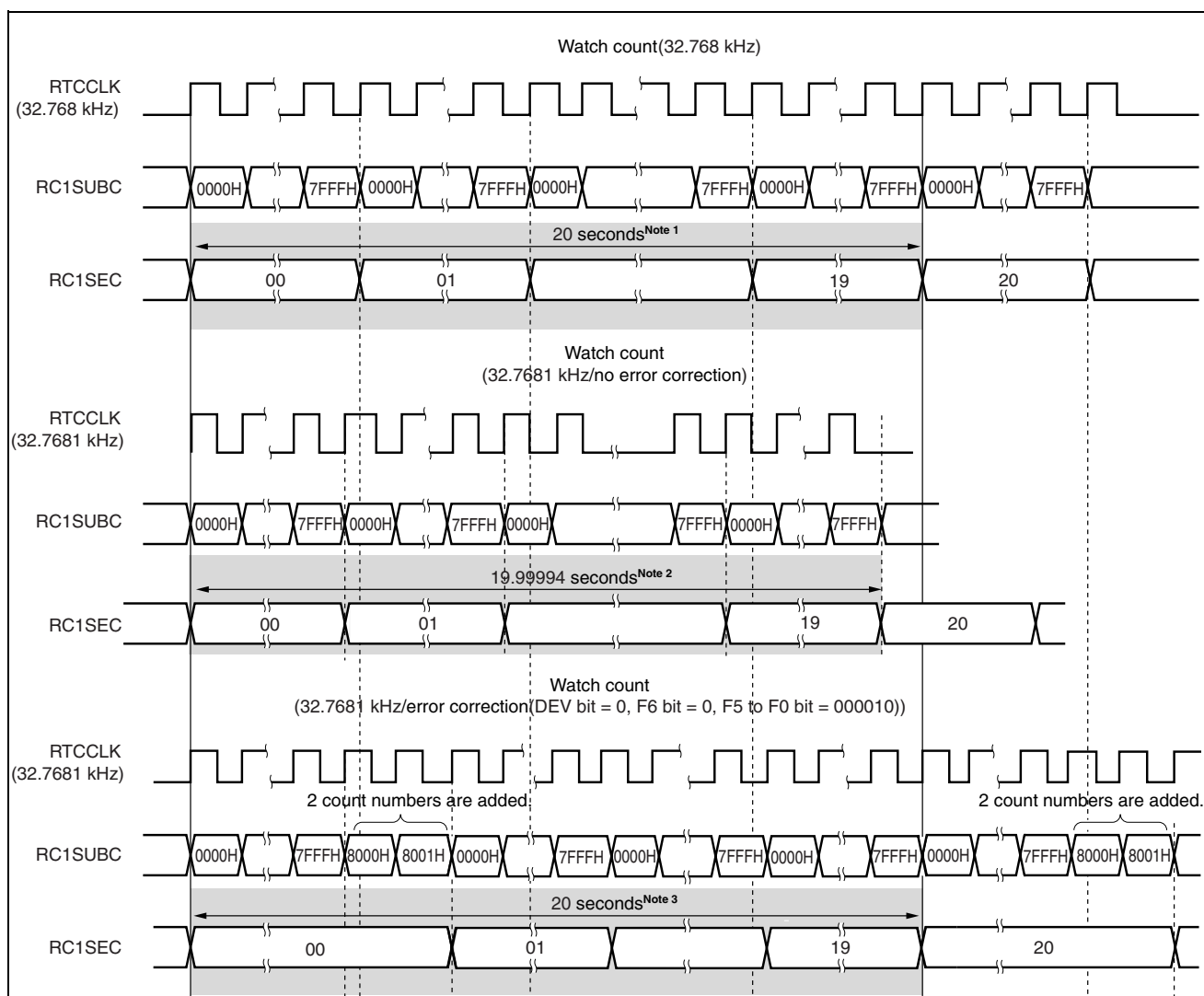


**Notes 1.** The RC1SEC counter counts 20 seconds every 32,768 cycles (0000H to 7FFFH) of the 32.768 kHz clock.

   **2.** When 32,768 cycles (0000H to 7FFFH) of the 32.7681 kHz clock are input, the time counted by the RC1SEC counter is calculated as follows: $32,768/3,268.1 \cong 0.999997$ seconds

   If this counting continues 20 times, the time is calculated as follows: $(32,768/32,768.1) \times 20 \cong 19.99994$ seconds, which causes an error of 0.00006 seconds.

   **3.** To precisely count 20 seconds by using a 32.7681 kHz clock, clear the DEV and F6 bits to 0 and set the F5 to F0 bits to 2H (000010B) in the RC1SUBU register.  As a result, two additional cycles are counted every 20 seconds (when the RC1SEC counter count is 00, 20, and 40 seconds), so that the number of cycles counted at these points is not 32,768, but 32,770 (0000H to 8001H), which is exactly 20 seconds.

As shown in Figure 12-10, the watch can be accurately counted by incrementing the RC1SUBC count value, if a positive error faster than 32.768 kHz occurs at the resonator.  Similarly, if a negative error slower than 32.768 kHz occurs at the resonator, the watch can be accurately counted by decrementing the RC1SUBC count value.

The RC1SUBC correction value is determined by using the RC1SUBU.F6 to RC1SUBU.F0 bits.

The F6 bit is used to determine whether to increment or decrement RC1SUBC and the F5 to F0 bits to determine the RC1SUBC value.

### (1)  Incrementing the RC1SUBC count value

The RC1SUBC count value is incremented by the value set using the F5 to F0 bits, by setting the F6 bit to 0.

Expression for calculating the increment value: (F5 to F0 bit value − 1) × 2

[Example of incrementing the RC1SUBC count value: F6 bit = 0]
    If 15H (010101B) is set to the F5 to F0 bits
      (15H − 1) × 2 = 40 (increments the RC1SUBC count value by 40)
    RC1SUBC count value = 32,768 + 40 = 32,808

### (2)  Decrementing the RC1SUBC count value

The RC1SUBC count value is decremented by an inverted value of the value set using the F5 to F0 bits, by setting the F6 bit to 1.

Expression for calculating the decrement value: (Inverted value of F5 to F0 bit value + 1) × 2

[Example of decrementing the RC1SUBC count value: F6 bit = 1]
    If 15H (010101B) is set to the F5 to F0 bits
      Inverted data of 15H (010101B) = 2AH (101010B)
      (2AH + 1) × 2 = 86 (decrements the RC1SUBC count value by 86)
    RC1SUBC count value = 32,768 − 86 = 32,682

**(3)  DEV bit**

The DEV bit determines when the setting by the F6 to F0 bits is enabled.

The value set by the F6 to F0 bits is reflected upon the next timing, but not to the RC1SUBC count value every time.

**Table 12-6.  DVE Bit Setting**

| DEV Bit Value | Timing of Reflecting Value to RC1SUBC |
|---|---|
| 0 | When RC1SEC is 00, 20, or 40 seconds. |
| 1 | When RC1SEC is 00 seconds. |

[Example when 0010101B is set to F6 to F0 bits]

- If the DEV bit is 0

    The RC1SUBC count value is 32,808 at 00, 20, or 40 seconds.

    Otherwise, it is 32,768.

- IF DEV bit is 1

    The RC1SUBC count value is 32,808 at 00 seconds.

    Otherwise, it is 32,768.

As described above, the RC1SUBC count value is corrected every 20 seconds or 60 seconds, instead of every second, in order to match the RC1SUBC count value with the deviation width of the resonator.

The range in which the resonator frequency can be actually corrected is shown below.

- If the DEV bit is 0:  32.76180000 kHz to 32.77420000 kHz
- If the DEV bit is 1:  32.76593333 kHz to 32.77006667 kHz

The range in which the frequency can be corrected when the DEV bit is 0 is three times wider than when the DEV bit is 1.

However, the accuracy of setting the frequency when the DEV bit is 1 is three times that when the DEV bit is 0.

Tables 12-7 and 12-8 show the setting values of the DEV, and F6 to F0 bits, and the corresponding frequencies that can be corrected.

**Table 12-7.  Range of Frequencies That Can Be Corrected When DEV Bit = 0**

| F6 | F5 to F0 | RC1SUBC Correction Value | Frequency of Connected Clock (Including Steady-State Deviation) |
|----|----------|--------------------------|-----------------------------------------------------------------|
| 0 | 000000 | No correction | – |
| 0 | 000001 | No correction | – |
| 0 | 000010 | Increments RC1SUBC count value by 2 once every 20 seconds | 32.76810000 kHz |
| 0 | 000011 | Increments RC1SUBC count value by 4 once every 20 seconds | 32.76820000 kHz |
| 0 | 000100 | Increments RC1SUBC count value by 6 once every 20 seconds | 32.76830000 kHz |
| | | ⋮ | |
| 0 | 111011 | Increments RC1SUBC count value by 120 once every 20 seconds | 32.77400000 kHz |
| 0 | 111110 | Increments RC1SUBC count value by 122 once every 20 seconds | 32.77410000 kHz |
| 0 | 111111 | Increments RC1SUBC count value by 124 once every 20 seconds | 32.77420000 kHz (upper limit) |
| 1 | 000000 | No correction | – |
| 1 | 000001 | No correction | – |
| 1 | 000010 | Decrements RC1SUBC count value by 124 once every 20 seconds | 32.76180000 kHz (lower limit) |
| 1 | 000011 | Decrements RC1SUBC count value by 122 once every 20 seconds | 32.76190000 kHz |
| 1 | 000100 | Decrements RC1SUBC count value by 120 once every 20 seconds | 32.76200000 kHz |
| | | ⋮ | |
| 1 | 11011 | Decrements RC1SUBC count value by 6 once every 20 seconds | 32.76770000 kHz |
| 1 | 11110 | Decrements RC1SUBC count value by 4 once every 20 seconds | 32.76780000 kHz |
| 1 | 11111 | Decrements RC1SUBC count value by 2 once every 20 seconds | 32.76790000 kHz |

**Table 12-8.  Range of Frequencies That Can Be Corrected When DEV Bit = 1**

| F6 | F5 to F0 | RC1SUBC Correction Value | Frequency of Connected Clock (Including Steady-State Deviation) |
|----|----------|--------------------------|-----------------------------------------------------------------|
| 0 | 000000 | No correction | – |
| 0 | 000001 | No correction | – |
| 0 | 000010 | Increments RC1SUBC count value by 2 once every 60 seconds | 32.76803333 kHz |
| 0 | 000011 | Increments RC1SUBC count value by 4 once every 60 seconds | 32.76806667 kHz |
| 0 | 000100 | Increments RC1SUBC count value by 6 once every 60 seconds | 32.76810000 kHz |
| | | ⋮ | |
| 0 | 111011 | Increments RC1SUBC count value by 120 once every 60 seconds | 32.77000000 kHz |
| 0 | 111110 | Increments RC1SUBC count value by 122 once every 60 seconds | 32.77003333 kHz |
| 0 | 111111 | Increments RC1SUBC count value by 124 once every 60 seconds | 32.77006667 kHz (upper limit) |
| 1 | 000000 | No correction | – |
| 1 | 000001 | No correction | – |
| 1 | 000010 | Decrements RC1SUBC count value by 124 once every 60 seconds | 32.76593333 kHz (lower limit) |
| 1 | 000011 | Decrements RC1SUBC count value by 122 once every 60 seconds | 32.76596667 kHz |
| 1 | 000100 | Decrements RC1SUBC count value by 120 once every 60 seconds | 32.76600000 kHz |
| | | ⋮ | |
| 1 | 11011 | Decrements RC1SUBC count value by 6 once every 60 seconds | 32.76790000 kHz |
| 1 | 11110 | Decrements RC1SUBC count value by 4 once every 60 seconds | 32.76793333 kHz |
| 1 | 11111 | Decrements RC1SUBC count value by 2 once every 60 seconds | 32.76796667 kHz |

## CHAPTER 13  FUNCTIONS OF WATCHDOG TIMER 2

### 13.1  Functions

Watchdog timer 2 has the following functions.

- Default-start watchdog timer[Note 1]
    → Reset mode: Reset operation upon overflow of watchdog timer 2 (generation of WDT2RES signal)
    → Non-maskable interrupt request mode: NMI operation upon overflow of watchdog timer 2 (generation of INTWDT2 signal)[Note 2]
- Input from main clock, internal oscillation clock, and subclock selectable as the source clock

   **Notes 1.**  Watchdog timer 2 automatically starts in the reset mode following reset release.
         When watchdog timer 2 is not used, either stop its operation before reset is executed via this function, or clear watchdog timer 2 once and stop it within the next interval time.
         Also, write to the WDTM2 register for verification purposes once, even if the default settings (reset mode, interval time: $f_R/2^{19}$) do not need to be changed.
   **2.**  For the non-maskable interrupt servicing due to a non-maskable interrupt request signal (INTWDT2), see **23.2.2 (2)  From INTWDT2 signal**.

## 13.2  Configuration

The following shows the block diagram of watchdog timer 2.

**Figure 13-1.  Block Diagram of Watchdog Timer 2**



**Remark**  $f_{XX}$:     Main clock frequency

$f_{XT}$:     Subclock frequency

$f_R$:      Internal oscillation clock frequency

INTWDT2:  Non-maskable interrupt request signal from watchdog timer 2

WDTRES2:  Watchdog timer 2 reset signal
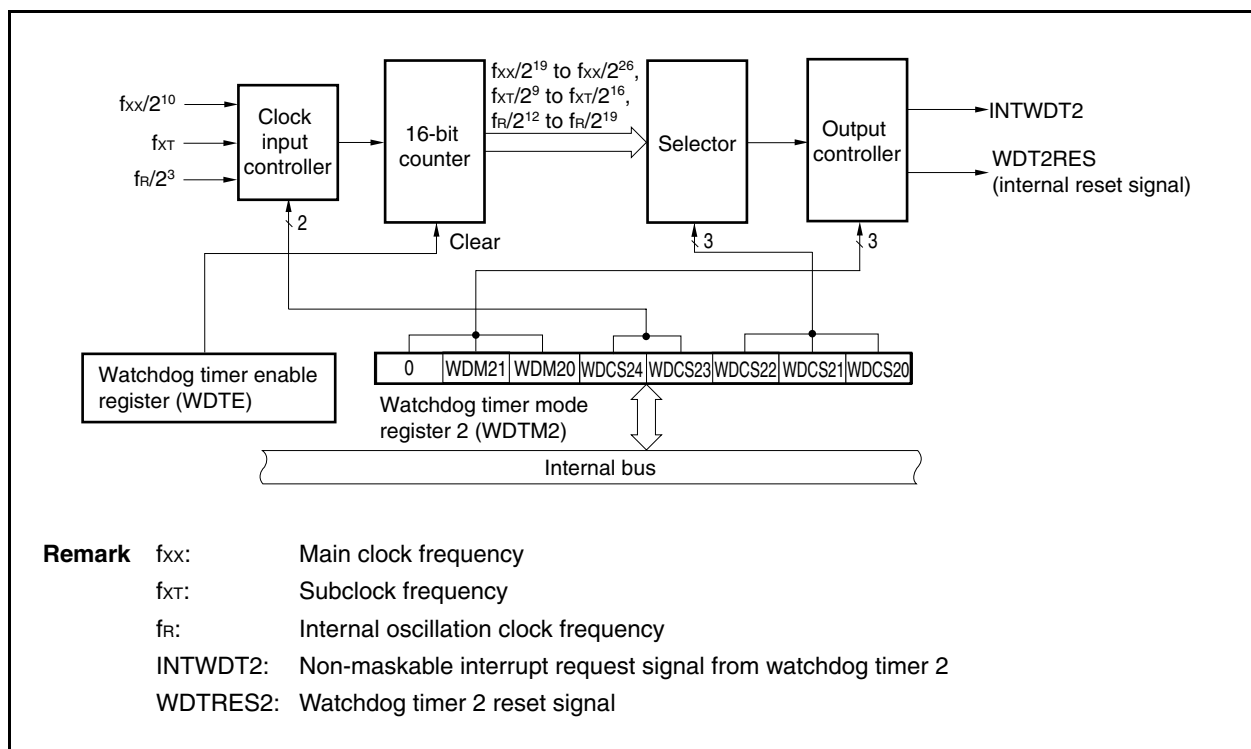
Watchdog timer 2 includes the following hardware.

**Table 13-1.  Configuration of Watchdog Timer 2**

| Item | Configuration |
|---|---|
| Control registers | Watchdog timer mode register 2 (WDTM2) |
| | Watchdog timer enable register (WDTE) |

## 13.3 Registers

**(1) Watchdog timer mode register 2 (WDTM2)**

The WDTM2 register sets the overflow time and operation clock of watchdog timer 2.

This register can be read or written in 8-bit units. This register can be read any number of times, but it can be written only once following reset release.

Reset sets this register to 67H.

**Caution Accessing the WDTM2 register is prohibited in the following statuses. For details, see 3.4.8 (2) Accessing specific on-chip peripheral I/O registers.**

- **When the CPU operates on the subclock and the main clock oscillation is stopped**
- **When the CPU operates on the internal oscillation clock**

After reset: 67H  R/W  Address: FFFFF6D0H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| WDTM2 | 0 | WDM21 | WDM20 | WDCS24 | WDCS23 | WDCS22 | WDCS21 | WDCS20 |

| WDM21 | WDM20 | Selection of operation mode of watchdog timer 2 |
|---|---|---|
| 0 | 0 | Stops operation |
| 0 | 1 | Non-maskable interrupt request mode (generation of INTWDT2 signal) |
| 1 | – | Reset mode (generation of WDT2RES signal) |

**Cautions 1. For details of the WDCS20 to WDCS24 bits, see Table 13-2  Watchdog Timer 2 Clock Selection.**

**2. Although watchdog timer 2 can be stopped just by stopping operation of the internal oscillator, clear the WDTM2 register to 00H to securely stop the timer (to avoid selection of the main clock or subclock due to an erroneous write operation).**

**3. If the WDTM2 register is rewritten twice after reset, an overflow signal is forcibly generated and the counter is reset.**

**4. To intentionally generate an overflow signal, write data to the WDTM2 register twice, or write a value other than "ACH" to the WDTE register once.**

**However, when the operation of watchdog timer 2 is set to be stopped, an overflow signal is not generated even if data is written to the WDTM2 register twice, or a value other than "ACH" is written to the WDTE register once.**

**5. To stop the operation of watchdog timer 2, set the RCM.RSTOP bit to 1 (to stop the internal oscillator) and write 00H in the WDTM2 register. If the RCM.RSTOP bit cannot be set to 1, set the WDCS23 bit to 1 ($2^n/f_{XX}$ is selected and the clock can be stopped in the IDLE1, IDLW2, sub-IDLE, and subclock operation modes).**

**Table 13-2.  Watchdog Timer 2 Clock Selection**

| WDCS24 | WDCS23 | WDCS22 | WDCS21 | WDCS20 | Selected Clock | 100 kHz (MIN.) | 220 kHz (TYP.) | 400 kHz (MAX.) |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | $2^{12}/f_R$ | 41.0 ms | 18.6 ms | 10.2 ms |
| 0 | 0 | 0 | 0 | 1 | $2^{13}/f_R$ | 81.9 ms | 37.2 ms | 20.5 ms |
| 0 | 0 | 0 | 1 | 0 | $2^{14}/f_R$ | 163.8 ms | 74.5 ms | 41.0 ms |
| 0 | 0 | 0 | 1 | 1 | $2^{15}/f_R$ | 327.7 ms | 148.9 ms | 81.9 ms |
| 0 | 0 | 1 | 0 | 0 | $2^{16}/f_R$ | 655.4 ms | 297.9 ms | 163.8 ms |
| 0 | 0 | 1 | 0 | 1 | $2^{17}/f_R$ | 1,310.7 ms | 595.8 ms | 327.7 ms |
| 0 | 0 | 1 | 1 | 0 | $2^{18}/f_R$ | 2,621.4 ms | 1191.6 ms | 655.4 ms |
| 0 | 0 | 1 | 1 | 1 | $2^{19}/f_R$ (Default value) | 5,242.9 ms | 2383.1 ms | 1,310.7 ms |
|  |  |  |  |  |  | $f_{XX}$ = 24 MHz | $f_{XX}$ = 32 MHz | $f_{XX}$ = 48 MHz |
| 0 | 1 | 0 | 0 | 0 | $2^{19}/f_{XX}$ | 21.8 ms | 16.4 ms | 10.9 ms |
| 0 | 1 | 0 | 0 | 1 | $2^{20}/f_{XX}$ | 43.7 ms | 32.8 ms | 21.8 ms |
| 0 | 1 | 0 | 1 | 0 | $2^{21}/f_{XX}$ | 87.4 ms | 65.5 ms | 43.7 ms |
| 0 | 1 | 0 | 1 | 1 | $2^{22}/f_{XX}$ | 174.8 ms | 131.1 ms | 87.4 ms |
| 0 | 1 | 1 | 0 | 0 | $2^{23}/f_{XX}$ | 349.5 ms | 262.1 ms | 174.8 ms |
| 0 | 1 | 1 | 0 | 1 | $2^{24}/f_{XX}$ | 699.1 ms | 524.3 ms | 349.5 ms |
| 0 | 1 | 1 | 1 | 0 | $2^{25}/f_{XX}$ | 1398.1 ms | 1048.6 ms | 699.1 ms |
| 0 | 1 | 1 | 1 | 1 | $2^{26}/f_{XX}$ | 2796.2 ms | 2097.2 ms | 1398.1 ms |
|  |  |  |  |  |  | $f_{XT}$ = 32.768 kHz |  |  |
| 1 | $\times$ | 0 | 0 | 0 | $2^{9}/f_{XT}$ | 15.625 ms |  |  |
| 1 | $\times$ | 0 | 0 | 1 | $2^{10}/f_{XT}$ | 31.25 ms |  |  |
| 1 | $\times$ | 0 | 1 | 0 | $2^{11}/f_{XT}$ | 62.5 ms |  |  |
| 1 | $\times$ | 0 | 1 | 1 | $2^{12}/f_{XT}$ | 125 ms |  |  |
| 1 | $\times$ | 1 | 0 | 0 | $2^{13}/f_{XT}$ | 250 ms |  |  |
| 1 | $\times$ | 1 | 0 | 1 | $2^{14}/f_{XT}$ | 500 ms |  |  |
| 1 | $\times$ | 1 | 1 | 0 | $2^{15}/f_{XT}$ | 1,000 ms |  |  |
| 1 | $\times$ | 1 | 1 | 1 | $2^{16}/f_{XT}$ | 2,000 ms |  |  |

**(2)  Watchdog timer enable register (WDTE)**

The counter of watchdog timer 2 is cleared and counting restarted by writing "ACH" to the WDTE register.

The WDTE register can be read or written in 8-bit units.

Reset sets this register to 9AH.

---

After reset:  9AH      R/W      Address:  FFFFF6D1H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| WDTE | | | | | | | | |

**Cautions 1.  When a value other than "ACH" is written to the WDTE register, an overflow signal is forcibly output.**

**2.  When a 1-bit memory manipulation instruction is executed for the WDTE register, an overflow signal is forcibly output.**

**3.  To intentionally generate an overflow signal, write a value other than "ACH" to the WDTE register once, or write data to the WDTM2 register twice.**

**However, when the operation of watchdog timer 2 is set to be stopped, an overflow signal is not generated even if data is written to the WDTM2 register twice, or a value other than "ACH" is written to the WDTE register once.**

**4.  The read value of the WDTE register is "9AH" (which differs from written value "ACH").**

---

## 13.4  Operation

Watchdog timer 2 automatically starts in the reset mode following reset release.

The WDTM2 register can be written to only once following reset using byte access.  To use watchdog timer 2, write the operation mode and the interval time to the WDTM2 register using an 8-bit memory manipulation instruction.  After this, the operation of watchdog timer 2 cannot be stopped.

The WDTM2.WDCS24 to WDTM2.WDCS20 bits are used to select the watchdog timer 2 loop detection time interval.

Writing ACH to the WDTE register clears the counter of watchdog timer 2 and starts the count operation again.  After the count operation has started, write ACH to WDTE within the loop detection time interval.

If the time interval expires without ACH being written to the WDTE register, a reset signal (WDT2RES) or a non-maskable interrupt request signal (INTWDT2) is generated, depending on the set values of the WDTM2.WDM21 and WDTM2.WDM20 bits.

When the WDTM2.WDM21 bit is set to 1 (reset mode), if a WDT overflow occurs during oscillation stabilization after a reset or standby is released, no internal reset will occur and the CPU clock will switch to the internal oscillation clock.

To not use watchdog timer 2, write 00H to the WDTM2 register.

For the non-maskable interrupt servicing while the non-maskable interrupt request mode is set, see **23.2.2 (2)  From INTWDT2 signal**.

## CHAPTER 14 REAL-TIME OUTPUT FUNCTION (RTO)

### 14.1 Function

The real-time output function transfers the data preset to the RTBL0 and RTBH0 registers to the output latches via hardware and outputs the data to an external device, at the same time as a timer interrupt occurs. The pins through which the data is output to an external device constitute a port called the real-time output (RTO) port.

Because signals without jitter can be output by using RTO, it is suitable for controlling a stepper motor.

In the V850ES/JG3-U and V850ES/JH3-U, one 6-bit real-time output port channel is provided.

The real-time output port can be set to the port mode or real-time output port mode in 1-bit units.

## 14.2  Configuration

The block diagram of RTO is shown below.

**Figure 14-1.  Block Diagram of RTO**
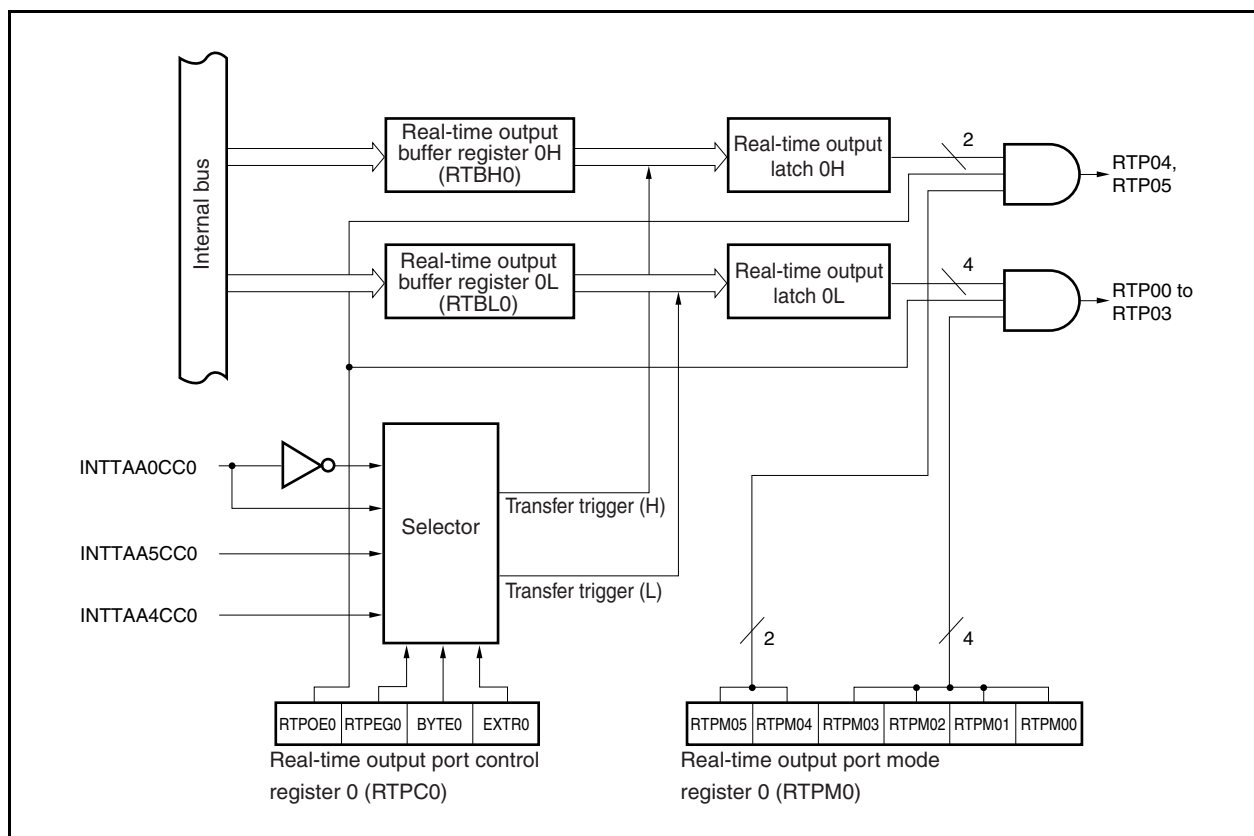


RTO includes the following hardware.

**Table 14-1.  Configuration of RTO**

| Item | Configuration |
|------|---------------|
| Registers | Real-time output buffer registers 0L, 0H (RTBL0, RTBH0) |
| Control registers | Real-time output port mode register 0 (RTPM0)<br>Real-time output port control register 0 (RTPC0) |

**(1) Real-time output buffer registers 0L, 0H (RTBL0, RTBH0)**

The RTBL0 and RTBH0 registers are 4-bit registers that hold output data in advance.

These registers are each mapped to independent addresses in the peripheral I/O register area.

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

If an operation mode of 4 bits × 1 channel or 2 bits × 1 channel is specified (RTPC0.BYTE0 bit = 0), data can be individually set to the RTBL0 and RTBH0 registers. The data of both these registers can be read at once by specifying the address of either of these registers.

If an operation mode of 6 bits × 1 channel is specified (BYTE0 bit = 1), 8-bit data can be set to both the RTBL0 and RTBH0 registers by writing the data to either of these registers. Moreover, the data of both these registers can be read at once by specifying the address of either of these registers.

Table 14-2 shows the operation when the RTBL0 and RTBH0 registers are manipulated.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| After reset: 00H   R/W   Address: RTBL0 FFFFF6E0H, RTBH0 FFFFF6E2H | | | | | | | | | |
| RTBL0 | | | | | | RTBL03 | RTBL02 | RTBL01 | RTBL00 |
| RTBH0 | | 0 | 0 | RTBH05 | RTBH04 | | | | |
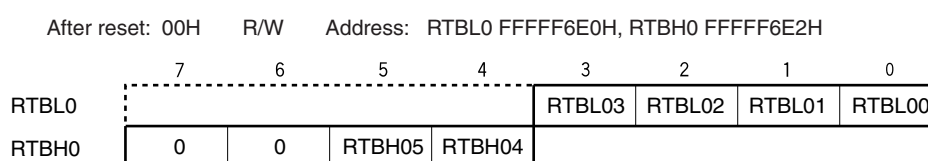
**Cautions**   **1. When writing to bits 6 and 7 of the RTBH0 register, always set 0.**

        **2. Accessing the RTBL0 and RTBH0 registers is prohibited in the following statuses. For details, see 3.4.8 (2) Accessing specific on-chip peripheral I/O registers.**

          • **When the CPU operates on the subclock and the main clock oscillation is stopped**

          • **When the CPU operates on the internal oscillation clock**

**Table 14-2. Operation During Manipulation of RTBL0 and RTBH0 Registers**

| Operation Mode | Register to Be Manipulated | Read | | Write[Note] | |
|---|---|---|---|---|---|
| | | Higher 4 Bits | Lower 4 Bits | Higher 4 Bits | Lower 4 Bits |
| 4 bits × 1 channel, 2 bits × 1 channel | RTBL0 | RTBH0 | RTBL0 | Invalid | RTBL0 |
| | RTBH0 | RTBH0 | RTBL0 | RTBH0 | Invalid |
| 6 bits × 1 channel | RTBL0 | RTBH0 | RTBL0 | RTBH0 | RTBL0 |
| | RTBH0 | RTBH0 | RTBL0 | RTBH0 | RTBL0 |

**Note**   After setting the real-time output port, set output data to the RTBL0 and RTBH0 registers by the time a real-time output trigger is generated.

## 14.3 Registers

RTO is controlled using the following two registers.

- Real-time output port mode register 0 (RTPM0)
- Real-time output port control register 0 (RTPC0)

### (1) Real-time output port mode register 0 (RTPM0)
The RTPM0 register selects the real-time output port mode or port mode in 1-bit units.
This register can be read or written in 8-bit or 1-bit units.
Reset sets this register to 00H.

After reset: 00H    R/W    Address: RTPM0 FFFFF6E4H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RTPM0 | 0 | 0 | RTPM05 | RTPM04 | RTPM03 | RTPM02 | RTPM01 | RTPM00 |

| RTPM0m | Control of real-time output port (m = 0 to 5) |
|---|---|
| 0 | Real-time output disabled |
| 1 | Real-time output enabled |

**Cautions 1. By enabling the real-time output operation (RTPC0.RTPOE0 bit = 1), the bits enabled for real-time output among the RTP00 to RTP05 signals perform real-time output, and the bits set to port mode output 0.**

**2. If real-time output is disabled (RTPOE0 bit = 0), the real-time output pins (RTP00 to RTP05) all output 0, regardless of the RTPM0 register setting.**

**3. In order to use this register for the real-time output pins (RTP00 to RTP05), set these pins as real-time output port pins using the PMC and PFC registers.**

**(2) Real-time output port control register 0 (RTPC0)**

The RTPC0 register is a register that sets the operation mode and output trigger of the real-time output port.

The relationship between the operation mode and output trigger of the real-time output port is as shown in Table 14-3.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

---

After reset: 00H     R/W     Address: FFFFF6E5H

| | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RTPC0 | RTPOE0 | RTPEG0 | BYTE0 | EXTR0 | 0 | 0 | 0 | 0 |

| RTPOE0 | Control of real-time output operation |
|---|---|
| 0 | Disables operation[Note 1] |
| 1 | Enables operation |

| RTPEG0 | Valid edge of INTTP0CC0 signal |
|---|---|
| 0 | Falling edge[Note 2] |
| 1 | Rising edge |

| BYTE0 | Specification of channel configuration for real-time output |
|---|---|
| 0 | 4 bits $\times$ 1 channels, 2 bits $\times$ 1 channels |
| 1 | 6 bits $\times$ 1 channels |

Notes 1. When the real-time output operation is disabled (RTPOE0 bit = 0), all real-time output pins (RTP00 to RTP05) output "0".

      2. The INTTAA0CC0 signal is output for 1 clock of the count clock selected by TAA0.

**Caution   Set the RTPEG0, BYTE0, and EXTR0 bits only when the RTPOE0 bit = 0.**

---

**Table 14-3. Operation Modes and Output Triggers of Real-Time Output Port**

| BYTE0 | EXTR0 | Operation Mode | RTBH0 (RTP04, RTP05) | RTBL0 (RTP00 to RTP03) |
|---|---|---|---|---|
| 0 | 0 | 4 bits $\times$ 1 channel, 2 bits $\times$ 1 channel | INTTAA5CC0 | INTTAA4CC0 |
| | 1 | | INTTAA4CC0 | INTTAA0CC0 |
| 1 | 0 | 6 bits $\times$ 1 channel | INTTAA4CC0 | |
| | 1 | | INTTAA0CC0 | |

## 14.4 Operation

If the real-time output operation is enabled by setting the RTPC0.RTPOE0 bit to 1, the data of the RTBH0 and RTBL0 registers is transferred to the real-time output latch in synchronization with the generation of the selected transfer trigger (set by the RTPC0.EXTR0 and RTPC0.BYTE0 bits). Of the transferred data, only the data of the bits for which real-time output is enabled by the RTPM0 register is output from the RTP00 to RTP05 bits. The bits for which real-time output is disabled by the RTPM0 register output 0.

If the real-time output operation is disabled by clearing the RTPOE0 bit to 0, the RTP00 to RTP05 signals output 0 regardless of the setting of the RTPM0 register.

**Figure 14-2. Example of Operation Timing of RTO0 (When EXTR0 Bit = 0, BYTE0 Bit = 0)**



A: Software processing by INTTAA5CC0 interrupt request (RTBH0 write)

B: Software processing by INTTAA4CC0 interrupt request (RTBL0 write)
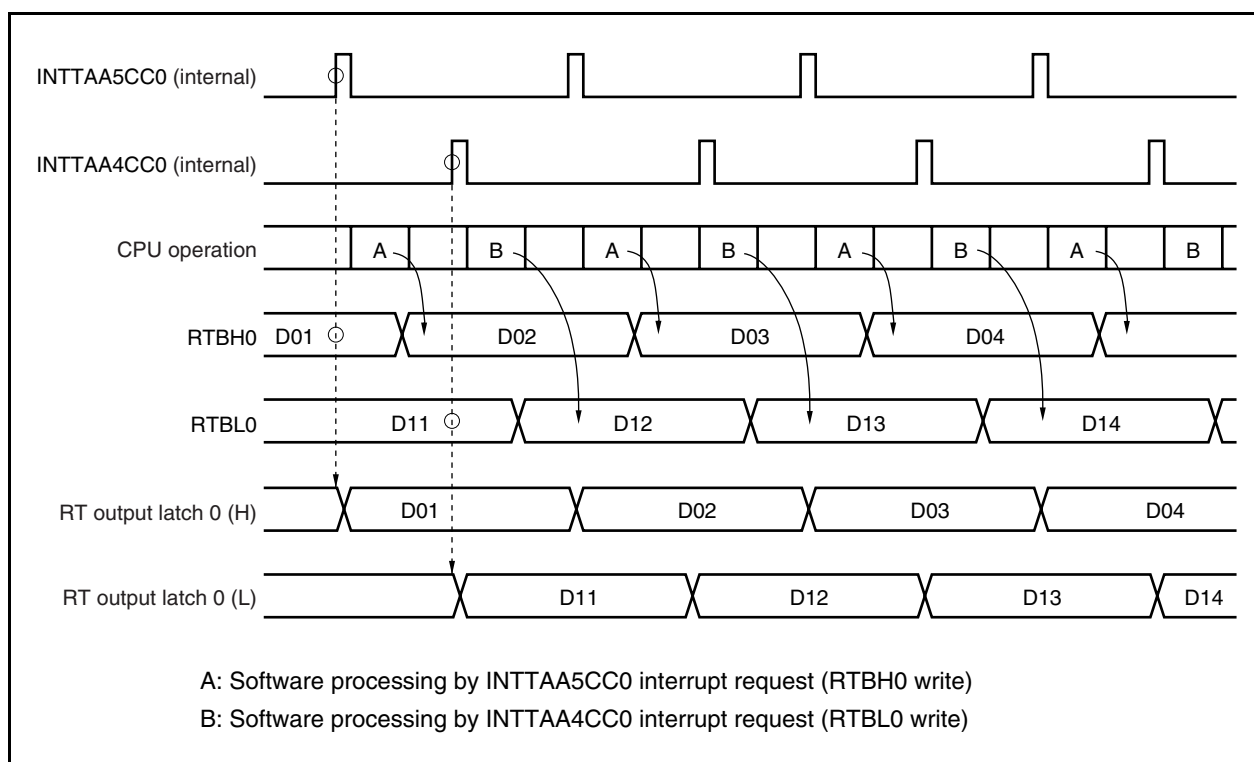
**Remark** For the operation during standby, see **CHAPTER 25 STANDBY FUNCTION**.

## 14.5  Usage

(1)  Disable real-time output.
  Clear the RTPC0.RTPOE0 bit to 0.

(2)  Perform initialization as follows.
  • Set the alternate-function pins of port 2 or port 5
    After setting the PFC2.PFC2m bit and PFCE2.PFCE2m bit to the RTO pin, set the PMC2.PMC2m bit to 1 (m = 0 to 3).
    After setting the PFC5.PFC5m bit and PFCE5.PFCE5m bit to the RTO pin, set the PMC5.PMC5m bit to 1 (m = 0 to 5).
  • Specify the real-time output port mode or port mode in 1-bit units.
    Set the RTPM0 register.
  • Channel configuration: Select the trigger and valid edge.
    Set the RTPC0.EXTR0, RTPC0.BYTE0, and RTPC0.RTPEG0 bits.
  • Set the initial values to the RTBH0 and RTBL0 registers[Note 1].

(3)  Enable real-time output.
  Set the RTPOE0 bit = 1.

(4)  Set the next output value to the RTBH0 and RTBL0 registers by the time the selected transfer trigger is generated[Note 2].

(5)  Sequentially set the next real-time output value to the RTBH0 and RTBL0 registers via interrupt servicing corresponding to the selected trigger.

  **Notes  1.**  If the RTBH0 and RTBL0 registers are written when the RTPOE0 bit = 0, that value is transferred to real-time output latches 0H and 0L.
  **2.**  Even if the RTBH0 and RTBL0 registers are written when the RTPOE0 bit = 1, data is not transferred to real-time output latches 0H and 0L.

## 14.6  Cautions

(1)  Prevent the following conflicts by software.

  • Conflict between real-time output disable/enable switching (RTPOE0 bit) and the selected real-time output trigger.
  • Conflict between writing to the RTBH0 and RTBL0 registers in the real-time output enabled status and the selected real-time output trigger.

(2)  Before performing initialization, disable real-time output (RTPOE0 bit = 0).

(3)  Once real-time output has been disabled (RTPOE0 bit = 0), be sure to initialize the RTBH0 and RTBL0 registers before enabling real-time output again (RTPOE0 bit = 0 → 1).

## CHAPTER 15  A/D CONVERTER

### 15.1  Overview

The A/D converter converts analog input signals into digital values, has a resolution of 10 bits, and can handle 12 analog input signal channels (ANI0 to ANI11).

The A/D converter has the following features.

○ 10-bit resolution
○ 12 channels
○ Successive approximation method
○ Operating voltage: $AV_{REF0}$ = 3.0 to 3.6 V
○ Analog input voltage: 0 V to $AV_{REF0}$
○ The following functions are provided as operation modes.
  • Continuous select mode
  • Continuous scan mode
  • One-shot select mode
  • One-shot scan mode
○ The following functions are provided as trigger modes.
  • Software trigger mode
  • External trigger mode (external, 1)
  • Timer trigger mode
○ Power-fail monitor function (conversion result compare function)

### 15.2  Functions

**(1)  10-bit resolution A/D conversion**

An analog input channel is selected from ANI0 to ANI11, and an A/D conversion operation is repeated at a resolution of 10 bits.  Each time A/D conversion has been completed, an interrupt request signal (INTAD) is generated.

**(2)  Power-fail detection function**

This function is used to detect a drop in the battery voltage.  The result of A/D conversion (the value of the ADA0CRnH register) is compared with the value of the ADA0PFT register, and the INTAD signal is generated only when a specified comparison condition is satisfied (n = 0 to 11).

## 15.3  Configuration

The block diagram of the A/D converter is shown below.

**Figure 15-1.  Block Diagram of A/D Converter**



**Note** The timer trigger signal from 6-phase PWM output circuit (TMQOP)
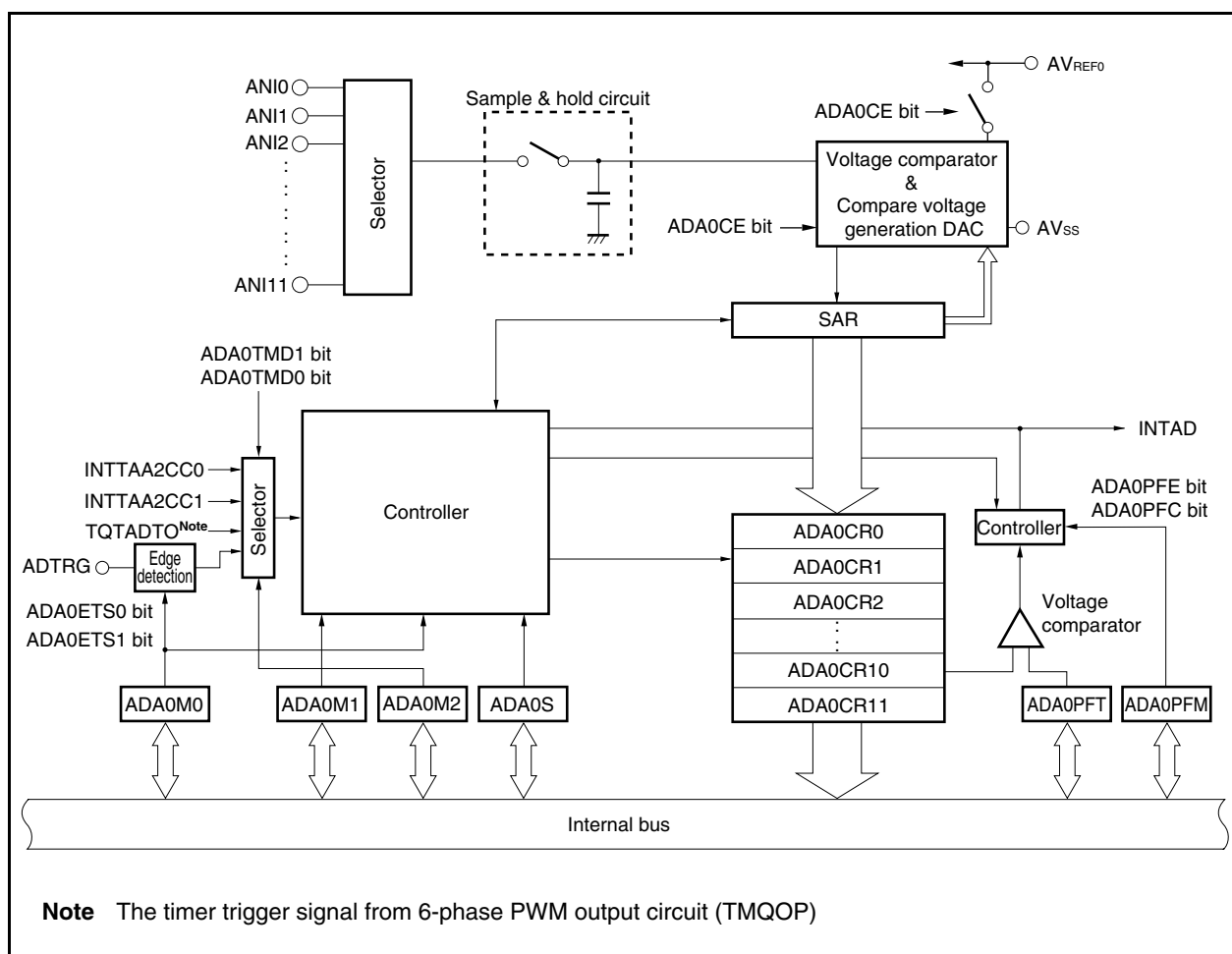
The A/D converter includes the following hardware.

**Table 15-1.  Configuration of A/D Converter**

| Item | Configuration |
|---|---|
| Analog inputs | 12 channels (ANI0 to ANI11 pins) |
| Registers | Successive approximation register (SAR)<br>A/D conversion result registers 0 to 11 (ADA0CR0 to ADA0CR11)<br>A/D conversion result registers 0H to 11H (ADCR0H to ADCR11H):  Only higher 8 bits can be read |
| Control registers | A/D converter mode registers 0 to 2 (ADA0M0 to ADA0M2)<br>A/D converter channel specification register 0 (ADA0S)<br>Power fail compare mode register (ADA0PFM)<br>Power fail compare threshold value register (ADA0PFT) |

**(1) Successive approximation register (SAR)**

The SAR compares the voltage value of the analog input signal with the output voltage of the compare voltage generation DAC (compare voltage), and holds the comparison result starting from the most significant bit (MSB). When the comparison result has been held down to the least significant bit (LSB) (i.e., when A/D conversion is complete), the contents of the SAR are transferred to the ADA0CRn register.

**Remark**   n = 0 to 11

**(2) A/D conversion result register n (ADA0CRn), A/D conversion result register nH (ADA0CRnH)**

The ADA0CRn register is a 16-bit register that stores the A/D conversion result. ADA0ARn consist of 12 registers and the A/D conversion result is stored in the 10 higher bits of the AD0CRn register corresponding to analog input. (The lower 6 bits are fixed to 0.)

**(3) A/D converter mode register 0 (ADA0M0)**

This register specifies the operation mode and controls the conversion operation by the A/D converter.

**(4) A/D converter mode register 1 (ADA0M1)**

This register sets the conversion time of the analog input signal to be converted.

**(5) A/D converter mode register 2 (ADA0M2)**

This register sets the hardware trigger mode.

**(6) A/D converter channel specification register (ADA0S)**

This register sets the input port that inputs the analog voltage to be converted.

**(7) Power-fail compare mode register (ADA0PFM)**

This register sets the power-fail monitor mode.

**(8) Power-fail compare threshold value register (ADA0PFT)**

The ADA0PFT register sets the threshold value that is compared with the value of A/D conversion result register nH (ADA0CRnH).

The 8-bit data set to the ADA0PFT register is compared with the higher 8 bits of the A/D conversion result register (ADA0CRnH).

**(9) Controller**

The controller compares the result of the A/D conversion (the value of the ADA0CRnH register) with the value of the ADA0PFT register when A/D conversion is completed or when the power-fail detection function is used, and generates the INTAD signal only when a specified comparison condition is satisfied.

**(10) Sample & hold circuit**

The sample & hold circuit samples each of the analog input signals selected by the input circuit and sends the sampled data to the voltage comparator. This circuit also holds the sampled analog input signal voltage during A/D conversion.

**(11) Voltage comparator**

The voltage comparator compares a voltage value that has been sampled and held with the output voltage of the compare voltage generation DAC.

**(12)  Compare voltage generation DAC**

This compare voltage generation DAC is connected between $AV_{REF0}$ and $AV_{SS}$ and generates a voltage for comparison with the analog input signal.

**(13)  ANI0 to ANI11 pins**

These are analog input pins for the 12 A/D converter channels and are used to input analog signals to be converted into digital signals.  Pins other than the one selected as the analog input by the ADA0S register can be used as input port pins.

**Caution   Make sure that the voltages input to the ANI0 to ANI11 pins do not exceed the rated values.  In particular if a voltage of $AV_{REF0}$ or higher is input to a channel, the conversion value of that channel becomes undefined, and the conversion values of the other channels may also be affected.**

**(14)  $AV_{REF0}$ pin**

This is the pin used to input the reference voltage of the A/D converter.  Always make the potential at this pin the same as that at the $V_{DD}$ pin even when the A/D converter is not used.

The signals input to the ANI0 to ANI11 pins are converted to digital signals based on the voltage applied between the $AV_{REF0}$ and $AV_{SS}$ pins.

**(15)  $AV_{SS}$ pin**

This is the ground potential pin of the A/D converter.  Always make the potential at this pin the same as that at the $V_{SS}$ pin even when the A/D converter is not used.

## 15.4  Registers

The A/D converter is controlled by the following registers.

- A/D converter mode registers 0, 1, 2 (ADA0M0, ADA0M1, ADA0M2)
- A/D converter channel specification register 0 (ADA0S)
- Power-fail compare mode register (ADA0PFM)

The following registers are also used.

- A/D conversion result register n (ADA0CRn)
- A/D conversion result register nH (ADA0CRnH)
- Power-fail compare threshold value register (ADA0PFT)

### (1)  A/D converter mode register 0 (ADA0M0)

The ADA0M0 register is an 8-bit register that specifies the operation mode and controls conversion operations.
This register can be read or written in 8-bit or 1-bit units.  However, the ADA0EF bit is read-only.
Reset sets this register to 00H.

(1/2)

After reset:  00H     R/W     Address:  FFFFF200H

| | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| ADA0M0 | ADA0CE | 0 | ADA0MD1 | ADA0MD0 | ADA0ETS1 | ADA0ETS0 | ADA0TMD | ADA0EF |

| ADA0CE | A/D conversion control |
|---|---|
| 0 | Stops A/D conversion |
| 1 | Enables A/D conversion |

| ADA0MD1 | ADA0MD0 | Specification of A/D converter operation mode |
|---|---|---|
| 0 | 0 | Continuous select mode |
| 0 | 1 | Continuous scan mode |
| 1 | 0 | One-shot select mode |
| 1 | 1 | One-shot scan mode |

| ADA0ETS1 | ADA0ETS0 | Specification of external trigger (ADTRG pin) input valid edge |
|---|---|---|
| 0 | 0 | No edge detection |
| 0 | 1 | Falling edge detection |
| 1 | 0 | Rising edge detection |
| 1 | 1 | Detection of both rising and falling edges |

| ADA0TMD | Trigger mode specification |
|---------|---------------------------|
| 0 | Software trigger mode |
| 1 | External trigger mode/timer trigger mode |

| ADA0EF | A/D converter status display |
|--------|------------------------------|
| 0 | A/D conversion stopped |
| 1 | A/D conversion in progress |

**Cautions 1.** Accessing the ADA0M0 register is prohibited in the following statuses.  For details, see 3.4.8 (2)  Accessing specific on-chip peripheral I/O registers.

- **When the CPU operates on the subclock and the main clock oscillation is stopped**
- **When the CPU operates on the internal oscillation clock**

**2.** A write operation to bit 0 is ignored.

**3.** Changing the ADA0M1.ADA0FR2 to ADA0M1.ADA0FR0 bits is prohibited while A/D conversion is enabled (ADA0CE bit = 1).

**4.** In the following modes, write data to the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT registers while A/D conversion is stopped (ADA0CE bit = 0), and then enable the A/D conversion operation (ADA0CE bit = 1).

- **Normal conversion mode**
- **One-shot select mode/one-shot scan mode in high-speed conversion mode**

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, and ADA0PFT registers are written in other modes during A/D conversion (ADA0EF bit = 1), the following will be performed according to the mode.

- **In software trigger mode**

   A/D conversion is stopped and started again from the beginning.

- **In hardware trigger mode**

   A/D conversion is stopped, and the trigger standby status is set.

**5.** To select the external trigger mode/timer trigger mode (ADA0TMD bit = 1), set the high-speed conversion mode (ADA0M1.ADA0HS1 bit = 1).  Do not input a trigger during stabilization time that is inserted once after the A/D conversion operation is enabled (ADA0CE bit = 1).

**6.** When not using the A/D converter, stop the operation by setting the ADA0CE bit to 0 to reduce the power consumption.

**(2) A/D converter mode register 1 (ADA0M1)**

The ADA0M1 register is an 8-bit register that specifies the conversion time.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this bit to 00H.

After reset:  00H     R/W     Address:  FFFFF201H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADA0M1 | ADA0HS1 | 0 | 0 | 0 | ADA0FR3 | ADA0FR2 | ADA0FR1 | ADA0FR0 |

| ADA0HS1 | Specification of normal conversion mode/high-speed mode (A/D conversion time) |
|---|---|
| 0 | Normal conversion mode |
| 1 | High-speed conversion mode |

**Cautions 1. Changing the ADA0M1 register is prohibited while A/D conversion is enabled (ADA0M0.ADA0CE bit = 1).**

**2. To select the external trigger mode/timer trigger mode (ADA0M0.ADA0TMD bit = 1), set the high-speed conversion mode (ADA0HS1 bit = 1). Do not input a trigger during the stabilization time that is inserted once after the A/D conversion operation is enabled (ADA0CE bit = 1).**

**3. Be sure to clear bits 6 to 4 to "0".**

**Remark**   For A/D conversion time setting examples, see **Tables 15-2** and **15-3**.

**Table 15-2.  Conversion Time Selection in Normal Conversion Mode (ADA0HS1 Bit = 0)**

| ADA0FR3 to ADA0FR0 Bits | A/D Conversion Time | | | |
|---|---|---|---|---|
| | Stabilization Time + Conversion Time + Wait Time | 48 MHz | 32 MHz | 24 MHz |
| 0000 | $26/f_{xx} + 52/f_{xx} + 54/f_{xx}$ | Setting prohibited | Setting prohibited | 5.50 $\mu$s |
| 0001 | $52/f_{xx} + 104/f_{xx} + 106/f_{xx}$ | 5.46 $\mu$s | 8.19 $\mu$s | Setting prohibited |
| 0010 | $78/f_{xx} + 156/f_{xx} + 158/f_{xx}$ | 8.17 $\mu$s | Setting prohibited | Setting prohibited |
| 0011 | $100/f_{xx} + 208/f_{xx} + 210/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 0100 | $100/f_{xx} + 260/f_{xx} + 262/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 0101 | $100/f_{xx} + 312/f_{xx} + 314/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 0110 | $100/f_{xx} + 364/f_{xx} + 366/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 0111 | $100/f_{xx} + 416/f_{xx} + 418/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 1000 | $100/f_{xx} + 468/f_{xx} + 470/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 1001 | $100/f_{xx} + 520/f_{xx} + 522/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 1010 | $100/f_{xx} + 572/f_{xx} + 574/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 1011 | $100/f_{xx} + 624/f_{xx} + 626/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 1100 | $100/f_{xx} + 676/f_{xx} + 678/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 1101 | $100/f_{xx} + 728/f_{xx} + 730/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 1110 | $100/f_{xx} + 780/f_{xx} + 782/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| 1111 | $100/f_{xx} + 832/f_{xx} + 834/f_{xx}$ | Setting prohibited | Setting prohibited | Setting prohibited |
| Other than above | Setting prohibited | | | |

**Remark**   Stabilization time:   A/D converter setup time (1 $\mu$s or longer)

Conversion time:   Actual A/D conversion time (2.17 to 9.75 $\mu$s)

Wait time:   Wait time inserted before the next conversion

$f_{xx}$:   Main clock frequency

In the normal conversion mode, the conversion is started after the stabilization time elapses after the ADA0M0.ADA0CE bit is set to 1, and A/D conversion is performed only during the conversion time (2.17 to 9.75 $\mu$s).  Operation is stopped after the conversion ends and the A/D conversion end interrupt request signal (INTAD) is generated after the wait time elapses.

Because the conversion operation is stopped during the wait time, operating current can be reduced.

**Cautions 1.   Set as 2.17 $\mu$s $\leq$ conversion time $\leq$ 9.75 $\mu$s.**

**2.   During A/D conversion, if the ADA0M0, ADA0M2, ADA0S, ADA0PFM, and ADA0PFT registers are written or a trigger is input, reconversion is carried out.  However, if the stabilization time end timing conflicts with writing to these registers, or if the stabilization time end timing conflicts with the trigger input, a stabilization time of 64 clocks is reinserted.**

**If a conflict occurs again with the reinserted stabilization time end timing, the stabilization time is reinserted.  Therefore do not set the trigger input interval and control register write interval to 64 clocks or lower.**

**Table 15-3. Conversion Time Selection in High-Speed Conversion Mode (ADA0HS1 Bit = 1)**

| ADA0FR3 to ADA0FR0 Bits | A/D Conversion Time | | | |
| --- | --- | --- | --- | --- |
| | Conversion Time (+ Stabilization Time) | 48 MHz | 32 MHz | 24 MHz |
| 0000 | 52/f$_{xx}$ (+26/f$_{xx}$) | Setting prohibited | Setting prohibited | 2.17 $\mu$s |
| 0001 | 104/f$_{xx}$ (+52/f$_{xx}$) | 2.17 $\mu$s | 3.25 $\mu$s | 4.33 $\mu$s |
| 0010 | 156/f$_{xx}$ (+78/f$_{xx}$) | 3.25 $\mu$s | 4.88 $\mu$s | 6.50 $\mu$s |
| 0011 | 208/f$_{xx}$ (+100/f$_{xx}$) | 4.33 $\mu$s | 6.50 $\mu$s | 8.67 $\mu$s |
| 0100 | 260/f$_{xx}$ (+100/f$_{xx}$) | 5.42 $\mu$s | 8.13 $\mu$s | Setting prohibited |
| 0101 | 312/f$_{xx}$ (+100/f$_{xx}$) | 6.50 $\mu$s | 9.75 $\mu$s | Setting prohibited |
| 0110 | 364/f$_{xx}$ (+100/f$_{xx}$) | 7.58 $\mu$s | Setting prohibited | Setting prohibited |
| 0111 | 416/f$_{xx}$ (+100/f$_{xx}$) | 8.67 $\mu$s | Setting prohibited | Setting prohibited |
| 1000 | 468/f$_{xx}$ (+100/f$_{xx}$) | 9.75 $\mu$s | Setting prohibited | Setting prohibited |
| 1001 | 520/f$_{xx}$ (+100/f$_{xx}$) | Setting prohibited | Setting prohibited | Setting prohibited |
| 1010 | 572/f$_{xx}$ (+100/f$_{xx}$) | Setting prohibited | Setting prohibited | Setting prohibited |
| 1011 | 624/f$_{xx}$ (+100/f$_{xx}$) | Setting prohibited | Setting prohibited | Setting prohibited |
| 1100 | 676/f$_{xx}$ (+100/f$_{xx}$) | Setting prohibited | Setting prohibited | Setting prohibited |
| 1101 | 728/f$_{xx}$ (+100/f$_{xx}$) | Setting prohibited | Setting prohibited | Setting prohibited |
| 1110 | 780/f$_{xx}$ (+100/f$_{xx}$) | Setting prohibited | Setting prohibited | Setting prohibited |
| 1111 | 832/f$_{xx}$ (+100/f$_{xx}$) | Setting prohibited | Setting prohibited | Setting prohibited |
| Other than above | Setting prohibited | | | |

**Remark** Stabilization time:     A/D converter setup time (1 $\mu$s or longer)

Conversion time:     Actual A/D conversion time (2.17 to 9.75 $\mu$s)

f$_{xx}$:     Main clock frequency

In the high-speed conversion mode, the conversion is started after the stabilization time elapses after the ADA0M0.ADA0CE bit is set to 1, and A/D conversion is performed only during the conversion time (2.17 to 9.75 $\mu$s). The A/D conversion end interrupt request signal (INTAD) is generated immediately after the conversion ends.

In continuous conversion mode, the stabilization time is inserted only before the first conversion, and not inserted after the second conversion (the A/D converter remains running).

**Cautions 1.   Set as 2.17 $\mu$s ≤ conversion time ≤ 9.75 $\mu$s.**

**2. In the high-speed conversion mode, rewriting of the ADA0M0, ADA0M2, ADA0S, ADA0PFM, and ADA0PFT registers and trigger input are prohibited during the stabilization time.**

**(3) A/D converter mode register 2 (ADA0M2)**

The ADA0M2 register specifies the hardware trigger mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address: FFFFF203H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADA0M2 | 0 | 0 | 0 | 0 | 0 | 0 | ADA0TMD1 | ADA0TMD0 |

| ADA0TMD1 | ADA0TMD0 | Specification of hardware trigger mode |
|---|---|---|
| 0 | 0 | External trigger mode (when ADTRG pin valid edge is detected) |
| 0 | 1 | Timer trigger mode 0 (when INTTAA2CC0 interrupt request is generated) |
| 1 | 0 | Timer trigger mode 1 (when INTTAA2CC1 interrupt request is generated) |
| 1 | 1 | Timer trigger mode 2 (TQTADT0 signal) |

**Cautions 1. In the following modes, write data to the ADA0M2 register while A/D conversion is stopped (ADA0M0.ADA0CE bit = 0), and then enable the A/D conversion operation (ADA0CE bit = 1).**
   - **Normal conversion mode**
   - **One-shot select mode/one-shot scan mode in high-speed conversion mode**

**2. Be sure to clear bits 7 to 2 to "0".**

**(4) Analog input channel specification register 0 (ADA0S)**

The ADA0S register specifies the pin that inputs the analog voltage to be converted into a digital signal.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address: FFFFF202H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADA0S | 0 | 0 | 0 | 0 | ADA0S3 | ADA0S2 | ADA0S1 | ADA0S0 |

| ADA0S3 | ADA0S2 | ADA0S1 | ADA0S0 | Select mode | Scan mode |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | ANI0 | ANI0 |
| 0 | 0 | 0 | 1 | ANI1 | ANI0, ANI1 |
| 0 | 0 | 1 | 0 | ANI2 | ANI0 to ANI2 |
| 0 | 0 | 1 | 1 | ANI3 | ANI0 to ANI3 |
| 0 | 1 | 0 | 0 | ANI4 | ANI0 to ANI4 |
| 0 | 1 | 0 | 1 | ANI5 | ANI0 to ANI5 |
| 0 | 1 | 1 | 0 | ANI6 | ANI0 to ANI6 |
| 0 | 1 | 1 | 1 | ANI7 | ANI0 to ANI7 |
| 1 | 0 | 0 | 0 | ANI8 | ANI0 to ANI8 |
| 1 | 0 | 0 | 1 | ANI9 | ANI0 to ANI9 |
| 1 | 0 | 1 | 0 | ANI10 | ANI0 to ANI10 |
| 1 | 0 | 1 | 1 | ANI11 | ANI0 to ANI11 |

**Cautions 1. In the following modes, write data to the ADA0S register while A/D conversion is stopped (ADA0M0.ADA0CE bit = 0), and then enable the A/D conversion operation (ADA0CE bit = 1).**

- **Normal conversion mode**
- **One-shot select mode/one-shot scan mode in high-speed conversion mode**

**2. Be sure to clear bits 7 to 4 to "0".**

**(5) A/D conversion result registers n, nH (ADA0CRn, ADA0CRnH)**

The ADA0CRn and ADA0CRnH registers store the A/D conversion results.

These registers are read-only, in 16-bit or 8-bit units.  However, the ADA0CRn register is used for 16-bit access and the ADA0CRnH register for 8-bit access.  The 10 bits of the conversion result are read to the higher 10 bits of the ADA0CRn register, and 0 is read to the lower 6 bits.  The higher 8 bits of the conversion result are read to the ADA0CRnH register.

**Caution   Accessing the ADA0CRn and ADA0CRnH registers is prohibited in the following statuses.**

**For details, see 3.4.8 (2)  Accessing specific on-chip peripheral I/O registers.**

**• When the CPU operates on the subclock and the main clock oscillation is stopped**

**• When the CPU operates on the internal oscillation clock**

---

After reset: Undefined        R        Address:  ADA0CR0 FFFFF210H, ADA0CR1 FFFFF212H,
                                                 ADA0CR2 FFFFF214H, ADA0CR3 FFFFF216H,
                                                 ADA0CR4 FFFFF218H, ADA0CR5 FFFFF21AH,
                                                 ADA0CR6 FFFFF21CH, ADA0CR7 FFFFF21EH,
                                                 ADA0CR8 FFFFF220H, ADA0CR9 FFFFF222H,
                                                 ADA0CR10 FFFFF224H, ADA0CR11 FFFFF226H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADA0CRn (n = 0 to 11) | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | 0 | 0 | 0 | 0 | 0 | 0 |

After reset: Undefined        R        Address:  ADA0CR0H FFFFF211H, ADA0CR1H FFFFF213H,
                                                 ADA0CR2H FFFFF215H, ADA0CR3H FFFFF217H,
                                                 ADA0CR4H FFFFF219H, ADA0CR5H FFFFF21BH,
                                                 ADA0CR6H FFFFF21DH, ADA0CR7H FFFFF21FH,
                                                 ADA0CR8H FFFFF221H, ADA0CR9H FFFFF223H,
                                                 ADA0CR10H FFFFF225H, ADA0CR11H FFFFF227H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADA0CRnH (n = 0 to 11) | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 |

**Caution   A write operation to the ADA0M0 and ADA0S registers may cause the contents of the ADA0CRn register to become undefined.  After the conversion, read the conversion result before writing to the ADA0M0 and ADA0S registers.  Correct conversion results may not be read at a timing other than the above.**

---

The relationship between the analog voltage input to the analog input pins (ANI0 to ANI11) and the A/D conversion result (ADA0CRn register) is as follows.

$$SAR = INT \left( \frac{V_{IN}}{AV_{REF0}} \times 1{,}024 + 0.5 \right)$$

$$ADA0CR^{Note} = SAR \times 64$$

Or,

$$(SAR - 0.5) \times \frac{AV_{REF0}}{1{,}024} \leq V_{IN} < (SAR + 0.5) \times \frac{AV_{REF0}}{1{,}024}$$

INT( ):      Function that returns the integer of the value in ( )

$V_{IN}$:      Analog input voltage

$AV_{REF0}$:      $AV_{REF0}$ pin voltage

ADA0CR:      Value of ADA0CRn register

**Note**    The lower 6 bits of the ADA0CRn register are fixed to 0.

The following shows the relationship between the analog input voltage and the A/D conversion results.

**Figure 15-2.  Relationship Between Analog Input Voltage and A/D Conversion Results**
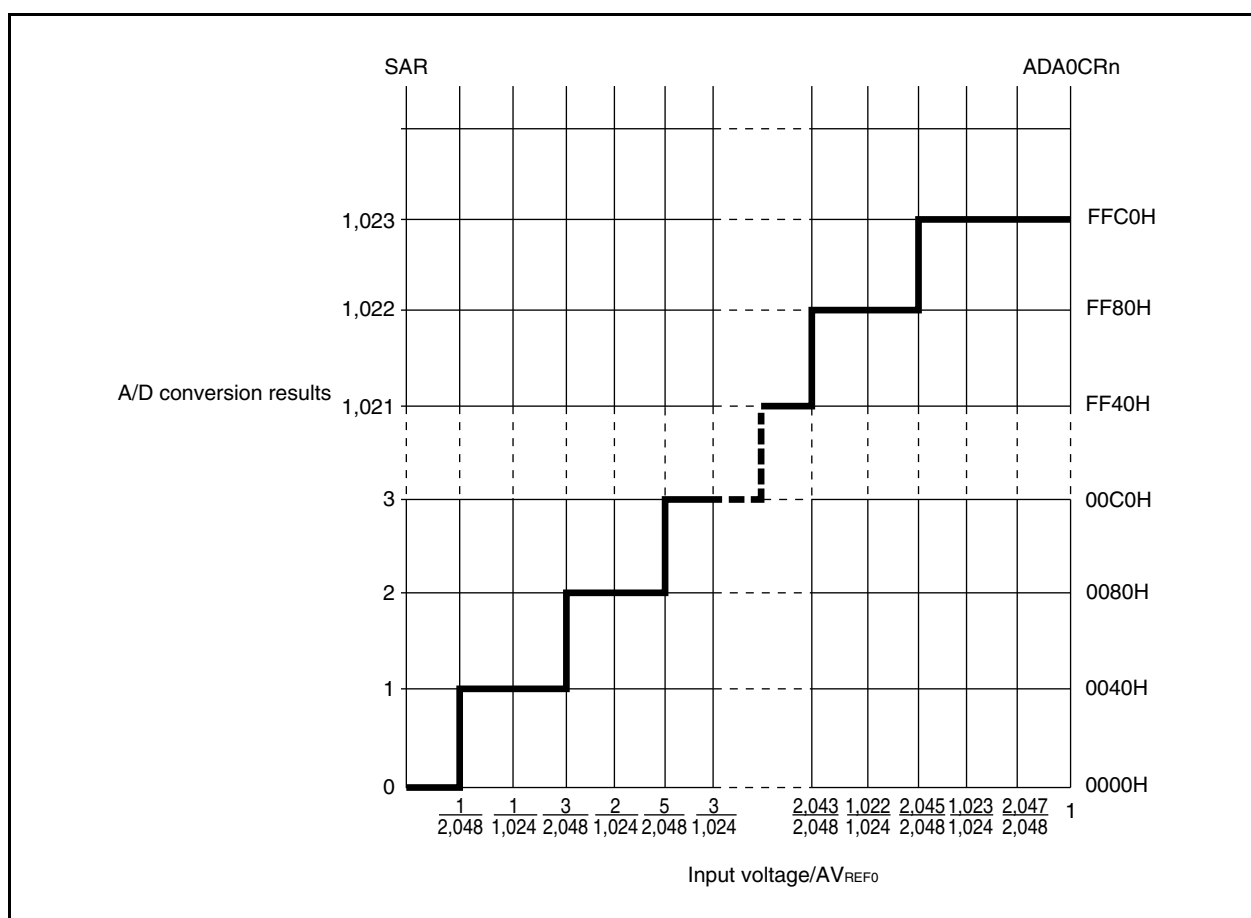
**(6) Power-fail compare mode register (ADA0PFM)**

The ADA0PFM register is an 8-bit register that sets the power-fail compare mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H       R/W       Address: FFFFF204H

| | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADA0PFM | ADA0PFE | ADA0PFC | 0 | 0 | 0 | 0 | 0 | 0 |

| ADA0PFE | Selection of power-fail compare enable/disable |
|---|---|
| 0 | Power-fail compare disabled |
| 1 | Power-fail compare enabled |

| ADA0PFC | Selection of power-fail compare mode |
|---|---|
| 0 | Generates an interrupt request signal (INTAD) when ADA0CRnH ≥ ADA0PFT |
| 1 | Generates an interrupt request signal (INTAD) when ADA0CRnH < ADA0PFT |

**Cautions 1. In the select mode, the 8-bit data set to the ADA0PFT register is compared with the value of the ADA0CRnH register specified by the ADA0S register. If the result matches the condition specified by the ADA0PFC bit, the conversion result is stored in the ADA0CRn register and the INTAD signal is generated. If it does not match, however, the interrupt signal is not generated.**

**2. In the scan mode, the 8-bit data set to the ADA0PFT register is compared with the contents of the ADA0CR0H register. If the result matches the condition specified by the ADA0PFC bit, the conversion result is stored in the ADA0CR0 register and the INTAD signal is generated. If it does not match, however, the INTAD signal is not generated. Regardless of the comparison result, the scan operation is continued and the conversion result is stored in the ADA0CRn register until the scan operation is completed. However, the INTAD signal is not generated after the scan operation has been completed.**

**3. In the following modes, write data to the ADA0PFM register while A/D conversion is stopped (ADA0M0.ADA0CE bit = 0), and then enable the A/D conversion operation (ADA0CE bit = 1).**

● **Normal conversion mode**

● **One-shot select mode/one-shot scan mode in high-speed conversion mode**

**(7) Power-fail compare threshold value register (ADA0PFT)**

The ADA0PFT register sets the compare value in the power-fail compare mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H      R/W      Address: FFFFF205H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADA0PFT | | | | | | | | |
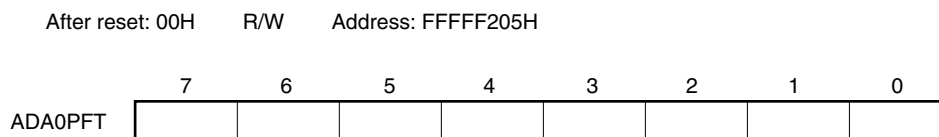
**Caution   In the following modes, write data to the ADA0PFT register while A/D conversion is stopped (ADA0M0.ADA0CE bit = 0), and then enable the A/D conversion operation (ADA0CE bit = 1).**

- **Normal conversion mode**
- **One-shot select mode/one-shot scan mode in high-speed conversion mode**

## 15.5  Operation

### 15.5.1  Basic operation

<1>  Set the operation mode, trigger mode, and conversion time for executing A/D conversion by using the ADA0M0, ADA0M1, ADA0M2, and ADA0S registers.  When the ADA0CE bit of the ADA0M0 register is set, conversion is started in the software trigger mode and the A/D converter waits for a trigger in the external or timer trigger mode.

<2>  When A/D conversion is started, the voltage input to the selected analog input channel is sampled by the sample & hold circuit.

<3>  When the sample & hold circuit samples the input channel for a specific time, it enters the hold status, and holds the input analog voltage until A/D conversion is complete.

<4>  Set bit 9 of the successive approximation register (SAR), and set the compare voltage generation DAC to (1/2) $AV_{REF0}$.

<5>  The voltage difference between the voltage of the compare voltage generation DAC and the analog input voltage is compared by the voltage comparator.  If the analog input voltage is higher than (1/2) $AV_{REF0}$, the MSB of the SAR remains set.  If it is lower than (1/2) $AV_{REF0}$, the MSB is reset.

<6>  Next, bit 8 of the SAR is automatically set and the next comparison is started.  Depending on the value of bit 9, to which a result has been already set, the compare voltage generation DAC is selected as follows.
   • Bit 9 = 1: (3/4) $AV_{REF0}$
   • Bit 9 = 0: (1/4) $AV_{REF0}$
   This compare voltage and the analog input voltage are compared and, depending on the result, bit 8 is manipulated as follows.
   Analog input voltage $\geq$ Compare voltage: Bit 8 = 1
   Analog input voltage $\leq$ Compare voltage: Bit 8 = 0

<7>  This comparison is continued to bit 0 of the SAR.

<8>  When comparison of the 10 bits is complete, the valid digital result remains in the SAR, and is then transferred to and stored in the ADA0CRn register.  After that, an A/D conversion end interrupt request signal (INTAD) is generated.

<9>  In one-shot select mode, conversion is stopped[Note].  In one-shot scan mode, conversion is stopped after scanning once[Note].  In continuous select mode, repeat steps <2> to <8> until the ADA0M0.ADA0CE bit is cleared to 0.  In continuous scan mode, repeat steps <2> to <8> for each channel.

   **Note**  In the external trigger mode, timer trigger mode 0, or timer trigger mode 1, the trigger standby status is entered.

   **Remark**  The trigger standby status means the status after the stabilization time has elapsed.

## 15.5.2  Conversion operation timing

**Figure 15-3.  Conversion Operation Timing (Continuous Conversion)**



**(1) Operation in normal conversion mode (ADA0HS1 bit = 0)**

**(2) Operation in high-speed conversion mode (ADA0HS1 bit = 1)**

**Remark**  The above timings are applicable when a trigger is generated within the stabilization time.  If a trigger is generated after the stabilization time has elapsed, a trigger response time is inserted.
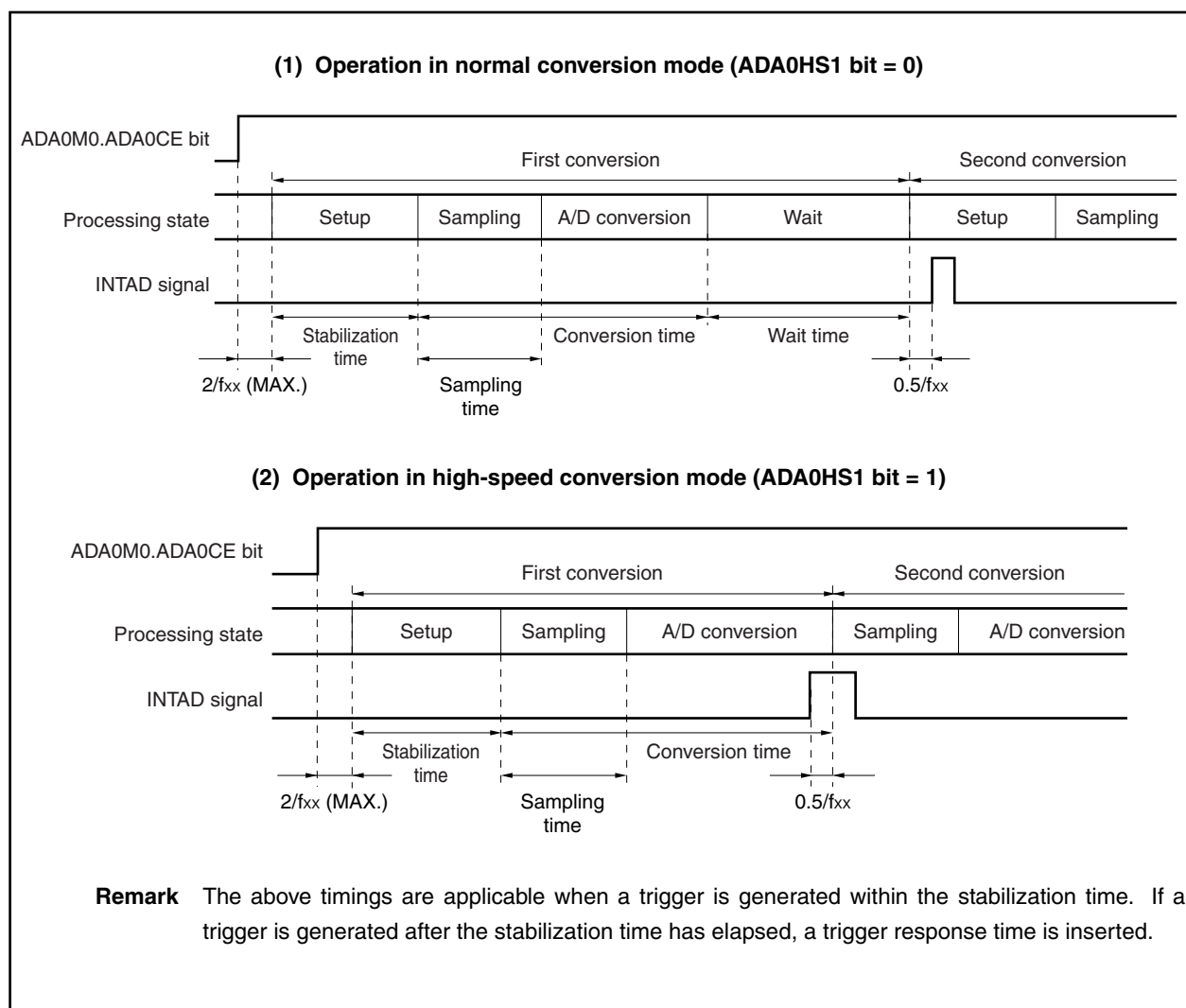
### 15.5.3  Trigger mode

The timing of starting the conversion operation is specified by setting the trigger mode.  The trigger mode includes the software trigger mode and hardware trigger modes.  The hardware trigger modes include timer trigger modes 0 and 1, and external trigger mode.  The ADA0M0.ADA0TMD bit is used to set the trigger mode.  The hardware trigger modes are set by the ADA0M2.ADA0TMD1 and ADA0M2.ADA0TMD0 bits.

#### (1)  Software trigger mode

When the ADA0M0.ADA0CE bit is set to 1, the signal of the analog input pin (ANI0 to ANI11 pin) specified by the ADA0S register is converted.  When conversion is complete, the result is stored in the ADA0CRn register.  At the same time, the A/D conversion end interrupt request signal (INTAD) is generated.

If the operation mode specified by the ADA0M0.ADA0MD1 and ADA0M0.ADA0MD0 bits is the continuous select/scan mode, the next conversion is repeated, unless the ADA0CE bit is cleared to 0 after completion of the conversion.  Conversion is performed once and ends if the operation mode is the one-shot select/scan mode.

When conversion is started, the ADA0M0.ADA0EF bit is set to 1 (indicating that conversion is in progress).

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written during conversion, the conversion is aborted and started again from the beginning.  However, writing to these registers is prohibited in the normal conversion mode and one-shot select mode/one-shot scan mode in the high-speed conversion mode.

#### (2)  External trigger mode

In this mode, converting the signal of the analog input pin (ANI0 to ANI11) specified by the ADA0S register is started when an external trigger is input (to the ADTRG pin).  Which edge of the external trigger is to be detected (i.e., the rising edge, falling edge, or both rising and falling edges) can be specified by using the ADA0M0.ADA0ETS1 and ADA0M0.ATA0ETS0 bits.  When the ADA0CE bit is set to 1, the A/D converter waits for the trigger, and starts conversion after the external trigger has been input.

When conversion is completed, the result of conversion is stored in the ADA0CRn register, regardless of whether the continuous select, continuous scan, one-shot select, or one-shot scan mode is set as the operation mode by the ADA0MD1 and ADA0MD0 bits.  At the same time, the INTAD signal is generated, and the A/D converter waits for the trigger again.

When conversion is started, the ADA0EF bit is set to 1 (indicating that conversion is in progress).  While the A/D converter is waiting for the trigger, however, the ADA0EF bit is cleared to 0 (indicating that conversion is stopped).  If the valid trigger is input during the conversion operation, the conversion is aborted and started again from the beginning.

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written during the conversion operation, the conversion is aborted, and the A/D converter waits for the trigger again.  However, writing to these registers is prohibited in the one-shot select mode/one-shot scan mode.

**Caution**  **To select the external trigger mode, set the high-speed conversion mode.  Do not input a trigger during the stabilization time that is inserted once after the A/D conversion operation is enabled (ADA0M0.ADA0CE bit = 1).**

**Remark**  The trigger standby status means the status after the stabilization time has elapsed.

**(3)  Timer trigger mode**

In this mode, converting the signal of the analog input pin (ANI0 to ANI11) specified by the ADA0S register is started by the compare match interrupt request signal (INTTAA2CC0 or INTTAA2CC1) of the capture/compare register connected to the timer.  The INTTAA2CC0 or INTTAA2CC1 signal is selected by the ADA0TMD1 and ADA0TMD0 bits, and conversion is started at the rising edge of the specified compare match interrupt request signal.  When the ADA0CE bit is set to 1, the A/D converter waits for a trigger, and starts conversion when the compare match interrupt request signal of the timer is input.

When conversion is completed, regardless of whether the continuous select, continuous scan, one-shot select, or one-shot scan mode is set as the operation mode by the ADA0MD1 and ADA0MD0 bits, the result of the conversion is stored in the ADA0CRn register.  At the same time, the INTAD signal is generated, and the A/D converter waits for the trigger again.

When conversion is started, the ADA0EF bit is set to 1 (indicating that conversion is in progress).  While the A/D converter is waiting for the trigger, however, the ADA0EF bit is cleared to 0 (indicating that conversion is stopped).  If the valid trigger is input during the conversion operation, the conversion is aborted and started again from the beginning.

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written during conversion, the conversion is stopped and the A/D converter waits for the trigger again.  However, writing to these registers is prohibited in the one-shot select mode/one-shot scan mode.

**Caution   To select the timer trigger mode, set the high-speed conversion mode.  Do not input a trigger during the stabilization time that is inserted once after the A/D conversion operation is enabled (ADA0M0.ADA0CE bit = 1).**

**Remark**   The trigger standby status means the status after the stabilization time has elapsed.

### 15.5.4  Operation mode

Four operation modes are available as the modes in which to set the ANI0 to ANI11 pins: continuous select mode, continuous scan mode, one-shot select mode, and one-shot scan mode.

The operation mode is selected by the ADA0M0.ADA0MD1 and ADA0M0.ADA0MD0 bits.

#### (1)  Continuous select mode

In this mode, the voltage of one analog input pin selected by the ADA0S register is continuously converted into a digital value.

The conversion result is stored in the ADA0CRn register corresponding to the analog input pin.  In this mode, an analog input pin corresponds to an ADA0CRn register on a one-to-one basis.  Each time A/D conversion is completed, the A/D conversion end interrupt request signal (INTAD) is generated.  After completion of conversion, the next conversion is started, unless the ADA0M0.ADA0CE bit is cleared to 0 (n = 0 to 11).

**Figure 15-4.  Timing Example of Continuous Select Mode Operation (ADA0S Register = 01H)**
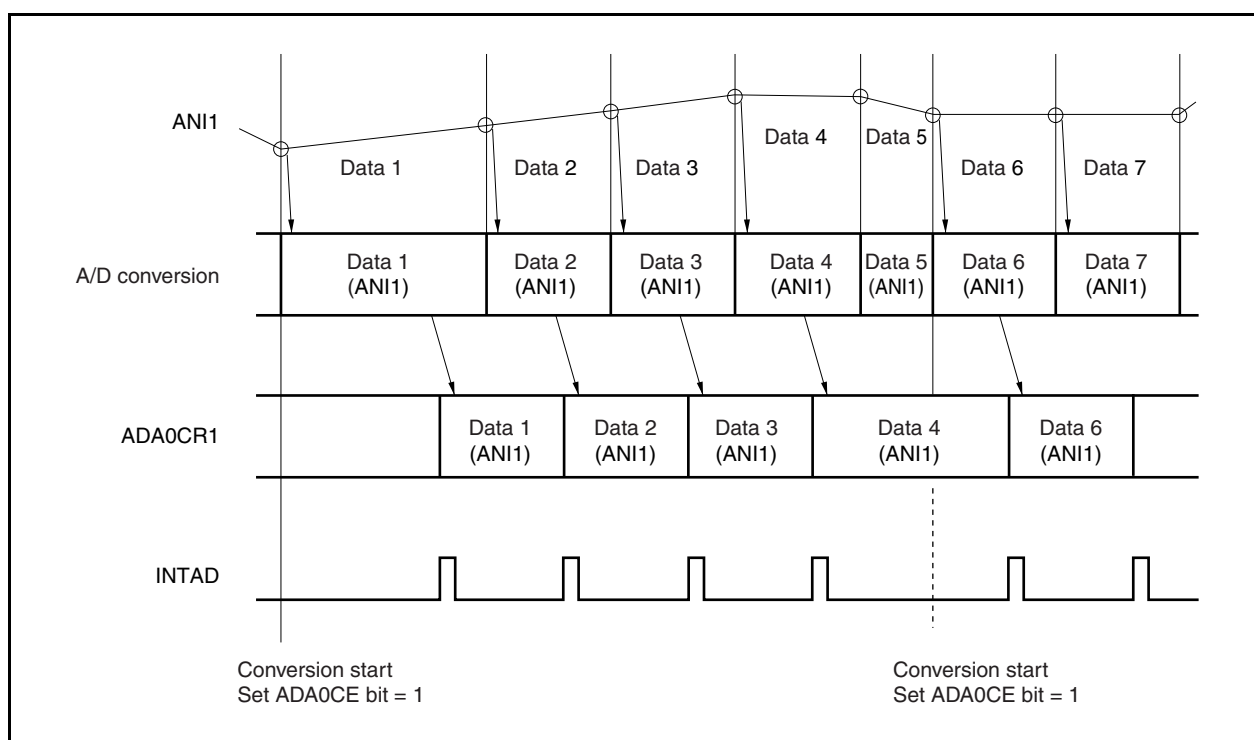


#### (2)  Continuous scan mode

In this mode, analog input pins are sequentially selected, from the ANI0 pin to the pin specified by the ADA0S register, and their values are continuously converted into digital values.

The result of each conversion is stored in the ADA0CRn register corresponding to the analog input pin.  When conversion of the analog input pin specified by the ADA0S register is complete, the INTAD signal is generated, and A/D conversion is started again from the ANI0 pin, unless the ADA0CE bit is cleared to 0 (n = 0 to 11).

**Figure 15-5.  Timing Example of Continuous Scan Mode Operation (ADA0S Register = 03H)**
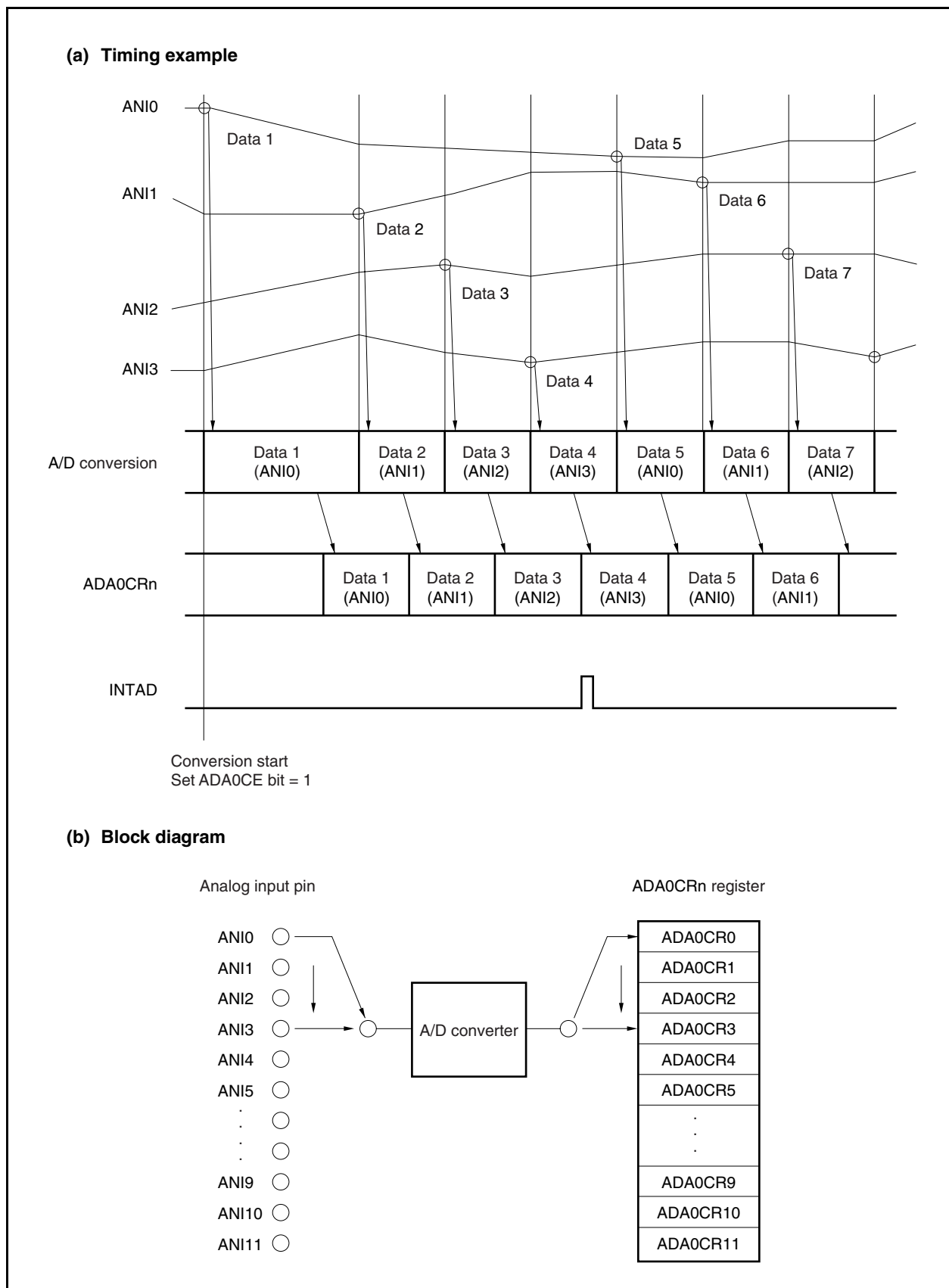
**(a)  Timing example**



Conversion start
Set ADA0CE bit = 1

**(b)  Block diagram**

**(3) One-shot select mode**

In this mode, the voltage of one analog input pin specified by the ADA0S register is converted into a digital value only once.

The conversion result is stored in the ADA0CRn register corresponding to the analog input pin. In this mode, an analog input pin and an ADA0CRn register correspond on a one-to-one basis. When A/D conversion has been completed once, the INTAD signal is generated. The A/D conversion operation is stopped after it has been completed (n = 0 to 11).

**Figure 15-6.  Timing Example of One-Shot Select Mode Operation (ADA0S Register = 01H)**
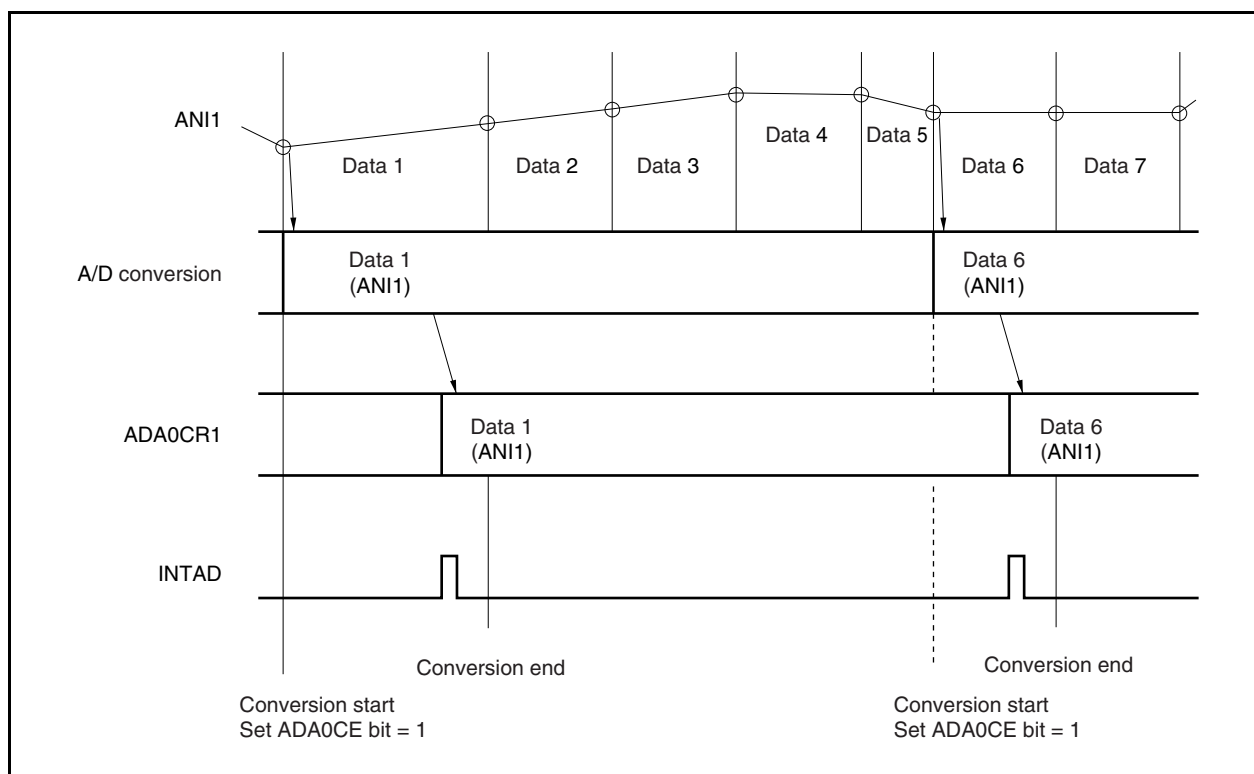


**(4) One-shot scan mode**

In this mode, analog input pins are sequentially selected, from the ANI0 pin to the pin specified by the ADA0S register, and their values are converted into digital values .

Each conversion result is stored in the ADA0CRn register corresponding to the analog input pin. When conversion of the analog input pin specified by the ADA0S register is complete, the INTAD signal is generated. A/D conversion is stopped after it has been completed (n = 0 to 11).

**Figure 15-7. Timing Example of One-Shot Scan Mode Operation (ADA0S Register = 03H)**

**(a) Timing example**



**(b) Block diagram**
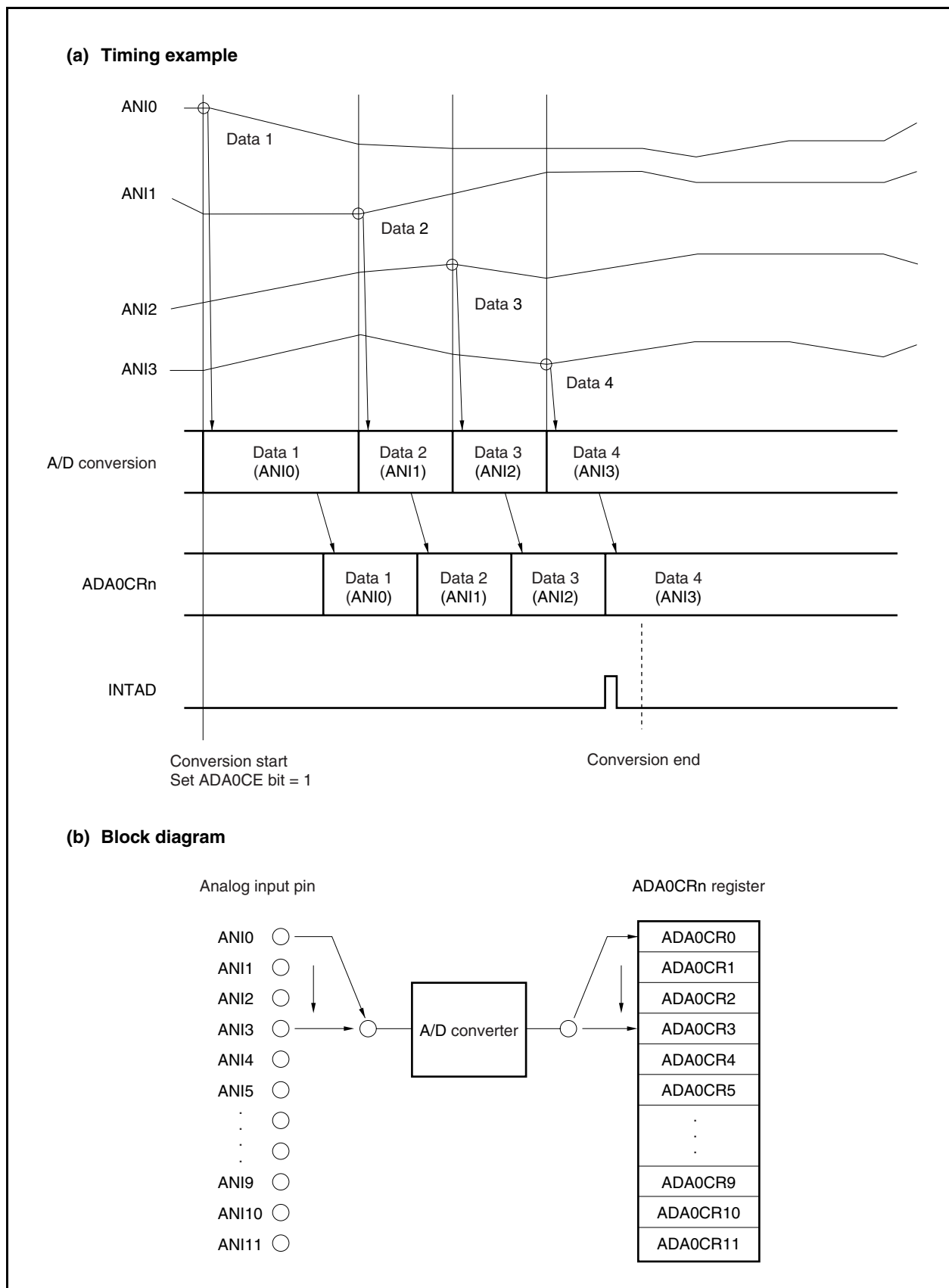
### 15.5.5 Power-fail compare mode

The A/D conversion end interrupt request signal (INTAD) can be controlled as follows by the ADA0PFM and ADA0PFT registers.

- When the ADA0PFM.ADA0PFE bit = 0, the INTAD signal is generated each time conversion is completed (normal use of the A/D converter).
- When the ADA0PFE bit = 1 and when the ADA0PFM.ADA0PFC bit = 0, the value of the ADA0CRnH register is compared with the value of the ADA0PFT register when conversion is completed, and the INTAD signal is generated only if ADA0CRnH $\geq$ ADA0PFT.
- When the ADA0PFE bit = 1 and when the ADA0PFC bit = 1, the value of the ADA0CRnH register is compared with the value of the ADA0PFT register when conversion is completed, and the INTAD signal is generated only if ADA0CRnH < ADA0PFT.

**Remark**    n = 0 to 11

In the power-fail compare mode, four modes are available as modes in which to set the ANI0 to ANI11 pins: continuous select mode, continuous scan mode, one-shot select mode, and one-shot scan mode.

**(1) Continuous select mode**

In this mode, the result of converting the voltage of the analog input pin specified by the ADA0S register is compared with the set value of the ADA0PFT register. If the result of power-fail comparison matches the condition set by the ADA0PFC bit, the conversion result is stored in the ADA0CRn register, and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CRn register, and the INTAD signal is not generated. After completion of the first conversion, the next conversion is started, unless the ADA0M0.ADA0CE bit is cleared to 0 (n = 0 to 11).

**Figure 15-8. Timing Example of Continuous Select Mode Operation**
**(When Power-Fail Comparison Is Made: ADA0S Register = 01H)**



**(2) Continuous scan mode**

In this mode, the results of converting the voltages of the analog input pins sequentially selected from the ANI0 pin to the pin specified by the ADA0S register are stored, and the set value of the ADA0CR0H register of channel 0 is compared with the value of the ADA0PFT register. If the result of power-fail comparison matches the condition set by the ADA0PFC bit, the conversion result is stored in the ADA0CR0 register, and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CR0 register, and the INTAD signal is not generated.

After the result of the first conversion has been stored in the ADA0CR0 register, the results of sequentially converting the voltages on the analog input pins up to the pin specified by the ADA0S register are continuously stored. After completion of conversion, the next conversion is started from the ANI0 pin again, unless the ADA0CE bit is cleared to 0.

**Figure 15-9.  Timing Example of Continuous Scan Mode Operation**
**(When Power-Fail Comparison Is Made: ADA0S Register = 03H)**

**(a)  Timing example**



**(b)  Block diagram**

**(3) One-shot select mode**

In this mode, the result of converting the voltage of the analog input pin specified by the ADA0S register is compared with the set value of the ADA0PFT register. If the result of power-fail comparison matches the condition set by the ADA0PFC bit, the conversion result is stored in the ADA0CRn register, and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CRn register, and the INTAD signal is not generated. Conversion is stopped after it has been completed.

**Figure 15-10.  Timing Example of One-Shot Select Mode Operation**
**(When Power-Fail Comparison Is Made: ADA0S Register = 01H)**



**(4) One-shot scan mode**

In this mode, the results of converting the voltages of the analog input pins sequentially selected from the ANI0 pin to the pin specified by the ADA0S register are stored, and the set value of the ADA0CR0H register of channel 0 is compared with the set value of the ADA0PFT register. If the result of power-fail comparison matches the condition set by the ADA0PFC bit, the conversion result is stored in the ADA0CR0 register and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CR0 register, and the INTAD0 signal is not generated. After the result of the first conversion has been stored in the ADA0CR0 register, the results of converting the signals on the analog input pins specified by the ADA0S register are sequentially stored. The conversion is stopped after it has been completed.

**Figure 15-11. Timing Example of One-Shot Scan Mode Operation**
**(When Power-Fail Comparison Is Made: ADA0S Register = 03H)**

**(a)  Timing example**



**(b)  Block diagram**
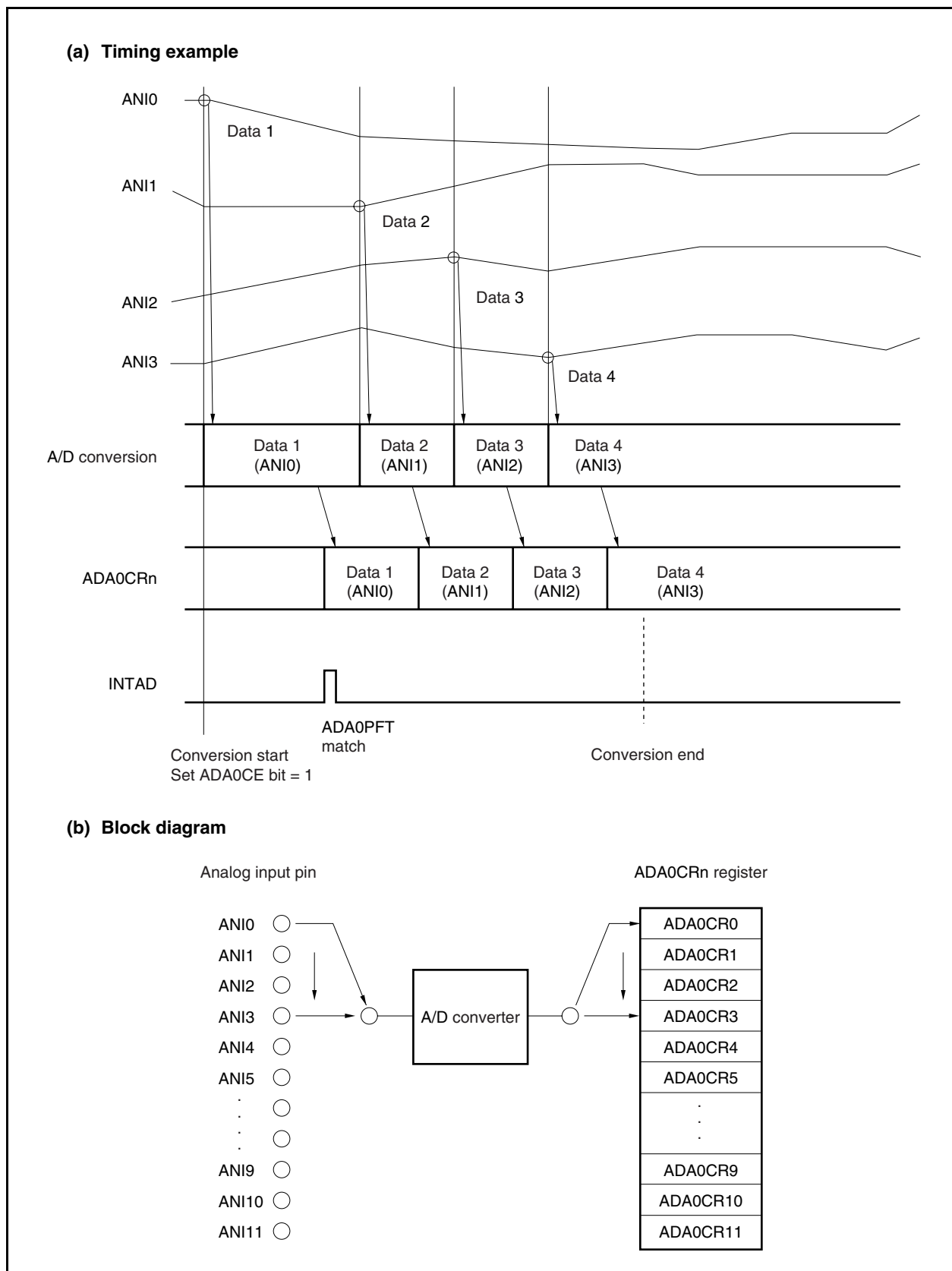
## 15.6  Cautions

**(1)  When A/D converter is not used**

When the A/D converter is not used, the power consumption can be reduced by clearing the ADA0M0.ADA0CE bit to 0.

**(2)  Input range of ANI0 to ANI11 pins**

Input the voltage within the specified range to the ANI0 to ANI11 pins. If a voltage equal to or higher than $AV_{REF0}$ or equal to or lower than $AV_{SS}$ (even within the range of the absolute maximum ratings) is input to any of these pins, the conversion value of that channel is undefined, and the conversion value of the other channels may also be affected.

**(3)  Countermeasures against noise**

To maintain the 10-bit resolution, the ANI0 to ANI11 pins must be effectively protected from noise. The effect of noise increases as the output impedance of the analog input source becomes higher. To lower the noise, connecting an external capacitor as shown in Figure 15-12 is recommended.

**Figure 15-12.  Processing of Analog Input Pin**
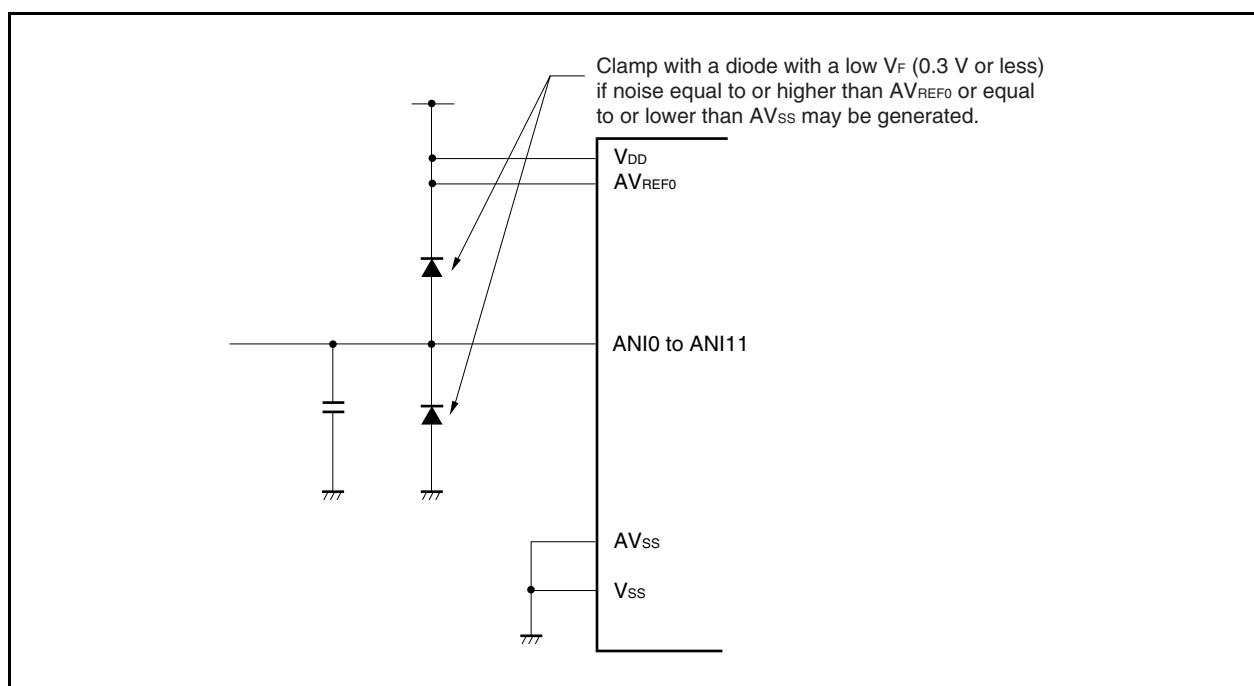


**(4)  Alternate I/O**

The analog input pins (ANI0 to ANI11) function alternately as port pins. When selecting one of the ANI0 to ANI11 pins to execute A/D conversion, do not execute an instruction to read an input port or write to an output port during conversion as the conversion resolution may drop.

Also the conversion resolution may drop at the pins set as output port pins during A/D conversion if the output current fluctuates due to the effect of the external circuit connected to the port pins.

If a digital pulse is applied to a pin adjacent to the pin whose input signal is being converted, the A/D conversion value may not be as expected due to the effect of coupling noise. Therefore, do not apply a pulse to a pin adjacent to the pin undergoing A/D conversion.

**(5) Interrupt request flag (ADIF)**

The interrupt request flag (ADIF) is not cleared even if the contents of the ADA0S register are changed. If the analog input pin is changed during A/D conversion, therefore, the result of converting the previously selected analog input signal may be stored and the conversion end interrupt request flag may be set immediately before the ADA0S register is rewritten. If the ADIF flag is read immediately after the ADA0S register is rewritten, the ADIF flag may be set even though the A/D conversion of the newly selected analog input pin has not been completed. When A/D conversion is stopped, clear the ADIF flag before resuming conversion.

**Figure 15-13.  Generation Timing of A/D Conversion End Interrupt Request**
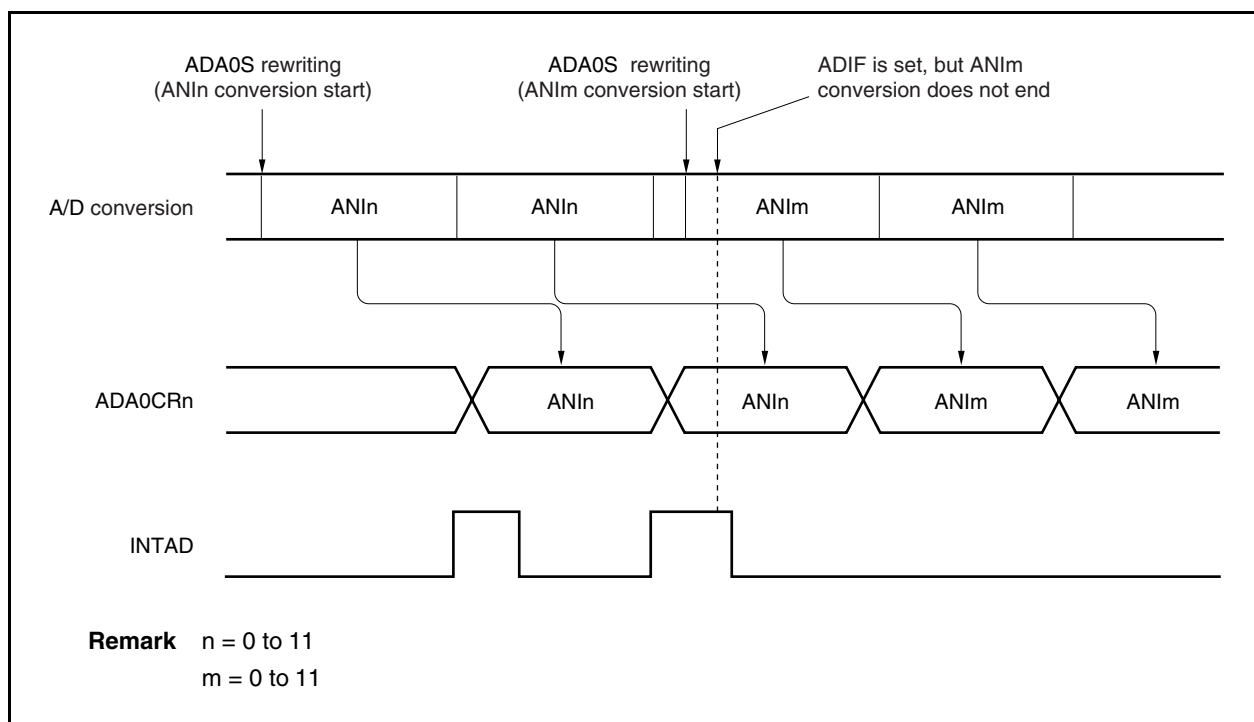


**(6) Internal equivalent circuit**

The following shows the equivalent circuit of the analog input block.

**Figure 15-14.  Internal Equivalent Circuit of ANIn Pin**



| $R_{IN}$ | $C_{IN}$ |
|---|---|
| 14 kΩ | 8.4 pF |

**Remarks 1.** The above values are reference values.
**2.** n = 0 to 11
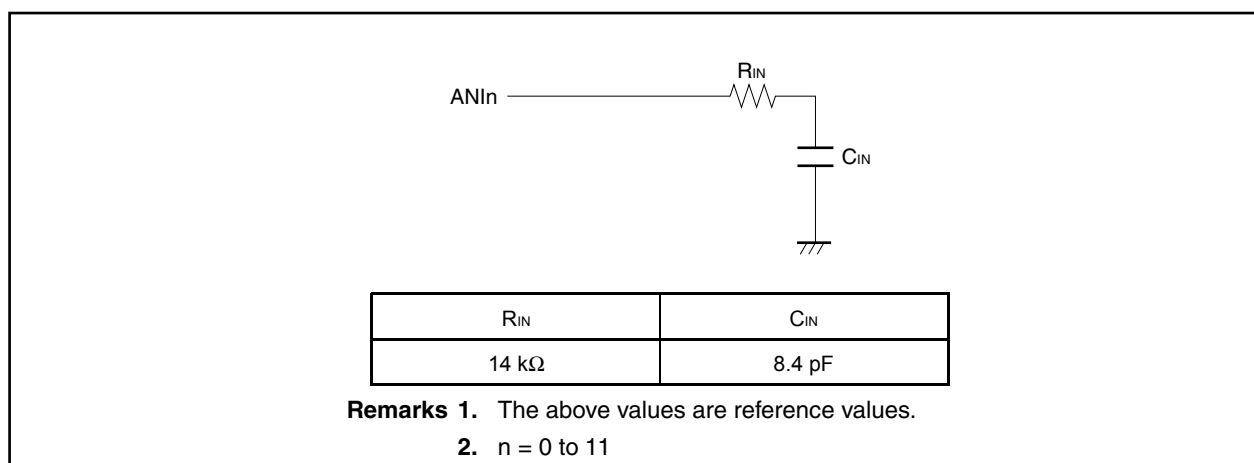
**(7) AV$_{REF0}$ pin**

(a) The AV$_{REF0}$ pin is used as the power supply pin of the A/D converter and also supplies power to the alternate-function ports.  In an application where a backup power supply is used, be sure to supply the same potential as V$_{DD}$ to the AV$_{REF0}$ pin as shown in Figure 15-15.

(b) The AV$_{REF0}$ pin is also used as the reference voltage pin of the A/D converter.  If the source supplying power to the AV$_{REF0}$ pin has a high impedance or if the power supply has a low current supply capability, the reference voltage may fluctuate due to the current that flows during conversion (especially, immediately after the conversion operation enable bit ADA0CE has been set to 1).  As a result, the conversion accuracy may drop. To avoid this, it is recommended to connect a capacitor across the AV$_{REF0}$ and AV$_{SS}$ pins to suppress the reference voltage fluctuation as shown in Figure 15-15.

(c) If the source supplying power to the AV$_{REF0}$ pin has a high DC resistance (for example, because of insertion of a diode), the voltage when conversion is enabled may be lower than the voltage when conversion is stopped, because of a voltage drop caused by the A/D conversion current.

**Figure 15-15.  AV$_{REF0}$ Pin Processing Example**



**Note**  Parasitic inductance
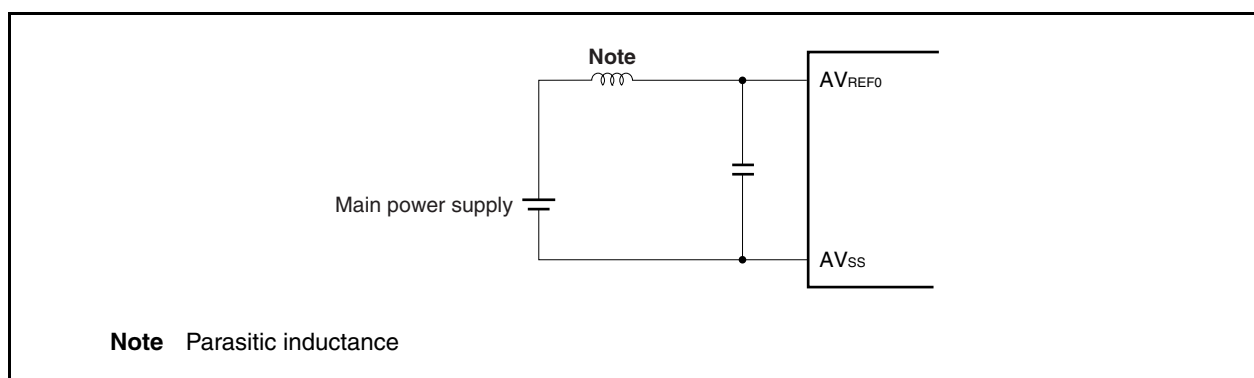
**(8)  Reading ADA0CRn register**
When the ADA0M0 to ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written, the contents of the ADA0CRn register may be undefined.  Read the conversion result after completion of conversion and before writing to the ADA0M0 to ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register.  Also, when an external/timer trigger is acknowledged, the contents of the ADA0CRn register may be undefined.  Read the conversion result after completion of conversion and before the next external/timer trigger is acknowledged.  The correct conversion result may not be read at a timing different from the above.

**(9)  External trigger mode**
When using the external trigger mode, the input trigger during A/D conversion will not be acknowledged.

**(10) Standby mode**

Because the A/D converter stops operating in the STOP mode, the conversion results are invalid, so power consumption can be reduced.  Operations are resumed after the STOP mode is released, but the A/D conversion results after the STOP mode is released are invalid.  When using the A/D converter after the STOP mode is released, clear the ADA0M0.ADA0CE bit to 0 before setting the STOP mode or after releasing the STOP mode, then set the ADA0CE bit to 1 after releasing the STOP mode.

In the IDLE1, IDLE2, or subclock operation mode, operation continues.  To lower the power consumption, therefore, clear the ADA0M0.ADA0CE bit to 0.  In the IDLE1 and IDLE2 modes, since the analog input voltage value cannot be retained, the A/D conversion results after the IDLE1 and IDLE2 modes are released are invalid.  The results of conversions before the IDLE1 and IDLE2 modes were set are valid.

**(11)  High-speed conversion mode**

In the high-speed conversion mode, rewriting the ADA0M0, ADA0M2, ADA0S, ADA0PFM, and ADA0PFT registers and trigger input during the stabilization time are prohibited.

**(12)  A/D conversion time**

The A/D conversion time is the total of the stabilization time, conversion time, wait time, and trigger response time (for details of these times, refer to **Table 15-2  Conversion Time Selection in Normal Conversion Mode (ADA0HS1 Bit = 0)** and **Table 15-3  Conversion Time Selection in High-Speed Conversion Mode (ADA0HS1 Bit = 1)**).

During A/D conversion in the normal conversion mode, if the ADA0M0, ADA0M2, ADA0S, ADA0PFM, and ADA0PFT registers are written or a trigger is input, reconversion is carried out.  However, if the stabilization time end timing conflicts with writing to these registers, or if the stabilization time end timing conflicts with the trigger input, a stabilization time of 64 clocks is reinserted.

If a conflict occurs again with the reinserted stabilization time end timing, the stabilization time is reinserted.  Therefore do not set the trigger input interval and control register write interval to 64 clocks or lower.

**(13)  Variation of A/D conversion results**

The results of A/D conversion may vary due to a fluctuation in the supply voltage or the effect of noise.  To reduce this variation, take countermeasures with the program such as averaging the A/D conversion results.

**(14)  A/D conversion result hysteresis characteristics**

The successive comparison type A/D converter holds the analog input voltage in the internal sample & hold capacitor and then performs A/D conversion.  After A/D conversion has finished, the analog input voltage remains in the internal sample & hold capacitor.  As a result, the following phenomena may occur.

- When the same channel is used for A/D conversions, if the voltage is higher or lower than the previous A/D conversion, then hysteresis characteristics may appear in which the conversion result is affected by the previous value.  Thus, even if the conversion is performed at the same potential, the result may vary.
- When switching the analog input channel, hysteresis characteristics may appear in which the conversion result is affected by the previous channel value.  This is because one A/D converter is used for the A/D conversions.  Thus, even if the conversion is performed at the same potential, the result may vary.

## 15.7  How to Read A/D Converter Characteristics Table

This section describes the terms related to the A/D converter.

### (1)  Resolution

The minimum analog input voltage that can be recognized, i.e., the ratio of an analog input voltage to 1 bit of digital output is called 1 LSB (least significant bit).  The ratio of 1 LSB to the full scale is expressed as %FSR (full-scale range).  %FSR is the ratio of a range of convertible analog input voltages expressed as a percentage, and can be expressed as follows, independently of the resolution.

$1\%FSR$ = (Maximum value of convertible analog input voltage – Minimum value of convertible analog input voltage)/100

$\quad\quad\quad\quad = (AV_{REF0} - 0)/100$

$\quad\quad\quad\quad = AV_{REF0}/100$

When the resolution is 10 bits, 1 LSB is as follows:

$1\ LSB = 1/2^{10} = 1/1{,}024$

$\quad\quad\quad\quad = 0.098\%FSR$

The accuracy is determined by the overall error, independently of the resolution.

### (2)  Overall error

This is the maximum value of the difference between an actually measured value and a theoretical value.

It is a total of zero-scale error, full-scale error, linearity error, and a combination of these errors.

The overall error in the characteristics table does not include the quantization error.

**Figure 15-16.  Overall Error**
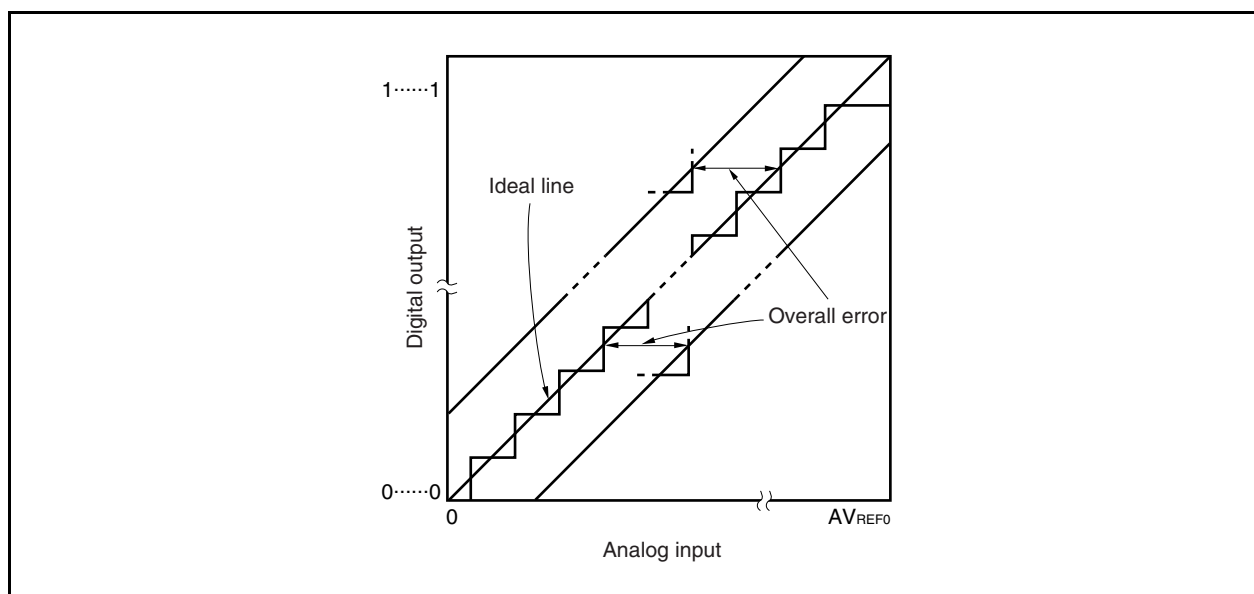
**(3) Quantization error**

This is an error of $\pm 1/2$ LSB that inevitably occurs when an analog value is converted into a digital value. Because the A/D converter converts analog input voltages in a range of $\pm 1/2$ LSB into the same digital codes, a quantization error is unavoidable.

This error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, or differential linearity error in the characteristics table.

**Figure 15-17.  Quantization Error**
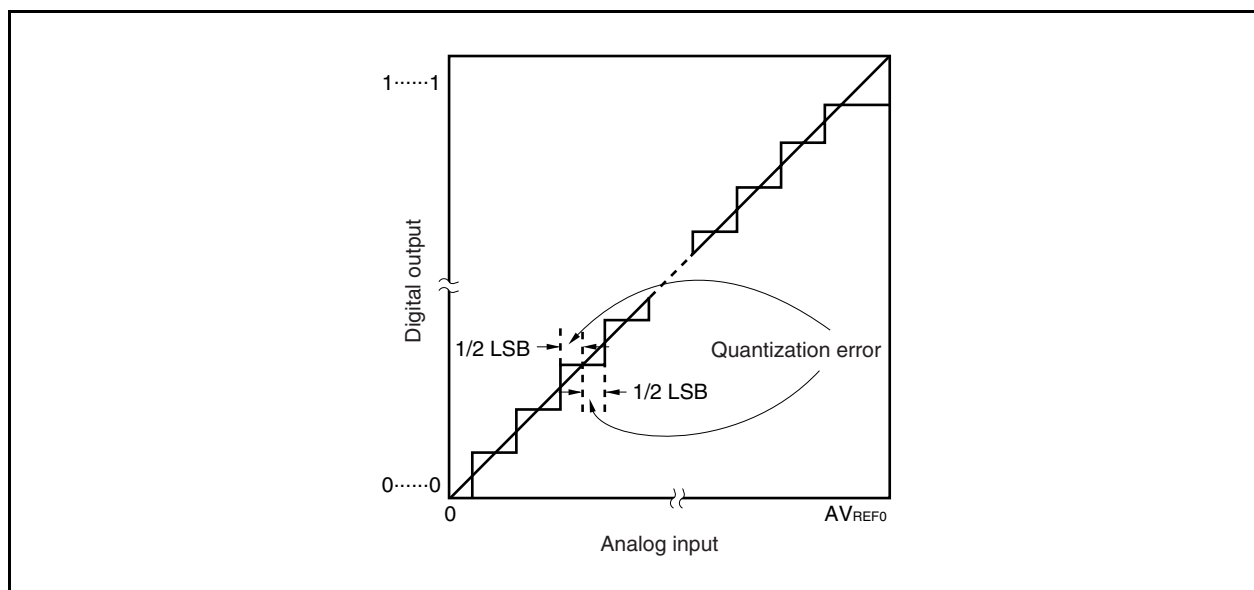


**(4) Zero-scale error**

This is the difference between the actually measured analog input voltage and its theoretical value when the digital output changes from 0…000 to 0…001 (1/2 LSB).

**Figure 15-18.  Zero-Scale Error**
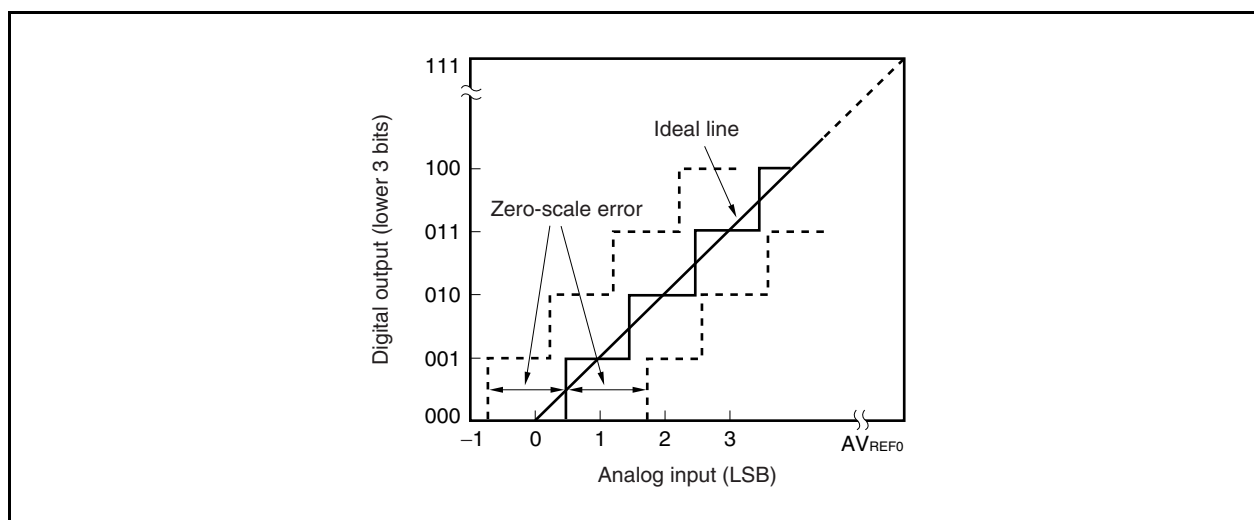
**(5) Full-scale error**

This is the difference between the actually measured analog input voltage and its theoretical value when the digital output changes from 1…110 to 1…111 (full scale − 3/2 LSB).

**Figure 15-19. Full-Scale Error**
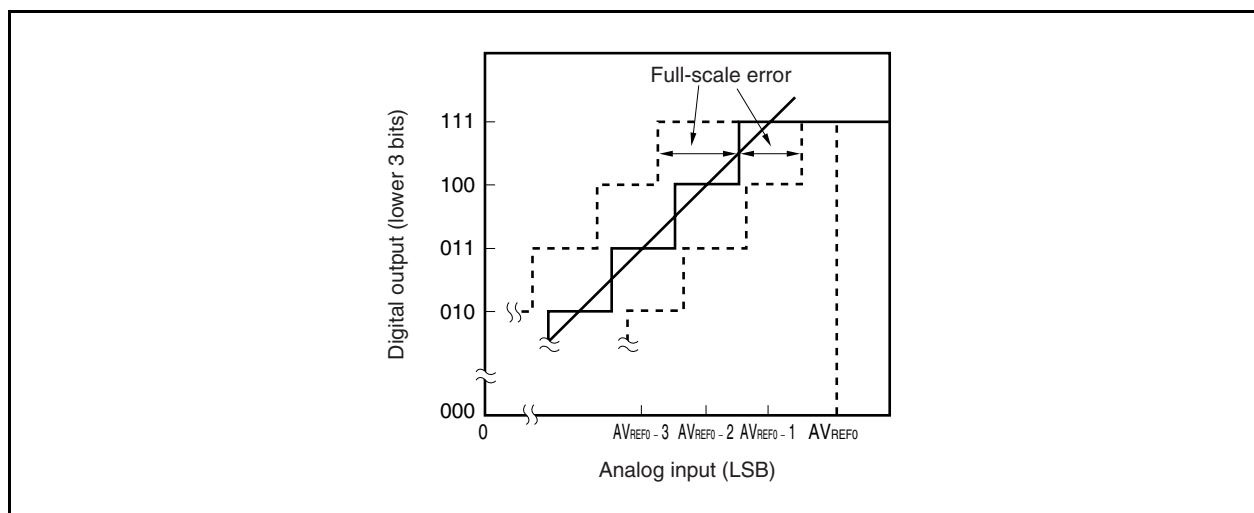


**(6) Differential linearity error**

Ideally, the width to output a specific code is 1 LSB. This error indicates the difference between the actually measured value and its theoretical value when a specific code is output. This indicates the basic characteristics of the A/D conversion when the voltage applied to the analog input pins of the same channel is consistently increased bit by bit from $AV_{SS}$ to $AV_{REF0}$. When the input voltage is increased or decreased, or when two or more channels are used, see **15.7 (2) Overall error**.

**Figure 15-20. Differential Linearity Error**
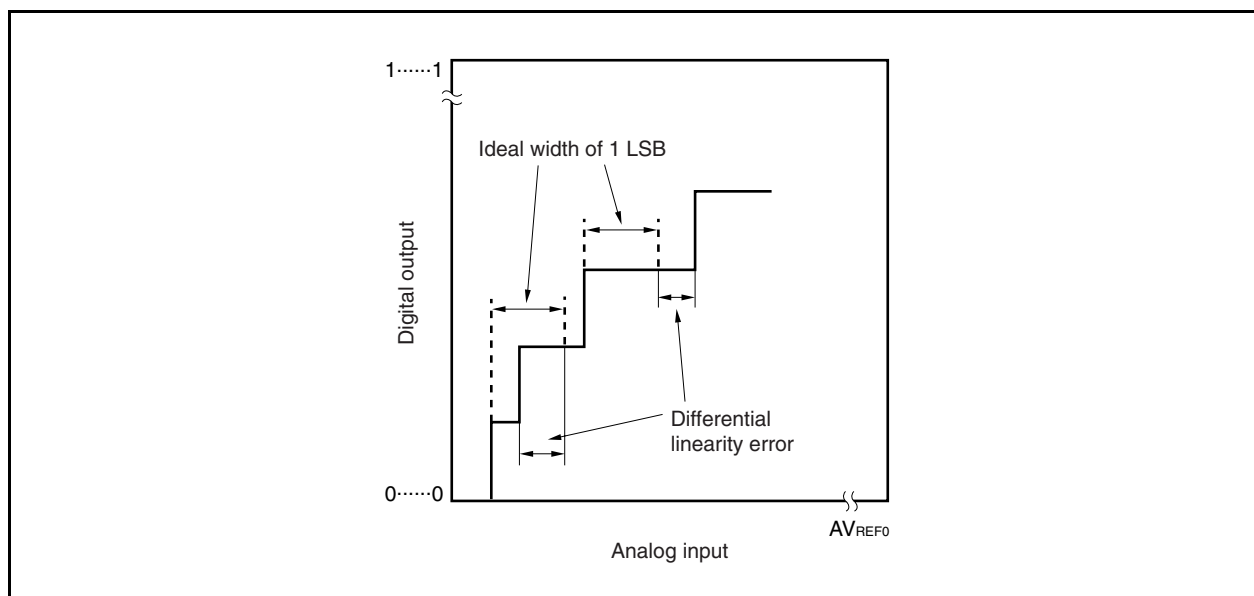
**(7) Integral linearity error**

This error indicates the extent to which the conversion characteristics differ from the ideal linear relationship.  It indicates the maximum value of the difference between the actually measured value and its theoretical value where the zero-scale error and full-scale error are 0.

**Figure 15-21.  Integral Linearity Error**
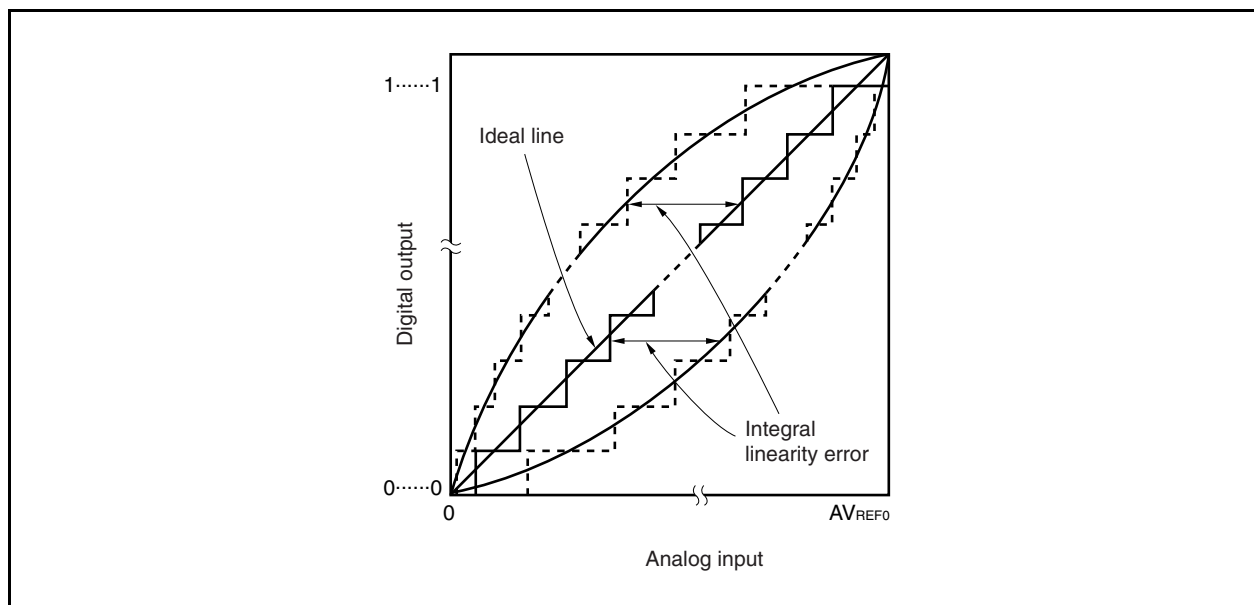


**(8) Conversion time**

This is the time required to obtain a digital output after each trigger has been generated.

The conversion time in the characteristics table includes the sampling time.

**(9) Sampling time**

This is the time for which the analog switch is ON to load an analog voltage to the sample & hold circuit.

**Figure 15-22.  Sampling Time**
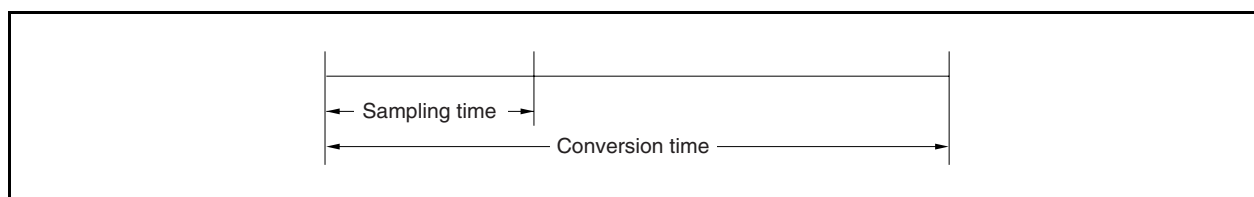
# CHAPTER 16 D/A CONVERTER

## 16.1 Functions

The D/A converter has the following functions.

○ 8-bit resolution × 2 channels (DA0CS0, DA0CS1)
○ R-2R ladder method
○ Settling time: 3 $\mu$s max. (when $AV_{REF1}$ is 3.0 to 3.6 V and external load is 20 pF)
○ Analog output voltage: $AV_{REF1}$ × m/256 (m = 0 to 255; value set to DA0CSn register)
○ Operation modes: Normal mode, real-time output mode

**Remark** n = 0, 1

## 16.2 Configuration

The D/A converter configuration is shown below.

**Figure 16-1. Block Diagram of D/A Converter**



**Cautions 1. DAC0 and DAC1 share the $AV_{REF1}$ pin.**
      **2. DAC0 and DAC1 share the $AV_{SS}$ pin. The $AV_{SS}$ pin is also shared by the A/D converter.**
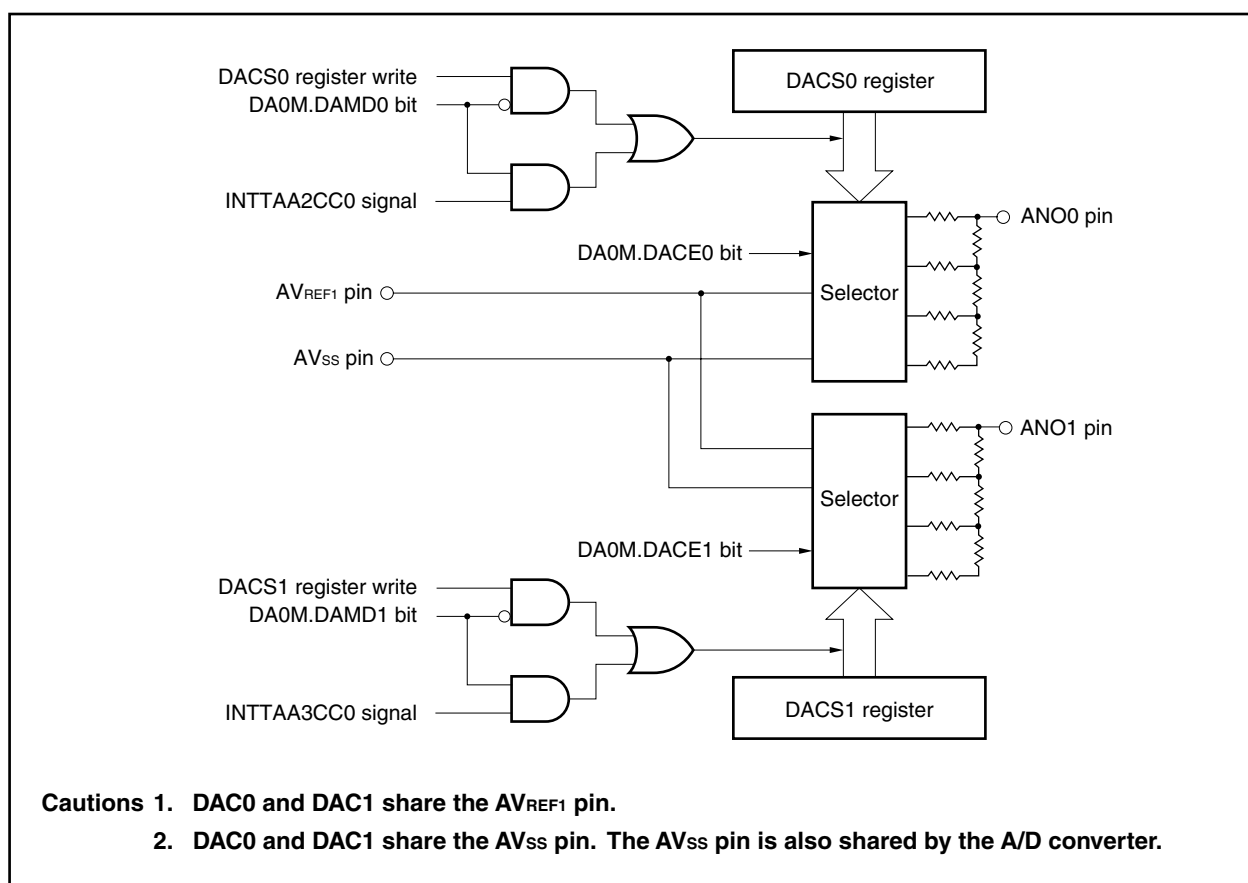
The D/A converter includes the following hardware.

**Table 16-1.  Configuration of D/A Converter**

| Item | Configuration |
|---|---|
| Control registers | D/A converter mode register (DA0M) |
| | D/A conversion value setting registers 0, 1 (DA0CS0, DA0CS1) |

## 16.3  Registers

The registers that control the D/A converter are as follows.

- D/A converter mode register (DA0M)
- D/A conversion value setting registers 0, 1 (DA0CS0, DA0CS1)

### (1)  D/A converter mode register (DA0M)

The DA0M register controls the operation of the D/A converter.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset:  00H     R/W     Address:  FFFFF282H

|  | 7 | 6 | <5> | <4> | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DA0M | 0 | 0 | DA0CE1 | DA0CE0 | 0 | 0 | DA0MD1 | DA0MD0 |

| DA0CEn | Control of D/A converter operation enable/disable (n = 0, 1) |
|---|---|
| 0 | Disables operation |
| 1 | Enables operation |

| DA0MDn | Selection of D/A converter operation mode (n = 0, 1) |
|---|---|
| 0 | Normal mode |
| 1 | Real-time output mode[Note] |

**Note**  The output trigger in the real-time output mode (DA0MDn bit = 1) is as follows.
- When n = 0: INTTAA2CC0 signal (see **CHAPTER 7  16-BIT TIMER/EVENT COUNTER AA (TAA)**)
- When n = 1: INTTAA3CC0 signal (see **CHAPTER 7  16-BIT TIMER/EVENT COUNTER AA (TAA)**)

**(2) D/A conversion value setting registers 0, 1 (DA0CS0, DA0CS1)**

The DA0CS0 and DA0CS1 registers set the analog voltage value output to the ANO0 and ANO1 pins.

These registers can be read or written in 8-bit units.

Reset sets these registers to 00H.

<div style="border:1px solid black;padding:20px;">

After reset: 00H       R/W       Address: DA0CS0 FFFFF280H, DA0CS1 FFFFF281H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DA0CSn | DA0CSn7 | DA0CSn6 | DA0CSn5 | DA0CSn4 | DA0CSn3 | DA0CSn2 | DA0CSn1 | DA0CSn0 |

**Caution**   **In the real-time output mode (DA0M.DA0MDn bit = 1), set the DA0CSn register before the INTTAA2CC0/INTTAA3CC0 signals are generated.  D/A conversion starts when the INTTAA2CC0/INTTAA3CC0 signals are generated.**

**Remark**   n = 0, 1

</div>

## 16.4 Operation

### 16.4.1 Operation in normal mode

D/A conversion is performed using a write operation to the DA0CSn register as the trigger.

The setting method is described below.

<1> Set the DA0M.DA0MDn bit to 0 (normal mode).

<2> Set the analog voltage value to be output to the ANOn pin to the DA0CSn register.

      Steps <1> and <2> above constitute the initial settings.

<3> Set the DA0M.DA0CEn bit to 1 (D/A conversion enable).

      D/A conversion starts when this setting is performed.

<4> To perform subsequent D/A conversions, write to the DA0CSn register.

      The previous D/A conversion result is held until the next D/A conversion is performed.

**Remarks 1.** For the alternate-function pin settings, see **Table 4-20 Using Port Pins as Alternate-Function Pins**.

        **2.** n = 0, 1

### 16.4.2 Operation in real-time output mode

D/A conversion is performed using the interrupt request signals (INTTAA2CC0 and INTTAA3CC0) of TAA2 and TAA3 as triggers.

The setting method is described below.

<1> Set the DA0M.DA0MDn bit to 1 (real-time output mode).

<2> Set the analog voltage value to be output to the ANOn pin to the DA0CSn register.

<3> Set the DA0M.DA0CEn bit to 1 (D/A conversion enable).

      Steps <1> to <3> above constitute the initial settings.

<4> Operate TAA2 and TAA3.

<5> D/A conversion starts when the INTTAA2CC0 and INTTAA3CC0 signals are generated.

<6> After that, the value set in DA0CSn register is output every time the INTTAA2CC0 and INTTAA3CC0 signals are generated.

**Remarks 1.** The output values of the ANO0 and ANO1 pins up to <5> above are undefined.

        **2.** For the output values of the ANO0 and ANO1 pins in the HALT, IDLE1, IDLE2, and STOP modes, see **CHAPTER 21 STANDBY FUNCTION**.

        **3.** For the alternate-function pin settings, see **Table 4-20 Using Port Pins as Alternate-Function Pins**.

### 16.4.3  Cautions

Observe the following cautions when using the D/A converter of the V850ES/JG3-U and V850ES/JH3-U.

(1)  Do not change the set value of the DA0CSn register while the trigger signal is being issued in the real-time output mode.

(2)  Before changing the operation mode, be sure to clear the DA0M.DA0CEn bit to 0.

(3)  When using one of the P10/AN00 and P11/AN01 pins as an I/O port and the other as a D/A output pin, do so in an application where the port I/O level does not change during D/A output.

(4)  Make sure that $AV_{REF0} = V_{DD} = AV_{REF1} = 3.0$ to 3.6 V.  If this range is exceeded, the operation is not guaranteed.

(5)  Apply power to $AV_{REF1}$ at the same timing as $AV_{REF0}$.

(6)  No current can be output from the ANOn pin (n = 0, 1) because the output impedance of the D/A converter is high. When connecting a resistor of 2 MΩ or less, insert a JFET input operational amplifier between the resistor and the ANOn pin.

**Figure 16-2.  External Pin Connection Example**
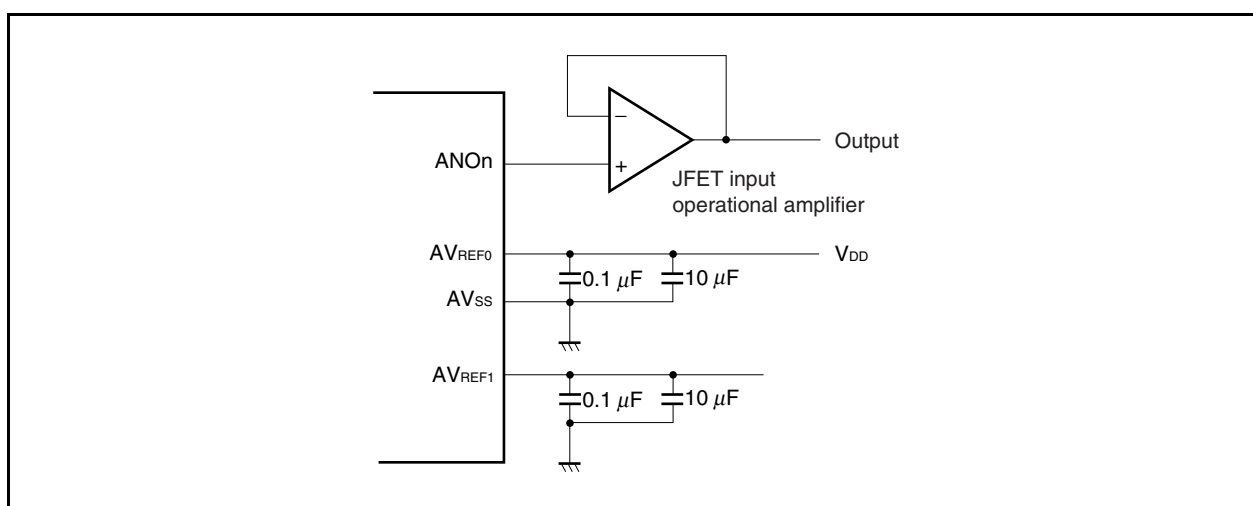


(7)  Because the D/A converter stops operating in the STOP mode, the ANO0 and ANO1 pins go into a high-impedance state, and the power consumption can be reduced.
In the IDLE1, IDLE2, or subclock operation mode, however, operation continues.  To lower the power consumption, therefore, clear the DA0M.DA0CEn bit to 0.

# CHAPTER 17  ASYNCHRONOUS SERIAL INTERFACE C (UARTC)

The V850ES/JG3-U and V850ES/JH3-U have a 5-channel UARTC.

## 17.1  Features

○ Transfer rate: 300 bps to 3 Mbps (using internal system clock of 24 MHz and dedicated baud rate generator)

○ Full-duplex communication:  Internal UARTCn receive data register (UCnRX)

                                   Internal UARTCn transmit data register (UCnTX)

○ 2-pin configuration:         TXDCn: Transmit data output pin

                                   RXDCn: Receive data input pin

○ Reception error detection function
- Parity error
- Framing error
- Overrun error

○ Interrupt sources: 2 types
- Reception completion interrupt (INTUCnR):  This interrupt occurs upon transfer of receive data from the receive shift register to the receive data register after serial transfer is complete, in the reception enabled status.
- Transmission enable interrupt (INTUCnT):  This interrupt occurs upon transfer of transmit data from the transmit data register to the transmit shift register in the transmission enabled status.

○ Character length: 7 to 9 bits

○ Parity function: Odd, even, 0, none

○ Transmission stop bit: 1, 2 bits

○ On-chip dedicated baud rate generator

○ MSB-/LSB-first transfer selectable

○ Transmit/receive data inverted input/output possible

○ SBF (Sync Break Field) transmission in the LIN (Local Interconnect Network) communication format
- 13 to 20 bits selectable for the SBF transmission
- Recognition of 11 bits or more possible for SBF reception
- SBF reception flag provided

**Remark**    n = 0 to 4

## 17.2 Configuration

The block diagram of the UARTCn is shown below.

**Figure 17-1. Block Diagram of Asynchronous Serial Interface Cn**



**Note** UARTC0 only

**Remarks 1.** n = 0 to 4
       **2.** For the configuration of the baud rate generator, see **Figure 17-19**.
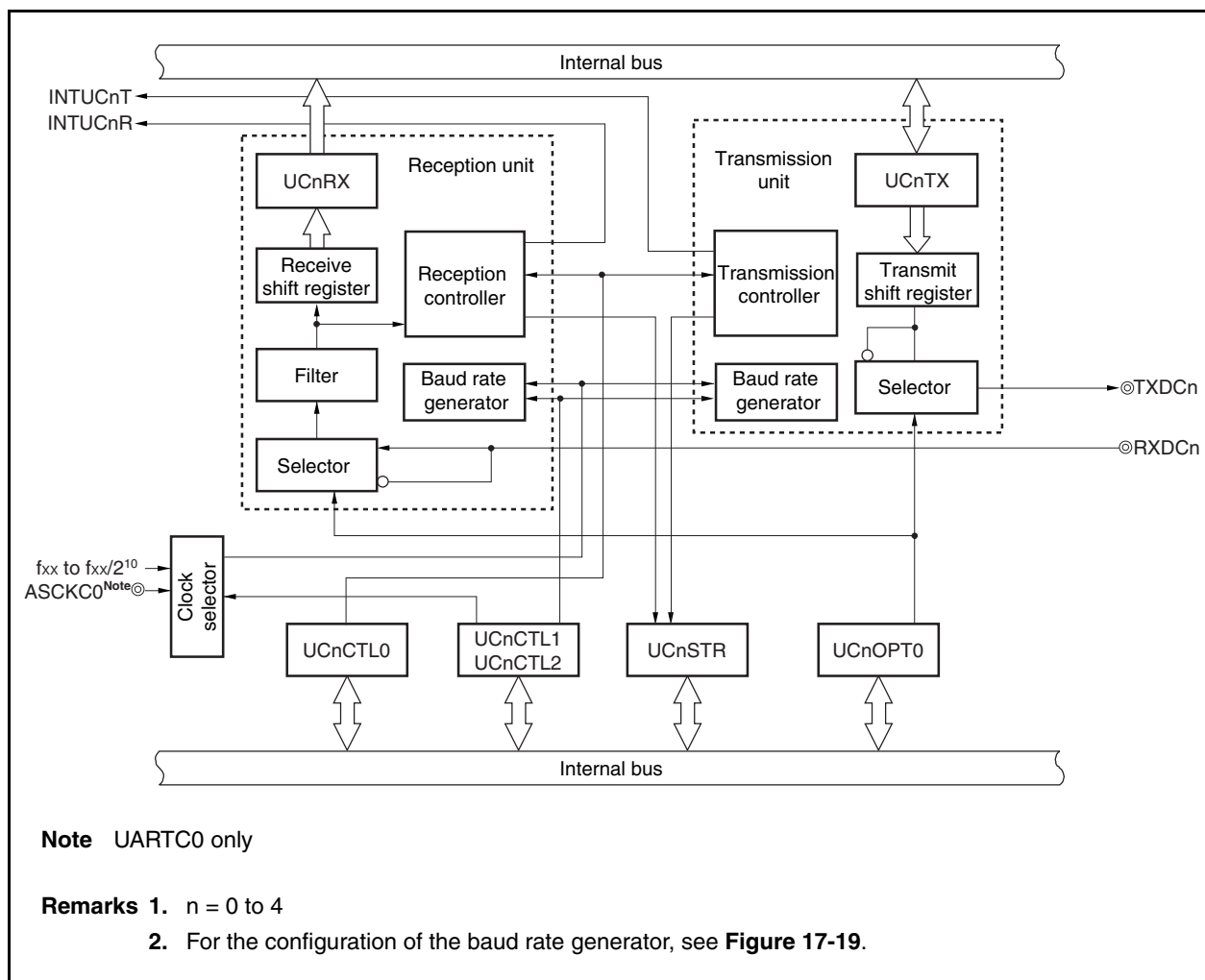
UARTCn includes the following hardware.

**Table 17-1. Configuration of UARTCn**

| Item | Configuration |
|---|---|
| Registers | UARTCn control register 0 (UCnCTL0)<br>UARTCn control register 1 (UCnCTL1)<br>UARTCn control register 2 (UCnCTL2)<br>UARTCn option control register 0 (UCnOPT0)<br>UARTCn option control register 1 (UCnOPT1)<br>UARTCn status register (UCnSTR)<br>UARTCn receive shift register<br>UARTCn receive data register (UCnRX)<br>UARTCn transmit shift register<br>UARTCn transmit data register (UCnTX) |

**(1) UARTCn control register 0 (UCnCTL0)**

The UCnCTL0 register is an 8-bit register used to specify the UARTCn operation.

**(2) UARTCn control register 1 (UCnCTL1)**

The UCnCTL1 register is an 8-bit register used to select the input clock for the UARTCn.

**(3) UARTCn control register 2 (UCnCTL2)**

The UCnCTL2 register is an 8-bit register used to control the baud rate for the UARTCn.

**(4) UARTCn option control register 0 (UCnOPT0)**

The UCnOPT0 register is an 8-bit register used to control serial transfer for the UARTCn.

**(5) UARTCn option control register 1 (UCnOPT1)**

The UCnOPT1 register is an 8-bit register used to control 9-bit length serial transfer for the UARTCn.

**(6) UARTCn status register (UCnSTR)**

The UCnSTRn register consists of flags indicating the error contents when a reception error occurs.  Each one of the reception error flags is set (to 1) upon occurrence of a reception error.

**(7) UARTCn receive shift register**

This is a shift register used to convert the serial data input to the RXDCn pin into parallel data.  Upon reception of 1 byte of data and detection of the stop bit, the receive data is transferred to the UCnRX register.

This register cannot be manipulated directly.

**(8) UARTCn receive data register (UCnRX)**

The UCnRX register is an 8-bit register that holds receive data.  When 7 characters are received, 0 is stored in the most significant bit (when data is received with the LSB first).

In the reception enabled status, receive data is transferred from the UARTCn receive shift register to the UCnRX register in synchronization with the completion of shift-in processing of 1 frame.

Transfer to the UCnRX register also causes the reception completion interrupt request signal (INTUCnR) to be output.

**(9) UARTCn transmit shift register**

The transmit shift register is a shift register used to convert the parallel data transferred from the UCnTX register into serial data.

When 1 byte of data is transferred from the UCnTX register, the shift register data is output from the TXDCn pin.

This register cannot be manipulated directly.

**(10) UARTCn transmit data register (UCnTX)**

The UCnTX register is an 8-bit transmit data buffer.  Transmission starts when transmit data is written to the UCnTX register.  When data can be written to the UCnTX register (when data of one frame is transferred from the UCnTX register to the UARTCn transmit shift register), the transmission enable interrupt request signal (INTUCnT) is generated.

## 17.3 Mode Switching Between UARTC and Other Serial Interfaces

### 17.3.1 Mode switching between UARTC0 and CSIF4

In the V850ES/JG3-U and V850ES/JH3-U, CSIF4 and UARTC0 share the same pins and therefore cannot be used simultaneously. Set UARTC0 in advance, using the PMC3 and PFC3 registers, before use.

**Caution** **The transmit/receive operation of CSIF4 and UARTC0 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.**

**Figure 17-2. CSIF4 and UARTC0 Mode Switch Settings**

After reset: 00H   R/W   Address: FFFFF446H

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC3 | PMC37 | PMC36 | PMC35 | PMC34 | PMC33 | PMC32 | PMC31 | PMC30 |

After reset: 00H   R/W   Address: FFFFF466H

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC3 | PFC37 | PFC36 | PFC35 | PFC34 | PFC33 | PFC32 | PFC31 | PFC30 |

After reset: 00H   R/W   Address: FFFFF706H

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE3 | PFCE37 | PFCE36 | PFCE35 | PFCE34 | PFCE33 | PFCE32 | PFCE31 | PFCE30 |

| PMC32 | PFCE32 | PFC32 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | ASCKC0 (UARTC0) |
| 1 | 0 | 1 | $\overline{\text{SCKF4}}$ (CSIF4) |

| PMC31 | PFCE31 | PFC31 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | RXDC0 (UARTC) |
| 1 | 0 | 1 | SIF4 (CSIF4) |

| PMC30 | PFCE30 | PFC30 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | TXDC0 (UARTC) |
| 1 | 0 | 1 | SOF4 (CSIF4) |

**Remark** × = don't care

### 17.3.2  Mode switching between UARTC1 and I²C02

In the V850ES/JG3-U and V850ES/JH3-U, UARTC1 and I²C02 share the same pins and therefore cannot be used simultaneously.  Set UARTC1 in advance, using the PMC9, PFC9 and PFCE9 registers, before use.

**Caution   The transmit/receive operation of UARTC1 and I²C02 is not guaranteed if these functions are switched during transmission or reception.  Be sure to disable the one that is not used.**

**Figure 17-3.  UARTC1 and I²C02 Mode Switch Settings**



| PMC91 | PFCE91 | PFC91 | Operation mode |
|-------|--------|-------|----------------|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 1 | TXDC1 (UARTC1) |
| 1 | 1 | 0 | SDA02 (I²C02) |

| PMC90 | PFCE90 | PFC90 | Operation mode |
|-------|--------|-------|----------------|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 1 | RXDC1 (UARTC1) |
| 1 | 1 | 0 | SCL02 (I²C02) |

**Remark**   × = don't care

### 17.3.3 Mode switching between UARTC2 and CSIF3

In the V850ES/JG3-U and V850ES/JH3-U, UARTC2 and CSIF3 share of the same pin and therefore cannot be used simultaneously. Set UARTC2 in advance, using the PMC9, PFC9 and PFCE9 registers, before use.

**Caution  The transmit/receive operation of UARTC2 and CSIF3 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.**

**Figure 17-4.  UARTC2 and CSIF3 Mode Switch Settings**

After reset:  0000H    R/W     Address:  FFFFF452H, FFFFF453H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PMC9 | PMC915 | PMC914 | PMC913 | PMC912 | PMC911 | PMC910 | PMC99 | PMC98 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PMC97 | PMC96 | PMC95 | PMC94 | PMC93 | PMC92 | PMC91 | PMC90 |

After reset:  0000H    R/W     Address:  FFFFF472H, FFFFF473H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFC9 | PFC915 | PFC914 | PFC913 | PFC912 | PFC911 | PFC910 | PFC99 | PFC98 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PFC97 | PFC96 | PFC95 | PFC94 | PFC93 | PFC92 | PFC91 | PFC90 |

After reset:  0000H    R/W     Address:  FFFFF712H, FFFFF713H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFCE9 | PFCE915 | PFCE914 | 0 | 0 | PFCE911 | PFCE910 | PFCE99 | PFCE98 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PFCE97 | PFCE96 | PFCE95 | PFCE94 | PFCE93 | PFCE92 | PFCE91 | PFCE90 |

| PMC91 | PFCE91 | PFC91 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 1 | TXDC1 (UARTC1) |
| 1 | 1 | 0 | SDA02 ($I^2$C02) |

| PMC90 | PFCE90 | PFC90 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 1 | RXDC1 (UARTC1) |
| 1 | 1 | 0 | SCL02 ($I^2$C02) |

**Remark**  × = don't care

### 17.3.4  Mode switching between UARTC3 and I²C00

In the V850ES/JG3-U and V850ES/JH3-U, UARTC3 and I²C00 share the same pins and therefore cannot be used simultaneously.  Set UARTC3 in advance, using the PMC3, PFC3 and PFCE3 registers, before use.

**Caution  The transmit/receive operation of UARTC3 and I²C00 is not guaranteed if these functions are switched during transmission or reception.  Be sure to disable the one that is not used.**

**Figure 17-5.  UARTC3 and I²C00 Mode Switch Settings**

After reset:  00H    R/W    Address:  FFFFF446H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC3 | PMC37 | PMC36 | PMC35 | PMC34 | PMC33 | PMC32 | PMC31 | PMC30 |

After reset:  00H    R/W    Address:  FFFFF466H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC3 | PFC37 | PFC36 | PFC35 | PFC34 | PFC33 | PFC32 | PFC31 | PFC30 |

After reset:  00H    R/W    Address:  FFFFF706H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE3 | PFCE37 | PFCE36 | PFCE35 | PFCE34 | PFCE33 | PFCE32 | PFCE31 | PFCE30 |

| PMC37 | PFCE37 | PFC37 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | RXDC3 (UARTC3) |
| 1 | 0 | 1 | SDA00 (I²C00) |

| PMC36 | PFCE36 | PFC36 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | TXDC3 (UARTC3) |
| 1 | 0 | 1 | SCL00 (I²C00) |

**Remark**  × = don't care

### 17.3.5  Mode switching between UARTC4, CSIF0, and I²C01

In the V850ES/JG3-U and V850ES/JH3-U, UARTC4, CSIF0, and I²C01 share the same pin and therefore cannot be used simultaneously.  Set UARTC4 in advance, using the PMC4, PFC4, and PMCE4 registers, before use.

**Caution   The transmit/receive operation of UARTC4, CSIF0, and I²C01 is not guaranteed if these functions are switched during transmission or reception.  Be sure to disable the one that is not used.**

**Figure 17-6.  UARTC4, CSIF0 and I²C01 Mode Switch Settings**

After reset:  00H     R/W     Address:  FFFFF448H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC4 | 0 | 0 | 0 | 0 | 0 | PMC42 | PMC41 | PMC40 |

After reset:  00H     R/W     Address:  FFFFF468H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC4 | 0 | 0 | 0 | 0 | 0 | PFC42 | PFC41 | PFC40 |

After reset:  00H     R/W     Address:  FFFFF708H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE4 | 0 | 0 | 0 | 0 | 0 | 0 | PFCE41 | PFCE40 |

| PMC41 | PFCE41 | PFC41 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | SOF0 (CSIF0) |
| 1 | 0 | 1 | RXDC4 (UARTC4) |
| 1 | 1 | 0 | SCL01 (I²C01) |

| PMC40 | PFCE40 | PFC40 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | SIF0 (CSIF0) |
| 1 | 0 | 1 | TXDC4 (UARTC4) |
| 1 | 1 | 0 | SDA01 (I²C01) |

**Remark**   × = don't care

## 17.4 Registers

### (1) UARTCn control register 0 (UCnCTL0)

The UCnCTL0 register is an 8-bit register that controls the UARTCn serial transfer operation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 10H.

(1/2)

After reset: 10H    R/W    Address: UC0CTL0 FFFFFA00H, UC1CTL0 FFFFFA10H,

UC2CTL0 FFFFFA20H, UC3CTL0 FFFFFA30H,

UC4CTL0 FFFFFA40H

| | <7> | <6> | <5> | <4> | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UCnCTL0 | UCnPWR | UCnTXE | UCnRXE | UCnDIR | UCnPS1 | UCnPS0 | UCnCL | UCnSL |

(n = 0 to 4)

| UCnPWR | UCRTCn operation control |
|---|---|
| 0 | Disable UCRTCn operation (UCRTCn reset asynchronously) |
| 1 | Enable UCRTCn operation |

The UARTCn operation is controlled by the UCnPWR bit. The TXDCn pin output is fixed to high level by clearing the UCnPWR bit to 0 (fixed to low level if UCnOPT0.UCnTDL bit = 1).

| UCnTXE | Transmission operation enable |
|---|---|
| 0 | Disable transmission operation |
| 1 | Enable transmission operation |

- To start transmission, set the UCnPWR bit to 1 and then set the UCnTXE bit to 1.
  To stop transmission, clear the UCnTXE bit to 0 and then UCnPWR bit to 0.
- To initialize the transmission unit, clear the UCnTXE bit to 0, wait for two cycles of the base clock, and then set the UCnTXE bit to 1 again. Otherwise, initialization may not be executed (for the base clock, see **17.7 (1) (a) Base clock**).

| UCnRXE | Reception operation enable |
|---|---|
| 0 | Disable reception operation |
| 1 | Enable reception operation |

- To start reception, set the UCnPWR bit to 1 and then set the UCnRXE bit to 1.
  To stop reception, clear the UCnRXE bit to 0 and then UCnPWR bit to 0.
- To initialize the reception unit, clear the UCnRXE bit to 0, wait for two periods of the base clock, and then set the UCnRXE bit to 1 again. Otherwise, initialization may not be executed (for the base clock, see **17.7 (1) (a) Base clock**).

| UCnDIR | Transfer direction selection |
|---|---|
| 0 | MSB-first transfer |
| 1 | LSB-first transfer |

- This register can be rewritten only when the UCnPWR bit = 0 or the UCnTXE bit = the UCnRXE bit = 0.
- When transmission and reception are performed in the LIN format, set the UCnDIR bit to 1.

| UCnPS1 | UCnPS0 | Parity selection during transmission | Parity selection during reception |
|---|---|---|---|
| 0 | 0 | No parity output | Reception with no parity |
| 0 | 1 | 0 parity output | Reception with 0 parity |
| 1 | 0 | Odd parity output | Odd parity check |
| 1 | 1 | Even parity output | Even parity check |

- This register is rewritten only when the UCnPWR bit = 0 or the UCnTXE bit = the UCnRXE bit = 0.
- If "Reception with 0 parity" is selected during reception, a parity check is not performed. Therefore, the UCnSTR.UCnPE bit is not set.
- When transmission and reception are performed in the LIN format, clear the UCnPS1 and UCnPS0 bits to 00.

| UCnCL | Specification of data character length of 1 frame of transmit/receive data |
|---|---|
| 0 | 7 bits |
| 1 | 8 bits |

- This register can be rewritten only when the UCnPWR bit = 0 or the UCnTXE bit = the UCnRXE bit = 0.
- When transmission and reception are performed in the LIN format, set the UCnCL bit to 1.

| UCnSL | Specification of length of stop bit for transmit data |
|---|---|
| 0 | 1 bit |
| 1 | 2 bits |

This register can be rewritten only when the UCnPWR bit = 0 or the UCnTXE bit = the UCnRXE bit = 0.

**Remark**  For details of parity, see **17.6.9  Parity types and operations**.


**(2)  UARTCn control register 1 (UCnCTL1)**

For details, see **17.7 (2)  UARTCn control register 1 (UCnCTL1)**.


**(3)  UARTCn control register 2 (UCnCTL2)**

For details, see **17.7 (3)  UARTCn control register 2 (UCnCTL2)**.

**(4)  UARTCn option control register 0 (UCnOPT0)**

The UCnOPT0 register is an 8-bit register that controls the serial transfer operation of the UARTCn register.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 14H.

(1/2)

After reset: 14H     R/W     Address:  UC0OPT0 FFFFFA03H, UC1OPT0 FFFFFA13H,

UC2OPT0 FFFFFA23H, UC3OPT0 FFFFFA33H,

UC4OPT0 FFFFFA43H

| | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UCnOPT0 | UCnSRF | UCnSRT | UCnSTT | UCnSLS2 | UCnSLS1 | UCnSLS0 | UCnTDL | UCnRDL |

(n = 0 to 4)

| UCnSRF | SBF reception flag |
|---|---|
| 0 | When the UCnCTL0.UCnPWR bit = UCnCTL0.UCnRXE bit = 0 are set, or upon normal end of SBF reception. |
| 1 | During SBF reception |

• SBF (Sync Brake Field) reception is judged during LIN communication.
• The UCnSRF bit is held at 1 when an SBF reception error occurs, and then SBF reception is started again.
• The UCnSRF bit is a read-only bit.

| UCnSRT | SBF reception trigger |
|---|---|
| 0 | − |
| 1 | SBF reception trigger |

• This is the SBF reception trigger bit during LIN communication, and when read, "0" is always read.  For SBF reception, set the UCnSRT bit (to 1) to enable SBF reception.
• Set the UCnSRT bit after setting the UCnPWR bit = UCnRXE bit = 1.

| UCnSTT | SBF transmission trigger |
|---|---|
| 0 | − |
| 1 | SBF transmission trigger |

• This is the SBF transmission trigger bit during LIN communication, and when read, "0" is always read.
• Set the UCnSTT bit after setting the UCnPWR bit = UCnTXE bit = 1.

**Caution   Do not set the UCnSRT and UCnSTT bits (to 1) during SBF reception (UCnSRF bit = 1).**

(2/2)

| UCnSLS2 | UCnSLS1 | UCnSLS0 | SBF transmission length selection |
|---------|---------|---------|-----------------------------------|
| 1 | 0 | 1 | 13-bit output (reset value) |
| 1 | 1 | 0 | 14-bit output |
| 1 | 1 | 1 | 15-bit output |
| 0 | 0 | 0 | 16-bit output |
| 0 | 0 | 1 | 17-bit output |
| 0 | 1 | 0 | 18-bit output |
| 0 | 1 | 1 | 19-bit output |
| 1 | 0 | 0 | 20-bit output |

This register can be set when the UCnPWR bit = 0 or when the UCnTXE bit = 0.

| UCnTDL | Transmit data level bit |
|--------|-------------------------|
| 0 | Normal output of transfer data |
| 1 | Inverted output of transfer data |

• The output level of the TXDCn pin can be inverted using the UCnTDL bit.
• This register can be set when the UCnPWR bit = 0 or when the UCnTXE bit = 0.

| UCnRDL | Receive data level bit |
|--------|------------------------|
| 0 | Normal input of transfer data |
| 1 | Inverted input of transfer data |

• The input level of the RXDCn pin can be inverted using the UCnRDL bit.
• This register can be set when the UCnPWR bit = 0 or the UCnRXE bit = 0.

**(5) UARTCn option control register 1 (UCnOPT1)**

The UCnOPT1 register is an 8-bit register that controls the serial transfer operation of UARTCn.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

**Caution    Set the UCnEBE bit while the operation of UARTC is disabled (UCnCTL0.UCnPWR = 0).**

After reset: 00H    R/W    Address: UC0OPT1 FFFFFA0AH, UC1OPT1 FFFFFA1AH,
UC2OPT1 FFFFFA2AH, UC3OPT1 FFFFFA3AH,
UC4OPT1 FFFFFA4AH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UCnOPT1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | UCnEBE |

(n = 0 to 4)

| UCnEBE | Extension bit enable/disable |
|---|---|
| 0 | Extension-bit operation is prohibited.  Transmission/reception is performed in the data length set by the UCnCTL0.UCnCL bit. |
| 1 | Extension-bit operation enabled.   Transmission/reception can be performed in 9-bit character length. |
| • When setting the UCnEBE bit to 1, and transmitting in 9-bit data length, be sure to set the following.  If this setting is not performed, the setting of UCnEBE bit is invalid.<br>  • UCnCTL0.UCnPS1, UCnPS0 = 00 (no parity)<br>  • CnCTL0.UCnCL = 1 (8-bit character length)<br>• If transmitting or receiving in the LIN communication format, set the UCnEBE to 0. | |

The following shows the relationship between the register setting value and the data format.

**Table 17-2. Relationship Between Register Setting and Data Format**

| Register Setting | | | | | Data Format | | | | |
|---|---|---|---|---|---|---|---|---|---|
| UCnCTL0 | | | | UCnOPT1 | D0 to D6 | D7 | D8 | D9 | D10 |
| UCnCL | UCnPS1 | UCnPS0 | UCnSL | UCnEBE | | | | | |
| 0 | 0 | 0 | 0 | 0 | Data | Stop | – | – | – |
| 0 | Other than 00 | | | | Data | Parity | Stop | – | – |
| 1 | 0 | 0 | | | Data | Data | Stop | – | – |
| 1 | Other than 00 | | | | Data | Data | Parity | Stop | – |
| 0 | 0 | 0 | 1 | 0 | Data | Stop | Stop | – | – |
| 0 | Other than 00 | | | | Data | Parity | Stop | Stop | – |
| 1 | 0 | 0 | | | Data | Data | Stop | Stop | – |
| 1 | Other than 00 | | | | Data | Data | Parity | Stop | Stop |
| 0 | 0 | 0 | 0 | 1 | Data | Stop | – | – | – |
| 0 | Other than 00 | | | | Data | Parity | Stop | – | – |
| 1 | 0 | 0 | | | Data | Data | Data | Stop | – |
| 1 | Other than 00 | | | | Data | Data | Parity | Stop | – |
| 0 | 0 | 0 | 1 | 1 | Data | Stop | Stop | – | – |
| 0 | Other than 00 | | | | Data | Parity | Stop | Stop | – |
| 1 | 0 | 0 | | | Data | Data | Data | Stop | Stop |
| 1 | Other than 00 | | | | Data | Data | Parity | Stop | Stop |

**Remark** Data: Data bit
     Stop: Stop bit
     Parity: Parity bit

**(6) UARTCn status register (UCnSTR)**

The UCnSTR register is an 8-bit register that displays the UARTCn transfer status and reception error contents.

This register can be read or written in 8-bit or 1-bit units, but the UCnTSF bit is a read-only bit, while the UCnPE, UCnFE, and UCnOVE bits can both be read and written. However, these bits can only be cleared by writing 0; they cannot be set by writing 1 (even if 1 is written to them, the value is retained).

The initialization conditions are shown below.

| Register/Bit | Initialization Conditions |
|---|---|
| UCnSTR register | • Reset<br>• UCnCTL0.UCnPWR = 0 |
| UCnTSF bit | • UCnCTL0.UCnTXE = 0 |
| UCnPE, UCnFE, UCnOVE bits | • 0 write<br>• UCnCTL0.UCnRXE = 0 |

After reset: 00H    R/W    Address: UC0STR FFFFFA04H, UC1STR FFFFFA14H,

UC2STR FFFFFA24H, UC3STR FFFFFA34H,

UC4STR FFFFFA44H

| | \<7\> | 6 | 5 | 4 | 3 | \<2\> | \<1\> | \<0\> |
|---|---|---|---|---|---|---|---|---|
| UCnSTR | UCnTSF | 0 | 0 | 0 | 0 | UCnPE | UCnFE | UCnOVE |

(n = 0 to 4)

| UCnTSF | Transfer status flag |
|---|---|
| 0 | • When the UCnPWR bit = 0 or the UCnTXE bit = 0 has been set. <br> • When, following transfer completion, there was no next data transfer from UCnTX register |
| 1 | Write to UCnTX register |

The UCnTSF bit is always 1 when performing continuous transmission. When initializing the transmission unit, check that the UCnTSF bit = 0 before performing initialization. The transmit data is not guaranteed when initialization is performed while the UCnTSF bit = 1.

| UCnPE | Parity error flag |
|---|---|
| 0 | • When the UCnPWR bit = 0 or the UCnRXE bit = 0 has been set. <br> • When 0 has been written |
| 1 | When parity of data and parity bit do not match during reception. |

• The operation of the UCnPE bit is controlled by the settings of the UCnCTL0.UCnPS1 and UCnCTL0.UCnPS0 bits.
• The UCnPE bit can be read and written, but it can only be cleared by writing 0 to it, and it cannot be set by writing 1 to it. When 1 is written to this bit, the value is retained.

| UCnFE | Framing error flag |
|---|---|
| 0 | • When the UCnPWR bit = 0 or the UCnRXE bit = 0 has been set <br> • When 0 has been written |
| 1 | When no stop bit is detected during reception |

• Only the first bit of the receive data stop bits is checked, regardless of the value of the UCnCTL0.UCnSL bit.
• The UCnFE bit can be both read and written, but it can only be cleared by writing 0 to it, and it cannot be set by writing 1 to it. When 1 is written to this bit, the value is retained.

| UCnOVE | Overrun error flag |
|---|---|
| 0 | • When the UCnPWR bit = 0 or the UCnRXE bit = 0 has been set. <br> • When 0 has been written |
| 1 | When receive data has been set to the UCnRX register and the next receive operation is completed before that receive data has been read |

• When an overrun error occurs, the data is discarded without the next receive data being written to the receive buffer.
• The UCnOVE bit can be both read and written, but it can only be cleared by writing 0 to it. When 1 is written to this bit, the value is retained.

**(7) UARTCn receive data register L (UCnRX) and UARTCn receive data register (UCnRX)**

The UCnRXL and UCnRX register are an 8- bit or 9-bit buffer register that stores parallel data converted by the receive shift register.

The data stored in the receive shift register is transferred to the UCnRXL and UCnRX register upon completion of reception of 1 byte of data.

During LSB-first reception when the data length has been specified as 7 bits, the receive data is transferred to bits 6 to 0 of the UCnRXL register and the MSB always becomes 0.  During MSB-first reception, the receive data is transferred to bits 7 to 1 of the UCnRXL register and the LSB always becomes 0.

When an overrun error (UCnOVE) occurs, the receive data at this time is not transferred to the UCnRXL and UCnRX register and is discarded.

The access unit or reset value differs depending on the character length.

- Character length 7/8-bit (UCnOPT1.UCnEBE = 0)
    This register is read-only, in 8-bit units.
    Reset or UCnCTL0.UCnPWR bit = 0 sets this register to FFH.
- Character length 9-bit (UCnOPT1.UCnEBE = 0)
    This register is read-only, in 16-bit units.
    Reset or UCnCTL0.UCnPWR bit = 0 sets this register to 01FFH.

---

**(a) Character length 7/8-bit (UCnOPT1.UCnEBE = 0)**

After reset: FFH    R    Address:  UC0RXL FFFFFA06H, UC1RXL FFFFFA16H,
                                   UC2RXL FFFFFA26H, UC3RXL FFFFFA36H,
                                   UC4RXL FFFFFA46H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UCnRXL (n = 0 to 4) | | | | | | | | |
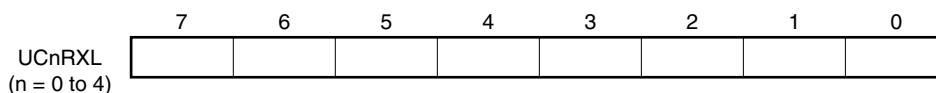
**(b) Character length 9-bit (UCnOPT1.UCnEBE = 1)**

After reset: 01FFH    R    Address:  UC0RX FFFFFA06H, UC1RX FFFFFA16H,
                                     UC2RX FFFFFA26H, UC3RX FFFFFA36H,
                                     UC4RX FFFFFA46H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UCnRX (n = 0 to 4) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | | | | | | | |

---
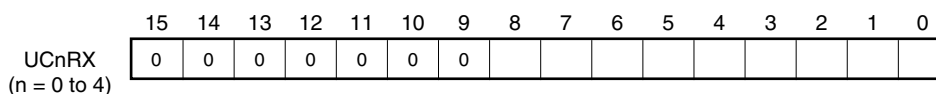
**(8) UARTCn transmit data register L (UCnTXL), UARTCn transmit data register (UCnTX)**

The UCnTXL and UCnTX register is an 8-bit or 9-bit register used to set transmit data.

During LSB-first transmission when the data length has been specified as 7 bits, the transmit data is transferred to bits 6 to 0 of the UCnRX register.  During MSB-first transmission, the receive data is transferred to bits 7 to 1 of the UCnRX register.

The access unit or reset value differs depending on the character length.

- Character length 7/8-bit (UCnOPT1.UCnEBE = 0)

    This register can be read or written in 8-bit units.

    Reset sets this register to FFH.

- Character length 9-bit (UCnOPT1.UCnEBE = 0)

    This register can be read or written in 16-bit units.

    Reset sets this register to 01FFH.

**Cautions 1.  In the transmission operation enable status (UCnPWR = 1 and UCnTXE = 1), Writing to the UCnTXL, UCnTX register, as operate as trigger of transmission star, if writing the value of as soon as before and save value, before the INTUCnT interrupt is occurred, the same data is transferred at twice.**

**2.  Data writing for consecutive transmission, after be generated the INTUCnT interrupt. If writing the next data before the INTUCnT interrupt is occurred, transmission start processing and source of conflict writing the UCnTXL, UCnTX register, unexpected operations may occur.**

**3.  If perform to write the UCnTXL, UCnTXLin the disable transmission operation register, can not be used as transmission start trigger.  Consequently, even if transmission enable status after perform to write the UCnTXL, UCnTX register in the disable transmission operation status, can not be started transmission.**

---

**(a) Character length 7/8-bit (UCnOPT1.UCnEBE = 0)**

After reset: FFH    R/W    Address:  UC0TXL FFFFFA08H, UC1TXL FFFFFA18H,
UC2TXL FFFFFA28H, UC3TXL FFFFFA38H,
UC4TXL FFFFFA48H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UCnTXL | | | | | | | | |

(n = 0 to 4)
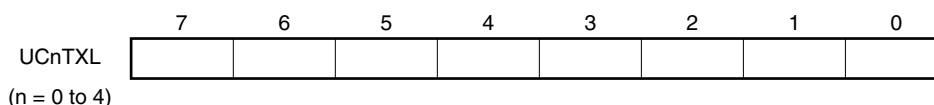
**(b) Character length 9-bit (UCnOPT1.UCnEBE = 1)**

After reset: 01FFH    R/W    Address:  UC0TX FFFFFA08H, UC1TX FFFFFA18H,
UC2TX FFFFFA28H, UC3TX FFFFFA38H,
UC4TX FFFFFA48H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UCnTX | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | | | | | | | |

(n = 0 to 4)
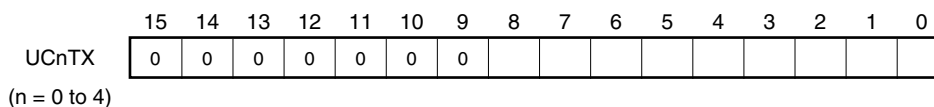
## 17.5  Interrupt Request Signals

The following two interrupt request signals are generated from UARTCn.

- Reception completion interrupt request signal (INTUCnR)
- Transmission enable interrupt request signal (INTUCnT)

The default priority for these two interrupt request signals is reception completion interrupt request signal then transmission enable interrupt request signal.

**Table 17-3.  Interrupts and Their Default Priorities**

| Interrupt | Priority |
|---|---|
| Reception complete | High |
| Transmission enable | Low |

**(1)  Reception completion interrupt request signal (INTUCnR)**

A reception completion interrupt request signal is output when data is shifted into the receive shift register and transferred to the UCnRX register in the reception enabled status.

A reception completion interrupt request signal is also output when a reception error occurs.  Therefore, when a reception completion interrupt request signal is acknowledged and the data is read, read the UCnSTR register and check that the reception result is not an error.

No reception completion interrupt request signal is generated in the reception disabled status.

**(2)  Transmission enable interrupt request signal (INTUCnT)**

If transmit data is transferred from the UCnTX register to the UARTCn transmit shift register with transmission enabled, the transmission enable interrupt request signal is generated.

## 17.6  Operation

### 17.6.1  Data format

Full-duplex serial data reception and transmission is performed.

As shown in Figure 17-7, one data frame of transmit/receive data consists of a start bit, character bits, parity bit, and stop bit(s).

Specification of the character bit length within 1 data frame, parity selection, specification of the stop bit length, and specification of MSB/LSB-first transfer are performed using the UCnCTL0 register.

Moreover, control of UART output/inverted output for the TXDCn bit is performed using the UCnOPT0.UCnTDL bit.

- Start bit..................1 bit
- Character bits ........7 bits/8 bits
- Parity bit ................Even parity/odd parity/0 parity/no parity
- Stop bit ..................1 bit/2 bits

**Figure 17-7.  UARTC Transmit/Receive Data Format**

**(a)  8-bit data length, LSB first, even parity, 1 stop bit, transfer data: 55H**

| 1 data frame |
|---|

| Start bit | D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | Parity bit | Stop bit |
|---|---|---|---|---|---|---|---|---|---|---|

**(b)  8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55H**

| 1 data frame |
|---|

| Start bit | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Parity bit | Stop bit |
|---|---|---|---|---|---|---|---|---|---|---|
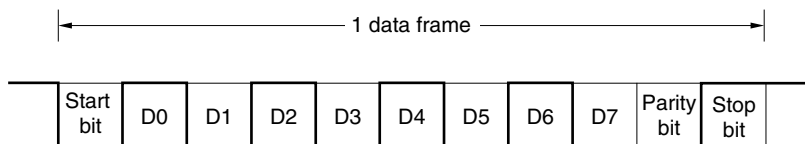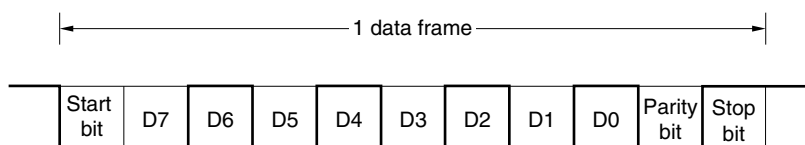
**(c)  8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55H, TXDCn inversion**

| 1 data frame |
|---|

| Start bit | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Parity bit | Stop bit |
|---|---|---|---|---|---|---|---|---|---|---|
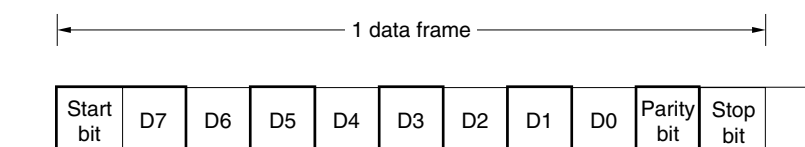
**(d)  7-bit data length, LSB first, odd parity, 2 stop bits, transfer data: 36H**

| 1 data frame |
|---|

| Start bit | D0 | D1 | D2 | D3 | D4 | D5 | D6 | Parity bit | Stop bit | Stop bit |
|---|---|---|---|---|---|---|---|---|---|---|
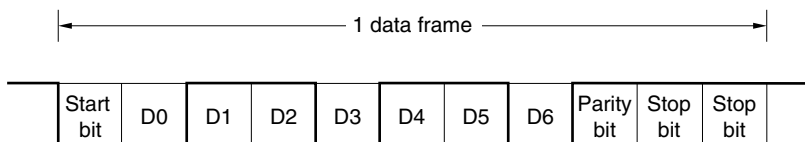
**(e)  8-bit data length, LSB first, no parity, 1 stop bit, transfer data: 87H**

| 1 data frame |
|---|

| Start bit | D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | Stop bit |
|---|---|---|---|---|---|---|---|---|---|
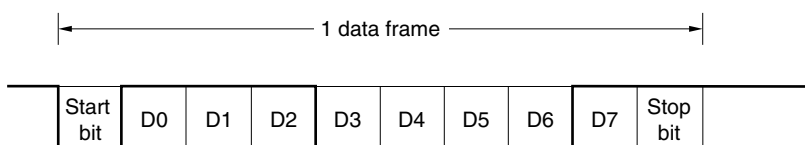
### 17.6.2  SBF transmission/reception format

The V850ES/JG3-U and V850ES/JH3-U have an SBF (Sync Break Field) transmission/reception control function to enable use of the LIN function.

**Remark**  LIN stands for Local Interconnect Network and is a low-speed (1 to 20 kbps) serial communication protocol intended to aid the cost reduction of an automotive network.

LIN communication is single-master communication, and up to 15 slaves can be connected to one master.

The LIN slaves are used to control the switches, actuators, and sensors, and these are connected to the LIN master via the LIN network.

Normally, the LIN master is connected to a network such as CAN (Controller Area Network).

In addition, the LIN bus uses a single-wire method and is connected to the nodes via a transceiver that complies with ISO9141.

In the LIN protocol, the master transmits a frame with baud rate information and the slave receives it and corrects the baud rate error.  Therefore, communication is possible when the baud rate error in the slave is ±15% or less.

Figures 17-8 and 17-9 outline the transmission and reception manipulations of LIN.

**Figure 17-8.  LIN Transmission Manipulation Outline**



**Notes 1.**  The interval between each field is controlled by software.

   **2.**  SBF output is performed by hardware.  The output width is the bit length set by the UCnOPT0.UCnSLS2 to UCnOPT0.UCnSLS0 bits.  If even finer output width adjustments are required, such adjustments can be performed using the UCnCTL2.UCnBRS7 to UCnCTL2.UCnBRS0 bits.

   **3.**  80H transfer in the 8-bit mode is substituted for the wakeup signal frame.

   **4.**  A transmission enable interrupt request signal (INTUCnT) is output at the start of each transmission. The INTUCnT signal is also output at the start of each SBF transmission.

**Figure 17-9.  LIN Reception Manipulation Outline**



Notes 1. The wakeup signal is sent by the pin edge detector, UARTCn is enabled, and the SBF reception
        mode is set.

    2. The receive operation is performed until detection of the stop bit.  Upon detection of SBF reception
       of 11 or more bits, normal SBF reception end is judged, and an interrupt signal is output.  Upon
       detection of SBF reception of less than 11 bits, an SBF reception error is judged, no interrupt signal
       is output, and the mode returns to the SBF reception mode.

    3. If SBF reception ends normally, an interrupt request signal is output.  The timer is enabled by an SBF
       reception completion interrupt.  Moreover, error detection for the UCnSTR.UCnOVE,
       UCnSTR.UCnPE, and UCnSTR.UCnFE bits is suppressed and UART communication error detection
       processing and UARTCn receive shift register and data transfer of the UCnRX register are not
       performed.  The UARTCn receive shift register holds the initial value, FFH.

    4. The RXDCn pin is connected to TI (capture input) of the timer, the transfer rate is calculated, and the
       baud rate error is calculated.  The value of the UCnCTL2 register obtained by correcting the baud
       rate error after dropping UARTC enable is set again, causing the status to become the reception
       status.

    5. Check-sum field distinctions are made by software.  UARTCn is initialized following CSF reception,
       and the processing for setting the SBF reception mode again is performed by software.

**17.6.3  SBF transmission**

When the UCnCTL0.UCnPWR bit = UCnCTL0.UCnTXE bit = 1, the transmission enabled status is entered, and SBF transmission is started by setting (to 1) the SBF transmission trigger (UCnOPT0.UCnSTT bit).

Thereafter, a low level the width of bits 13 to 20 specified by the UCnOPT0.UCnSLS2 to UCnOPT0.UCnSLS0 bits is output. A transmission enable interrupt request signal (INTUCnT) is generated upon SBF transmission start. Following the end of SBF transmission, the UCnSTT bit is automatically cleared. Thereafter, the UART transmission mode is restored.

Transmission is suspended until the data to be transmitted next is written to the UCnTX register, or until the SBF transmission trigger (UCnSTT bit) is set.

**Figure 17-10.  SBF Transmission**
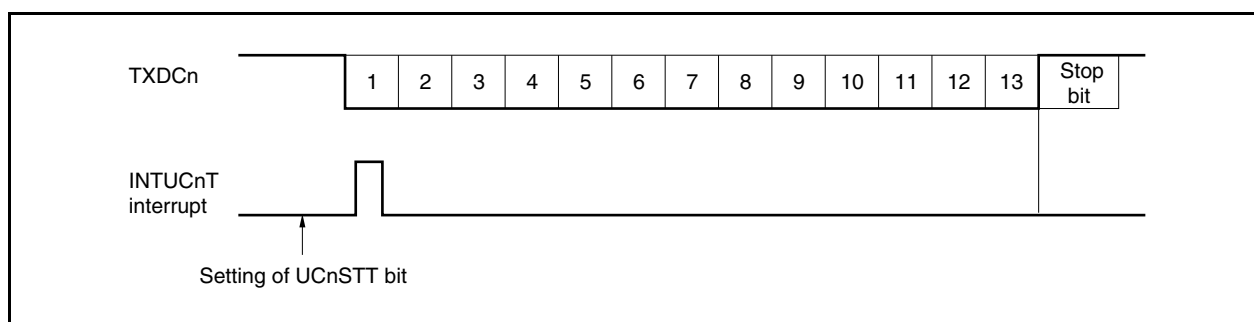
### 17.6.4  SBF reception

The reception wait status is entered by setting the UCnCTL0.UCnPWR bit to 1 and then setting the UCnCTL0.UCnRXE bit to 1.

The SBF reception wait status is set by setting the SBF reception trigger (UCnOPT0.UCnSRT bit) to 1.

In the SBF reception wait status, similarly to the UART reception wait status, the RXDCn pin is monitored and start bit detection is performed.

Following detection of the start bit, reception is started and the internal counter counts up according to the set baud rate.

When a stop bit is received, if the SBF width is 11 or more bits, normal processing is judged and a reception completion interrupt request signal (INTUCnR) is output. The UCnOPT0.UCnSRF bit is automatically cleared and SBF reception ends. Error detection for the UCnSTR.UCnOVE, UCnSTR.UCnPE, and UCnSTR.UCnFE bits is suppressed and UART communication error detection processing is not performed. Moreover, data transfer of the UARTCn reception shift register and UCnRX register is not performed and FFH, the initial value, is held. If the SBF width is 10 or fewer bits, reception is terminated as error processing without outputting an interrupt, and the SBF reception mode is returned to. The UCnSRF bit is not cleared at this time.

**Cautions 1.  If SBF is transmitted during a data reception, a framing error occurs.**

**2.  Do not set the SBF reception trigger bit (UCnSRT) and SBF transmission trigger bit (UCnSTT) to 1 during an SBF reception (UCnSRF = 1).**

**Figure 17-11.  SBF Reception**



(a)  Normal SBF reception (detection of stop bit in more than 10.5 bits)

(b)  SBF reception error (detection of stop bit in 10.5 or fewer bits)
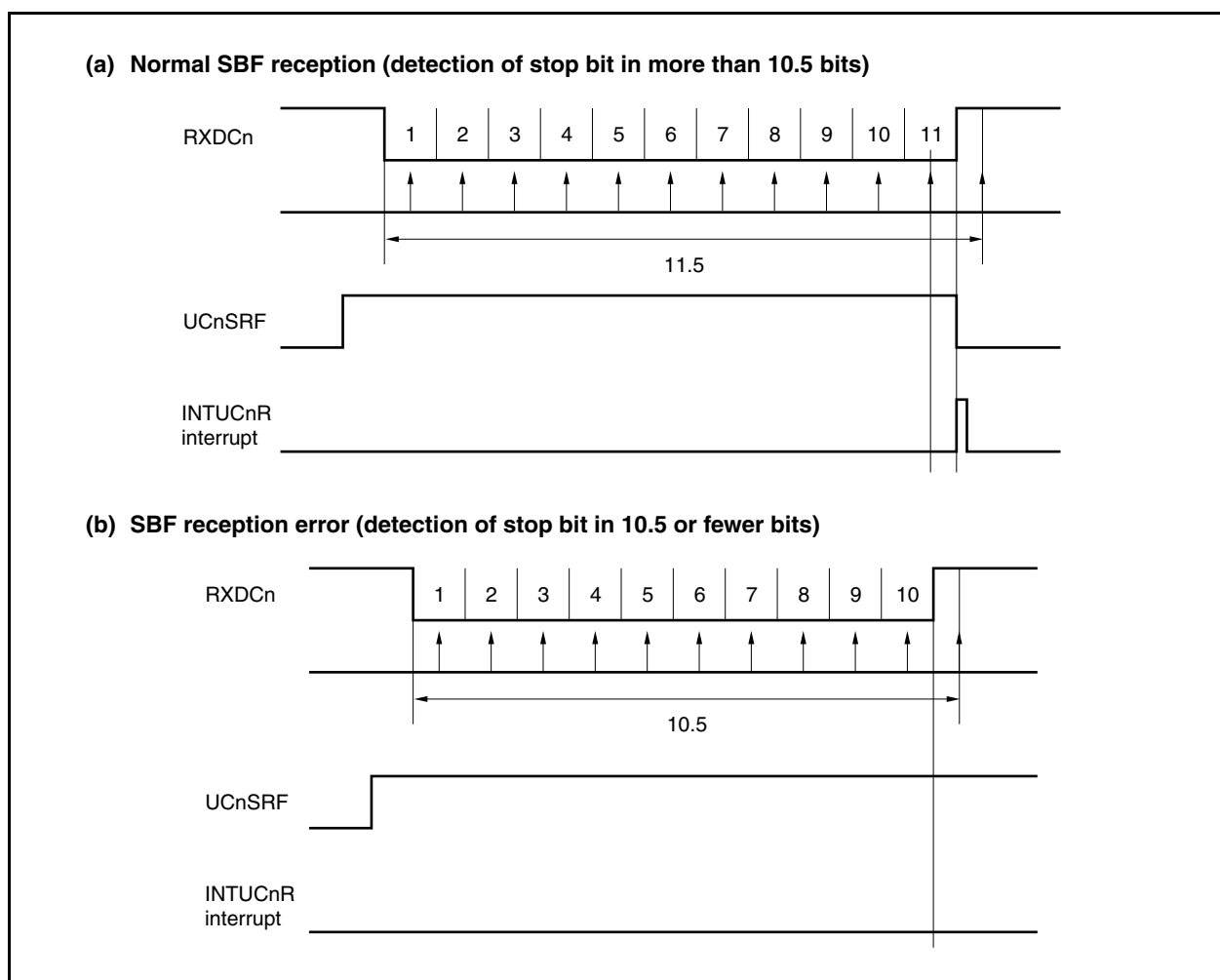
### 17.6.5 UART transmission

A high level is output to the TXDCn pin by setting the UCnCTL0.UCnPWR bit to 1.

Next, the transmission enabled status is set by setting the UCnCTL0.UCnTXE bit to 1, and transmission is started by writing transmit data to the UCnTX register. The start bit, parity bit, and stop bit are automatically added.

Since the CTS (transmit enable signal) input pin is not provided in UARTCn, use a port to check that reception is enabled at the transmit destination.

The data in the UCnTX register is transferred to the UARTCn transmit shift register upon the start of the transmit operation.

A transmission enable interrupt request signal (INTUCnT) is generated upon completion of transmission of the data of the UCnTX register to the UARTCn transmit shift register, and thereafter the contents of the UARTCn transmit shift register are output to the TXDCn pin.

Write of the next transmit data to the UCnTX register is enabled after the INTUCnT signal is generated.

#### Figure 17-12. UART Transmission



**Remark** LSB first
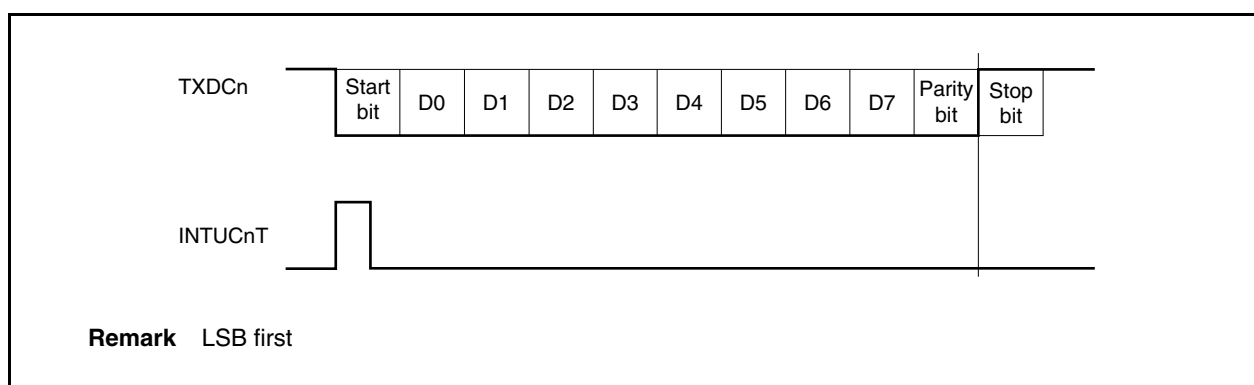
**17.6.6  Continuous transmission procedure**

UARTCn can write the next transmit data to the UCnTX register when the UARTCn transmit shift register starts the shift operation.  The transmit timing of the UARTCn transmit shift register can be judged from the transmission enable interrupt request signal (INTUCnT).  An efficient communication rate is realized by writing the data to be transmitted next to the UCnTX register during transfer.

**Caution   When initializing transmissions during the execution of continuous transmissions, make sure that the UCnSTR.UCnTSF bit is 0, then perform the initialization.  Transmit data that is initialized when the UCnTSF bit is 1 cannot be guaranteed.**

**Figure 17-13.  Continuous Transmission Processing Flow**
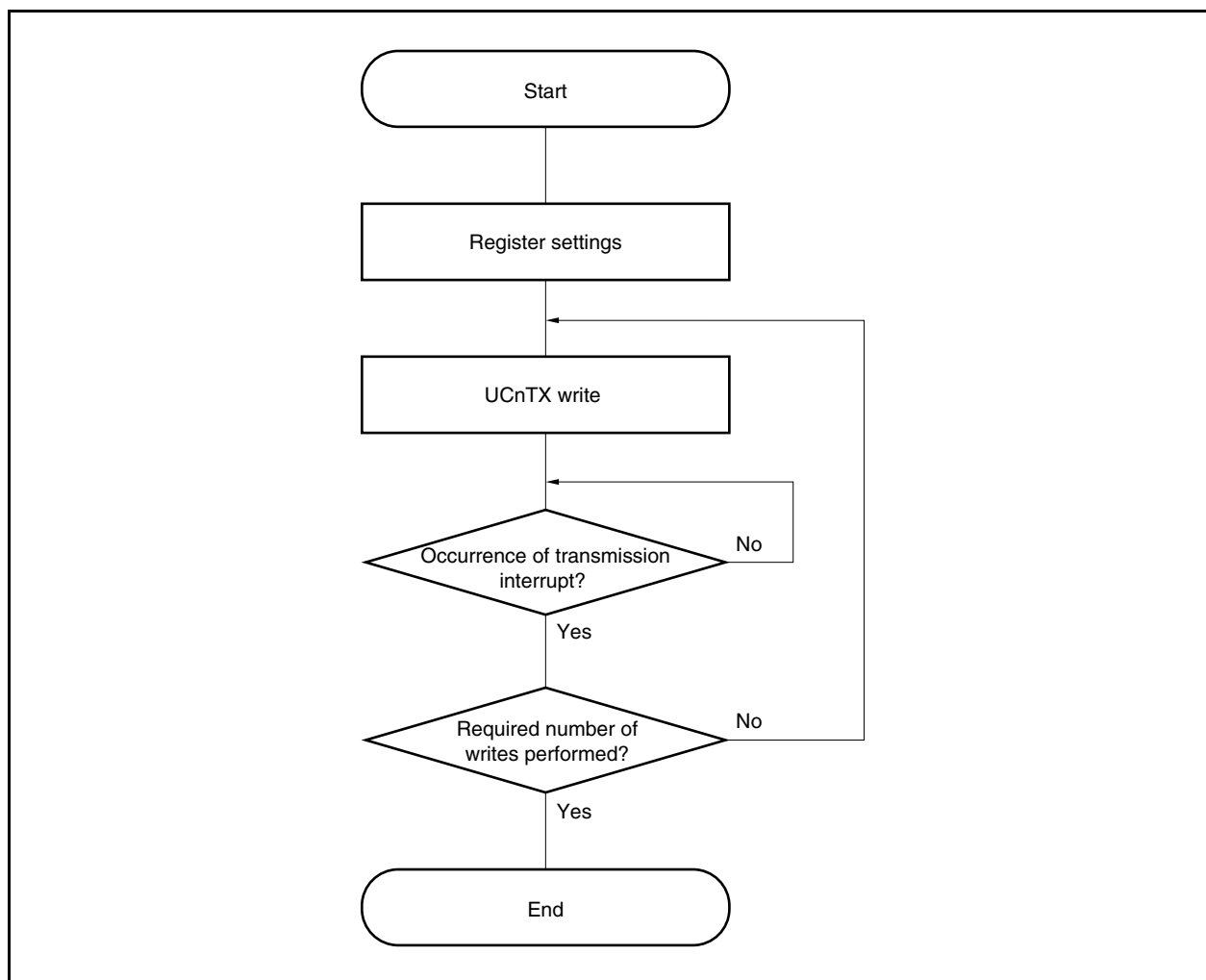
**Figure 17-14. Continuous Transmission Operation Timing**
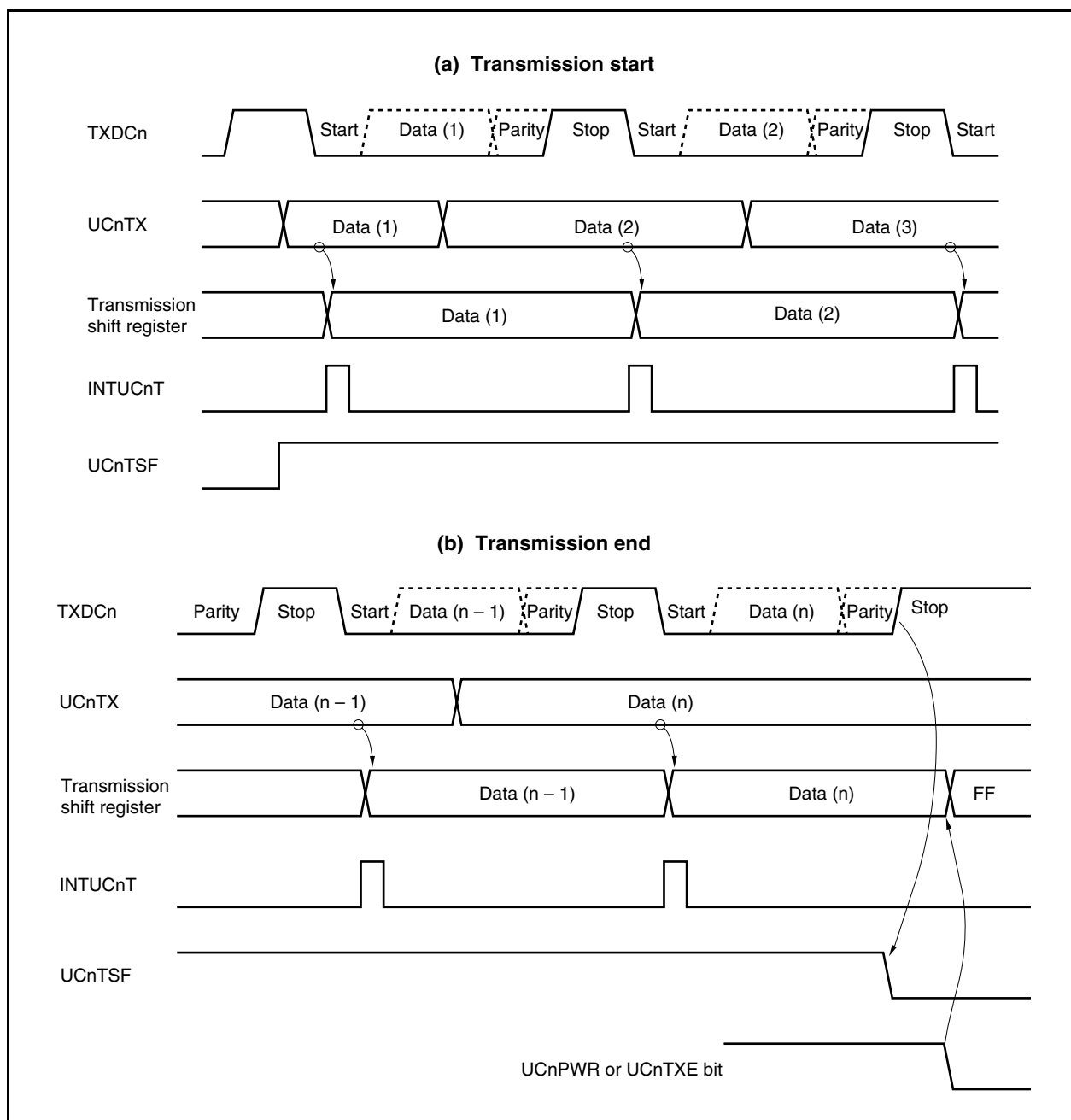
### 17.6.7  UART reception

The reception wait status is set by setting the UCnCTL0.UCnPWR bit to 1 and then setting the UCnCTL0.UCnRXE bit to 1.  In the reception wait status, the RXDCn pin is monitored and start bit detection is performed.

Start bit detection is performed using a two-step detection routine.

First the rising edge of the RXDCn pin is detected and sampling is started at the falling edge.  The start bit is recognized if the RXDCn pin is low level at the start bit sampling point.  After a start bit has been recognized, the receive operation starts, and serial data is saved to the UARTCn receive shift register according to the set baud rate.

When the reception completion interrupt request signal (INTUCnR) is output upon reception of the stop bit, the data of the UARTCn receive shift register is written to the UCnRX register.  However, if an overrun error (UCnSTR.UCnOVE bit) occurs, the receive data at this time is not written to the UCnRX register and is discarded.

Even if a parity error (UCnSTR.UCnPE bit) or a framing error (UCnSTR.UCnFE bit) occurs during reception, reception continues until the reception position of the first stop bit, and INTUCnR is output following reception completion.

**Figure 17-15.  UART Reception**
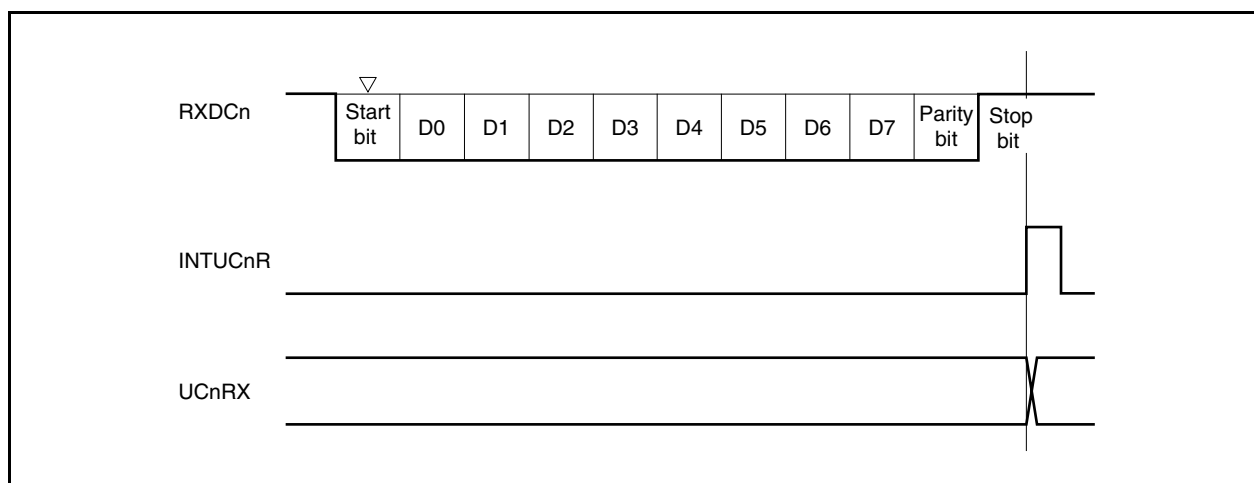


**Cautions 1.**  **Be sure to read the UCnRX register even when a reception error occurs.  If the UCnRX register is not read, an overrun error occurs during reception of the next data, and reception errors continue occurring indefinitely.**

**2.**  **The operation during reception is performed assuming that there is only one stop bit.  A second stop bit is ignored.**

**3.**  **When reception is completed, read the UCnRX register after the reception completion interrupt request signal (INTUCnR) has been generated, and clear the UCnPWR or UCnRXE bit to 0.  If the UCnPWR or UCnRXE bit is cleared to 0 before the INTUCnR signal is generated, the read value of the UCnRX register cannot be guaranteed.**

**4.**  **If receive completion processing (INTUCnR signal generation) of UARTCn and the UCnPWR bit = 0 or UCnRXE bit = 0 conflict, the INTUCnR signal may be generated in spite of these being no data stored in the UCnRX register.**

**To complete reception without waiting for the INTUCnR signal to be generated, be sure to set (1) the interrupt mask flag (UCnRMK) of the interrupt control register (UCnRIC), clear (0) the UCnPWR bit or UCnRXE bit, and then clear the interrupt request flag (UCnRIF) of the UCnRIC register.**

### 17.6.8 Reception errors

Errors during a receive operation are of three types: parity errors, framing errors, and overrun errors. Data reception result error flags are set in the UCnSTR register and a reception completion interrupt request signal (INTUCnR) is output when an error occurs.

It is possible to ascertain which error occurred during reception by reading the contents of the UCnSTR register.

Clear the reception error flag by writing 0 to it after reading it.

**Figure 17-16. Receive Data Read Flow**



**Caution    When an INTUCnR signal is generated, the UCnSTR register must be read to check for errors.**
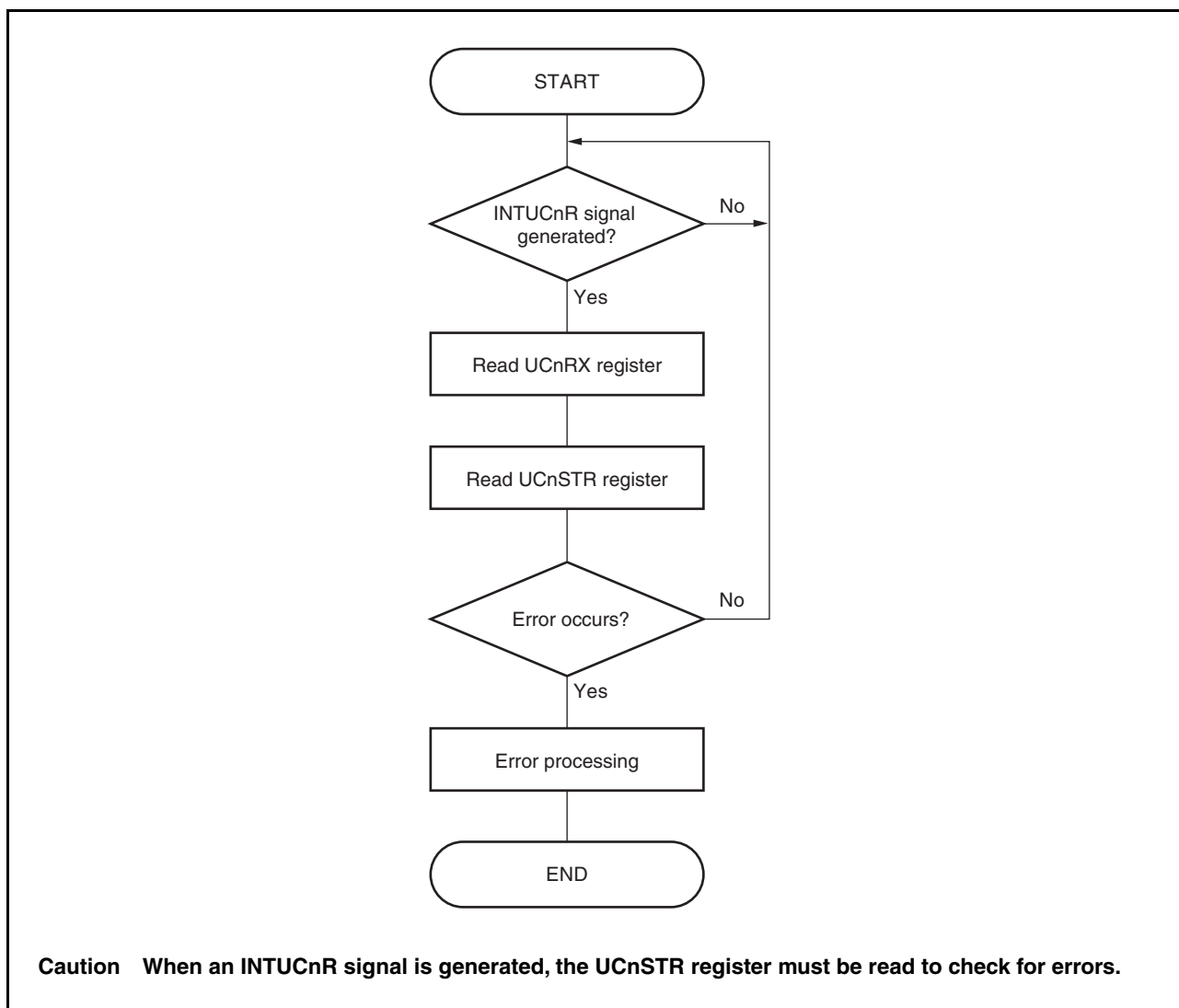
- Reception error causes

| Error Flag | Reception Error | Cause |
|---|---|---|
| UCnPE | Parity error | Received parity bit does not match the setting |
| UCnFE | Framing error | Stop bit not detected |
| UCnOVE | Overrun error | Reception of next data completed before data was read from receive buffer |

When reception errors occur, perform the following procedures depending upon the kind of error.

- Parity error

  If false data is received due to problems such as noise in the reception line, discard the received data and retransmit.

- Framing error

  A baud rate error may have occurred between the reception side and transmission side or the start bit may have been erroneously detected.  Since this is a fatal error for the communication format, check the operation stop in the transmission side, perform initialization processing each other, and then start the communication again.

- Overrun error

  Since the next reception is completed before reading receive data, 1 frame of data is discarded.  If this data was needed, do a retransmission.

**Caution   If a receive error interrupt occurs during continuous reception, read the contents of the UCnSTR register must be read before the next reception is completed, then perform error processing.**

**17.6.9 Parity types and operations**

**Caution　When using the LIN function, fix the UCnCTL0.UCnPS1 and UCnCTL0.UCnPS0 bits to 00.**

The parity bit is used to detect bit errors in the communication data.　Normally the same parity is used on the transmission side and the reception side.

In the case of even parity and odd parity, it is possible to detect odd-count bit errors.　In the case of 0 parity and no parity, errors cannot be detected.

**(a) Even parity**

　**(i) During transmission**
　　The number of bits whose value is "1" among the transmit data, including the parity bit, is controlled so as to be an even number.　The parity bit values are as follows.

　　- Odd number of bits whose value is "1" among transmit data:　1
　　- Even number of bits whose value is "1" among transmit data: 0

　**(ii) During reception**
　　The number of bits whose value is "1" among the reception data, including the parity bit, is counted, and if it is an odd number, a parity error is output.

**(b) Odd parity**

　**(i) During transmission**
　　Opposite to even parity, the number of bits whose value is "1" among the transmit data, including the parity bit, is controlled so that it is an odd number.　The parity bit values are as follows.

　　- Odd number of bits whose value is "1" among transmit data:　0
　　- Even number of bits whose value is "1" among transmit data: 1

　**(ii) During reception**
　　The number of bits whose value is "1" among the receive data, including the parity bit, is counted, and if it is an even number, a parity error is output.

**(c) 0 parity**
　During transmission, the parity bit is always made 0, regardless of the transmit data.
　During reception, parity bit check is not performed.　Therefore, no parity error occurs, regardless of whether the parity bit is 0 or 1.

**(d) No parity**
　No parity bit is added to the transmit data.
　Reception is performed assuming that there is no parity bit.　No parity error occurs since there is no parity bit.

**17.6.10  Receive data noise filter**

This filter samples the RXDCn pin using the base clock of the prescaler output.

When the same sampling value is read twice, the match detector output changes and the RXDCn signal is sampled as the input data.  Therefore, data not exceeding 2 clock width is judged to be noise and is not delivered to the internal circuit (see **Figure 17-18**).  See **17.7 (1) (a)  Base clock** regarding the base clock.

Moreover, since the circuit is as shown in Figure 17-17, the processing that goes on within the receive operation is delayed by 3 clocks in relation to the external signal status.

**Figure 17-17.  Noise Filter Circuit**
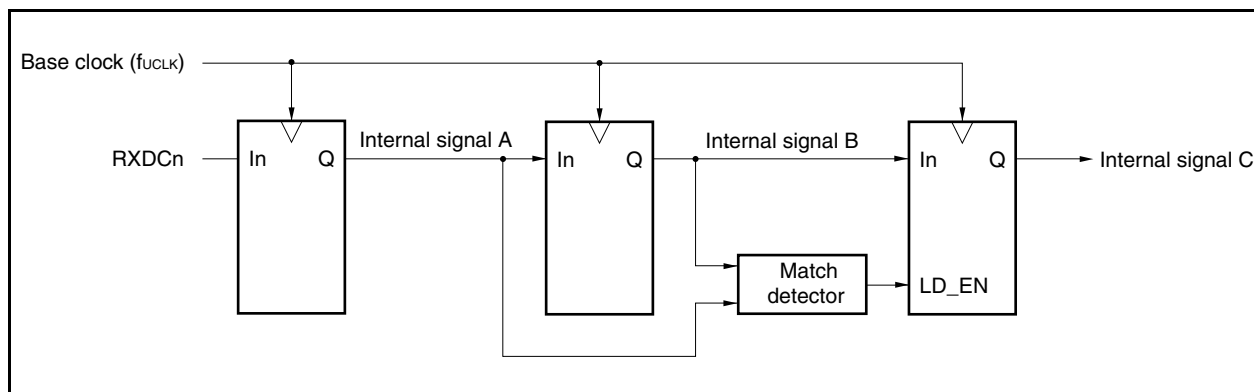


**Figure 17-18.  Timing of RXDCn Signal Judged as Noise**
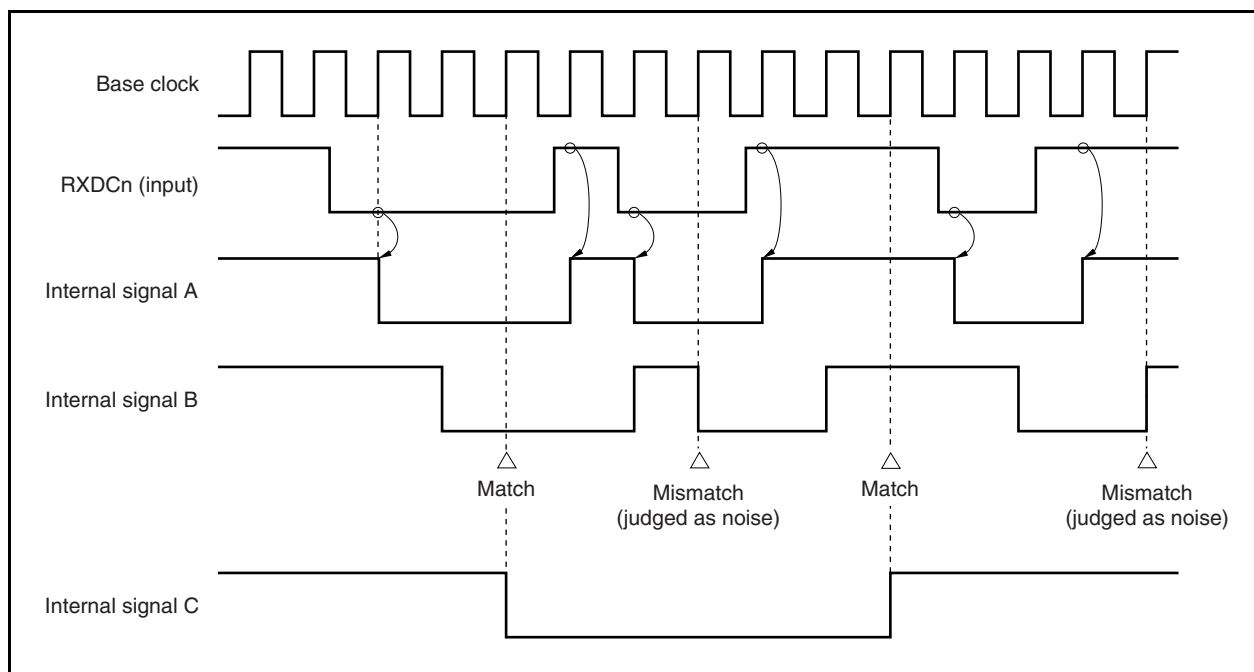
## 17.7  Dedicated Baud Rate Generator

The dedicated baud rate generator consists of a source clock selector block and an 8-bit programmable counter, and generates a serial clock during transmission and reception with UARTCn.  Regarding the serial clock, a dedicated baud rate generator output can be selected for each channel.

There is an 8-bit counter for transmission and another one for reception.

### (1)  Baud rate generator configuration

**Figure 17-19.  Configuration of Baud Rate Generator**



**Note**  Only UARTC0 is valid; setting UARTC1 and UARTC4 is prohibited.

**Remarks 1.**  n = 0 to 4

**2.**  $f_{XX}$:    Main clock frequency

$f_{UCLK}$:  Base clock frequency
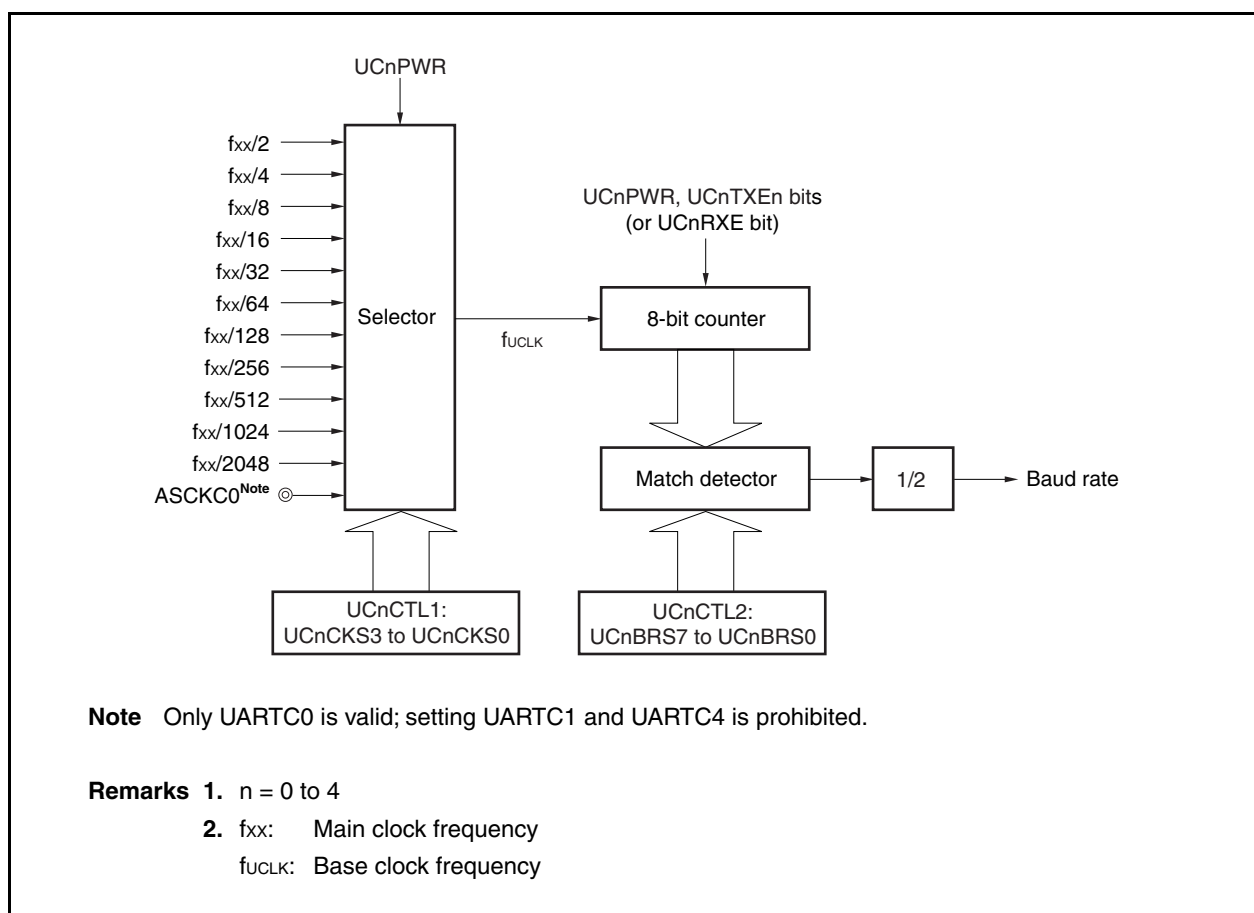
### (a)  Base clock

When the UCnCTL0.UCnPWR bit is 1, the clock selected by the UCnCTL1.UCnCKS3 to UCnCTL1.UCnCKS0 bits is supplied to the 8-bit counter.  This clock is called the base clock ($f_{UCLK}$).

### (b)  Serial clock generation

A serial clock can be generated by setting the UCnCTL1 register and the UCnCTL2 register (n = 0 to 4).

The base clock is selected by UCnCTL1.UCnCKS3 to UCnCTL1.UCnCKS0 bits.

The frequency division value for the 8-bit counter can be set using the UCnCTL2.UCnBRS7 to UCnCTL2.UCnBRS0 bits.

**(2) UARTCn control register 1 (UCnCTL1)**

The UCnCTL1 register is an 8-bit register that selects the UARTCn base clock.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**Caution   Clear the UCnCTL0.UCnPWR bit to 0 before rewriting the UCnCTL1 register.**

After reset: 00H    R/W    Address: UC0CTL1 FFFFFA01H, UC1CTL1 FFFFFA11H,
UC2CTL1 FFFFFA21H, UC3CTL1 FFFFFA31H,
UC4CTL1 FFFFFA41H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UCnCTL1 | 0 | 0 | 0 | 0 | UCnCKS3 | UCnCKS2 | UCnCKS1 | UCnCKS0 |

(n = 0 to 4)

| UCnCKS3 | UCnCKS2 | UCnCKS1 | UCnCKS0 | Base clock ($f_{UCLK}$) selection |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | $f_{xx}/2$ |
| 0 | 0 | 0 | 1 | $f_{xx}/4$ |
| 0 | 0 | 1 | 0 | $f_{xx}/8$ |
| 0 | 0 | 1 | 1 | $f_{xx}/16$ |
| 0 | 1 | 0 | 0 | $f_{xx}/32$ |
| 0 | 1 | 0 | 1 | $f_{xx}/64$ |
| 0 | 1 | 1 | 0 | $f_{xx}/128$ |
| 0 | 1 | 1 | 1 | $f_{xx}/256$ |
| 1 | 0 | 0 | 0 | $f_{xx}/512$ |
| 1 | 0 | 0 | 1 | $f_{xx}/1,024$ |
| 1 | 0 | 1 | 0 | $f_{xx}/2,048$ |
| 1 | 0 | 1 | 1 | External clock[Note] (ASCKC0 pin) |
| Other than above | | | | Setting prohibited |

**Note**   Only UARTC0 is valid; setting UARTC1 to UARTC4 is prohibited.

**Remark**   $f_{xx}$:  Main clock frequency

**(3)  UARTCn control register 2 (UCnCTL2)**

The UCnCTL2 register is an 8-bit register that selects the baud rate (serial transfer speed) clock of UARTCn.

This register can be read or written in 8-bit units.

Reset sets this register to FFH.

**Caution   Clear the UCnCTL0.UCnPWR bit to 0 or clear the UCnTXE and UCnRXE bits to 00 before rewriting the UCnCTL2 register.**

After reset FFH     R/W     Address:  UC0CTL2 FFFFFA02H, UC1CTL2 FFFFFA12H,

UC2CTL2 FFFFFA22H, UC3CTL2 FFFFFA32H,

UC4CTL2 FFFFFA42H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UCnCTL2<br>(n = 0 to 4) | UCnBRS7 | UCnBRS6 | UCnBRS5 | UCnBRS4 | UCnBRS3 | UCnBRS2 | UCnBRS1 | UCnBRS0 |

| UCn<br>BRS7 | UCn<br>BRS6 | UCn<br>BRS5 | UCn<br>BRS4 | UCn<br>BRS3 | UCn<br>BRS2 | UCn<br>BRS1 | UCn<br>BRS0 | Default<br>(k) | Serial<br>clock |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | × | × | × | Setting<br>prohibited |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 4 | $f_{UCLK}/4$ |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 5 | $f_{UCLK}/5$ |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 6 | $f_{UCLK}/6$ |
| : | : | : | : | : | : | : | : | : | : |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 252 | $f_{UCLK}/252$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 253 | $f_{UCLK}/253$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 254 | $f_{UCLK}/254$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 255 | $f_{UCLK}/255$ |

**Remark**  $f_{UCLK}$:  Clock frequency selected by the UCnCTL1.UCnCKS3 to UCnCTL1.UCnCKS0 bits

**(4) Baud rate**

The baud rate is obtained by the following equation.

$$\text{Baud rate} = \frac{f_{UCLK}}{2 \times k} \text{ [bps]}$$

When using the internal clock, the equation will be as follows (when using the ASCKC0 pin as clock at UARTC0, calculate using the above equation).

$$\text{Baud rate} = \frac{f_{XX}}{2^{m+1} \times k} \text{ [bps]}$$

**Remark**  $f_{UCLK}$ = Frequency of base clock selected by the UCnCTL1.UCnCKS3 to UCnCTL1.UCnCKS0 bits
$f_{XX}$: Main clock frequency
m = Value set using the UCnCTL1.UCnCKS3 to UCnCTL1.UCnCKS0 bits (m = 0 to 10)
k = Value set using the UCnCTL2.UCnBRS7 to UCnCTL2.UCnBRS0 bits (k = 4 to 255)

The baud rate error is obtained by the following equation.

$$\text{Error (\%)} = \left( \frac{\text{Actual baud rate (baud rate with error)}}{\text{Target baud rate (correct baud rate)}} - 1 \right) \times 100 \text{ [\%]}$$

$$= \left( \frac{f_{UCLK}}{2 \times k \times \text{Target baud rate}} - 1 \right) \times 100 \text{ [\%]}$$

When using the internal clock, the equation will be as follows (when using the ASCKC0 pin input as the clock for UARTC0, calculate the baud rate error using the above equation).

$$\text{Error (\%)} = \left( \frac{f_{XX}}{2^{m+1} \times k \times \text{Target baud rate}} - 1 \right) \times 100 \text{ [\%]}$$

**Cautions 1. The baud rate error during transmission must be within the error tolerance on the receiving side.**
**2. The baud rate error during reception must satisfy the range indicated in (5) Allowable baud rate range during reception.**

To set the baud rate, perform the following calculation for setting the UCnCTL1 and UCnCTL2 registers (when using internal clock).

<1> Set k to $f_{XX}/2/(2 \times$ target baud rate) and m to 0.

<2> If k is 256 or greater (k ≥ 256), reduce k to half (k/2) and increment m by 1 (m + 1).

<3> Repeat Step <2> until k becomes less than 256 (k < 256).

<4> Round off the first decimal point of k to the nearest whole number.
If k becomes 256 after round-off, perform Step <2> again to set k to 128.

<5> Set the value of m to UCnCTL1 register and the value of k to the UCnCTL2 register.

Example:  When $f_{XX}$ = 48 MHz and target baud rate = 153,600 bps

<1> k = 480,000,000/2/(2 × 153,600) = 78.125…, m = 0

<2>, <3> k = 78.125… < 256, m = 0

<4> Set value of UCnCTL2 register: k = 78 = 4EH, set value of UCnCTL1 register: m = 0

$$\text{Actual baud rate} = 48,000,000/2/(2 \times 78)$$
$$= 153,846 \text{ [bps]}$$

$$\text{Baud rate error} = \{48,000,000/2/(2 \times 78 \times 153,600) - 1\} \times 100$$
$$= 0.160 \text{ [\%]}$$

The representative examples of baud rate settings are shown below.

**Table 17-4.  Baud Rate Generator Setting Data**

| Baud Rate (bps) | $f_{XX}$ = 48 MHz | | | $f_{XX}$ = 32 MHz | | | $f_{XX}$ = 24 MHz | | |
|---|---|---|---|---|---|---|---|---|---|
| | UCnCTL1 | UCnCTL2 | ERR (%) | UCnCTL1 | UCnCTL2 | ERR (%) | UCnCTL1 | UCnCTL2 | ERR (%) |
| 300 | 08H | 9CH | 0.16 | 07H | D0H | 0.16 | 07H | 9CH | −2.3 |
| 600 | 07H | 9CH | 0.16 | 06H | D0H | 0.16 | 06H | 9CH | 0.16 |
| 1,200 | 06H | 9CH | 0.16 | 05H | D0H | 0.16 | 05H | 9CH | 0.16 |
| 2,400 | 05H | 9CH | 0.16 | 04H | D0H | 0.16 | 04H | 9CH | 0.16 |
| 4,800 | 04H | 9CH | 0.16 | 03H | D0H | 0.16 | 03H | 9CH | 0.16 |
| 9,600 | 03H | 9CH | 0.16 | 02H | D0H | 0.16 | 02H | 9CH | 0.16 |
| 19,200 | 02H | 9CH | 0.16 | 01H | D0H | 0.16 | 01H | 9CH | 0.16 |
| 31,250 | 01H | C0H | 0.00 | 01H | 80H | 0.00 | 00H | C0H | 0.00 |
| 38,400 | 01H | 9CH | 0.16 | 00H | D0H | 0.16 | 00H | 9CH | 0.16 |
| 76,800 | 00H | 9CH | 0.16 | 00H | 68H | 0.16 | 00H | 4EH | 0.16 |
| 153,600 | 00H | 4EH | 0.16 | 00H | 34H | 0.16 | 00H | 27H | 0.16 |
| 312,500 | 00H | 26H | 1.05 | 00H | 1AH | −1.54 | 00H | 13H | 1.05 |
| 625,000 | 00H | 13H | 1.05 | 00H | 0DH | −1.54 | 00H | 0AH | −4.00 |
| 1,000,000 | 00H | 0CH | 0.00 | 00H | 08H | 0.00 | 00H | 06H | 0.00 |
| 1,250,000 | 00H | 0AH | −4.00 | Setting prohibited | | | 00H | 05H | −4.00 |
| 2,000,000 | 00H | 06H | 0.00 | 00H | 04H | 0.00 | Setting prohibited | | |
| 2,500,000 | 00H | 05H | −4.00 | Setting prohibited | | | | | |
| 3,000,000 | 00H | 04H | 0.00 | | | | | | |

**Remark**  $f_{XX}$:  Main clock frequency
ERR: Baud rate error (%)

**(5) Allowable baud rate range during reception**

The baud rate error range at the destination that is allowable during reception is shown below.

**Caution   The baud rate error during reception must be set within the allowable error range using the following equation.**

**Figure 17-20.  Allowable Baud Rate Range During Reception**
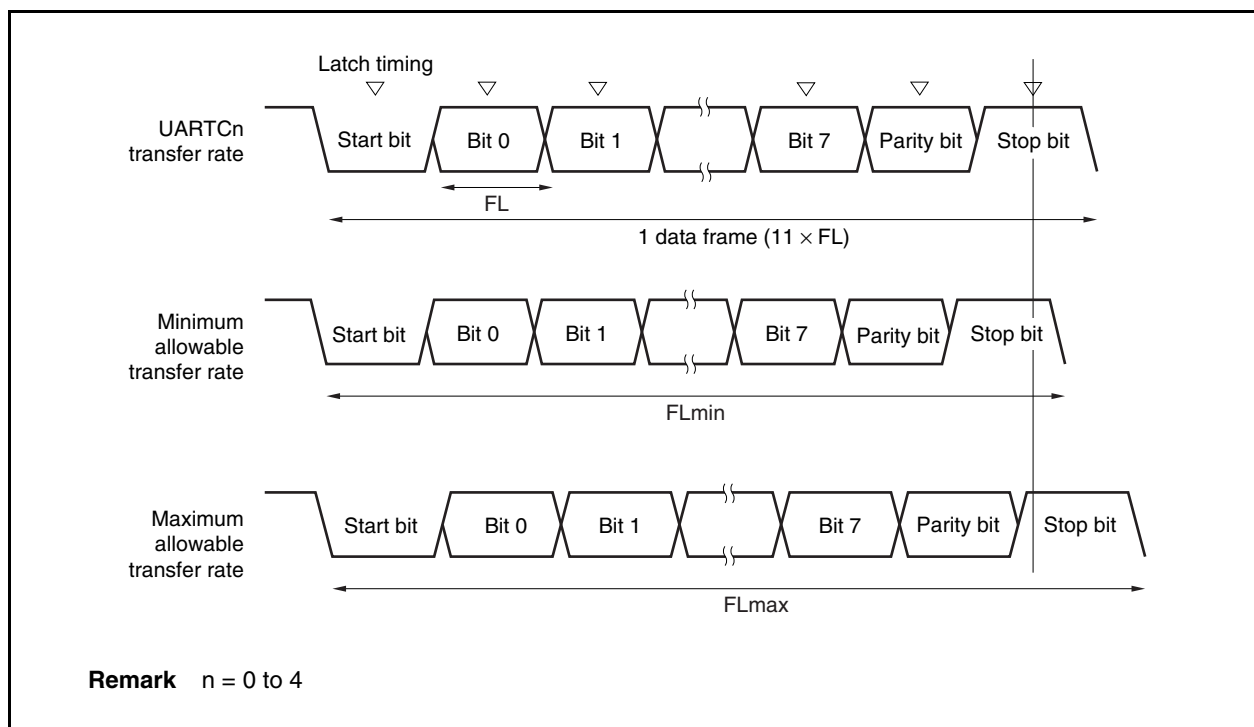


As shown in Figure 17-20, the receive data latch timing is determined by the counter set using the UCnCTL2 register following start bit detection.  The transmit data can be normally received if up to the last data (stop bit) can be received in time for this latch timing.

When this is applied to 11-bit reception, the following is the theoretical result.

$$FL = (Brate)^{-1}$$

Brate:  UARTCn baud rate (n = 0 to 4)
k:        Set value of UCnCTL2.UCnBRS7 to UCnCTL2.UCnBRS0 bits (n = 0 to 4)
FL:       1-bit data length
Latch timing margin: 2 clocks

Minimum allowable transfer rate: $FLmin = 11 \times FL - \dfrac{k-2}{2k} \times FL = \dfrac{21k+2}{2k} FL$

Therefore, the maximum baud rate that can be received by the destination is as follows.

$$BRmax = (FLmin/11)^{-1} = \frac{22k}{21k + 2} \text{ Brate}$$

Similarly, obtaining the following maximum allowable transfer rate yields the following.

$$\frac{10}{11} \times FLmax = 11 \times FL - \frac{k + 2}{2 \times k} \times FL = \frac{21k - 2}{2 \times k} FL$$

$$FLmax = \frac{21k - 2}{20 \, k} FL \times 11$$

Therefore, the minimum baud rate that can be received by the destination is as follows.

$$BRmin = (FLmax/11)^{-1} = \frac{20k}{21k - 2} \text{ Brate}$$

Obtaining the allowable baud rate error for UARTCn and the destination from the above-described equations for obtaining the minimum and maximum baud rate values yields the following.

**Table 17-5.  Maximum/Minimum Allowable Baud Rate Error**

| Division Ratio (k) | Maximum Allowable Baud Rate Error | Minimum Allowable Baud Rate Error |
|---|---|---|
| 4 | +2.32% | −2.43% |
| 8 | +3.52% | −3.61% |
| 20 | +4.26% | −4.30% |
| 50 | +4.56% | −4.58% |
| 100 | +4.66% | −4.67% |
| 255 | +4.72% | −4.72% |

**Remarks 1.** The reception accuracy depends on the bit count in 1 frame, the input clock frequency, and the division ratio (k).  The higher the input clock frequency and the larger the division ratio (k), the higher the accuracy.

**2.** k: Set value of UCnCTL2.UCnBRS7 to UCnCTL2.UCnBRS0 bits (n = 0 to 4)

**(6) Transfer rate during continuous transmission**

During continuous transmission, the transfer rate from the stop bit to the next start bit is usually 2 base clocks longer.  However, timing initialization is performed via start bit detection by the receiving side, so this has no influence on the transfer result.

**Figure 17-21.  Transfer Rate During Continuous Transfer**
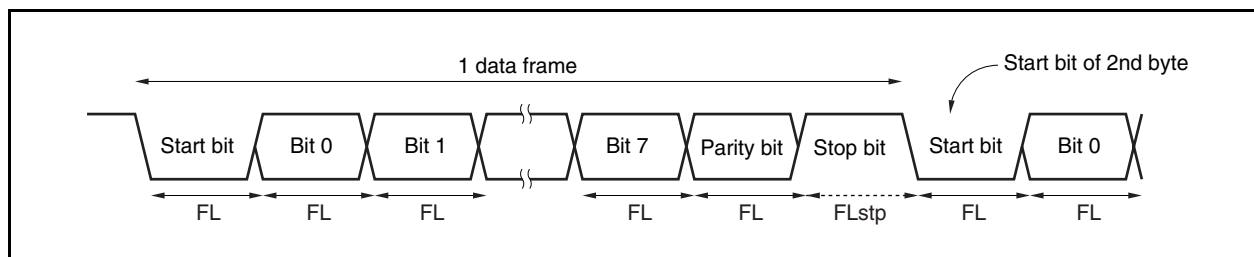


The following equation can be obtained assuming 1 bit data length: FL; stop bit length: FLstp; and base clock frequency: $f_{UCLK}$.

FLstp = FL + 2/$f_{UCLK}$

Therefore, the transfer rate during continuous transmission is as follows.

Transfer rate = 11 $\times$ FL + (2/$f_{UCLK}$)

## 17.8 Cautions

(1) When the clock supply to UARTCn is stopped (for example, in IDLE1, IDLE2, or STOP mode), the operation stops with each register retaining the value it had immediately before the clock supply was stopped. The TXDCn pin output also holds and outputs the value it had immediately before the clock supply was stopped. However, the operation is not guaranteed after the clock supply is resumed. Therefore, after the clock supply is resumed, the circuits should be initialized by setting the UCnCTL0.UCnPWR, UCnCTL0.UCnRXEn, and UCnCTL0.UCnTXEn bits to 000.

(2) The RXDC1 and KR7 pins must not be used at the same time. To use the RXDC1 pin, do not use the KR7 pin. To use the KR7 pin, do not use the RXDC1 pin (it is recommended to set the PFC91 bit to 1 and clear PFCE91 bit to 0).

(3) Start up the UARTCn in the following sequence.
&lt;1&gt; Set the UCnCTL0.UCnPWR bit to 1.
&lt;2&gt; Set the ports.
&lt;3&gt; Set the UCnCTL0.UCnTXE bit to 1, UCnCTL0.UCnRXE bit to 1.

(4) Stop the UARTCn in the following sequence.
&lt;1&gt; Set the UCnCTL0.UCnTXE bit to 0, UCnCTL0.UCnRXE bit to 0.
&lt;2&gt; Set the ports and set the UCnCTL0.UCnPWR bit to 0 (it is not a problem if port setting is not changed).

(5) In transmit mode (UCnCTL0.UCnPWR bit = 1 and UCnCTL0.UCnTXE bit = 1), do not overwrite the same value to the UCnTX register by software because transmission starts by writing to this register. To transmit the same value continuously, overwrite the same value.

(6) In continuous transmission, the communication rate from the stop bit to the next start bit is extended 2 base clocks more than usual. However, the reception side initializes the timing by detecting the start bit, so the reception result is not affected.

# CHAPTER 18  3-WIRE  VARIABLE-LENGTH  SERIAL  I/O  (CSIF)

## 18.1  Mode Switching of CSIF and Other Serial Interfaces

### 18.1.1  CSIF4 and UARTC0 mode switching

In the V850ES/JG3-U and V850ES/JH3-U, CSIF4 and UARTC0 share the same pins and therefore cannot be used simultaneously.  To use CSIF4, the use of CSIF4 must be set in advance, using the PMC3, PFC3 and PFCE3 registers.

**Caution   The transmit/receive operation of CSIF4 and UARTC0 is not guaranteed if these functions are switched during transmission or reception.  Be sure to disable the one that is not used.**

**Figure 18-1.  CSIF4 and UARTC0 Mode Switch Settings**

After reset:  00H      R/W      Address:  FFFFF446H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC3 | PMC37 | PMC36 | PMC35 | PMC34 | PMC33 | PMC32 | PMC31 | PMC30 |

After reset:  00H      R/W      Address:  FFFFF466H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC3 | PFC37 | PFC36 | PFC35 | PFC34 | PFC33 | PFC32 | PFC31 | PFC30 |

After reset:  00H      R/W      Address:  FFFFF706H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE3 | PFCE37 | PFCE36 | PFCE35 | PFCE34 | PFCE33 | PFCE32 | PFCE31 | PFCE30 |

| PMC32 | PFCE32 | PFC32 | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | ASCKC0 |
| 1 | 0 | 1 | $\overline{\text{SCKF4}}$ |

| PMC3n | PFC3n | Operation mode |
|---|---|---|
| 0 | × | Port I/O mode |
| 1 | 0 | UARTC0 mode |
| 1 | 1 | CSIF4 mode |

**Remarks 1.**  n = 0, 1
**2.**  × = don't care

### 18.1.2  CSIF0, UARTC4, and I²C01 mode switching

In the V850ES/JG3-U and V850ES/JH3-U, CSIF0, UARTC4, and I²C01 share the same pins and therefore cannot be used simultaneously.  Switching among CSIF0, UARTC4, and I²C01 must be set in advance, using the PMC4, PFC4, and PFCE4 registers.

**Caution   The transmit/receive operation of CSIF0, UARTC4, and I²C01 is not guaranteed if these functions are switched during transmission or reception.  Be sure to disable the one that is not used.**

**Figure 18-2.  CSIF0, UARTC4, and I²C01 Mode Switch Settings**

After reset:  00H     R/W     Address:  FFFFF448H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC4 | 0 | 0 | 0 | 0 | 0 | PMC42 | PMC41 | PMC40 |

After reset:  00H     R/W     Address:  FFFFF468H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC4 | 0 | 0 | 0 | 0 | 0 | PFC42 | PFC41 | PFC40 |

After reset:  00H     R/W     Address:  FFFFF708H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE4 | 0 | 0 | 0 | 0 | 0 | 0 | PFCE41 | PFCE40 |

| PMC4n | PFCE4n | PFC4n | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | CSIF0 mode |
| 1 | 0 | 1 | UARTC4 mode |
| 1 | 1 | 0 | I²C01 mode |

**Remarks 1.**  n = 0, 1
**2.**  × = don't care

### 18.1.3 CSIF3 and UARTC2 mode switching

In the V850ES/JG3-U and V850ES/JH3-U, CSIF3 and UARTC2 share the same pins and therefore cannot be used simultaneously. Switching between CSIF3 and UARTC2 must be set in advance, using the PMC9, PFC9 and PFCE9 registers.

**Caution   The transmit/receive operation of CSIF3 and UARTC2 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.**

**Figure 18-3.  CSIF3 and UARTC2 Mode Switch Settings**

After reset: 0000H    R/W    Address: FFFFF452H, FFFFF453H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PMC9 | PMC915 | PMC914 | PMC913 | PMC912 | PMC911 | PMC910 | PMC99 | PMC98 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PMC97 | PMC96 | PMC95 | PMC94 | PMC93 | PMC92 | PMC91 | PMC90 |

After reset: 0000H    R/W    Address: FFFFF472H, FFFFF473H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFC9 | PFC915 | PFC914 | PFC913 | PFC912 | PFC911 | PFC910 | PFC99 | PFC98 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PFC97 | PFC96 | PFC95 | PFC94 | PFC93 | PFC92 | PFC91 | PFC90 |

After reset: 0000H    R/W    Address: FFFFF712H, FFFFF713H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFCE9 | PFCE915 | PFCE914 | 0 | 0 | PFCE911 | PFCE910 | PFCE99 | PFCE98 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PFCE97 | PFCE96 | PFCE95 | PFCE94 | PFCE93 | PFCE92 | PFCE91 | PFCE90 |

| PMC91n | PFCE91n | PFC91n | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | CSIF3 mode |
| 1 | 0 | 1 | UARTC2 mode |

**Remarks 1.**  n = 0, 1
         **2.**  × = don't care

## 18.2  Features

○ Transfer rate:  12 Mbps max. ($f_{XX}$ = 48 MHz, using internal clock, master mode: CSIF3)

                    8 Mbps ($f_{XX}$ = 48 MHz, using internal clock, master mode: CSIF0 to CSIF4)

○ Master mode and slave mode selectable

○ 8-bit to 16-bit transfer, 3-wire serial interface

○ Interrupt request signals (INTCFnT, INTCFnR)

○ Serial clock and data phase switchable

○ Transfer data length selectable in 1-bit units between 8 and 16 bits

○ Transfer data MSB-first/LSB-first switchable

○ 3-wire transfer    SOFn:    Serial data output

                       SIFn:      Serial data input

                       $\overline{\text{SCKFn}}$: Serial clock I/O

○ Transmission mode, reception mode, and transmission/reception mode specifiable


**Remark**    n = 0 to 4

## 18.3  Configuration

The following shows the block diagram of CSIFn.

**Figure 18-4.  Block Diagram of CSIFn**



**Note**   For CSIF3:  $f_{XX}/2$,  $f_{XX}/4$,  $f_{XX}/8$,  $f_{XX}/16$,  $f_{XX}/32$,  $f_{XX}/64$

**Remarks**  $f_{CCLK}$:     Communication clock

   $f_{XX}$:       Main clock frequency

   $f_{BRGm}$:    Count clock of the baud rate generator

   n = 0 to 4

   m = 1 (n = 0, 1)

   m = 2 (n = 2, 3)

   m = 3 (n = 4)
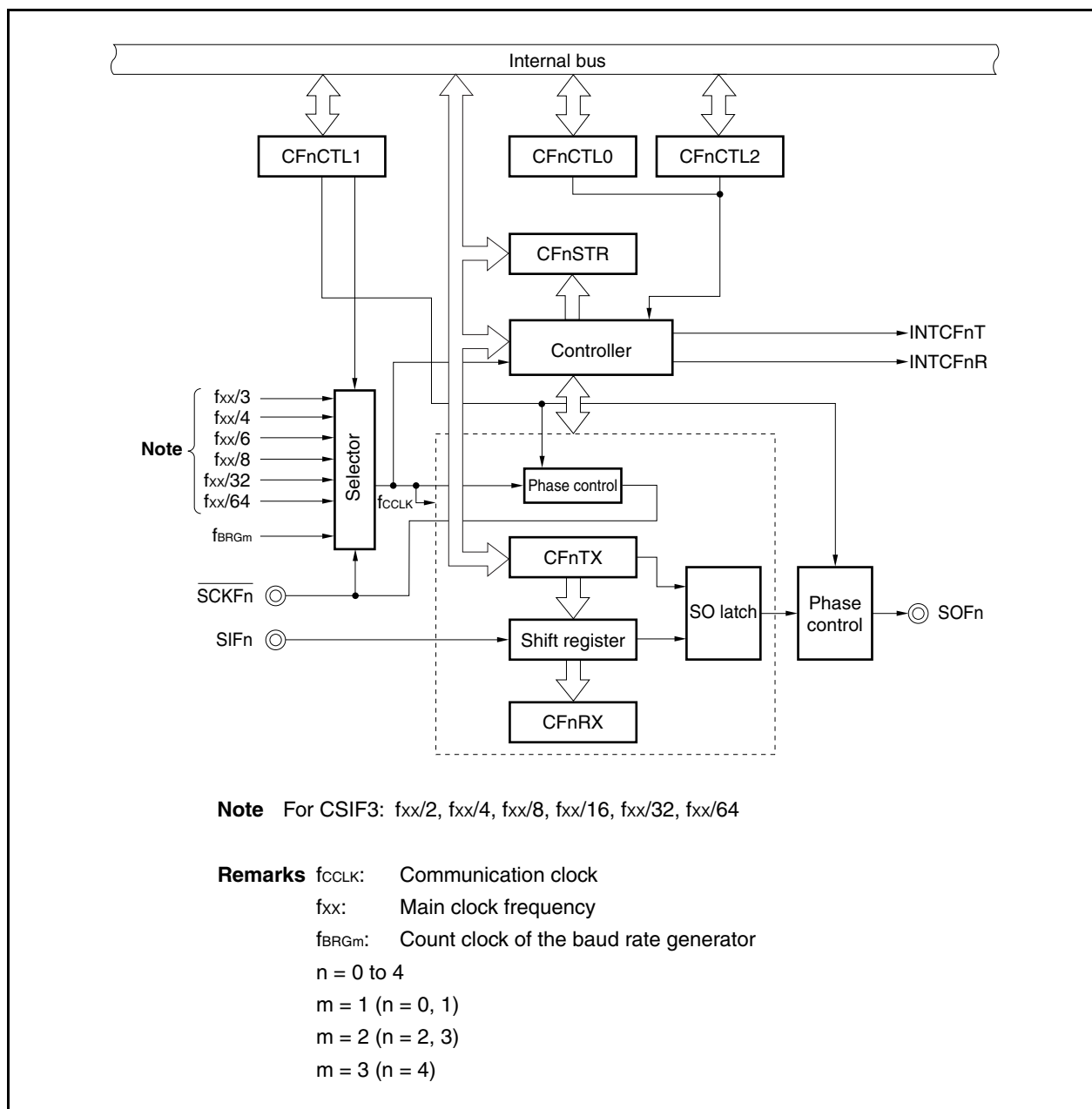
CSIFn includes the following hardware.

**Table 18-1. Configuration of CSIFn**

| Item | Configuration |
|------|---------------|
| Registers | CSIFn receive data register (CFnRX)<br>CSIFn transmit data register (CFnTX)<br>CSIFn control register 0 (CFnCTL0)<br>CSIFn control register 1 (CFnCTL1)<br>CSIFn control register 2 (CFnCTL2)<br>CSIFn status register (CFnSTR) |

**(1) CSIFn receive data register (CFnRX)**

The CFnRX register is a 16-bit buffer register that holds receive data.

This register is read-only, in 16-bit units.

The receive operation is started by reading the CFnRX register in the reception enabled status.

If the transfer data length is 8 bits, the lower 8 bits of this register are read-only in 8-bit units as the CFnRXL register.

Reset sets this register to 0000H.

In addition to reset input, the CFnRX register can be initialized by clearing (to 0) the CFnPWR bit of the CFnCTL0 register.
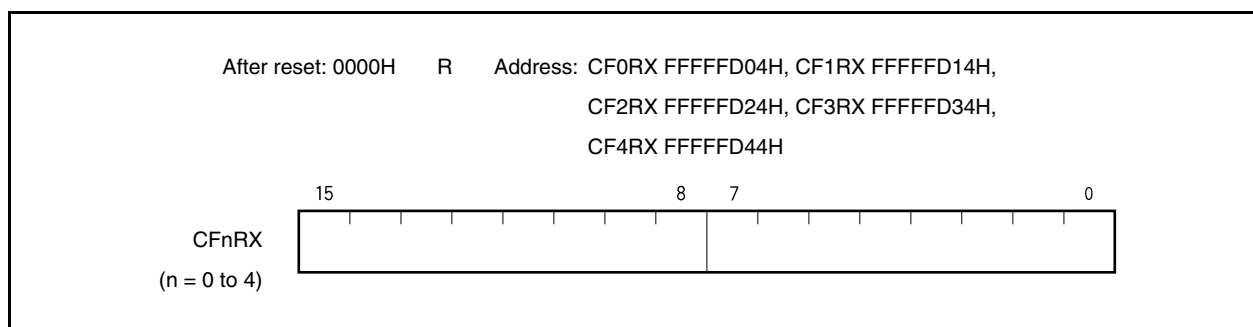
**(2) CSIFn transmit data register (CFnTX)**

The CFnTX register is a 16-bit buffer register used to write the CSIFn transfer data.

This register can be read or written in 16-bit units.

The transmit operation is started by writing data to the CFnTX register in the transmission enabled status.

If the transfer data length is 8 bits, the lower 8 bits of this register are read-only in 8-bit units as the CFnTXL register.

Reset sets this register to 0000H.

After reset 0000H      R/W      Address:  CF0TX FFFFFD06H, CF1TX FFFFFD16H,
                                          CF2TX FFFFFD26H, CF3TX FFFFFD36H,
                                          CF4TX FFFFFD46H

|   | 15 | | | | | | | 8 | 7 | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
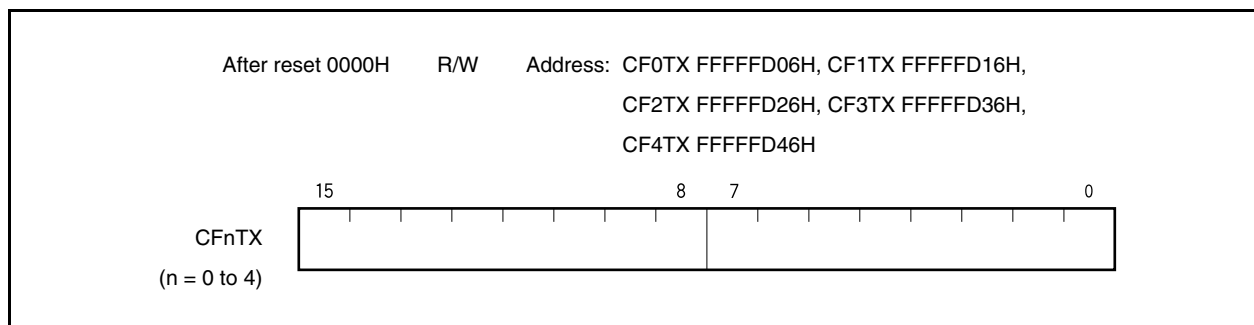
CFnTX
(n = 0 to 4)

**Remark** The communication start conditions are shown below.

| | |
|---|---|
| Transmission mode (CFnTXE bit = 1, CFnRXE bit = 0): | Write to CFnTX register |
| Transmission/reception mode (CFnTXE bit = 1, CFnRXE bit = 1): | Write to CFnTX register |
| Reception mode (CFnTXE bit = 0, CFnRXE bit = 1): | Read from CFnRX register |

## 18.4  Registers

The following registers are used to control CSIFn.

- CSIFn control register 0 (CFnCTL0)
- CSIFn control register 1 (CFnCTL1)
- CSIFn control register 2 (CFnCTL2)
- CSIFn status register (CFnSTR)

**(1)  CSIFn control register 0 (CFnCTL0)**

CFnCTL0 is a register that controls the CSIFn serial transfer operation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 01H.

(1/3)

After reset: 01H    R/W    Address:  CF0CTL0 FFFFFD00H, CF1CTL0 FFFFFD10H,

CF2CTL0 FFFFFD20H, CF3CTL0 FFFFFD30H,

CF4CTL0 FFFFFD40H

| | <7> | <6> | <5> | <4> | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| CFnCTL0 | CFnPWR | CFnTXE[Note] | CFnRXE[Note] | CFnDIR[Note] | 0 | 0 | CFnTMS[Note] | CFnSCE |

(n = 0 to 4)

| CFnPWR | Specification of CSIFn operation disable/enable |
|---|---|
| 0 | Disables CSIFn operation and resets the CFnSTR register |
| 1 | Enables CSIFn operation |
| • The CFnPWR bit controls the CSIFn operation and resets the internal circuit. | |

| CFnTXE[Note] | Specification of transmit operation disable/enable |
|---|---|
| 0 | Disables transmit operation |
| 1 | Enables transmit operation |
| • The SOFn output is low level when the CFnTXE bit is 0. | |

| CFnRXE[Note] | Specification of receive operation disable/enable |
|---|---|
| 0 | Disables receive operation |
| 1 | Enables receive operation |
| • No reception completion interrupt is output even when the prescribed data is transferred, and the receive data (CFnRX register) is not updated, because the receive operation is disabled by clearing the CFnRXE bit to 0. | |

**Note**  These bits can only be rewritten when the CFnPWR bit = 0. However, CFnPWR bit = 1 can also be set at the same time as rewriting these bits.

**Caution  To forcibly suspend transmission/reception, clear the CFnPWR bit to 0 instead of the CFnRXE and CFnTXE bits. At this time, the clock output is stopped.**

(2/3)

| CFnDIR[Note] | Specification of transfer direction mode (MSB/LSB) |
|---|---|
| 0 | MSB-first transfer |
| 1 | LSB-first transfer |

| CFnTMS[Note] | Transfer mode specification |
|---|---|
| 0 | Single transfer mode |
| 1 | Continuous transfer mode |

[In single transfer mode]
  The reception completion interrupt (INTCFnR) occurs when communication is complete.
  Even if transmission is enabled (CFnTXE bit = 1), the transmission enable interrupt (INTCFnT) does not occur.
  If the next transmit data is written during communication (CFnSTR.CFnTSF bit = 1), it is ignored and the next communication is not started.  Also, if reception-only communication is set (CFnTXE bit = 0, CFnRXE bit = 1), the next communication is not started even if the receive data is read during communication (CFnSTR. CFnTSF bit = 1).

[In continuous transfer mode]
  The continuous transmission is enabled by writing the next transmit data during communication (CFnSTR.CFnTSF bit = 1).
  Writing the next transmission data is enabled after a transmission enable interrupt (INTCFnT) occurs.
  If reception-only communication is set (CFnTXE bit = 0, CFnRXE bit = 1) in the continuous transfer mode, the next reception is started immediately after a reception completion interrupt (INTCFnR), regardless of the read operation of the CFnRX register.
  Therefore, immediately read the receive data from the CFnRX register.  If this read operation is delayed, an overrun error (CFnOVE bit = 1) occurs.

**Note**  These bits can only be rewritten when the CFnPWR bit = 0.  However, the CFnPWR can be set to 1 at the same time as these bits are rewritten.

| CFnSCE | Specification of start transfer disable/enable |
|--------|-----------------------------------------------|
| 0 | Communication start trigger invalid |
| 1 | Communication start trigger valid |

- In master mode
  This bit enables or disables the communication start trigger.
  (a) In single transmission or transmission/reception mode, or continuous
      transmission or continuous transmission/reception mode
      A communication operation can be started by writing data to the CFnTX
      register when the CFnSCE bit is 1.
      Set the CFnSCE bit to 1.
  (b) In single reception mode
      Disable starting the next receive operation by clearing the CFnSCE bit to 0
      before reading the last receive data, because a receive operation is started by
      reading receive data (CFnRX register)[Note 1].
  (c) In continuous reception mode
      Clear the CFnSCE bit to 0 one communication clock before reception of the
      last data is completed to disable the start of reception after the last data is
      received[Note 2].
- In slave mode
  This bit enables or disables the communication start trigger.
  Set the CFnSCE bit to 1.

[Usage of CFnSCE bit]
- In single reception mode
  <1> When reception of the last data is completed by INTCFnR interrupt
      servicing, clear the CFnSCE bit to 0 before reading the CFnRX register.
  <2> After confirming the CFnSTR.CFnTSF bit = 0, clear the CFnRXE bit to 0 to
      disable reception.
      To continue reception, set the CFnSCE bit to 1 to start the next reception
      by dummy-reading the CFnRX register.
- In continuous reception mode
  <1> Clear the CFnSCE bit to 0 during reception of the last data by INTCFnR
      interrupt servicing.
  <2> Read the CFnRX register.
  <3> Read the last reception data by reading the CFnRX register after
      acknowledging the CFnTIR interrupt.
  <4> After confirming the CFnSTR.CFnTSF bit = 0, clear the CFnRXE bit to 0 to
      disable reception.
      To continue reception, set the CFnSCE bit to 1 to wait for the next reception
      by dummy-reading the CFnRX register.

Notes  1.  If the CFnSCE bit is read while it is 1, the next communication operation is started.

       2.  The CFnSCE bit is not cleared to 0 one communication clock before the completion of
           the last data reception, the next communication operation is automatically started.

Caution   Be sure to clear bits 3 and 2 to "0".

**(2) CSIFn control register 1 (CFnCTL1)**

CFnCTL1 is an 8-bit register that controls the CSIFn serial transfer operation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

**Caution   The CFnCTL1 register can be rewritten only when the CFnCTL0.CFnPWR bit = 0.**

After reset: 00H     R/W     Address: CF0CTL1 FFFFFD01H, CF1CTL1 FFFFFD11H,

CF2CTL1 FFFFFD21H, CF3CTL1 FFFFFD31H,

CF4CTL1 FFFFFD41H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CFnCTL1 (n = 0 to 4) | 0 | 0 | 0 | CFnCKP | CFnDAP | CFnCKS2 | CFnCKS1 | CFnCKS0 |

| | CFnCKP | CFnDAP | Specification of data transmission/ reception timing in relation to $\overline{\text{SCKFn}}$ |
|---|---|---|---|
| Communication type 1 | 0 | 0 |  |
| Communication type 2 | 0 | 1 |  |
| Communication type 3 | 1 | 0 |  |
| Communication type 4 | 1 | 1 |  |

| CFnCKS2 | CFnCKS1 | CFnCKS0 | Communication clock ($f_{CCLK}$) | | Mode |
|---|---|---|---|---|---|
| | | | n = 0 to 2, 4[Note 1] | n = 3[Note 2] | |
| 0 | 0 | 0 | $f_{XX}/3$ | $f_{XX}/2$ | Master mode |
| 0 | 0 | 1 | $f_{XX}/4$ | $f_{XX}/4$ | Master mode |
| 0 | 1 | 0 | $f_{XX}/6$ | $f_{XX}/8$ | Master mode |
| 0 | 1 | 1 | $f_{XX}/8$ | $f_{XX}/16$ | Master mode |
| 1 | 0 | 0 | $f_{XX}/32$ | $f_{XX}/32$ | Master mode |
| 1 | 0 | 1 | $f_{XX}/64$ | $f_{XX}/64$ | Master mode |
| 1 | 1 | 0 | $f_{BRGm}$ | | Master mode |
| 1 | 1 | 1 | External clock ($\overline{\text{SCKFn}}$) | | Slave mode |

**Notes 1.** Set the communication clock ($f_{CCLK}$) to 8 MHz or lower (master/slave mode).

**2.** Set the communication clock ($f_{CCLK}$) to 12 MHz or lower (master mode), 8 MHz or lower (master/slave mode).

**Remark**   When n = 0, 1, m = 1

When n = 2, 3, m = 2

When n = 4, m = 3

For details of $f_{BRGm}$, see **18.8  Baud Rate Generator**.

**(3) CSIFn control register 2 (CFnCTL2)**

　　CFnCTL2 is an 8-bit register that controls the number of CSIFn serial transfer bits.

　　This register can be read or written in 8-bit units.

　　Reset sets this register to 00H.

　　**Caution　The CFnCTL2 register can be rewritten only when the CFnCTL0.CFnPWR bit = 0 or when both the CFnTXE and CFnRXE bits = 0.**

After reset: 00H　　R/W　　Address: CF0CTL2 FFFFFD02H, CF1CTL2 FFFFFD12H,
　　　　　　　　　　　　　　　　　CF2CTL2 FFFFFD22H, CF3CTL2 FFFFFD32H,
　　　　　　　　　　　　　　　　　CF4CTL2 FFFFFD42H

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CFnCTL2 | 0 | 0 | 0 | 0 | CFnCL3 | CFnCL2 | CFnCL1 | CFnCL0 |

(n = 0 to 4)

| CFnCL3 | CFnCL2 | CFnCL1 | CFnCL0 | Serial register bit length |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 8 bits |
| 0 | 0 | 0 | 1 | 9 bits |
| 0 | 0 | 1 | 0 | 10 bits |
| 0 | 0 | 1 | 1 | 11 bits |
| 0 | 1 | 0 | 0 | 12 bits |
| 0 | 1 | 0 | 1 | 13 bits |
| 0 | 1 | 1 | 0 | 14 bits |
| 0 | 1 | 1 | 1 | 15 bits |
| 1 | × | × | × | 16 bits |

**Remarks　1.** If the number of transfer bits is other than 8 or 16, prepare and use data stuffed from the LSB of the CFnTX and CFnRX registers.

　　　　　　**2.** ×: don't care

**(a) Transfer data length change function**

The CSIFn transfer data length can be set in 1-bit units between 8 and 16 bits using the CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits.

When the transfer bit length is set to a value other than 16 bits, set the data to the CFnTX or CFnRX register starting from the LSB, regardless of whether the transfer start bit is the MSB or LSB. Any data can be set for the higher bits that are not used, but the receive data becomes 0 following serial transfer.

**(i) Transfer bit length = 10 bits, MSB first**



**(ii) Transfer bit length = 12 bits, LSB first**
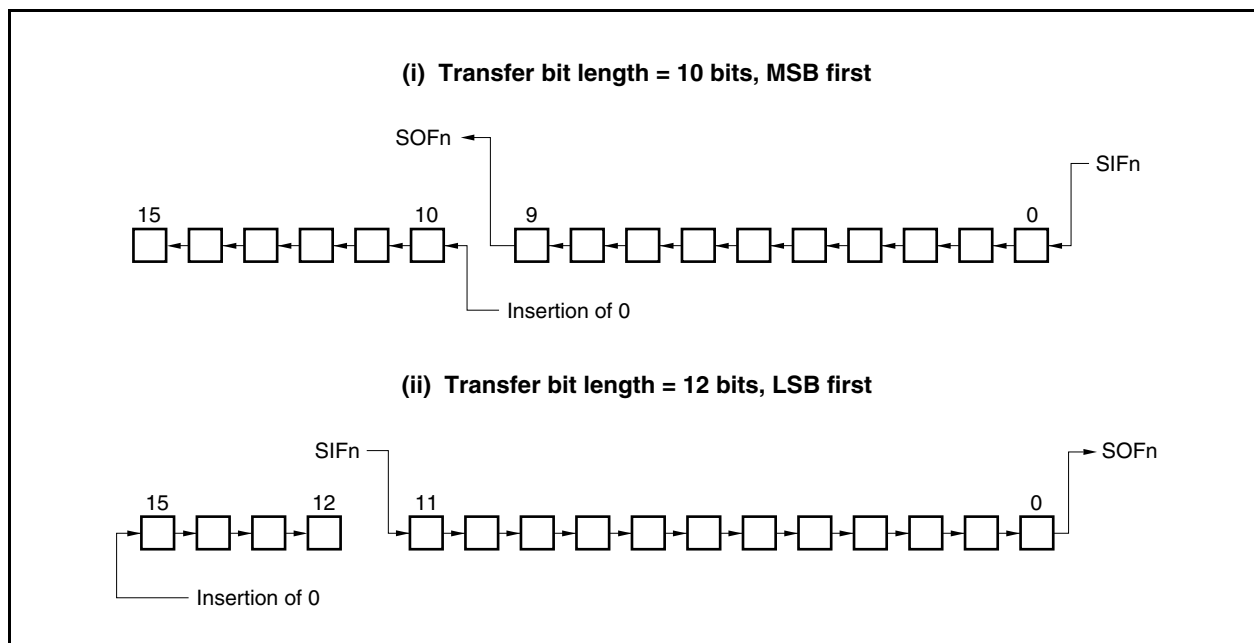
**(4) CSIFn status register (CFnSTR)**

CFnSTR is an 8-bit register that displays the CSIFn status.

This register can be read or written in 8-bit or 1-bit units, but the CFnTSF flag is read-only.

Reset sets this register to 00H.

In addition to reset input, the CFnSTR register can be initialized by clearing (0) the CFnCTL0.CFnPWR bit.

After reset: 00H　　R/W　　Address: CF0STR FFFFFD03H, CF1STR FFFFFD13H,
CF2STR FFFFFD23H, CF3STR FFFFFD33H,
CF4STR FFFFFD43H

| | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| CFnSTR | CFnTSF | 0 | 0 | 0 | 0 | 0 | 0 | CFnOVE |

(n = 0 to 4)

| CFnTSF | Communication status flag |
|---|---|
| 0 | Communication stopped |
| 1 | Communicating |
| • During transmission, this register is set when data is prepared in the CFnTX register, and during reception, it is set when a dummy read of the CFnRX register is performed.<br>When transfer ends, this flag is cleared to 0 at the last edge of the clock. | |

| CFnOVE | Overrun error flag |
|---|---|
| 0 | No overrun |
| 1 | Overrun |
| • An overrun error occurs when the next reception is completed without the CPU reading the value of the receive buffer, upon completion of the receive operation. The CFnOVE flag displays the overrun error occurrence status in this case.<br>• The CFnOVE bit is valid also in the single transfer mode. Therefore, when only using transmission, note the following.<br>• Do not check the CFnOVE flag.<br>• Read this bit even if reading the reception data is not required.<br>• The CFnOVE flag is cleared by writing 0 to it. It cannot be set even by writing 1 to it. | |

## 18.5  Interrupt Request Signals

CSIFn can generate the following two types of interrupt request signals.

- Reception completion interrupt request signal (INTCFnR)
- Transmission enable interrupt request signal (INTCFnT)

Of these two interrupt request signals, the reception completion interrupt request signal has the higher priority by default, and the priority of the transmission enable interrupt request signal is lower.

**Table 18-2.  Interrupts and Their Default Priority**

| Interrupt | Priority |
|---|---|
| Reception complete | High |
| Transmission enable | Low |

**(1)  Reception completion interrupt request signal (INTCFnR)**

When receive data is transferred to the CFnRX register while reception is enabled, the reception completion interrupt request signal is generated.

This interrupt request signal can also be generated if an overrun error occurs.

When the reception completion interrupt request signal is acknowledged and the data is read, read the CFnSTR register to check that the result of reception is not an error.

In the single transfer mode, the INTCFnR interrupt request signal is generated upon completion of transmission, even when only transmission is executed.

**(2)  Transmission enable interrupt request signal (INTCFnT)**

In the continuous transmission or continuous transmission/reception mode, transmit data is transferred from the CFnTX register and, as soon as writing to CFnTX has been enabled, the transmission enable interrupt request signal is generated.

In the single transmission and single transmission/reception modes, the INTCFnT interrupt is not generated.

## 18.6  Operation

### 18.6.1  Single transfer mode (master mode, transmission mode)

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$ (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 000), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)

### (1)  Operation flow



Remarks  1.  The broken lines indicate the hardware processing.
2.  The numbers in this figure correspond to the processing numbers in **(2)  Operation timing**.
3.  n = 0 to 4

**(2) Operation timing**



(1) Write 00H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$, and master mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write C1H to the CFnCTL0 register, and select the transmission mode and MSB first at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) Set the CFnSTR.CFnTSF bit to 1 by writing the transmit data to the CFnTX register, and start transmission.

(5) When transmission is started, output the serial clock to the $\overline{SCKFn}$ pin, and output the transmit data from the SOFn pin in synchronization with the serial clock.

(6) When transmission of the transfer data length set with the CFnCTL2 register is completiond, stop the serial clock output and transmit data output, generate the reception completion interrupt request signal (INTCFnR) at the last edge of the serial clock, and clear the CFnTSF bit to 0.

(7) To continue transmission, start the next transmission by writing the transmit data to the CFnTX register again after the INTCFnR signal is generated.

(8) To end transmission, write CFnCTL0.CFnPWR bit = 0 and CFnCTL0.CFnTXE bit = 0.

**Remark**    n = 0 to 4

**18.6.2  Single transfer mode (master mode, reception mode)**

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$ (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 000), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)

**(1)  Operation flow**



START

(1), (2), (3)
CFnCTL1 register ← 00H
CFnCTL2 register ← 00H
CFnCTL0 register ← A1H

(4)
CFnRX register
dummy read

(5)
Start reception

(6)
INTCFnR interrupt
generated?          No

Yes

Reception completed?          No (7)

Yes                 Read CFnRX register

(8)
CFnSCE bit = 0
(CFnCTL0)

(9)
Read CFnRX register

(10)
CFnCTL0 register ← 00H

END

**Remarks 1.** The broken lines indicate the hardware processing.
**2.** The numbers in this figure correspond to the processing numbers in **(2)  Operation timing**.
**3.** n = 0 to 4

**(2) Operation timing**



(1) Write 00H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$, and master mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write A1H to the CFnCTL0 register, and select the reception mode and MSB first at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) Set the CFnSTR.CFnTSF bit to 1 by performing a dummy read of the CFnRX register, and start reception.

(5) When reception is started, output the serial clock to the $\overline{SCKFn}$ pin, and capture the receive data of the SIFn pin in synchronization with the serial clock.

(6) When reception of the transfer data length set with the CFnCTL2 register is completed, stop the serial clock output and data capturing, generate the reception completion interrupt request signal (INTCFnR) at the last edge of the serial clock, and clear the CFnTSF bit to 0.

(7) To continue reception, read the CFnRX register while keeping the CFnCTL0.CFnSCE bit = 1 after the INTCFnR signal is generated.

(8) To read the CFnRX register without starting the next reception, write the CFnSCE bit = 0.

(9) Read the CFnRX register.

(10) To end reception, write CFnCTL0.CFnPWR bit = 0 and CFnCTL0.CFnRXE bit = 0.

**Remark**   n = 0 to 4

### 18.6.3  Single transfer mode (master mode, transmission/reception mode)

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$ (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 000), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)

#### (1)  Operation flow

```
                          ┌─────────────┐
                          │    START    │
                          └──────┬──────┘
                                 │
                    ┌────────────────────────────┐
   (1), (2), (3)    │ CFnCTL1 register ← 07H      │
                    │ CFnCTL2 register ← 00H      │
                    │ CFnCTL0 register ← E1H      │
                    └────────────┬───────────────┘
                                 │◄──────────────────────┐
                    ┌────────────────────────────┐       │
      (4)           │    Write CFnTX register     │       │
                    └────────────┬───────────────┘       │
                    ┌ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ┐       │
      (5)           │  Start transmission/reception │     │
                    └ ─ ─ ─ ─ ─ ─┬─ ─ ─ ─ ─ ─ ─ ┘       │
                                 │◄──────────┐            │
                              ╱  ╲           │            │
                            ╱      ╲    No    │            │
      (6)                 ╱ INTCFnR  ╲───────┘            │
                          ╲ interrupt ╱                    │
                            ╲generated?╱                   │
                              ╲   ╱                        │
                               │ Yes                       │
                    ┌────────────────────────────┐         │
   (7), (9)         │    Read CFnRX register      │         │
                    └────────────┬───────────────┘         │
                              ╱  ╲                          │
                            ╱      ╲         No (8)          │
                          ╱Transmission╲──────────────────┘
                          ╲ /reception ╱
                            ╲completed?╱
                              ╲   ╱
                               │ Yes
                    ┌────────────────────────────┐
      (10)          │      CFnCTL0 ← 00H          │
                    └────────────┬───────────────┘
                          ┌──────┴──────┐
                          │     END     │
                          └─────────────┘
```

**Remarks 1.** The broken lines indicate the hardware processing.
  **2.** The numbers in this figure correspond to the processing numbers in **(2)  Operation timing**.
  **3.** n = 0 to 4

**(2) Operation timing**



(1) Write 00H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$, and master mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write E1H to the CFnCTL0 register, and select the transmission/reception mode and MSB first at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) Set the CFnSTR.CFnTSF bit to 1 by writing the transmit data to the CFnTX register, and start transmission/reception.

(5) When transmission/reception is started, output the serial clock to the $\overline{SCKFn}$ pin, output the transmit data to the SOFn pin in synchronization with the serial clock, and capture the receive data of the SIFn pin.

(6) When transmission/reception of the transfer data length set by the CFnCTL2 register is completed, stop the serial clock output, transmit data output, and data capturing, generate the reception completion interrupt request signal (INTCFnR) at the last edge of the serial clock, and clear the CFnTSF bit to 0.

(7) Read the CFnRX register.

(8) To continue transmission/reception, write the transmit data to the CFnTX register again.

(9) Read the CFnRX register.

(10) To end transmission/reception, write CFnCTL0.CFnPWR bit = 0, CFnCTL0.CFnTXE bit = 0, and CFnCTL0.CFnRXE bit = 0.

**Remark**   n = 0 to 4

#### 18.6.4 Single transfer mode (slave mode, transmission mode)

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$) (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 111), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)

#### (1) Operation flow



**Remarks 1.** The broken lines indicate the hardware processing.
**2.** The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.
**3.** n = 0 to 4

**(2) Operation timing**



(1) Write 07H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$), and slave mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write C1H to the CFnCTL0 register, and select the transmission mode and MSB first at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) The CFnSTR.CFnTSF bit is set to 1 by writing the transmit data to the CFnTX register, and the device waits for a serial clock input.

(5) When a serial clock is input, output the transmit data from the SOFn pin in synchronization with the serial clock.

(6) When transmission of the transfer data length set with the CFnCTL2 register is completed, stop the serial clock input and transmit data output, generate the reception completion interrupt request signal (INTCFnR) at the last edge of the serial clock, and clear the CFnTSF bit to 0.

(7) To continue transmission, write the transmit data to the CFnTX register again after the INTCFnR signal is generated, and wait for a serial clock input.

(8) To end transmission, write CFnCTL0.CFnPWR bit = 0 and CFnCTL0.CFnTXE bit = 0.

**Remark**   n = 0 to 4

### 18.6.5  Single transfer mode (slave mode, reception mode)

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$) (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 111), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)

**(1)  Operation flow**



> **Remarks 1.** The broken lines indicate the hardware processing.
> **2.** The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.
> **3.** n = 0 to 4

**(2) Operation timing**



(1) Write 07H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$), and slave mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write A1H to the CFnCTL0 register, and select the reception mode and MSB first at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) The CFnSTR.CFnTSF bit is set to 1 by performing a dummy read of the CFnRX register, and the device waits for a serial clock input.

(5) When a serial clock is input, capture the receive data of the SIFn pin in synchronization with the serial clock.

(6) When reception of the transfer data length set with the CFnCTL2 register is completed, stop the serial clock input and data capturing, generate the reception completion interrupt request signal (INTCFnR) at the last edge of the serial clock, and clear the CFnTSF bit to 0.

(7) To continue reception, read the CFnRX register while keeping the CFnCTL0.CFnSCE bit = 1 after the INTCFnR signal is generated, and wait for a serial clock input.

(8) To end reception, write the CFnSCE bit = 0.

(9) Read the CFnRX register.

(10) To end reception, write CFnCTL0.CFnPWR bit = 0 and CFnCTL0.CFnRXE bit = 0.

**Remark**  n = 0 to 4

### 18.6.6 Single transfer mode (slave mode, transmission/reception mode)

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = external clock ($\overline{\text{SCKFn}}$) (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 111), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)

**(1)  Operation flow**



| | |
|---|---|
| START | |
| (1), (2), (3) | CFnCTL1 register ← 07H<br>CFnCTL2 register ← 00H<br>CFnCTL0 register ← E1H |
| (4) | Write CFnTX register |
| (4) | $\overline{\text{SCKFn}}$ pin input started? — No |
| (5) | Start transmission/reception |
| (6) | INTCFnR interrupt generated? — No |
| (7), (9) | Read CFnRX register |
| | Transmission/reception completed? — No (8) |
| (10) | CFnCTL0 ← 00H |
| END | |

**Remarks  1.** The broken lines indicate the hardware processing.
   **2.** The numbers in this figure correspond to the processing numbers in **(2)  Operation timing**.
   **3.**  n = 0 to 4

**(2) Operation timing**



(1) Write 07H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$), and slave mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write E1H to the CFnCTL0 register, and select the transmission/reception mode and MSB first at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) The CFnSTR.CFnTSF bit is set to 1 by writing the transmit data to the CFnTX register, and the device waits for a serial clock input.

(5) When a serial clock is input, output the transmit data to the SOFn pin in synchronization with the serial clock, and capture the receive data of the SIFn pin.

(6) When transmission/reception of the transfer data length set with the CFnCTL2 register is completed, stop the serial clock input, transmit data output, and data capturing, generate the reception completion interrupt request signal (INTCFnR) at the last edge of the serial clock, and clear the CFnTSF bit to 0.

(7) Read the CFnRX register.

(8) To continue transmission/reception, write the transmit data to the CFnTX register again, and wait for a serial clock input.

(9) Read the CFnRX register.

(10) To end transmission/reception, write CFnCTL0.CFnPWR bit = 0, CFnCTL0.CFnTXE bit = 0, and CFnCTL0.CFnRXE bit = 0.
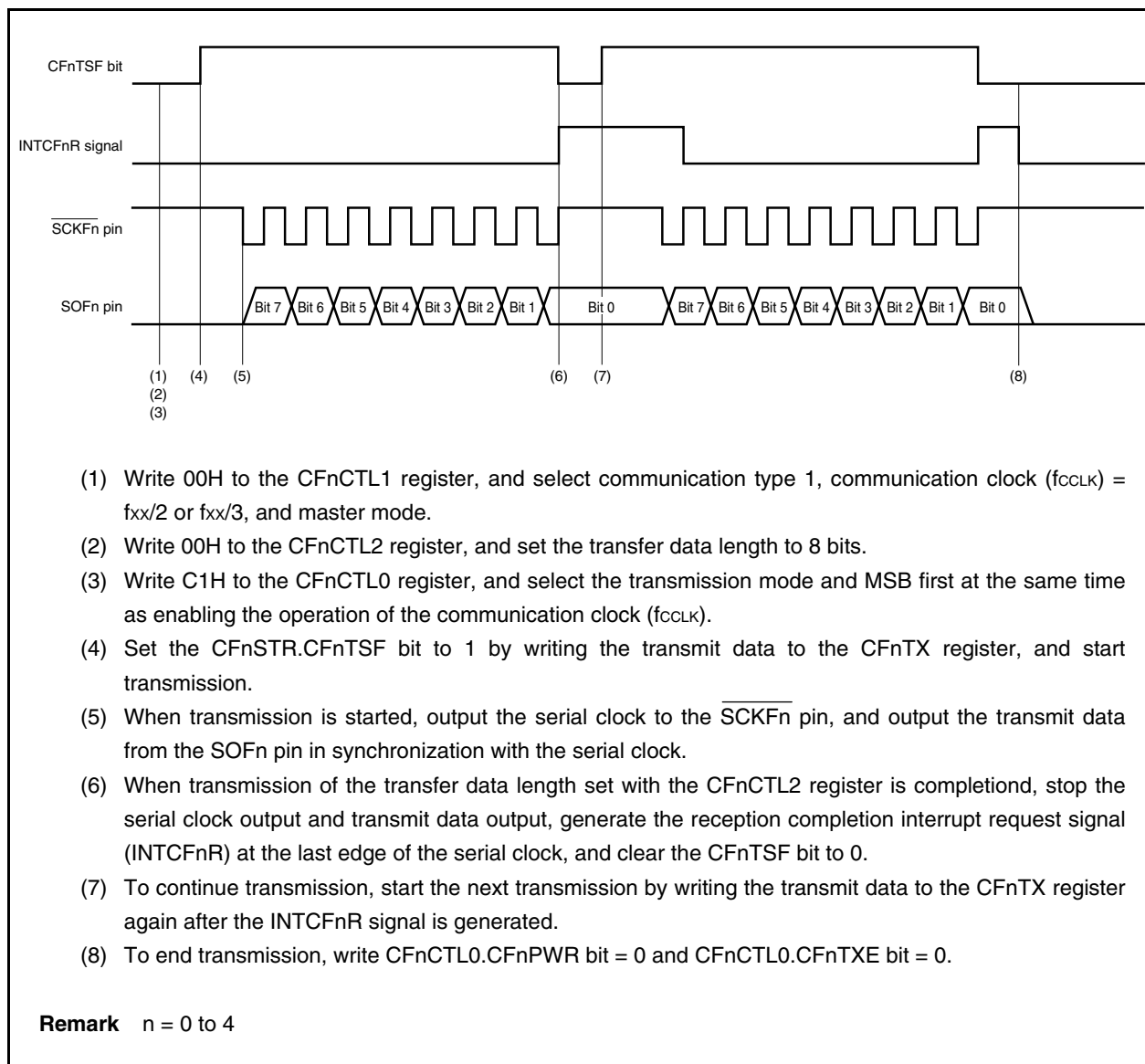
**Remark**   n = 0 to 4

### 18.6.7  Continuous transfer mode (master mode, transmission mode)

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$ (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 000), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)
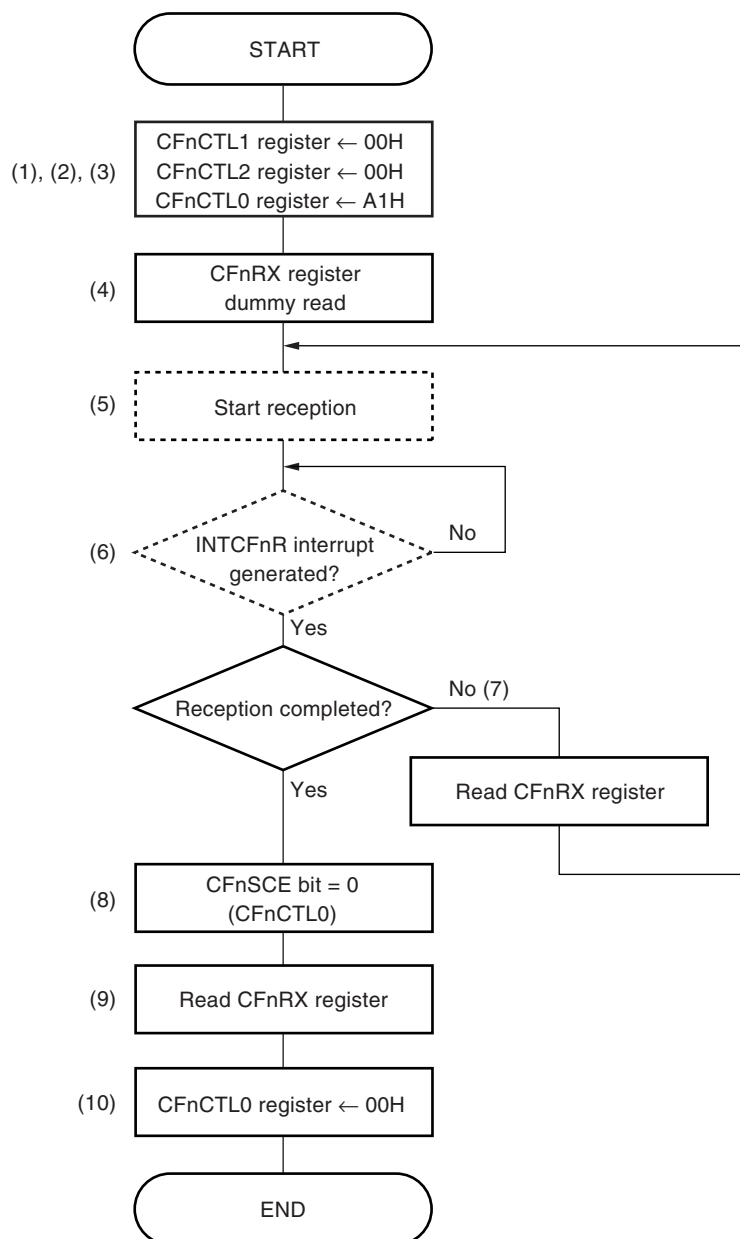
**(1)  Operation flow**



Remarks **1.** The broken lines indicate the hardware processing.

**2.** The numbers in this figure correspond to the processing numbers in **(2)  Operation timing**.

**3.** n = 0 to 4

**(2) Operation timing**



(1) Write 00H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$, and master mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write C3H to the CFnCTL0 register, and select the transmission mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) Set the CFnSTR.CFnTSF bit to 1 by writing the transmit data to the CFnTX register, and start transmission.

(5) When transmission is started, output the serial clock to the $\overline{SCKFn}$ pin, and output the transmit data from the SOFn pin in synchronization with the serial clock.

(6) When transfer of the transmit data from the CFnTX register to the shift register is completed and writing to the CFnTX register is enabled, the transmission enable interrupt request signal (INTCFnT) is generated.

(7) To continue transmission, write the transmit data to the CFnTX register again after the INTCFnT signal is generated.

(8) When a new transmit data is written to the CFnTX register before communication completion, the next communication is started following communication completion.

(9) The transfer of the transmit data from the CFnTX register to the shift register is completed and the INTCFnT signal is generated. To end continuous transmission with the current transmission, do not write to the CFnTX register.

(10) When the next transmit data is not written to the CFnTX register before transfer completion, stop the serial clock output to the $\overline{SCKFn}$ pin after transfer completion, and clear the CFnTSF bit to 0.

(11) To release the transmission enable status, write CFnCTL0.CFnPWR bit = 0 and CFnCTL0.CFnTXE bit = 0 after checking that the CFnTSF bit = 0.

**Caution   In continuous transmission mode, the reception completion interrupt request signal (INTCFnR) is not generated.**

**Remark**   n = 0 to 4

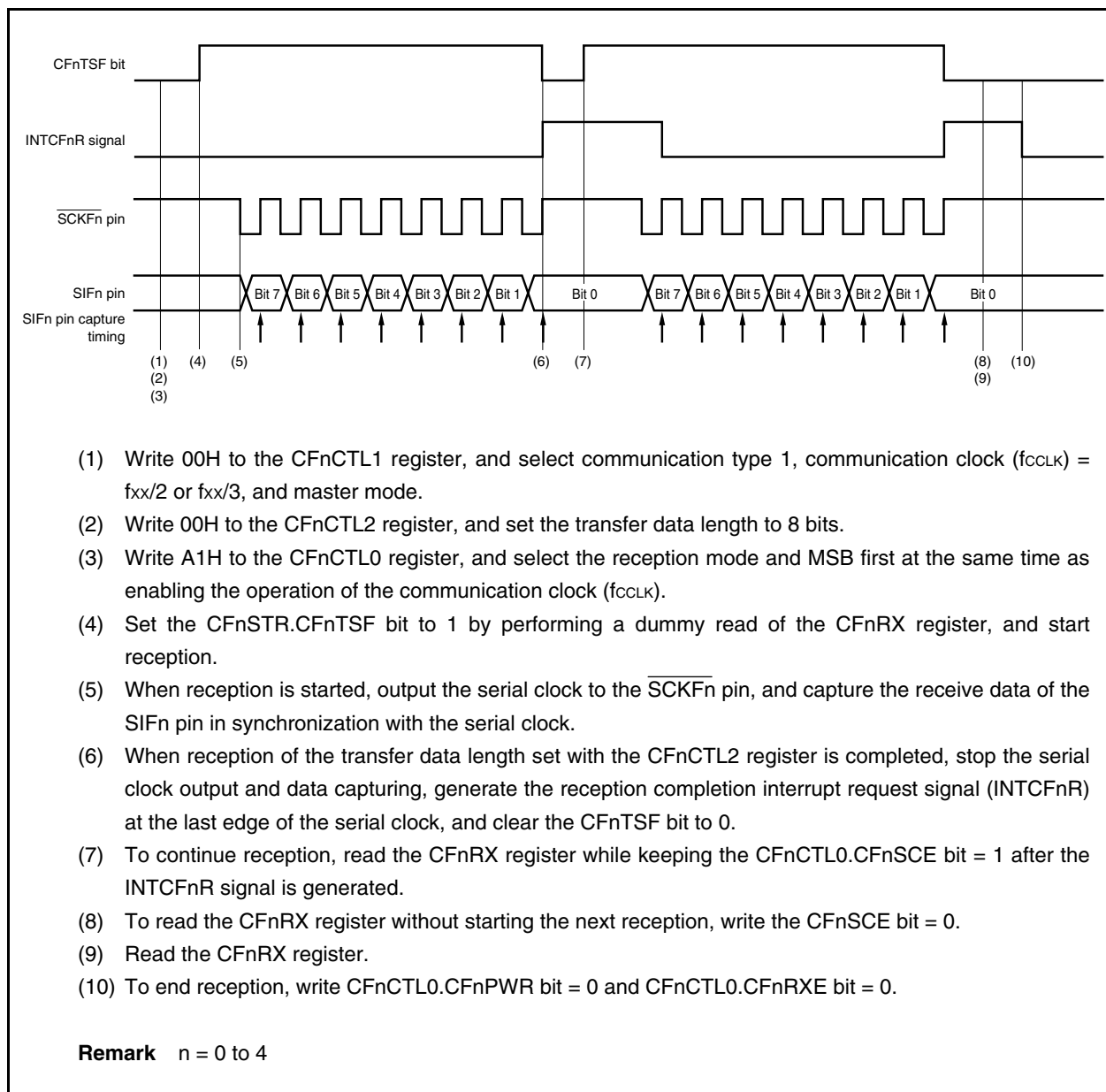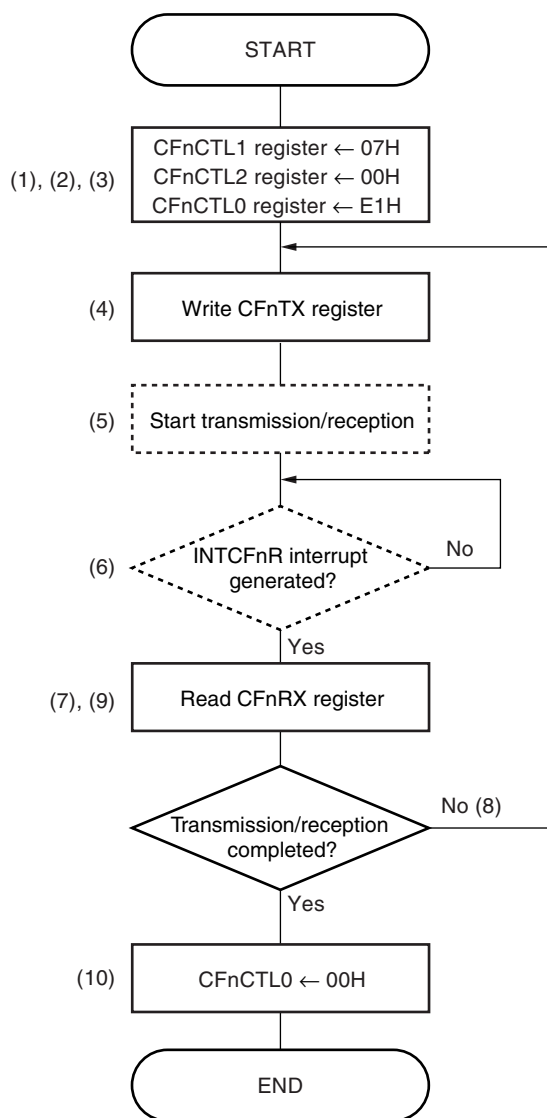**18.6.8 Continuous transfer mode (master mode, reception mode)**

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$ (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 000), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)
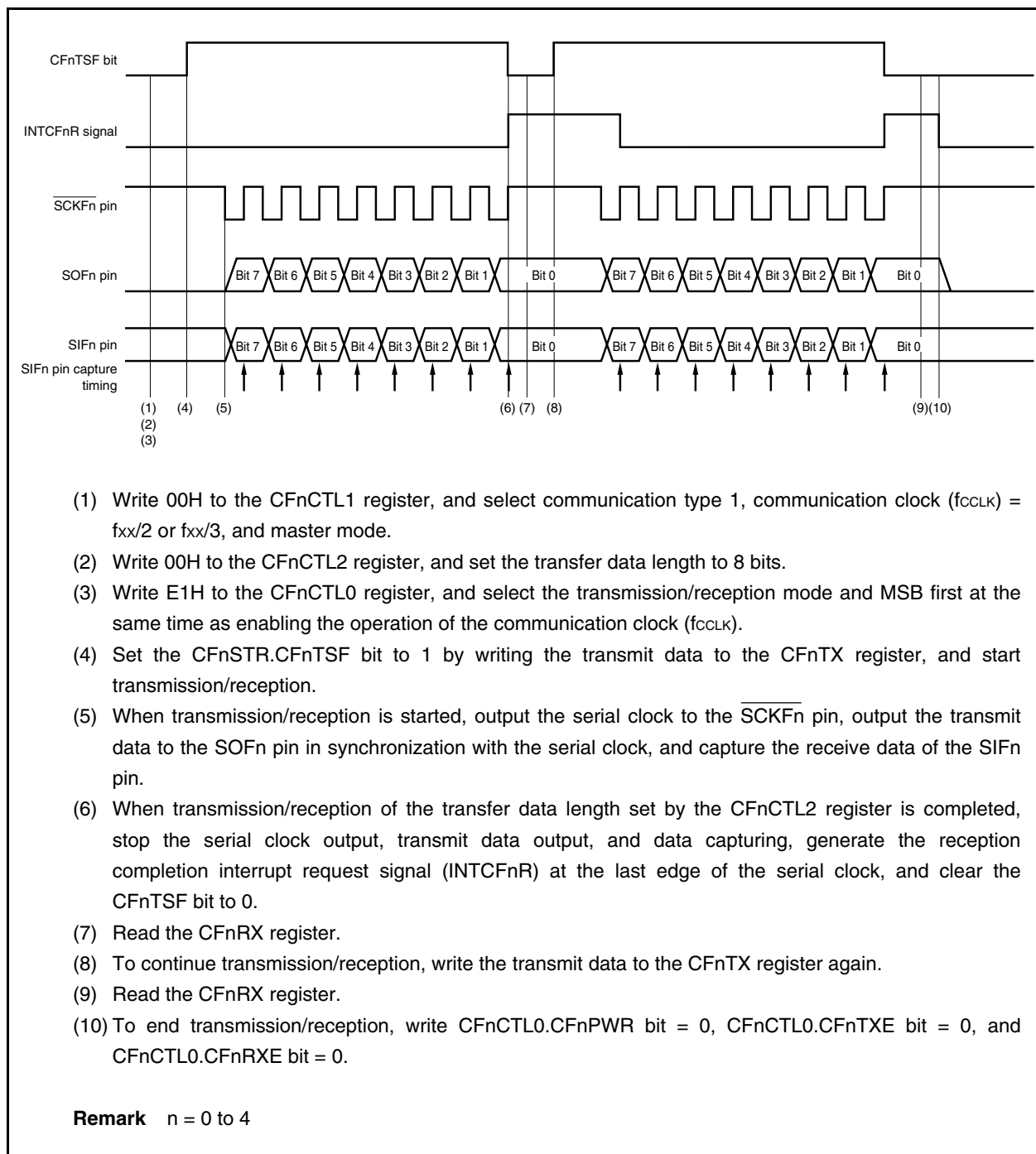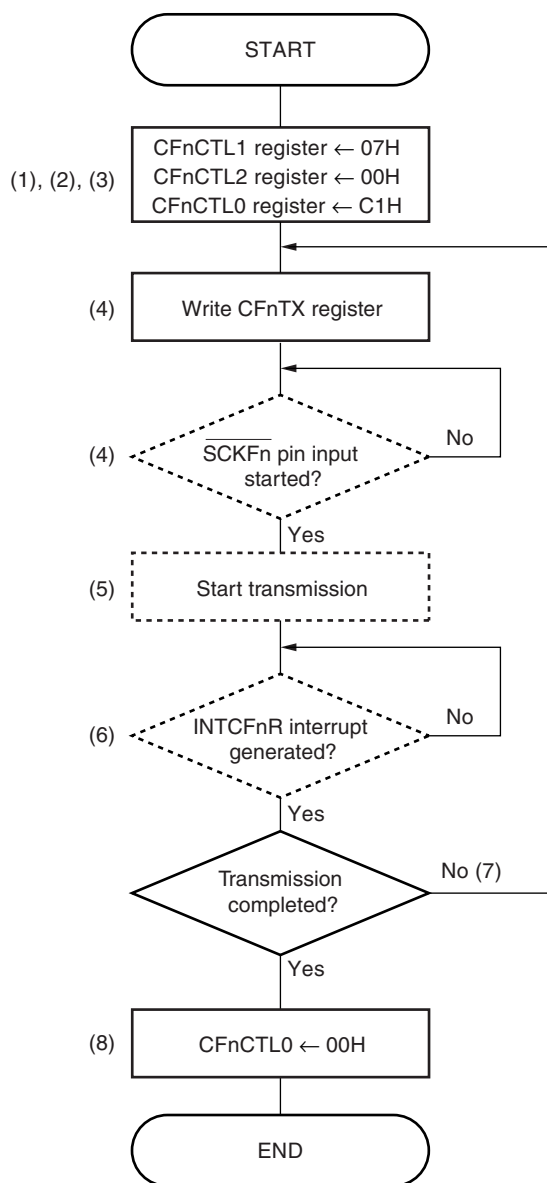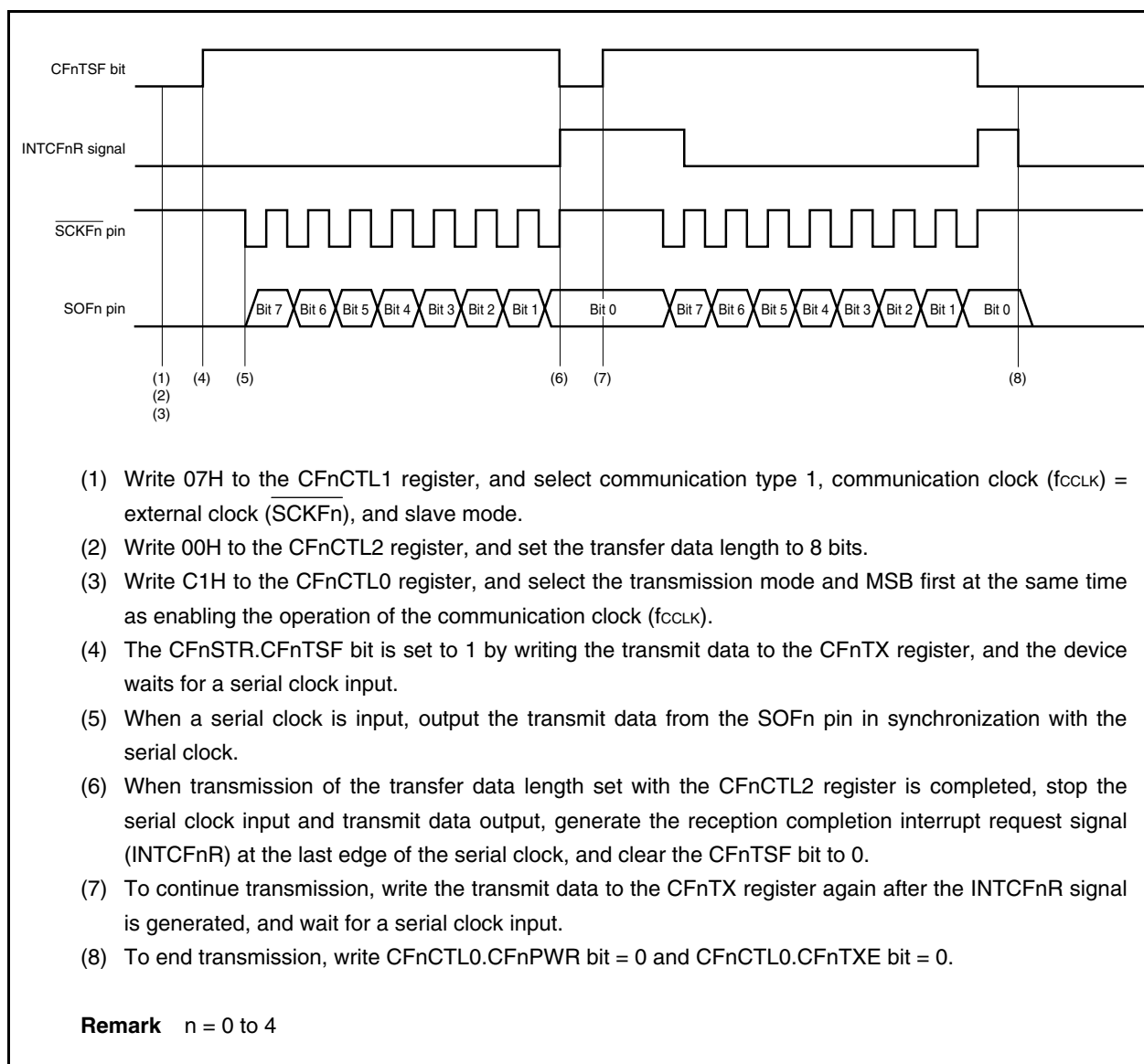
**(1)  Operation flow**



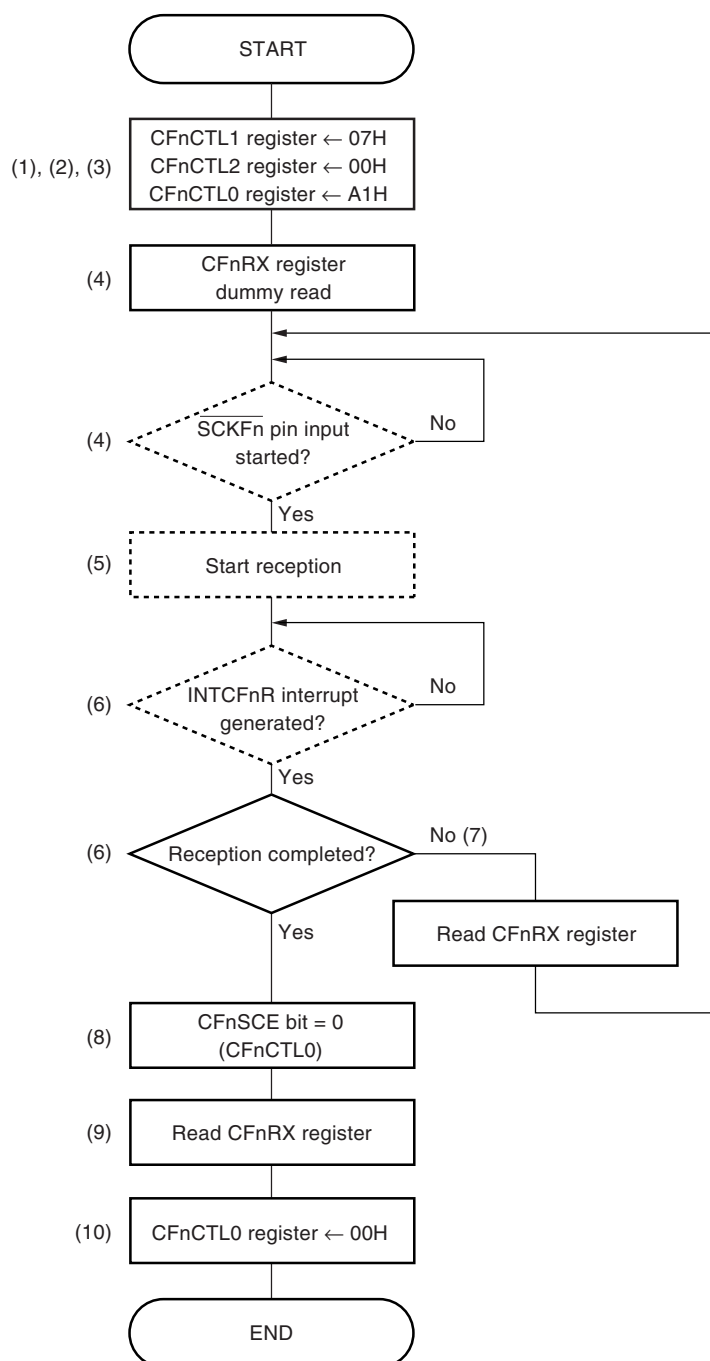**Remarks 1.** The broken lines indicate the hardware processing.
**2.** The numbers in this figure correspond to the processing numbers in **(2)  Operation timing**.
**3.** n = 0 to 4

**(2) Operation timing**



(1) Write 00H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = $f_{XX}/2$, and master mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write A3H to the CFnCTL0 register, and select the reception mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) Set the CFnSTR.CFnTSF bit to 1 by performing a dummy read of the CFnRX register, and start reception.

(5) When reception is started, output the serial clock to the $\overline{SCKFn}$ pin, and capture the receive data of the SIFn pin in synchronization with the serial clock.

(6) When reception is completed, the reception completion interrupt request signal (INTCFnR) is generated, and reading of the CFnRX register is enabled.

(7) When the CFnCTL0.CFnSCE bit = 1 upon communication completion, the next communication is started following communication completion.

(8) To end continuous reception with the current reception, write the CFnSCE bit = 0.

(9) Read the CFnRX register.

(10) When reception is completed, the INTCFnR signal is generated, and reading of the CFnRX register is enabled. When the CFnSCE bit = 0 is set before communication completion, stop the serial clock output to the $\overline{SCKFn}$ pin, and clear the CFnTSF bit to 0, to end the receive operation.

(11) Read the CFnRX register.

(12) If an overrun error occurs, write the CFnSTR.CFnOVE bit = 0, and clear the error flag.

(13) To release the reception enable status, write CFnCTL0.CFnPWR bit = 0 and CFnCTL0.CFnRXE bit = 0 after checking that the CFnTSF bit = 0.

**Remark**   n = 0 to 4

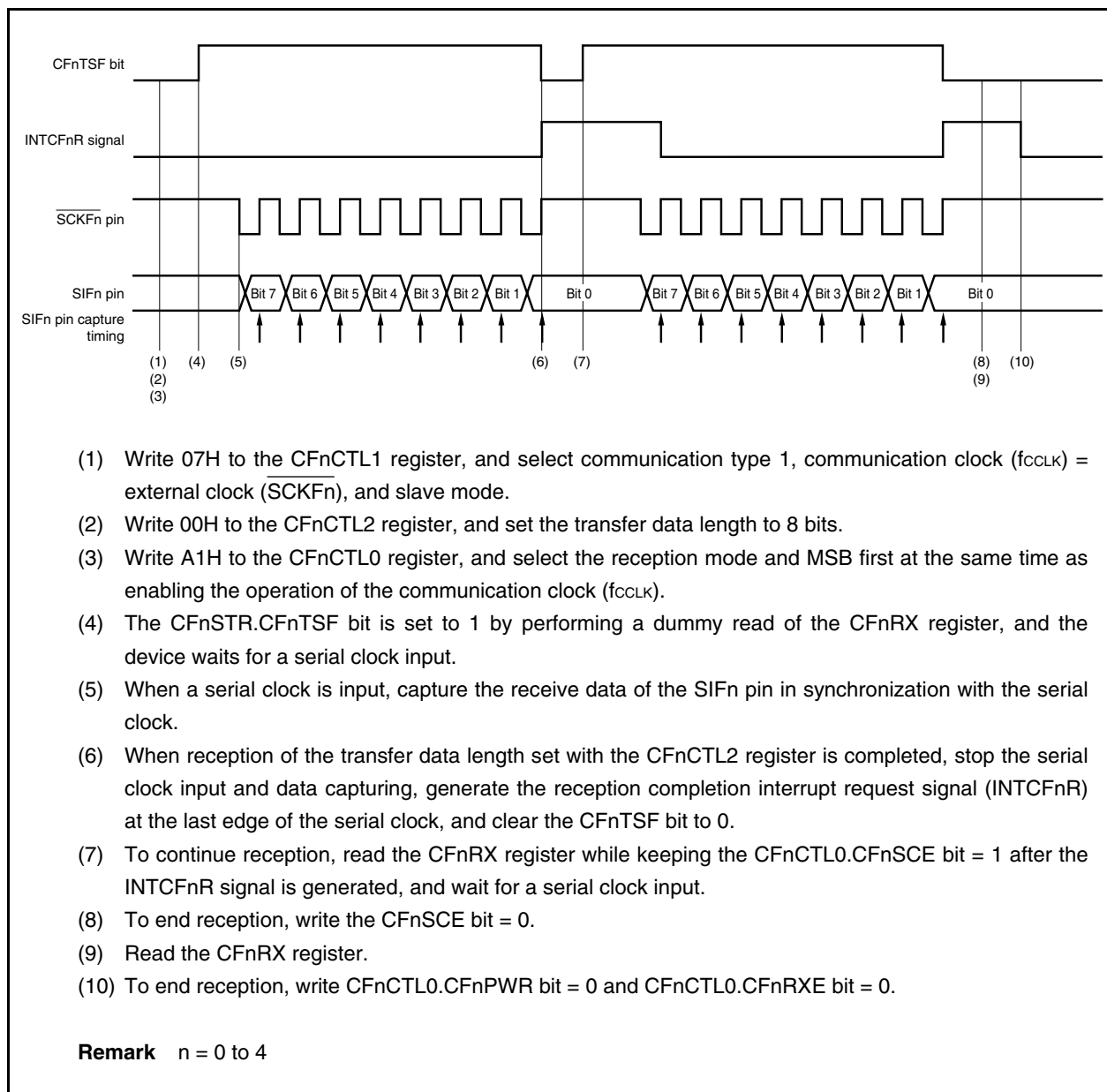**18.6.9 Continuous transfer mode (master mode, transmission/reception mode)**

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$ (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 000), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)

**(1) Operation flow**



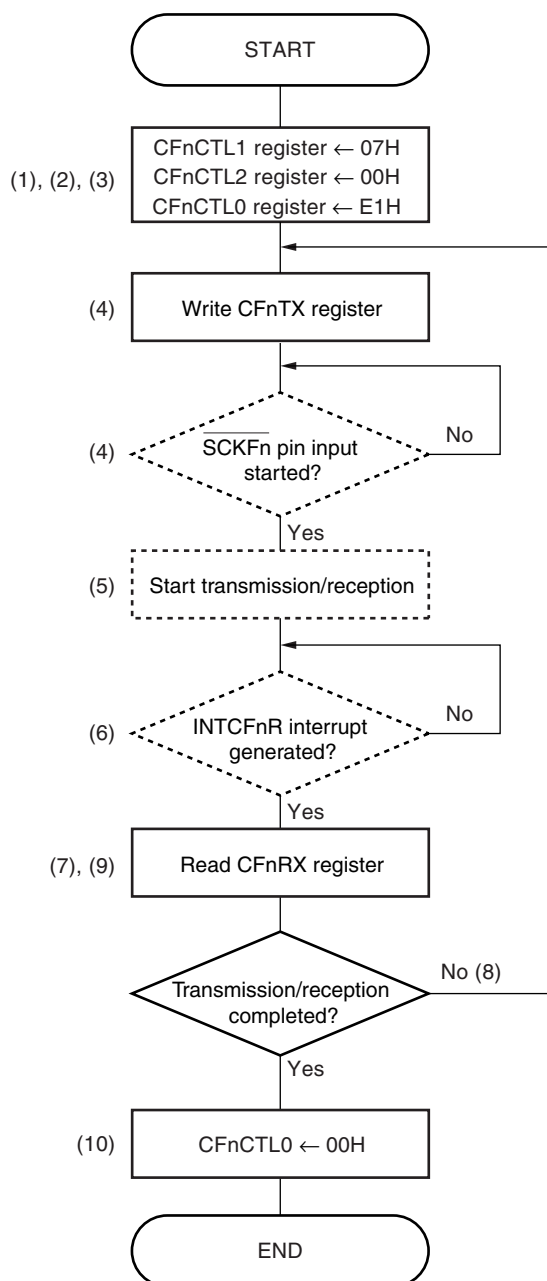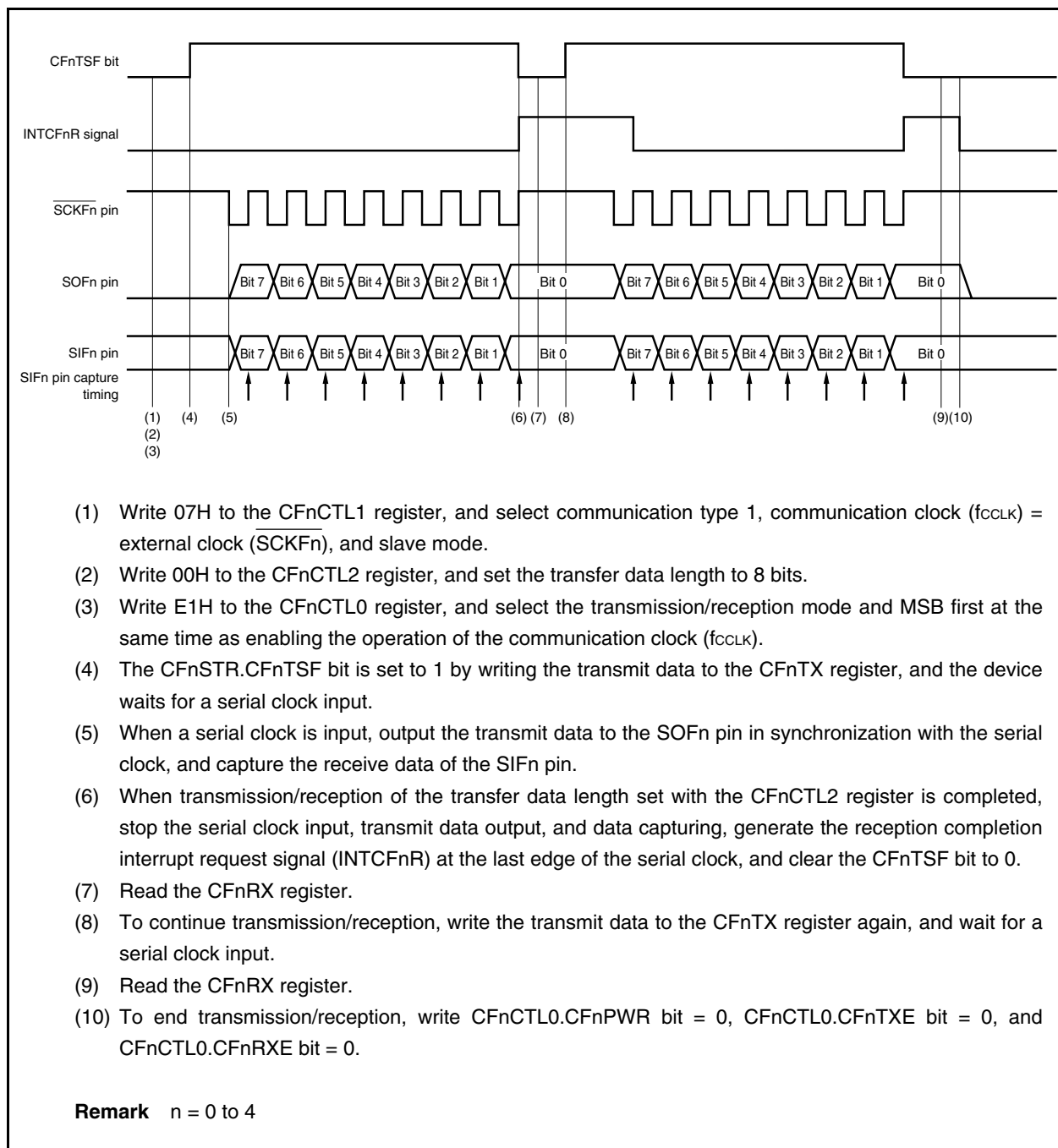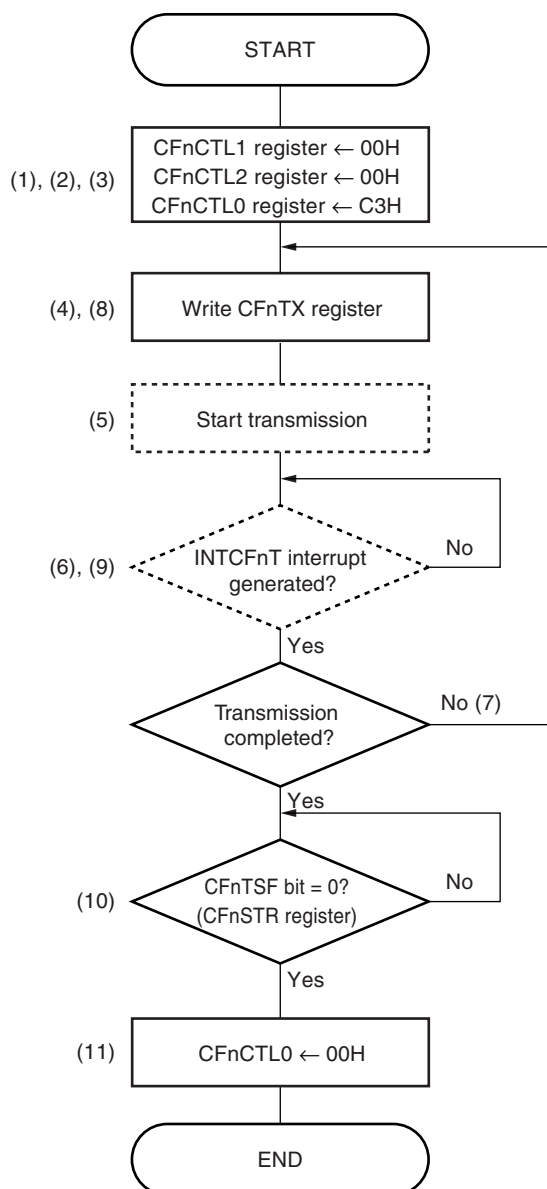**Remarks 1.** The broken lines indicate the hardware processing.
      **2.** The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.
      **3.** n = 0 to 4

**(2) Operation timing**

(1/2)



(1) Write 00H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = $f_{XX}/2$ or $f_{XX}/3$, and master mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write E3H to the CFnCTL0 register, and select the transmission/reception mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) Set the CFnSTR.CFnTSF bit to 1 by writing the transmit data to the CFnTX register, and start transmission/reception.

(5) When transmission/reception is started, output the serial clock to the $\overline{SCKFn}$ pin, output the transmit data to the SOFn pin in synchronization with the serial clock, and capture the receive data of the SIFn pin.

(6) When transfer of the transmit data from the CFnTX register to the shift register is completed and writing to the CFnTX register is enabled, the transmission enable interrupt request signal (INTCFnT) is generated.

(7) To continue transmission/reception, write the transmit data to the CFnTX register again after the INTCFnT signal is generated.

(8) When one transmission/reception is completed, the reception completion interrupt request signal (INTCFnR) is generated, and reading of the CFnRX register is enabled.

(9) When a new transmit data is written to the CFnTX register before communication completion, the next communication is started following communication completion.

(10) Read the CFnRX register.

**Remark**   n = 0 to 4

(2/2)

(11) The transfer of the transmit data from the CFnTX register to the shift register is completed and the INTCFnT signal is generated.   To end continuous transmission/reception with the current transmission/reception, do not write to the CFnTX register.

(12) When the next transmit data is not written to the CFnTX register before transfer completion, stop the serial clock output to the $\overline{\text{SCKFn}}$ pin after transfer completion, and clear the CFnTSF bit to 0.

(13) When the reception error interrupt request signal (INTCFnR) is generated, read the CFnRX register.

(14) If an overrun error occurs, write CFnSTR.CFnOVE bit = 0, and clear the error flag.

(15) To release the transmission/reception enable status, write CFnCTL0.CFnPWR bit = 0, CFnCTL0.CFnTXE bit = 0, and CFnCTL0.CFnRXE bit = 0 after checking that the CFnTSF bit = 0.


**Remark**   n = 0 to 4

### 18.6.10 Continuous transfer mode (slave mode, transmission mode)

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$) (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 111), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)

### (1) Operation flow



**Remarks 1.** The broken lines indicate the hardware processing.
**2.** The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.
**3.** n = 0 to 4

**(2) Operation timing**



(1) Write 07H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$), and slave mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write C3H to the CFnCTL0 register, and select the transmission mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) The CFnSTR.CFnTSF bit is set to 1 by writing the transmit data to the CFnTX register, and the device waits for a serial clock input.

(5) When a serial clock is input, output the transmit data from the SOFn pin in synchronization with the serial clock.

(6) When transfer of the transmit data from the CFnTX register to the shift register is completed and writing to the CFnTX register is enabled, the transmission enable interrupt request signal (INTCFnT) is generated.

(7) To continue transmission, write the transmit data to the CFnTX register again after the INTCFnT signal is generated.

(8) When a serial clock is input following completion of the transmission of the transfer data length set with the CFnCTL2 register, continuous transmission is started.

(9) When transfer of the transmit data from the CFnTX register to the shift register is completed and writing to the CFnTX register is enabled, the INTCFnT signal is generated. To end continuous transmission with the current transmission, do not write to the CFnTX register.

(10) When the clock of the transfer data length set with the CFnCTL2 register is input without writing to the CFnTX register, clear the CFnTSF bit to 0 to end transmission.

(11) To release the transmission enable status, write CFnCTL0.CFnPWR bit = 0 and CFnCTL0.CFnTXE bit = 0 after checking that the CFnTSF bit = 0.

**Caution** **In continuous transmission mode, the reception completion interrupt request signal (INTCFnR) is not generated.**
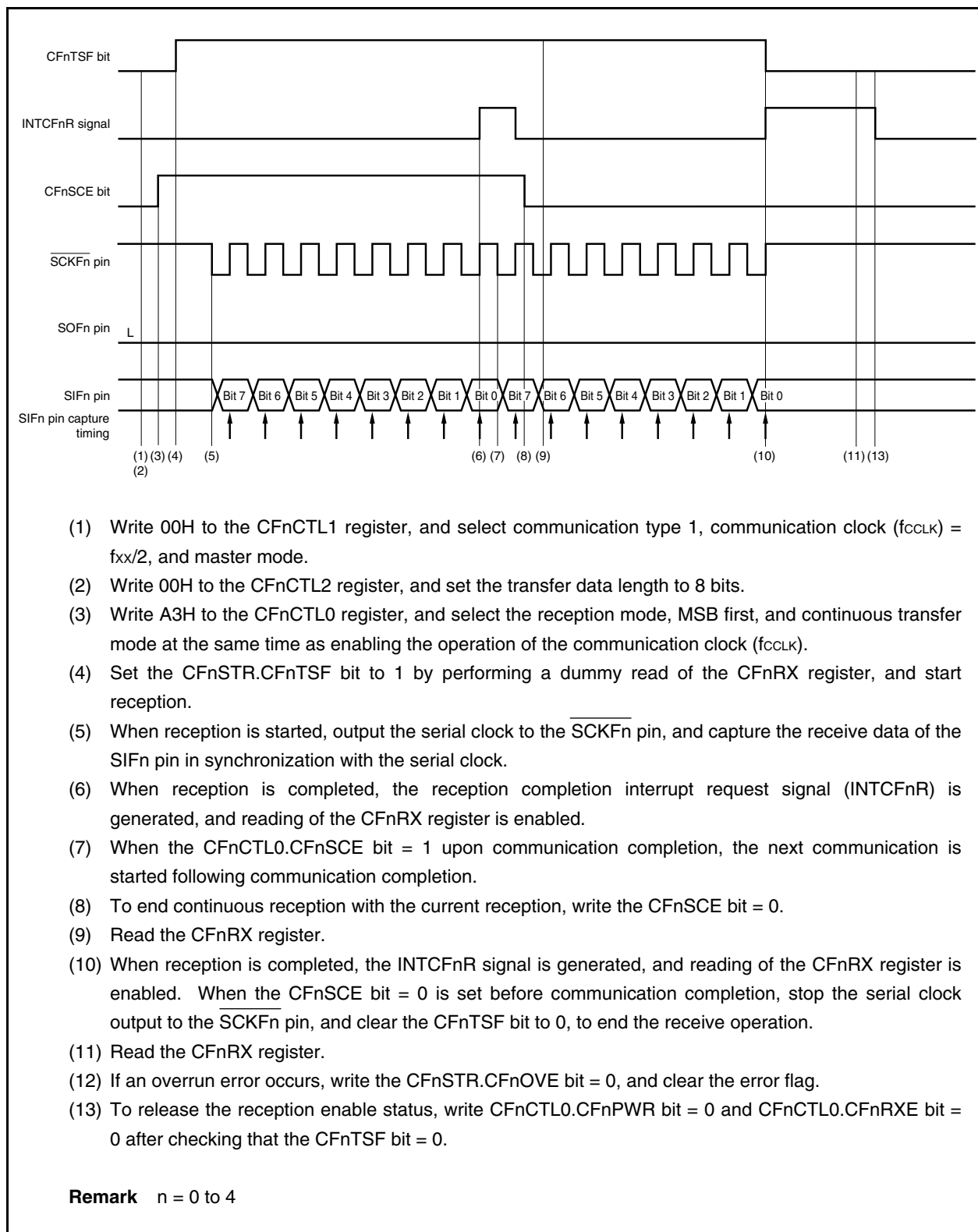
**Remark** n = 0 to 4

**18.6.11  Continuous transfer mode (slave mode, reception mode)**

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$) (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 111), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)
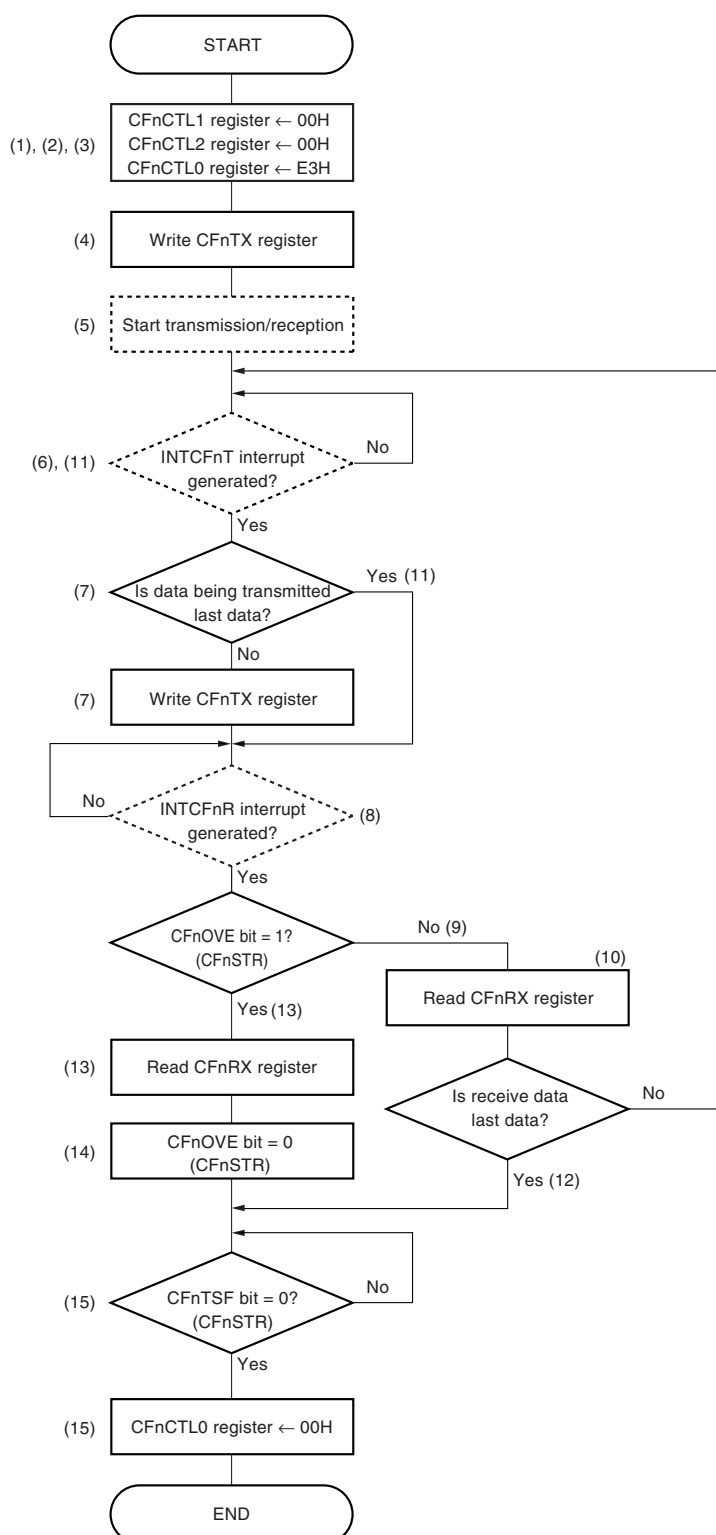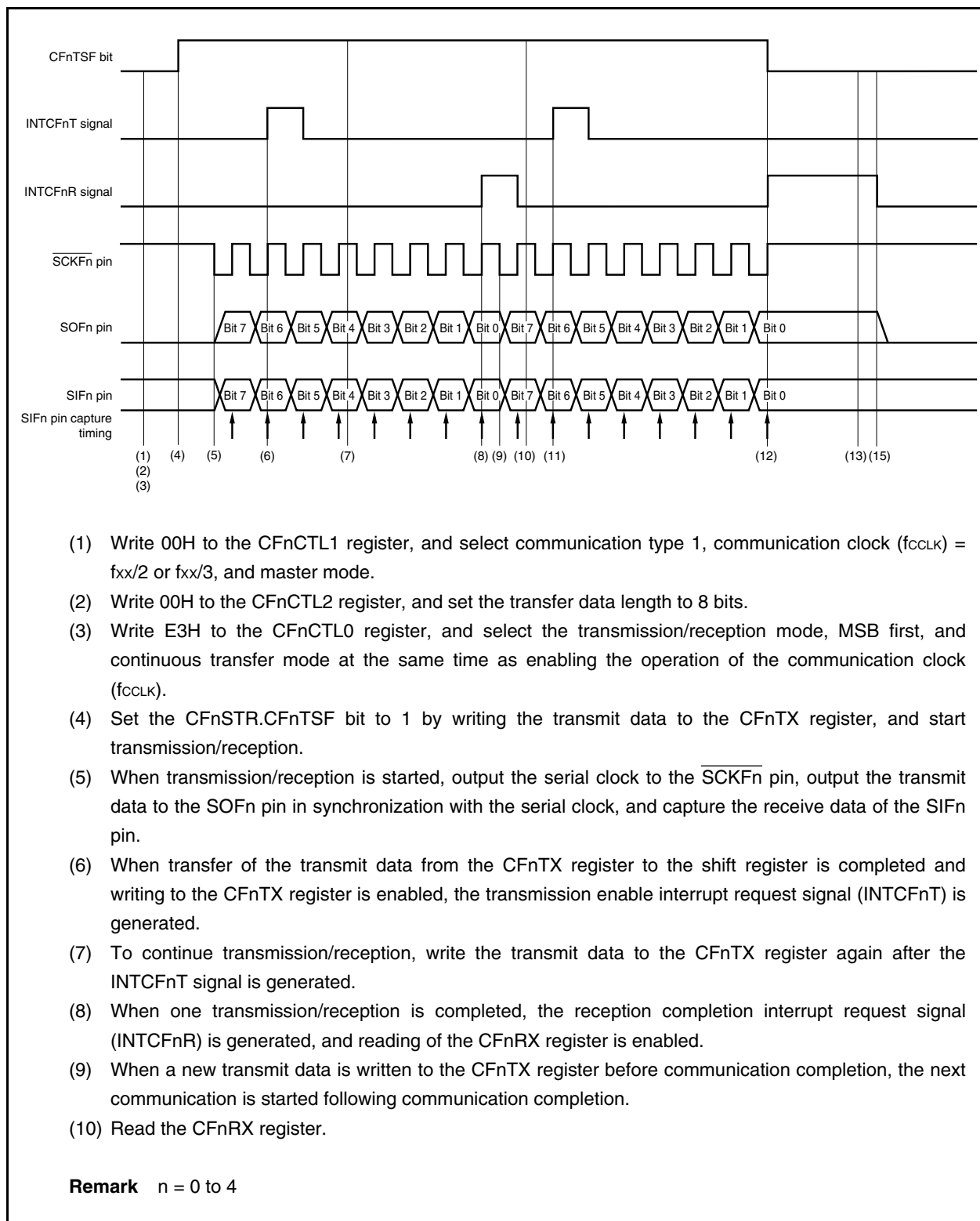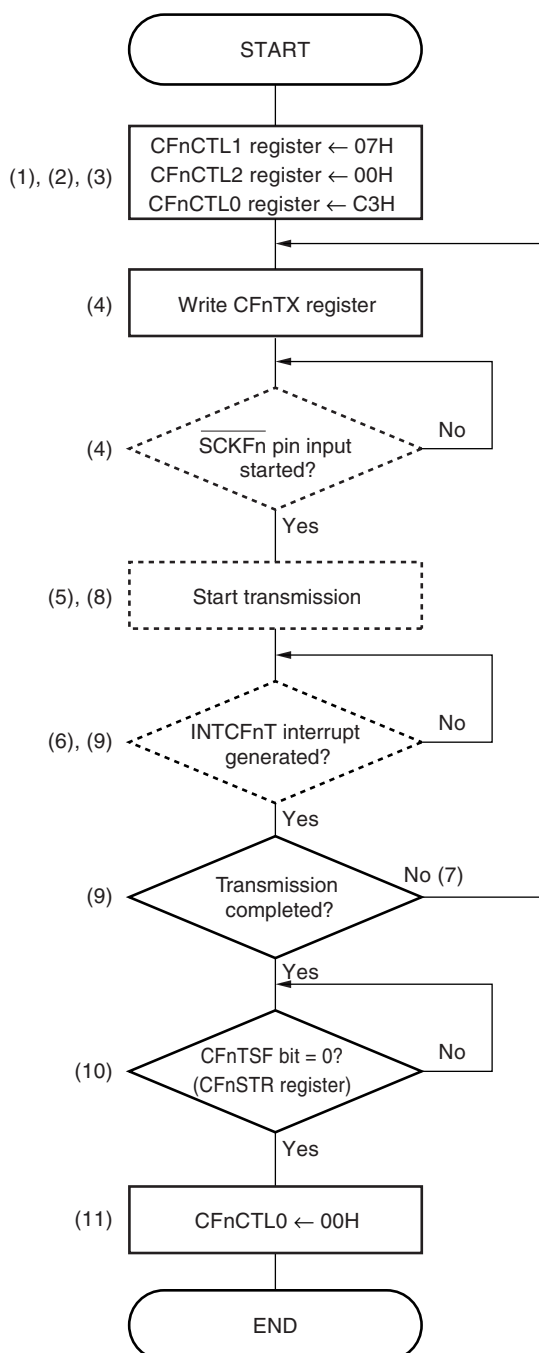
**(1) Operation flow**



**Remarks 1.** The broken lines indicate the hardware processing.

**2.** The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

**3.** n = 0 to 4

**(2) Operation timing**



(1) Write 07H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$), and slave mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write A3H to the CFnCTL0 register, and select the reception mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) The CFnSTR.CFnTSF bit is set to 1 by performing a dummy read of the CFnRX register, and the device waits for a serial clock input.

(5) When a serial clock is input, capture the receive data of the SIFn pin in synchronization with the serial clock.

(6) When reception is completed, the reception completion interrupt request signal (INTCFnR) is generated, and reading of the CFnRX register is enabled.

(7) When a serial clock is input in the CFnCTL0.CFnSCE bit = 1 status, continuous reception is started.

(8) To end continuous reception with the current reception, write the CFnSCE bit = 0.

(9) Read the CFnRX register.

(10) When reception is completed, the INTCFnR signal is generated, and reading of the CFnRX register is enabled. When CFnSCE bit = 0 is set before communication completion, clear the CFnTSF bit to 0 to end the receive operation.

(11) Read the CFnRX register.

(12) If an overrun error occurs, write CFnSTR.CFnOVE bit = 0, and clear the error flag.

(13) To release the reception enable status, write CFnCTL0.CFnPWR bit = 0 and CFnCTL0.CFnRXE bit = 0 after checking that the CFnTSF bit = 0.

**Remark**    n = 0 to 4

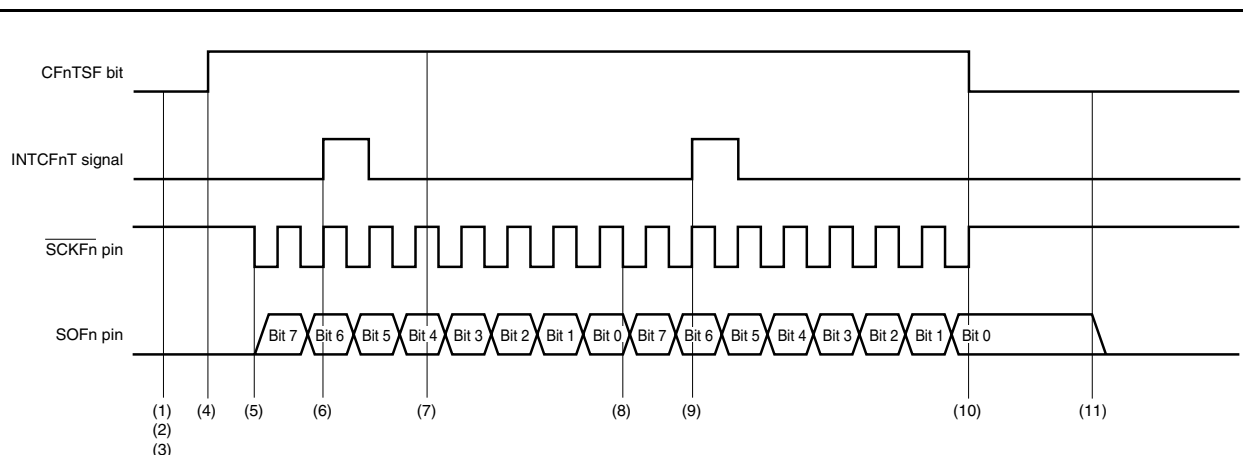**18.6.12  Continuous transfer mode (slave mode, transmission/reception mode)**

MSB first (CFnCTL0.CFnDIR bit = 0), communication type 1 (CFnCTL1.CFnCKP and CFnCTL1.CFnDAP bits = 00), communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$) (CFnCTL1.CFnCKS2 to CFnCTL1.CFnCKS0 bits = 111), transfer data length = 8 bits (CFnCTL2.CFnCL3 to CFnCTL2.CFnCL0 bits = 0000)

**(1)  Operation flow**

```
                              START

(1), (2), (3)    CFnCTL1 register ← 07H
                 CFnCTL2 register ← 00H
                 CFnCTL0 register ← E3H

(4)              Write CFnTX register

(4)              SCKFn pin input          No
                 started?

                 Yes

(5)              Start transmission/reception

(6), (11)        INTCFnT interrupt        No
                 generated?

                 Yes

(7)              Is data being transmitted    Yes (11)
                 last data?

                 No

(7)              Write CFnTX register

            No   INTCFnR interrupt         (8)
                 generated?

                 Yes

                 CFnOVE bit = 1?           No (9)
                 (CFnSTR)

                 Yes (13)                      (10)
                                           Read CFnRX register

(13)             Read CFnRX register

                                           Is receive data       No
(14)             CFnOVE bit = 0            last data?
                 (CFnSTR)
                                           Yes (12)

(15)             CFnTSF bit = 0?           No
                 (CFnSTR)

                 Yes

(15)             CFnCTL0 register ← 00H

                              END
```
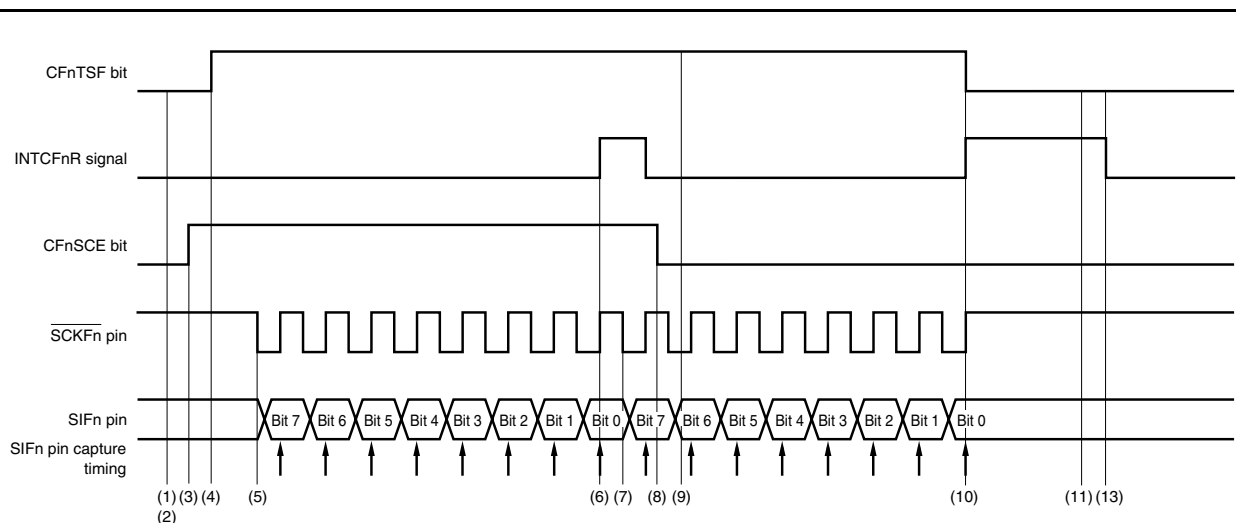
**Remarks 1.** The broken lines indicate the hardware processing.
 **2.** The numbers in this figure correspond to the processing numbers in **(2)  Operation timing**.
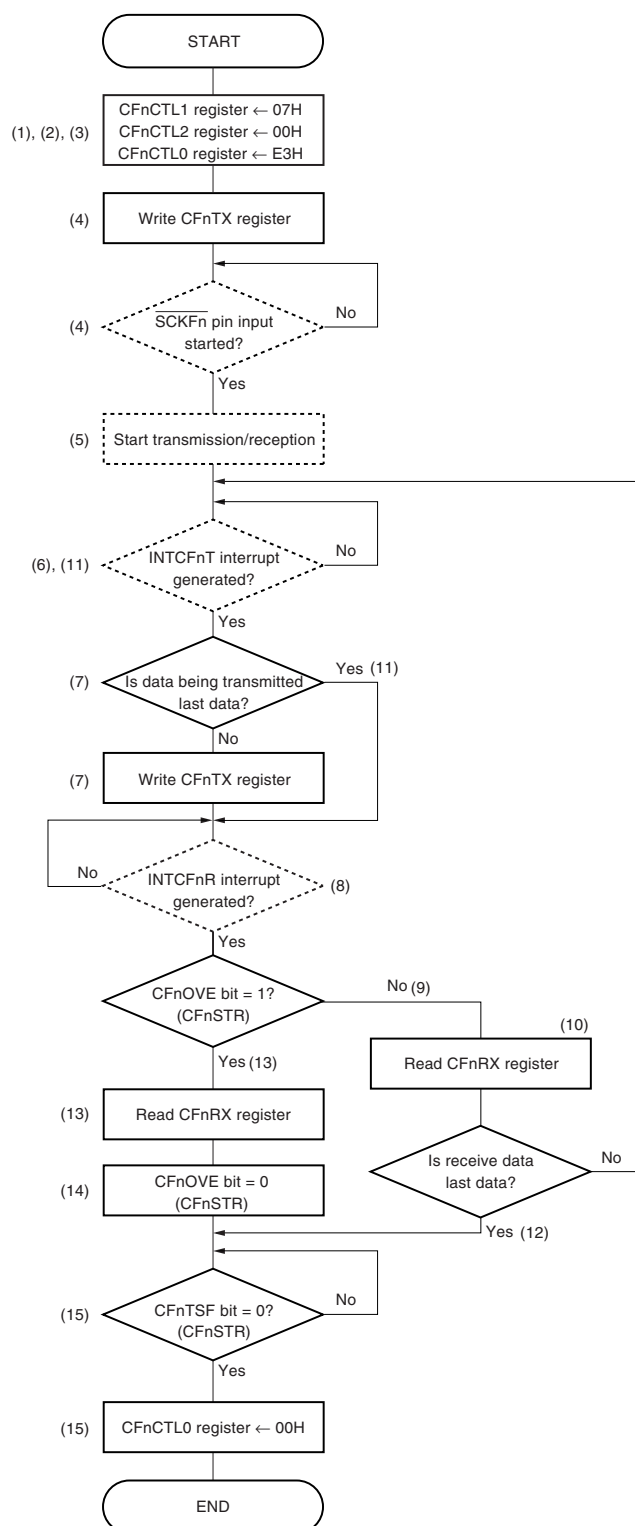 **3.** n = 0 to 4

**(2) Operation timing**

(1/2)



(1) Write 07H to the CFnCTL1 register, and select communication type 1, communication clock ($f_{CCLK}$) = external clock ($\overline{SCKFn}$), and slave mode.

(2) Write 00H to the CFnCTL2 register, and set the transfer data length to 8 bits.

(3) Write E3H to the CFnCTL0 register, and select the transmission/reception mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock ($f_{CCLK}$).

(4) The CFnSTR.CFnTSF bit is set to 1 by writing the transmit data to the CFnTX register, and the device waits for a serial clock input.

(5) When a serial clock is input, output the transmit data to the SOFn pin in synchronization with the serial clock, and capture the receive data of the SIFn pin.

(6) When transfer of the transmit data from the CFnTX register to the shift register is completed and writing to the CFnTX register is enabled, the transmission enable interrupt request signal (INTCFnT) is generated.

(7) To continue transmission, write the transmit data to the CFnTX register again after the INTCFnT signal is generated.

(8) When reception of the transfer data length set with the CFnCTL2 register is completed, the reception completion interrupt request signal (INTCFnR) is generated, and reading of the CFnRX register is enabled.

(9) When a serial clock is input continuously, continuous transmission/reception is started.

(10) Read the CFnRX register.

(11) When transfer of the transmit data from the CFnTX register to the shift register is completed and writing to the CFnTX register is enabled, the INTCFnT signal is generated. To end continuous transmission/reception with the current transmission/reception, do not write to the CFnTX register.

**Remark**    n = 0 to 4

(12) When the clock of the transfer data length set with the CFnCTL2 register is input without writing to the CFnTX register, the INTCFnR signal is generated.  Clear the CFnTSF bit to 0 to end transmission/reception.

(13) When the INTCFnR signal is generated, read the CFnRX register.

(14) If an overrun error occurs, write CFnSTR.CFnOVE bit = 0, and clear the error flag.

(15) To release the transmission/reception enable status, write CFnCTL0.CFnPWR bit = 0, CFnCTL0.CFnTXE bit = 0, and CFnCTL0.CFnRXE bit = 0 after checking that the CFnTSF bit = 0.

**Remark**    n = 0 to 4

**18.6.13  Reception error**

When transfer is performed with reception enabled (CFnCTL0.CFnRXE bit = 1) in the continuous transfer mode, the reception completion interrupt request signal (INTCFnR) is generated again when the next receive operation is completed before the CFnRX register is read after the INTCFnR signal is generated, and the overrun error flag (CFnSTR.CFnOVE) is set to 1.

Even if an overrun error has occurred, the previous receive data is lost since the CFnRX register is updated. Even if a reception error has occurred, the INTCFnR signal is generated again upon the next reception completion if the CFnRX register is not read.

To avoid an overrun error, complete reading the CFnRX register by one half clock before sampling the last bit of the next receive data from the INTCFnR signal generation.

**(1) Operation timing**
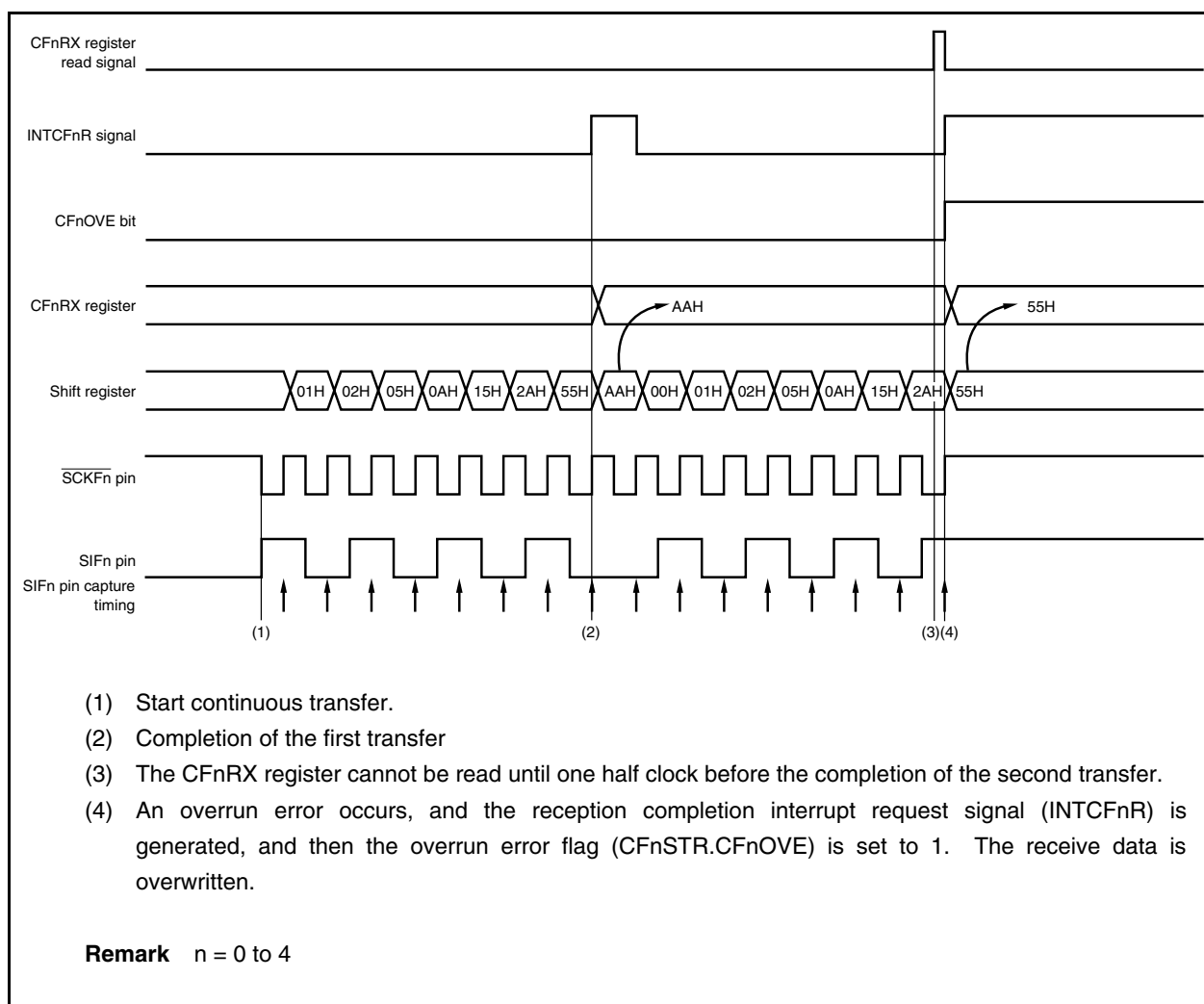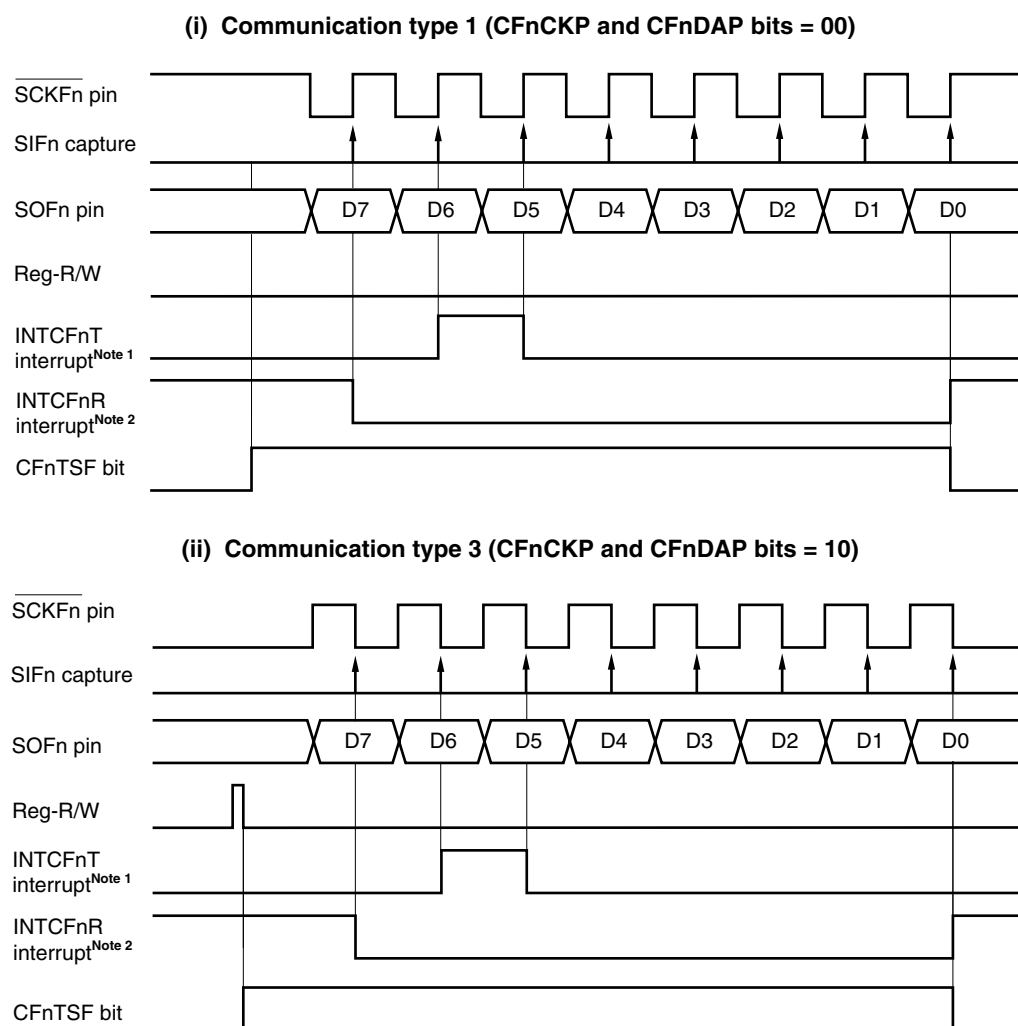


(1)  Start continuous transfer.

(2)  Completion of the first transfer

(3)  The CFnRX register cannot be read until one half clock before the completion of the second transfer.

(4)  An overrun error occurs, and the reception completion interrupt request signal (INTCFnR) is generated, and then the overrun error flag (CFnSTR.CFnOVE) is set to 1. The receive data is overwritten.

**Remark**   n = 0 to 4

### 18.6.14 Clock timing

(1/2)

**(i) Communication type 1 (CFnCKP and CFnDAP bits = 00)**



**(ii) Communication type 3 (CFnCKP and CFnDAP bits = 10)**



**Notes 1.** The INTCFnT interrupt is set when the data written to the CFnTX register is transferred to the data shift register in the continuous transmission or continuous transmission/reception mode. In the single transmission or single transmission/reception mode, the INTCFnT interrupt request signal is not generated, but the INTCFnR interrupt request signal is generated upon end of communication.

**2.** The INTCFnR interrupt occurs if reception is correctly ended and receive data is ready in the CFnRX register while reception is enabled. In the single mode, the INTCFnR interrupt request signal is generated even in the transmission mode, upon end of communication.

**Caution** **In single transfer mode, writing to the CFnTX register with the CFnTSF bit set to 1 is ignored. This has no effect on the operation during transfer.**
**For example, if the next data is written to the CFnTX register when DMA is started by generating the INTCFnR signal, the written data is not transferred because the CFnTSF bit is set to 1.**
**Use the continuous transfer mode, not the single transfer mode, for such applications.**

**Remark** n = 0 to 4

(2/2)

**(iii)  Communication type 2 (CFnCKP and CFnDAP bits = 01)**



**(iv)  Communication type 4 (CFnCKP and CFnDAP bits = 11)**
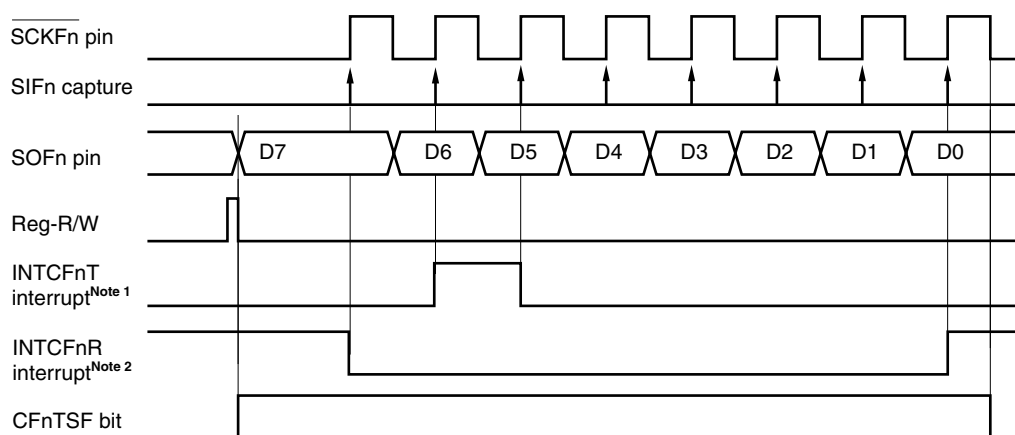


**Notes 1.** The INTCFnT interrupt is set when the data written to the CFnTX register is transferred to the data shift register in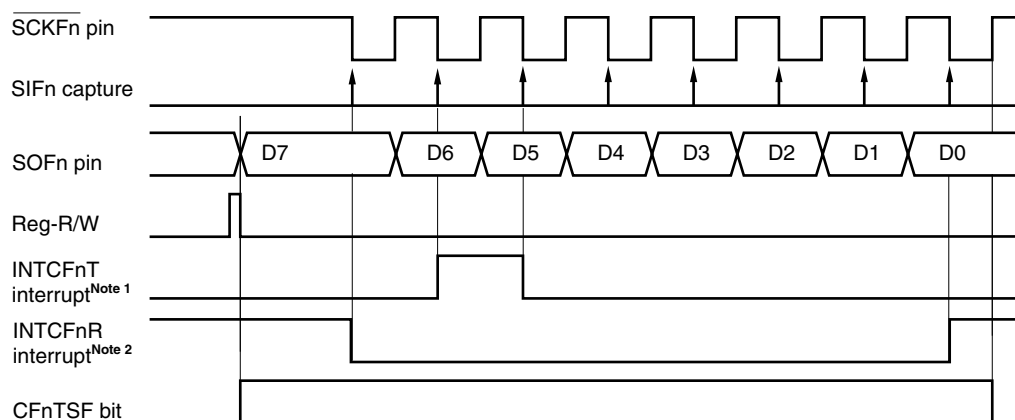 the continuous transmission or continuous transmission/reception modes.  In the single transmission or single transmission/reception modes, the INTCFnT interrupt request signal is not generated, but the INTCFnR interrupt request signal is generated upon end of communication.

    **2.** The INTCFnR interrupt occurs if reception is correctly ended and receive data is ready in the CFnRX register while reception is enabled.  In the single mode, the INTCFnR interrupt request signal is generated even in the transmission mode, upon end of communication.

**Caution** **In single transfer mode, writing to the CFnTX register with the CFnTSF bit set to 1 is ignored. This has no effect on the operation during transfer.**

                 **For example, if the next data is written to the CFnTX register when DMA is started by generating the INTCFnR signal, the written data is not transferred because the CFnTSF bit is set to 1.**

                 **Use the continuous transfer mode, not the single transfer mode, for such applications.**

**Remark** n = 0 to 4

## 18.7  Output Pins

**(1)  $\overline{\text{SCKFn}}$ pin**

When CSIFn operation is disabled (CFnCTL0.CFnPWR bit = 0), the $\overline{\text{SCKFn}}$ pin output status is as follows.

| CFnCKP | CFnCKS2 | CFnCKS1 | CFnCKS0 | $\overline{\text{SCKFn}}$ Pin Output |
|--------|---------|---------|---------|----------------|
| 0 | 1 | 1 | 1 | High impedance |
| | Other than above | | | Fixed to high level |
| 1 | 1 | 1 | 1 | High impedance |
| | Other than above | | | Fixed to low level |

**Remarks 1.** The output level of the $\overline{\text{SCKFn}}$ pin changes if any of the CFnCTL1.CFnCKP or CFnCKS2 to CFnCKS0 bits is rewritten.

**2.** n = 0 to 4

**(2)  SOFn pin**

When CSIFn operation is disabled (CFnPWR bit = 0), the SOFn pin output status is as follows.

| CFnTXE | CFnDAP | CFnDIR | SOFn Pin Output |
|--------|--------|--------|-----------------|
| 0 | × | × | Fixed to low level |
| 1 | 0 | × | SOFn latch value (low level) |
| | 1 | 0 | CFnTX value (MSB) |
| | | 1 | CFnTX value (LSB) |

**Remarks 1.** The SOFn pin output changes when any one of the CFnCTL0.CFnTXE, CFnCTL0.CFnDIR or CFnCTL1.CFnDAP bit is rewritten.

**2.** ×: Don't care

**3.** n = 0 to 4

## 18.8  Baud Rate Generator

The BRG1 to BRG3 baud rate generators are connected to CSIF0 to CSIF4 as shown in the following block diagram.
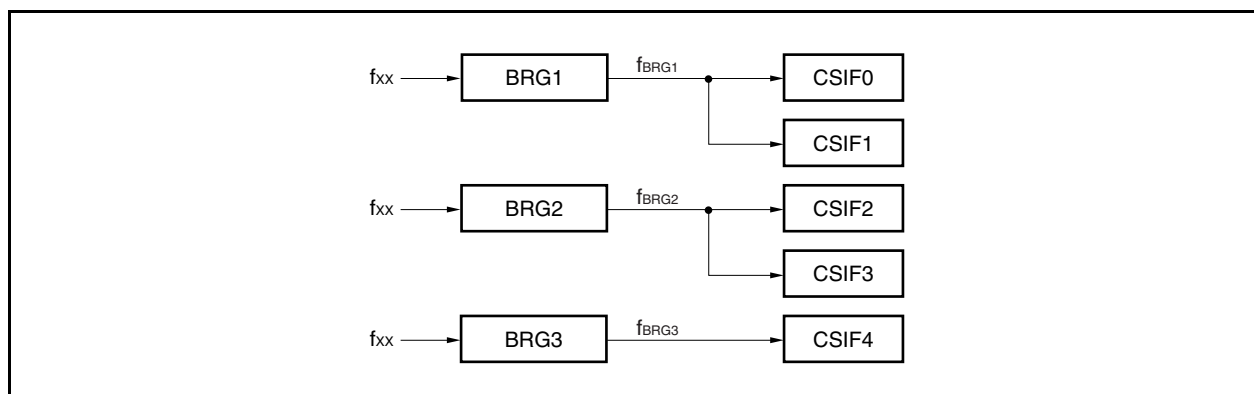


### (1)  Prescaler mode registers 1 to 3 (PRSM1 to PRSM3)

The PRSM1 to PRSM3 registers control generation of the baud rate signal for CSIF.

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

After reset: 00H    R/W    Address: PRSM1 FFFFF320H, PRSM2 FFFFF324H,
                                    PRSM3 FFFFF328H

| | 7 | 6 | 5 | <4> | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PRSMm (m = 1 to 3) | 0 | 0 | 0 | BGCEm | 0 | 0 | BGCSm1 | BGCSm0 |

| BGCEm | Baud rate output |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

| BGCSm1 | BGCSm0 | Input clock selection ($f_{BGCSm}$) | Setting value (k) |
|---|---|---|---|
| 0 | 0 | $f_{xx}$ | 0 |
| 0 | 1 | $f_{xx}/2$ | 1 |
| 1 | 0 | $f_{xx}/4$ | 2 |
| 1 | 1 | $f_{xx}/8$ | 3 |

**Cautions 1.  Do not rewrite the PRSMm register during operation.**

**2.  Set the PRSMm register before setting the BGCEm bit to 1.**

**2.  Be sure to set bits 7 to 5, 3, and 2 to "0".**

**(2)  Prescaler compare registers 1 to 3 (PRSCM1 to PRSCM3)**

The PRSCM1 to PRSCM3 registers are 8-bit compare registers.

These registers can be read or written in 8-bit units.

Reset sets these registers to 00H.

---

After reset: 00H      R/W      Address: PRSCM1 FFFFF321H, PRSCM2 FFFFF325H,
                                              PRSCM3 FFFFF329H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PRSCMm | PRSCMm7 | PRSCMm6 | PRSCMm5 | PRSCMm4 | PRSCMm3 | PRSCMm2 | PRSCMm1 | PRSCMm0 |

**Cautions  1.  Do not rewrite the PRSCMm register during operation.**

**2.  Set the PRSCMm register before setting the PRSMm.BGCEm bit to 1.**

---

**18.8.1  Baud rate generation**

The transmission/reception clock is generated by dividing the main clock.  The baud rate generated from the main clock is obtained by the following equation.

$$f_{BRGm} = \frac{f_{XX}}{2^{k+1} \times N}$$

**Caution   Set f$_{BRGm}$ to 8 MHz (CSIF0 to CSIF2 and CSIF4), 12 MHz (CSIF3), or lower.**

**Remark**   f$_{BRGm}$:   BRGm count clock

f$_{XX}$:     Main clock oscillation frequency

k:       PRSMm register setting value = 0 to 3

N:       PRSCMm register setting value = 1 to 256

However, N = 256 only when the PRSCMm register is set to 00H.

m = 1 to 3

## 18.9  Cautions

(1)    When transferring transmit data and receive data using DMA transfer, error processing cannot be performed even if an overrun error occurs during serial transfer.  Check that the no overrun error has occurred by reading the CFnSTR.CFnOVE bit after DMA transfer has been completed.

(2)    If a register that is prohibited to be rewritten during operation (CFnCTL0.CFnPWR bit = 1) is rewritten by mistake during operation, set the CFnCTL0.CFnPWR bit to 0 once, then initialize CSIFn.
Registers to which rewriting during operation is prohibited are shown below.

- CFnCTL0 register: CFnTXE, CFnRXE, CFnDIR, CFnTMS bits

- CFnCTL1 register: CFnCKP, CFnDAP, CFnCKS2 to CFnCKS0 bits

- CFnCTL2 register: CFnCL3 to CFnCL0 bits

(3)   In communication type 2 or 4 (CFnCTL1.CFnDAP bit = 1), the CFnSTR.CFnTSF bit is cleared half a $\overline{\text{SCKFn}}$ clock after the occurrence of a reception completion interrupt (INTCFnR).
In the single transfer mode, writing the next transmit data is ignored during communication (CFnTSF bit = 1), and the next communication is not started.  Also if reception-only communication (CFnCTL0.CFnTXE bit = 0, CFnCTL0.CFnRXE bit = 1) is set, the next communication is not started even if the receive data is read during communication (CFnTSF bit = 1).
Therefore, when using the single transfer mode with communication type 2 or 4 (CFnDAP bit = 1), pay particular attention to the following.

- To start the next transmission, confirm that CFnTSF bit = 0 and then write the transmit data to the CFnTX register.
- To perform the next reception continuously when reception-only communication (CFnTXE bit = 0, CFnRXE bit = 1) is set, confirm that CFnTSF bit = 0 and then read the CFnRX register.

Or, use the continuous transfer mode instead of the single transfer mode.  Use of the continuous transfer mode is recommended especially when using DMA.

**Remark**   n = 0 to 4

# CHAPTER 19  I²C  BUS

To use the I²C bus function, set the P36/SCL00, P37/SDA00, P40/SDA01, P41/SCL01, P90/SDA02, and P91/SCL02 pins as alternate-function pins, and set them to N-ch open-drain output.

## 19.1  Mode Switching of I²C Bus and Other Serial Interfaces

### 19.1.1  UARTC3 and I²C00 mode switching

In the V850ES/JG3-U and V850ES/JH3-U, UARTC3 and I²C00 share the same pins and therefore cannot be used simultaneously.  Switching between UARTC3 and I²C00 must be set in advance, using the PMC3, PFC3, and PFCE3 registers.

**Caution   The transmit/receive operation of UARTC3 and I²C00 is not guaranteed if these functions are switched during transmission or reception.  Be sure to disable the one that is not used.**

**Figure 19-1.  UARTC3 and I²C00 Mode Switch Settings**

After reset:  00H      R/W      Address:  FFFFF446H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC3 | PMC37 | PMC36 | PMC35 | PMC34 | PMC33 | PMC32 | PMC31 | PMC30 |

After reset:  00H      R/W      Address:  FFFFF466H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC3 | PFC37 | PFC36 | PFC35 | PFC34 | PFC33 | PFC32 | PFC31 | PFC30 |

After reset:  00H      R/W      Address:  FFFFF706H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE3 | PFCE37 | PFCE36 | PFCE35 | PFCE34 | PFCE33 | PFCE32 | PFCE31 | PFCE30 |

| PMC3n | PFCE3n | PFC3n | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | UARTC3 mode |
| 1 | 0 | 1 | I²C00 mode |

**Remarks 1.**  n = 6, 7
**2.**  × = don't care

**19.1.2  UARTC4, CSIF0, and I$^2$C01 mode switching**

In the V850ES/JG3-U and V850ES/JH3-U, UARTC4, CSIF0, and I$^2$C01 share the same pins and therefore cannot be used simultaneously.  Switching among UARTC4, CSIF0, and I$^2$C01 must be set in advance, using the PMC4, PFC4, and PFCE4 registers.

**Caution   The transmit/receive operation of UARTC4, CSIF0, and I$^2$C01 is not guaranteed if these functions are switched during transmission or reception.  Be sure to disable the one that is not used.**

**Figure 19-2.  UARTC4, CSIF0, and I$^2$C01 Mode Switch Settings**

After reset:  00H     R/W     Address:  FFFFF448H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC4 | 0 | 0 | 0 | 0 | 0 | PMC42 | PMC41 | PMC40 |

After reset:  00H     R/W     Address:  FFFFF468H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC4 | 0 | 0 | 0 | 0 | 0 | PFC42 | PFC41 | PFC40 |

After reset:  00H     R/W     Address:  FFFFF708H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE4 | 0 | 0 | 0 | 0 | 0 | 0 | PFCE41 | PFCE40 |

| PMC4n | PFC4n | PFCE4n | Operation mode |
|---|---|---|---|
| 0 | × | × | Port I/O mode |
| 1 | 0 | 0 | CSIF0 mode |
| 1 | 0 | 1 | I$^2$C01 mode |
| 1 | 1 | 0 | UARTC4 mode |

**Remarks 1.**  n = 0, 1
**2.**  × = don't care

### 19.1.3  UARTC1 and I²C02 mode switching

In the V850ES/JG3-U and V850ES/JH3-U, UARTC1 and I²C02 share the same pins and therefore cannot be used simultaneously.  Switching between UARTC1 and I²C02 must be set in advance, using the PMC9, PFC9, and PFCE9 registers.

**Caution   The transmit/receive operation of UARTC1 and I²C02 is not guaranteed if these functions are switched during transmission or reception.  Be sure to disable the one that is not used.**

**Figure 19-3.  UARTC1 and I²C02 Mode Switch Settings**

After reset:  0000H       R/W       Address:  FFFFF452H, FFFFF453H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PMC9 | PMC915 | PMC914 | PMC913 | PMC912 | PMC911 | PMC910 | PMC99 | PMC98 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PMC97 | PMC96 | PMC95 | PMC94 | PMC93 | PMC92 | PMC91 | PMC90 |

After reset:  0000H       R/W       Address:  FFFFF472H, FFFFF473H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFC9 | PFC915 | PFC914 | PFC913 | PFC912 | PFC911 | PFC910 | PFC99 | PFC98 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PFC97 | PFC96 | PFC95 | PFC94 | PFC93 | PFC92 | PFC91 | PFC90 |

After reset:  0000H       R/W       Address:  FFFFF712H, FFFFF713H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFCE9 | PFCE915 | PFCE914 | 0 | 0 | PFCE911 | PFCE910 | PFCE99 | PFCE98 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PFCE97 | PFCE96 | PFCE95 | PFCE94 | PFCE93 | PFCE92 | PFCE91 | PFCE90 |

| PMC9n | PFCE9n | PFC9n | Operation mode |
|---|---|---|---|
| 1 | 0 | 1 | UARTC1 mode |
| 1 | 1 | 0 | I²C02 mode |

**Remark**   n = 0, 1

<R>

## 19.2  Features

I²C00 to I²C02 have the following two modes.

- Operation stop mode
- I²C (Inter IC) bus mode (multimasters supported)

**(1)  Operation stop mode**

In this mode, serial transfer is not performed, thus enabling a reduction in power consumption.

**(2)  I²C bus mode (multimaster support)**

This mode is used for 8-bit data transfers with several devices via two lines: a serial clock pin (SCL0n) and a serial data bus pin (SDA0n).

This mode complies with the I²C bus format and the master device can generate "start condition", "address", "transfer direction specification", "data", and "stop condition" data for the slave device on the serial data bus.  The slave device automatically detects the received statuses and data by hardware.  This function can simplify the part of an application program that controls the I²C bus.

Since SCL0n and SDA0n pins are used for N-ch open-drain outputs, I²C0n requires pull-up resistors for the serial clock line and the serial data bus line.

**Remark**   n = 0 to 2

## 19.3  Configuration

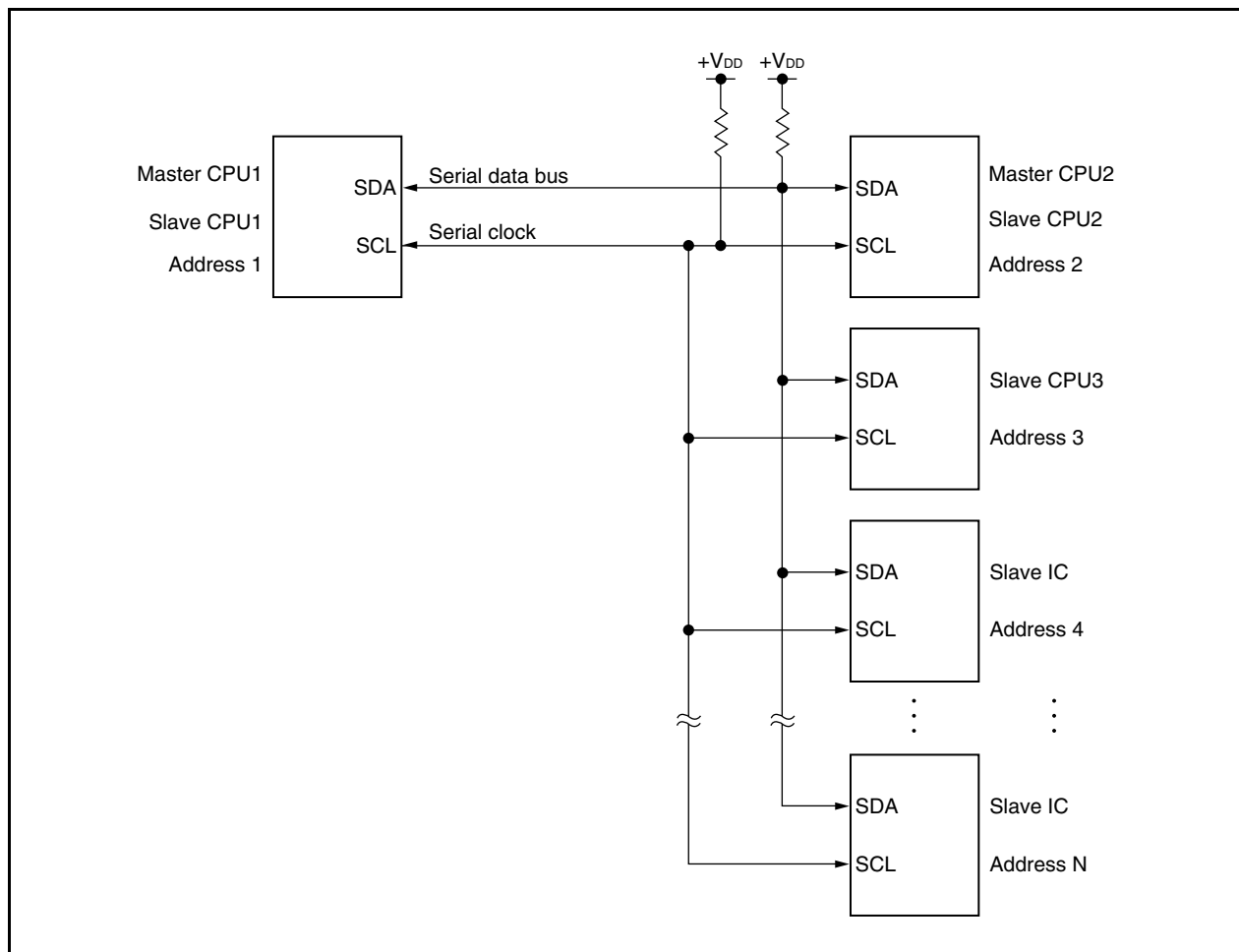The block diagram of the I²C0n is shown below.

**Figure 19-4.  Block Diagram of I²C0n**



**Remark**   n = 0 to 2
          m = 0, 1
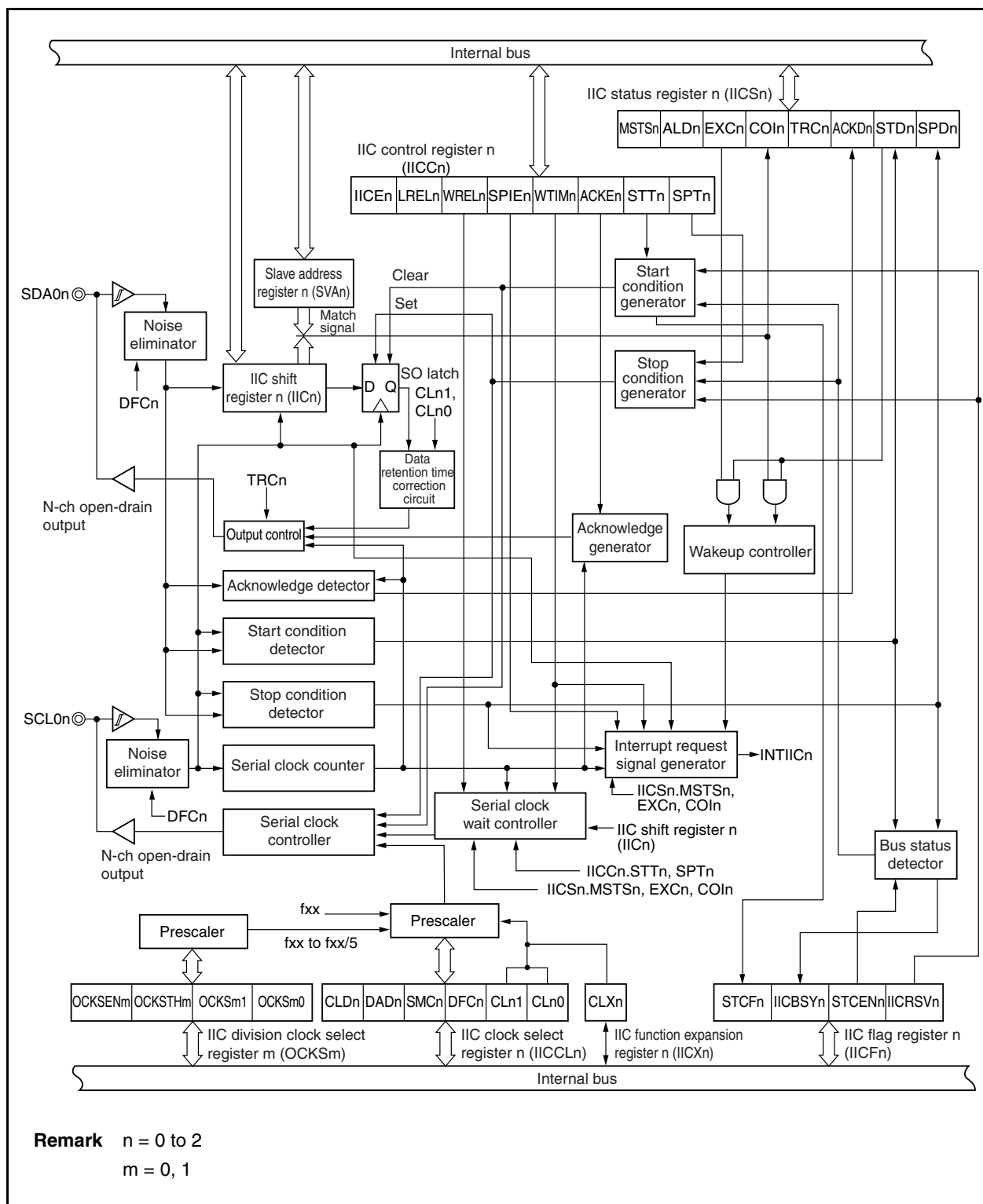
A serial bus configuration example is shown below.

**Figure 19-5. Serial Bus Configuration Example Using I²C Bus**

I²C0n includes the following hardware (n = 0 to 2).

**Table 19-1. Configuration of I²C0n**

| Item | Configuration |
|---|---|
| Registers | IIC shift register n (IICn) |
| | Slave address register n (SVAn) |
| Control registers | IIC control register n (IICCn) |
| | IIC status register n (IICSn) |
| | IIC flag register n (IICF0n) |
| | IIC clock select register n (IICCLn) |
| | IIC function expansion register n (IICXn) |
| | IIC division clock select registers 0, 1 (OCKS0, OCKS1) |

**(1) IIC shift register n (IICn)**

The IICn register converts 8-bit serial data into 8-bit parallel data and vice versa, and can be used for both transmission and reception (n = 0 to 2).

Write and read operations to the IICn register are used to control the actual transmit and receive operations.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**(2) Slave address register n (SVAn)**

The SVAn register sets local addresses when in slave mode (n = 0 to 2).

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**(3) SO latch**

The SO latch is used to retain the output level of the SDA0n pin (n = 0 to 2).

**(4) Wakeup controller**

This circuit generates an interrupt request signal (INTIICn) when the address value set to the SVAn register matches the received address or when an extension code is received (n = 0 to 2).

**(5) Prescaler**

This selects the sampling clock to be used.

**(6) Serial clock counter**

This counter counts the serial clocks that are output or input during transmit/receive operations and is used to verify that 8-bit data was transmitted or received.

**(7) Interrupt request signal generator**

This circuit controls the generation of interrupt request signals (INTIICn).

An I²C interrupt is generated by either of the following two triggers.

- The falling edge of the eighth or ninth clock of the serial clock (set by IICCn.WTIMn bit)
- Interrupt occurrence due to stop condition detection (set by IICCn.SPIEn bit)

**Remark**   n = 0 to 2

**(8) Serial clock controller**

In master mode, this circuit generates the clock output via the SCL0n pin from the sampling clock (n = 0 to 2).

**(9) Serial clock wait controller**

This circuit controls the wait timing.

**(10) $\overline{ACK}$ generator, stop condition detector, start condition detector, and $\overline{ACK}$ detector**

These circuits are used to generate and detect various statuses.

**(11) Data hold time correction circuit**

This circuit generates the hold time for the data corresponding to the falling edge of the SCL0n pin.

**(12) Start condition generator**

This circuit generates a start condition when the IICCn.STTn bit is set.

However, when in the communication reservation disabled status (IICFn.IICRSVn bit = 1) and when the bus is not released (IICFn.IICBSYn bit = 1), this request is ignored and the IICFn.STCFn bit is set to 1.

**(13) Stop condition generator**

This circuit generates a stop condition when the IICCn.SPTn bit is set.

**(14) Bus status detector**

This circuit detects whether the bus is released by detecting a start condition and stop condition.

However, the bus status cannot be detected immediately after operation, so set the bus status detector to the initial status by using the IICFn.STCENn bit.

### 19.4　Registers

I²C00 to I²C02 are controlled by the following registers.

- IIC control registers 0 to 2 (IICC0 to IICC2)
- IIC status registers 0 to 2 (IICS0 to IICS2)
- IIC flag registers 0 to 2 (IICF0 to IICF2)
- IIC clock select registers 0 to 2 (IICCL0 to IICCL2)
- IIC function expansion registers 0 to 2 (IICX0 to IICX2)
- IIC division clock select registers 0, 1 (OCKS0, OCKS1)

The following registers are also used.

- IIC shift registers 0 to 2 (IIC0 to IIC2)
- Slave address registers 0 to 2 (SVA0 to SVA2)

**Remark**　For the alternate-function pin settings, see **Table 4-20　Settings When Port Pins Are Used for Alternate Functions**.

**(1)　IIC control registers 0 to 2 (IICC0 to IICC2)**

The IICCn registers enable/stop I²C0n operations, set the wait timing, and set other I²C operations (n = 0 to 2). These registers can be read or written in 8-bit or 1-bit units. However, set the SPIEn, WTIMn, and ACKEn bits when the IICEn bit is 0 or during the wait period. When changing the IICEn bit from "0" to "1", these bits can also be set at the same time.

Reset sets these registers to 00H.

(1/4)

After reset: 00H          R/W          Address: IICC0 FFFFFD82H, IICC1 FFFFFD92H, IICC2 FFFFFDA2H

| | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IICCn | IICEn | LRELn | WRELn | SPIEn | WTIMn | ACKEn | STTn | SPTn |

(n = 0 to 2)

| IICEn | Specification of I²Cn operation enable/disable |
|---|---|
| 0 | Operation stopped.  IICSn register reset[Note 1].  Internal operation stopped. |
| 1 | Operation enabled. |

Be sure to set this bit to 1 when the SCL0n and SDA0n lines are high level.

| Condition for clearing (IICEn bit = 0) | Condition for setting (IICEn bit = 1) |
|---|---|
| • Cleared by instruction<br>• After reset | • Set by instruction |

| LRELn[Note 2] | Exit from communications |
|---|---|
| 0 | Normal operation |
| 1 | This exits from the current communication operation and sets standby mode.  This setting is automatically cleared after being executed.  Its uses include cases in which a locally irrelevant extension code has been received.<br>The SCL0n and SDA0n lines are set to high impedance.<br>The STTn and SPTn bits and the MSTSn, EXCn, COIn, TRCn, ACKDn, and STDn bits of the IICSn register are cleared. |

The standby mode following exit from communications remains in effect until the following communication entry conditions are met.
• After a stop condition is detected, restart is in master mode.
• An address match occurs or an extension code is received after the start condition.

| Condition for clearing (LRELn bit = 0) | Condition for setting (LRELn bit = 1) |
|---|---|
| • Automatically cleared after execution<br>• After reset | • Set by instruction |

| WRELn[Note 2] | Wait state cancellation control |
|---|---|
| 0 | Wait state not canceled |
| 1 | Wait state canceled.  This setting is automatically cleared after wait state is canceled. |

| Condition for clearing (WRELn bit = 0) | Condition for setting (WRELn bit = 1) |
|---|---|
| • Automatically cleared after execution<br>• After reset | • Set by instruction |

**Notes  1.**  The IICSn register, IICFn.STCFn and IICFn.IICBSYn bits, and IICCLn.CLDn and IICCLn.DADn bits are reset.
      **2.**  This flag's signal is invalid when the IICEn bit = 0.

**Caution**   **If the I²Cn operation is enabled (IICEn bit = 1) when the SCL0n line is high level and the SDA0n line is low level, the start condition is detected immediately.  To avoid this, after enabling the I²Cn operation, immediately set the LRELn bit to 1 with a bit manipulation instruction.**

**Remark**   The LRELn and WRELn bits are 0 when read after the data has been set.

| SPIEn[Note] | Enabling/disabling generation of interrupt request when stop condition is detected |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

| Condition for clearing (SPIEn bit = 0) | Condition for setting (SPIEn bit = 1) |
|---|---|
| • Cleared by instruction<br>• After reset | • Set by instruction |

| WTIMn[Note] | Control of wait state and interrupt request generation |
|---|---|
| 0 | Interrupt request is generated at the eighth clock's falling edge.<br>Master mode:  After output of eight clocks, clock output is set to low level and the wait state is set.<br>Slave mode:   After input of eight clocks, the clock is set to low level and the wait state is set for the master device. |
| 1 | Interrupt request is generated at the ninth clock's falling edge.<br>Master mode:  After output of nine clocks, clock output is set to low level and the wait state is set.<br>Slave mode:   After input of nine clocks, the clock is set to low level and the wait state is set for the master device. |

During address transfer, an interrupt occurs at the falling edge of the ninth clock regardless of this bit setting.  This bit setting becomes valid when the address transfer is completed.  In master mode, a wait state is inserted at the falling edge of the ninth clock during address transfer.  For a slave device that has received a local address, a wait state is inserted at the falling edge of the ninth clock after $\overline{ACK}$ is generated.  The slave device that has received an extension code, however, enters a wait state at the falling edge of the eighth clock.

| Condition for clearing (WTIMn bit = 0) | Condition for setting (WTIMn bit = 1) |
|---|---|
| • Cleared by instruction<br>• After reset | • Set by instruction |

| ACKEn[Note] | Acknowledgment control |
|---|---|
| 0 | Acknowledgment disabled. |
| 1 | Acknowledgment enabled.  During the ninth clock period, the SDA0n line is set to low level. |

The ACKEn bit setting is invalid for address reception by the slave device.  In this case, $\overline{ACK}$ is generated when the addresses match.
However, the ACKEn bit setting is valid for reception of the extension code.  Set the ACKEn bit in the system that receives the extension code.

| Condition for clearing (ACKEn bit = 0) | Condition for setting (ACKEn bit = 1) |
|---|---|
| • Cleared by instruction<br>• After reset | • Set by instruction |

**Note**    This flag's signal is invalid when the IICEn bit = 0.

**Remark**    n = 0 to 2

(3/4)

| STTn | Start condition trigger |
|------|------------------------|
| 0 | Start condition is not generated. |
| 1 | When bus is released (in STOP mode):<br>    A start condition is generated (for starting as master).  The SDA0n line is changed from high level to low level while the SCLn line is high level and then the start condition is generated.  Next, after the rated amount of time has elapsed, the SCL0n line is changed to low level.<br>During communication with a third party:<br>    If the communication reservation function is enabled (IICFn.IICRSVn bit = 0)<br>    • This trigger functions as a start condition reserve flag.  When set to 1, it releases the bus and then automatically generates a start condition.<br>    If the communication reservation function is disabled (IICRSVn = 1)<br>    • The IICFn.STCFn bit is set to 1 and information set (1) to the STTn bit is cleared.  This trigger does not generate a start condition.<br>In the wait state (when master device):<br>    A restart condition is generated after the wait state is released. |

Cautions concerning set timing

| | |
|---|---|
| For master reception: | Cannot be set to 1 during transfer.  Can be set to 1 only when the ACKEn bit has been set to 0 and the slave has been notified of final reception. |
| For master transmission: | A start condition may not be generated normally during the $\overline{\text{ACK}}$ period.  Set to 1 during the wait period that follows output of the ninth clock. |
| For slave: | Even when the communication reservation function is disabled (IICRSVn bit = 1), the communication reservation status is entered. |

• Setting to 1 at the same time as the SPTn bit is prohibited.
• When the STTn bit is set to 1, setting the STTn bit to 1 again is disabled until the setting is cleared to 0.

| Condition for clearing (STTn bit = 0) | Condition for setting (STTn bit = 1) |
|---------------------------------------|--------------------------------------|
| • When the STTn bit is set to 1 in the communication reservation disabled status<br>• Cleared by loss in arbitration<br>• Cleared after start condition is generated by master device<br>• When the LRELn bit = 1 (communication save)<br>• When the IICEn bit = 0 (operation stop)<br>• After reset | • Set by instruction |

**Remarks 1.** The STTn bit is 0 if it is read immediately after data setting.
           **2.** n = 0 to 2

| SPTn | Stop condition trigger |
|------|------------------------|
| 0 | Stop condition is not generated. |
| 1 | Stop condition is generated (termination of master device's transfer). After the SDA0n line goes to low level, either set the SCL0n line to high level or wait until the SCL0n pin goes to high level.  Next, after the rated amount of time has elapsed, the SDA0n line is changed from low level to high level and a stop condition is generated. |

Cautions concerning set timing

For master reception:     Cannot be set to 1 during transfer.

   Can be set to 1 only when the ACKEn bit has been set to 0 and during the wait period after the slave has been notified of final reception.

For master transmission:  A stop condition may not be generated normally during the $\overline{\text{ACK}}$ reception period.  Set to 1 during the wait period that follows output of the ninth clock.

- Cannot be set to 1 at the same time as the STTn bit.
- The SPTn bit can be set to 1 only when in master mode[Note].
- When the WTIMn bit has been set to 0, if the SPTn bit is set to 1 during the wait period that follows output of eight clocks, note that a stop condition will be generated during the high-level period of the ninth clock.
  The WTIMn bit should be changed from 0 to 1 during the wait period following output of eight clocks, and the SPTn bit should be set to 1 during the wait period that follows output of the ninth clock.
- When the SPTn bit is set to 1, setting the SPTn bit to 1 again is disabled until the setting is cleared to 0.

| Condition for clearing (SPTn bit = 0) | Condition for setting (SPTn bit = 1) |
|---------------------------------------|--------------------------------------|
| • Cleared by loss in arbitration<br>• Automatically cleared after stop condition is detected<br>• Cleared when the LRELn bit = 1 (communication save)<br>• When the IICEn bit = 0 (operation stop)<br>• After reset | • Set by instruction |

**Note**   Set the SPTn bit to 1 only in master mode.  However, to perform a master operation before detecting the first stop condition after operation has been enabled when the IICRSVn bit is 0, the SPTn bit must be set to 1 and a stop condition must be set.  For details, see **19.15  Cautions**.

**Caution**   **If the WRELn bit is set to 1 during the ninth clock and the wait state is canceled when the TRCn bit is 1, the TRCn bit is cleared to 0 and the SDA0n line is set to high impedance.**

**Remarks  1.**  The SPTn bit is 0 if it is read immediately after data setting.

**2.**  n = 0 to 2

**(2) IIC status registers 0 to 2 (IICS0 to IICS2)**

The IICSn registers indicate the status of I²C0n (n = 0 to 2).

These registers are read-only, in 8-bit or 1-bit units.  However, the IICSn registers can only be read when the IICCn.STTn bit is 1 or during the wait period.

Reset sets these registers to 00H.

**Caution   Accessing the IICSn registers is prohibited in the following statuses.  For details, see 3.4.7 (2) Accessing specific on-chip peripheral I/O registers.**
- **When the CPU operates on the subclock and the main clock oscillation is stopped**
- **When the CPU operates on the internal oscillation clock**

(1/3)

After reset: 00H          R            Address: IICS0 FFFFFD86H, IICS1 FFFFFD96H, IICS2 FFFFFDA6H

|        | <7>    | <6>  | <5>  | <4>  | <3>  | <2>   | <1>  | <0>  |
|--------|--------|------|------|------|------|-------|------|------|
| IICSn  | MSTSn  | ALDn | EXCn | COIn | TRCn | ACKDn | STDn | SPDn |

(n = 0 to 2)

| MSTSn | Master device status |
|-------|----------------------|
| 0 | Slave device status or communication standby status |
| 1 | Master device communication status |

| Condition for clearing (MSTSn bit = 0) | Condition for setting (MSTSn bit = 1) |
|---|---|
| • When a stop condition is detected<br>• When the ALDn bit = 1 (arbitration loss)<br>• Cleared by LRELn bit = 1 (communication save)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• After reset | • When a start condition is generated |

| ALDn | Arbitration loss detection |
|------|----------------------------|
| 0 | This status means either that there was no arbitration or that the arbitration result was a "win". |
| 1 | This status indicates the arbitration result was a "loss".  The MSTSn bit is cleared to 0. |

| Condition for clearing (ALDn bit = 0) | Condition for setting (ALDn bit = 1) |
|---|---|
| • Automatically cleared after the IICSn register is read**Note**<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• After reset | • When the arbitration result is a "loss". |

| EXCn | Detection of extension code reception |
|------|---------------------------------------|
| 0 | Extension code was not received. |
| 1 | Extension code was received. |

| Condition for clearing (EXCn bit = 0) | Condition for setting (EXCn bit = 1) |
|---|---|
| • When a start condition is detected<br>• When a stop condition is detected<br>• Cleared by LRELn bit = 1 (communication save)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• After reset | • When the higher four bits of the received address data are either "0000" or "1111" (set at the rising edge of the eighth clock). |

**Note**   This bit is also cleared when a bit manipulation instruction is executed for another bit in the IICSn register.

| COIn | Matching address detection |
| --- | --- |
| 0 | Addresses do not match. |
| 1 | Addresses match. |

| Condition for clearing (COIn bit = 0) | Condition for setting (COIn bit = 1) |
| --- | --- |
| • When a start condition is detected<br>• When a stop condition is detected<br>• Cleared by LRELn bit = 1 (communication save)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• After reset | • When the received address matches the local address (SVAn register) (set at the rising edge of the eighth clock). |

| TRCn | Transmit/receive status detection |
| --- | --- |
| 0 | Receive status (other than transmit status).  The SDA0n line is set to high impedance. |
| 1 | Transmit status.  The value in the SO latch is enabled for output to the SDA0n line (valid starting at the falling edge of the first byte's ninth clock). |

| Condition for clearing (TRCn bit = 0) | Condition for setting (TRCn bit = 1) |
| --- | --- |
| • When a stop condition is detected<br>• Cleared by LRELn bit = 1 (communication save)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• Cleared by IICCn.WRELn bit = 1[Note]<br>• When the ALDn bit changes from 0 to 1 (arbitration loss)<br>• After reset<br>Master<br>• When "1" is output to the first byte's LSB (transfer direction specification bit)<br>Slave<br>• When a start condition is detected<br>When not used for communication | Master<br>• When a start condition is generated<br>• When "0" is output to the first byte's LSB (transfer direction specification bit)<br>Slave<br>• When "1" is input by the first byte's LSB (transfer direction specification bit) |

| ACKDn | ACK detection |
| --- | --- |
| 0 | ACK was not detected. |
| 1 | ACK was detected. |

| Condition for clearing (ACKDn bit = 0) | Condition for setting (ACKD bit = 1) |
| --- | --- |
| • When a stop condition is detected<br>• At the rising edge of the next byte's first clock<br>• Cleared by LRELn bit = 1 (communication save)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• After reset | • After the SDA0n bit is set to low level at the rising edge of the SCL0n pin's ninth clock |

**Note**  The TRCn bit is cleared to 0 and SDA0n line becomes high impedance when the WRELn bit is set to 1 and the wait state is canceled to 0 at the ninth clock by TRCn bit = 1.

**Remark**   n = 0 to 2

| STDn | Start condition detection |
|------|---------------------------|
| 0 | Start condition was not detected. |
| 1 | Start condition was detected.  This indicates that the address transfer period is in effect |

| Condition for clearing (STDn bit = 0) | Condition for setting (STDn bit = 1) |
|---------------------------------------|--------------------------------------|
| • When a stop condition is detected<br>• At the rising edge of the next byte's first clock following address transfer<br>• Cleared by LRELn bit = 1 (communication save)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• After reset | • When a start condition is detected |

| SPDn | Stop condition detection |
|------|--------------------------|
| 0 | Stop condition was not detected. |
| 1 | Stop condition was detected.  The master device's communication is terminated and the bus is released. |

| Condition for clearing (SPDn bit = 0) | Condition for setting (SPDn bit = 1) |
|---------------------------------------|--------------------------------------|
| • At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• After reset | • When a stop condition is detected |

**Remark**   n = 0 to 2

**(3) IIC flag registers 0 to 2 (IICF0 to IICF2)**

The IICFn registers set the I²C0n operation mode and indicate the I²C bus status.

These registers can be read or written in 8-bit or 1-bit units. However, the STCFn and IICBSYn bits are read-only.

IICRSVn enables/disables the communication reservation function (see **19.14 Communication Reservation**).

The initial value of the IICBSYn bit is set by using the STCENn bit (see **19.15 Cautions**).

The IICRSVn and STCENn bits can be written only when operation of I²C0n is disabled (IICCn.IICEn bit = 0). After operation is enabled, IICFn can be read (n = 0 to 2).

Reset sets these registers to 00H.

After reset: 00H          R/W^Note          Address: IICF0 FFFFFD8AH, IICF1 FFFFFD9AH, IICF2 FFFFFDAAH

| | <7> | <6> | 5 | 4 | 3 | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IICFn | STCFn | IICBSYn | 0 | 0 | 0 | 0 | STCENn | IICRSVn |

(n = 0 to 2)

| STCFn | STTn bit clear |
|---|---|
| 0 | Start condition issued |
| 1 | Start condition cannot be issued, STTn bit cleared |

| Condition for clearing (STCFn bit = 0) | Condition for setting (STCFn bit = 1) |
|---|---|
| • Cleared by IICCn.STTn bit = 1<br>• When the IICCn.IICEn bit = 0<br>• After reset | • When start condition is not issued and STTn flag is cleared to 0 when communication reservation is disabled (IICRSVn bit = 1). |

| IICBSYn | I²C0n bus status |
|---|---|
| 0 | Bus release status (default communication status when STCENn bit = 1) |
| 1 | Bus communication status (default communication status when STCENn bit = 0) |

| Condition for clearing (IICBSYn bit = 0) | Condition for setting (IICBSYn bit = 1) |
|---|---|
| • When stop condition is detected<br>• When the IICEn bit = 0<br>• After reset | • When start condition is detected<br>• By setting the IICEn bit when the STCENn bit = 0 |

| STCENn | Initial start enable trigger |
|---|---|
| 0 | Start conditions cannot be generated until a stop condition is detected following operation enable (IICEn bit = 1). |
| 1 | Start conditions can be generated even if a stop condition is not detected following operation enable (IICEn bit = 1). |

| Condition for clearing (STCENn bit = 0) | Condition for setting (STCENn bit = 1) |
|---|---|
| • When start condition is detected<br>• After reset | • Setting by instruction |

| IICRSVn | Communication reservation function disable bit |
|---|---|
| 0 | Communication reservation enabled |
| 1 | Communication reservation disabled |

| Condition for clearing (IICRSVn bit = 0) | Condition for setting (IICRSVn bit = 1) |
|---|---|
| • Clearing by instruction<br>• After reset | • Setting by instruction |

**Note**  Bits 6 and 7 are read-only bits.

**Cautions  1.  Write the STCENn bit only when operation is stopped (IICEn bit = 0).**
           **2.  When the STCENn bit = 1, the bus released status (IICBSYn bit = 0) is recognized regardless of the actual bus status immediately after the I²Cn bus operation is enabled. Therefore, to issue the first start condition (STTn bit = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.**
           **3.  Write the IICRSVn bit only when operation is stopped (IICEn bit = 0).**

**(4) IIC clock select registers 0 to 2 (IICCL0 to IICCL2)**

The IICCLn registers set the transfer clock for I²C0n.

These registers can be read or written in 8-bit or 1-bit units.  However, the CLDn and DADn bits are read-only.

Set the IICCLn registers when the IICCn.IICEn bit = 0.

The SMCn, CLn1, and CLn0 bits are set by combining the IICXn.CLXn bit and the OCKSTHm, OCKSm1, and OCKSm0 bits of the OCKSm register (see **19.4 (6)  I²C0n transfer clock setting method**) (n = 0 to 2, m = 0, 1).

Reset sets these registers to 00H.

---

After reset: 00H　　　R/W^Note　　　Address: IICCL0 FFFFFD84H, IICCL1 FFFFFD94H, IICCL2 FFFFFDA4H

|  | 7 | 6 | <5> | <4> | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IICCLn | 0 | 0 | CLDn | DADn | SMCn | DFCn | CLn1 | CLn0 |

(n = 0 to 2)

| CLDn | Detection of SCL0n pin level (valid only when IICCn.IICEn bit = 1) |
|---|---|
| 0 | The SCL0n pin was detected at low level. |
| 1 | The SCL0n pin was detected at high level. |

| Condition for clearing (CLDn bit = 0) | Condition for setting (CLDn bit = 1) |
|---|---|
| • When the SCL0n pin is at low level<br>• When the IICEn bit = 0 (operation stop)<br>• After reset | • When the SCL0n pin is at high level |

| DADn | Detection of SDA0n pin level (valid only when IICEn bit = 1) |
|---|---|
| 0 | The SDA0n pin was detected at low level. |
| 1 | The SDA0n pin was detected at high level. |

| Condition for clearing (DADn bit = 0) | Condition for setting (DAD0n bit = 1) |
|---|---|
| • When the SDA0n pin is at low level<br>• When the IICEn bit = 0 (operation stop)<br>• After reset | • When the SDA0n pin is at high level |

| SMCn | Operation mode switching |
|---|---|
| 0 | Operation in standard mode. |
| 1 | Operation in high-speed mode. |

| DFCn | Digital filter operation control |
|---|---|
| 0 | Digital filter off. |
| 1 | Digital filter on. |

The digital filter can be used only in high-speed mode.
In high-speed mode, the transfer clock does not vary according to the DFCn bit setting (on/off).
The digital filter is used to eliminate noise in high-speed mode.

**Note**　Bits 4 and 5 are read-only bits.

**Caution**　**Be sure to clear bits 7 and 6 to "0".**

**Remark**　When the IICCn.IICEn bit = 0, 0 is read when reading the CLDn and DADn bits.

---

**(5)  IIC function expansion registers 0 to 2 (IICX0 to IICX2)**

The IICXn registers set I$^2$C0n function expansion (valid only in the high-speed mode).

These registers can be read or written in 8-bit or 1-bit units.

Setting of the CLXn bit is performed in combination with the SMCn, CLn1, and CLn0 bits of the IICCLn register and the OCKSTHm, OCKSm1, and OCKSm0 bits of the OCKSm register (see **19.4 (6)  I$^2$C0n transfer clock setting method**) (m = 0, 1).

Set the IICXn registers when the IICCn.IICEn bit = 0.

Reset sets these registers to 00H.

---

After reset:  00H      R/W      Address:  IICX0 FFFFFD85H, IICX1 FFFFFD95H, IICX2 FFFFFDA5H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| IICXn | 0 | 0 | 0 | 0 | 0 | 0 | 0 | CLXn |

(n = 0 to 2)

---

**(6)  I$^2$C0n transfer clock setting method**

The I$^2$C0n transfer clock frequency (f$_{SCL}$) is calculated using the following expression (n = 0 to 2).

$$f_{SCL} = 1/(m \times T + t_R + t_F)$$

m = 24, 48, 72, 96, 108, 120, 144, 172, 192, 240, 264, 344, 352, 396, 440, 516, 688, 860 (see **Table 19-2 Clock Settings**).

T:   1/f$_{XX}$

t$_R$:   SCL0n pin rise time

t$_F$:   SCL0n pin fall time

For example, the I$^2$C0n transfer clock frequency (f$_{SCL}$) when f$_{XX}$ = 19.2 MHz, m = 198, t$_R$ = 200 ns, and t$_F$ = 50 ns is calculated using following expression.

$$f_{SCL} = 1/(198 \times 52 \text{ ns} + 200 \text{ ns} + 50 \text{ ns}) \cong 94.7 \text{ kHz}$$
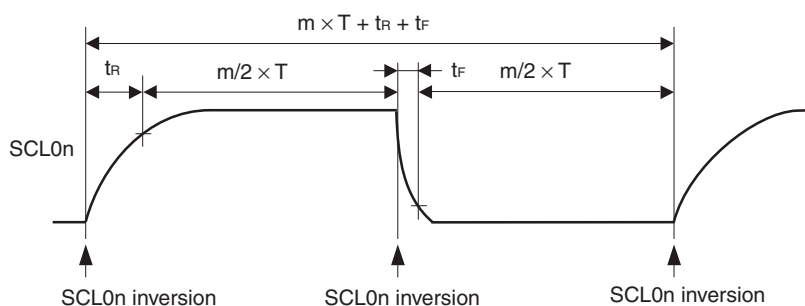


The clock to be selected can be set by combining of the SMCn, CLn1, and CLn0 bits of the IICCLn register, the CLXn bit of the IICXn register, and the OCKSTHm, OCKSm1, and OCKSm0 bits of the OCKSm register (n = 0 to 2, m = 0, 1).

**Table 19-2.  Clock Settings**

| IICXn | IICCLn | | | Selection Clock | Transfer Clock | Settable Main Clock Frequency (f$_{XX}$) Range | Transfer Speed | Operating Mode |
|---|---|---|---|---|---|---|---|---|
| CLXn | SMCn | CLn1 | CLn0 | | | | | |
| 0 | 0 | 0 | 0 | fxx/6 (OCKSm = 11H) | fxx/264 | 24.00 MHz ≤ fxx ≤ 25.14 MHz | 90.91 kHz to 95.23 kHz | Standard mode (SMCn = 0) |
| | | | | fxx/8 (OCKSm = 12H) | fxx/352 | 24.00 MHz ≤ fxx ≤ 33.52 MHz | 68.18 kHz to 95.23 kHz | |
| | | | | fxx/10 (OCKSm = 13H) | fxx/440 | 30.00 MHz ≤ fxx ≤ 41.90 MHz | 68.18 kHz to 95.23 kHz | |
| 0 | 0 | 0 | 1 | fxx/4 (OCKSm = 10H) | fxx/344 | 24.00 MHz ≤ fxx ≤ 33.52 MHz | 48.72 kHz to 97.44 kHz | |
| | | | | fxx/6 (OCKSm = 11H) | fxx/516 | 25.14 MHz ≤ fxx ≤ 48.00 MHz | 48.72 kHz to 93.02 kHz | |
| | | | | fxx/8 (OCKSm = 12H) | fxx/688 | 33.52 MHz ≤ fxx ≤ 48.00 MHz | 48.72 kHz to 69.77 kHz | |
| | | | | fxx/10 (OCKSm = 13H) | fxx/860 | 41.90 MHz ≤ fxx ≤ 48.00 MHz | 48.72 kHz to 55.81 kHz | |
| 0 | 0 | 1 | 1 | fxx/4 (OCKSm = 10H) | fxx/264 | 24.00 MHz ≤ fxx ≤ 25.60 MHz | 90.91 kHz to 96.97 kHz | |
| | | | | fxx/6 (OCKSm = 11H) | fxx/396 | 38.40 MHz | 96.97 kHz | |
| 0 | 1 | 0 | X | fxx/4 (OCKSm = 10H) | fxx/96 | 24.00 MHz ≤ fxx ≤ 33.52 MHz | 250.00 kHz to 349.17 kHz | High-speed mode (SMCn = 1) |
| | | | | fxx/6 (OCKSm = 11H) | fxx/144 | 24.00 MHz ≤ fxx ≤ 48.00 MHz | 166.67 kHz to 333.33 kHz | |
| | | | | fxx/8 (OCKSm = 12H) | fxx/192 | 32.00 MHz ≤ fxx ≤ 48.00 MHz | 166.67 kHz to 250.00 kHz | |
| | | | | fxx/10 (OCKSm = 13H) | fxx/240 | 40.00 MHz ≤ fxx ≤ 48.00 MHz | 166.67 kHz to 200.00 kHz | |
| 0 | 1 | 1 | 1 | fxx/4 (OCKSm = 10H) | fxx/72 | 24.00 MHz ≤ fxx ≤ 25.60 MHz | 333.33 kHz to 355.56 kHz | |
| | | | | fxx/6 (OCKSm = 11H) | fxx/108 | 38.40 MHz | 355.56 kHz | |
| 1 | 1 | 0 | X | fxx/6 (OCKSm = 11H) | fxx/72 | 24.00 MHz ≤ fxx ≤ 25.14 MHz | 333.33 kHz to 349.17 kHz | |
| | | | | fxx/8 (OCKSm = 12H) | fxx/96 | 32.00 MHz ≤ fxx ≤ 33.52 MHz | 333.33 kHz to 349.17 kHz | |
| | | | | fxx/10 (OCKSm = 13H) | fxx/120 | 40.00 MHz ≤ fxx ≤ 41.90 MHz | 333.33 kHz to 349.17 kHz | |
| Other than above | | | | Setting prohibited | – | – | – | – |

**Remarks 1.**  n = 0 to 2, m = 0 and 1

**2.**  ×: don't care

**(7)  IIC division clock select registers 0, 1 (OCKS0, OCKS1)**

The OCKSm registers control the I²C0n division clock (n = 0 to 2, m = 0, 1).

These registers control the I²C00 division clock via the OCKS0 register and the I²C01 and I²C02 division clocks via the OCKS1 register.

These registers can be read or written in 8-bit units.

Reset sets these registers to 00H.

---

After reset: 00H     R/W     Address:  OCKS0 FFFFF340H, OCKS1 FFFFF344H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OCKSm | 0 | 0 | 0 | OCKSENm | OCKSTHm | 0 | OCKSm1 | OCKSm0 |

(m = 0, 1)

| OCKSENm | Operation setting of I²C division clock |
|---|---|
| 0 | Stops I²C division clock operation |
| 1 | Enables I²C division clock operation |

| OCKSTHm | OCKSm1 | OCKSm0 | Selection of I²C division clock |
|---|---|---|---|
| 0 | 0 | 0 | $f_{XX}/4$ |
| 0 | 0 | 1 | $f_{XX}/6$ |
| 0 | 1 | 0 | $f_{XX}/8$ |
| 0 | 1 | 1 | $f_{XX}/10$ |
| 1 | 0 | 0 | $f_{XX}/2$ |

---

**(8)  IIC shift registers 0 to 2 (IIC0 to IIC2)**

The IICn registers are used for serial transmission/reception (shift operations) synchronized with the serial clock. These registers can be read or written in 8-bit units, but data should not be written to the IICn registers during a data transfer.

Access (read/write) the IICn registers only during the wait period.  Accessing these registers in communication states other than the wait period is prohibited.  However, for the master device, the IICn registers can be written once only after the transmission trigger bit (IICCn.STTn bit) has been set to 1.

A wait state is released by writing the IICn registers during the wait period, and data transfer is started (n = 0 to 2).

Reset sets these registers to 00H.

---

After reset: 00H        R/W        Address: IIC0 FFFFFD80H, IIC1 FFFFFD90H, IIC2 FFFFFDA0H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IICn | | | | | | | | |

(n = 0 to 2)

---

**(9) Slave address registers 0 to 2 (SVA0 to SVA2)**

The SVAn registers hold the I²C bus's slave address.

These registers can be read or written in 8-bit units, but bit 0 should be fixed to 0. However, rewriting these registers is prohibited when the IICSn.STDn bit = 1 (start condition detection).

Reset sets these registers to 00H.

After reset: 00H        R/W        Address: SVA0 FFFFFD83H, SVA1 FFFFFD93H, SVA2 FFFFFDA3H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| SVAn | | | | | | | | 0 |

(n = 0 to 2)

## 19.5  I²C Bus Mode Functions

### 19.5.1  Pin configuration

The serial clock pin (SCL0n) and serial data bus pin (SDA0n) are configured as follows (n = 0 to 2).


SCL0n ................This pin is used for serial clock input and output.

This pin is an N-ch open-drain output for both master and slave devices.  Input is Schmitt input.

SDA0n ...............This pin is used for serial data input and output.

This pin is an N-ch open-drain output for both master and slave devices.  Input is Schmitt input.


Since outputs from the serial clock line and the serial data bus line are N-ch open-drain outputs, an external pull-up resistor is required.

**Figure 19-6.  Pin Configuration Diagram**
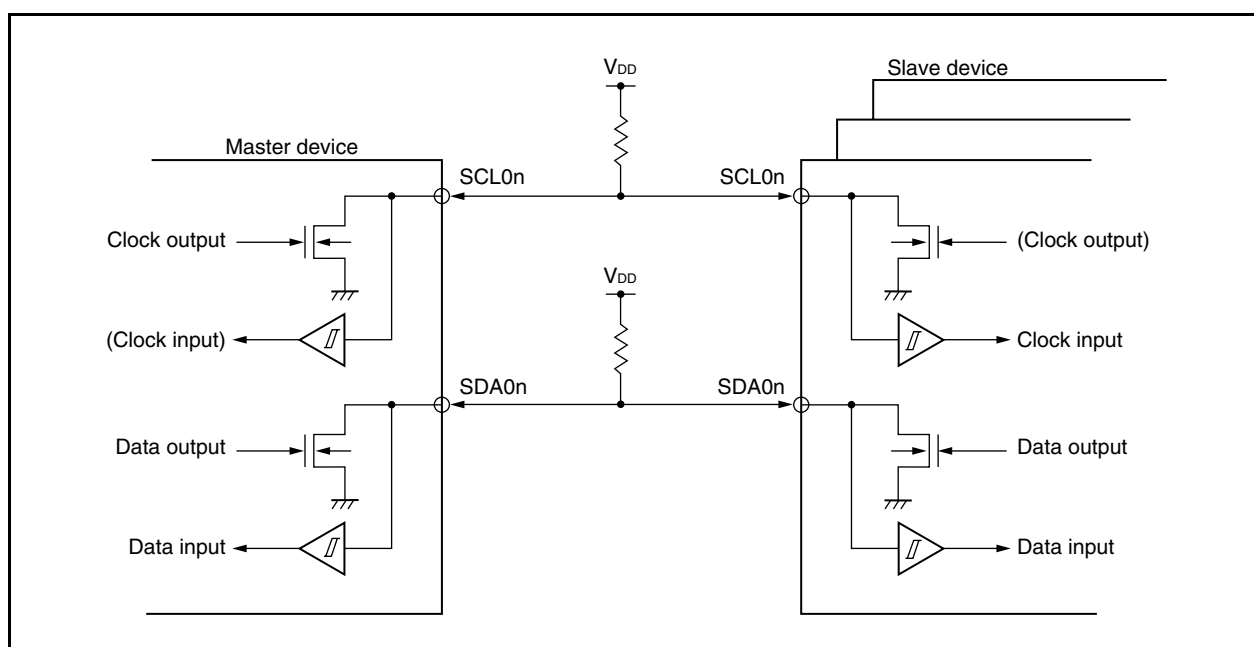
## 19.6  I²C Bus Definitions and Control Methods

The following section describes the I²C bus's serial data communication format and the signals used by the I²C bus.

The transfer timing for the "start condition", "address", "transfer direction specification", "data", and "stop condition" generated on the I²C bus's serial data bus is shown below.

**Figure 19-7.  I²C Bus Serial Data Transfer Timing**
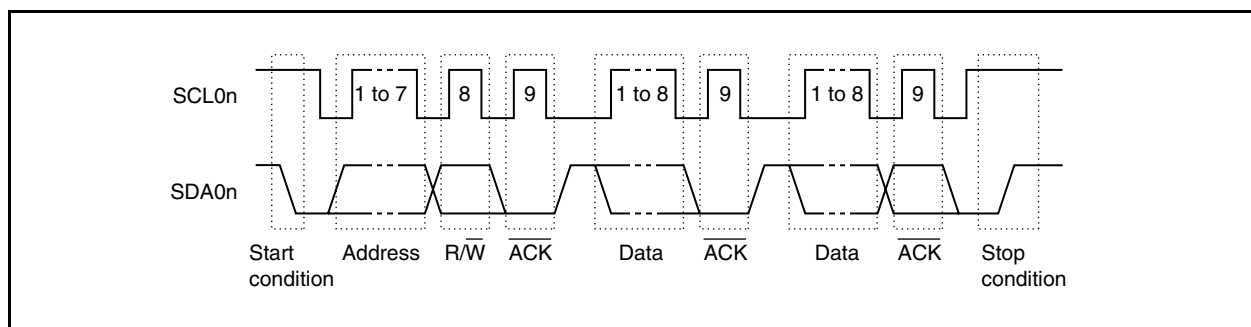


The master device generates the start condition, slave address, and stop condition.

$\overline{ACK}$ can be generated by either the master or slave device (normally, it is generated by the device that receives 8-bit data).

The serial clock (SCL0n) is continuously output by the master device.  However, in the slave device, the SCL0n pin's low-level period can be extended and a wait state can be inserted (n = 0 to 2).

### 19.6.1  Start condition

A start condition is met when the SCL0n pin is high level and the SDA0n pin changes from high level to low level.  The start condition for the SCL0n and SDA0n pins is a signal that the master device outputs to the slave device when starting a serial transfer.  The slave device can detect the start condition (n = 0 to 2).

**Figure 19-8.  Start Condition**
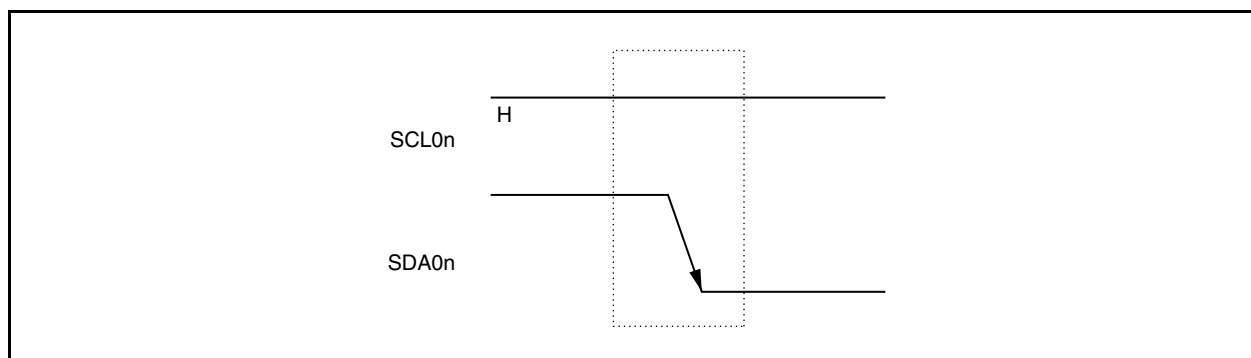


A start condition is output when the IICCn.STTn bit is set (1) after a stop condition has been detected (IICSn.SPDn bit = 1).  When a start condition is detected, the IICSn.STDn bit is set (1) (n = 0 to 2).

**Caution    When the IICCn.IICEn bit of the V850ES/JG3-U and V850ES/JH3-U is set to 1 while other devices are communicating, the start condition may be detected depending on the status of the communication line.  Be sure to set the IICCn.IICEn bit to 1 when the SCL0n and SDA0n lines are high level.**

**19.6.2  Addresses**

The 7 bits of data that follow the start condition are defined as an address.

An address is a 7-bit data segment that is output so that the master device can select one of the slave devices that are connected to the bus lines.  Therefore, each slave device connected via the bus lines must have a unique address.

The slave devices detect via hardware the start condition and check whether or not the 7-bit address data matches the data values stored in the SVAn register.  If the address data matches the values of the SVAn register, the slave device is selected and communicates with the master device until the master device generates a start condition or stop condition (n = 0 to 2).

**Figure 19-9.  Address**



**Note**  The interrupt request signal (INTIICn) is generated if a local address or extension code is received during slave device operation.

**Remark**   n = 0 to 2
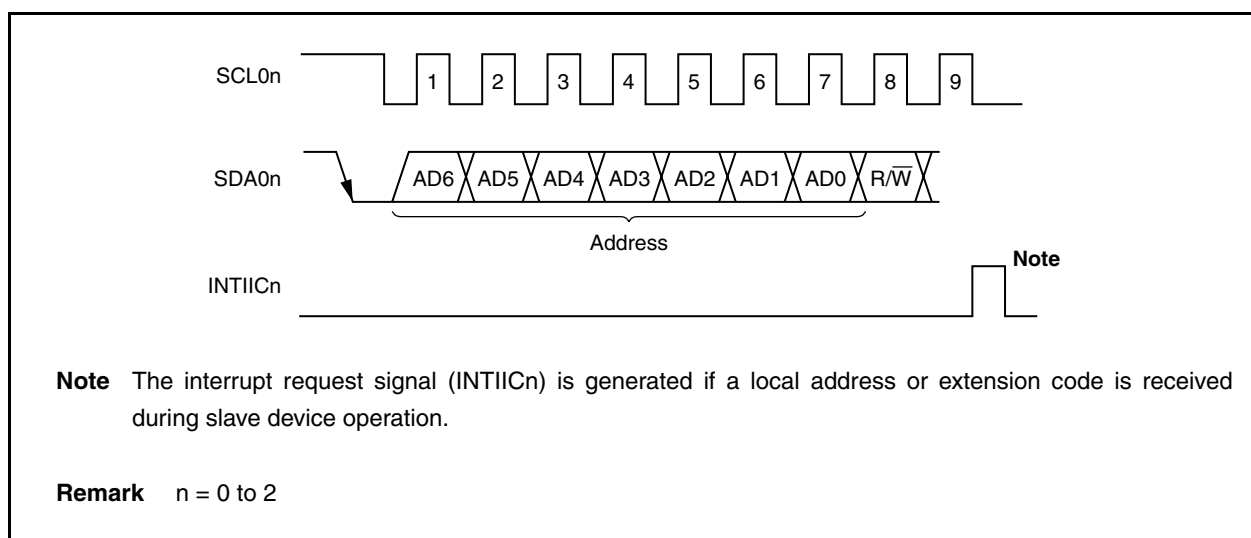
An address is output when the slave address and the transfer direction described in **19.6.3  Transfer direction specification** are written together to the IICn registers as eight bits of data.  Received addresses are written to the IICn register (n = 0 to 2).

The slave address is assigned to the higher 7 bits of the IICn register.

**19.6.3  Transfer direction specification**

In addition to the 7-bit address data, the master device sends 1 bit of data that specifies the transfer direction.

When this transfer direction specification bit has a value of 0, it indicates that the master device is transmitting data to a slave device.  When the transfer direction specification bit has a value of 1, it indicates that the master device is receiving data from a slave device.

**Figure 19-10.  Transfer Direction Specification**



**Note**  The INTIICn signal is generated if a local address or extension code is received during slave device operation.

**Remark**   n = 0 to 2
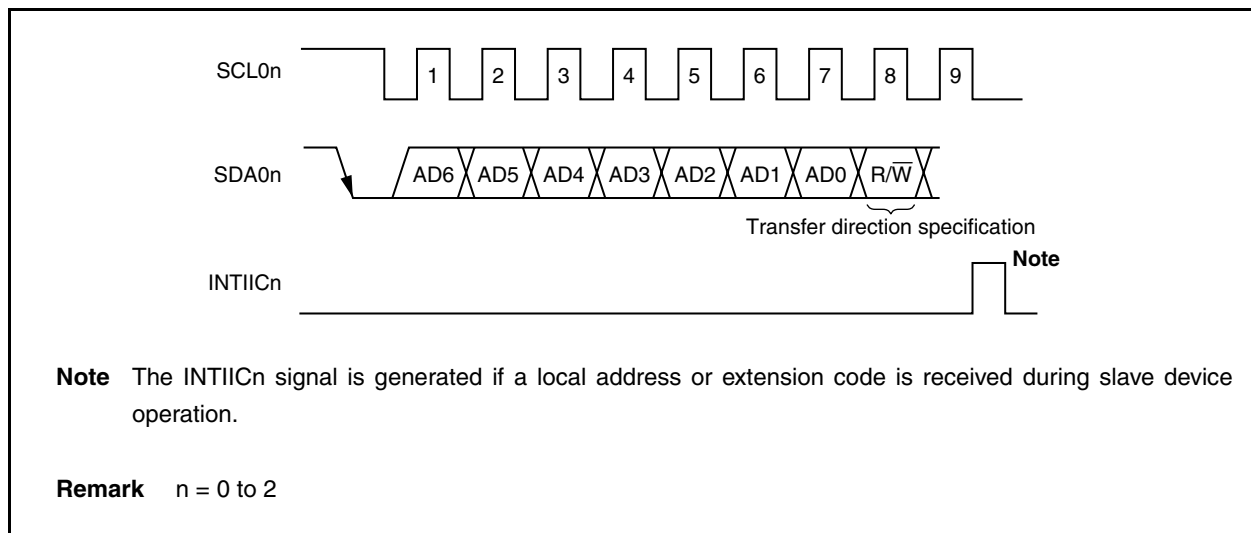
**19.6.4  $\overline{\text{ACK}}$**

$\overline{\text{ACK}}$ is used to confirm the serial data status of the transmitting and receiving devices.

The receiving device returns $\overline{\text{ACK}}$ for every 8 bits of data it receives.

The transmitting device normally receives $\overline{\text{ACK}}$ after transmitting 8 bits of data.  When $\overline{\text{ACK}}$ is returned from the receiving device, the reception is judged as normal and processing continues.  The detection of $\overline{\text{ACK}}$ is confirmed using the IICSn.ACKDn bit.

When the master device is the receiving device, after receiving the final data, it does not return $\overline{\text{ACK}}$ and generates a stop condition.  When the slave device is the receiving device and does not return $\overline{\text{ACK}}$, the master device generates either a stop condition or a restart condition, and then stops the current transmission.  Failure to return $\overline{\text{ACK}}$ may be caused by the following factors.

(a)  Reception was not performed normally.

(b)  The final data was received.

(c)  The receiving device (slave) does not exist at the specified address.

When the receiving device sets the SDA0n line to low level during the ninth clock, $\overline{\text{ACK}}$ is generated (normal reception).

When the IICCn.ACKEn bit is set to 1, automatic $\overline{\text{ACK}}$ generation is enabled.  Transmission of the eighth bit following the 7 address data bits causes the IICSn.TRCn bit to be set.  Normally, set the ACKEn bit to 1 for reception (TRCn bit = 0).

When the slave device is receiving (when TRCn bit = 0), if the slave device cannot receive data or does not need the subsequent data, clear the ACKEn bit to 0 to indicate to the master that no more data can be received.

Similarly, when the master device is receiving (when TRCn bit = 0) and the subsequent data is not needed, clear the ACKEn bit to 0 to prevent $\overline{\text{ACK}}$ from being generated.  This notifies the slave device (transmitting device) of the end of the data transmission (transmission stopped).

**Figure 19-11.  $\overline{\text{ACK}}$**



**Remark**  n = 0 to 2
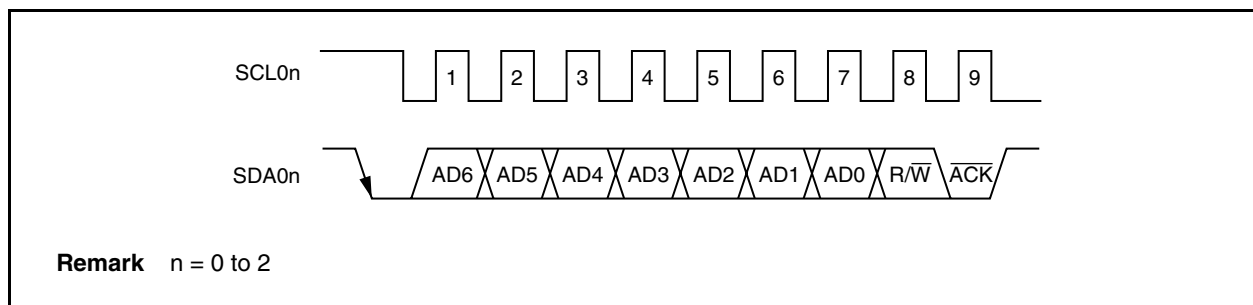
When the local address is received, $\overline{\text{ACK}}$ is automatically generated regardless of the value of the ACKEn bit.  No $\overline{\text{ACK}}$ is generated if an address other than the local address is received (NACK).

When receiving the extension code, set the ACKEn bit to 1 in advance to generate $\overline{\text{ACK}}$.

The $\overline{\text{ACK}}$ generation method during data reception is based on the wait timing setting, as described by the following.

• When 8-clock wait is selected (IICCn.WTIMn bit = 0):

  $\overline{\text{ACK}}$ is generated in synchronization with the falling edge of the SCL0n pin's eighth clock if the ACKEn bit is set to 1 before wait state cancellation.

• When 9-clock wait is selected (IICCn.WTIMn bit = 1):

  $\overline{\text{ACK}}$ is generated if the ACKEn bit is set to 1 in advance.

**Remark**  n = 0 to 2

**19.6.5  Stop condition**

When the SCL0n pin is high level, changing the SDA0n pin from low level to high level generates a stop condition (n = 0 to 2).

A stop condition is generated when serial transfer from the master device to the slave device has been completed. When the V850ES/JG3-U or V850ES/JH3-U is used as the slave device, it can detect the stop condition.

**Figure 19-12.  Stop Condition**



**Remark**   n = 0 to 2
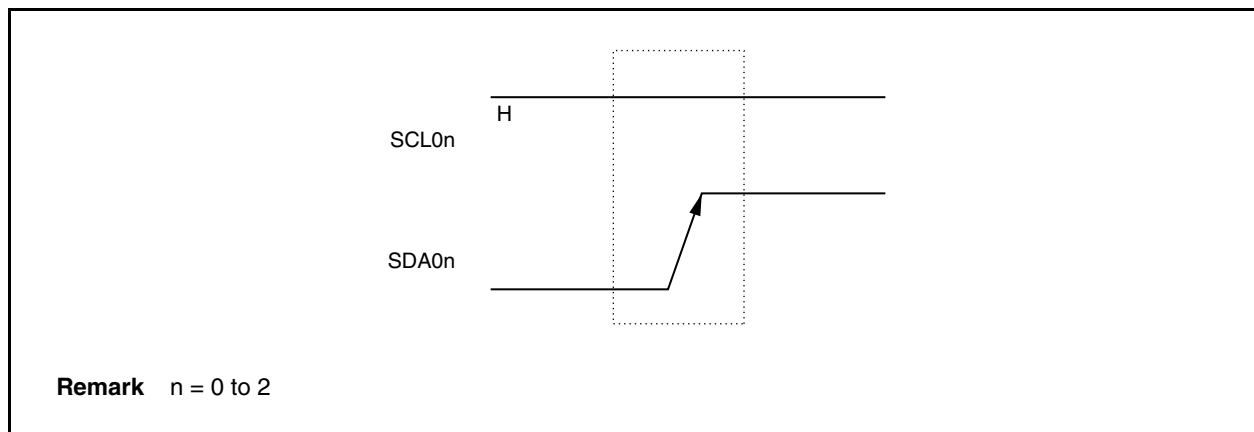
A stop condition is generated when the IICCn.SPTn bit is set to 1.  When the stop condition is detected, the IICSn.SPDn bit is set to 1 and the interrupt request signal (INTIICn) is generated when the IICCn.SPIEn bit is set to 1 (n = 0 to 2).

**19.6.6  Wait state**

A wait state is used to notify the communication partner that a device (master or slave) is preparing to transmit or receive data (i.e., is in a wait state).

Setting the SCL0n pin to low level notifies the communication partner of the wait state.  When the wait state has been canceled for both the master and slave devices, the next data transfer can begin (n = 0 to 2).

**Figure 19-13.  Wait State (1/2)**



**(a)  When master device is in a nine-clock wait state and slave device is in an eight-clock wait state (master: transmission, slave: reception, and IICCn.ACKEn bit = 1)**

**Remark**   n = 0 to 2

**Figure 19-13. Wait State (2/2)**



**(b) When master and slave devices are both in a nine-clock wait state
(master: transmission, slave: reception, and ACKEn bit = 1)**
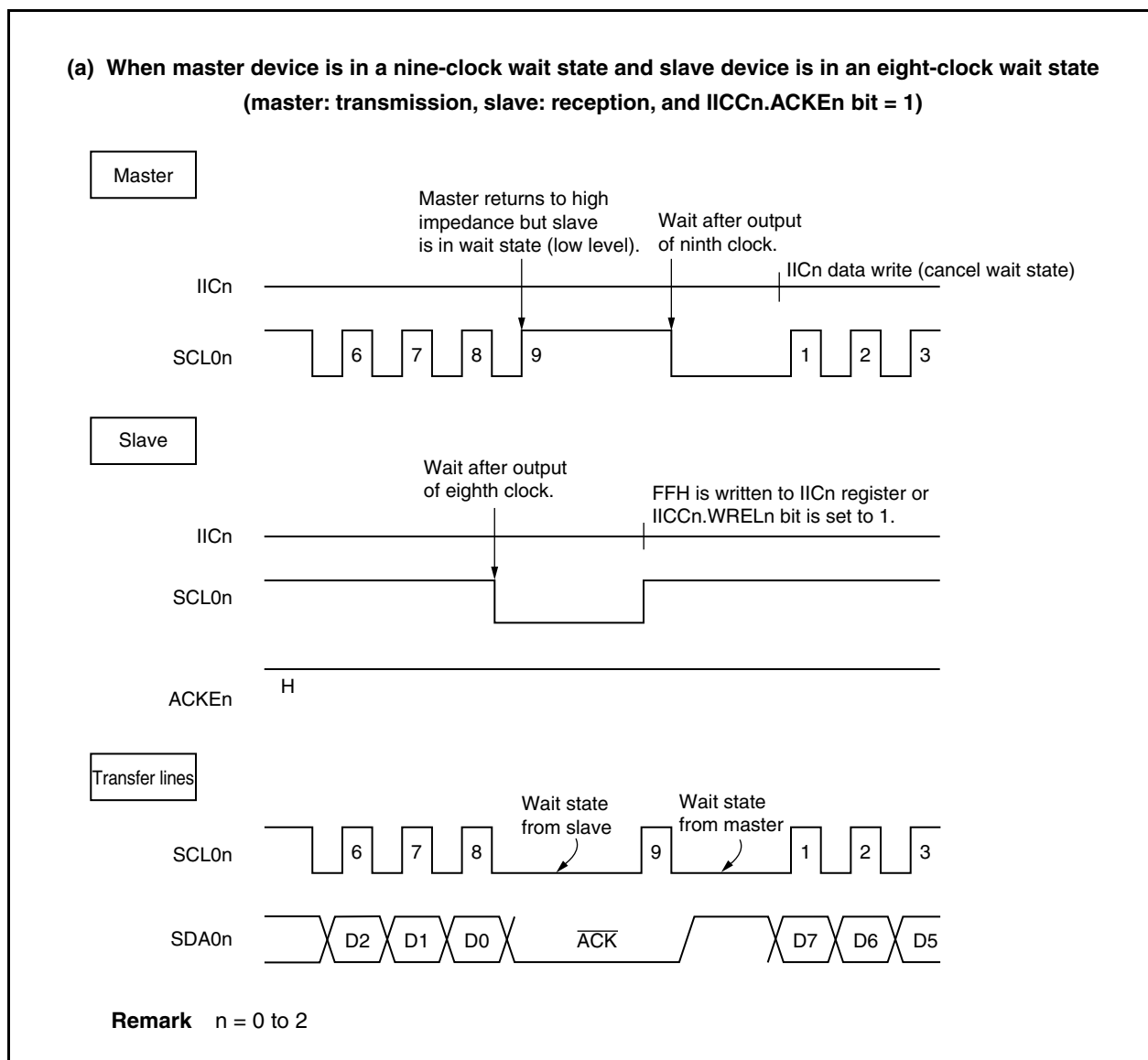
**Remark** n = 0 to 2
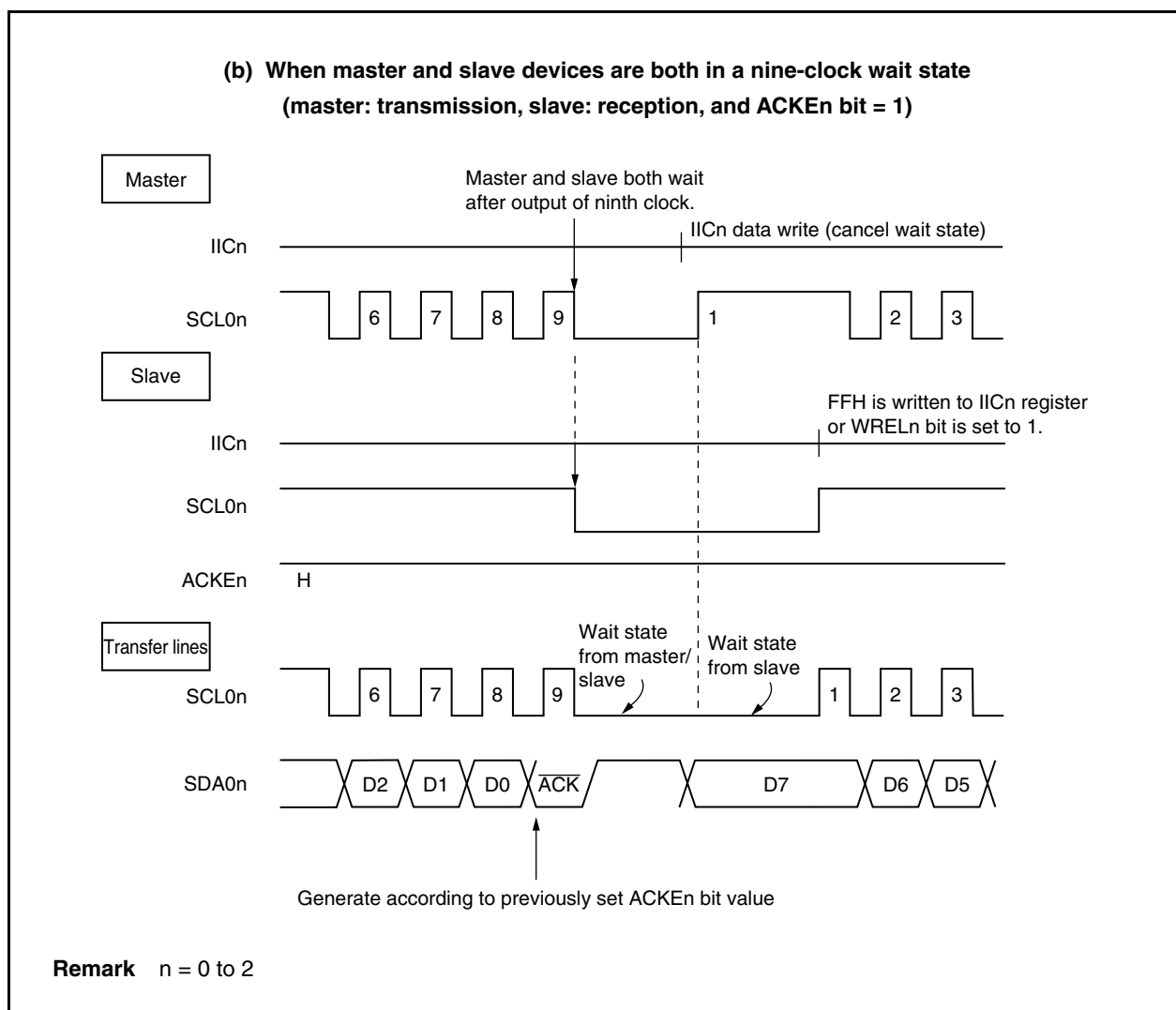
A wait state may be automatically generated depending on the setting of the IICCn.WTIMn bit (n = 0 to 2).

Normally, the receiving side cancels the wait state when the IICCn.WRELn bit is set to 1 or when FFH is written to the IICn register and the transmitting side cancels the wait state when data is written to the IICn register.

The master device can also cancel the wait state via either of the following methods.

- By setting the IICCn.STTn bit to 1
- By setting the IICCn.SPTn bit to 1

### 19.6.7  Wait state cancellation method

In the case of I²C0n, a wait state can be canceled normally in the following ways (n = 0 to 2).

- By writing data to the IICn register
- By setting the IICCn.WRELn bit to 1 (wait state cancellation)
- By setting the IICCn.STTn bit to 1 (start condition generation)
- By setting the IICCn.SPTn bit to 1 (stop condition generation)

If any of these wait state cancellation actions is performed, I²C0n will cancel the wait state and restart communication.

When canceling the wait state and sending data (including address), write data to the IICn register.

To receive data after canceling the wait state, or to complete data transmission, set the WRELn bit to 1.

To generate a restart condition after canceling the wait state, set the STTn bit to 1.

To generate a stop condition after canceling the wait state, set the SPTn bit to 1.

Execute cancellation only once for each wait state.

For example, if data is written to the IICn register following wait state cancellation by setting the WRELn bit to 1, a conflict between the SDA0n line change timing and IICn register write timing may result in the data output to SDA0n being an incorrect value.

Even in other operations, if communication is stopped halfway, clearing the IICCn.IICEn bit to 0 will stop communication, enabling the wait state to be cancelled.

If the I²C bus dead-locks due to noise, etc., setting the IICCn.LRELn bit to 1 causes the communication operation to be exited, enabling the wait state to be cancelled.

## 19.7  I²C Interrupt Request Signals (INTIICn)

The following shows the value of the IICSn register at the INTIICn interrupt request signal generation timing and at the INTIICn signal timing (n = 0 to 2).

### 19.7.1  Master device operation

**(1)  Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)**

**<1>  When IICCn.WTIMn bit = 0**

IICCn.SPTn bit = 1
↓

| ST | AD6 to AD0 | R/$\overline{W}$ | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | SP |
|----|-----------|------|------|----------|------|----------|------|----|

▲1    ▲2    ▲3  ▲4    Δ5
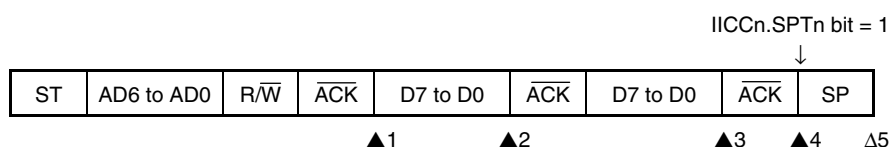
▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X000B

▲3: IICSn register = 1000X000B (WTIMn bit = 1)

▲4: IICSn register = 1000XX00B

Δ 5: IICSn register = 00000001B

**Remarks 1.** ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

**2.** n = 0 to 2

**<2>  When WTIMn bit = 1**

SPTn bit = 1
↓

| ST | AD6 to AD0 | R/$\overline{W}$ | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | SP |
|----|-----------|------|------|----------|------|----------|------|----|

▲1         ▲2         ▲3    Δ4

▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X100B

▲3: IICSn register = 1000XX00B

Δ 4: IICSn register = 00000001B

**Remarks 1.** ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

**2.** n = 0 to 2

**(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)**

**<1> When WTIMn bit = 0**

STTn bit = 1
↓

SPTn bit = 1
↓

| ST | AD6 to AD0 | R/W̄ | ĀCK | D7 to D0 | ĀCK | ST | AD6 to AD0 | R/W̄ | ĀCK | D7 to D0 | ĀCK | SP |
|----|------------|-----|-----|----------|-----|----|------------|-----|-----|----------|-----|-----|

▲1       ▲2   ▲3                    ▲4       ▲5   ▲6   △7

▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X000B (WTIMn bit = 1)

▲3: IICSn register = 1000XX00B (WTIMn bit = 0)

▲4: IICSn register = 1000X110B (WTIMn bit = 0)

▲5: IICSn register = 1000X000B (WTIMn bit = 1)

▲6: IICSn register = 1000XX00B

△7: IICSn register = 00000001B

**Remarks 1. ▲:** Always generated

**△:** Generated only when SPIEn bit = 1

**X:** don't care

**2.** n = 0 to 2

**<2> When WTIMn bit = 1**

STTn bit = 1
↓

SPTn bit = 1
↓

| ST | AD6 to AD0 | R/W̄ | ĀCK | D7 to D0 | ĀCK | ST | AD6 to AD0 | R/W̄ | ĀCK | D7 to D0 | ĀCK | SP |
|----|------------|-----|-----|----------|-----|----|------------|-----|-----|----------|-----|-----|

▲1              ▲2                     ▲3              ▲4   △5
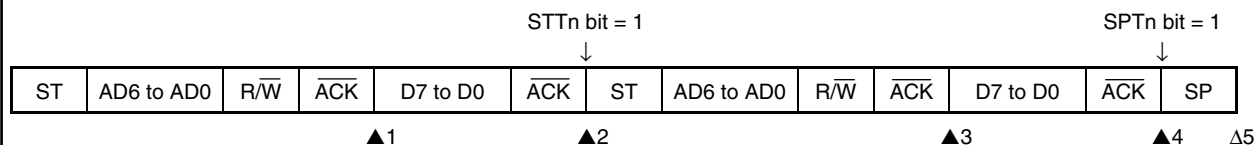
▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000XX00B

▲3: IICSn register = 1000X110B

▲4: IICSn register = 1000XX00B

△5: IICSn register = 00000001B

**Remarks 1. ▲:** Always generated

**△:** Generated only when SPIEn bit = 1

**X:** don't care

**2.** n = 0 to 2

**(3)  Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)**

**<1>  When WTIMn bit = 0**

SPTn bit = 1
↓

| ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | D7 to D0 | ACK̄ | SP |
|----|------------|-----|------|----------|------|----------|------|-----|

▲1       ▲2            ▲3   ▲4    △5

▲1: IICSn register = 1010X110B

▲2: IICSn register = 1010X000B

▲3: IICSn register = 1010X000B (WTIMn bit = 1)

▲4: IICSn register = 1010XX00B

△ 5: IICSn register = 00000001B

**Remarks 1.** ▲:  Always generated

△:   Generated only when SPIEn bit = 1

X:   don't care

**2.** n = 0 to 2

**<2>  When WTIMn bit = 1**

SPTn bit = 1
↓

| ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | D7 to D0 | ACK̄ | SP |
|----|------------|-----|------|----------|------|----------|------|-----|

▲1              ▲2            ▲3    △4
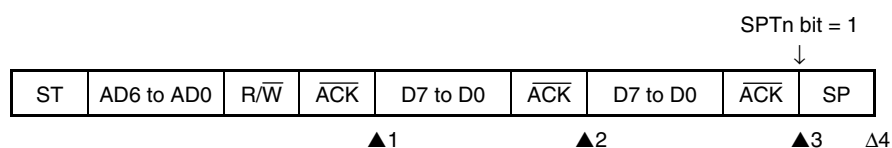
▲1: IICSn register = 1010X110B

▲2: IICSn register = 1010X100B

▲3: IICSn register = 1010XX00B

△ 4: IICSn register = 00000001B

**Remarks 1.** ▲:  Always generated

△:   Generated only when SPIEn bit = 1

X:   don't care

**2.** n = 0 to 2

**19.7.2 Slave device operation (when receiving slave address data (address match))**

**(1) Start ~ Address ~ Data ~ Data ~ Stop**

**<1> When IICCn.WTIMn bit = 0**

| ST | AD6 to AD0 | R/$\overline{W}$ | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | SP |
|----|-----------|-----|-----|---------|-----|---------|-----|----|

▲1          ▲2              ▲3          Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0001X000B

Δ 4: IICSn register = 00000001B

**Remarks 1.** ▲: Always generated

Δ:  Generated only when IICCn.SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**<2> When WTIMn bit = 1**

| ST | AD6 to AD0 | R/$\overline{W}$ | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | SP |
|----|-----------|-----|-----|---------|-----|---------|-----|----|

▲1                ▲2              ▲3    Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X100B

▲3: IICSn register = 0001XX00B

Δ 4: IICSn register = 00000001B

**Remarks 1.** ▲: Always generated

Δ:  Generated only when SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**(2)  Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop**

**<1>  When WTIMn bit = 0 (after restart, address match)**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|-----------|------|------|----------|------|----|-----------|------|------|----------|------|----|
|    |           |      | ▲1   |          | ▲2   |    |           |      |      | ▲3       | ▲4   | ∆5 |

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0001X110B

▲4: IICSn register = 0001X000B

∆ 5: IICSn register = 00000001B

**Remarks 1.** ▲: Always generated

∆:  Generated only when SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**<2>  When WTIMn bit = 1 (after restart, address match)**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|-----------|------|------|----------|------|----|-----------|------|------|----------|------|----|
|    |           |      | ▲1   |          | ▲2   |    |           |      |      | ▲3       | ▲4   | ∆5 |

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001XX00B

▲3: IICSn register = 0001X110B

▲4: IICSn register = 0001XX00B

∆ 5: IICSn register = 00000001B

**Remarks 1.** ▲: Always generated

∆:  Generated only when SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop**

**<1> When WTIMn bit = 0 (after restart, extension code reception)**

| ST | AD6 to AD0 | R/$\overline{W}$ | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | ST | AD6 to AD0 | R/$\overline{W}$ | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | SP |
|----|-----------|------|------|----------|------|----|-----------|------|------|----------|------|----|

▲1    ▲2                              ▲3              ▲4              △5

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0010X010B

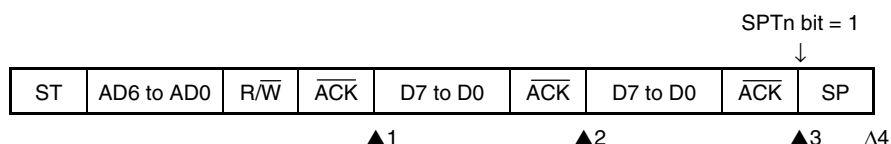▲4: IICSn register = 0010X000B

△5: IICSn register = 00000001B

**Remarks 1.** ▲:  Always generated

△:  Generated only when SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**<2> When WTIMn bit = 1 (after restart, extension code reception)**

| ST | AD6 to AD0 | R/$\overline{W}$ | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | ST | AD6 to AD0 | R/$\overline{W}$ | $\overline{ACK}$ | D7 to D0 | $\overline{ACK}$ | SP |
|----|-----------|------|------|----------|------|----|-----------|------|------|----------|------|----|

▲1    ▲2                              ▲3    ▲4                    ▲5    △6

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001XX00B

▲3: IICSn register = 0010X010B

▲4: IICSn register = 0010X110B

▲5: IICSn register = 0010XX00B

△6: IICSn register = 00000001B

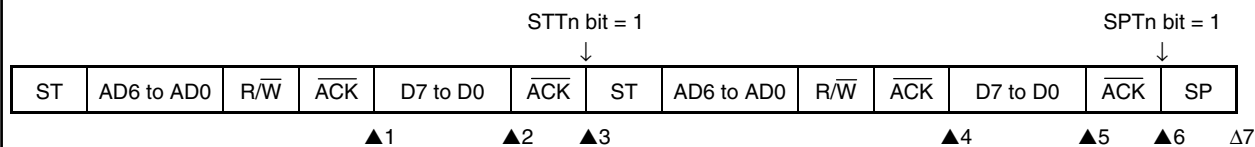**Remarks 1.** ▲:  Always generated

△:  Generated only when SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**(4)  Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop**

**<1>  When WTIMn bit = 0 (after restart, address mismatch (= not extension code))**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|-----------|------|------|---------|------|----|-----------|------|------|---------|------|----|

             ▲1          ▲2                         ▲3        Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 00000X10B

Δ 4: IICSn register = 00000001B

**Remarks 1.** ▲:  Always generated

          Δ:  Generated only when SPIEn bit = 1

          X:  don't care

     **2.** n = 0 to 2

**<2>  When WTIMn bit = 1 (after restart, address mismatch (= not extension code))**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|-----------|------|------|---------|------|----|-----------|------|------|---------|------|----|

             ▲1          ▲2                         ▲3        Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001XX00B

▲3: IICSn register = 00000X10B

Δ 4: IICSn register = 00000001B
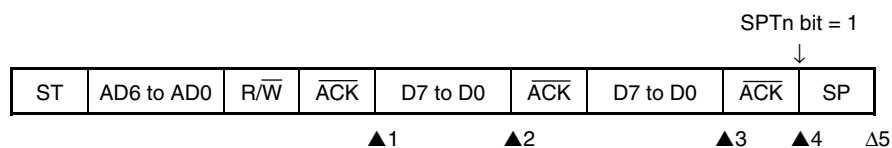
**Remarks 1.** ▲:  Always generated

          Δ:  Generated only when SPIEn bit = 1

          X:  don't care

     **2.** n = 0 to 2

**19.7.3  Slave device operation (when receiving extension code)**

**(1)  Start ~ Code ~ Data ~ Data ~ Stop**

**<1>  When IICCn.WTIMn bit = 0**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|------------|------|------|----------|------|----------|------|----|

        ▲1        ▲2        ▲3     △4

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0010X000B

△ 4: IICSn register = 00000001B

**Remarks  1.** ▲:  Always generated

△:  Generated only when IICCn.SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**<2>  When WTIMn bit = 1**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|------------|------|------|----------|------|----------|------|----|

       ▲1 ▲2       ▲3      ▲4 △5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010X100B

▲4: IICSn register = 0010XX00B

△ 5: IICSn register = 00000001B

**Remarks  1.** ▲:  Always generated
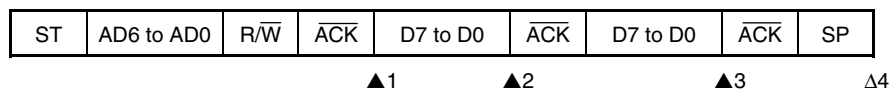
△:  Generated only when SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop**

**<1> When WTIMn bit = 0 (after restart, address match)**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|-----------|-----|-----|---------|-----|----|-----------|-----|-----|---------|-----|----|

▲1             ▲2                                  ▲3      ▲4        △5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

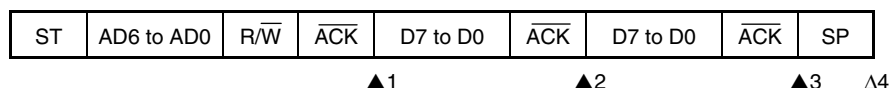▲3: IICSn register = 0001X110B

▲4: IICSn register = 0001X000B

△ 5: IICSn register = 00000001B

**Remarks 1.** ▲:  Always generated

　　　　　△:  Generated only when SPIEn bit = 1

　　　　　X:  don't care

**2.** n = 0 to 2

**<2> When WTIMn bit = 1 (after restart, address match)**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|-----------|-----|-----|---------|-----|----|-----------|-----|-----|---------|-----|----|

▲1     ▲2              ▲3                           ▲4        ▲5     △6

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010XX00B

▲4: IICSn register = 0001X110B

▲5: IICSn register = 0001XX00B

△ 6: IICSn register = 00000001B

**Remarks 1.** ▲:  Always generated

　　　　　△:  Generated only when SPIEn bit = 1

　　　　　X:  don't care

**2.** n = 0 to 2

**(3)  Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop**

**<1>  When WTIMn bit = 0 (after restart, extension code reception)**

| ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | SP |
|----|-----------|-----|------|----------|------|----|-----------|-----|------|----------|------|-----|

▲1     ▲2          ▲3       ▲4      △5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0010X010B

▲4: IICSn register = 0010X000B

△ 5: IICSn register = 00000001B

**Remarks 1.** ▲:  Always generated

△:  Generated only when SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**<2>  When WTIMn bit = 1 (after restart, extension code reception)**

| ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | SP |
|----|-----------|-----|------|----------|------|----|-----------|-----|------|----------|------|-----|

▲1   ▲2            ▲3              ▲4   ▲5              ▲6  △7

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010XX00B

▲4: IICSn register = 0010X010B

▲5: IICSn register = 0010X110B

▲6: IICSn register = 0010XX00B

△ 7: IICSn register = 00000001B
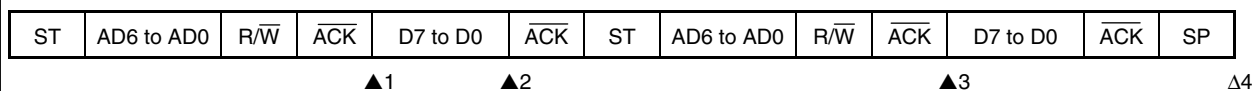
**Remarks 1.** ▲:  Always generated

△:  Generated only when SPIEn bit = 1

X:  don't care

**2.** n = 0 to 2

**(4)  Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop**

**<1>  When WTIMn bit = 0 (after restart, address mismatch (= not extension code))**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|------------|------|------|----------|------|----|------------|------|------|----------|------|----|

▲1                        ▲2                                                                     ▲3                    Δ4

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 00000X10B
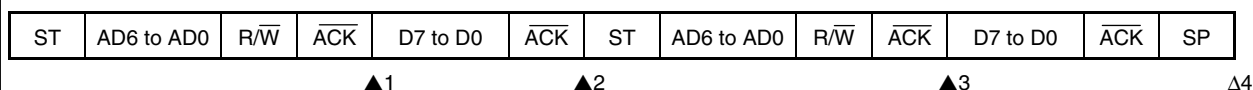
Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

**<2>  When WTIMn bit = 1 (after restart, address mismatch (= not extension code))**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|------------|------|------|----------|------|----|------------|------|------|----------|------|----|

▲1      ▲2                    ▲3                                                       ▲4                    Δ5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010XX00B

▲4: IICSn register = 00000X10B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

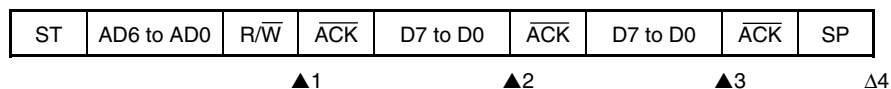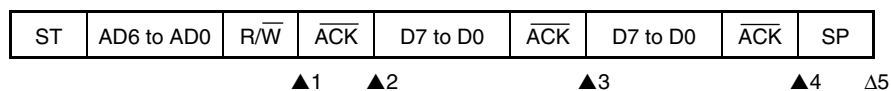Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

### 19.7.4  Operation without communication

**(1)  Start ~ Code ~ Data ~ Data ~ Stop**

| ST | AD6 to AD0 | R/W̅ | A̅C̅K̅ | D7 to D0 | A̅C̅K̅ | D7 to D0 | A̅C̅K̅ | SP |
|----|------------|-----|-----|----------|-----|----------|-----|----|

Δ1

Δ 1: IICSn register = 00000001B

**Remarks  1.** Δ:   Generated only when SPIEn bit = 1
**2.** n = 0 to 2

### 19.7.5  Arbitration loss operation (operation as slave after arbitration loss)

**(1)  When arbitration loss occurs during transmission of slave address data**

**<1>  When IICCn.WTIMn bit = 0**

| ST | AD6 to AD0 | R/W̅ | A̅C̅K̅ | D7 to D0 | A̅C̅K̅ | D7 to D0 | A̅C̅K̅ | SP |
|----|------------|-----|-----|----------|-----|----------|-----|----|

▲1        ▲2              ▲3        Δ4

▲1: IICSn register = 0101X110B (Example: When IICSn.ALDn bit is read during interrupt servicing)

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0001X000B

Δ 4: IICSn register = 00000001B

**Remarks  1.** ▲:  Always generated
Δ:  Generated only when IICCn.SPIEn bit = 1
X:  don't care
**2.** n = 0 to 2

**<2>  When WTIMn bit = 1**

| ST | AD6 to AD0 | R/W̅ | A̅C̅K̅ | D7 to D0 | A̅C̅K̅ | D7 to D0 | A̅C̅K̅ | SP |
|----|------------|-----|-----|----------|-----|----------|-----|----|

▲1              ▲2            ▲3  Δ4

▲1: IICSn register = 0101X110B (Example: When ALDn bit is read during interrupt servicing)

▲2: IICSn register = 0001X100B

▲3: IICSn register = 0001XX00B

Δ 4: IICSn register = 00000001B

**Remarks  1.** ▲:  Always generated
Δ:  Generated only when SPIEn bit = 1
X:  don't care
**2.** n = 0 to 2

**(2)  When arbitration loss occurs during transmission of extension code**

---

**<1>  When WTIMn bit = 0**

| ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | D7 to D0 | ACK̄ | SP |
|----|-----------|------|------|----------|------|----------|------|-----|

            ▲1         ▲2         ▲3      Δ4

▲1: IICSn register = 0110X010B (Example: When ALDn bit is read during interrupt servicing)

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0010X000B

Δ 4: IICSn register = 00000001B

    **Remarks 1.** ▲:  Always generated

              Δ:  Generated only when SPIEn bit = 1

              X:  don't care

        **2.** n = 0 to 2

**<2>  When WTIMn bit = 1**

| ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | D7 to D0 | ACK̄ | SP |
|----|-----------|------|------|----------|------|----------|------|-----|

            ▲1  ▲2         ▲3       ▲4  Δ5

▲1: IICSn register = 0110X010B (Example: When ALDn bit is read during interrupt servicing)

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010X100B

▲4: IICSn register = 0010XX00B

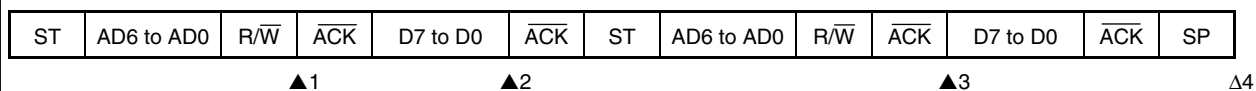Δ 5: IICSn register = 00000001B

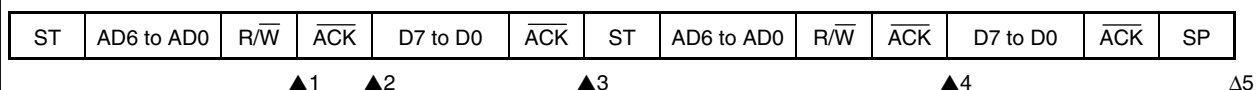    **Remarks 1.** ▲:  Always generated

              Δ:  Generated only when SPIEn bit = 1

              X:  don't care

        **2.** n = 0 to 2

**19.7.6  Operation when arbitration loss occurs (no communication after arbitration loss)**

**(1)  When arbitration loss occurs during transmission of slave address data**

| ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | D7 to D0 | ACK̄ | SP |
|----|------------|-----|------|----------|------|----------|------|----|

▲1                                                                              △2

▲1: IICSn register = 01000110B (Example: When IICSn.ALDn bit is read during interrupt servicing)

△ 2: IICSn register = 00000001B

   **Remarks 1.** ▲:  Always generated

          △:   Generated only when IICCn.SPIEn bit = 1

        **2.** n = 0 to 2

**(2)  When arbitration loss occurs during transmission of extension code**

| ST | AD6 to AD0 | R/W̄ | ACK̄ | D7 to D0 | ACK̄ | D7 to D0 | ACK̄ | SP |
|----|------------|-----|------|----------|------|----------|------|----|

▲1                                                                              △2

▲1:        IICSn register = 0110X010B (Example: When ALDn bit is read during interrupt servicing)

IICCn.LRELn bit is set to 1 by software

△ 2:        IICSn register = 00000001B

   **Remarks 1.** ▲:  Always generated

          △:   Generated only when SPIEn bit = 1

          X:   don't care

        **2.** n = 0 to 2

**(3) When arbitration loss occurs during data transfer**

**<1> When IICCn.WTIMn bit = 0**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|---|---|---|---|---|---|---|---|---|

             ▲1        ▲2              △3

▲1: IICSn register = 10001110B

▲2: IICSn register = 01000000B (Example: When ALDn bit is read during interrupt servicing)

△3: IICSn register = 00000001B

    **Remarks 1.** ▲: Always generated

           △: Generated only when SPIEn bit = 1

       **2.** n = 0 to 2

**<2> When WTIMn bit = 1**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|---|---|---|---|---|---|---|---|---|

             ▲1         ▲2              △3

▲1: IICSn register = 10001110B

▲2: IICSn register = 01000100B (Example: When ALDn bit is read during interrupt servicing)

△3: IICSn register = 00000001B

    **Remarks 1.** ▲: Always generated

           △: Generated only when SPIEn bit = 1

       **2.** n = 0 to 2

**(4)  When arbitration loss occurs due to restart condition during data transfer**

**<1>  Not extension code (Example: Address mismatch)**

| ST | AD6 to AD0 | R/W̄ | ĀC̄K̄ | D7 to Dn | ST | AD6 to AD0 | R/W̄ | ĀC̄K̄ | D7 to D0 | ĀC̄K̄ | SP |
|----|------------|-----|------|----------|----|------------|-----|------|----------|------|----|

▲1: IICSn register = 1000X110B

▲2: IICSn register = 01000110B (Example: When ALDn bit is read during interrupt servicing)

∆ 3: IICSn register = 00000001B

> **Remarks 1.** ▲:  Always generated
> ∆:  Generated only when SPIEn bit = 1
> X:  don't care
> **2.** Dn = D6 to D0
> n = 0 to 2

**<2>  Extension code**

| ST | AD6 to AD0 | R/W̄ | ĀC̄K̄ | D7 to Dn | ST | AD6 to AD0 | R/W̄ | ĀC̄K̄ | D7 to D0 | ĀC̄K̄ | SP |
|----|------------|-----|------|----------|----|------------|-----|------|----------|------|----|

▲1: IICSn register = 1000X110B

▲2: IICSn register = 0110X010B (Example: When ALDn bit is read during interrupt servicing)

IICCn.LRELn bit is set to 1 by software

∆ 3: IICSn register = 00000001B

> **Remarks 1.** ▲:  Always generated
> ∆:  Generated only when SPIEn bit = 1
> X:  don't care
> **2.** Dn = D6 to D0
> n = 0 to 2

**(5)  When arbitration loss occurs due to stop condition during data transfer**

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to Dn | SP |
|----|-----------|------|------|----------|-----|

▲1: IICSn register = 1000X110B

△ 2: IICSn register = 01000001B

> **Remarks  1.** ▲: Always generated
>
> △: Generated only when SPIEn bit = 1
>
> X: don't care
>
> **2.** Dn = D6 to D0
>
> n = 0 to 2

**(6) When arbitration loss occurs due to low level of SDA0n pin when attempting to generate a restart condition**

**<1> When WTIMn bit = 0**

IICCn.STTn bit = 1
↓

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|-----------|------|-----|----------|-----|----------|-----|----------|-----|----|

▲1       ▲2  ▲3       ▲4                                  Δ5

▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X000B (WTIMn bit = 1)

▲3: IICSn register = 1000XX00B (WTIMn bit = 0)

▲4: IICSn register = 01000000B (Example: When ALDn bit is read during interrupt servicing)
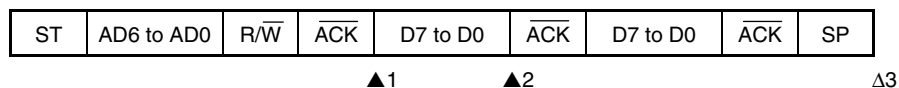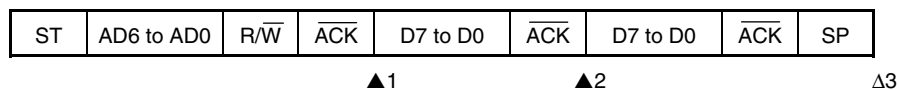
Δ 5: IICSn register = 00000001B

**Remarks 1.** ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

**2.** n = 0 to 2

**<2> When WTIMn bit = 1**

IICCn.STTn bit = 1
↓

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|-----------|------|-----|----------|-----|----------|-----|----------|-----|----|

▲1            ▲2             ▲3                    Δ4

▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000XX00B

▲3: IICSn register = 01000100B (Example: When ALDn bit is read during interrupt servicing)
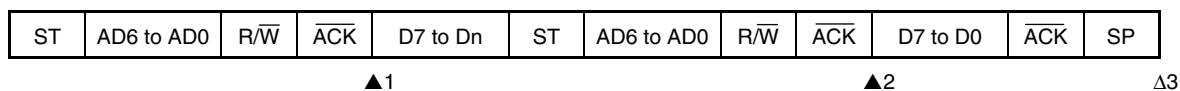
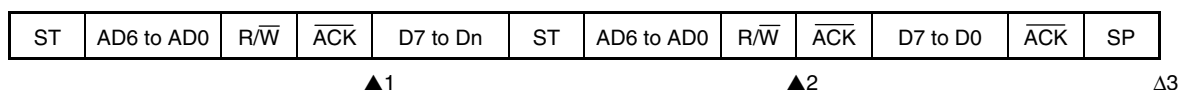Δ 4: IICSn register = 00000001B

**Remarks 1.** ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

**2.** n = 0 to 2

**(7) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition**

**<1> When WTIMn bit = 0**

STTn bit = 1
↓

| ST | AD6 to AD0 | R/W̅ | A̅C̅K̅ | D7 to D0 | A̅C̅K̅ | SP |
|----|------------|-----|------|----------|------|-----|

▲1　　　　　　▲2　　▲3　　△4

▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X000B (WTIMn bit = 1)

▲3: IICSn register = 1000XX00B

△4: IICSn register = 01000001B

**Remarks 1.** ▲: Always generated

△: Generated only when SPIEn bit = 1

X: don't care

**2.** n = 0 to 2

**<2> When WTIMn bit = 1**

STTn bit = 1
↓

| ST | AD6 to AD0 | R/W̅ | A̅C̅K̅ | D7 to D0 | A̅C̅K̅ | SP |
|----|------------|-----|------|----------|------|-----|

▲1　　　　　　　　　▲2　　△3
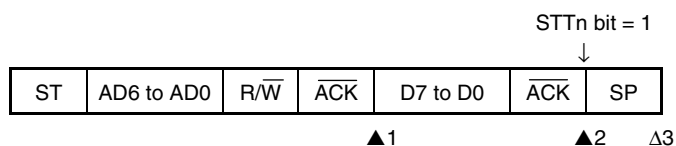
▲1: IICSn register = 1000X110B

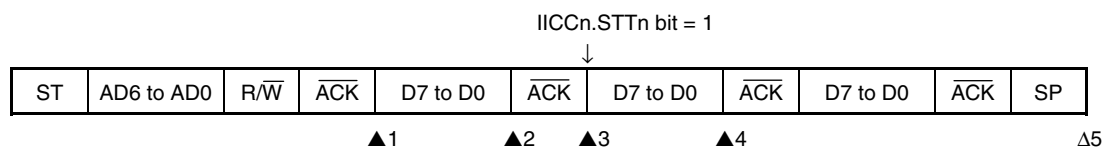▲2: IICSn register = 1000XX00B

△3: IICSn register = 01000001B

**Remarks 1.** ▲: Always generated

△: Generated only when SPIEn bit = 1

X: don't care

**2.** n = 0 to 2

**(8) When arbitration loss occurs due to low level of SDA0n pin when attempting to generate a stop condition**

**<1> When WTIMn bit = 0**

IICCn.SPTn bit = 1
↓

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|------------|------|------|----------|------|----------|------|----------|------|----|

▲1            ▲2  ▲3        ▲4                        △5

▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X000B (WTIMn bit = 1)

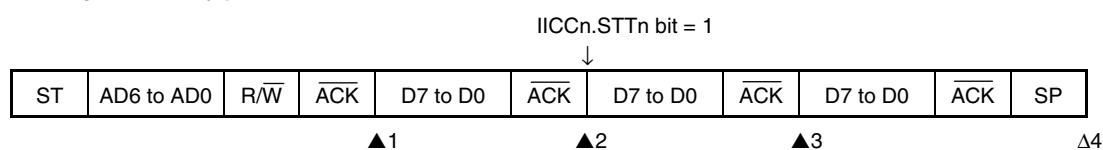▲3: IICSn register = 1000XX00B (WTIMn bit = 0)

▲4: IICSn register = 01000000B (Example: When ALDn bit is read during interrupt servicing)

△5: IICSn register = 00000001B

**Remarks 1. ▲**: Always generated

△: Generated only when SPIEn bit = 1

X: don't care

**2.** n = 0 to 2

**<2> When WTIMn bit = 1**

IICCn.SPTn bit = 1
↓

| ST | AD6 to AD0 | R/$\overline{\text{W}}$ | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | D7 to D0 | $\overline{\text{ACK}}$ | SP |
|----|------------|------|------|----------|------|----------|------|----------|------|----|

▲1                ▲2              ▲3            △4

▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000XX00B

▲3: IICSn register = 01000000B (Example: When ALDn bit is read during interrupt servicing)

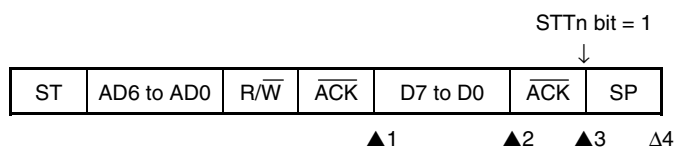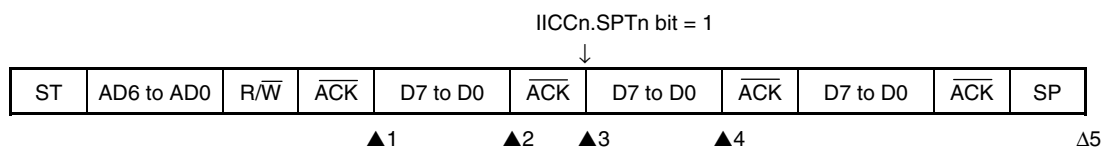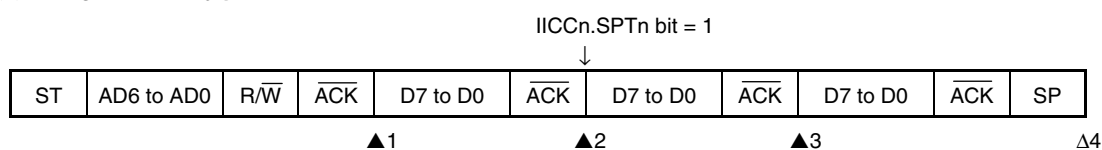△4: IICSn register = 00000001B

**Remarks 1. ▲**: Always generated

△: Generated only when SPIEn bit = 1

X: don't care

**2.** n = 0 to 2

## 19.8  Interrupt Request Signal (INTIICn) Generation Timing and Wait Control

The setting of the IICCn.WTIMn bit determines the timing by which the INTIICn register is generated and the corresponding wait control, as shown below (n = 0 to 2).

**Table 19-3.  INTIICn Generation Timing and Wait Control**

| WTIMn Bit | During Slave Device Operation | | | During Master Device Operation | | |
|---|---|---|---|---|---|---|
| | Address | Data Reception | Data Transmission | Address | Data Reception | Data Transmission |
| 0 | 9[Notes 1, 2] | 8[Note 2] | 8[Note 2] | 9 | 8 | 8 |
| 1 | 9[Notes 1, 2] | 9[Note 2] | 9[Note 2] | 9 | 9 | 9 |

**Notes 1.** The slave device's INTIICn signal and wait period occur at the falling edge of the ninth clock only when there is a match with the address set to the SVAn register.
At this point, the $\overline{\text{ACK}}$ is generated regardless of the value set to the IICCn.ACKEn bit.  For a slave device that has received an extension code, the INTIICn signal occurs at the falling edge of the eighth clock.
When the address does not match after restart, the INTIICn signal is generated at the falling edge of the ninth clock, but no wait occurs.

**2.** If the received address does not match the contents of the SVAn register and an extension code is not received, neither the INTIICn signal nor a wait occurs.

**Remarks 1.** The numbers in the table indicate the number of the serial clock's clock signals.  Interrupt requests and wait control are both synchronized with the falling edge of these clock signals.

**2.** n = 0 to 2

### (1)  During address transmission/reception
- Slave device operation:  Interrupt and wait timing are determined according to the conditions described in **Notes 1** and **2** above, regardless of the WTIMn bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIMn bit.

### (2)  During data reception
- Master/slave device operation: Interrupt and wait timing are determined according to the WTIMn bit.

### (3)  During data transmission
- Master/slave device operation: Interrupt and wait timing are determined according to the WTIMn bit.

**(4) Wait cancellation method**

The four wait cancellation methods are as follows.

- By setting the IICCn.WRELn bit to 1
- By writing to the IICn register
- By start condition setting (IICCn.STTn bit = 1)**Note**
- By stop condition setting (IICCn.SPTn bit = 1)**Note**

**Note**   Master only

When an 8-clock wait has been selected (WTIMn bit = 0), whether or not $\overline{\text{ACK}}$ has been generated must be determined prior to wait cancellation.

**Remark**   n = 0 to 2

**(5) Stop condition detection**

The INTIICn signal is generated when a stop condition is detected.

**Remark**   n = 0 to 2

## 19.9  Address Match Detection Method

In I²C bus mode, the master device can select a particular slave device by transmitting the corresponding slave address. Address match detection is performed automatically by hardware. The INTIICn signal occurs when a local address has been set to the SVAn register and when the address set to the SVAn register matches the slave address sent by the master device, or when an extension code has been received (n = 0 to 2).

## 19.10  Error Detection

In I²C bus mode, the status of the serial data bus pin (SDA0n) during data transmission is captured by the IICn register of the transmitting device, so the data of the IICn register prior to transmission can be compared with the transmitted IICn data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match (n = 0 to 2).

## 19.11  Extension Code

(1) When the higher 4 bits of the receive address are either 0000 or 1111, the extension code flag (IICSn.EXCn bit) is set for extension code reception and an interrupt request signal (INTIICn) is issued at the falling edge of the eighth clock (n = 0 to 2).
    The local address stored in the SVAn register is not affected.

(2) If 11110xx0 is set to the SVAn register by a 10-bit address transfer and 11110xx0 is transferred from the master device, the results are as follows. Note that the INTIICn signal occurs at the falling edge of the eighth clock (n = 0 to 2)

   - Higher four bits of data match: EXCn bit = 1
   - Seven bits of data match:      IICSn.COIn bit = 1

(3) Since the processing after the interrupt request signal occurs differs according to the data that follows the extension code, such processing is performed by software.
    For example, when operation as a slave is not desired after the extension code is received, set the IICCn.LRELn bit to 1 and the CPU will enter the next communication wait state.

**Table 19-4.  Extension Code Bit Definitions**

| Slave Address | R/W̄ Bit | Description |
|---|---|---|
| 0000    000 | 0 | General call address |
| 1111    0xx | 0 | 10-bit slave address specification (when the address is authorized) |
| 1111    0xx | 1 | 10-bit slave address specification (when the read command is issued after the address match) |

**Remark**   For the expansion codes other than the above, see **I²C bus specifications** issued by NXP.

## 19.12  Arbitration

When several master devices simultaneously generate a start condition (when the IICCn.STTn bit is set to 1 before the IICSn.STDn bit is set to 1), communication between the master devices is performed while the number of clocks is adjusted until the data differs.  This kind of operation is called arbitration (n = 0 to 2).

When one of the master devices loses in arbitration, an arbitration loss flag (IICSn.ALDn bit) is set to 1 via the timing at which the arbitration loss occurred, and the SCL0n and SDA0n lines are both set to high impedance, which releases the bus (n = 0 to 2).

Arbitration loss is detected based on the timing of the next interrupt request signal (INTIICn) (the eighth or ninth clock, when a stop condition is detected, etc.) and the setting of the ALDn bit to 1, which is made by software (n = 0 to 2).

For details of interrupt request timing, see **19.7  I²C Interrupt Request Signals (INTIICn)**.

**Figure 19-14.  Arbitration Timing Example**
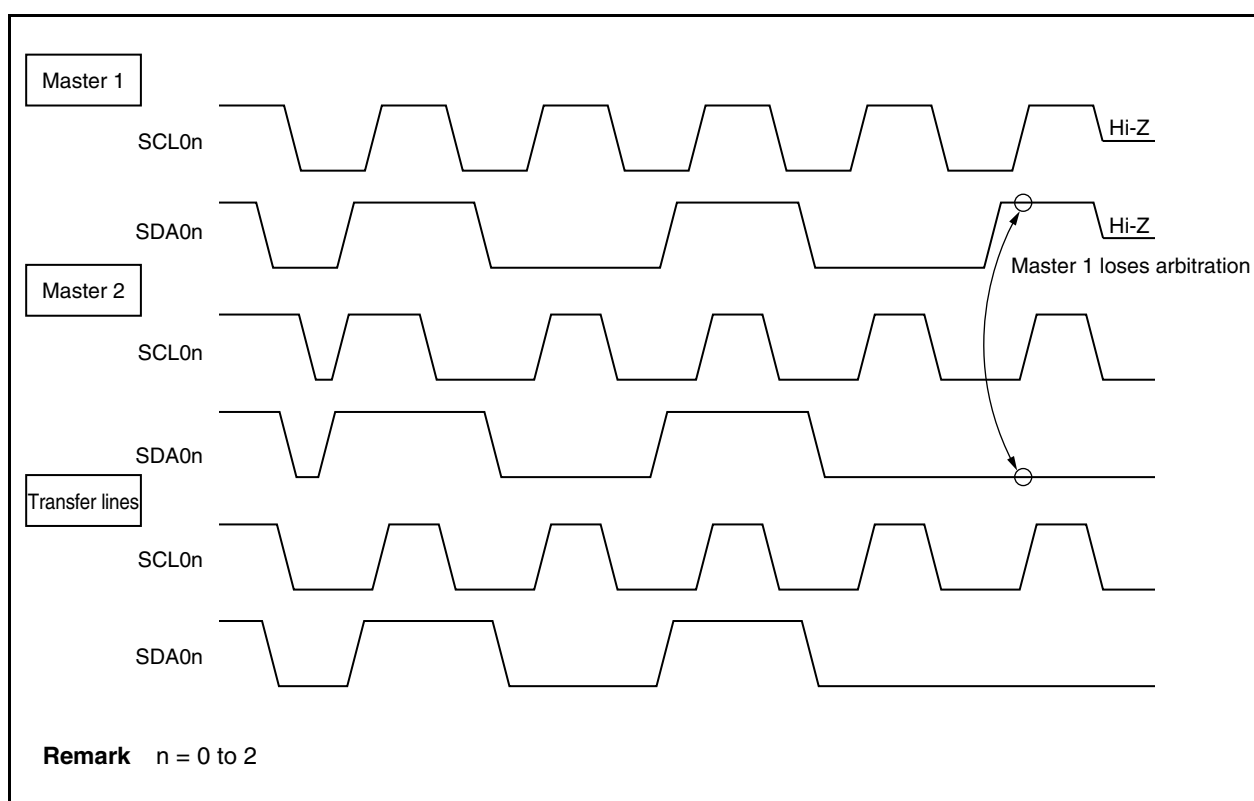
**Table 19-5.  Status During Arbitration and Interrupt Request Signal Generation Timing**

| Status During Arbitration | Interrupt Request Generation Timing |
|---|---|
| Transmitting address transmission | At falling edge of eighth or ninth clock following byte transfer[Note 1] |
| Read/write data after address transmission | |
| Transmitting extension code | |
| Read/write data after extension code transmission | |
| Transmitting data | |
| $\overline{\text{ACK}}$ transfer period after data reception | |
| When restart condition is detected during data transfer | |
| When stop condition is detected during data transfer | When stop condition is generated (when IICCn.SPIEn bit = 1)[Note 2] |
| When SDA0n pin is low level while attempting to generate restart condition | At falling edge of eighth or ninth clock following byte transfer[Note 1] |
| When stop condition is detected while attempting to generate restart condition | When stop condition is generated (when IICCn.SPIEn bit = 1)[Note 2] |
| When DSA0n pin is low level while attempting to generate stop condition | At falling edge of eighth or ninth clock following byte transfer[Note 1] |
| When SCL0n pin is low level while attempting to generate restart condition | |

**Notes 1.** When the IICCn.WTIMn bit = 1, an INTIICn signal occurs at the falling edge of the ninth clock.  When the WTIMn bit = 0 and the extension code's slave address is received, an INTIICn signal occurs at the falling edge of the eighth clock (n = 0 to 2).

**2.** When there is a possibility that arbitration will occur, set the SPIEn bit to 1 for master device operation (n = 0 to 2).

## 19.13  Wakeup Function

The makeup function is a function that generates an interrupt request signal (INTIICn) when a local address or extension code has been received by using the slave function of the I²C bus.  This function makes processing more efficient by preventing unnecessary INTIICn signals from occurring when addresses do not match.

When a start condition is detected, wakeup standby mode is set.  This wakeup standby mode is in effect while addresses are transmitted due to the possibility that an arbitration loss may change the master device (which generated the start condition) to a slave device.

However, when a stop condition is detected, the IICCn.SPIEn bit is set regardless of the wakeup function, and this determines whether INTIICn signal is enabled or disabled (n = 0 to 2).

## 19.14  Communication Reservation

### 19.14.1  When communication reservation function is enabled (IICFn.IICRSVn bit = 0)

To start master device communications when not currently using the bus, a communication reservation can be made to enable transmission of a start condition when the bus is released.  There are two modes in which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ is not returned and the bus was released when the IICCn.LRELn bit was set to 1) (n = 0 to 2).

If the IICCn.STTn bit is set to 1 while the bus is not being used, a start condition is automatically generated and a wait status is set after the bus is released (after the stop condition is detected).

When the bus release is detected (when the stop condition is detected), writing to the IICn register causes master address transfer to start.  At this point, the IICCn.SPIEn bit should be set to 1 (n = 0 to 2).

When STTn has been set to 1, the operation mode (as start condition or as communication reservation) is determined according to the bus status (n = 0 to 2).

If the bus has been released ............................................A start condition is generated

If the bus has not been released (standby mode)..............Communication reservation

To detect which operation mode has been determined, set the STTn bit to 1, wait for the wait period, then check the IICSn.MSTSn bit (n = 0 to 2).

The wait periods, which should be set via software, are listed in Table 19-6.  These wait periods can be set by the SMCn, CLn1, and CLn0 bits of the IICCLn register and the IICXn.CLXn bit (n = 0 to 2).

**Table 19-6.  Wait Periods**

| Clock Selection | CLXn | SMCn | CLn1 | CLn0 | Wait Period |
|---|---|---|---|---|---|
| f$_{xx}$/6 (OCKSm = 11H) | 0 | 0 | 0 | 0 | 156 clocks |
| f$_{xx}$/8 (OCKSm = 12H) | 0 | 0 | 0 | 0 | 208 clocks |
| f$_{xx}$/10 (OCKSm = 13H) | 0 | 0 | 0 | 0 | 260 clocks |
| f$_{xx}$/4 (OCKSm = 10H) | 0 | 0 | 0 | 1 | 188 clocks |
| f$_{xx}$/6 (OCKSm = 11H) | 0 | 0 | 0 | 1 | 282 clocks |
| f$_{xx}$/8 (OCKSm = 12H) | 0 | 0 | 0 | 1 | 376 clocks |
| f$_{xx}$/10 (OCKSm = 13H) | 0 | 0 | 0 | 1 | 470 clocks |
| f$_{xx}$/4 (OCKSm = 10H) | 0 | 0 | 1 | 1 | 148 clocks |
| f$_{xx}$/6 (OCKSm = 11H) | 0 | 0 | 1 | 1 | 222 clocks |
| f$_{xx}$/4 (OCKSm = 10H) | 0 | 1 | 0 | × | 64 clocks |
| f$_{xx}$/6 (OCKSm = 11H) | 0 | 1 | 0 | × | 96 clocks |
| f$_{xx}$/8 (OCKSm = 12H) | 0 | 1 | 0 | × | 128 clocks |
| f$_{xx}$/10 (OCKSm = 13H) | 0 | 1 | 0 | × | 160 clocks |
| f$_{xx}$/4 (OCKSm = 10H) | 0 | 1 | 1 | 1 | 52 clocks |
| f$_{xx}$/6 (OCKSm = 11H) | 0 | 1 | 1 | 1 | 78 clocks |
| f$_{xx}$/6 (OCKSm = 11H) | 1 | 1 | 0 | × | 60 clocks |
| f$_{xx}$/8 (OCKSm = 12H) | 1 | 1 | 0 | × | 80 clocks |
| f$_{xx}$/10 (OCKSm = 13H) | 1 | 1 | 0 | × | 100 clocks |

**Remarks 1.**  m = 0 and 1

n = 0 to 2

**2.**  × = Don't care

The communication reservation timing is shown below.

**Figure 19-15.  Communication Reservation Timing**
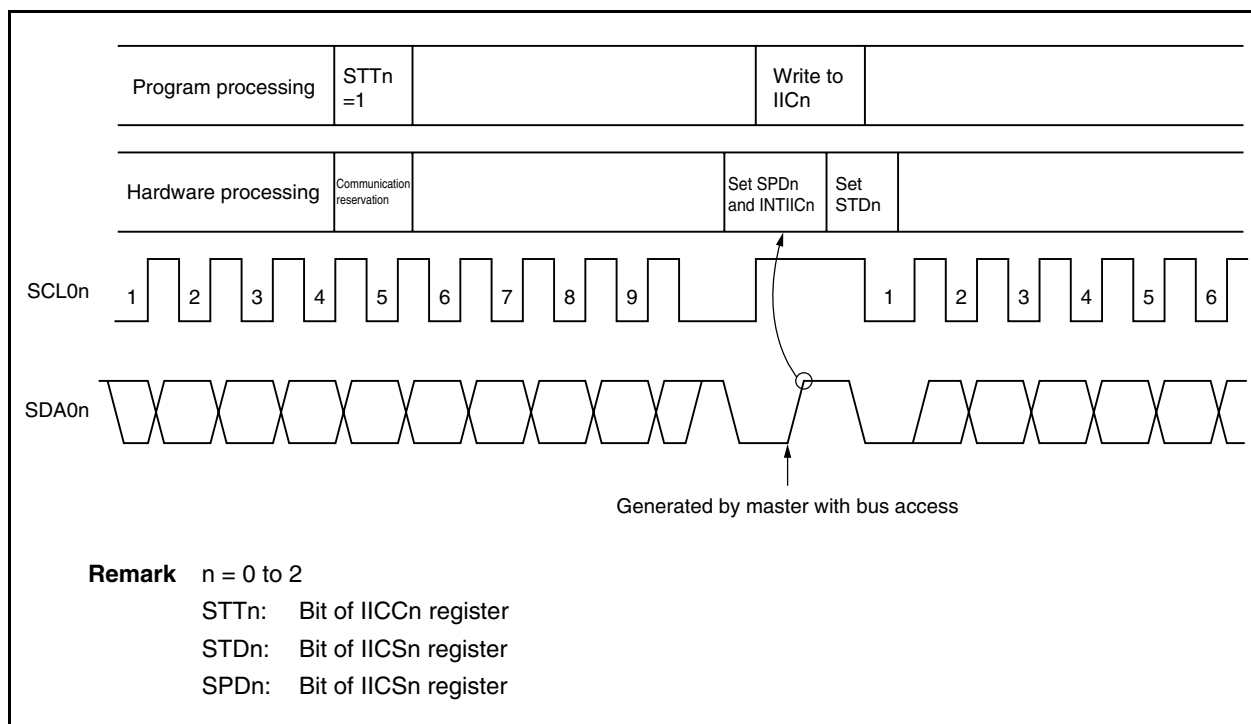


Communication reservations are accepted via the following timing.  After the IICSn.STDn bit is set to 1, a communication reservation can be made by setting the IICCn.STTn bit to 1 before a stop condition is detected (n = 0 to 2).

**Figure 19-16.  Timing for Accepting Communication Reservations**
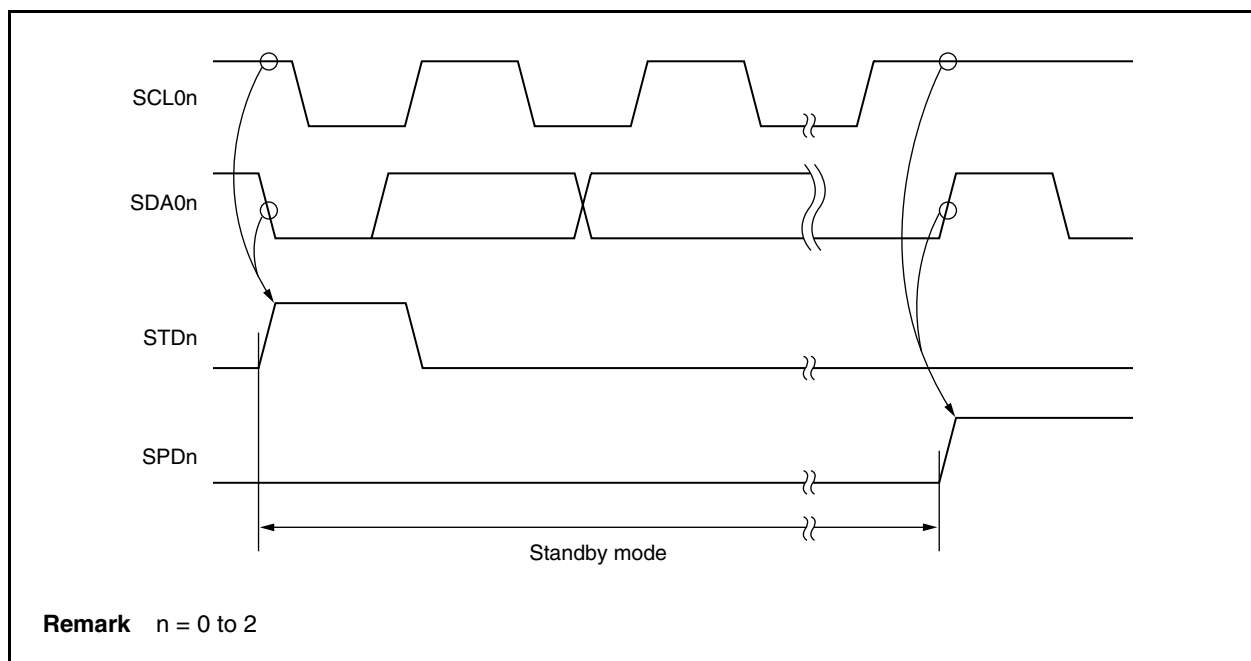
The communication reservation flowchart is illustrated below.

**Figure 19-17.  Communication Reservation Flowchart**

```
                    ┌─────────────────┐
                    │       DI        │
                    └────────┬────────┘
                             │
                    ┌─────────────────┐
                    │    SET1 STTn     │        Sets STTn bit (communication reservation).
                    └────────┬────────┘
                             │
                    ┌─────────────────┐
                    │ Define communication │    Defines that communication reservation is in effect
                    │    reservation    │        (defines and sets user flag to any RAM).
                    └────────┬────────┘
                             │
                    ┌─────────────────┐
                    │      Wait        │        Secures wait period set by software (see **Table 19-6**).
                    └────────┬────────┘
                             │
  (Communication reservation)│  Note
                    ◇─────────────────◇
                    │   MSTSn bit = 0?  │        Confirmation of communication reservation
              Yes   ◇─────────────────◇
                             │ No
                        (Generate start condition)
                    ┌─────────────────┐
                    │ Cancel communication │    Clears user flag.
                    │    reservation    │
                    └────────┬────────┘
                             │
                    ┌─────────────────┐
                    │  IICn register ← xxH │     IICn register write operation
                    └────────┬────────┘
                             │
                    ┌─────────────────┐
                    │       EI        │
                    └────────┬────────┘
```

**Note**  The communication reservation operation executes a write to the IICn register when a stop condition interrupt request occurs.

**Remark**  n = 0 to 2

**19.14.2  When communication reservation function is disabled (IICFn.IICRSVn bit = 1)**

When the IICCn.STTn bit is set when the bus is not being used for communication during bus communication, this request is rejected and a start condition is not generated. There are two modes in which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ is not returned and the bus was released when the IICCn.LRELn bit was set to 1) (n = 0 to 2)

To confirm whether the start condition was generated or the request was rejected, check the IICFn.STCFn flag. The time shown in Table 19-7 is required until the STCFn flag is set after setting the STTn bit to 1. Therefore, secure the time by software.

**Table 19-7.  Wait Periods**

| OCKSENm | OCKSm1 | OCKSm0 | CLn1 | CLn0 | Wait Period |
|---------|--------|--------|------|------|-------------|
| 1 | 0 | 0 | 0 | × | 20 clocks |
| 1 | 0 | 1 | 0 | × | 30 clocks |
| 1 | 1 | 0 | 0 | × | 40 clocks |
| 1 | 1 | 1 | 0 | × | 50 clocks |
| 0 | 0 | 0 | 1 | 0 | 10 clocks |

**Remarks  1.** ×:  don't care
**2.** n = 0 to 2
m = 0, 1

## 19.15  Cautions

(1)  When IICFn.STCENn bit = 0

Immediately after the I²C0n operation is enabled, the bus communication status (IICFn.IICBSYn bit = 1) is recognized regardless of the actual bus status.  To execute master communication in the status where a stop condition has not been detected, generate a stop condition and then release the bus before starting the master communication.

Use the following sequence for generating a stop condition.

<1> Set the IICCLn register.
<2> Set the IICCn.IICEn bit.
<3> Set the IICCn.SPTn bit.

(2)  When IICFn.STCENn bit = 1

Immediately after I²C0n operation is enabled, the bus release status (IICBSYn bit = 0) is recognized regardless of the actual bus status.  To generate the first start condition (IICCn.STTn bit = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

(3)  When the IICCn.IICEn bit of the V850ES/JG3-U and V850ES/JH3-U is set to 1 while other devices are communicating, the start condition may be detected depending on the status of the communication line.  Be sure to set the IICCn.IICEn bit to 1 when the SCL0n and SDA0n lines are high level.

(4)  Determine the operation clock frequency by using the IICCLn, IICXn, and OCKSm registers before enabling the operation (IICCn.IICEn bit = 1).  To change the operation clock frequency, first clear the IICCn.IICEn bit to 0.

(5)  After the IICCn.STTn and IICCn.SPTn bits have been set to 1, they must not be re-set without being cleared to 0 first.

(6)  If transmission has been reserved, set the IICCN.SPIEn bit to 1 so that an interrupt request is generated by the detection of a stop condition.  After an interrupt request has been generated, the wait status will be released by writing communication data to I²Cn, then transfer will begin.  If an interrupt is not generated by the detection of a stop condition, transmission will halt in the wait status because an interrupt request was not generated.  However, it is not necessary to set the SPIEn bit to 1 for the software to detect the IICSn.MSTSn bit.

**Remark**   n = 0 to 2
m = 0, 1

## 19.16  Communication Operations

The following shows three operation procedures together with flowcharts.

**(1)  Master operation in single master system**

The flowchart when using the V850ES/JG3-U and V850ES/JH3-U as the master in a single master system is shown below.

This flowchart is broadly divided into initial settings and communication processing.  Execute the initial settings at startup.  If communication with a slave is required, prepare the communication and then execute communication processing.

**(2)  Master operation in multimaster system**

In the I$^2$C0n bus multimaster system, whether the bus is released or used cannot be judged by the I$^2$C bus specifications when the bus takes part in a communication.  Here, when data and the clock are at a high level for a certain period (1 frame), the V850ES/JG3-U or V850ES/JH3-U takes part in communication in a bus release state.

This flowchart is broadly divided into initial settings, communication waiting, and communication processing.  The processing when the V850ES/JG3-U or V850ES/JH3-U loses in arbitration and is specified as the slave is omitted here, and only the processing as the master is shown.  Execute the initial settings at startup to take part in communication.  Then, wait for the communication request as the master or wait for the specification as the slave. The actual communication is performed in the communication processing, and includes arbitration with other masters data as well as transmission/reception with the slave.

**(3)  Slave operation**

An example of when the V850ES/JG3-U or V850ES/JH3-U is used as the slave of the I$^2$C0n bus is shown below.

When used as the slave, operation is started by an interrupt.  Execute the initial settings at startup, then wait for INTIICn interrupt occurrence (communication waiting).  When the INTIICn interrupt occurs, the communication status is judged and its result is passed as a flag to the main processing.

By checking the flags, the necessary communication processing can be performed.

**Remark**    n = 0 to 2

### 19.16.1  Master operation in single master system

**Figure 19-18.  Master Operation in Single Master System**



**Note** Release the I²C0n bus (SCL0n, SDA0n pins = high level) in compliance with the specifications of the communicating product.  For example, when the EEPROM™ outputs a low level to the SDA0n pin, set the SCL0n pin to the output port and output clock pulses from that output port until the SDA0n pin becomes constantly high level.

**Remarks 1.** For the transmission and reception formats, comply with the specifications of the communicating product.

**2.** n = 0 to 2, m = 0, 1

**19.16.2  Master operation in multimaster system**

**Figure 19-19.  Master Operation in Multimaster System (1/3)**



**Note** Confirm that the bus release status (IICCLn.CLDn bit = 1, IICCLn.DADn bit = 1) has been maintained for a certain period (1 frame, for example).  When the SDA0n pin is constantly low level, determine whether to release the I²C0n bus (SCL0n, SDA0n pins = high level) by referring to the specifications of the communicating product.

**Remark**  n = 0 to 2, m = 0, 1
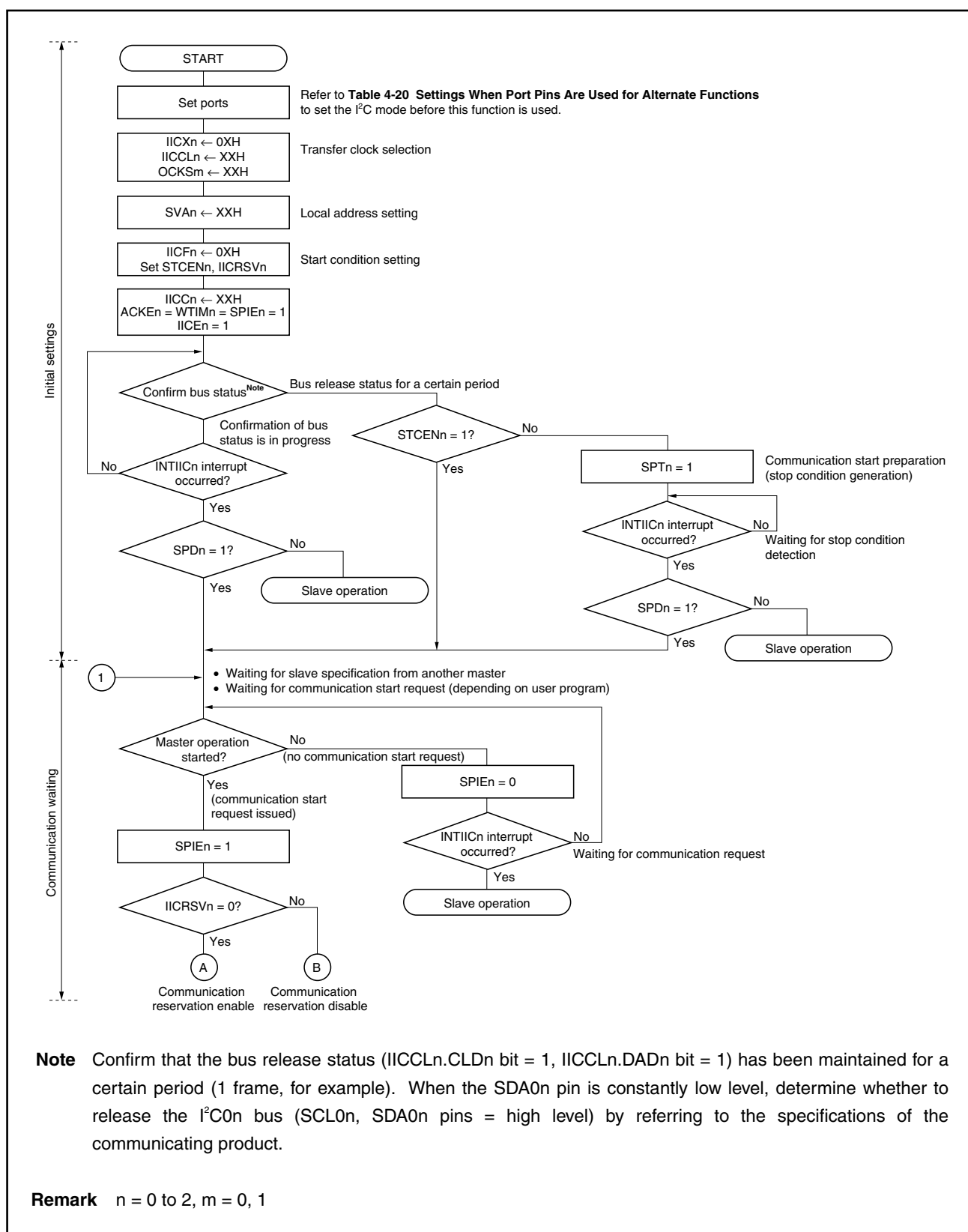
**Figure 19-19.  Master Operation in Multimaster System (2/3)**



**Remark**  n = 0 to 2
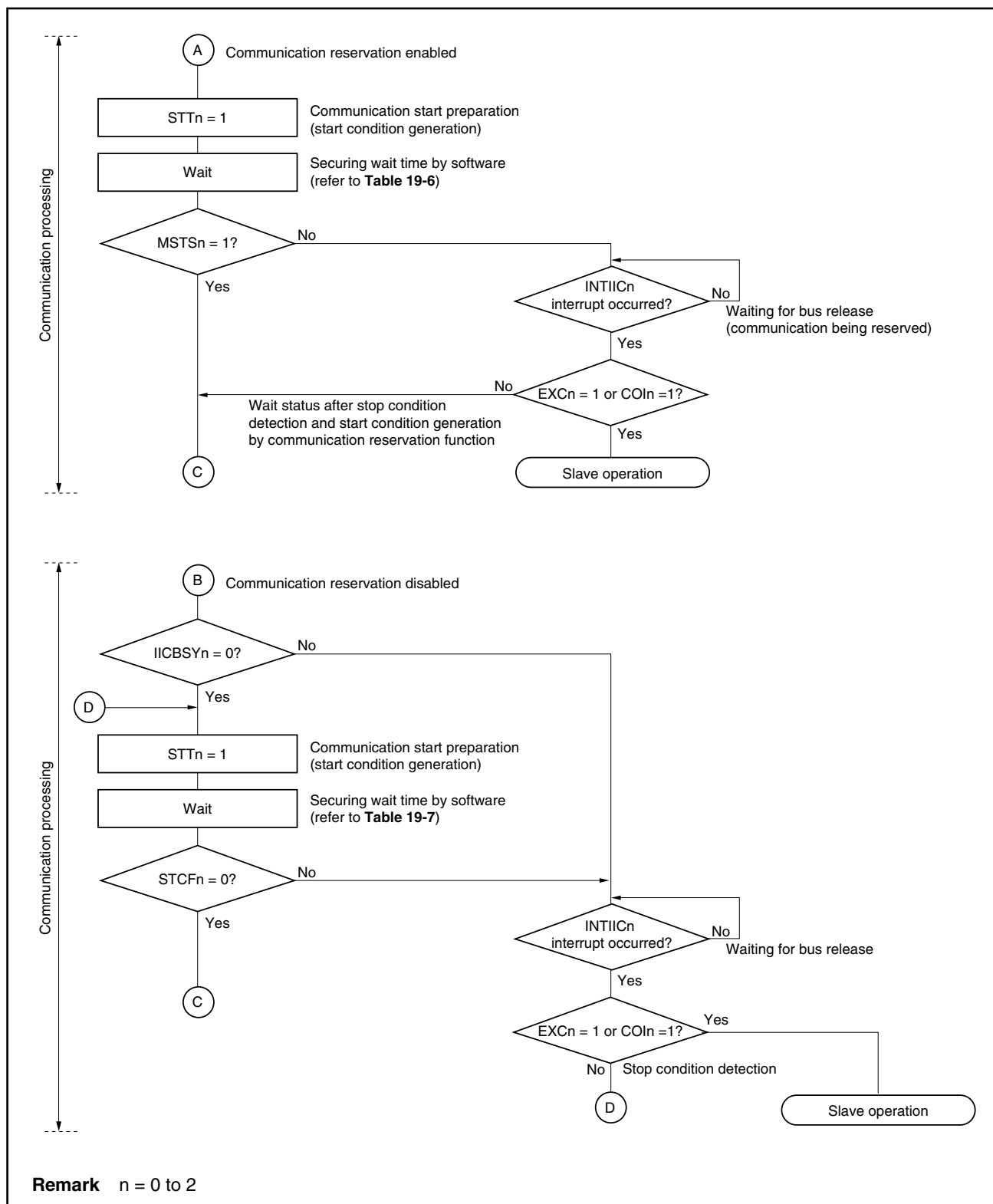
**Figure 19-19.  Master Operation in Multimaster System (3/3)**
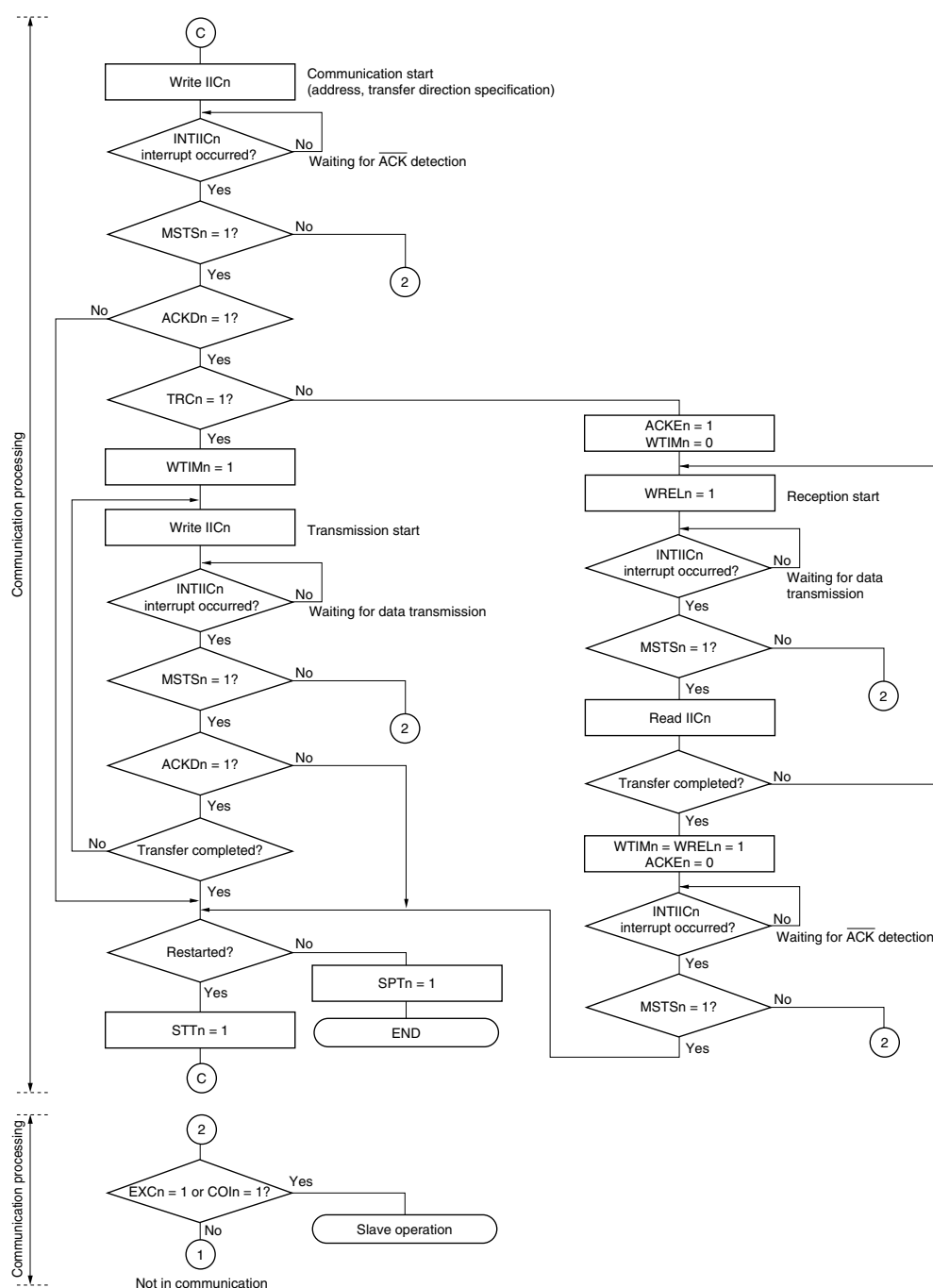


**Remarks 1.** For the transmission and reception formats, comply with the specifications of the communicating product.

**2.** When using the V850ES/JG3-U or V850ES/JH3-U as the master in a multimaster system, read the IICSn.MSTSn bit for each INTIICn interrupt occurrence to confirm the arbitration result.

**3.** When using the V850ES/JG3-U or V850ES/JH3-U as the slave in a multimaster system, confirm the status using the IICSn and IICFn registers for each INTIICn interrupt occurrence to determine the next processing.

**4.** n = 0 to 2

### 19.16.3  Slave operation

The following shows the processing procedure of slave operation.

Basically, the operation of the slave device is event-driven.  Therefore, processing by an INTIICn interrupt (processing requiring a significant change of the operation status, such as stop condition detection during communication) is necessary.

The following description assumes that data communication does not support extension codes.  Also, it is assumed that the INTIICn interrupt servicing performs only status change processing and that the actual data communication is performed during the main processing.

**Figure 19-20.  Overview of Software During Slave Operation**



Therefore, the following three flags are prepared so that the data transfer processing can be performed by passing these flags to the main processing instead of the INTIICn signal.

**(1)  Communication mode flag**

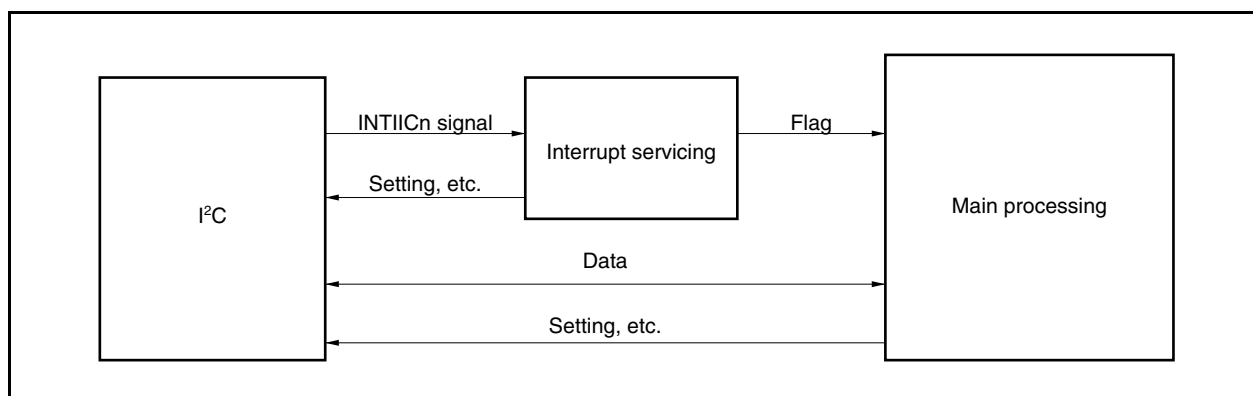This flag indicates the following communication statuses.

Clear mode:              Data communication not in progress

Communication mode:  Data communication in progress (valid address detection stop condition detection, $\overline{ACK}$ from

master not detected, address mismatch)

**(2)  Ready flag**

This flag indicates that data communication is enabled.  This is the same status as an INTIICn interrupt during normal data transfer.  This flag is set in the interrupt processing block and cleared in the main processing block.  The ready flag for the first data for transmission is not set in the interrupt processing block, so the first data is transmitted without clear processing (the address match is regarded as a request for the next data).

**(3)  Communication direction flag**

This flag indicates the direction of communication and is the same as the value of IICSn.TRCn bit.

The following shows the operation of the main processing block during slave operation.

I²C0n starts and waits for the communication enable status.  When communication is enabled, I²C0n performs transfer using the communication mode flag and ready flag (the processing of the stop condition and start condition is performed by interrupts, conditions are confirmed by flags).

For transmission, the transmission operation is repeated until the master device stops returning $\overline{ACK}$.  When the master device stops returning $\overline{ACK}$, transfer is complete.

For reception, the required number of data is received and $\overline{\text{ACK}}$ is not returned for the next data immediately after transfer is complete. After that, the master device generates the stop condition or restart condition. This causes exit from communications.

**Figure 19-21. Slave Operation Flowchart (1)**



**Remark**   n = 0 to 2, m = 0, 1
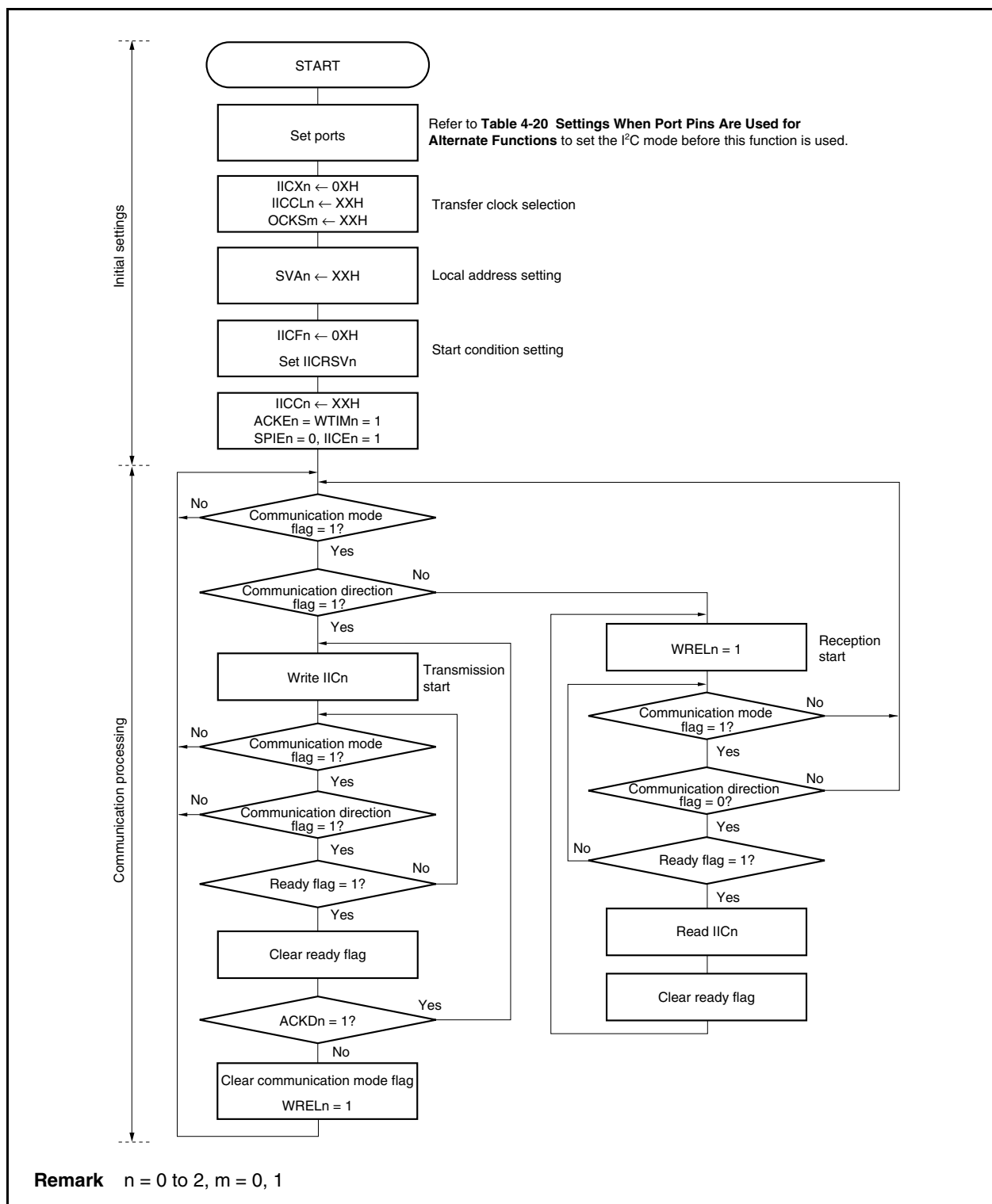
The following shows an example of the processing of the slave device by an INTIICn interrupt (it is assumed that no extension codes are used here).  During INTIICn interrupt servicing, the status is confirmed and the following steps are executed.

<1> When a stop condition is detected, communication is terminated.

<2> When a start condition is detected, the address is confirmed.  If the address does not match, communication is terminated.  If the address matches, the communication mode is set, the wait state is released, and operation returns from the interrupt (the ready flag is cleared).

<3> For data transmission/reception, when the ready flag is set, operation returns from the interrupt while the I²C0n bus remains in the wait status.

**Remark**    <1> to <3> above correspond to <1> to <3> in **Figure 19-22  Slave Operation Flowchart (2)**.

**Figure 19-22.  Slave Operation Flowchart (2)**
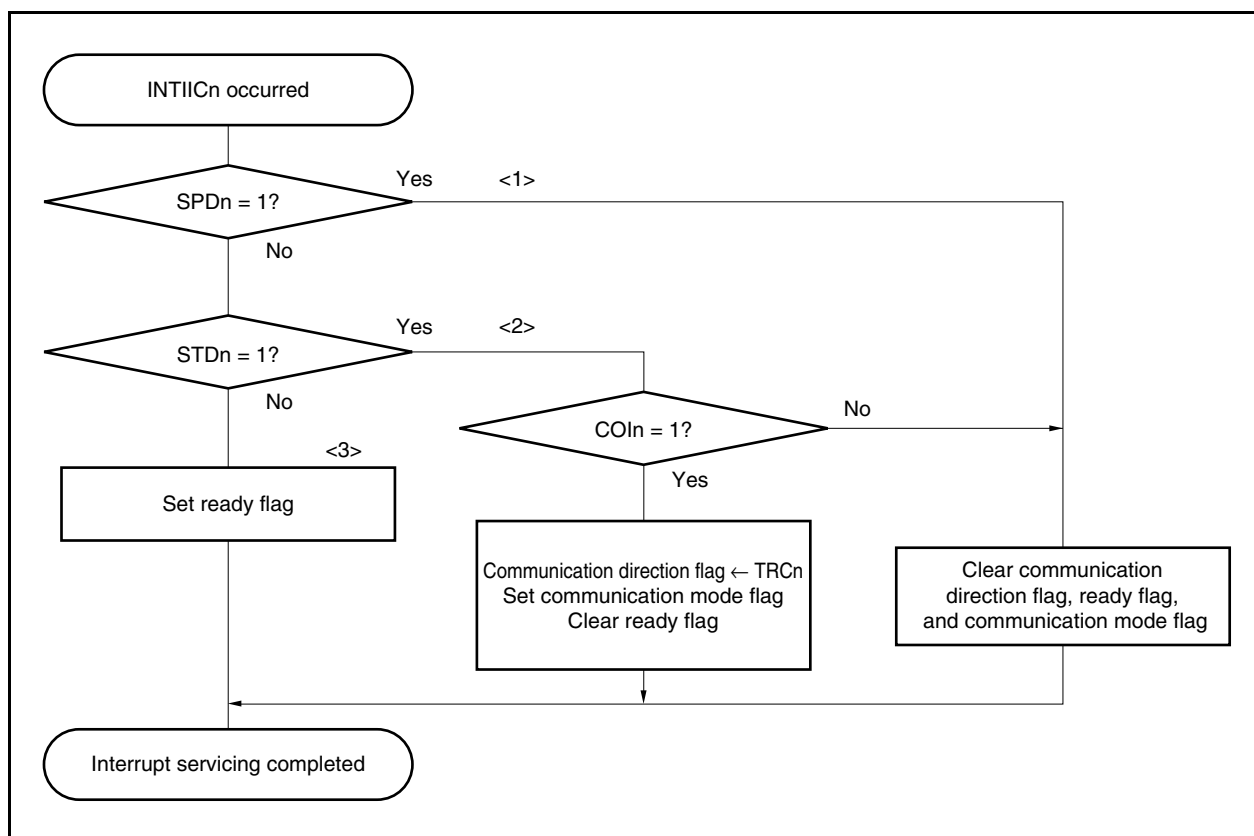
## 19.17  Timing of Data Communication

When using I$^2$C bus mode, the master device outputs an address via the serial bus to select one of several slave devices as its communication partner.

After outputting the slave address, the master device transmits the IICSn.TRCn bit, which specifies the data transfer direction, and then starts serial communication with the slave device.

The shift operation of the IICn register is synchronized with the falling edge of the serial clock pin (SCL0n).  The transmit data is transferred to the SO latch and is output (MSB first) via the SDA0n pin.

Data input via the SDA0n pin is captured by the IICn register at the rising edge of the SCL0n pin.

The data communication timing is shown below.

**Remark**   n = 0 to 2

**Figure 19-23. Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)**



**(a) Start condition ~ address**

**Notes 1.** Cancel the wait during a master transmission by writing data to IICn, not by setting WRELn.

**2.** To cancel the slave wait state, write FFH to IICn or set WRELn.

**Remark** n = 0 to 2

**Figure 19-23.  Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)**



**(b) Data**

**Notes 1.** Cancel the wait during a master transmission by writing data to IICn, not by setting WRELn.

**2.** To cancel the slave wait state, write FFH to IICn or set WRELn.

**Remark**  n = 0 to 2

**Figure 19-23.  Example of Master to Slave Communication**
**(When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)**



**(c) Stop condition**

**Notes 1.** Cancel the wait during a master transmission by writing data to IICn, not by setting WRELn.

   **2.** To cancel the slave wait state, write FFH to IICn or set WRELn.

**Remark**   n = 0 to 2

**Figure 19-24.  Example of Slave to Master Communication**
**(When 8-Clock Wait for Master and 9-Clock Wait for Slave Are Selected) (1/3)**



**(a)  Start condition ~ address**

Notes 1. To cancel the master wait state, write FFH to IICn or set WRELn.

   2. Cancel the wait during a slave transmission by writing to IICn, not by setting WRELn.

**Remark**   n = 0 to 2

**Figure 19-24.  Example of Slave to Master Communication
(When 8-Clock Wait for Master and 9-Clock Wait for Slave Are Selected) (2/3)**



**(b) Data**

Notes 1.   To cancel the master wait state, write FFH to IICn or set WRELn.

2.   Cancel the wait during a slave transmission by writing to IICn, not by setting WRELn.

Remark   n = 0 to 2

**Figure 19-24.  Example of Slave to Master Communication
(When 8-Clock Wait for Master and 9-Clock Wait for Slave Are Selected) (3/3)**



**(c) Stop condition**

**Notes 1.**  To cancel the wait state, write FFH to IICn or set WRELn.

   **2.**  Cancel the wait during a slave transmission by writing to IICn, not by setting WRELn.

   **3.**  When the wait during a slave transmission is canceled by setting WRELn, TRCn is cleared.

**Remark**  n = 0 to 2

# CHAPTER 20 USB FUNCTION CONTROLLER (USBF)

The V850ES/JG3-U and V850ES/JH3-U have an internal USB function controller (USBF) conforming to the Universal Serial Bus Specification. Data communication using the polling method is performed between the USB function controller and external host device by using a token-based protocol.

## 20.1 Overview

- Conforms to the Universal Serial Bus Specification
- Supports 12 Mbps (full-speed) transfer
- Endpoint for transfer incorporated

| Endpoint Name | FIFO Size (Bytes) | Transfer Type | Remark |
|---|---|---|---|
| Endpoint0 Read | 64 | Control transfer | – |
| Endpoint0 Write | 64 | Control transfer | – |
| Endpoint1 | 64 × 2 | Bulk 1 transfer (IN) | 2-buffer configuration |
| Endpoint2 | 64 × 2 | Bulk 1 transfer (OUT) | 2-buffer configuration |
| Endpoint3 | 64 × 2 | Bulk 2 transfer (IN) | 2-buffer configuration |
| Endpoint4 | 64 × 2 | Bulk 2 transfer (OUT) | 2-buffer configuration |
| Endpoint7 | 8 | Interrupt transfer | – |

- Bulk transfer (IN/OUT) can be executed as DMA transfer (2-cycle single-transfer mode)
- Clock: Internal clock (6 MHz external clock × internal clock multiplied by 8 = 48 MHz internal clock) or external clock (external clock input to UCLK pin ($f_{USB}$ = 48 MHz)) selectable

**Caution** **The registers listed in 20.6.2 USB function controller register list must be accessed after specifying that the internal clock or the external clock is to be used as the USB clock and enabling clock supply to the USB function controller.**

## 20.2  Configuration

### 20.2.1  Block diagram

**Figure 20-1.  Block Diagram of USB Function Controller**



**Remarks 1.**  Inside broken lines:  These functions are included in the USB function controller.
     **2.**  n = 0, 1
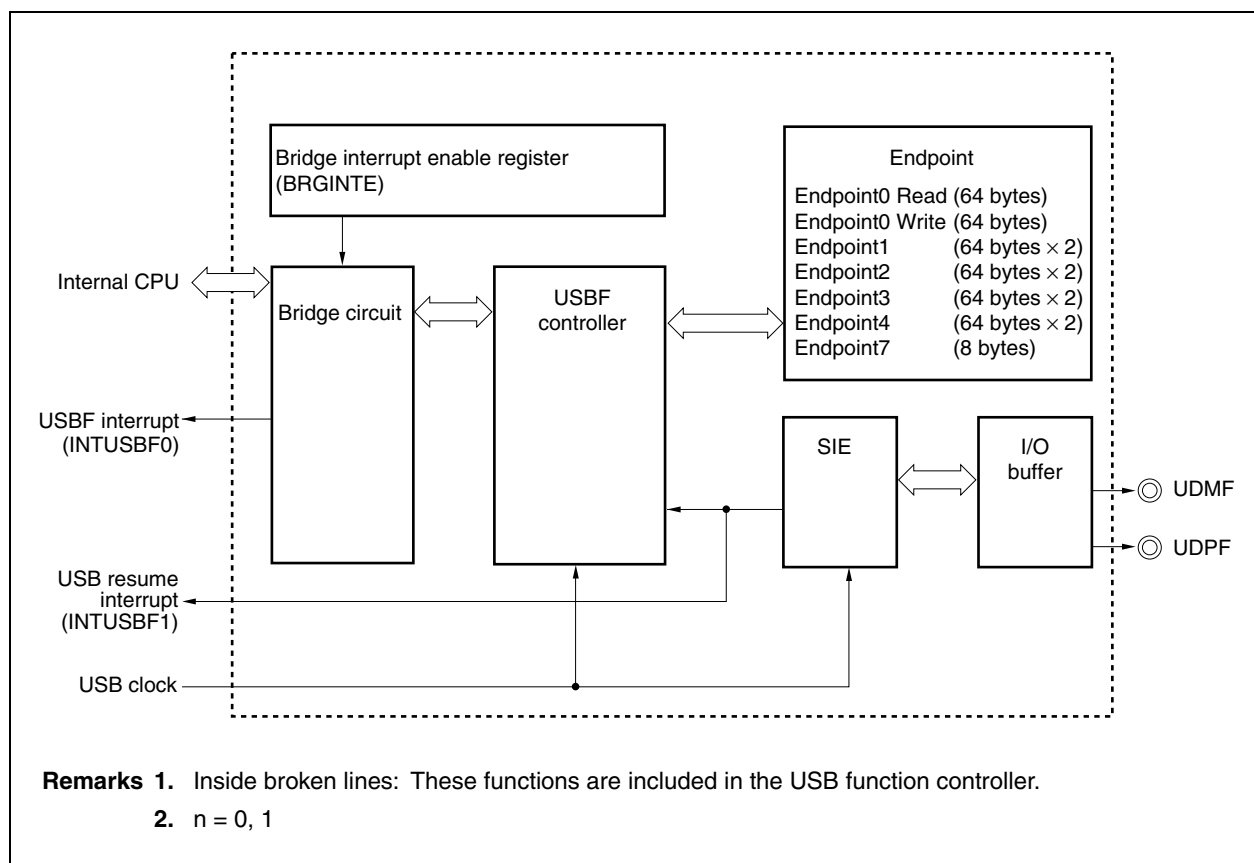
**20.2.2  USB memory map**

The USB function controller seen from the CPU is assigned to the CS1 space in the microcontroller.  The memory space is divided for use as follows.

**Table 20-1.  Division of CPU Memory Space**

| Address | Area | |
|---|---|---|
| 00200000H to 00200092H | EPC control register area | |
| 00200100H to 00200114H | EPC data hold register area | |
| 00200144H to 002003C4H | EPC request data register area | |
| 00200400H to 00200408H | Bridge register area | |
| 00200500H to 0020050EH | DMA register area | |
| 00201000H | Bulk-in register area | EP1 (Bulk-IN1) |
| 00202000H | | EP3 (Bulk-IN2) |
| 00210000H | Bulk-out register area | EP2 (Bulk-Out1) |
| 00220000H | | EP4 (Bulk-Out2) |
| 00240000H | Peripheral control register area | |

## 20.3  External Circuit Configuration

### 20.3.1  Outline

In USB transmission, when communication is performed with the host controller and function controller facing each other, pull-up/pull-down resistors must be connected to the USB signal (D+/D−) to identify the communication partner. Moreover in the V850ES/JG3-U and V850ES/JH3-U, series resistors must also be connected.

Because the V850ES/JG3-U and V850ES/JH3-U do not include these pull-up/pull-down resistors and series resistors, be sure to connect them externally.

The following shows the outline configuration of the USB transmission line.  For details of the external configuration, see the description provided in each section.

**Figure 20-2.  Outline Configuration of Pull-up, Pull-down, Series Resistors in USB Transmission Line**
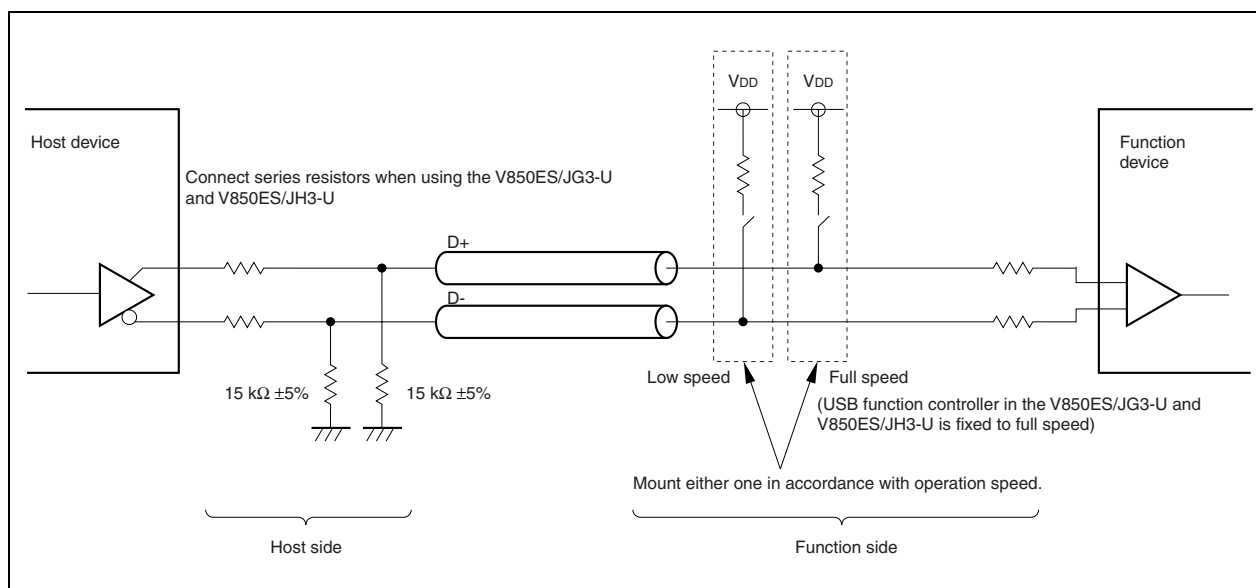
### 20.3.2 Connection configuration

**Figure 20-3. Example of USB Function Controller Connection**
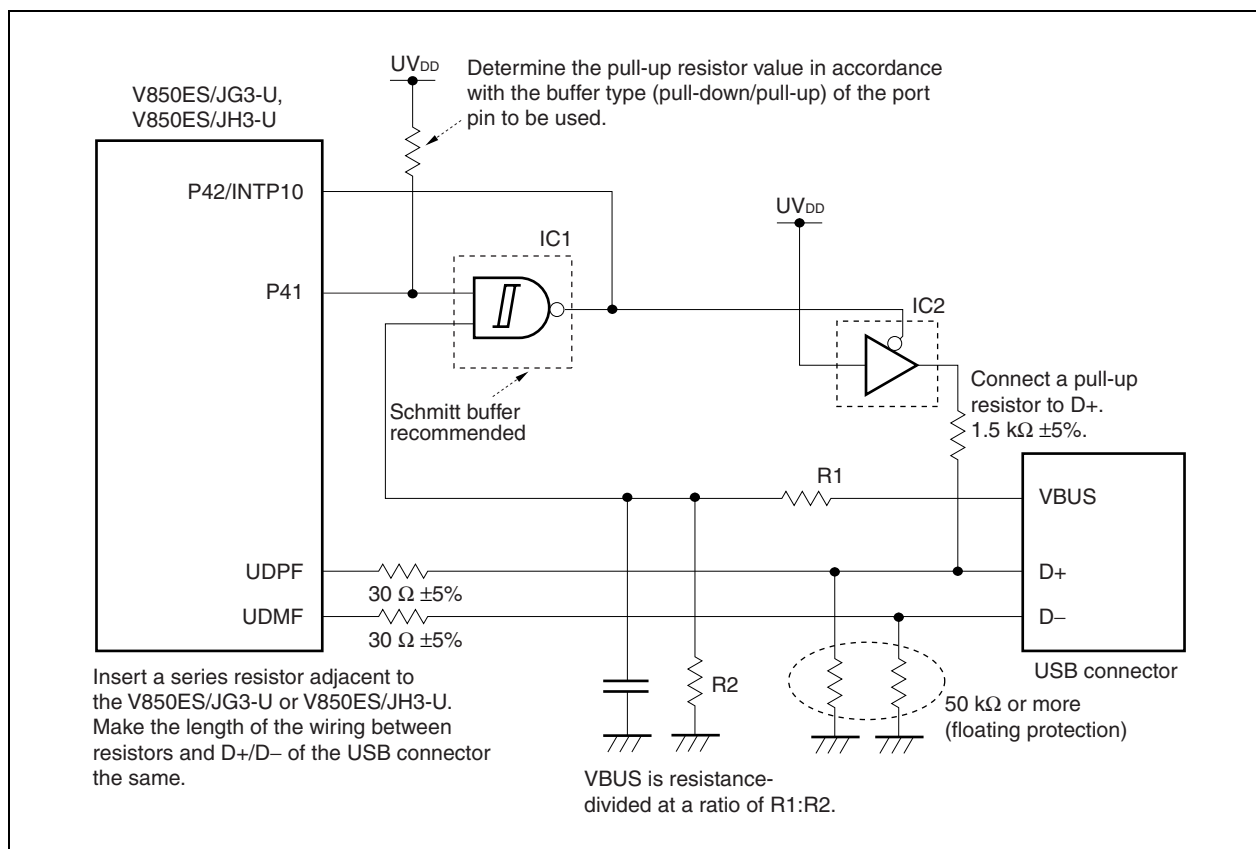


**(1) Series resistor connection to D+/D–**

Connect series resistors of 30 Ω ±5% to the D+/D– pins (UFDP, UFDM) of the USB function controller in the V850ES/JG3-U and V850ES/JH3-U. If they are not connected, the impedance rating cannot be satisfied and the output waveform may be disturbed.

Allocate the series resistors adjacent to the V850ES/JG3-U or V850ES/JH3-U, and make the length of the wiring between the series resistors and the USB connectors the same, to make the impedance of D+ and D– equal (a differential with 90 Ω ±5% is recommended).

**(2) Pull-up control of D+**

Because the function controller of the V850ES/JG3-U and V850ES/JH3-U is fixed to full speed (FS), be sure to pull up the D+ pin (UFDP) by 1.5 kΩ ±5% to UV$_{DD}$.

To disable a connection report (D+ pull up) to the USB host/HUB (such as during high priority servicing or initialization), control the pull-up resistor of D+ via a general-purpose port in the system. For a circuit such as the one shown in Figure 20-3, control the pull-up control signal and the VBUS input signal of the D+ pin by using a general-purpose port and the USB cable VBUS (AND circuit). In Figure 20-3, if the general-purpose port is low level, pulling up of D+ is prohibited.

For the IC2 in Figure 20-3, use an IC to which voltage can be applied when the system power is off.

**(3) Detection of USB cable connection/disconnection**

The USB function controller (USBF) requires a VBUS input signal to recognize whether the USB cable is connected or disconnected, because the state of the USBF is controlled by hardware. The voltage from the USB host or HUB (5 V) is applied as the VBUS input signal when the USB cable VBUS is connected to the USB host or HUB while the USBF power is off. Therefore, for IC1 in Figure 20-3, use an IC to which voltage can be applied when the system power is off. When disconnecting the USB cable in the circuit in Figure 20-3, the input signal to INTP10 may be unstable while the VBUS voltage is dropping. It is therefore recommended to use a Schmitt buffer for IC1 in Figure 20-3.

**(4) Floating protection during initialization or when USBF is unused**

When the USB function controller is initialized or unused, to avoid a floating status, pull the D+/D− pins down using a resistor of 50 kΩ or higher.

## 20.4　Cautions

**(1)　Clock accuracy**

To operate the USB function controller, the internal clock (6 MHz external clock × internal clock multiplied by 8 = 48 MHz internal clock) or external clock (external clock input to UCLK pin ($f_{USB}$ = 48 MHz)) must be used as the USB clock.　When the internal clock is used as the USB clock, use a resonator with an accuracy of 6 MHz ±500 ppm (max.).　When the external clock is used, apply a clock with an accuracy of 48 MHz ±500 ppm (max.) to the UCLK pin.　If the USB clock accuracy drops, the transmission data cannot satisfy the USB rating.

**(2)　Stopping the USB clock**

When the main clock ($f_{XX}$) has been selected as the USB function controller clock, be sure to stop the USB function controller (set bits 1 and 0 of the UFCKMSK register to 1 after the USB function has been stopped by software) before stopping the main clock ($f_{XX}$).　Otherwise, a malfunction might occur due to noise in the main clock pulse when the USB function controller is restarted.

Similarly, when an external clock input from the EXCLK pin has been selected as the USB function controller clock, take measures to prevent noise from being generated in the clock pulse or be sure to stop the USB function controller before stopping the main clock ($f_{XX}$).

## 20.5  Requests

The USB standard has a request command that reports requests from the host device to the function device to execute response processing.

The requests are received in the SETUP stage of control transfer, and most can be automatically processed via the hardware of the USB function controller (USBF).

### 20.5.1  Automatic requests

**(1) Decode**

The following tables show the request format and the correspondence between requests and decoded values.

**Table 20-2.  Request Format**

| Offset | Field Name | |
|---|---|---|
| 0 | bmRequestType | |
| 1 | bRequest | |
| 2 | wValue | Lower side |
| 3 | | Higher side |
| 4 | wIndex | Lower side |
| 5 | | Higher side |
| 6 | wLength | Lower side |
| 7 | | Higher side |

**Table 20-3. Correspondence Between Requests and Decoded Values**

| Request \ Offset | bmRequestType | bRequest | wValue | | wIndex | | wLength | | Df | Ad | Cf | Data Stage |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 3 | 2 | 5 | 4 | 7 | 6 | | | | |
| GET_INTERFACE | 81H | 0AH | 00H | 00H | 00H | 0nH | 00H | 01H | STALL | STALL | ACK NAK | √ |
| GET_CONFIGURATION | 80H | 08H | 00H | 00H | 00H | 00H | 00H | 01H | ACK NAK | ACK NAK | ACK NAK | √ |
| GET_DESCRIPTOR Device | 80H | 06H | 01H | 00H | 00H | 00H | XXH | XXH[Note 1] | ACK NAK | ACK NAK | ACK NAK | √ |
| GET_DESCRIPTOR Configuration | 80H | 06H | 02H | 00H | 00H | 00H | XXH | XXH[Note 1] | ACK NAK | ACK NAK | ACK NAK | √ |
| GET_STATUS Device | 80H | 00H | 00H | 00H | 00H | 00H | 00H | 02H | ACK NAK | ACK NAK | ACK NAK | √ |
| GET_STATUS Endpoint 0 | 82H | 00H | 00H | 00H | 00H | 00H 80H | 00H | 02H | ACK NAK | ACK NAK | ACK NAK | √ |
| GET_STATUS Endpoint X | 82H | 00H | 00H | 00H | 00H | $$H | 00H | 02H | STALL | STALL | ACK NAK | √ |
| CLEAR_FEATURE Device[Note 2] | 00H | 01H | 00H | 01H | 00H | 00H | 00H | 00H | ACK NAK | ACK NAK | ACK NAK | × |
| CLEAR_FEATURE Endpoint 0[Note 2] | 02H | 01H | 00H | 00H | 00H | 00H 80H | 00H | 00H | ACK NAK | ACK NAK | ACK NAK | × |
| CLEAR_FEATURE Endpoint X[Note 2] | 02H | 01H | 00H | 00H | 00H | $$H | 00H | 00H | STALL | STALL | ACK NAK | × |
| SET_FEATURE Device[Note 3] | 00H | 03H | 00H | 01H | 00H | 00H | 00H | 00H | ACK NAK | ACK NAK | ACK NAK | × |
| SET_FEATURE Endpoint 0[Note 3] | 02H | 03H | 00H | 00H | 00H | 00H 80H | 00H | 00H | ACK NAK | ACK NAK | ACK NAK | × |
| SET_FEATURE Endpoint X[Note 3] | 02H | 03H | 00H | 00H | 00H | $$H | 00H | 00H | STALL | STALL | ACK NAK | × |
| SET_INTERFACE | 01H | 0BH | 00H | 0#H | 00H | 0?H | 00H | 00H | STALL | STALL | ACK NAK | × |
| SET_CONFIGURATION[Note 4] | 00H | 09H | 00H | 00H 01H | 00H | 00H | 00H | 00H | ACK NAK | ACK NAK | ACK NAK | × |
| SET_ADDRESS | 00H | 05H | XXH | XXH | 00H | 00H | 00H | 00H | ACK NAK | ACK NAK | ACK NAK | × |

**Remark**    √: Data stage

         ×: No data stage

**Notes 1.** If the wLength value is lower than the prepared value, the wLength value is returned; if the wLength value is the prepared value or higher, the prepared value is returned.

    **2.** The CLEAR_FEATURE request clears UF0 device status register L (UF0DSTL) and UF0 EPn status register L (UF0EnSL) (n = 0 to 4, 7) when ACK is received in the status stage.

**Notes 3.** The SET_FEATURE request sets the UF0 device status register L (UF0DSTL) and UF0 EPn status register L (UF0EnSL) (n = 0 to 4, 7) when ACK is received in the status stage.  If the E0HALT bit of the UF0E0SL register is set, a STALL response is made in the status stage or data stage of control transfer for a request other than the GET_STATUS Endpoint0 request, SET_FEATURE Endpoint0 request, and a request generated by the CPUDEC interrupt request, until the CLEAR_FEATURE Endpoint0 request is received.  A STALL response to an unsupported request does not set the E0HALT bit of the UF0E0SL register to 1, and the STALL response is cleared as soon as the next SETUP token has been received.

**4.** If the wValue is not the default value, an automatic STALL response is made.

**Cautions 1.** **The sequence of control transfer defined by the Universal Serial Bus Specification is not satisfied under the following conditions.  The operation is not guaranteed under these conditions.**

- **If an IN/OUT token is suddenly received without a SETUP stage**
- **If DATA PID1 is sent in the data phase of the SETUP stage**
- **If a token of 128 addresses or more is received**
- **If the request data transmitted in the SETUP stage is of less than 8 bytes**

**2.** **An ACK response is made even when the host transmits data other than a Null packet in the status stage.**

**3.** **If the wLength value is 00H during control transfer (read) of FW processing, a Null packet is automatically transmitted for control transfer (without data).  The FW request does not automatically transmit a Null packet.**

**Remarks 1.** Df: Default state, Ad: Addressed state, Cf: Configured state

**2.** n = 0 to 4
It is determined by the setting of the UF0 active interface number register (UF0AIFN) whether a request with Interface number 1 to 4 is correctly responded to, depending on whether the Interface number of the target is valid or not.

**3.** $$: Valid endpoint number including transfer direction
The valid endpoint is determined by the currently set Alternate Setting number (see **20.6.3 (36)  UF0 active alternative setting register (UF0AAS)**, **(38)   UF0 endpoint 1 interface mapping register (UF0E1IM)** to **(42)  UF0 endpoint 7 interface mapping register (UF0E7IM)**).

**4.** ? and #: Value transmitted from host (information on Interface numbers 0 to 4)
It is determined by the UF0 active interface number register (UF0AIFN) and UF0 active alternative setting register (UF0AAS) whether an Alternate Setting request corresponding to each Interface number is correctly responded to or not, depending on whether the Interface number and Alternate Setting of the target are valid or not.

**(2) Processing**

The processing of an automatic request in the Default state, Addressed state, and Configured state is described below.

**Remark**   Default state: State in which an operation is performed with the Default address

Addressed state: State after an address has been allocated

Configured state: State after SET_CONFIGURATION wValue = 1 has been correctly received

**(a) CLEAR_FEATURE() request**

A STALL response is made in the status stage if the CLEAR_FEATURE() request cannot be cleared, if FEATURE does not exist, or if the target is an interface or an endpoint that does not exist.  A STALL response is also made if the wLength value is other than 0.

- Default state:      The correct response is made when the CLEAR_FEATURE() request has been received only if the target is a device or a request for Endpoint0; otherwise a STALL response is made in the status stage.
- Addressed state:   The correct response is made when the CLEAR_FEATURE() request has been received only if the target is a device or a request for Endpoint0; otherwise a STALL response is made in the status stage.
- Configured state:  The correct response is made when the CLEAR_FEATURE() request has been received only if the target is a device or a request for an endpoint that exists; otherwise a STALL response is made in the status stage.

When the CLEAR_FEATURE() request has been correctly processed, the corresponding bit of the UF0 CLR request register (UF0CLR) is set to 1, the EnHALT bit of the UF0 EPn status register L (UF0EnSL) is cleared to 0, and an interrupt is issued (n = 0 to 4, 7).  If the CLEAR_FEATURE() request is received when the subject is an endpoint, the toggle bit (that controls switching between DATA0 and DATA1) of the corresponding endpoint is always re-set to DATA0.

**(b) GET_CONFIGURATION() request**

A STALL response is made in the data stage if any of wValue, wIndex, or wLength is other than the values shown in Table 20-3.

- Default state:      The value stored in the UF0 configuration register (UF0CNF) is returned when the GET_CONFIGURATION() request has been received.
- Addressed state:   The value stored in the UF0CNF register is returned when the GET_CONFIGURATION() request has been received.
- Configured state:  The value stored in the UF0CNF register is returned when the GET_CONFIGURATION() request has been received.

**(c) GET_DESCRIPTOR() request**

If the subject descriptor has a length that is a multiple of wMaxPacketSize, a Null packet is returned to indicate the end of the data stage. If the length of the descriptor at this time is less than the wLength value, the entire descriptor is returned; if the length of the descriptor is greater than the wLength value, the descriptor up to the wLength value is returned.

- Default state: The value stored in UF0 device descriptor register n (UF0DDn) and UF0 configuration/interface/endpoint descriptor register m (UF0CIEm) is returned (n = 0 to 17, m = 0 to 255) when the GET_DESCRIPTOR() request has been received.
- Addressed state: The value stored in the UF0DDn register and UF0CIEm register is returned when the GET_DESCRIPTOR() request has been received.
- Configured state: The value stored in the UF0DDn register and UF0CIEm register is returned when the GET_DESCRIPTOR() request has been received.

A descriptor of up to 256 bytes can be stored in the UF0CIEm register. To return a descriptor of more than 256 bytes, set the CDCGDST bit of the UF0MODC register to 1 and process the GET_DESCRIPTOR() request by FW.

Store the value of the total number of bytes of the descriptor set by the UF0CIEm register − 1 in the UF0 descriptor length register (UF0DSCL). The transfer data is controlled by the value of this data + 1 and wLength.

**(d) GET_INTERFACE() request**

If either of wValue and wLength is other than that shown in Table 20-3, or if wIndex is other than that set by the UF0 active interface number register (UF0AIFN), a STALL response is made in the data stage.

- Default state: A STALL response is made in the data stage when the GET_INTERFACE() request has been received.
- Addressed state: A STALL response is made in the data stage when the GET_INTERFACE() request has been received.
- Configured state: The value stored in the UF0 interface n register (UF0IFn) corresponding to the wIndex value is returned (n = 0 to 4) when the GET_INTERFACE() request has been received.

**(e) GET_STATUS() request**

A STALL response is made in the data stage if any of wValue, wIndex, or wLength is other than the values shown in Table 20-3. A STALL response is also made in the data stage if the target is an interface or an endpoint that does not exist.

- Default state:       The value stored in the target status register[Note] is returned only when the GET_STATUS() request has been received and when the request is for a device or Endpoint0; otherwise a STALL response is made in the data stage.
- Addressed state:   The value stored in the target status register[Note] is returned only when the GET_STATUS() request has been received and when the request is for a device or Endpoint0; otherwise a STALL response is made in the data stage.
- Configured state:  The value stored in the target status register[Note] is returned only when the GET_STATUS() request has been received and when the request is for a device or an endpoint that exists; otherwise a STALL response is made in the data stage.

   **Note** The target status register is as follows.
   - If the target is a device: UF0 device status register L (UF0DSTL)
   - If the target is endpoint 0: UF0 EP0 status register L (UF0E0SL)
   - If the target is endpoint n: UF0 EPn status register L (UF0EnSL) (n = 1 to 4, 7)

**(f) SET_ADDRESS() request**

A STALL response is made in the status stage if either of wIndex or wLength is other than the values shown in Table 20-3. A STALL response is also made if the specified device address is greater than 127.

- Default state:       The device enters the Addressed state and changes the USB Address value to be input to SIE into a specified address value if the specified address is other than 0 when the SET_ADDRESS() request has been received. If the specified address is 0, the device remains in the Default state.
- Addressed state:   The device enters the Default state and returns the USB Address value to be input to SIE to the default address if the specified address is 0 when the SET_ADDRESS() request has been received. If the specified address is other than 0, the device remains in the Addressed state, and changes the USB Address value to be input to SIE into a specified new address value.
- Configured state:  The device remains in the Configured state and returns the USB Address value to be input to SIE to the default address if the specified address is 0 when the SET_ADDRESS() request has been received. In this case, the endpoints other than endpoint 0 remain valid, and control transfer (IN), control transfer (OUT), bulk transfer and interrupt transfer for an endpoint other than endpoint 0 are also acknowledged. If the specified address is other than 0, the device remains in the Configured state and changes the USB Address value to be input to SIE into a specified new address value.

**(g) SET_CONFIGURATION() request**

If any of wValue, wIndex, or wLength is other than the values shown in Table 20-3, a STALL response is made in the status stage.

- Default state: The CONF bit of the UF0 mode status register (UF0MODS) and the UF0 configuration register (UF0CNF) are set to 1 if the specified configuration value is 1 when the SET_CONFIGURATION() request has been received. If the specified configuration value is 0, the CONF bit of the UF0MODS register and UF0CNF register are cleared to 0. In other words, the device skips the Addressed state and moves to the Configured state in which it responds to the Default address.
- Addressed state: The CONF bit of the UF0MODS register and UF0CNF register are set to 1 and the device enters the Configured state if the specified configuration value is 1 when the SET_CONFIGURATION() request has been received. If the specified configuration value is 0, the device remains in the Addressed state.
- Configured state: The CONF bit of the UF0MODS register and UF0CNF register are set to 1 and the device returns to the Addressed state if the specified configuration value is 0 when the SET_CONFIGURATION() request has been received. If the specified configuration value is 1, the device remains in the Configured state.

If the SET_CONFIGURATION() request has been correctly processed, the target bit of the UF0 SET request register (UF0SET) is set to 1, and an interrupt is issued. All Halt Features are cleared after the SET_CONFIGURATION() request has been completed even if the specified configuration value is the same as the current configuration value. If the SET_CONFIGURATION() request has been correctly processed, the data toggle of all endpoints is always initialized to DATA0 again (it is defined that the default status, Alternative Setting 0, is set from when the SET_CONFIGURATION request is received to when the SET_INTERFACE request is received).

**(h) SET_FEATURE() request**

A STALL response is made in the status stage if the SET_FEATURE() request is for a Feature that cannot be set or does not exist, or if the target is an interface or an endpoint that does not exist. A STALL response is also made if the wLength value is other than 0.

- Default state: The correct response is made when the SET_FEATURE() request has been received, only if the request is for a device or Endpoint0; otherwise a STALL response is made in the status stage.
- Addressed state: The correct response is made when the SET_FEATURE() request has been received, only if the request is for a device or Endpoint0; otherwise a STALL response is made in the status stage.
- Configured state: The correct response is made when the SET_FEATURE() request has been received, only if the request is for a device or an endpoint that exists; otherwise a STALL response is made in the status stage.

When the SET_FEATURE() request has been correctly processed, the target bit of the UF0 SET request register (UF0SET) and the EnHALT bit of the UF0 EPn status register L (UF0EnSL) are set to 1, and an interrupt is issued (n = 0 to 4, 7).

**(i) SET_INTERFACE() request**

If wLength is other than the values shown in Table 20-3, if wIndex is other than the value set to the UF0 active interface number register (UF0AIFN), or if wValue is other than the value set to the UF0 active alternative setting register (UF0AAS), a STALL response is made in the status stage.

- Default state: A STALL response is made in the status stage when the SET_INTERFACE() request has been received.
- Addressed state: A STALL response is made in the status stage when the SET_INTERFACE() request has been received.
- Configured state: Null packet is transmitted in the status stage when the SET_INTERFACE() request has been received.

When the SET_INTERFACE() request has been correctly processed, an interrupt is issued. All the Halt Features of the endpoint linked to the target Interface are cleared after the SET_INTERFACE() request has been cleared. The data toggle of all the endpoints related to the target Interface number is always initialized again to DATA0. When the currently selected Alternative Setting is to be changed by correctly processing the SET_INTERFACE() request, the FIFO of the endpoint that is affected is completely cleared, and all the related interrupt sources are also initialized.

When the SET_INTERFACE() request has been completed, the FIFO of all the endpoints linked to the target Interface are cleared. At the same time, Halt Feature and Data PID are initialized, and the related UF0 INT status n register (UF0ISn) is cleared to 0 (n = 0 to 4). (Only Halt Feature and Data PID are cleared when the SET_CONFIGURATION request has been completed.)

If the target Endpoint is not supported by the SET_INTERFACE() request during DMA transfer, the DMA request signal is immediately deasserted, and the FIFO of the Endpoint that has been linked when the SET_INTERFACE() request has been completed is completely cleared. As a result of this clearing of the FIFO, data transferred by DMA is not correctly processed.

**20.5.2 Other requests**

**(1) Response and processing**

The following table shows how other requests are responded to and processed.

**Table 20-4. Response and Processing of Other Requests**

| Request | Response and Processing |
|---|---|
| GET_DESCRIPTOR String | Generation of CPUDEC interrupt request |
| GET_STATUS Interface | Automatic STALL response |
| CLEAR_FEATURE Interface | Automatic STALL response |
| SET_FEATURE Interface | Automatic STALL response |
| all SET_DESCRIPTOR | Generation of CPUDEC interrupt request |
| All other requests | Generation of CPUDEC interrupt request |

## 20.6  Register Configuration

### 20.6.1  USB control registers

#### (1)  USB clock select register (UCKSEL)

The UCKSEL register selects the operation clock of the USB controller.

The UCKSEL register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H     R/W     Address: FFFFFF40H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UCKSEL | 0 | 0 | 0 | 0 | 0 | 0 | UUSEL1 | 0 |

| UUSEL1 | Selection of USB controller operation clock |
|---|---|
| 0 | External clock input from UCLK pin ($f_{USB}$ = 48 MHz) |
| 1 | Main clock ($f_{XX}$ = 48 MHz) |

**Caution    Be sure to set bits 7 to 2, and 0 to "0".**

#### (2)  USB function control register (UFCKMSK)

The UFCKMSK register controls enable/disable of USB function controller operation.

The UFCKMSK register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 03H.

After reset: 03H     R/W     Address: FFFFFF41H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UFCKMSK | 0 | 0 | 0 | 0 | 0 | 0 | UFBUFMSK | UFMSK |

| UFBUFMSK | UFMSK | USB function controller operation enable/stop |
|---|---|---|
| 0 | 0 | Operation enabled |
| 0 | 1 | Operation stopped (set while USB is suspended) |
| 1 | 1 | Operation stopped |
| Other than above | | Setting prohibited |

**(3) USB function select register (UHCKMSK)**

The UHCKMSK register controls the operation of the data-only RAM when the USB controller function is used.

Even when the USB function controller is not being used, the data-only RAM can be used by setting the UHCKMSK register.

The UHCKMSK register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 03H.

After reset: 03H     R/W     Address: FFFFFF42H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UHCKMSK | 0 | 0 | 0 | 0 | 0 | 0 | UHBUFMSK | UHMSK |

| UHBUFMSK | UHMSK | USB controller function selection |
|---|---|---|
| 0 | 0 | USB host controller operation enabled |
| 1 | 0 | Data-only RAM (8 KB) operation enabled (using USB function) |
| 1 | 1 | Use of USB host controller/data-only RAM (8 KB) disabled |
| Other than above | | Setting prohibited |

**Caution    Be sure to set bits 7 to 2 to "0".**

### 20.6.2 USB function controller register list

**(1) EPC control register**

(1/2)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|:---:|:---:|:---:|---------------|
| | | | | 1 | 8 | 16 | |
| 00200000H | UF0 EP0NAK register | UF0E0N | R/W | | √ | | 00H |
| 00200002H | UF0 EP0NAKALL register | UF0E0NA | R/W | | √ | | 00H |
| 00200004H | UF0 EPNAK register | UF0EN | R/W | | √ | | 00H |
| 00200006H | UF0 EPNAK mask register | UF0ENM | R/W | | √ | | 00H |
| 00200008H | UF0 SNDSIE register | UF0SDS | R/W | | √ | | 00H |
| 0020000AH | UF0 CLR request register | UF0CLR | R | | √ | | 00H |
| 0020000CH | UF0 SET request register | UF0SET | R | | √ | | 00H |
| 0020000EH | UF0 EP status 0 register | UF0EPS0 | R | | √ | | 00H |
| 00200010H | UF0 EP status 1 register | UF0EPS1 | R | | √ | | 00H |
| 00200012H | UF0 EP status 2 register | UF0EPS2 | R | | √ | | 00H |
| 00200020H | UF0 INT status 0 register | UF0IS0 | R | | √ | | 00H |
| 00200022H | UF0 INT status 1 register | UF0IS1 | R | | √ | | 00H |
| 00200024H | UF0 INT status 2 register | UF0IS2 | R | | √ | | 00H |
| 00200026H | UF0 INT status 3 register | UF0IS3 | R | | √ | | 00H |
| 00200028H | UF0 INT status 4 register | UF0IS4 | R | | √ | | 00H |
| 0020002EH | UF0 INT mask 0 register | UF0IM0 | R/W | | √ | | 00H |
| 00200030H | UF0 INT mask 1 register | UF0IM1 | R/W | | √ | | 00H |
| 00200032H | UF0 INT mask 2 register | UF0IM2 | R/W | | √ | | 00H |
| 00200034H | UF0 INT mask 3 register | UF0IM3 | R/W | | √ | | 00H |
| 00200036H | UF0 INT mask 4 register | UF0IM4 | R/W | | √ | | 00H |
| 0020003CH | UF0 INT clear 0 register | UF0IC0 | W | | √ | | FFH |
| 0020003EH | UF0 INT clear 1 register | UF0IC1 | W | | √ | | FFH |
| 00200040H | UF0 INT clear 2 register | UF0IC2 | W | | √ | | FFH |
| 00200042H | UF0 INT clear 3 register | UF0IC3 | W | | √ | | FFH |
| 00200044H | UF0 INT clear 4 register | UF0IC4 | W | | √ | | FFH |
| 0020004CH | UF0 INT & DMARQ register | UF0IDR | R/W | | √ | | 00H |
| 0020004EH | UF0 DMA status 0 register | UF0DMS0 | R | | √ | | 00H |
| 00200050H | UF0 DMA status 1 register | UF0DMS1 | R | | √ | | 00H |
| 00200060H | UF0 FIFO clear 0 register | UF0FIC0 | W | | √ | | 00H |
| 00200062H | UF0 FIFO clear 1 register | UF0FIC1 | W | | √ | | 00H |
| 0020006AH | UF0 data end register | UF0DEND | R/W | | √ | | 00H |
| 0020006EH | UF0 GPR register | UF0GPR | W | | √ | | 00H |
| 00200074H | UF0 mode control register | UF0MODC | R/W | | √ | | 00H |
| 00200078H | UF0 mode status register | UF0MODS | R | | √ | | 00H |

(2/2)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits 1 | 8 | 16 | Default Value |
|---------|------------------------|--------|-----|----|----|-----|---------------|
| 00200080H | UF0 active interface number register | UF0AIFN | R/W | | √ | | 00H |
| 00200082H | UF0 active alternative setting register | UF0AAS | R/W | | √ | | 00H |
| 00200084H | UF0 alternative setting status register | UF0ASS | R | | √ | | 00H |
| 00200086H | UF0 endpoint 1 interface mapping register | UF0E1IM | R/W | | √ | | 00H |
| 00200088H | UF0 endpoint 2 interface mapping register | UF0E2IM | R/W | | √ | | 00H |
| 0020008AH | UF0 endpoint 3 interface mapping register | UF0E3IM | R/W | | √ | | 00H |
| 0020008CH | UF0 endpoint 4 interface mapping register | UF0E4IM | R/W | | √ | | 00H |
| 00200092H | UF0 endpoint 7 interface mapping register | UF0E7IM | R/W | | √ | | 00H |

## (2) EPC data hold register

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits 1 | 8 | 16 | Default Value |
|---------|------------------------|--------|-----|----|----|-----|---------------|
| 00200100 H | UF0 EP0 read register | UF0E0R | R | | √ | | Undefined |
| 00200102H | UF0 EP0 length register | UF0E0L | R | | √ | | 00H |
| 00200104H | UF0 EP0 setup register | UF0E0ST | R | | √ | | 00H |
| 00200106H | UF0 EP0 write register | UF0E0W | W | | √ | | Undefined |
| 00200108H | UF0 bulk-out 1 register | UF0BO1 | R | | √ | | Undefined |
| 0020010AH | UF0 bulk-out 1 length register | UF0BO1L | R | | √ | | 00H |
| 0020010CH | UF0 bulk-out 2 register | UF0BO2 | R | | √ | | Undefined |
| 0020010EH | UF0 bulk-out 2 length register | UF0BO2L | R | | √ | | 00H |
| 00200110H | UF0 bulk-in 1 register | UF0BI1 | W | | √ | | Undefined |
| 00200112H | UF0 bulk-in 2 register | UF0BI2 | W | | √ | | Undefined |
| 00200114H | UF0 interrupt 1 register | UF0INT1 | W | | √ | | Undefined |

**(3) EPC request data register**

(1/13)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| 00200144H | UF0 device status register L | UF0DSTL | R/W | | √ | | 00H |
| 0020014CH | UF0 EP0 status register L | UF0E0SL | R/W | | √ | | 00H |
| 00200150H | UF0 EP1 status register L | UF0E1SL | R/W | | √ | | 00H |
| 00200154H | UF0 EP2 status register L | UF0E2SL | R/W | | √ | | 00H |
| 00200158H | UF0 EP3 status register L | UF0E3SL | R/W | | √ | | 00H |
| 0020015CH | UF0 EP4 status register L | UF0E4SL | R/W | | √ | | 00H |
| 00200168H | UF0 EP7 status register L | UF0E7SL | R/W | | √ | | 00H |
| 00200180H | UF0 address register | UF0ADRS | R | | √ | | 00H |
| 00200182H | UF0 configuration register | UF0CNF | R | | √ | | 00H |
| 00200184H | UF0 interface 0 register | UF0IF0 | R | | √ | | 00H |
| 00200186H | UF0 interface 1 register | UF0IF1 | R | | √ | | 00H |
| 00200188H | UF0 interface 2 register | UF0IF2 | R | | √ | | 00H |
| 0020018AH | UF0 interface 3 register | UF0IF3 | R | | √ | | 00H |
| 0020018CH | UF0 interface 4 register | UF0IF4 | R | | √ | | 00H |
| 002001A0H | UF0 descriptor length register | UF0DSCL | R/W | | √ | | 00H |
| 002001A2H | UF0 device descriptor register 0 | UF0DD0 | R/W | | √ | | Undefined |
| 002001A4H | UF0 device descriptor register 1 | UF0DD1 | R/W | | √ | | Undefined |
| 002001A6H | UF0 device descriptor register 2 | UF0DD2 | R/W | | √ | | Undefined |
| 002001A8H | UF0 device descriptor register 3 | UF0DD3 | R/W | | √ | | Undefined |
| 002001AAH | UF0 device descriptor register 4 | UF0DD4 | R/W | | √ | | Undefined |
| 002001ACH | UF0 device descriptor register 5 | UF0DD5 | R/W | | √ | | Undefined |
| 002001AEH | UF0 device descriptor register 6 | UF0DD6 | R/W | | √ | | Undefined |
| 002001B0H | UF0 device descriptor register 7 | UF0DD7 | R/W | | √ | | Undefined |
| 002001B2H | UF0 device descriptor register 8 | UF0DD8 | R/W | | √ | | Undefined |
| 002001B4H | UF0 device descriptor register 9 | UF0DD9 | R/W | | √ | | Undefined |
| 002001B6H | UF0 device descriptor register 10 | UF0DD10 | R/W | | √ | | Undefined |
| 002001B8H | UF0 device descriptor register 11 | UF0DD11 | R/W | | √ | | Undefined |
| 002001BAH | UF0 device descriptor register 12 | UF0DD12 | R/W | | √ | | Undefined |
| 002001BCH | UF0 device descriptor register 13 | UF0DD13 | R/W | | √ | | Undefined |
| 002001BEH | UF0 device descriptor register 14 | UF0DD14 | R/W | | √ | | Undefined |
| 002001C0H | UF0 device descriptor register 15 | UF0DD15 | R/W | | √ | | Undefined |
| 002001C2H | UF0 device descriptor register 16 | UF0DD16 | R/W | | √ | | Undefined |
| 002001C4H | UF0 device descriptor register 17 | UF0DD17 | R/W | | √ | | Undefined |
| 002001C6H | UF0 configuration/interface/endpoint descriptor register 0 | UF0CIE0 | R/W | | √ | | Undefined |
| 002001C8H | UF0 configuration/interface/endpoint descriptor register 1 | UF0CIE1 | R/W | | √ | | Undefined |
| 002001CAH | UF0 configuration/interface/endpoint descriptor register 2 | UF0CIE2 | R/W | | √ | | Undefined |
| 002001CCH | UF0 configuration/interface/endpoint descriptor register 3 | UF0CIE3 | R/W | | √ | | Undefined |

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|----------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| 002001CEH | UF0 configuration/interface/endpoint descriptor register 4 | UF0CIE4 | R/W | | √ | | Undefined |
| 002001D0H | UF0 configuration/interface/endpoint descriptor register 5 | UF0CIE5 | R/W | | √ | | Undefined |
| 002001D2H | UF0 configuration/interface/endpoint descriptor register 6 | UF0CIE6 | R/W | | √ | | Undefined |
| 002001D4H | UF0 configuration/interface/endpoint descriptor register 7 | UF0CIE7 | R/W | | √ | | Undefined |
| 002001D6H | UF0 configuration/interface/endpoint descriptor register 8 | UF0CIE8 | R/W | | √ | | Undefined |
| 002001D8H | UF0 configuration/interface/endpoint descriptor register 9 | UF0CIE9 | R/W | | √ | | Undefined |
| 002001DAH | UF0 configuration/interface/endpoint descriptor register 10 | UF0CIE10 | R/W | | √ | | Undefined |
| 002001DCH | UF0 configuration/interface/endpoint descriptor register 11 | UF0CIE11 | R/W | | √ | | Undefined |
| 002001DEH | UF0 configuration/interface/endpoint descriptor register 12 | UF0CIE12 | R/W | | √ | | Undefined |
| 002001E0H | UF0 configuration/interface/endpoint descriptor register 13 | UF0CIE13 | R/W | | √ | | Undefined |
| 002001E2H | UF0 configuration/interface/endpoint descriptor register 14 | UF0CIE14 | R/W | | √ | | Undefined |
| 002001E4H | UF0 configuration/interface/endpoint descriptor register 15 | UF0CIE15 | R/W | | √ | | Undefined |
| 002001E6H | UF0 configuration/interface/endpoint descriptor register 16 | UF0CIE16 | R/W | | √ | | Undefined |
| 002001E8H | UF0 configuration/interface/endpoint descriptor register 17 | UF0CIE17 | R/W | | √ | | Undefined |
| 002001EAH | UF0 configuration/interface/endpoint descriptor register 18 | UF0CIE18 | R/W | | √ | | Undefined |
| 002001ECH | UF0 configuration/interface/endpoint descriptor register 19 | UF0CIE19 | R/W | | √ | | Undefined |
| 002001EEH | UF0 configuration/interface/endpoint descriptor register 20 | UF0CIE20 | R/W | | √ | | Undefined |
| 002001F0H | UF0 configuration/interface/endpoint descriptor register 21 | UF0CIE21 | R/W | | √ | | Undefined |
| 002001F2H | UF0 configuration/interface/endpoint descriptor register 22 | UF0CIE22 | R/W | | √ | | Undefined |
| 002001F4H | UF0 configuration/interface/endpoint descriptor register 23 | UF0CIE23 | R/W | | √ | | Undefined |
| 002001F6H | UF0 configuration/interface/endpoint descriptor register 24 | UF0CIE24 | R/W | | √ | | Undefined |
| 002001F8H | UF0 configuration/interface/endpoint descriptor register 25 | UF0CIE25 | R/W | | √ | | Undefined |

(3/13)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| 002001FAH | UF0 configuration/interface/endpoint descriptor register 26 | UF0CIE26 | R/W | | √ | | Undefined |
| 002001FCH | UF0 configuration/interface/endpoint descriptor register 27 | UF0CIE27 | R/W | | √ | | Undefined |
| 002001FEH | UF0 configuration/interface/endpoint descriptor register 28 | UF0CIE28 | R/W | | √ | | Undefined |
| 00200200H | UF0 configuration/interface/endpoint descriptor register 29 | UF0CIE29 | R/W | | √ | | Undefined |
| 00200202H | UF0 configuration/interface/endpoint descriptor register 30 | UF0CIE30 | R/W | | √ | | Undefined |
| 00200204H | UF0 configuration/interface/endpoint descriptor register 31 | UF0CIE31 | R/W | | √ | | Undefined |
| 00200206H | UF0 configuration/interface/endpoint descriptor register 32 | UF0CIE32 | R/W | | √ | | Undefined |
| 00200208H | UF0 configuration/interface/endpoint descriptor register 33 | UF0CIE33 | R/W | | √ | | Undefined |
| 0020020AH | UF0 configuration/interface/endpoint descriptor register 34 | UF0CIE34 | R/W | | √ | | Undefined |
| 0020020CH | UF0 configuration/interface/endpoint descriptor register 35 | UF0CIE35 | R/W | | √ | | Undefined |
| 0020020EH | UF0 configuration/interface/endpoint descriptor register 36 | UF0CIE36 | R/W | | √ | | Undefined |
| 00200210H | UF0 configuration/interface/endpoint descriptor register 37 | UF0CIE37 | R/W | | √ | | Undefined |
| 00200212H | UF0 configuration/interface/endpoint descriptor register 38 | UF0CIE38 | R/W | | √ | | Undefined |
| 00200214H | UF0 configuration/interface/endpoint descriptor register 39 | UF0CIE39 | R/W | | √ | | Undefined |
| 00200216H | UF0 configuration/interface/endpoint descriptor register 40 | UF0CIE40 | R/W | | √ | | Undefined |
| 00200218H | UF0 configuration/interface/endpoint descriptor register 41 | UF0CIE41 | R/W | | √ | | Undefined |
| 0020021AH | UF0 configuration/interface/endpoint descriptor register 42 | UF0CIE42 | R/W | | √ | | Undefined |
| 0020021CH | UF0 configuration/interface/endpoint descriptor register 43 | UF0CIE43 | R/W | | √ | | Undefined |
| 0020021EH | UF0 configuration/interface/endpoint descriptor register 44 | UF0CIE44 | R/W | | √ | | Undefined |
| 00200220H | UF0 configuration/interface/endpoint descriptor register 45 | UF0CIE45 | R/W | | √ | | Undefined |
| 00200222H | UF0 configuration/interface/endpoint descriptor register 46 | UF0CIE46 | R/W | | √ | | Undefined |
| 00200224H | UF0 configuration/interface/endpoint descriptor register 47 | UF0CIE47 | R/W | | √ | | Undefined |

RENESAS

(4/13)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|:---:|:---:|:---:|---------------|
| | | | | 1 | 8 | 16 | |
| 00200226H | UF0 configuration/interface/endpoint descriptor register 48 | UF0CIE48 | R/W | | √ | | Undefined |
| 00200228H | UF0 configuration/interface/endpoint descriptor register 49 | UF0CIE49 | R/W | | √ | | Undefined |
| 0020022AH | UF0 configuration/interface/endpoint descriptor register 50 | UF0CIE50 | R/W | | √ | | Undefined |
| 0020022CH | UF0 configuration/interface/endpoint descriptor register 51 | UF0CIE51 | R/W | | √ | | Undefined |
| 0020022EH | UF0 configuration/interface/endpoint descriptor register 52 | UF0CIE52 | R/W | | √ | | Undefined |
| 00200230H | UF0 configuration/interface/endpoint descriptor register 53 | UF0CIE53 | R/W | | √ | | Undefined |
| 00200232H | UF0 configuration/interface/endpoint descriptor register 54 | UF0CIE54 | R/W | | √ | | Undefined |
| 00200234H | UF0 configuration/interface/endpoint descriptor register 55 | UF0CIE55 | R/W | | √ | | Undefined |
| 00200236H | UF0 configuration/interface/endpoint descriptor register 56 | UF0CIE56 | R/W | | √ | | Undefined |
| 00200238H | UF0 configuration/interface/endpoint descriptor register 57 | UF0CIE57 | R/W | | √ | | Undefined |
| 0020023AH | UF0 configuration/interface/endpoint descriptor register 58 | UF0CIE58 | R/W | | √ | | Undefined |
| 0020023CH | UF0 configuration/interface/endpoint descriptor register 59 | UF0CIE59 | R/W | | √ | | Undefined |
| 0020023EH | UF0 configuration/interface/endpoint descriptor register 60 | UF0CIE60 | R/W | | √ | | Undefined |
| 00200240H | UF0 configuration/interface/endpoint descriptor register 61 | UF0CIE61 | R/W | | √ | | Undefined |
| 00200242H | UF0 configuration/interface/endpoint descriptor register 62 | UF0CIE62 | R/W | | √ | | Undefined |
| 00200244H | UF0 configuration/interface/endpoint descriptor register 63 | UF0CIE63 | R/W | | √ | | Undefined |
| 00200246H | UF0 configuration/interface/endpoint descriptor register 64 | UF0CIE64 | R/W | | √ | | Undefined |
| 00200248H | UF0 configuration/interface/endpoint descriptor register 65 | UF0CIE65 | R/W | | √ | | Undefined |
| 0020024AH | UF0 configuration/interface/endpoint descriptor register 66 | UF0CIE66 | R/W | | √ | | Undefined |
| 0020024CH | UF0 configuration/interface/endpoint descriptor register 67 | UF0CIE67 | R/W | | √ | | Undefined |
| 0020024EH | UF0 configuration/interface/endpoint descriptor register 68 | UF0CIE68 | R/W | | √ | | Undefined |
| 00200250H | UF0 configuration/interface/endpoint descriptor register 69 | UF0CIE69 | R/W | | √ | | Undefined |

(5/13)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|:---:|:---:|:---:|---------------|
| | | | | 1 | 8 | 16 | |
| 00200252H | UF0 configuration/interface/endpoint descriptor register 70 | UF0CIE70 | R/W | | √ | | Undefined |
| 00200254H | UF0 configuration/interface/endpoint descriptor register 71 | UF0CIE71 | R/W | | √ | | Undefined |
| 00200256H | UF0 configuration/interface/endpoint descriptor register 72 | UF0CIE72 | R/W | | √ | | Undefined |
| 00200258H | UF0 configuration/interface/endpoint descriptor register 73 | UF0CIE73 | R/W | | √ | | Undefined |
| 0020025AH | UF0 configuration/interface/endpoint descriptor register 74 | UF0CIE74 | R/W | | √ | | Undefined |
| 0020025CH | UF0 configuration/interface/endpoint descriptor register 75 | UF0CIE75 | R/W | | √ | | Undefined |
| 0020025EH | UF0 configuration/interface/endpoint descriptor register 76 | UF0CIE76 | R/W | | √ | | Undefined |
| 00200260H | UF0 configuration/interface/endpoint descriptor register 77 | UF0CIE77 | R/W | | √ | | Undefined |
| 00200262H | UF0 configuration/interface/endpoint descriptor register 78 | UF0CIE78 | R/W | | √ | | Undefined |
| 00200264H | UF0 configuration/interface/endpoint descriptor register 79 | UF0CIE79 | R/W | | √ | | Undefined |
| 00200266H | UF0 configuration/interface/endpoint descriptor register 80 | UF0CIE80 | R/W | | √ | | Undefined |
| 00200268H | UF0 configuration/interface/endpoint descriptor register 81 | UF0CIE81 | R/W | | √ | | Undefined |
| 0020026AH | UF0 configuration/interface/endpoint descriptor register 82 | UF0CIE82 | R/W | | √ | | Undefined |
| 0020026CH | UF0 configuration/interface/endpoint descriptor register 83 | UF0CIE83 | R/W | | √ | | Undefined |
| 0020026EH | UF0 configuration/interface/endpoint descriptor register 84 | UF0CIE84 | R/W | | √ | | Undefined |
| 00200270H | UF0 configuration/interface/endpoint descriptor register 85 | UF0CIE85 | R/W | | √ | | Undefined |
| 00200272H | UF0 configuration/interface/endpoint descriptor register 86 | UF0CIE86 | R/W | | √ | | Undefined |
| 00200274H | UF0 configuration/interface/endpoint descriptor register 87 | UF0CIE87 | R/W | | √ | | Undefined |
| 00200276H | UF0 configuration/interface/endpoint descriptor register 88 | UF0CIE88 | R/W | | √ | | Undefined |
| 00200278H | UF0 configuration/interface/endpoint descriptor register 89 | UF0CIE89 | R/W | | √ | | Undefined |
| 0020027AH | UF0 configuration/interface/endpoint descriptor register 90 | UF0CIE90 | R/W | | √ | | Undefined |
| 0020027CH | UF0 configuration/interface/endpoint descriptor register 91 | UF0CIE91 | R/W | | √ | | Undefined |

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|:---:|:---:|:---:|---------------|
| | | | | 1 | 8 | 16 | |
| 0020027EH | UF0 configuration/interface/endpoint descriptor register 92 | UF0CIE92 | R/W | | √ | | Undefined |
| 00200280H | UF0 configuration/interface/endpoint descriptor register 93 | UF0CIE93 | R/W | | √ | | Undefined |
| 00200282H | UF0 configuration/interface/endpoint descriptor register 94 | UF0CIE94 | R/W | | √ | | Undefined |
| 00200284H | UF0 configuration/interface/endpoint descriptor register 95 | UF0CIE95 | R/W | | √ | | Undefined |
| 00200286H | UF0 configuration/interface/endpoint descriptor register 96 | UF0CIE96 | R/W | | √ | | Undefined |
| 00200288H | UF0 configuration/interface/endpoint descriptor register 97 | UF0CIE97 | R/W | | √ | | Undefined |
| 0020028AH | UF0 configuration/interface/endpoint descriptor register 98 | UF0CIE98 | R/W | | √ | | Undefined |
| 0020028CH | UF0 configuration/interface/endpoint descriptor register 99 | UF0CIE99 | R/W | | √ | | Undefined |
| 0020028EH | UF0 configuration/interface/endpoint descriptor register 100 | UF0CIE100 | R/W | | √ | | Undefined |
| 00200290H | UF0 configuration/interface/endpoint descriptor register 101 | UF0CIE101 | R/W | | √ | | Undefined |
| 00200292H | UF0 configuration/interface/endpoint descriptor register 102 | UF0CIE102 | R/W | | √ | | Undefined |
| 00200294H | UF0 configuration/interface/endpoint descriptor register 103 | UF0CIE103 | R/W | | √ | | Undefined |
| 00200296H | UF0 configuration/interface/endpoint descriptor register 104 | UF0CIE104 | R/W | | √ | | Undefined |
| 00200298H | UF0 configuration/interface/endpoint descriptor register 105 | UF0CIE105 | R/W | | √ | | Undefined |
| 0020029AH | UF0 configuration/interface/endpoint descriptor register 106 | UF0CIE106 | R/W | | √ | | Undefined |
| 0020029CH | UF0 configuration/interface/endpoint descriptor register 107 | UF0CIE107 | R/W | | √ | | Undefined |
| 0020029EH | UF0 configuration/interface/endpoint descriptor register 108 | UF0CIE108 | R/W | | √ | | Undefined |
| 002002A0H | UF0 configuration/interface/endpoint descriptor register 109 | UF0CIE109 | R/W | | √ | | Undefined |
| 002002A2H | UF0 configuration/interface/endpoint descriptor register 110 | UF0CIE110 | R/W | | √ | | Undefined |
| 002002A4H | UF0 configuration/interface/endpoint descriptor register 111 | UF0CIE111 | R/W | | √ | | Undefined |
| 002002A6H | UF0 configuration/interface/endpoint descriptor register 112 | UF0CIE112 | R/W | | √ | | Undefined |
| 002002A8H | UF0 configuration/interface/endpoint descriptor register 113 | UF0CIE113 | R/W | | √ | | Undefined |

(7/13)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| 002002AAH | UF0 configuration/interface/endpoint descriptor register 114 | UF0CIE114 | R/W | | √ | | Undefined |
| 002002ACH | UF0 configuration/interface/endpoint descriptor register 115 | UF0CIE115 | R/W | | √ | | Undefined |
| 002002AEH | UF0 configuration/interface/endpoint descriptor register 116 | UF0CIE116 | R/W | | √ | | Undefined |
| 002002B0H | UF0 configuration/interface/endpoint descriptor register 117 | UF0CIE117 | R/W | | √ | | Undefined |
| 002002B2H | UF0 configuration/interface/endpoint descriptor register 118 | UF0CIE118 | R/W | | √ | | Undefined |
| 002002B4H | UF0 configuration/interface/endpoint descriptor register 119 | UF0CIE119 | R/W | | √ | | Undefined |
| 002002B6H | UF0 configuration/interface/endpoint descriptor register 120 | UF0CIE120 | R/W | | √ | | Undefined |
| 002002B8H | UF0 configuration/interface/endpoint descriptor register 121 | UF0CIE121 | R/W | | √ | | Undefined |
| 002002BAH | UF0 configuration/interface/endpoint descriptor register 122 | UF0CIE122 | R/W | | √ | | Undefined |
| 002002BCH | UF0 configuration/interface/endpoint descriptor register 123 | UF0CIE123 | R/W | | √ | | Undefined |
| 002002BEH | UF0 configuration/interface/endpoint descriptor register 124 | UF0CIE124 | R/W | | √ | | Undefined |
| 002002C0H | UF0 configuration/interface/endpoint descriptor register 125 | UF0CIE125 | R/W | | √ | | Undefined |
| 002002C2H | UF0 configuration/interface/endpoint descriptor register 126 | UF0CIE126 | R/W | | √ | | Undefined |
| 002002C4H | UF0 configuration/interface/endpoint descriptor register 127 | UF0CIE127 | R/W | | √ | | Undefined |
| 002002C6H | UF0 configuration/interface/endpoint descriptor register 128 | UF0CIE128 | R/W | | √ | | Undefined |
| 002002C8H | UF0 configuration/interface/endpoint descriptor register 129 | UF0CIE129 | R/W | | √ | | Undefined |
| 002002CAH | UF0 configuration/interface/endpoint descriptor register 130 | UF0CIE130 | R/W | | √ | | Undefined |
| 002002CCH | UF0 configuration/interface/endpoint descriptor register 131 | UF0CIE131 | R/W | | √ | | Undefined |
| 002002CEH | UF0 configuration/interface/endpoint descriptor register 132 | UF0CIE132 | R/W | | √ | | Undefined |
| 002002D0H | UF0 configuration/interface/endpoint descriptor register 133 | UF0CIE133 | R/W | | √ | | Undefined |
| 002002D2H | UF0 configuration/interface/endpoint descriptor register 134 | UF0CIE134 | R/W | | √ | | Undefined |
| 002002D4H | UF0 configuration/interface/endpoint descriptor register 135 | UF0CIE135 | R/W | | √ | | Undefined |

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|----------------------|--------|-----|:---:|:---:|:---:|--------------|
| | | | | 1 | 8 | 16 | |
| 002002D6H | UF0 configuration/interface/endpoint descriptor register 136 | UF0CIE136 | R/W | | √ | | Undefined |
| 002002D8H | UF0 configuration/interface/endpoint descriptor register 137 | UF0CIE137 | R/W | | √ | | Undefined |
| 002002DAH | UF0 configuration/interface/endpoint descriptor register 138 | UF0CIE138 | R/W | | √ | | Undefined |
| 002002DCH | UF0 configuration/interface/endpoint descriptor register 139 | UF0CIE139 | R/W | | √ | | Undefined |
| 002002DEH | UF0 configuration/interface/endpoint descriptor register 140 | UF0CIE140 | R/W | | √ | | Undefined |
| 002002E0H | UF0 configuration/interface/endpoint descriptor register 141 | UF0CIE141 | R/W | | √ | | Undefined |
| 002002E2H | UF0 configuration/interface/endpoint descriptor register 142 | UF0CIE142 | R/W | | √ | | Undefined |
| 002002E4H | UF0 configuration/interface/endpoint descriptor register 143 | UF0CIE143 | R/W | | √ | | Undefined |
| 002002E6H | UF0 configuration/interface/endpoint descriptor register 144 | UF0CIE144 | R/W | | √ | | Undefined |
| 002002E8H | UF0 configuration/interface/endpoint descriptor register 145 | UF0CIE145 | R/W | | √ | | Undefined |
| 002002EAH | UF0 configuration/interface/endpoint descriptor register 146 | UF0CIE146 | R/W | | √ | | Undefined |
| 002002ECH | UF0 configuration/interface/endpoint descriptor register 147 | UF0CIE147 | R/W | | √ | | Undefined |
| 002002EEH | UF0 configuration/interface/endpoint descriptor register 148 | UF0CIE148 | R/W | | √ | | Undefined |
| 002002F0H | UF0 configuration/interface/endpoint descriptor register 149 | UF0CIE149 | R/W | | √ | | Undefined |
| 002002F2H | UF0 configuration/interface/endpoint descriptor register 150 | UF0CIE150 | R/W | | √ | | Undefined |
| 002002F4H | UF0 configuration/interface/endpoint descriptor register 151 | UF0CIE151 | R/W | | √ | | Undefined |
| 002002F6H | UF0 configuration/interface/endpoint descriptor register 152 | UF0CIE152 | R/W | | √ | | Undefined |
| 002002F8H | UF0 configuration/interface/endpoint descriptor register 153 | UF0CIE153 | R/W | | √ | | Undefined |
| 002002FAH | UF0 configuration/interface/endpoint descriptor register 154 | UF0CIE154 | R/W | | √ | | Undefined |
| 002002FCH | UF0 configuration/interface/endpoint descriptor register 155 | UF0CIE155 | R/W | | √ | | Undefined |
| 002002FEH | UF0 configuration/interface/endpoint descriptor register 156 | UF0CIE156 | R/W | | √ | | Undefined |
| 00200300H | UF0 configuration/interface/endpoint descriptor register 157 | UF0CIE157 | R/W | | √ | | Undefined |

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| 00200302H | UF0 configuration/interface/endpoint descriptor register 158 | UF0CIE158 | R/W | | √ | | Undefined |
| 00200304H | UF0 configuration/interface/endpoint descriptor register 159 | UF0CIE159 | R/W | | √ | | Undefined |
| 00200306H | UF0 configuration/interface/endpoint descriptor register 160 | UF0CIE160 | R/W | | √ | | Undefined |
| 00200308H | UF0 configuration/interface/endpoint descriptor register 161 | UF0CIE161 | R/W | | √ | | Undefined |
| 0020030AH | UF0 configuration/interface/endpoint descriptor register 162 | UF0CIE162 | R/W | | √ | | Undefined |
| 0020030CH | UF0 configuration/interface/endpoint descriptor register 163 | UF0CIE163 | R/W | | √ | | Undefined |
| 0020030EH | UF0 configuration/interface/endpoint descriptor register 164 | UF0CIE164 | R/W | | √ | | Undefined |
| 00200310H | UF0 configuration/interface/endpoint descriptor register 165 | UF0CIE165 | R/W | | √ | | Undefined |
| 00200312H | UF0 configuration/interface/endpoint descriptor register 166 | UF0CIE166 | R/W | | √ | | Undefined |
| 00200314H | UF0 configuration/interface/endpoint descriptor register 167 | UF0CIE167 | R/W | | √ | | Undefined |
| 00200316H | UF0 configuration/interface/endpoint descriptor register 168 | UF0CIE168 | R/W | | √ | | Undefined |
| 00200318H | UF0 configuration/interface/endpoint descriptor register 169 | UF0CIE169 | R/W | | √ | | Undefined |
| 0020031AH | UF0 configuration/interface/endpoint descriptor register 170 | UF0CIE170 | R/W | | √ | | Undefined |
| 0020031CH | UF0 configuration/interface/endpoint descriptor register 171 | UF0CIE171 | R/W | | √ | | Undefined |
| 0020031EH | UF0 configuration/interface/endpoint descriptor register 172 | UF0CIE172 | R/W | | √ | | Undefined |
| 00200320H | UF0 configuration/interface/endpoint descriptor register 173 | UF0CIE173 | R/W | | √ | | Undefined |
| 00200322H | UF0 configuration/interface/endpoint descriptor register 174 | UF0CIE174 | R/W | | √ | | Undefined |
| 00200324H | UF0 configuration/interface/endpoint descriptor register 175 | UF0CIE175 | R/W | | √ | | Undefined |
| 00200326H | UF0 configuration/interface/endpoint descriptor register 176 | UF0CIE176 | R/W | | √ | | Undefined |
| 00200328H | UF0 configuration/interface/endpoint descriptor register 177 | UF0CIE177 | R/W | | √ | | Undefined |
| 0020032AH | UF0 configuration/interface/endpoint descriptor register 178 | UF0CIE178 | R/W | | √ | | Undefined |
| 0020032CH | UF0 configuration/interface/endpoint descriptor register 179 | UF0CIE179 | R/W | | √ | | Undefined |

(10/13)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| 0020032EH | UF0 configuration/interface/endpoint descriptor register 180 | UF0CIE180 | R/W | | √ | | Undefined |
| 00200330H | UF0 configuration/interface/endpoint descriptor register 181 | UF0CIE181 | R/W | | √ | | Undefined |
| 00200332H | UF0 configuration/interface/endpoint descriptor register 182 | UF0CIE182 | R/W | | √ | | Undefined |
| 00200334H | UF0 configuration/interface/endpoint descriptor register 183 | UF0CIE183 | R/W | | √ | | Undefined |
| 00200336H | UF0 configuration/interface/endpoint descriptor register 184 | UF0CIE184 | R/W | | √ | | Undefined |
| 00200338H | UF0 configuration/interface/endpoint descriptor register 185 | UF0CIE185 | R/W | | √ | | Undefined |
| 0020033AH | UF0 configuration/interface/endpoint descriptor register 186 | UF0CIE186 | R/W | | √ | | Undefined |
| 0020033CH | UF0 configuration/interface/endpoint descriptor register 187 | UF0CIE187 | R/W | | √ | | Undefined |
| 0020033EH | UF0 configuration/interface/endpoint descriptor register 188 | UF0CIE188 | R/W | | √ | | Undefined |
| 00200340H | UF0 configuration/interface/endpoint descriptor register 189 | UF0CIE189 | R/W | | √ | | Undefined |
| 00200342H | UF0 configuration/interface/endpoint descriptor register 190 | UF0CIE190 | R/W | | √ | | Undefined |
| 00200344H | UF0 configuration/interface/endpoint descriptor register 191 | UF0CIE191 | R/W | | √ | | Undefined |
| 00200346H | UF0 configuration/interface/endpoint descriptor register 192 | UF0CIE192 | R/W | | √ | | Undefined |
| 00200348H | UF0 configuration/interface/endpoint descriptor register 193 | UF0CIE193 | R/W | | √ | | Undefined |
| 0020034AH | UF0 configuration/interface/endpoint descriptor register 194 | UF0CIE194 | R/W | | √ | | Undefined |
| 0020034CH | UF0 configuration/interface/endpoint descriptor register 195 | UF0CIE195 | R/W | | √ | | Undefined |
| 0020034EH | UF0 configuration/interface/endpoint descriptor register 196 | UF0CIE196 | R/W | | √ | | Undefined |
| 00200350H | UF0 configuration/interface/endpoint descriptor register 197 | UF0CIE197 | R/W | | √ | | Undefined |
| 00200352H | UF0 configuration/interface/endpoint descriptor register 198 | UF0CIE198 | R/W | | √ | | Undefined |
| 00200354H | UF0 configuration/interface/endpoint descriptor register 199 | UF0CIE199 | R/W | | √ | | Undefined |
| 00200356H | UF0 configuration/interface/endpoint descriptor register 200 | UF0CIE200 | R/W | | √ | | Undefined |
| 00200358H | UF0 configuration/interface/endpoint descriptor register 201 | UF0CIE201 | R/W | | √ | | Undefined |

(11/13)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| 0020035AH | UF0 configuration/interface/endpoint descriptor register 202 | UF0CIE202 | R/W | | √ | | Undefined |
| 0020035CH | UF0 configuration/interface/endpoint descriptor register 203 | UF0CIE203 | R/W | | √ | | Undefined |
| 0020035EH | UF0 configuration/interface/endpoint descriptor register 204 | UF0CIE204 | R/W | | √ | | Undefined |
| 00200360H | UF0 configuration/interface/endpoint descriptor register 205 | UF0CIE205 | R/W | | √ | | Undefined |
| 00200362H | UF0 configuration/interface/endpoint descriptor register 206 | UF0CIE206 | R/W | | √ | | Undefined |
| 00200364H | UF0 configuration/interface/endpoint descriptor register 207 | UF0CIE207 | R/W | | √ | | Undefined |
| 00200366H | UF0 configuration/interface/endpoint descriptor register 208 | UF0CIE208 | R/W | | √ | | Undefined |
| 00200368H | UF0 configuration/interface/endpoint descriptor register 209 | UF0CIE209 | R/W | | √ | | Undefined |
| 0020036AH | UF0 configuration/interface/endpoint descriptor register 210 | UF0CIE210 | R/W | | √ | | Undefined |
| 0020036CH | UF0 configuration/interface/endpoint descriptor register 211 | UF0CIE211 | R/W | | √ | | Undefined |
| 0020036EH | UF0 configuration/interface/endpoint descriptor register 212 | UF0CIE212 | R/W | | √ | | Undefined |
| 00200370H | UF0 configuration/interface/endpoint descriptor register 213 | UF0CIE213 | R/W | | √ | | Undefined |
| 00200372H | UF0 configuration/interface/endpoint descriptor register 214 | UF0CIE214 | R/W | | √ | | Undefined |
| 00200374H | UF0 configuration/interface/endpoint descriptor register 215 | UF0CIE215 | R/W | | √ | | Undefined |
| 00200376H | UF0 configuration/interface/endpoint descriptor register 216 | UF0CIE216 | R/W | | √ | | Undefined |
| 00200378H | UF0 configuration/interface/endpoint descriptor register 217 | UF0CIE217 | R/W | | √ | | Undefined |
| 0020037AH | UF0 configuration/interface/endpoint descriptor register 218 | UF0CIE218 | R/W | | √ | | Undefined |
| 0020037CH | UF0 configuration/interface/endpoint descriptor register 219 | UF0CIE219 | R/W | | √ | | Undefined |
| 0020037EH | UF0 configuration/interface/endpoint descriptor register 220 | UF0CIE220 | R/W | | √ | | Undefined |
| 00200380H | UF0 configuration/interface/endpoint descriptor register 221 | UF0CIE221 | R/W | | √ | | Undefined |
| 00200382H | UF0 configuration/interface/endpoint descriptor register 222 | UF0CIE222 | R/W | | √ | | Undefined |
| 00200384H | UF0 configuration/interface/endpoint descriptor register 223 | UF0CIE223 | R/W | | √ | | Undefined |

(12/13)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| 00200386H | UF0 configuration/interface/endpoint descriptor register 224 | UF0CIE224 | R/W | | √ | | Undefined |
| 00200388H | UF0 configuration/interface/endpoint descriptor register 225 | UF0CIE225 | R/W | | √ | | Undefined |
| 0020038AH | UF0 configuration/interface/endpoint descriptor register 226 | UF0CIE226 | R/W | | √ | | Undefined |
| 0020038CH | UF0 configuration/interface/endpoint descriptor register 227 | UF0CIE227 | R/W | | √ | | Undefined |
| 0020038EH | UF0 configuration/interface/endpoint descriptor register 228 | UF0CIE228 | R/W | | √ | | Undefined |
| 00200390H | UF0 configuration/interface/endpoint descriptor register 229 | UF0CIE229 | R/W | | √ | | Undefined |
| 00200392H | UF0 configuration/interface/endpoint descriptor register 230 | UF0CIE230 | R/W | | √ | | Undefined |
| 00200394H | UF0 configuration/interface/endpoint descriptor register 231 | UF0CIE231 | R/W | | √ | | Undefined |
| 00200396H | UF0 configuration/interface/endpoint descriptor register 232 | UF0CIE232 | R/W | | √ | | Undefined |
| 00200398H | UF0 configuration/interface/endpoint descriptor register 233 | UF0CIE233 | R/W | | √ | | Undefined |
| 0020039AH | UF0 configuration/interface/endpoint descriptor register 234 | UF0CIE234 | R/W | | √ | | Undefined |
| 0020039CH | UF0 configuration/interface/endpoint descriptor register 235 | UF0CIE235 | R/W | | √ | | Undefined |
| 0020039EH | UF0 configuration/interface/endpoint descriptor register 236 | UF0CIE236 | R/W | | √ | | Undefined |
| 002003A0H | UF0 configuration/interface/endpoint descriptor register 237 | UF0CIE237 | R/W | | √ | | Undefined |
| 002003A2H | UF0 configuration/interface/endpoint descriptor register 238 | UF0CIE238 | R/W | | √ | | Undefined |
| 002003A4H | UF0 configuration/interface/endpoint descriptor register 239 | UF0CIE239 | R/W | | √ | | Undefined |
| 002003A6H | UF0 configuration/interface/endpoint descriptor register 240 | UF0CIE240 | R/W | | √ | | Undefined |
| 002003A8H | UF0 configuration/interface/endpoint descriptor register 241 | UF0CIE241 | R/W | | √ | | Undefined |
| 002003AAH | UF0 configuration/interface/endpoint descriptor register 242 | UF0CIE242 | R/W | | √ | | Undefined |
| 002003ACH | UF0 configuration/interface/endpoint descriptor register 243 | UF0CIE243 | R/W | | √ | | Undefined |
| 002003AEH | UF0 configuration/interface/endpoint descriptor register 244 | UF0CIE244 | R/W | | √ | | Undefined |
| 002003B0H | UF0 configuration/interface/endpoint descriptor register 245 | UF0CIE245 | R/W | | √ | | Undefined |

(13/13)

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | |
| 002003B2H | UF0 configuration/interface/endpoint descriptor register 246 | UF0CIE246 | R/W | | √ | | Undefined |
| 002003B4H | UF0 configuration/interface/endpoint descriptor register 247 | UF0CIE247 | R/W | | √ | | Undefined |
| 002003B6H | UF0 configuration/interface/endpoint descriptor register 248 | UF0CIE248 | R/W | | √ | | Undefined |
| 002003B8H | UF0 configuration/interface/endpoint descriptor register 249 | UF0CIE249 | R/W | | √ | | Undefined |
| 002003BAH | UF0 configuration/interface/endpoint descriptor register 250 | UF0CIE250 | R/W | | √ | | Undefined |
| 002003BCH | UF0 configuration/interface/endpoint descriptor register 251 | UF0CIE251 | R/W | | √ | | Undefined |
| 002003BEH | UF0 configuration/interface/endpoint descriptor register 252 | UF0CIE252 | R/W | | √ | | Undefined |
| 002003C0H | UF0 configuration/interface/endpoint descriptor register 253 | UF0CIE253 | R/W | | √ | | Undefined |
| 002003C2H | UF0 configuration/interface/endpoint descriptor register 254 | UF0CIE254 | R/W | | √ | | Undefined |
| 002003C4H | UF0 configuration/interface/endpoint descriptor register 255 | UF0CIE255 | R/W | | √ | | Undefined |

**(4) Bridge register**

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| 00200400H | Bridge interrupt control register | BRGINTT | R/W | | | √ | 0000H |
| 00200402H | Bridge interrupt enable register | BRGINTE | R/W | | | √ | 0000H |
| 00200404H | EPC macro control register | EPCCLT | R/W | | | √ | 0000H |
| 00200408H | CPU I/F bus control register | CPUBCTL | R/W | | | √ | 0000H |

**(5) DMA register**

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| 00200500H | EP1 DMA control register 1 | UF0E1DC1 | R/W | | | √ | 0000H |
| 00200502H | EP1 DMA control register 2 | UF0E1DC2 | R/W | | | √ | 0000H |
| 00200504H | EP2 DMA control register 1 | UF0E2DC1 | R/W | | | √ | 0000H |
| 00200506H | EP2 DMA control register 2 | UF0E2DC2 | R/W | | | √ | 0000H |
| 00200508H | EP3 DMA control register 1 | UF0E3DC1 | R/W | | | √ | 0000H |
| 0020050AH | EP3 DMA control register 2 | UF0E3DC2 | R/W | | | √ | 0000H |
| 0020050CH | EP4 DMA control register 1 | UF0E4DC1 | R/W | | | √ | 0000H |
| 0020050EH | EP4 DMA control register 2 | UF0E4DC2 | R/W | | | √ | 0000H |

**(6) Bulk-in register**

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| 00201000H | UF0 EP1 bulk-in transfer data register | UF0EP1BI | W | | | √ | 0000H |
| 00202000H | UF0 EP3 bulk-in transfer data register | UF0EP3BI | W | | | √ | 0000H |

**(7) Bulk-out register**

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| 00210000H | UF0 EP2 bulk-out transfer data register | UF0EP2BO | R | | √ | √ | 0000H |
| 00220000H | UF0 EP4 bulk-out transfer data register | UF0EP4BO | R | | √ | √ | 0000H |

**(8) Peripheral control register**

| Address | Function Register Name | Symbol | R/W | Manipulatable Bits | | | Default Value |
|---------|------------------------|--------|-----|---|---|----|---------------|
| | | | | 1 | 8 | 16 | |
| 00240000H | USBF DMA request enable register | UFDRQEN | R/W | | √ | √ | 0000H |

### 20.6.3　EPC control registers

#### (1)　UF0 EP0NAK register (UF0E0N)

This register controls NAK of Endpoint0 (except an automatically executed request).

This register can be read or written in 8-bit units (however, bit 0 can only be read).

It takes five USB clocks to reflect the status on this register after the UF0FIC0 and UF0FIC1 registers have been set.  If it is necessary to read the status correctly, therefore, separate a write signal that accesses the UF0FIC0 and UF0FIC1 registers from a read signal that accesses the UF0EPS0, UF0EPS1, UF0EPS2, UF0E0N, and UF0EN registers by at least four USB clocks.

While NAK is being transmitted to Endpoint0 Read, Endpoint2, and Endpoint4, a write access to the EP0NKR bit is ignored.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E0N | 0 | 0 | 0 | 0 | 0 | 0 | EP0NKR | EP0NKW | 00200000H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 1 | EP0NKR | This bit controls NAK to the OUT token to Endpoint0 (except an automatically executed request).  It is automatically set to 1 by hardware when Endpoint0 has correctly received data.  It is also cleared to 0 by hardware when the data of the UF0E0R register has been read by FW (counter value = 0).<br>　　1: Transmit NAK.<br>　　0: Do not transmit NAK (default value).<br>Set this bit to 1 by FW when data should not be received from the USB bus for some reason even when USBF is ready for receiving data.  In this case, USBF continues transmitting NAK until this bit is cleared to 0 by FW.  This bit is also cleared to 0 as soon as the UF0E0R register has been cleared. |
| 0 | EP0NKW | This bit indicates how NAK to the IN token to Endpoint0 is controlled (except an automatically executed request).  This bit is automatically cleared to 0 by hardware when the data of Endpoint0 is transmitted and the host correctly receives the transmitted data.  The data of the UF0E0W register is retained until this bit is cleared.  Therefore, it is not necessary to rewrite this bit even in the case of a retransmission request that is made if the host could not receive data correctly.  To send a short packet, be sure to set the E0DED bit of the UF0DEND register to 1.  This bit is automatically set to 1 when the FIFO is full.  As soon as the E0DED bit of the UF0DEND register is set to 1, the EP0NKW bit is automatically set to 1 at the same time.<br>　　1: Do not transmit NAK.<br>　　0: Transmit NAK (default value).<br>If control transfer enters the status stage while ACK cannot be correctly received in the data stage, this bit is cleared to 0 as soon as the UF0E0W register is cleared.  This bit is also cleared to 0 when UF0E0W is cleared by FW. |

Next, the procedure of a SETUP transaction that uses IN/OUT tokens is explained below.

**(a) When IN token is used (except a request automatically executed by hardware)**

FW should be used to clear the PROT bit of the UF0IS1 register to 0 after receiving the CPUDEC interrupt and before reading data from the UF0E0ST register.  Next, perform processing in accordance with the request and, if it is necessary to return data by an IN token, write data to the UF0E0W register.  Confirm that the PROT bit of the UF0IS1 register is 0 after writing has been completed, and set the E0DED bit of the UF0DEND register to 1. The hardware sends out data at the first IN token after the EP0NKW bit has been set to 1.  If the PROT bit of the UF0IS1 register is 1, it indicates that a SETUP transaction has occurred again before completion of control transfer.  In this case, clear the PROT bit of the UF0IS1 register to 0 by clearing the PROTC bit of the UF0IC1 register to 0, and then read data from the UF0E0ST register again.  A request received later can be read.

**(b) When OUT token is used (except a request automatically executed by hardware)**

FW should be used to clear the PROT bit of the UF0IS1 register after receiving the CPUDEC interrupt and before reading data from the UF0E0ST register.  Confirm that the PROT bit of the UF0IS1 register is 0 before reading data from the UF0E0R register.  If the PROT bit is 1, it means that invalid data is retained.  Clear the FIFO by FW (the EP0NKR bit is automatically cleared to 0).  If the PROT bit of the UF0IS1 register is 0, read the data of the UF0E0L register and read as many data from the UF0E0R register as set.  When reading data from the UF0E0R register has been completed (when the counter of the UF0E0R register has been cleared to 0), the hardware automatically clears the EP0NKR bit to 0.

**(2) UF0 EP0NAKALL register (UF0E0NA)**

This register controls NAK to all the requests of Endpoint0.  It is also valid for automatically executed requests.

This register can be read or written in 8-bit units.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E0NA | 0 | 0 | 0 | 0 | 0 | 0 | 0 | EP0NKA | 00200002H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 0 | EP0NKA | This bit controls NAK to a transaction other than a SETUP transaction to Endpoint0 (including an automatically executed request).  This bit is manipulated by FW.<br>    1: Transmit NAK.<br>    0: Do not transmit NAK (default value).<br>This register is used to prevent a conflict between a write access by FW and a read access from SIE when the data used for an automatically executed request is to be changed.  It postpones reflecting a write access on this bit from FW while an access from SIE is being made.  Before rewriting the request data register from FW, confirm that this bit has been correctly set to 1.<br>Setting this bit to 1 is reflected only in the following cases.<br>  • Immediately after USBF has been reset and a SETUP token has never been received<br>  • Immediately after reception of Bus Reset and a SETUP token has never been received<br>  • PID of a SETUP token has been detected<br>  • The stage has been changed to the status stage<br>Clearing this bit to 0 is reflected immediately, except while an IN token is being received and a NAK response is being made.<br>Setting the EP0NKA bit to 1 is reflected in the above four cases during Endpoint0 transfer, but it is reflected immediately after data has been written to the bit while Endpoint0 is transferring no data. |

RENESAS

**(3) UF0 EPNAK register (UF0EN)**

This register controls NAK of endpoints other than Endpoint0.

This register can be read or written in 8-bit units (however, bits 4, 1, and 0 can only be read).

The BKO2NK bit can be written only when the BKO2NKM bit of the UF0ENM register is 1 and the BKO1NK bit can be written only when the BKO1NKM bit of the UF0ENM register is 1.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7) and the current setting of the interface.

It takes five USB clocks to reflect the status on this register after the UF0FIC0 and UF0FIC1 registers have been set. If it is necessary to read the status correctly, therefore, separate a write signal that accesses the UF0FIC0 and UF0FIC1 registers from a read signal that accesses the UF0EPS0, UF0EPS1, UF0EPS2, UF0E0N, and UF0EN registers by at least four USB clocks.

While NAK is being transmitted to Endpoint0 Read, Endpoint2, and Endpoint4, a write access to the BKO1NK and BKO2NK bits is ignored.

Be sure to clear bits 7 to 5 to "0". If it is set to 1, the operation is not guaranteed.

(1/4)

| | 7 | 6 | | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0EN | 0 | 0 | 0 | IT1NK | BKO2NK | BKO1NK | BKI2NK | BKI1NK | 00200004H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 4 | IT1NK | This bit controls NAK to Endpoint7 (interrupt 1 transfer).<br>It is automatically set to 1 and transmission is started when the UF0INT1 register has become full as a result of writing data to it. To send a short packet that does not make the FIFO full, set the IT1DEND bit of the UF0DEND register to 1. As soon as the IT1DEND bit has been set to 1, this bit is automatically set to 1.<br>    1: Do not transmit NAK.<br>    0: Transmit NAK (default value).<br>This bit is also cleared to 0 when the UF0INT1 register has been cleared. |

| Bit position | Bit name | Function |
|---|---|---|
| 3 | BKO2NK | This bit controls NAK to Endpoint4 (bulk 2 transfer (OUT)).<br>　　1: Transmit NAK.<br>　　0: Do not transmit NAK (default value).<br>This bit is set to 1 only when the FIFO connected to the SIE side of the UF0BO2 register (64-byte FIFO of bank configuration) cannot receive data. It is cleared to 0 when a toggle operation is performed. The bank is changed (toggle operation) when the following conditions are satisfied.<br>　• Data correctly received is stored in the FIFO connected to the SIE side.<br>　• The value of the FIFO counter connected to the CPU side is 0 (completion of reading).<br>FW should be used to read data of the UF0BO2L register when it has received the BLKO2DT interrupt request and read as many data from the UF0BO2 register as the value of that data. To not receive data from the USB bus for some reason even if USBF is ready to receive data, set this bit to 1 by FW. In this case, USBF keeps transmitting NAK until the FW clears this bit to 0. This bit is also cleared to 0 as soon as the UF0BO2 register has been cleared. |
| 2 | BKO1NK | This bit controls NAK to Endpoint2 (bulk 1 transfer (OUT)).<br>　　1: Transmit NAK.<br>　　0: Do not transmit NAK (default value).<br>This bit is set to 1 only when the FIFO connected to the SIE side of the UF0BO1 register (64-byte FIFO of bank configuration) cannot receive data. It is cleared to 0 when a toggle operation is performed. The bank is changed (toggle operation) when the following conditions are satisfied.<br>　• Data correctly received is stored in the FIFO connected to the SIE side.<br>　• The value of the FIFO counter connected to the CPU side is 0 (completion of reading).<br>FW should be used to read data of the UF0BO1L register when it has received the BLKO1DT interrupt request and read as many data from the UF0BO1 register as the value of that data. To not receive data from the USB bus for some reason even if USBF is ready to receive data, set this bit to 1 by FW. In this case, USBF keeps transmitting NAK until the FW clears this bit to 0. This bit is also cleared to 0 as soon as the UF0BO1 register has been cleared. |

**Cautions 1. If DMA is enabled while data is being read from the UF0BO2 register in the PIO mode, a DMA request is immediately issued.**

**2. If the last data of the FIFO on the CPU side is read in the DMA transfer mode, the DMA request signal becomes inactive.**

**3. If the TC signal is received in the DMA transfer mode, the DMA request signal becomes inactive.**

| Bit position | Bit name | Function |
|---|---|---|
| 1 | BKI2NK | This bit controls NAK to Endpoint3 (bulk 2 transfer (IN)).<br>　　1: Do not transmit NAK.<br>　　0: Transmit NAK (default value).<br>This bit is cleared to 0 only when the FIFO connected to the SIE side of the UF0BI2 register (64-byte FIFO of bank configuration) cannot receive data.  It is set to 1 when a toggle operation is performed (the data of the UF0BI2 register is retained until transmission has been correctly completed).  The bank is changed (toggle operation) when the following conditions are satisfied.<br>　• Data is correctly written to the FIFO connected to the CPU bus side (writing has been completed and the FIFO is full or the UF0DEND register is set).<br>　• The value of the FIFO counter connected to the SIE side is 0.<br>This bit is automatically set to 1 and data transmission is started when the FIFO on the CPU side becomes full and a FIFO toggle operation is performed as a result of writing data to the FIFO.  However, if the FIFO on the CPU side becomes full as a result of writing data to it by DMA while the BKI2T bit of the UF0DEND register is cleared to 0, the toggle operation is not performed because the condition of the toggle operation is not satisfied until the BKI2DED bit of the UF0DEND register is set to 1.  To send a short packet that does not make the FIFO on the CPU side full, set the BKI2DED bit to 1 after completing writing data.  When the BKI2DED bit is set to 1, a toggle operation is performed and at the same time, this bit is automatically set to 1.  This bit is also cleared to 0 as soon as the UF0BI2 register has been cleared. |

**Cautions 1.  If DMA is enabled while data is being written to the UF0BI2 register in the PIO mode, a DMA request is immediately issued.**

**2.  If 64-byte data is written in the DMA transfer mode, the DMA request signal becomes inactive.  If the BKI2NK bit is then set to 1, data is transmitted in synchronization with an IN token.  The DMA request signal becomes active again as long as the DMA request is not masked as soon as the FIFO is toggled.  If the BKI2NK bit is not set, data is not transmitted even if an IN token has been received.  In this case, set the BKI2DED bit of the UF0DEND register to 1.**

**3.  If the TC signal is received in the DMA transfer mode, the DMA request signal becomes inactive.  At the same time, the DMA request is masked.  If the BKI2NK bit is not set to 1, data is not transmitted even if an IN token is received.  When the BKI2DED bit of the UF0DEND register is set to 1 by FW, data is transmitted in synchronization with the IN token. To execute DMA transfer again, unmask the DMA request.**

(4/4)

| Bit position | Bit name | Function |
|---|---|---|
| 0 | BKI1NK | This bit controls NAK to Endpoint1 (bulk 1 transfer (IN)).<br>    1: Do not transmit NAK.<br>    0: Transmit NAK (default value).<br>This bit is cleared to 0 only when the FIFO connected to the SIE side of the UF0BI1 register (64-byte FIFO of bank configuration) cannot receive data. It is set to 1 when a toggle operation is performed (the data of the UF0BI1 register is retained until transmission has been correctly completed). The bank is changed (toggle operation) when the following conditions are satisfied.<br>   • Data is correctly written to the FIFO connected to the CPU bus side (writing has been completed and the FIFO is full or the UF0DEND register is set).<br>   • The value of the FIFO counter connected to the SIE side is 0.<br>This bit is automatically set to 1 and data transmission is started when the FIFO on the CPU side becomes full and a FIFO toggle operation is performed as a result of writing data to the FIFO. However, if the FIFO on the CPU side becomes full as a result of writing data to it by DMA while the BKI1T bit of the UF0DEND register is cleared to 0, the toggle operation is not performed because the condition of the toggle operation is not satisfied until the BKI1DED bit of the UF0DEND register is set to 1. To send a short packet that does not make the FIFO on the CPU side full, set the BKI1DED bit to 1 after completing writing data. When the BKI1DED bit is set to 1, a toggle operation is performed and at the same time, this bit is automatically set to 1. This bit is also cleared to 0 as soon as the UF0BI1 register has been cleared. |

**Cautions 1. If DMA is enabled while data is being written to the UF0BI1 register in the PIO mode, a DMA request is immediately issued.**

**2. If 64-byte data is written in the DMA transfer mode, the DMA request signal becomes inactive. If the BKI1NK bit is then set to 1, data is transmitted in synchronization with an IN token. The DMA request signal becomes active again as long as the DMA request is not masked as soon as the FIFO is toggled. If the BKI1NK bit is not set, data is not transmitted even if an IN token has been received. In this case, set the BKI1DED bit of the UF0DEND register to 1.**

**3. If the TC signal is received in the DMA transfer mode, the DMA request signal becomes inactive. At the same time, the DMA request is masked. If the BKI1NK bit is not set to 1, data is not transmitted even if an IN token is received. When the BKI1DED bit of the UF0DEND register is set to 1 by FW, data is transmitted in synchronization with the IN token. To execute DMA transfer again, unmask the DMA request.**

**(4) UF0 EPNAK mask register (UF0ENM)**

This register controls masking a write access to the UF0EN register.

This register can be read or written in 8-bit units.

Be sure to clear bits 7 to 4, 1, and 0 to "0". If it is set to 1, the operation is not guaranteed.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0ENM | 0 | 0 | 0 | 0 | BKO2NKM | BKO1NKM | 0 | 0 | 00200006H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 3 | BKO2NKM | This bit specifies whether a write access to bit 3 (BKO2NK) of the UF0EN register is masked or not.<br>    1: Do not mask.<br>    0: Mask (default value). |
| 2 | BKO1NKM | This bit specifies whether a write access to bit 2 (BKO1NK) of the UF0EN register is masked or not.<br>    1: Do not mask.<br>    0: Mask (default value). |

**(5) UF0 SNDSIE register (UF0SDS)**

This register performs manipulation such as no handshake. It can directly manipulate the pins of SIE.

This register can be read or written in 8-bit units.

Be sure to clear bit 2 to "0". If it is set to 1, the operation is not guaranteed.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0SDS | 0 | 0 | 0 | 0 | SNDSTL | 0 | 0 | RSUMIN | 00200008H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 3 | SNDSTL | This bit makes Endpoint0 issue a STALL handshake. Setting this bit to 1 if a request for CPUDEC processing is not supported by the system results in a STALL handshake response. If an unsupported wValue is sent by the SET_CONFIGURATION or SET_INTERFACE request, the hardware sets this bit to 1. If a problem occurs in Endpoint0 due to overrun of an automatically executed request, this bit is also set to 1. However, the E0HALT bit of the UF0E0SL register is not set to 1.<br>   1: Respond with STALL handshake.<br>   0: Do not respond with STALL handshake (default value).<br>This bit is cleared to 0 and the handshake response to the bus is other than STALL when the next SETUP token is received. To set the SNDSTL bit to 1 by FW, do not write data to the UF0E0W register. Depending on the timing of setting this bit, the STALL response is not made in time, and it may be made to the next transfer after a NAK response has been made.<br>Setting this bit is valid only while an FW-executed request is under execution when this bit is set to 1. It is automatically cleared to 0 when the next SETUP token is received.<br><br>**Remark** The SNDSTL bit is valid only for an FW-executed request. |
| 0 | RSUMIN | This bit outputs the Resume signal onto the USB bus. Writing this bit is invalid unless the RMWK bit of the UF0DSTL register is set to 1.<br>   1: Generate the Resume signal.<br>   0: Do not generate the Resume signal (default value).<br>While this bit is set to 1, the Resume signal continues to be generated. Clear this bit to 0 by FW after a specific time has elapsed. Because the signal is internally sampled at the clock, the operation is guaranteed only while CLK is supplied. Care must be exercised when CLK of the system is stopped. |

**(6) UF0 CLR request register (UF0CLR)**

This register indicates the target of the received CLEAR_FEATURE request.

This register is read-only, in 8-bit units.

This register is meaningful only when an interrupt request is generated.  Each bit is set to 1 after completion of the status stage, and automatically cleared to 0 when this register is read.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7) and the current setting of the interface.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0CLR | 0 | CLREP7 | CLREP4 | CLREP3 | CLREP2 | CLREP1 | CLREP0 | CLRDEV | 0020000AH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 6 to 1 | CLREPn | These bits indicate that a CLEAR_FEATURE Endpoint n request is received and automatically processed.<br>　　1: Automatically processed<br>　　0: Not automatically processed (default value) |
| 0 | CLRDEV | This bit indicates that a CLEAR_FEATURE Device request is received and automatically processed.<br>　　1: Automatically processed<br>　　0: Not automatically processed (default value) |

**Remark**  n = 0 to 4, 7

**(7) UF0 SET request register (UF0SET)**

This register indicates the target of the automatically processed SET_XXXX (except SET_INTERFACE) request.

This register is read-only, in 8-bit units.

This register is meaningful only when an interrupt request is generated.  Each bit is set to 1 after completion of the status stage, and automatically cleared to 0 when this register is read.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0SET | SETCON | 0 | 0 | 0 | 0 | SETEP | 0 | SETDEV | 0020000CH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 | SETCON | This bit indicates that a SET_CONFIGURATION request is received and automatically processed.<br>1: Automatically processed<br>0: Not automatically processed (default value) |
| 2 | SETEP | This bit indicates that a SET_FEATURE Endpoint n request (n = 0 to 4, 7) is received and automatically processed.<br>1: Automatically processed<br>0: Not automatically processed (default value) |
| 0 | SETDEV | This bit indicates that a SET_FEATURE Device request is received and automatically processed.<br>1: Automatically processed<br>0: Not automatically processed (default value) |

**(8) UF0 EP status 0 register (UF0EPS0)**

This register indicates the USB bus status and the presence or absence of register data.

This register is read-only, in 8-bit units.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7) and the current setting of the interface.

It takes five USB clocks to reflect the status on this register after the UF0FIC0 and UF0FIC1 registers have been set. If it is necessary to read the status correctly, therefore, separate writing to the UF0FIC0 and UF0FIC1 registers from reading from the UF0EPS0, UF0EPS1, UF0EPS2, UF0E0N, and UF0EN registers by at least four USB clocks.

(1/2)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0EPS0 | 0 | IT1 | BKOUT2 | BKOUT1 | BKIN2 | BKIN1 | EP0W | EP0R | 0020000EH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 6 | IT1 | These bits indicate that data is in the UF0INT1 register (FIFO). By setting the IT1DEND bit of the UF0DEND register to 1, the status in which data is in the UF0INT1 register can be created even if data is not written to the register (Null data transmission). As soon as the IT1DEND bit of the UF0DEND register is set to 1 even when the counter of the UF0INT1 register is 0, this bit is set to 1 by hardware. It is cleared to 0 after correct transmission.<br>  1: Data is in the register.<br>  0: No data is in the register (default value). |
| 5, 4 | BKOUTn | These bits indicate that data is in the UF0BOn register (FIFO) connected to the CPU side. When the FIFO configuring the UF0BOn register is toggled, this bit is automatically set to 1 by hardware. It is automatically cleared to 0 by hardware when reading the UF0BOn register (FIFO) connected to the CPU side has been completed (counter value = 0). It is not set to 1 when Null data is received (toggling the FIFO does not take place either).<br>  1: Data is in the register.<br>  0: No data is in the register (default value). |
| 3, 2 | BKINn | These bits indicate that data is in the UF0BIn register (FIFO) connected to the CPU side. By setting the BKInDED bit of the UF0DEND register to 1, the status in which data is in the UF0BIn register can be created even if data is not written to the register (Null data transmission). As soon as the BKInDED bit of the UF0DEND register has been set to 1 while the counter of the UF0BIn register is 0, this bit is set to 1 by hardware. It is cleared to 0 when a toggle operation is performed.<br>  1: Data is in the register.<br>  0: No data is in the register (default value). |

**Remark**   n = 1, 2

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 1 | EP0W | This bit indicates that data is in the UF0E0W register (FIFO).  By setting the E0DED bit of the UF0DEND register to 1, the status in which data is in the UF0E0W register can be created even if data is not written to the register (Null data transmission).  As soon as the E0DED bit of the UF0DEND register is set to 1 even when the counter of the UF0E0W register is 0, this bit is set to 1 by hardware.  It is cleared to 0 after correct transmission.<br><br>    1: Data is in the register.<br>    0: No data is in the register (default value). |
| 0 | EP0R | This bit indicates that data is in the UF0E0R register (FIFO).  It is automatically cleared to 0 by hardware when reading the UF0E0R register (FIFO) has been completed (counter value = 0).  It is not set to 1 if Null data is received.<br><br>    1: Data is in the register.<br>    0: No data is in the register (default value). |

**(9)  UF0 EP status 1 register (UF0EPS1)**

This register indicates the USB bus status and the presence or absence of register data.

This register is read-only, in 8-bit units.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0EPS1 | RSUM | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 00200010H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 | RSUM | This bit indicates that the USB bus is in the Resume status.  This bit is meaningful only when an interrupt request is generated.<br>　　1: Suspend status<br>　　0: Resume status (default value)<br>Because sampling is internally performed with the clock, the operation is guaranteed only when CLK is supplied.  Care must be exercised when CLK of the system is stopped.  The INTUSBF1 signal of SIE operates even when CLK is stopped.  It can therefore be supported by making the interrupt control register (UFIC1) valid or lowering the frequency of CLK to the USBF.<br>This bit is automatically cleared to 0 when it is read. |

**(10) UF0 EP status 2 register (UF0EPS2)**

This register indicates the USB bus status and the presence or absence of register data.

This register is read-only, in 8-bit units.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0EPS2 | 0 | 0 | HALT7 | HALT4 | HALT3 | HALT2 | HALT1 | HALT0 | 00200012H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 5 to 0 | HALTn | These bits indicate that Endpoint n is currently stalled. They are set to 1 when a stall condition, such as occurrence of an overrun and reception of an undefined request, is satisfied. These bits are automatically set to 1 by hardware.<br>　1: Endpoint is stalled.<br>　0: Endpoint is not stalled (default value).<br>The SNDSTL bit is set to 1 as soon as the HALT0 bit has been set to 1 as a result of occurrence of an overrun or reception of an undefined request. If the next SETUP token is received in this status, the SNDSTL bit is cleared to 0 and, therefore, the HALT0 bit is also cleared to 0. If Endpoint0 is stalled by the SET_FEATURE Endpoint0 request, this bit is not cleared to 0 until the CLEAR_FEATURE Endpoint0 request is received or Halt Feature is cleared by FW. If the GET_STATUS Endpoint0, CLEAR_FEATURE Endpoint0, or SET_FEATURE Endpoint0 request is received, or if a request to be processed by FW is received due to the CPUDEC interrupt request, the HALT0 bit is masked and cleared to 0, until the next SETUP token is received.<br>The HALTn bit is not cleared to 0 until Endpoint n receives the CLEAR_FEATURE Endpoint request, Halt Feature is cleared by the SET_INTERFACE or SET_CONFIGURATION request to the interface to which the endpoint is linked, or Halt Feature is cleared by FW. When the SET_INTERFACE or SET_CONFIGURATION request is correctly processed, the Halt Feature of all the target endpoints, except Endpoint0, is cleared after the request has been processed, even if the wValue is the same as the currently set value, and these bits are also cleared to 0. Halt Feature of Endpoint0 cannot be cleared if it is set because the STALL response is made in response to the SET_INTERFACE and SET_CONFIGURATION requests. |

**Remark**   n = 0 to 4, 7, 8

**(11) UF0 INT status 0 register (UF0IS0)**

This register indicates the interrupt source. If the contents of this register are changed, the EPCINT0B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSBF0) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC0 register.

**Caution  In the USBF, multiple interrupt sources, such as Bus Reset, Resume, and Short, are ORed internally and are issued as a single interrupt request (INTUSBF0). Therefore, in the case of the occurrence of multiple interrupt sources, they are ORed and issued as an INTUSBF0 interrupt request.**

**For example, if a Bus Reset interrupt source and Resume interrupt source occur, the two sources are ORed and an INTUSBF0 interrupt request is issued.**

**Under these conditions, if the Bus Reset interrupt source is cleared to 0 (UF0IC0.BUSRSTC = 0), the V850ES/JG3-U or V850ES/JH3-U internal INTUSBF0 interrupt request may remain set to 1 since the Resume interrupt source will still remain. The new interrupt request flag (US0BIC.US0BIF), therefore, might not be set to 1.**

**In this case, after performing clear processing for each interrupt request with the INTUSBF0 interrupt servicing routine, confirm the flag status for the UF0IS0 and UF0IS1 registers again, and if there are any interrupt sources with flags set to 1, perform flag clearing (only the applicable bits need to be cleared (do not perform a batch clearing)).**

(1/2)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IS0 | BUSRST | RSUSPD | 0 | SHORT | DMAED | SETRQ | CLRRQ | EPHALT | 00200020H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 | BUSRST | This bit indicates that Bus Reset has occurred.<br>　1: Bus Reset has occurred (interrupt request is generated).<br>　0: Not Bus Reset status (default value) |
| 6 | RSUSPD | This bit indicates that the Resume or Suspend status has occurred. Reference bit 7 of the UF0EPS1 register by FW.<br>　1: Resume or Suspend status has occurred (interrupt request is generated).<br>　0: Resume or Suspend status has not occurred (default value). |
| 4 | SHORT | This bit indicates that data is read from the FIFO of either the UF0BO1 or UF0BO2 register and that the USBSPnB signal (n = 2, 4) is active. It is valid only when the FIFO is full in the DMA mode.<br>　1: USBSPnB signal is active (interrupt request is generated).<br>　0: USBSPnB signal is not active (default value).<br>Identify on which endpoint the operation is performed, by using the UF0DMS1 register.<br>This bit is not automatically cleared to 0 even when the UF0DMS1 register is read by FW. |

| Bit position | Bit name | Function |
|---|---|---|
| 3 | DMAED | This bit indicates that the DMA end (TC) signal for Endpoint n (n = 1 to 4, 7) is active.<br>    1: DMA end signal for Endpoint n has been input (interrupt request is generated).<br>    0: DMA end signal for Endpoint n has not been input (default value).<br>When this bit is set to 1, the DMA request signal for Endpoint n becomes inactive. The DMA request signal for Endpoint n does not become active unless FW enables DMA transfer.<br>Use the UF0DMS0 register to confirm on which endpoint the operation is actually performed. However, this bit is not automatically cleared to 0 even if the UF0DMS0 register is read by FW. |
| 2 | SETRQ | This bit indicates that the SET_XXXX request to be automatically processed has been received and automatically processed (XXXX = CONFIGURATION or FEATURE).<br>    1: SET_XXXX request to be automatically processed has been received (interrupt request is generated).<br>    0: SET_XXXX request to be automatically processed has not been received (default value).<br>This bit is set to 1 after completion of the status stage. Reference the UF0SET register to identify what is the target of the request. This bit is not automatically cleared to 0 even if the UF0SET register is read by FW.<br>The EPHALT bit is also set to 1 when the SET_FEATURE Endpoint request has been received. |
| 1 | CLRRQ | This bit indicates that the CLEAR_FEATURE request has been received and automatically processed.<br>    1: CLEAR_FEATURE request has been received (interrupt request is generated).<br>    0: CLEAR_FEATURE request has not been received (default value).<br>This bit is set to 1 after completion of the status stage. Reference the UF0CLR register to identify what is the target of the request. This bit is not automatically cleared to 0 even if the UF0CLR register is read by FW. |
| 0 | EPHALT | This bit indicates that an endpoint has stalled.<br>    1: Endpoint has stalled (interrupt request is generated).<br>    0: Endpoint has not stalled (default value).<br>This bit is also set to 1 when an endpoint has stalled by setting FW.<br>Identify the endpoint that has stalled, by referencing the UF0EPS2 register. This bit is not automatically cleared to 0 even when the CLEAR_FEATURE Endpoint, SET_INTERFACE, or SET_CONFIGURATION request is received. It is not automatically cleared to 0, either, if the next SETUP token is received in case of overrun of Endpoint0.<br><br>**Caution  Even if Halt Feature of Endpoint0 is set and this interrupt request is generated, bit 0 of the UF0EPS2 register is masked and cleared to 0 between when a SET_FEATURE Endpoint0, CLEAR_FEATURE Endpoint0, or GET_STATUS Endpoint0 request, or FW-processed request is received and when a SETUP token other than the above is received.** |

**(12) UF0 INT status 1 register (UF0IS1)**

This register indicates the interrupt source. If the contents of this register are changed, the EPCINT0B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSBF0) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC1 register. However, the SUCES and STG bits of the UF0IS1 register are automatically cleared to 0 when the next SETUP token has been received.

**Caution**  **In the USBF, multiple interrupt sources, such as Bus Reset, Resume, and Short, are ORed internally and are issued as a single interrupt request (INTUSBF0). Therefore, in the case of the occurrence of multiple interrupt sources, they are ORed and issued as an INTUSBF0 interrupt request.**

**For example, if a Bus Reset interrupt source and Resume interrupt source occur, the two sources are ORed and an INTUSBF0 interrupt request is issued.**

**Under these conditions, if the Bus Reset interrupt source is cleared to 0 (UF0IC0.BUSRSTC = 0), the V850ES/JG3-U or V850ES/JH3-U internal INTUSBF0 interrupt request may remain set to 1 since the Resume interrupt source will still be remaining. The new interrupt request flag (US0BIC.US0BIF), therefore, might not be set to 1.**

**In this case, after performing clear processing for each interrupt request with the INTUSBF0 interrupt servicing routine, confirm the flag status for the UF0IS0 and UF0IS1 registers again, and if there are any interrupt sources with flags set to 1, perform flag clearing (only the applicable bits need to be cleared (do not perform a batch clearing)).**

(1/2)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IS1 | 0 | E0IN | E0INDT | E0ODT | SUCES | STG | PROT | CPU DEC | 00200022H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 6 | E0IN | This bit indicates that an IN token for Endpoint0 has been received and that the hardware has automatically transmitted NAK.<br>　1: IN token is received and NAK is transmitted (interrupt request is generated).<br>　0: IN token is not received (default value). |
| 5 | E0INDT | This bit indicates that data has been correctly transmitted from the UF0E0W register.<br>　1: Transmission from UF0E0W register is completed (interrupt request is generated).<br>　0: Transmission from UF0E0W register is not completed (default value).<br>Data is transmitted in synchronization with the IN token next to the one that set the EP0NKW bit of the UF0E0N register to 1. This bit is automatically set to 1 by hardware when the host correctly receives that data. It is also set to 1 even if the data is a Null packet. This bit is automatically cleared to 0 by hardware when the first write access is made to the UF0E0W register. |

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 4 | E0ODT | This bit indicates that data has been correctly received in the UF0E0R register.<br>　　1: Data is in UF0E0R register (interrupt request is generated).<br>　　0: Data is not in UF0E0R register (default value).<br>This bit is automatically set to 1 by hardware when data has been correctly received. At the same time, the EP0R bit of the UF0EPS0 register is also set to 1. If a Null packet has been received, this bit is not set to 1. It is automatically cleared to 0 by hardware when the FW reads the UF0E0R register and the value of the UF0E0L register becomes 0. |
| 3 | SUCES | This bit indicates that either an FW-processed or hardware-processed request has been received and that the status stage has been correctly completed.<br>　　1: Control transfer has been correctly processed (interrupt request is generated).<br>　　0: Control transfer has not been processed correctly (default value).<br>This bit is set to 1 upon completion of the status stage. It is automatically cleared to 0 by hardware when the next SETUP token is received.<br>This bit is also set to 1 when data with Data PID of 0 (Null data) is received in the status stage of control transfer. |
| 2 | STG | This bit is set to 1 when the stage of control transfer has changed to the status stage. It is valid for both FW-processed and hardware-processed requests. This bit is also set to 1 when the stage of control transfer (without data) has changed to the status stage.<br>　　1: Status stage (interrupt request is generated)<br>　　0: Not status stage (default value)<br>This bit is automatically cleared to 0 by hardware when the next SETUP token is received. It is also set to 1 when the stage of control transfer has changed to the status stage while ACK cannot be correctly received in the data stage. In this case, the EP0NKW bit of the UF0E0N register is also cleared to 0 as soon as the UF0E0W register has been cleared, if the FW is processing control transfer (read). |
| 1 | PROT | This bit indicates that a SETUP token has been received. It is valid for both FW-processed and hardware-processed requests.<br>　　1: SETUP token is correctly received (interrupt request is generated).<br>　　0: SETUP token is not received (default value).<br>This bit is set to 1 when data has been correctly received in the UF0E0ST register. Clear this bit to 0 by FW when the first read access is made to the UF0E0ST register. If it is not cleared to 0 by FW, reception of the next SETUP token cannot be correctly recognized. This bit is used to accurately recognize that a SETUP transaction has been executed again during control transfer. If the SETUP transaction is re-executed during control transfer and if a second request is executed by hardware, the CPUDEC bit is not set to 1, but the PROT bit can be used for recognition of the re-execution. |
| 0 | CPUDEC | This bit indicates that the UF0E0ST register has a request that is to be decoded by FW.<br>　　1: Data is in UF0E0ST register (interrupt request is generated).<br>　　0: Data is not in UF0E0ST register (default value).<br>This bit is automatically cleared to 0 by hardware when all the data of the UF0E0ST register is read. |

**(13) UF0 INT status 2 register (UF0IS2)**

This register indicates the interrupt source. If the contents of this register are changed, the EPCINT1B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSBF0) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC2 register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IS2 | BKI2IN | BKI2DT | BKI1IN | BKI1DT | 0 | 0 | 0 | IT1DT | 00200024H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 5 | BKInIN | These bits indicate that an IN token has been received in the UF0BIn register (Endpoint m) and that NAK has been returned.<br>   1: IN token is received and NAK is transmitted (interrupt request is generated).<br>   0: IN token is not received (default value). |
| 6, 4 | BKInDT | These bits indicate that the FIFO of the UF0BIn register (Endpoint m) has been toggled. This means that data can be written to Endpoint m.<br>   1: FIFO has been toggled (interrupt request is generated).<br>   0: FIFO has not been toggled (default value).<br>The data written to Endpoint m is transmitted in synchronization with the IN token next to the one that set the BKInNK bit of the UF0EN register to 1. When the FIFO has been toggled and then data can be written from the CPU, this bit is automatically set to 1 by hardware. It is also set to 1 when the FIFO has been toggled, even if the data is a Null packet. This bit is automatically cleared to 0 by hardware when the first write access is made to the UF0BIn register. |
| 0 | IT1DT | These bits indicate that data has been correctly received from the UF0INT1 register (Endpoint x).<br>   1: Transmission is completed (interrupt request is generated).<br>   0: Transmission is not completed (default value).<br>Data is transmitted in synchronization with the IN token next to the one that set the ITnNK bit of the UF0EN register to 1. This bit is automatically set to 1 by hardware when the host has correctly received that data. It is automatically cleared to 0 by hardware when the first write access is made to the UF0INT1 register. This bit is also set to 1 even when the data is a Null packet. |

**Remark**  n = 1, 2
          m = 1 and x = 7 where n = 1
          m = 3 where n = 2

**(14) UF0 INT status 3 register (UF0IS3)**

This register indicates the interrupt source.  If the contents of this register are changed, the EPCINT1B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSBF0) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC3 register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 2, 4) and the current setting of the interface.

(1/2)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IS3 | BKO2FL | BKO2NL | BKO2 NAK | BKO2DT | BKO1FL | BKO1NL | BKO1 NAK | BKO1DT | 00200026H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 3 | BKOnFL | These bits indicate that data has been correctly received in the UF0BOn register (Endpoint m) and that both the FIFOs of the CPU and SIE hold the data.<br>  1: Received data is in both the FIFOs of the UF0BOn register (interrupt request is generated).<br>  0: Received data is not in the FIFO on the SIE side of the UF0BOn register (default value).<br>If data is held in both the FIFOs of the CPU and SIE, these bits are automatically set to 1 by hardware.  They are automatically cleared to 0 by hardware when the FIFO is toggled. |
| 6, 2 | BKOnNL | These bits indicate that a Null packet (packet with a length of 0) has been received in the UF0BOn register (Endpoint m).<br>  1: Null packet is received (interrupt request is generated).<br>  0: Null packet is not received (default value).<br>These bits are set to 1 immediately after reception of a Null packet when the FIFO is empty.  They are set to 1 when the FIFO on the CPU side has been completely read if data is in that FIFO. |
| 5, 1 | BKOnNAK | These bits indicate that an OUT token has been received to the UF0BOn register (Endpoint m) and that NAK has been returned.<br>  1: OUT token is received and NAK is transmitted (interrupt request is generated).<br>  0: OUT token is not received (default value). |

**Remark**  n = 1, 2
         m = 2 where n = 1
         m = 4 where n = 2

| Bit position | Bit name | Function |
|---|---|---|
| 4, 0 | BKOnDT | These bits indicate that data has been correctly received in the UF0BOn register (Endpoint m).<br><br>   1: Reception has been completed correctly (interrupt request is generated).<br>   0: Reception has not been completed (default value).<br><br>These bits are automatically set to 1 by hardware when data has been correctly received and the FIFO has been toggled. At the same time, the corresponding bits of the UF0EPS0 register are also set to 1. They are not set to 1 when the data is a Null packet.<br>These bits are automatically cleared to 0 by hardware when the value of the UF0BOnL register becomes 0 as a result of reading the UF0BOn register by FW.<br>These bits are automatically cleared to 0 when all the contents of the FIFO on the CPU side have been read. However, the interrupt request is not cleared if data is in the FIFO on the SIE side at this time, and the INTUSBF1 signal does not become inactive. The signal is kept active if data is successively received. |

**Remark**   n = 1, 2

         m = 2 where n = 1

         m = 4 where n = 2

**(15) UF0 INT status 4 register (UF0IS4)**

This register indicates the interrupt source. If the contents of this register are changed, the EPCINT2B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSBF0) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC4 register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IS4 | 0 | 0 | SETINT | 0 | 0 | 0 | 0 | 0 | 00200028H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 5 | SETINT | This bit indicates that the SET_INTERFACE request has been received and automatically processed.<br>   1: The request has been automatically processed (interrupt request is generated).<br>   0: The request has not been automatically processed (default value).<br>The current setting of this bit can be identified by reading the UF0ASS or UF0IFn register (n = 0 to 4). |

**(16) UF0 INT mask 0 register (UF0IM0)**

This register controls masking of the interrupt sources indicated by the UF0IS0 register.

This register can be read or written in 8-bit units.

FW can mask occurrence of an interrupt request from USBF (INTUSBF0) by writing 1 to the corresponding bit of this register.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IM0 | BUS RSTM | RSU SPDM | 0 | SHORTM | DMA EDM | SET RQM | CLR RQM | EP HALTM | 0020002EH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 | BUSRSTM | This bit masks the Bus Reset interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 6 | RSUSPDM | This bit masks the Resume/Suspend interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 4 | SHORTM | This bit masks the Short interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 3 | DMAEDM | This bit masks the DMA_END interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 2 | SETRQM | This bit masks the SET_RQ interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 1 | CLRRQM | This bit masks the CLR_RQ interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 0 | EPHALTM | This bit masks the EP_Halt interrupt.<br>1: Mask<br>0: Do not mask (default value) |

**(17) UF0 INT mask 1 register (UF0IM1)**

This register controls masking of the interrupt sources indicated by the UF0IS1 register.

This register can be read or written in 8-bit units.

FW can mask occurrence of an interrupt request from USBF (INTUSBF0) by writing 1 to the corresponding bit of this register.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IM1 | 0 | E0INM | E0 INDTM | E0 ODTM | SUCESM | STGM | PROTM | CPU DECM | 00200030H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 6 | E0INM | This bit masks the EP0IN interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 5 | E0INDTM | This bit masks the EP0INDT interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 4 | E0ODTM | This bit masks the EP0OUTDT interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 3 | SUCESM | This bit masks the Success interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 2 | STGM | This bit masks the Stg interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 1 | PROTM | This bit masks the Protect interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 0 | CPUDECM | This bit masks the CPUDEC interrupt.<br>1: Mask<br>0: Do not mask (default value) |

**(18) UF0 INT mask 2 register (UF0IM2)**

This register controls masking of the interrupt sources indicated by the UF0IS2 register.

This register can be read or written in 8-bit units.

FW can mask occurrence of an interrupt request from USBF (INTUSBF0) by writing 1 to the corresponding bit of this register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IM2 | BKI2INM | BKI2 DTM | BKI1INM | BKI1 DTM | 0 | 0 | 0 | IT1DTM | 00200032H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 5 | BKInINM | These bits mask the BLKInIN interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 6, 4 | BKInDTM | These bits mask the BLKInDT interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 0 | IT1DTM | These bits mask the INTnDT interrupt.<br>1: Mask<br>0: Do not mask (default value) |

**Remark**   n = 1, 2

**(19)  UF0 INT mask 3 register (UF0IM3)**

This register controls masking of the interrupt sources indicated by the UF0IS3 register.

This register can be read or written in 8-bit units.

FW can mask occurrence of an interrupt request from USBF (INTUSBF0) by writing 1 to the corresponding bit of this register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 2, 4) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IM3 | BKO2 FLM | BKO2 NLM | BKO2 NAKM | BKO2 DTM | BKO1 FLM | BKO1 NLM | BKO1 NAKM | BKO1 DTM | 00200034H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 3 | BKOnFLM | These bits mask the BLKOnFL interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 6, 2 | BKOnNLM | These bits mask the BLKOnNL interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 5, 1 | BKOnNAKM | These bits mask the BLKOnNK interrupt.<br>1: Mask<br>0: Do not mask (default value) |
| 4, 0 | BKOnDTM | These bits mask the BLKOnDT interrupt.<br>1: Mask<br>0: Do not mask (default value) |

**Remark**   n = 1, 2

**(20) UF0 INT mask 4 register (UF0IM4)**

This register controls masking of the interrupt sources indicated by the UF0IS4 register.

This register can be read or written in 8-bit units.

FW can mask occurrence of an interrupt request from USBF (INTUSBF0) by writing 1 to the corresponding bit of this register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IM4 | 0 | 0 | SETINTM | 0 | 0 | 0 | 0 | 0 | 00200036H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 5 | SETINTM | This bit masks the SET_INT interrupt.<br>1: Mask<br>0: Do not mask (default value) |

**(21) UF0 INT clear 0 register (UF0IC0)**

This register controls clearing the interrupt sources indicated by the UF0IS0 register.

This register is write-only, in 8-bit units.  If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register.  Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware.  Writing 0 to a bit of this register automatically sets the bit to 1.  Writing 1 is invalid.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IC0 | BUS RSTC | RSU SPDC | 1 | SHORTC | DMA EDC | SET RQC | CLR RQC | EP HALTC | 0020003CH | FFH |

| Bit position | Bit name | Function |
|---|---|---|
| 7 | BUSRSTC | This bit clears the Bus Reset interrupt.<br>　0: Clear |
| 6 | RSUSPDC | This bit clears the Resume/Suspend interrupt.<br>　0: Clear |
| 4 | SHORTC | This bit clears the Short interrupt.<br>　0: Clear |
| 3 | DMAEDC | This bit clears the DMA_END interrupt.<br>　0: Clear |
| 2 | SETRQC | This bit clears the SET_RQ interrupt.<br>　0: Clear |
| 1 | CLRRQC | This bit clears the CLR_RQ interrupt.<br>　0: Clear |
| 0 | EPHALTC | This bit clears the EP_Halt interrupt.<br>　0: Clear |

**(22) UF0 INT clear 1 register (UF0IC1)**

This register controls clearing the interrupt sources indicated by the UF0IS1 register.

This register is write-only, in 8-bit units.  If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register.  Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware.  Writing 0 to a bit of this register automatically sets the bit to 1.  Writing 1 is invalid.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IC1 | 1 | E0INC | E0 INDTC | E0ODTC | SUCESC | STGC | PROTC | CPU DECC | 0020003EH | FFH |

| Bit position | Bit name | Function |
|---|---|---|
| 6 | E0INC | This bit clears the EP0IN interrupt.<br>  0: Clear |
| 5 | E0INDTC | This bit clears the EP0INDT interrupt.<br>  0: Clear |
| 4 | E0ODTC | This bit clears the EP0OUTDT interrupt.<br>  0: Clear |
| 3 | SUCESC | This bit clears the Success interrupt.<br>  0: Clear |
| 2 | STGC | This bit clears the Stg interrupt.<br>  0: Clear |
| 1 | PROTC | This bit clears the Protect interrupt.<br>  0: Clear |
| 0 | CPUDECC | This bit clears the CPUDEC interrupt.<br>  0: Clear |

**(23)  UF0 INT clear 2 register (UF0IC2)**

This register controls clearing the interrupt sources indicated by the UF0IS2 register.

This register is write-only, in 8-bit units.  If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register.  Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware.  Writing 0 to a bit of this register automatically sets the bit to 1.  Writing 1 is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IC2 | BKI2INC | BKI2 DTC | BKI1INC | BKI1 DTC | 1 | 1 | 1 | IT1DTC | 00200040H | FFH |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 5 | BKInINC | These bits clear the BLKInIN interrupt.<br>  0: Clear |
| 6, 4 | BKInDTC | These bits clear the BLKInDT interrupt.<br>  0: Clear |
| 0 | IT1DTC | These bits clear the INTnDT interrupt.<br>  0: Clear |

**Remark**   n = 1, 2

**(24) UF0 INT clear 3 register (UF0IC3)**

This register controls clearing the interrupt sources indicated by the UF0IS3 register.

This register is write-only, in 8-bit units.  If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register.  Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware.  Writing 0 to a bit of this register automatically sets the bit to 1.  Writing 1 is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 2, 4) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IC3 | BKO2 FLC | BKO2 NLC | BKO2 NAKC | BKO2 DTC | BKO1 FLC | BKO1 NLC | BKO1 NAKC | BKO1 DTC | 00200042H | FFH |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 3 | BKOnFLC | These bits clear the BLKOnFL interrupt.<br>0: Clear |
| 6, 2 | BKOnNLC | These bits clear the BLKOnNL interrupt.<br>0: Clear |
| 5, 1 | BKOnNAKC | These bits clear the BLKOnNK interrupt.<br>0: Clear |
| 4, 0 | BKOnDTC | These bits clear the BLKOnDT interrupt.<br>0: Clear |

**Remark**   n = 1, 2

**(25) UF0 INT clear 4 register (UF0IC4)**

This register controls clearing the interrupt sources indicated by the UF0IS4 register.

This register is write-only, in 8-bit units.  If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register.  Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware.  Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IC4 | 1 | 1 | SETINTC | 1 | 1 | 1 | 1 | 1 | 00200044H | FFH |

| Bit position | Bit name | Function |
|---|---|---|
| 5 | SETINTC | This bit clears the SET_INT interrupt.<br>　0: Clear |

**(26) UF0 INT & DMARQ register (UF0IDR)**

This register selects reporting via an interrupt request or starting DMA.

This register can be read or written in 8-bit units.

If data exists in either the UF0BO1 or UF0BO1 register, or if data can be written to the UF0BI1 or UF0BI2 register, this register selects whether it is reported to the FW by an interrupt request, or whether starting DMA is requested.

If starting DMA is requested, the DMA transfer mode can be selected according to the setting of bits 0 and 1.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4) and the current setting of the interface.

Be sure to clear bits 3 and 2 to "0". If they are set to 1, the operation is not guaranteed.

**Caution   If the target endpoint is not supported by the SET_INTERFACE request under DMA transfer, the DMA request signal becomes inactive immediately, and the corresponding bit is automatically cleared to 0 by hardware.**

(1/2)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IDR | DQBI2 MS | DQBI1 MS | DQBO2 MS | DQBO1 MS | 0 | 0 | MODE1 | MODE0 | 0020004CH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 6 | DQBInMS | These bits enable (mask) a write DMA transfer request (DMA request signal for Endpoint m) to the UF0BIn register. When these bits are set to 1, the DMA request signal for Endpoint m becomes active while writing data can be acknowledged. If the DMA end signal for Endpoint m is input (if the DMA controller issues TC), these bits are automatically cleared to 0 by hardware. To continue DMA transfer, re-set these bits to 1 by FW.<br>1: Enables active DMA request signal for Endpoint m (masks BKInDT interrupt).<br>0: Disables active DMA request signal for Endpoint m (default value). |
| 5, 4 | DQBOnMS | These bits enable (mask) a read DMA transfer request (DMA request signal for Endpoint x) to the UF0BOn register. When these bits are set to 1, the DMA request signal for Endpoint x becomes active if the data to be read is prepared in the UF0BOn register. If the DMA end signal for Endpoint x is input (if the DMA controller issues TC), these bits are automatically cleared to 0 by hardware. They are also cleared to 0 when the USBSPxB signal is active. To continue DMA transfer, re-set these bits to 1 by FW.<br>1: Enables active DMA request signal for Endpoint x (masks BKOnDT interrupt).<br>0: Disables active DMA request signal for Endpoint x (default value). |

**Remark**   n = 1, 2

m = 1 and x = 2 where n = 1

m = 3 and x = 4 where n = 2

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 1, 0 | MODE1, MODE0 | These bits select the DMA transfer mode.<br><br>{table below} |

| MODE1 | MODE0 | Mode | Remark |
|---|---|---|---|
| 1 | 0 | Demand mode | DMA request signal becomes active as long as there is data.  It becomes inactive if there is no more data. |
| Other than above | | | Setting prohibited |

**(27) UF0 DMA status 0 register (UF0DMS0)**

This register indicates the DMA status of Endpoint1 to Endpoint4.

This register is read-only, in 8-bit units.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4) and the current setting of the interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0DMS0 | 0 | 0 | DQE4 | DQE3 | DQE2 | DQE1 | 0 | 0 | 0020004EH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 5 | DQE4 | This bit indicates that a DMA read request is being issued from Endpoint4 to memory.<br>   1: DMA read request from Endpoint4 is being issued.<br>   0: DMA read request from Endpoint4 is not being issued (default value). |
| 4 | DQE3 | This bit indicates that a DMA write request is being issued from memory to Endpoint3.<br>Note that, even if data is in Endpoint3 (when the FIFO is not full and after the BKI2DED bit has been set to 1), the DMA request signal becomes active immediately and DMA transfer is started when the DQBI2MS bit of the UF0IDR register is set to 1.<br>   1: DMA write request for Endpoint3 is being issued.<br>   0: DMA write request for Endpoint3 is not being issued (default value). |
| 3 | DQE2 | This bit indicates that a DMA read request is being issued from Endpoint2 to memory.<br>   1: DMA read request from Endpoint2 is being issued.<br>   0: DMA read request from Endpoint2 is not being issued (default value). |
| 2 | DQE1 | This bit indicates that a DMA write request is being issued from memory to Endpoint1.<br>Note that, even if data is in Endpoint1 (when the FIFO is not full and after the BKI1DED bit has been set to 1), the DMA request signal becomes active immediately and DMA transfer is started when the DQBI1MS bit of the UF0IDR register is set to 1.<br>   1: DMA write request for Endpoint1 is being issued.<br>   0: DMA write request for Endpoint1 is not being issued (default value). |

**(28) UF0 DMA status 1 register (UF0DMS1)**

This register indicates the DMA status of Endpoint1 to Endpoint4.

This register is read-only, in 8-bit units.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4) and the current setting of the interface.

Each bit is automatically cleared to 0 when this register is read. Even when this register is read, however, bits 4 and 3 of the UF0IS0 register are not cleared to 0. If the target endpoint is no longer supported by the SET_INTERFACE request, each bit is automatically cleared to 0 by hardware (however, the DMA_END interrupt request and Short interrupt request are not cleared).

|         | 7     | 6     | 5     | 4     | 3     | 2     | 1   | 0   | Address   | After reset |
|---------|-------|-------|-------|-------|-------|-------|-----|-----|-----------|-------------|
| UF0DMS1 | DEDE4 | DSPE4 | DEDE3 | DEDE2 | DSPE2 | DEDE1 | 0   | 0   | 00200050H | 00H         |

| Bit position | Bit name | Function |
|--------------|----------|----------|
| 7, 5, 4, 2 | DEDEn | These bits indicate that the DMA end (TC) signal for Endpoint n becomes active and DMA is stopped while a DMA read request is being issued from Endpoint n to memory.<br>1: DMA end signal for Endpoint n is active.<br>0: DMA end signal for Endpoint n is inactive (default value). |
| 6, 3 | DSPEm | These bits indicate that, although a DMA read request was being issued from Endpoint m to memory, DMA has been stopped because the received data is a short packet and there is no more data to be transferred.<br>1: DMASTOP_EPm signal is active.<br>0: DMASTOP_EPm signal is inactive (default value). |

**Remark**  n = 1 to 4
          m = 2, 4

**(29) UF0 FIFO clear 0 register (UF0FIC0)**

This register clears each FIFO.

This register is write-only, in 8-bit units.  If this register is read, 00H is read.

FW can clear the target FIFO by writing 1 to the corresponding bit of this register.  The bit to which 1 has been written is automatically cleared to 0.  Writing 0 to the bit is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7) and the current setting of the interface.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0FIC0 | BKI2SC | BKI2CC | BKI1SC | BKI1CC | 0 | ITR1C | EP0WC | EP0RC | 00200060H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 5 | BKInSC | These bits clear only the FIFO on the SIE side of the UF0BIn register (reset the counter).<br>    1: Clear<br>Writing these bits is invalid while an IN token for Endpoint m is being processed with the BKInNK bit set to 1.<br>The BKInNK bit is automatically cleared to 0 by clearing the FIFO.  Make sure that the FIFO on the CPU side is empty when these bits are used. |
| 6, 4 | BKInCC | These bits clear only the FIFO on the CPU side of the UF0BIn register (reset the counter).<br>    1: Clear |
| 2 | ITR1C | These bits clear the UF0INT1 register (reset the counter).<br>    1: Clear<br>Writing these bits is invalid while an IN token for Endpoint 7 is being processed with the IT1NK bit set to 1.<br>The IT1NK bit is automatically cleared to 0 by clearing the FIFO. |
| 1 | EP0WC | This bit clears the UF0E0W register (resets the counter).<br>    1: Clear<br>Writing this bit is invalid while an IN token for Endpoint0 is being processed with the EP0NKW bit set to 1.<br>The EP0NKW bit is automatically cleared to 0 by clearing the FIFO. |
| 0 | EP0RC | This bit clears the UF0E0R register (resets the counter).<br>    1: Clear<br>When the EP0NKR bit is set to 1 (except when it has been set by FW), the EP0NKR bit is automatically cleared to 0 by clearing the FIFO. |

**Remark**   n = 1, 2

m = 1 when n = 1

m = 3 when n = 2

**(30) UF0 FIFO clear 1 register (UF0FIC1)**

This register clears each FIFO.

This register is write-only, in 8-bit units.  If this register is read, 00H is read.

FW can clear the target FIFO by writing 1 to the corresponding bit of this register.  The bit to which 1 has been written is automatically cleared to 0.  Writing 0 to the bit is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 2, 4) and the current setting of the interface.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0FIC1 | 0 | 0 | 0 | 0 | BKO2C | BKO2CC | BKO1C | BKO1CC | 00200062H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 3, 1 | BKOnC | These bits clear the FIFOs on both the SIE and CPU sides of the UF0BOn register (reset the counter).<br>　1: Clear<br>When the BKOnNK bit is set to 1 (except when it has been set by FW), the BKOnNK bit is automatically cleared to 0 by clearing the FIFO. |
| 2, 0 | BKOnCC | These bits clear only the FIFO on the CPU side of the UF0BOn register (reset the counter).<br>　1: Clear<br>When the BKOnNK bit is set to 1 (except when it has been set by FW), the BKOnNK bit is automatically cleared to 0 by clearing the FIFO. |

**Remark**　n = 1, 2

**(31)  UF0 data end register (UF0DEND)**

This register reports the end of writing to the transmission system.

This register is write-only, in 8-bit units (however, bits 7 and 6 can be read and written).  If this register is read, 00H is read.

FW can start data transfer of the target endpoint by writing 1 to the corresponding bit of this register.  The bit to which 1 has been written is automatically cleared to 0.  Writing 0 to the bit is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7) and the current setting of the interface.

(1/2)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0DEND | BKI2T | BKI1T | 0 | 0 | IT1DEND | BKI2DED | BKI1DED | E0DED | 0020006AH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 6 | BKInT | These bits specify whether toggling the FIFO is automatically executed if the FIFO on the CPU side of the UF0BIn register becomes full as a result of DMA.<br>1: Automatically execute a toggle operation of the FIFO as soon as the FIFO has become full.<br>0: Do not automatically execute a toggle operation of the FIFO even if the FIFO becomes full (default value). |
| 3 | IT1DEND | Set these bits to 1 to transmit the data of the UF1INT1 register.  When these bits are set to 1, the IT1NK bit is set to 1 and data transfer is executed.<br>1: Transmit a short packet.<br>0: Do not transmit a short packet (default value).<br>If the ITR1C bit of the UF0FIC0 register is set to 1 and then these bits are set to 1 (counter of UF0INT1 register = 0 and the corresponding bit of the UF0EPS0 register = 1), a Null packet (with a data length of 0) is transmitted.<br>If data exists in the UF0INT1 register and if these bits are set to 1 (counter of UF0INT1 register ≠ 0 and the corresponding bit of the UF0EPS0 register = 1), a short packet is transmitted.<br>These bits are automatically controlled by hardware when the FIFO is full. |

**Remark**   n = 1, 2

| Bit position | Bit name | Function |
|---|---|---|
| 2, 1 | BKInDED | Set these bits to 1 when writing transmit data to the UF0BIn register has been completed. When these bits are set to 1, the FIFO is toggled as soon as possible, the BKInNK bit is set to 1, and data is transferred.<br>    1: Transmit a short packet.<br>    0: Do not transmit a short packet (default value).<br>These bits control the FIFO on the CPU side.<br>If the BKInCC bit of the UF0FIC0 register is set to 1 and then these bits are set to 1 (counter of UF0BIn register = 0), a Null packet (with a data length of 0) is transmitted.<br>If data exists in the UF0BIn register and if these bits are set to 1 (counter of UF0BIn register ≠ 0), and if the FIFO is not full, a short packet is transmitted.<br>If the FIFO on the CPU side of the UF0BIn register becomes full as a result of DMA, with the PIO or BKInT bit set to 1, the hardware starts data transmission even if these bits are not set to 1.<br>If the FIFO on the CPU side of the UF0BIn register becomes full as a result of DMA, with the BKInT bit cleared to 0, be sure to set these bits to 1 (see **20.6.3 (3)  UF0 EPNAK register (UF0EN)**). |
| 0 | E0DED | Set this bit to 1 to transmit data of the UF0E0W register.  When this bit is set to 1, the EP0NKW bit is set to 1 and data is transferred.<br>    1: Transmit a short packet.<br>    0: Do not transmit a short packet (default value).<br>If the EP0WC bit of the UF0FIC0 register is set to 1 and if this bit is set to 1 (counter of UF0E0W register = 0 and bit 1 of UF0EPS0 register = 1), a Null packet (with a data length of 0) is transmitted.<br>If data exists in the UF0E0W register and if this bit is set to 1 (counter of UF0E0W register ≠ 0 and bit 1 of the UF0EPS0 register = 1), and if the FIFO is not full, a short packet is transmitted. |

**Remark**   n = 1, 2

**(32) UF0 GPR register (UF0GPR)**

This register controls USBF and the USB interface.

This register is write-only, in 8-bit units.  If this register is read, 00H is read.  Be sure to clear bits 7 to 1 to "0".

FW can reset the USBF by writing 1 to bit 0 of this register.  This bit is automatically cleared to 0 after 1 has been written to it.  Writing 0 to this bit is invalid.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0GPR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | MRST | 0020006EH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 0 | MRST | Set this bit to 1 to reset USBF.<br>　1: Reset<br>Actually, USBF is reset two USB clocks after this bit has been set to 1 by FW and the write signal has become inactive.<br>Resetting USBF by the MRST bit while the system clock is operating has the same result as resetting by the $\overline{\text{RESET}}$ pin (hardware reset) (register value back to default value). |

**(33) UF0 mode control register (UF0MODC)**

This register controls CPUDEC processing.

This register can be read or written in 8-bit units.

By setting each bit of this register, the setting of the UF0MODS register can be changed.  The bit of this register is automatically cleared to 0 only at hardware reset and when the MRST bit of the UF0GRP register has been set to 1.

Even if the bit of this register has automatically been set to 1 by hardware, the setting by FW takes precedence.

Be sure to clear bits 7 and 5 to 2 to "0".  If they are set to 1, the operation is not guaranteed.

**Caution   This register is provided for debugging purposes.  Usually, do not set this register except for verifying the operation or when a special mode is used.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0MODC | 0 | CDC GDST | 0 | 0 | 0 | 0 | 0 | 0 | 00200074H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 6 | CDCGDST | Set this bit to 1 to switch the GET_DESCRIPTOR Configuration request to CPUDEC processing.  By setting this bit to 1, the CDCGD bit of the UF0MODS register can be forcibly set to 1.<br>1: Forcibly change the GET_DESCRIPTOR Configuration request to CPUDEC processing (sets the CDCGD bit of the UF0MODS register to 1).<br>0: Automatically process the GET_DESCRIPTOR Configuration request (default value). |

**(34)  UF0 mode status register (UF0MODS)**

This register indicates the configuration status.

This register is read-only, in 8-bit units.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0MODS | 0 | CDCGD | 0 | MPACK | DFLT | CONF | 0 | 0 | 00200078H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 6 | CDCGD | This bit specifies whether CPUDEC processing is performed for the GET_DESCRIPTOR Configuration request.<br>  1: Forcibly change the GET_DESCRIPTOR Configuration request to CPUDEC processing.<br>  0: Automatically process the GET_DESCRIPTOR Configuration request (default value). |
| 4 | MPACK | This bit indicates the transmit packet size of Endpoint0.<br>  1: Transmit a packet of other than 8 bytes.<br>  0: Transmit a packet of 8 bytes (default value).<br>This bit is automatically set to 1 by hardware after the GET_DESCRIPTOR Device request has been processed (on normal completion of the status stage).  It is not cleared to 0 until the USBF has been reset (it is not cleared to 0 by Bus Reset).<br>If this bit is not set to 1, the hardware transfers only the automatically-executed request in 8-byte units.  Therefore, even if data of more than 8 bytes is sent by the OUT token to be processed by FW before completion of the GET_DESCRIPTOR Device request, the data is correctly received.<br>This bit is ignored if the size of Endpoint0 is 8 bytes. |
| 3 | DFLT | This bit indicates the default status (DFLT bit = 1).<br>  1: Enables response.<br>  0: Disables response (always no response) (default value).<br>This bit is automatically set to 1 by Bus Reset.  The transaction for all the endpoints is not responded to until this bit is set to 1. |
| 2 | CONF | This bit indicates whether the SET_CONFIGURATION request has been completed.<br>  1: SET_CONFIGURATION request has been completed.<br>  0: SET_CONFIGURATION request has not been completed (default value).<br>This bit is set to 1 when Configuration value = 1 is received by the SET_CONFIGURATION request.<br>Unless this bit is set to 1, access to an endpoint other than Endpoint0 is ignored.<br>This bit is cleared to 0 when Configuration value = 0 is received by the SET_CONFIGURATION request.  It is also cleared to 0 when Bus Reset is detected. |

**(35) UF0 active interface number register (UF0AIFN)**

This register sets the valid Interface number that correctly responds to the GET/SET_INTERFACE request. Because Interface 0 is always valid, Interfaces 1 to 4 can be selected.

This register can be read or written in 8-bit units.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0AIFN | ADDIF | 0 | 0 | 0 | 0 | 0 | IFNO1 | IFNO0 | 00200080H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 | ADDIF | This bit allows use of Interfaces numbered other than 0.<br>  1: Support up to the Interface number specified by the IFNO1 and IFNO0 bits.<br>  0: Support only Interface 0 (default value).<br>Setting bits 1 and 0 of this register is invalid when this bit is not set to 1. |
| 1, 0 | IFNO1, IFNO0 | These bits specify the range of Interface numbers to be supported.<br><br>|

| IFNO1 | IFNO0 | Valid Interface No. |
|---|---|---|
| 1 | 1 | 0, 1, 2, 3, 4 |
| 1 | 0 | 0, 1, 2, 3 |
| 0 | 1 | 0, 1, 2 |
| 0 | 0 | 0, 1 |

**(36)  UF0 active alternative setting register (UF0AAS)**

This register specifies a link between the Interface number and Alternative Setting.

This register can be read or written in 8-bit units.

USBF of the V850ES/JG3-U and V850ES/JH3-U can set a five-series Alternative Setting (Alternate Setting 0, 1, 2, 3, and 4 can be defined) and a two-series Alternative Setting (Alternative Setting 0 and 1 can be defined) for one Interface.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0AAS | ALT2 | IFAL21 | IFAL20 | ALT2EN | ALT5 | IFAL51 | IFAL50 | ALT5EN | 00200082H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7, 3 | ALTn | These bits specify whether an n-series Alternative Setting is linked with Interface 0.  When these bits are set to 1, the setting of the IFALn1 and IFALn0 bits is invalid.<br>  1: Link n-series Alternative Setting with Interface 0.<br>  0: Do not link n-series Alternative Setting with Interface 0 (default value). |
| 6, 5, 2, 1 | IFALn1, IFALn0 | These bits specify the Interface number to be linked with the n-series Alternative Setting. If the linked Interface number is outside the range specified by the UF0AIFN register, the n-series Alternative Setting is invalid (ALTnEN bit = 0).<br><br><table><tr><td>IFALn1</td><td>IFALn0</td><td>Interface number to be linked</td></tr><tr><td>1</td><td>1</td><td>Links Interface 4.</td></tr><tr><td>1</td><td>0</td><td>Links Interface 3.</td></tr><tr><td>0</td><td>1</td><td>Links Interface 2.</td></tr><tr><td>0</td><td>0</td><td>Links Interface 1.</td></tr></table><br>Do not link a five-series Alternative Setting and a two-series Alternative Setting with the same Interface number. |
| 4, 0 | ALTnEN | These bits validate the n-series Alternative Setting.  Unless these bits are set to 1, the setting of the ALTn, IFALn1, and IFALn0 bits is invalid.<br>  1: Validate the n-series Alternative Setting.<br>  0: Do not validate the n-series Alternative Setting (default value). |

**Remark**  n = 2, 5

For example, when the UF0AIFN register is set to 82H and the UF0AAS register is set to 15H, Interfaces 0, 1, 2, and 3 are valid.  Interfaces 0 and 2 support only Alternative Setting 0.  Interface 1 supports Alternative Setting 0 and 1, and Interface 3 supports Alternative Setting 0, 1, 2, 3, and 4.  With this setting, requests GET_INTERFACE wIndex = 0/1/2/3, SET_INTERFACE wValue = 0 & wIndex = 0/2, SET_INTERFACE wValue = 0/1 & wIndex = 1, and SET_INTERFACE wValue = 0/1/2/3/4 & wIndex = 3 are automatically responded to, and a STALL response is made to the other GET/SET_INTERFACE requests.

**(37) UF0 alternative setting status register (UF0ASS)**

This register indicates the current status of the Alternative Setting.

This register is read-only, in 8-bit units.

Check this register when the SET_INT interrupt request has been issued.  The value received by the SET_INTERFACE request is reflected on the UF0IFn register (n = 0 to 4) as well as on this register.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0ASS | 0 | 0 | 0 | 0 | AL5ST3 | AL5ST2 | AL5ST1 | AL2ST | 00200084H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 3 to 1 | AL5ST3 to AL5ST1 | These bits indicate the current status of the five-series Alternative Setting.<br><br>{table below} |
| 0 | AL2ST | This bit indicates the current status of the two-series Alternative Setting (selected Alternative Setting number).<br>1: Alternative Setting 1<br>0: Alternative Setting 0 |

| AL5ST3 | AL5ST2 | AL5ST1 | Selected Alternative Setting number |
|---|---|---|---|
| 1 | 0 | 0 | Alternative Setting 4 |
| 0 | 1 | 1 | Alternative Setting 3 |
| 0 | 1 | 0 | Alternative Setting 2 |
| 0 | 0 | 1 | Alternative Setting 1 |
| 0 | 0 | 0 | Alternative Setting 0 |

**(38) UF0 endpoint 1 interface mapping register (UF0E1IM)**

This register specifies for which Interface and Alternative Setting Endpoint1 is valid.

This register can be read or written in 8-bit units.

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint1 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint1 request and the IN transaction to Endpoint1 are responded to, and whether the related bits are valid or invalid.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E1IM | E1EN2 | E1EN1 | E1EN0 | E12AL1 | E15AL4 | E15AL3 | E15AL2 | E15AL1 | 00200086H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 5 | E1EN2 to E1EN0 | These bits set a link between the Interface of Endpoint1 and the two-/five-series Alternative Setting.  The endpoint is linked with Alternative Setting 0.  The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4.<br><br><table><tr><td>E1EN2</td><td>E1EN1</td><td>E1EN0</td><td>Link status</td></tr><tr><td>1</td><td>1</td><td>1</td><td>Not linked with Interface</td></tr><tr><td>1</td><td>1</td><td>0</td><td></td></tr><tr><td>1</td><td>0</td><td>1</td><td>Linked with Interface 4 and Alternative Setting 0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>Linked with Interface 3 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>Linked with Interface 2 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>Linked with Interface 1 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>Linked with Interface 0 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>Not linked with Interface (default value)</td></tr></table><br>When these bits are set to 110 or 111, they are invalid even if the E12AL1 bit is cleared to 0.<br>If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint1 is valid. |
| 4 | E12AL1 | This bit validates Endpoint1 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1.<br>  1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1.<br>  0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value).<br>This bit is valid when the E15AL4 to E15AL1 bits are 0000. |
| 3 to 0 | E15ALn | These bits validate Endpoint1 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n.<br>  1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1.<br>  0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value). |

**Remark**   n = 1 to 4

**(39)  UF0 endpoint 2 interface mapping register (UF0E2IM)**

This register specifies for which Interface and Alternative Setting Endpoint2 is valid.

This register can be read or written in 8-bit units.

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint2 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint2 request and the OUT transaction to Endpoint2 are responded to, and whether the related bits are valid or invalid.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E2IM | E2EN2 | E2EN1 | E2EN0 | E22AL1 | E25AL4 | E25AL3 | E25AL2 | E25AL1 | 00200088H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 5 | E2EN2 to E2EN0 | These bits set a link between the Interface of Endpoint2 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4. <br><br> <table><tr><th>E2EN2</th><th>E2EN1</th><th>E2EN0</th><th>Link status</th></tr><tr><td>1</td><td>1</td><td>1</td><td>Not linked with Interface</td></tr><tr><td>1</td><td>1</td><td>0</td><td></td></tr><tr><td>1</td><td>0</td><td>1</td><td>Linked with Interface 4 and Alternative Setting 0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>Linked with Interface 3 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>Linked with Interface 2 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>Linked with Interface 1 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>Linked with Interface 0 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>Not linked with Interface (default value)</td></tr></table> <br> When these bits are set to 110 or 111, they are invalid even if the E22AL1 bit is cleared to 0. <br> If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint2 is valid. |
| 4 | E22AL1 | This bit validates Endpoint2 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1. <br>   1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1. <br>   0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value). <br> This bit is valid when the E25AL4 to E25AL1 bits are 0000. |
| 3 to 0 | E25ALn | These bits validate Endpoint2 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n. <br>   1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1. <br>   0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value). |

**Remark**   n = 1 to 4

**(40) UF0 endpoint 3 interface mapping register (UF0E3IM)**

This register specifies for which Interface and Alternative Setting Endpoint3 is valid.

This register can be read or written in 8-bit units.

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint3 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint3 request and the IN transaction to Endpoint3 are responded to, and whether the related bits are valid or invalid.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E3IM | E3EN2 | E3EN1 | E3EN0 | E32AL1 | E35AL4 | E35AL3 | E35AL2 | E35AL1 | 0020008AH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 5 | E3EN2 to E3EN0 | These bits set a link between the Interface of Endpoint3 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4. <table><tr><td>E3EN2</td><td>E3EN1</td><td>E3EN0</td><td>Link status</td></tr><tr><td>1</td><td>1</td><td>1</td><td rowspan="2">Not linked with Interface</td></tr><tr><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>Linked with Interface 4 and Alternative Setting 0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>Linked with Interface 3 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>Linked with Interface 2 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>Linked with Interface 1 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>Linked with Interface 0 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>Not linked with Interface (default value)</td></tr></table> When these bits are set to 110 or 111, they are invalid even if the E32AL1 bit is cleared to 0. If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint3 is valid. |
| 4 | E32AL1 | This bit validates Endpoint3 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1. 1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1. 0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value). This bit is valid when the E35AL4 to E35AL1 bits are 0000. |
| 3 to 0 | E35ALn | These bits validate Endpoint3 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n. 1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1. 0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value). |

**Remark**   n = 1 to 4

**(41) UF0 endpoint 4 interface mapping register (UF0E4IM)**

This register specifies for which Interface and Alternative Setting Endpoint4 is valid.

This register can be read or written in 8-bit units.

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint4 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint4 request and the OUT transaction to Endpoint4 are responded to, and whether the related bits are valid or invalid.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E4IM | E4EN2 | E4EN1 | E4EN0 | E42AL1 | E45AL4 | E45AL3 | E45AL2 | E45AL1 | 0020008CH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 5 | E4EN2 to E4EN0 | These bits set a link between the Interface of Endpoint4 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4. <br><br> <table><tr><td>E4EN2</td><td>E4EN1</td><td>E4EN0</td><td>Link status</td></tr><tr><td>1</td><td>1</td><td>1</td><td rowspan=2>Not linked with Interface</td></tr><tr><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>Linked with Interface 4 and Alternative Setting 0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>Linked with Interface 3 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>Linked with Interface 2 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>Linked with Interface 1 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>Linked with Interface 0 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>Not linked with Interface (default value)</td></tr></table> <br> When these bits are set to 110 or 111, they are invalid even if the E42AL1 bit is cleared to 0. <br> If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint4 is valid. |
| 4 | E42AL1 | This bit validates Endpoint4 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1. <br>   1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1. <br>   0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value). <br> This bit is valid when the E45AL4 to E45AL1 bits are 0000. |
| 3 to 0 | E45ALn | These bits validate Endpoint4 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n. <br>   1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1. <br>   0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value). |

**Remark**   n = 1 to 4

**(42)  UF0 endpoint 7 interface mapping register (UF0E7IM)**

This register specifies for which Interface and Alternative Setting Endpoint7 is valid.

This register can be read or written in 8-bit units.

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint7 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint7 request and the IN transaction to Endpoint7 are responded to, and whether the related bits are valid or invalid.

|        | 7     | 6     | 5     | 4      | 3      | 2      | 1      | 0      | Address    | After reset |
|--------|-------|-------|-------|--------|--------|--------|--------|--------|------------|-------------|
| UF0E7IM | E7EN2 | E7EN1 | E7EN0 | E72AL1 | E75AL4 | E75AL3 | E75AL2 | E75AL1 | 00200092H  | 00H         |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 5 | E7EN2 to E7EN0 | These bits set a link between the Interface of Endpoint7 and the two-/five-series Alternative Setting.  The endpoint is linked with Alternative Setting 0.  The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4. <br><br> <table><tr><td>E7EN2</td><td>E7EN1</td><td>E7EN0</td><td>Link status</td></tr><tr><td>1</td><td>1</td><td>1</td><td>Not linked with Interface</td></tr><tr><td>1</td><td>1</td><td>0</td><td></td></tr><tr><td>1</td><td>0</td><td>1</td><td>Linked with Interface 4 and Alternative Setting 0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>Linked with Interface 3 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>Linked with Interface 2 and Alternative Setting 0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>Linked with Interface 1 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>Linked with Interface 0 and Alternative Setting 0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>Not linked with Interface (default value)</td></tr></table> <br> When these bits are set to 110 or 111, they are invalid even if the E72AL1 bit is cleared to 0. <br> If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint7 is valid. |
| 4 | E72AL1 | This bit validates Endpoint7 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1. <br>   1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1. <br>   0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value). <br> This bit is valid when the E75AL4 to E75AL1 bits are 0000. |
| 3 to 0 | E75ALn | These bits validate Endpoint7 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n. <br>   1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1. <br>   0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value). |

**Remark**    n = 1 to 4

**20.6.4  Data hold registers**

**(1)  UF0 EP0 read register (UF0E0R)**

The UF0E0R register is a 64-byte FIFO that stores the OUT data sent from the host in the data stage of control transfer to/from Endpoint0.

This register is read-only, in 8-bit units.  A write access to this register is ignored.

The hardware automatically transfers data to the UF0E0R register when it has received the data from the host. When the data has been correctly received, the E0ODT bit of the UF0IS1 register is set to 1.  The UF0E0L register holds the quantity of the received data, and an interrupt request (INTUSBF0) is issued.  The UF0E0L register always updates the length of the received data while it is receiving data.  If the final transfer is correct reception, the interrupt request is generated.  If the reception is abnormal, the UF0E0L register is cleared to 0 and the interrupt request is not generated.

The data held by the UF0E0R register must be read by FW up to the value of the amount of data read by the UF0E0L register.  Check that all data has been read by using the EP0R bit of the UF0EPS0 register (EP0R bit = 0 when all data has been read).  If the value of the UF0E0L register is 0, the EP0NKR bit of the UF0E0N register is cleared to 0, and the UF0E0R register is ready for reception.  The UF0E0R register is cleared when the next SETUP token has been received.

**Caution   Read all the data stored.  Clear the FIFO to discard some data.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E0R | E0R7 | E0R6 | E0R5 | E0R4 | E0R3 | E0R2 | E0R1 | E0R0 | 00200100H | Undefined |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | E0R7 to E0R0 | These bits store the OUT data sent from the host in the data stage of control transfer to/from Endpoint0. |

The operation of the UF0E0R register is illustrated below.

**Figure 20-4.  Operation of UF0E0R Register**
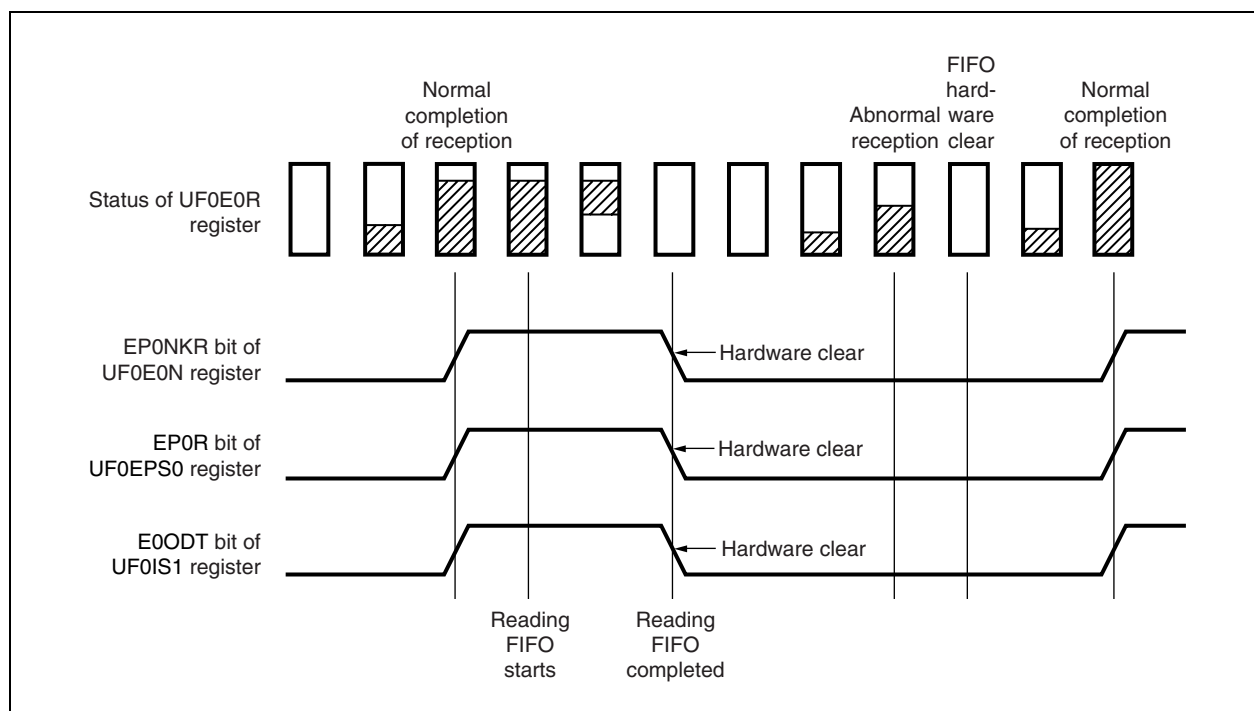


**(2)  UF0 EP0 length register (UF0E0L)**

The UF0E0L register stores the data length held by the UF0E0R register.

This register is read-only, in 8-bit units.  A write access to this register is ignored.

The UF0E0L register always updates the length of the received data while it is receiving data.  If the final transfer is abnormal reception, the UF0E0L register is cleared to 0 and the interrupt request is not generated.  The interrupt request is generated only when the reception is normal, and the FW can read as many data from the UF0E0R register as the value read from the UF0E0L register.  The value of the UF0E0L register is decremented each time the UF0E0R register has been read.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E0L | E0L7 | E0L6 | E0L5 | E0L4 | E0L3 | E0L2 | E0L1 | E0L0 | 00200102H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | E0L7 to E0L0 | These bits store the data length held by the UF0E0R register. |

**(3) UF0 EP0 setup register (UF0E0ST)**

The UF0E0ST register holds the SETUP data sent from the host.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The UF0E0ST register always writes data when a SETUP transaction has been received. The hardware sets the PROT bit of the UF0IS1 register when it has correctly received the SETUP transaction. It sets the CPUDEC bit of the UF0IS1 register in the case of an FW-processed request. Then an interrupt request (INTUSBF0) is issued. In the case of an FW-processed request, be sure to read the request in 8-byte units. If it is not read in 8-byte units, the subsequent requests cannot be correctly decoded. The read counter of the UF0E0ST register is not cleared even when Bus Reset is received. Always read this counter in 8-byte units regardless of whether Bus Reset is received or not.

Because the UF0E0ST register always enables writing, the hardware overwrites data to this register even if a SETUP transaction is received while the data of the register is being read. Even if the SETUP transaction cannot be correctly received, the CPUDEC interrupt request and Protect interrupt request are not generated, but the previous data is discarded. If a SETUP token of less than 8 bytes is received, however, the received SETUP token is discarded, and the previously received SETUP data is retained. If the SETUP token is received more than once when control transfer is executed once, be sure to check the PROT bit of the UF0IS1 register under the conditions below. If PROT bit = 1, read the UF0E0ST register again because the SETUP transaction has been received more than once.

<1>  If a request is decoded by FW and the UF0E0R register is read or the UF0E0W register is written

<2>  When preparing for a STALL response for the request to which the decode result does not correspond

**Caution**   **Be sure to read all the stored data. The UF0E0ST register is always updated by the request in the SETUP transaction.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E0ST | E0S7 | E0S6 | E0S5 | E0S4 | E0S3 | E0S2 | E0S1 | E0S0 | 00200104H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | E0S7 to E0S0 | These bits hold the SETUP data sent from the host. |

The operation of the UF0E0ST register is illustrated below.

**Figure 20-5. Operation of UF0E0ST Register**
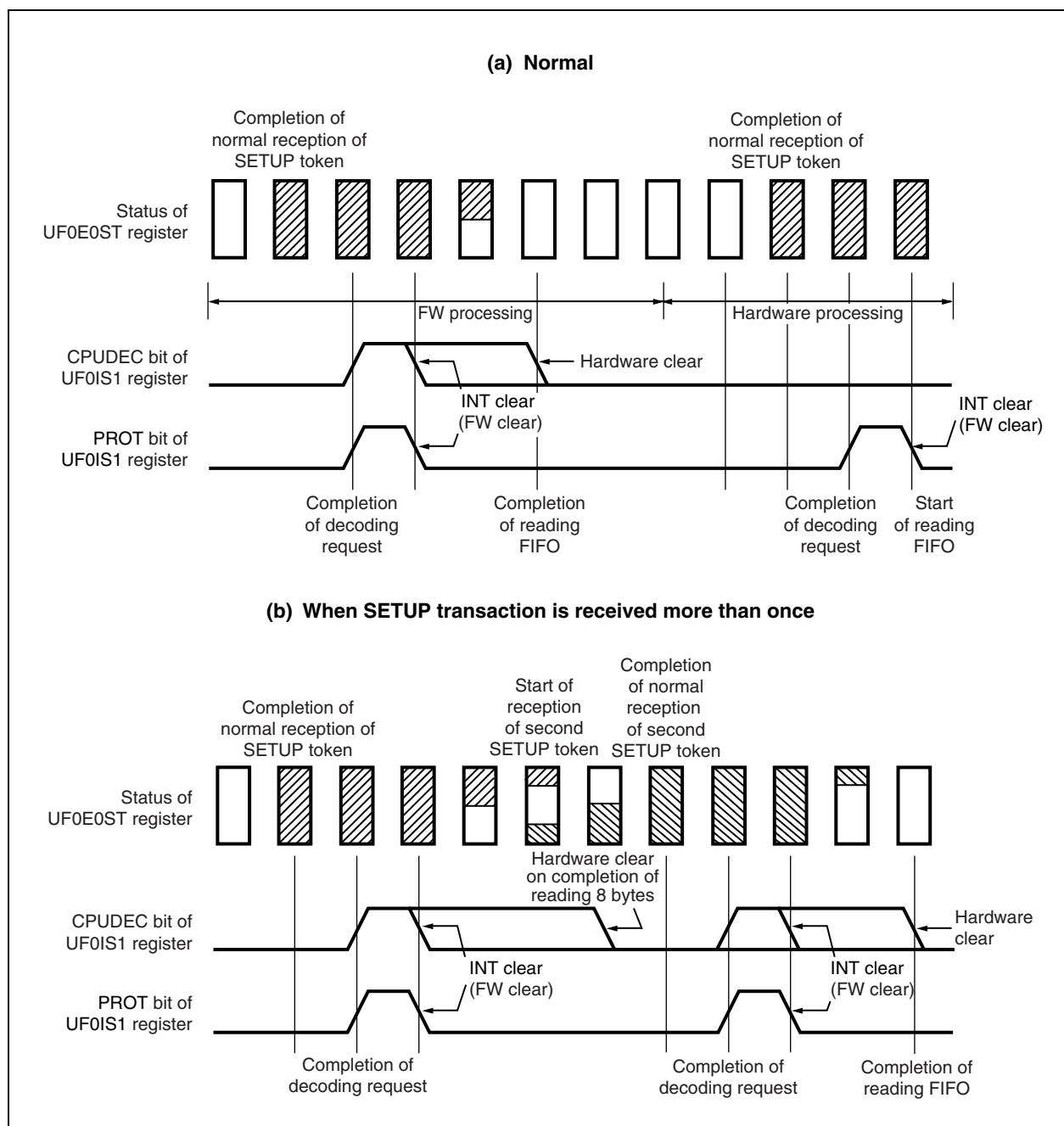
**(4) UF0 EP0 write register (UF0E0W)**

The UF0E0W register is a 64-byte FIFO that stores the IN data (passes it to SIE) sent to the host in the data stage to Endpoint0.

This register is write-only, in 8-bit units. When this register is read, 00H is read.

The hardware transmits data to the USB bus in synchronization with an IN token only when the EP0NKW bit of the UF0E0N register is set to 1 (when NAK is not transmitted). When data is transmitted and when the host correctly receives the data, the EP0NKW bit of the UF0E0N register is automatically cleared to 0 by hardware. A short packet is transmitted when data is written to the UF0E0W register and the E0DED bit of the UF0DEND register is set to 1 (EP0W bit of the UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0E0W register is cleared and the E0DED bit of the UF0DEND register is set to 1 (EP0W bit of the UF0EPS0 register = 1 (data exists)).

The UF0E0W register is cleared to 0 when the next SETUP token is received while transmission has not been completed yet. If the stage of control transfer (read) changes to the status stage while ACK has not been correctly received in the data stage, the UF0E0W register is automatically cleared to 0. At the same time, it is also cleared to 0 if the EP0NKW bit of the UF0E0N register is 1.

If the UF0E0W register is read while no data is in it, 00H is read.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E0W | E0W7 | E0W6 | E0W5 | E0W4 | E0W3 | E0W2 | E0W1 | E0W0 | 00200106H | Undefined |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | E0W7 to E0W0 | These bits store the IN data sent to the host in the data stage to Endpoint0. |

The operation of the UF0E0W register is illustrated below.

**Figure 20-6.  Operation of UF0E0W Register**



(a)  16-byte transmission

(b)  When Null packet or short packet is transmitted
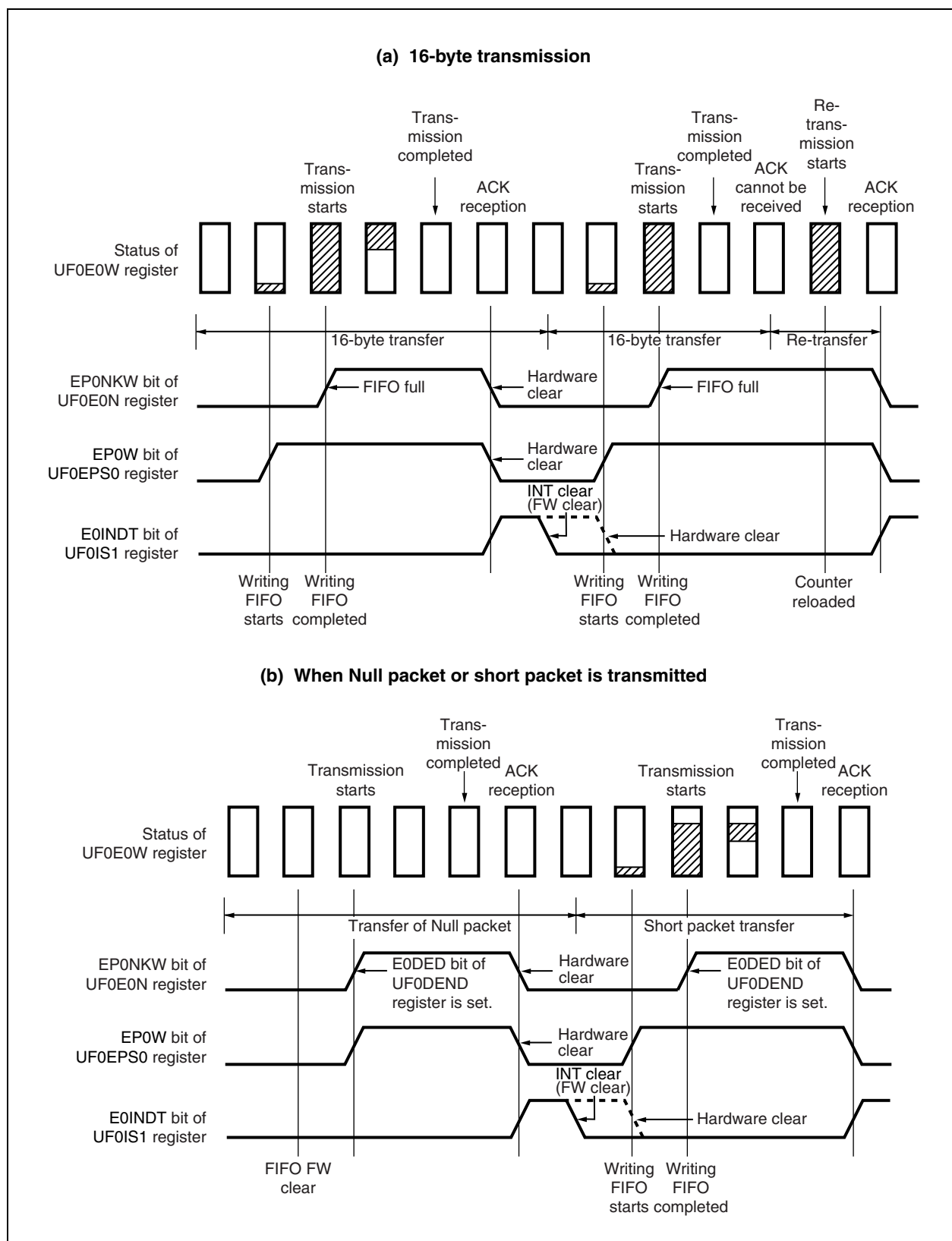
**(5)  UF0 bulk-out 1 register (UF0BO1)**

The UF0BO1 register is a 64-byte × 2 FIFO that stores data for Endpoint2.  This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides.  The toggle operation takes place when data is in the FIFO on the SIE side and when no data is in the FIFO on the CPU side (counter value = 0).

This register is read-only, in 8-bit units.  A write access to this register is ignored.

When the hardware receives data for Endpoint2 from the host, it automatically transfers the data to the UF0BO1 register.  When the register correctly receives the data, a FIFO toggle operation occurs.  As a result, the BKO1DT bit of the UF0IS3 register is set to 1, the quantity of the received data is held by the UF0BO1L register, and an interrupt request or DMA request is issued to the CPU.  Whether the interrupt request or DMA request is issued can be selected by using the DQBO1MS bit of the UF0IDR register.

Read the data held by the UF0BO1 register by FW, up to the value of the amount of data read by the UF0BO1L register.  When the correct received data is held by the FIFO connected to the SIE side and the value of the UF0BO1L register reaches 0, the toggle operation of the FIFO occurs, and the BKO1NK bit of the UF0EN register is automatically cleared to 0.  If data greater than the value of the UF0BO1L register is read and if the FIFO toggle condition is satisfied, the toggle operation of the FIFO occurs.  As a result, the next packet may be read by mistake.  Note that, if the toggle condition is not satisfied, the first data is repeatedly read.

If overrun data is received while data is held by the FIFO connected to the CPU side, Endpoint2 stalls, and the FIFO on the CPU side is cleared.

When the UF0BO1 register is read while no data is in it, an undefined value is read.

**Caution    Be sure to read all the data stored in this register.**

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0BO1 | BKO17 | BKO16 | BKO15 | BKO14 | BKO13 | BKO12 | BKO11 | BKO10 | 00200108H | Undefined |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | BKO17 to BKO10 | These bits store data for Endpoint2. |

The operation of the UF0BO1 register is illustrated below.

**Figure 20-7.  Operation of UF0BO1 Register (1/2)**

**Figure 20-7.  Operation of UF0BO1 Register (2/2)**
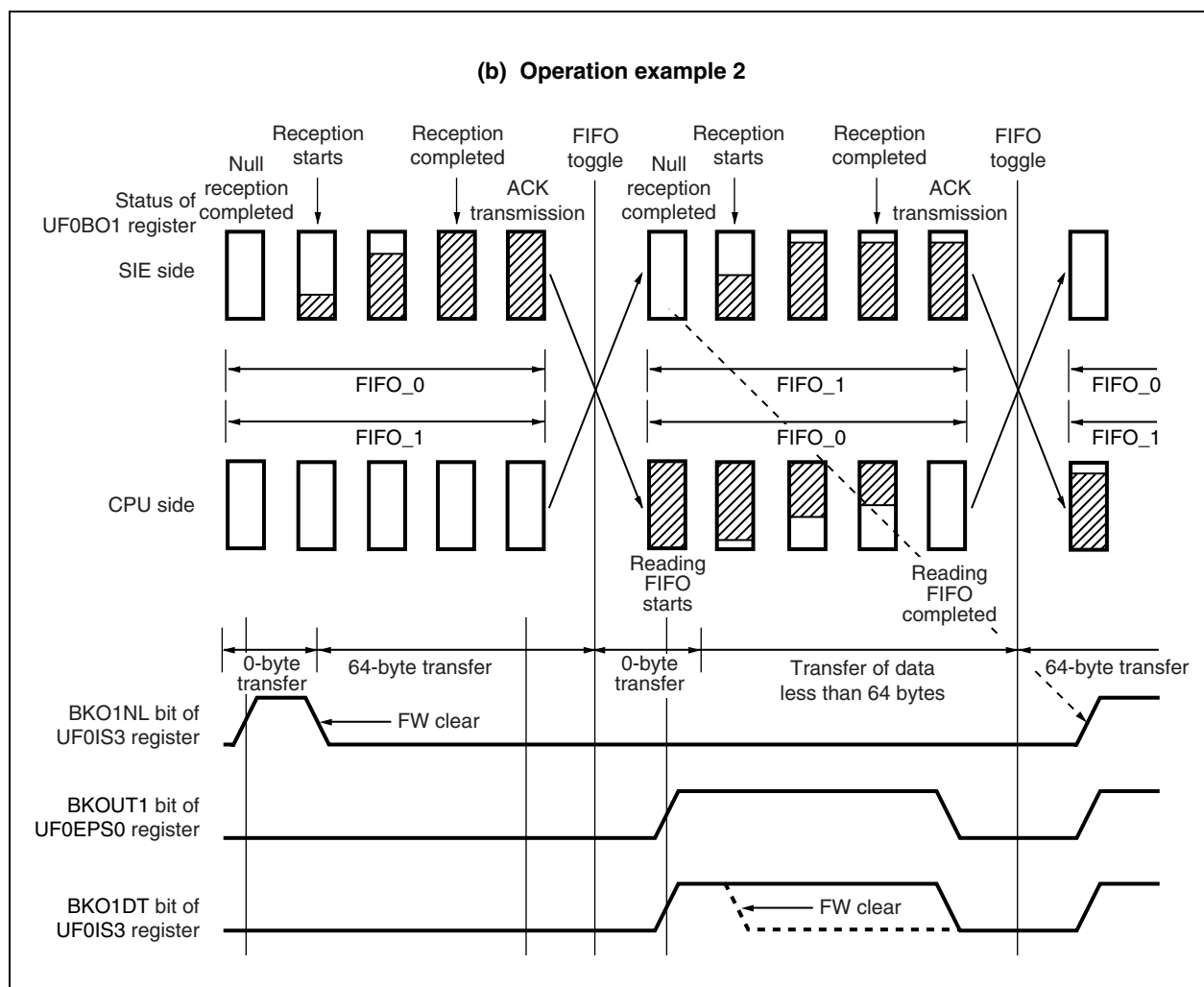
**(6) UF0 bulk-out 1 length register (UF0BO1L)**

The UF0BO1L register stores the length of the data held by the UF0BO1 register.

This register is read-only, in 8-bit units.  A write access to this register is ignored.

The UF0BO1L register always updates the received data length while it is receiving data.  If the final transfer is abnormal reception, the UF0BO1L register is cleared to 00H, and an interrupt request is not generated.  Only if the reception is normal, the interrupt request is generated, and FW can read as much data from the UF0BO1 register as the value read from the UF0BO1L register.  The value of the UF0BO1L register is decremented each time the UF0BO1 register has been read.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0BO1L | BKO1L7 | BKO1L6 | BKO1L5 | BKO1L4 | BKO1L3 | BKO1L2 | BKO1L1 | BKO1L0 | 0020010AH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | BKO1L7 to BKO1L0 | These bits store the length of the data held by the UF0BO1 register. |

**(7)  UF0 bulk-out 2 register (UF0BO2)**

The UF0BO2 register is a 64-byte × 2 FIFO that stores data for Endpoint4.  This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides.  The toggle operation takes place when data is in the FIFO on the SIE side and when no data is in the FIFO on the CPU side (counter value = 0).

This register is read-only, in 8-bit units.  A write access to this register is ignored.

When the hardware receives data for Endpoint4 from the host, it automatically transfers the data to the UF0BO2 register.  When the register correctly receives the data, a FIFO toggle operation occurs.  As a result, the BKO2DT bit of the UF0IS3 register is set to 1, the quantity of the received data is held by the UF0BO2L register, and an interrupt request or DMA request is issued to the CPU.  Whether the interrupt request or DMA request is issued can be selected by using the DQBO2MS bit of the UF0IDR register.

Read the data held by the UF0BO2 register by FW, up to the value of the amount of data read by the UF0BO2L register.  When the correct received data is held by the FIFO connected to the SIE side and the value of the UF0BO2L register reaches 0, the toggle operation of the FIFO occurs, and the BKO2NK bit of the UF0EN register is automatically cleared to 0.  If data greater than the value of the UF0BO2L register is read and if the FIFO toggle condition is satisfied, the toggle operation of the FIFO occurs.  As a result, the next packet may be read by mistake.  Note that, if the toggle condition is not satisfied, the first data is repeatedly read.

If overrun data is received while data is held by the FIFO connected to the CPU side, Endpoint4 stalls, and the FIFO on the CPU side is cleared.

When the UF0BO2 register is read while no data is in it, an undefined value is read.

**Caution   Be sure to read all the data stored in this register.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0BO2 | BKO27 | BKO26 | BKO25 | BKO24 | BKO23 | BKO22 | BKO21 | BKO20 | 0020010CH | Undefined |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | BKO27 to BKO20 | These bits store data for Endpoint4. |

The operation of the UF0BO2 register is illustrated below.

**Figure 20-8.  Operation of UF0BO2 Register (1/2)**
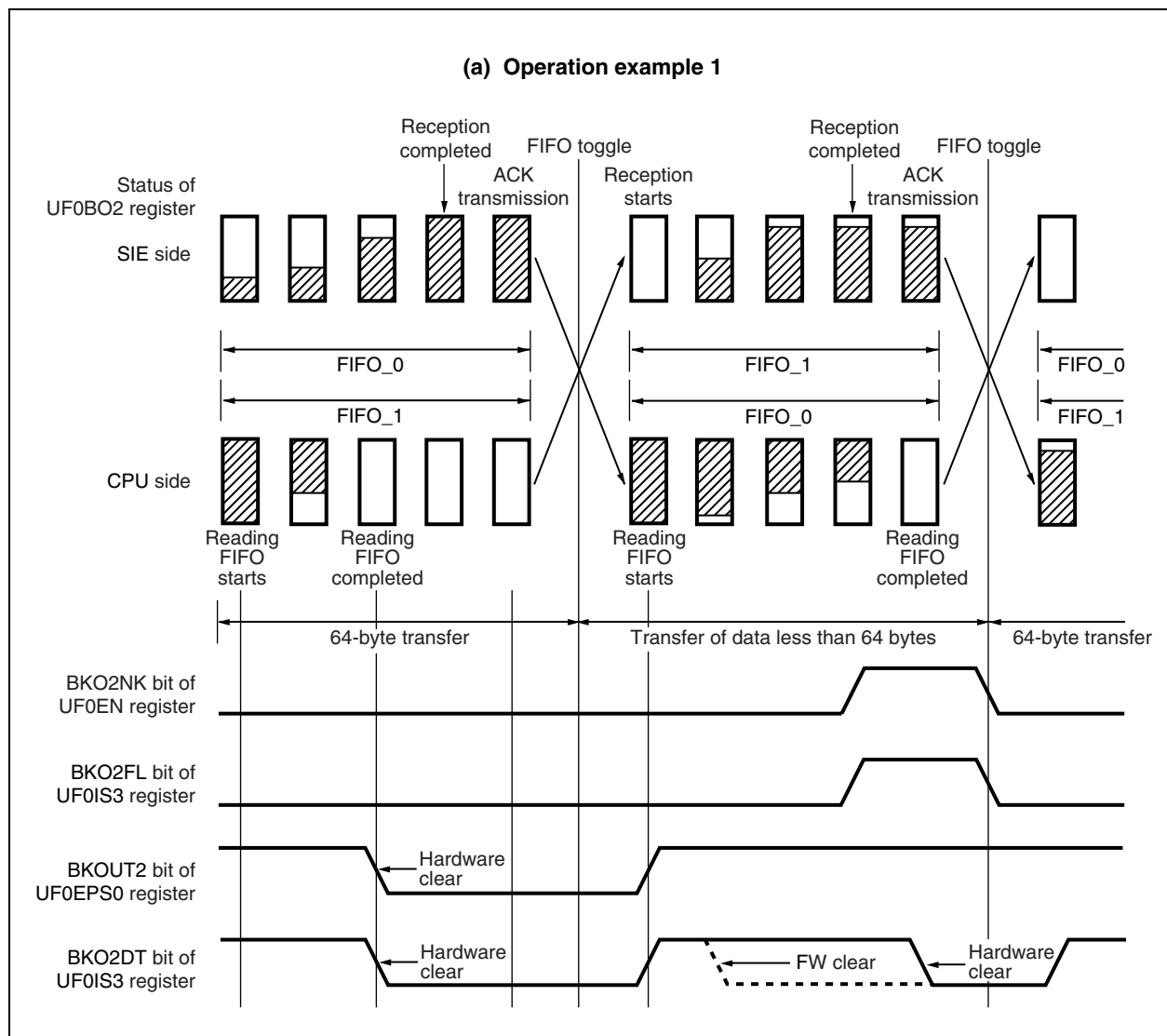
**Figure 20-8. Operation of UF0BO2 Register (2/2)**



(b) Operation example 2
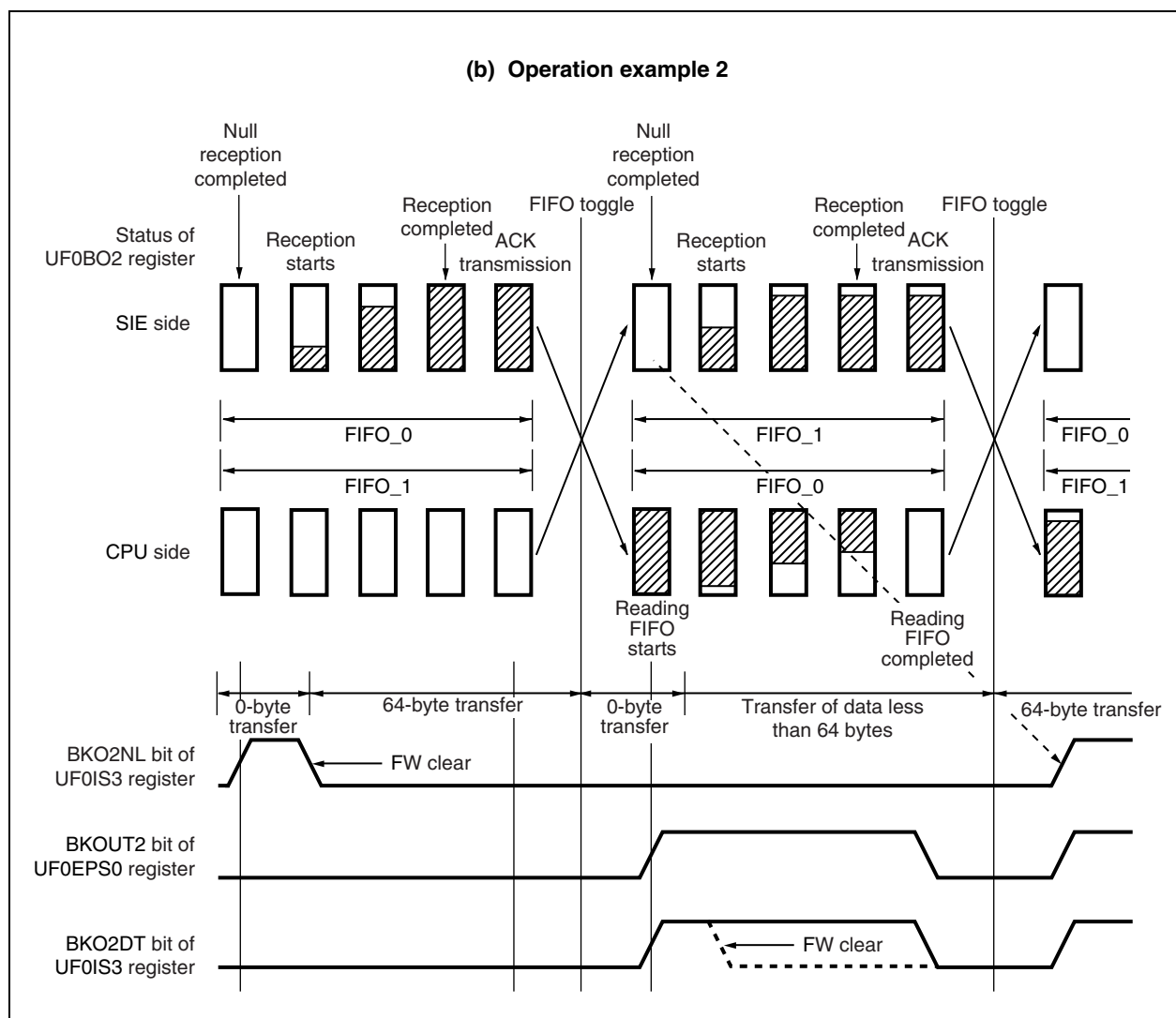
**(8)  UF0 bulk-out 2 length register (UF0BO2L)**

The UF0BO2L register stores the length of the data held by the UF0BO2 register.

This register is read-only, in 8-bit units.  A write access to this register is ignored.

The UF0BO2L register always updates the received data length while it is receiving data.  If the final transfer is abnormal reception, the UF0BO2L register is cleared to 00H, and an interrupt request is not generated.  Only if the reception is normal, the interrupt request is generated, and FW can read as much data from the UF0BO2 register as the value read from the UF0BO2L register.  The value of the UF0BO2L register is decremented each time the UF0BO2 register has been read.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0BO2L | BKO2L7 | BKO2L6 | BKO2L5 | BKO2L4 | BKO2L3 | BKO2L2 | BKO2L1 | BKO2L0 | 0020010EH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | BKO2L7 to BKO2L0 | These bits store the length of the data held by the UF0BO2 register. |

**(9)  UF0 bulk-in 1 register (UF0BI1)**

The UF0BI1 register is a 64-byte × 2 FIFO that stores data for Endpoint1.  This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides.  The toggle operation takes place when no data is in the FIFO on the SIE side (counter value = 0) and when the FIFO on the CPU side is correctly written (FIFO full or BKI1DED bit = 1).

This register is write-only, in 8-bit units.  When this register is read, 00H is read.

The hardware transmits data to the USB bus in synchronization with the IN token for Endpoint1 only when the BKI1NK bit of the UF0EN register is set to 1 (when NAK is not transmitted).  The address at which data is to be written or read is managed by the hardware.  Therefore, FW can transmit data to the host only by writing the data to the UF0BI1 register sequentially.  A short packet is transmitted when data is written to the UF0BI1 register and the BKI1DED bit of the UF0DEND register is set to 1 (BKIN1 bit of UF0EPS0 register = 1 (data exists)).  A Null packet is transmitted when the UF0BI1 register is cleared and the BKI1DED bit of the UF0DEND register is set to 1 (BKIN1 bit of the UF0EPS0 register = 1 (data exists)).  When the data is transmitted correctly, a FIFO toggle operation occurs.  The BKI1DT bit of the UF0IS2 register is set to 1, and an interrupt request is generated for the CPU.  An interrupt request or DMA request can be selected by using the DQBI1MS bit of the UF0IDR register.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0BI1 | BKI17 | BKI16 | BKI15 | BKI14 | BKI13 | BKI12 | BKI11 | BKI10 | 00200110H | Undefined |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | BKI17 to BKI10 | These bits store data for Endpoint1. |

The operation of the UF0BI1 register is illustrated below.

**Figure 20-9.  Operation of UF0BI1 Register (1/3)**



**(a)  Operation example 1**
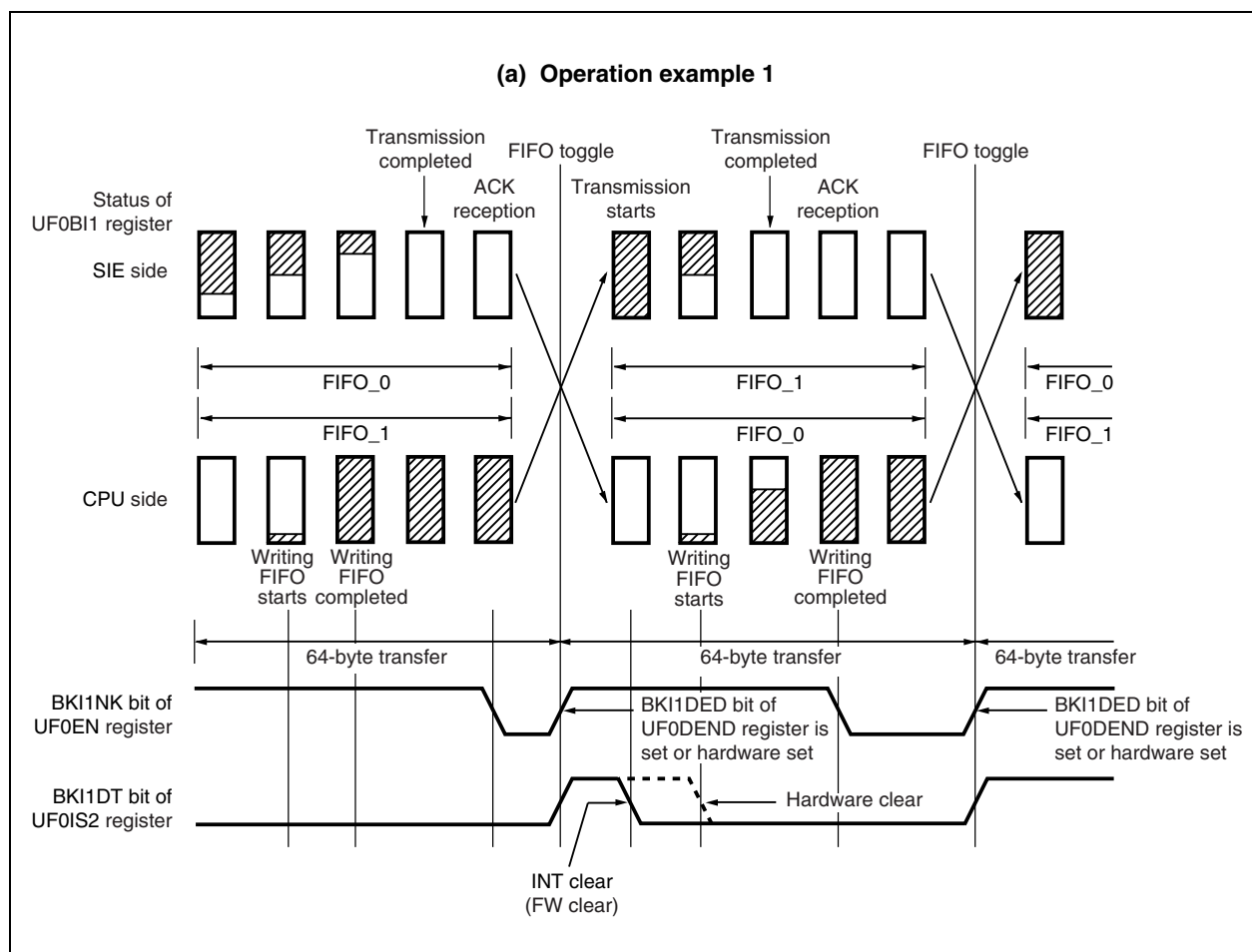
**Figure 20-9. Operation of UF0BI1 Register (2/3)**



(b) Operation example 2
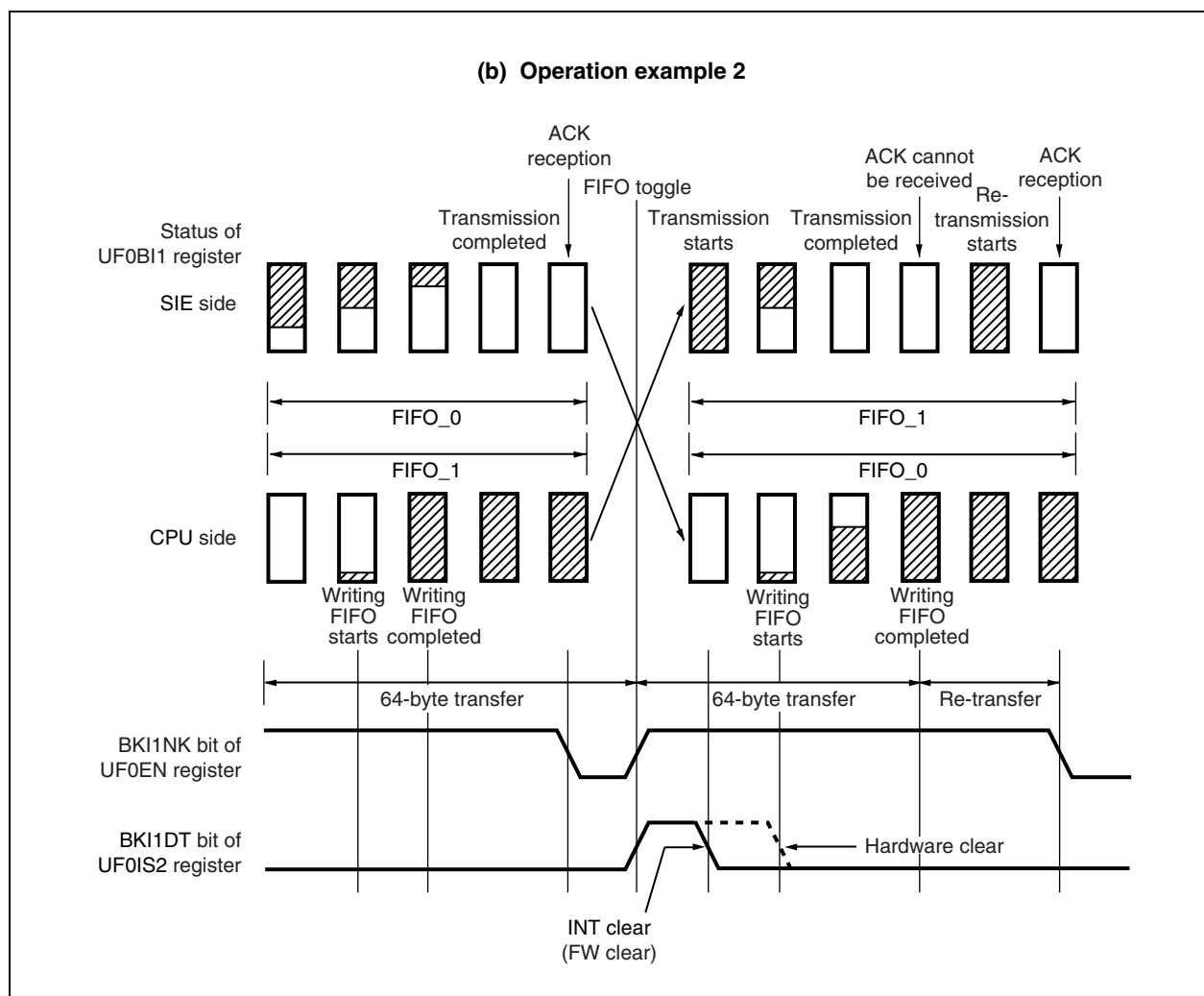
**Figure 20-9.  Operation of UF0BI1 Register (3/3)**
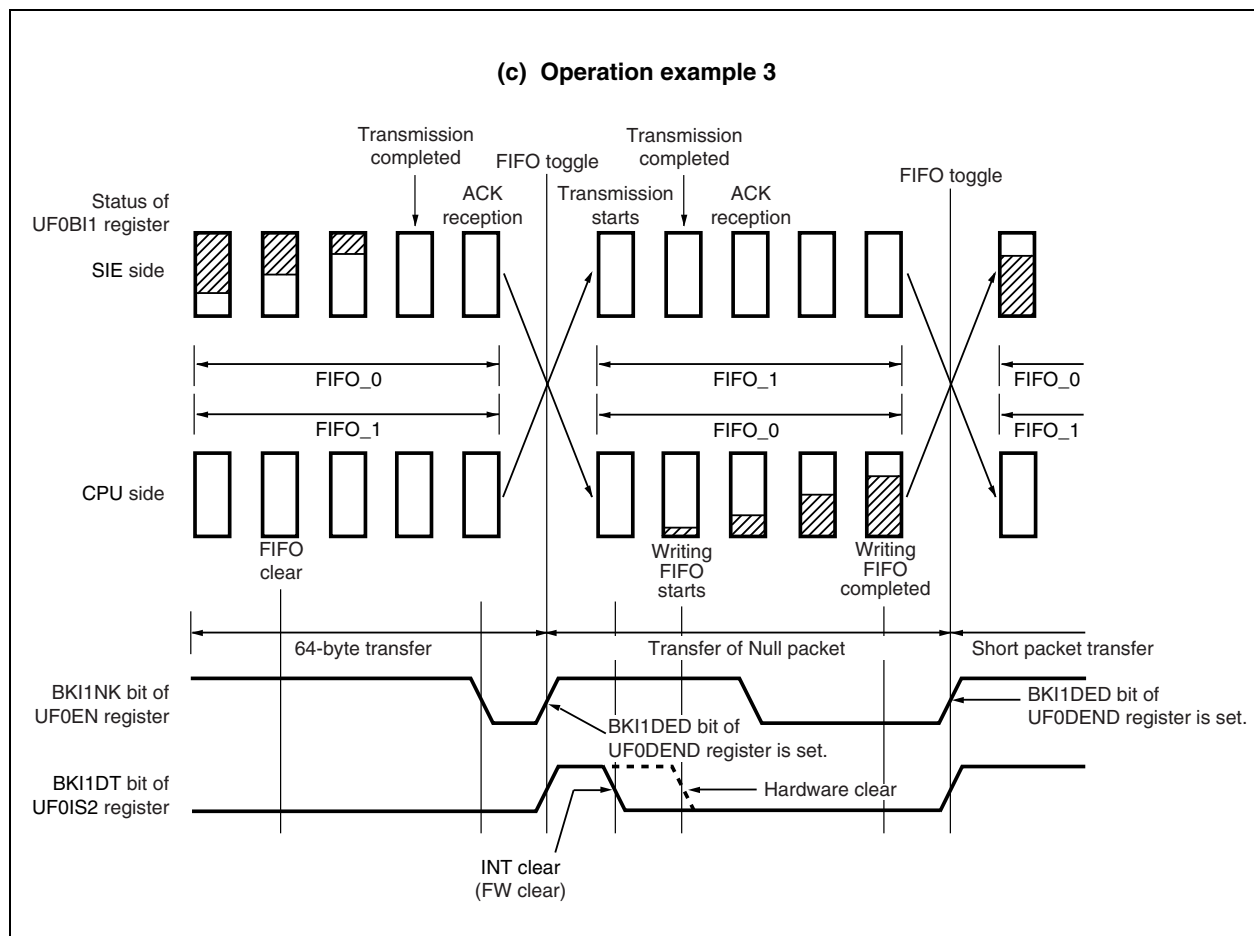
**(10) UF0 bulk-in 2 register (UF0BI2)**

The UF0BI2 register is a 64-byte × 2 FIFO that stores data for Endpoint3. This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides. The toggle operation takes place when no data is in the FIFO on the SIE side (counter value = 0) and when the FIFO on the CPU side is correctly written (FIFO full or BKI2DED bit = 1).

This register is write-only, in 8-bit units. When this register is read, 00H is read.

The hardware transmits data to the USB bus in synchronization with the IN token for Endpoint3 only when the BKI2NK bit of the UF0EN register is set to 1 (when NAK is not transmitted). The address at which data is to be written or read is managed by the hardware. Therefore, FW can transmit data to the host only by writing the data to the UF0BI2 register sequentially. A short packet is transmitted when data is written to the UF0BI2 register and the BKI2DED bit of the UF0DEND register is set to 1 (BKIN2 bit of UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0BI2 register is cleared and the BKI2DED bit of the UF0DEND register is set to 1 (BKIN2 bit of the UF0EPS0 register = 1 (data exists)). When the data is transmitted correctly, a FIFO toggle operation occurs. The BKI2DT bit of the UF0IS2 register is set to 1, and an interrupt request is generated for the CPU. An interrupt request or DMA request can be selected by using the DQBI2MS bit of the UF0IDR register.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0BI2 | BKI27 | BKI26 | BKI25 | BKI24 | BKI23 | BKI22 | BKI21 | BKI20 | 00200112H | Undefined |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | BKI27 to BKI20 | These bits store data for Endpoint3. |

The operation of the UF0BI2 register is illustrated below.

**Figure 20-10.  Operation of UF0BI2 Register (1/3)**
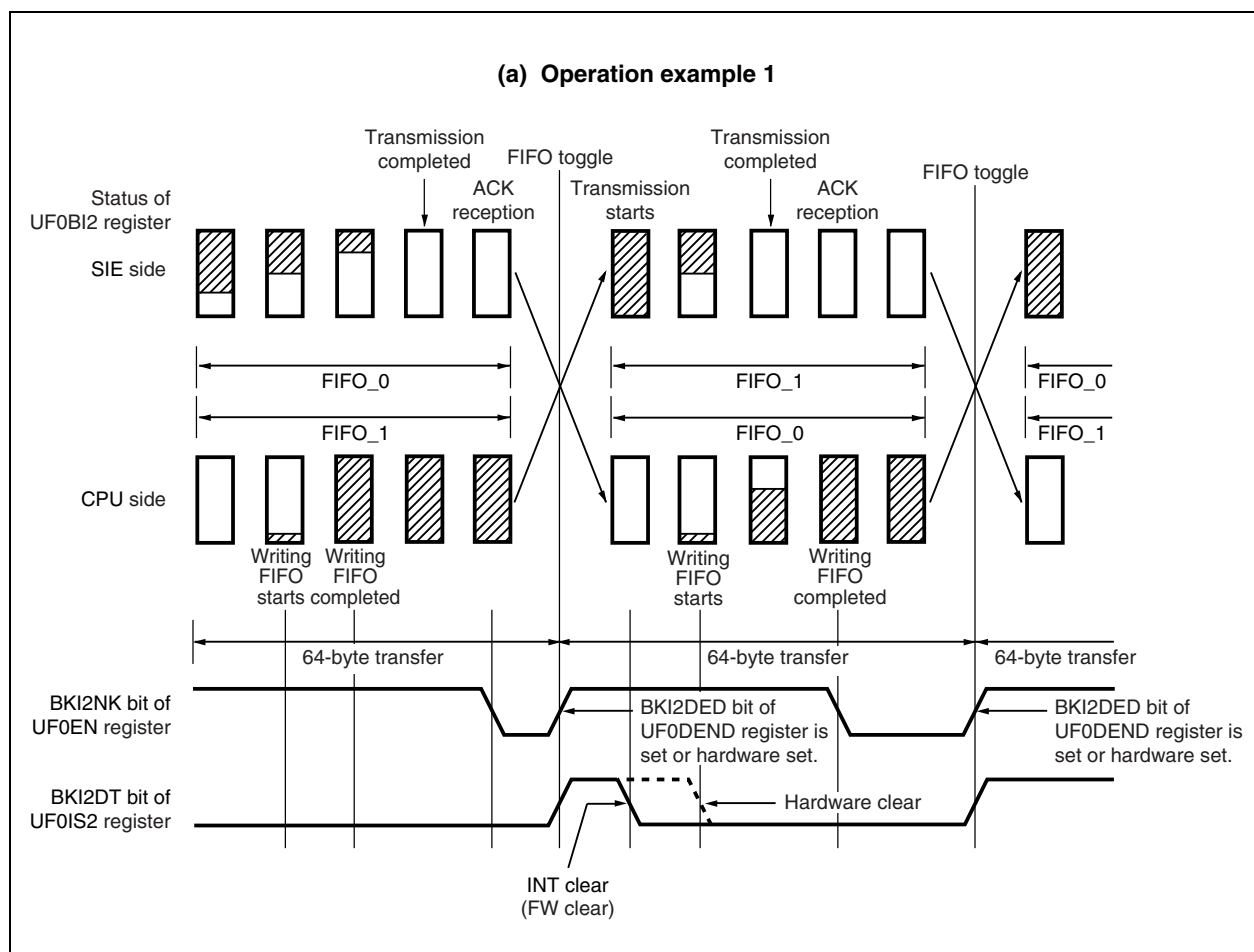
**Figure 20-10.  Operation of UF0BI2 Register (2/3)**



(b)  Operation example 2
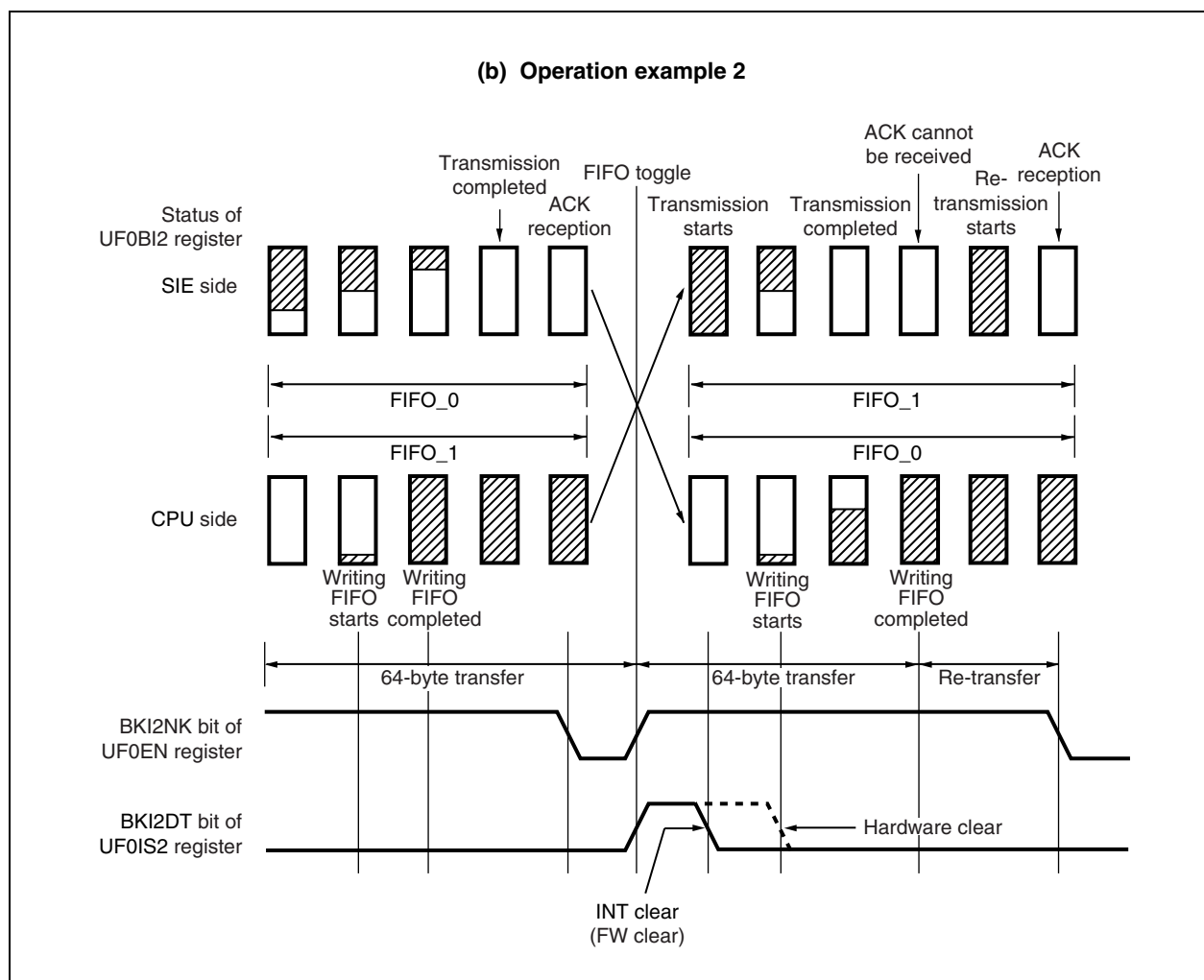
**Figure 20-10.  Operation of UF0BI2 Register (3/3)**
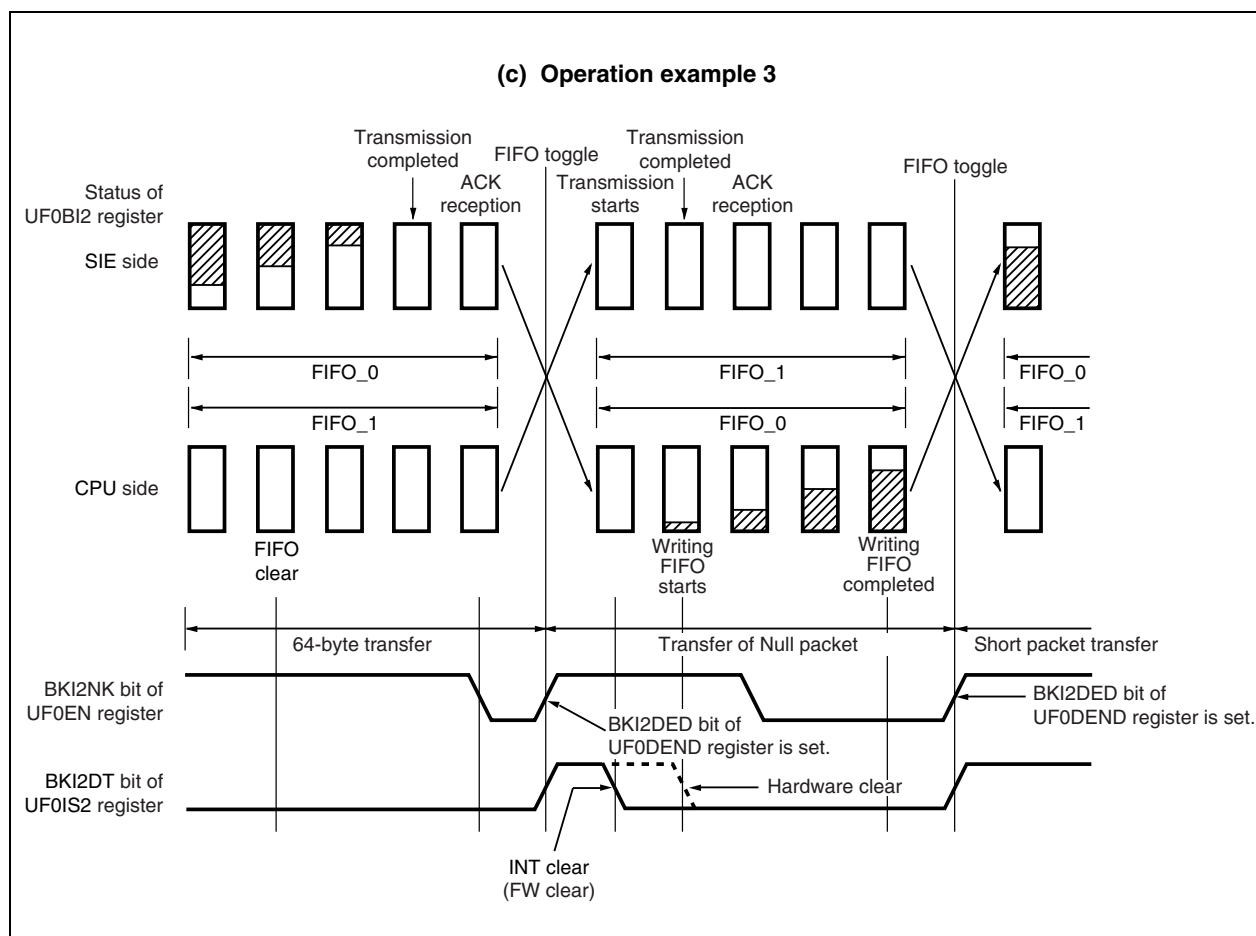
**(11) UF0 interrupt 1 register (UF0INT1)**

The UF0INT1 register is an 8-byte FIFO that stores data for Endpoint7 (to be passed to SIE).

This register is write-only, in 8-bit units. When this register is read, 00H is read.

The hardware transmits data to the USB bus in synchronization with the IN token for Endpoint7 only when the IT1NK bit of the UF0EN register is set to 1 (when NAK is not transmitted). When the data is transmitted and the host correctly receives it, the IT1NK bit of the UF0EN register is automatically cleared to 0 by hardware. A short packet is transmitted when data is written to the UF0INT1 register and the IT1DEND bit of the UF0DEND register is set to 1 (IT1 bit of the UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0INT1 register is cleared and the IT1DEND bit of the UF0DEND register is set to 1 (IT1 bit of the UF0EPS0 register = 1 (data exists)).

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0INT1 | IT17 | IT16 | IT15 | IT14 | IT13 | IT12 | IT11 | IT10 | 00200114H | Undefined |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | IT17 to IT10 | These bits store data for Endpoint7. |

The operation of the UF0INT1 register is illustrated below.

**Figure 20-11.  Operation of UF0INT1 Register**
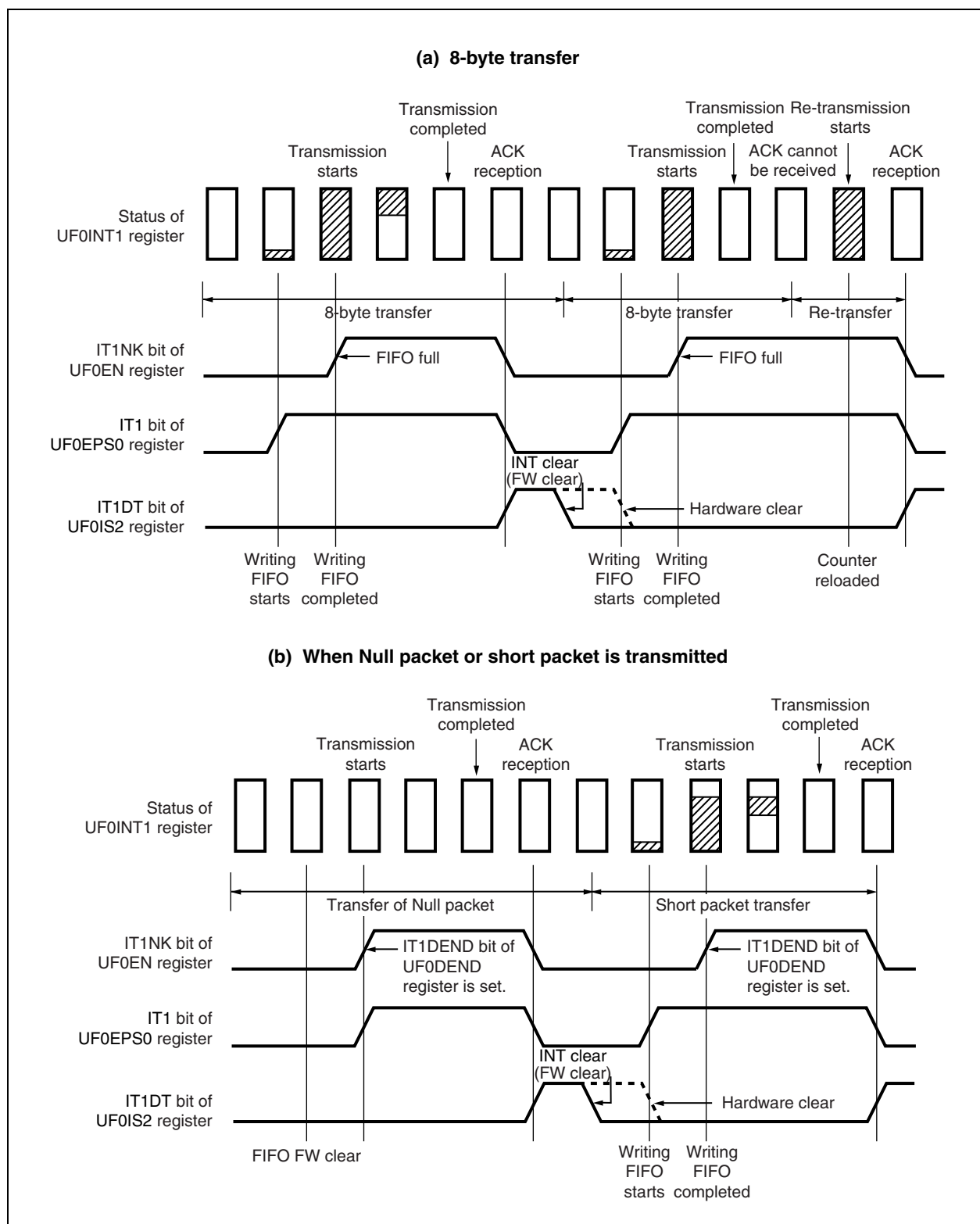
### 20.6.5 EPC request data registers

**(1) UF0 device status register L (UF0DSTL)**

This register stores the value that is to be returned in response to the GET_STATUS Device request.

This register can be read or written in 8-bit units.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Device request.

**Caution   To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0DSTL | 0 | 0 | 0 | 0 | 0 | 0 | RMWK | SFPW | 00200144H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 1 | RMWK | This bit specifies whether the remote wakeup function of the device is used.<br>   1: Enabled<br>   0: Disabled<br>If the device supports a remote wakeup function, this bit is set to 1 by hardware when the SET_FEATURE Device request has been received, and is cleared to 0 by hardware when the CLEAR_FEATURE Device request has been received. If the device does not support a remote wakeup function, make sure that the SET_FEATURE Device request is not issued from the host. |
| 0 | SFPW | This bit indicates whether the device is self-powered or bus-powered.<br>   1: Self-powered<br>   0: Bus-powered |

**(2) UF0 EP0 status register L (UF0E0SL)**

This register stores the value that is to be returned in response to the GET_STATUS Endpoint0 request.

This register can be read or written in 8-bit units.  Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in USBF, the E0HALT bit is set to 1 by FW.  A write access to this register is ignored while a USB-side access to Endpoint0 is being received.

When the E0HALT bit is set to 1 by FW, it is not reflected until the next SETUP token is received if the control transfer immediately before is for the SET_FEATURE Endpoint0, CLEAR_FEATURE Endpoint0, GET_STATUS Endpoint0 request, or an FW-processed request.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint0 request.  If Endpoint0 has stalled, the UF0E0W and UF0E0R registers are cleared, and the EP0NKW and EP0NKR bits of the UF0E0N register are cleared to 0.

**Caution  To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E0SL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | E0HALT | 0020014CH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 0 | E0HALT | This bit indicates the status of Endpoint0.<br> 1: Stalled<br> 0: Not stalled<br>This bit is set to 1 by hardware when the SET_FEATURE Endpoint0 request has been received, and cleared to 0 by hardware when the CLEAR_FEATURE Endpoint0 request has been received.  DATA PID is initialized to DATA0. |

**(3)  UF0 EP1 status register L (UF0E1SL)**

This register stores the value that is to be returned in response to the GET_STATUS Endpoint1 request.

This register can be read or written in 8-bit units.  Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint1, the E1HALT bit is set to 1.  A write access to this register is ignored while a USB-side access to Endpoint1 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint1 request.  If Endpoint1 has stalled, the UF0BI1 register is cleared and the BKI1NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint1, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

**Caution   To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E1SL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | E1HALT | 00200150H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 0 | E1HALT | This bit indicates the status of Endpoint1.<br>　1: Stalled<br>　0: Not stalled<br>This bit is set to 1 by hardware when the SET_FEATURE Endpoint1 request has been received.  It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint1 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint1 is linked has correctly been received.  DATA PID is initialized to DATA0. |

**(4) UF0 EP2 status register L (UF0E2SL)**

This register stores the value that is to be returned in response to the GET_STATUS Endpoint2 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint2, the E2HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint2 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint2 request. If Endpoint2 has stalled, the UF0BO1 register is cleared and the BKO1NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint2, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

**Caution    To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E2SL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | E2HALT | 00200154H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 0 | E2HALT | This bit indicates the status of Endpoint2.<br>  1: Stalled<br>  0: Not stalled<br>This bit is set to 1 by hardware when the SET_FEATURE Endpoint2 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint2 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint2 is linked has correctly been received. DATA PID is initialized to DATA0. |

**(5) UF0 EP3 status register L (UF0E3SL)**

This register stores the value that is to be returned in response to the GET_STATUS Endpoint3 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint3, the E3HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint3 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint3 request. If Endpoint3 has stalled, the UF0BI2 register is cleared and the BKI2NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint3, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

**Caution    To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E3SL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | E3HALT | 00200158H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 0 | E3HALT | This bit indicates the status of Endpoint3.<br>  1: Stalled<br>  0: Not stalled<br>This bit is set to 1 by hardware when the SET_FEATURE Endpoint3 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint3 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint3 is linked has correctly been received. DATA PID is initialized to DATA0. |

**(6)  UF0 EP4 status register L (UF0E4SL)**

This register stores the value that is to be returned in response to the GET_STATUS Endpoint4 request.

This register can be read or written in 8-bit units.  Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint4, the E4HALT bit is set to 1.  A write access to this register is ignored while a USB-side access to Endpoint4 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint4 request.  If Endpoint4 has stalled, the UF0BO2 register is cleared and the BKO2NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint4, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

**Caution   To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E4SL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | E4HALT | 0020015CH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 0 | E4HALT | This bit indicates the status of Endpoint4.<br>    1: Stalled<br>    0: Not stalled<br>This bit is set to 1 by hardware when the SET_FEATURE Endpoint4 request has been received.  It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint4 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint4 is linked has correctly been received.  DATA PID is initialized to DATA0. |

**(7) UF0 EP7 status register L (UF0E7SL)**

This register stores the value that is to be returned in response to the GET_STATUS Endpoint7 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint7, the E7HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint7 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint7 request. If Endpoint7 has stalled, the UF0INT1 register is cleared and the IT1NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint7, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

**Caution   To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0E7SL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | E7HALT | 00200168H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 0 | E7HALT | This bit indicates the status of Endpoint7.<br>　1: Stalled<br>　0: Not stalled<br>This bit is set to 1 by hardware when the SET_FEATURE Endpoint7 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint7 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint7 is linked has correctly been received. DATA PID is initialized to DATA0. |

**(8)  UF0 address register (UF0ADRS)**

This register stores the device address.

This register is read-only, in 8-bit units.

The device address sent by the SET_ADDRESS request is analyzed and the resultant value is automatically written to this register.  If the SET_ADDRESS request is processed by FW, the value of this register is reflected as the device address when the SUCCESS signal is received in the status stage.

**Caution   Do not execute a write access to this register.  If written, the operation is not guaranteed.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0ADRS | 0 | ADRS6 | ADRS5 | ADRS4 | ADRS3 | ADRS2 | ADRS1 | ADRS0 | 00200180H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 6 to 0 | ADRS6 to ADRS0 | These bits hold the device address of SIE. |

**(9) UF0 configuration register (UF0CNF)**

This register stores the value that is to be returned in response to the GET_CONFIGURATION request.

This register is read-only, in 8-bit units.

When the SET_CONFIGURATION request is received, its wValue is automatically written to this register.

When a change of the value of this register from 00H to other than 00H is detected, the CONF bit of the UF0MODS register is set to 1.  If the SET_CONFIGURATION request is processed by FW, the status of this register is immediately reflected on the UF0MODS register as soon as data has been written to this register (CONF bit = 1 before completion of the status stage).

**Caution   Do not execute a write access to this register.  If written, the operation is not guaranteed.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0CNF | 0 | 0 | 0 | 0 | 0 | 0 | CONF1 | CONF0 | 00200182H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 1, 0 | CONF1, CONF0 | These bits hold the data to be returned in response to the GET_CONFIGURATION request. |

**(10)  UF0 interface 0 register (UF0IF0)**

This register stores the value that is to be returned in response to the GET_INTERFACE wIndex = 0 request.

This register is read-only, in 8-bit units.

When the SET_INTERFACE request is received, its wValue is automatically written to this register.

If the SET_INTERFACE request is processed by FW, wIndex and wValue are decoded, and the setting of endpoint is automatically changed.  At this time, the status bit of the target endpoint and DPID are automatically cleared to 0, depending on the setting.  The FIFO is not cleared automatically.

**Caution    Do not execute a write access to this register.  If written, the operation is not guaranteed.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IF0 | 0 | 0 | 0 | 0 | 0 | IF02 | IF01 | IF00 | 00200184H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 2 to 0 | IF02 to IF00 | These bits hold the data to be returned in response to GET_INTERFACE wIndex = 0 request. |

**(11) UF0 interface 1 to 4 registers (UF0IF1 to UF0IF4)**

These registers store the value that is to be returned in response to the GET_INTERFACE wIndex = n request (n = 1 to 4).

These registers are read-only, in 8-bit units.

When the SET_INTERFACE request is received, its wValue is automatically written to these registers.

These registers are invalidated according to the setting of the UF0AIFN and UF0AAS registers.

If the SET_INTERFACE request is processed by FW, wIndex and wValue are decoded, and the setting of endpoint is automatically changed.  At this time, the status bit of the target endpoint and DPID are automatically cleared to 0, depending on the setting.  The FIFO is not cleared automatically.

**Caution    Do not execute a write access to this register.  If written, the operation is not guaranteed.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0IF1 | 0 | 0 | 0 | 0 | 0 | IF12 | IF11 | IF10 | 00200186H | 00H |
| UF0IF2 | 0 | 0 | 0 | 0 | 0 | IF22 | IF21 | IF20 | 00200188H | 00H |
| UF0IF3 | 0 | 0 | 0 | 0 | 0 | IF32 | IF31 | IF30 | 0020018AH | 00H |
| UF0IF4 | 0 | 0 | 0 | 0 | 0 | IF42 | IF41 | IF40 | 0020018CH | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 2 to 0 | IFn2 to IFn0 | These bits hold the data to be returned in response to GET_INTERFACE wIndex = n request. |

**Remark**    n = 1 to 4

**(12) UF0 descriptor length register (UF0DSCL)**

This register stores the length of the value that is to be returned in response to the GET_DESCRIPTOR Configuration request.  The value of this register is the number of bytes of all the descriptors set by the UF0CIEn register minus 1 (n = 0 to 255).  The total descriptor length that is to be returned in response to the GET_DESCRIPTOR Configuration request is determined according to the value of this register.

This register can be read or written in 8-bit units.  However, data can be written to this register only when the EP0NKA bit is set to 1.

Processing of wLength is automatically controlled.  If this register is set to 00H, it means that the descriptor to be returned is 1 byte long.  If the register is set to FFH, a descriptor length of 256 bytes is returned.  When a descriptor exceeding 256 bytes in length is used, set the CDCGDST bit of the UF0MODC register to 1 and process the GET_DESCRIPTOR request by FW (at this time, the CDCGD bit of the UF0MODS register is also set to 1).

**Caution    To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0DSCL | DPL7 | DPL6 | DPL5 | DPL4 | DPL3 | DPL2 | DPL1 | DPL0 | 002001A0H | 00H |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | DPL7 to DPL0 | These bits set the value of the number of bytes of all the descriptors to be returned in response to the GET_DESCRIPTOR Configuration request minus 1. |

**(13) UF0 device descriptor registers 0 to 17 (UF0DD0 to UF0DD17)**

These registers store the value to be returned in response to the GET_DESCRIPTOR Device request.

These registers can be read or written in 8-bit units. However, data can be written to these registers only when the EP0NKA bit is set to 1.

    **Cautions 1.**  **To rewrite these registers, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

        **2.**  **Use the value defined by USB Specification Ver. 2.0 and the latest Class Specification as the set value.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---|---|
| UF0DDn (n = 0 to 17) | | | | | | | | | See **Table 20-5**. | Undefined |

**Table 20-5. Mapping and Data of UF0 Device Descriptor Registers**

| Symbol | Address | Field Name | Contents |
|---|---|---|---|
| UF0DD0 | 002001A2H | bLength | Size of this descriptor |
| UF0DD1 | 002001A4H | bDescriptorType | Device descriptor type |
| UF0DD2 | 002001A6H | bcdUSB | Value below decimal point of Rev. number of USB specification |
| UF0DD3 | 002001A8H | | Value above decimal point of Rev. number of USB specification |
| UF0DD4 | 002001AAH | bDeviceClass | Class code |
| UF0DD5 | 002001ACH | bDeviceSubClass | Subclass code |
| UF0DD6 | 002001AEH | bDeviceProtocol | Protocol code |
| UF0DD7 | 002001B0H | bMaxPacketSize0 | Maximum packet size of Endpoint0 |
| UF0DD8 | 002001B2H | idVendor | Lower value of vendor ID |
| UF0DD9 | 002001B4H | | Higher value of vendor ID |
| UF0DD10 | 002001B6H | idProduct | Lower value of product ID |
| UF0DD11 | 002001B8H | | Higher value of product ID |
| UF0DD12 | 002001BAH | bcdDevice | Lower value of device release number |
| UF0DD13 | 002001BCH | | Higher value of device release number |
| UF0DD14 | 002001BEH | iManufacturer | Index of string descriptor describing manufacturer |
| UF0DD15 | 002001C0H | iProduct | Index of string descriptor describing product |
| UF0DD16 | 002001C2H | lSerialNumber | Index of string descriptor describing device serial number |
| UF0DD17 | 002001C4H | BNumConfigurations | Number of settable configurations |

**(14) UF0 configuration/interface/endpoint descriptor registers 0 to 255 (UF0CIE0 to UF0CIE255)**

These registers store the value to be returned in response to the GET_DESCRIPTOR Configuration request.

These registers can be read or written in 8-bit units. However, data can be written to these registers only when the EP0NKA bit is set to 1.

Descriptor information of up to 256 bytes can be stored in these registers. Store each descriptor in the order of Configuration, Interface, and Endpoint (see **Table 20-6**). If there are two or more Interfaces, repeatedly store the data following the Interface descriptor.

**Table 20-6. Mapping of UF0CIEn Register**

| Address | Descriptor Stored |
|---------|-------------------|
| 002001C6H | Configuration descriptor (9 bytes) |
| 002001D8H | Interface descriptor (9 bytes) |
| 002001EAH | Endpoint1 descriptor (7 bytes) |
| 002001F8H | Endpoint2 descriptor (7 bytes) |
| 00200206H | Endpoint3 descriptor (7 bytes) |
| : | : |
| 002002xxH | Interface descriptor (9 bytes) |
| 002002xxH+9 | Endpoint1 descriptor (7 bytes) |
| 002002xxH+16 | Endpoint2 descriptor (7 bytes) |
| 002002xxH+23 | Endpoint3 descriptor (7 bytes) |
| : | : |

The range of the valid data that can be set to these registers varies according to the setting of the UF0DSCL register. In addition to the descriptors listed in Table 20-7, descriptors peculiar to classes and vendors can also be stored.

If all the values are fixed, they can be stored in ROM.

**Cautions 1.** **To rewrite these registers, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**

**2.** **Use the value defined by USB Specification Ver. 2.0 and the latest Class Specification as the set value.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset |
|---|---|---|---|---|---|---|---|---|---------|-------------|
| UF0CIEn<br>(n = 0 to 255) | | | | | | | | | 002001C6H to<br>002003C4H | Undefined |

**Table 20-7. Data of UF0CIEn Register**

**(a) Configuration descriptor (9 bytes)**

| Offset | Field Name | Contents |
|--------|------------|----------|
| 0 | bLength | Size of this descriptor |
| 1 | bDescriptorType | Descriptor type |
| 2 | wTotalLength | Lower value of the total number of bytes of Configuration, all Interface, and all Endpoint descriptors |
| 3 | | Higher value of the total number of bytes of Configuration, all Interface, and all Endpoint descriptors |
| 4 | bNumInterface | Number of Interfaces |
| 5 | bConfigurationValue | Value to select this Configuration |
| 6 | iConfiguration | Index of string descriptor describing this Configuration |
| 7 | bmAttributes | Features of this Configuration (self-powered, without remote wakeup) |
| 8 | MaxPower | Maximum power consumption of this Configuration (unit: mA)[Note] |

**Note**    Shown in 2 mA units. (example: 50 = 100 mA)

**(b) Interface descriptor (9 bytes)**

| Offset | Field Name | Contents |
|--------|------------|----------|
| 0 | bLength | Size of this descriptor |
| 1 | bDescriptorType | Descriptor type |
| 2 | bInterfaceNumber | Value of this Interface |
| 3 | bAlternateSetting | Value to select alternative setting of Interface |
| 4 | bNumEndpoints | Number of usable Endpoints |
| 5 | bInterfaceClass | Class code |
| 6 | bInterfaceSubClass | Subclass code |
| 7 | bInterfaceProtocol | Protocol code |
| 8 | Interface | Index of string descriptor describing this Interface |

**(c) Endpoint descriptor (7 bytes)**

| Offset | Field Name | Contents |
|--------|------------|----------|
| 0 | bLength | Size of this descriptor |
| 1 | bDescriptorType | Descriptor type |
| 2 | bEndpointAddress | Address/transfer direction of this Endpoint |
| 3 | bmAttributes | Transfer type |
| 4 | wMaxPaketSize | Lower value of maximum number of transfer data |
| 5 | | Higher value of maximum number of transfer data |
| 6 | bInterval | Transfer interval |

### 20.6.6　Bridge register

**(1)　Bridge interrupt control register (BRGINTT)**

The BRGINTT register controls the DMA transfer status of the interrupt generate status, and each end point (EP1 to EP4) from EPC to bridge circuit.

The BRGINTT register can be read or written in 16-bit units.

After reset: 0000H　　R/W　　Address: 00200400H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| BRGINTT | 0 | 0 | 0 | 0 | EP4INT | EP3INT | EP2INT | EP1INT |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | EPCINT2B | EPCINT1B | EPCINT0B |

| Bit position | Bit name | Function |
|---|---|---|
| 11 | EP4INT | In EP4, when the DMA transfer is normal terminate, or the error finished in the DMA transferring, this bit is setting.  Clearing to "0" by writing "1".<br>　　0: DMA transfer not completion<br>　　1: DMA transfer completion |
| 10 | EP3NT | In EP3, when the DMA transfer is normal terminate, or the error finished in the DMA transferring, this bit is setting.  Clearing to "0" by writing "1".<br>　　0: DMA transfer not completion<br>　　1: DMA transfer completion |
| 9 | EP2NT | In EP2, when the DMA transfer is normal terminate, or the error finished in the DMA transferring, this bit is setting.  Clearing to "0" by writing "1".<br>　　0: DMA transfer not completion<br>　　1: DMA transfer completion |
| 8 | EP1NT | In EP1, when the DMA transfer is normal terminate, or the error finished in the DMA transferring, this bit is setting.  Clearing to "0" by writing "1".<br>　　0: DMA transfer not completion<br>　　1: DMA transfer completion |
| 2 | EPCINT2B | Showing the status of the interrupt signal "EPC_INT2B" from EPC.<br>Clear controlling from the request of EPC register<br>　　0: Interrupt not issued<br>　　1: Interrupt issued |
| 1 | EPCINT1B | Showing the status of the interrupt signal "EPC_INT1B" from EPC.<br>Clear controlling from the request of EPC register<br>　　0: Interrupt not issued<br>　　1: Interrupt issued |
| 0 | EPCINT0B | Showing the status of the interrupt signal "EPC_INT0B" from EPC.<br>Clear controlling from the request of EPC register<br>　　0: Interrupt not issued<br>　　1: Interrupt issued |

**(2) Bridge interrupt enable register (BRGINTE)**

The BRGINTE register controls whether the interrupt generated in the bridge circuit is enabled or disabled.

The BRGINTE register can be read or written in 16-bit units.

After reset: 0000H    R/W    Address: 00200402H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| BRGINTE | 0 | 0 | 0 | 0 | EP4INTN | EP3INTN | EP2INTN | EP1INTN |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | EPC INT2BEN | EPC INT1BEN | EPC INT0BEN |

| Bit position | Bit name | Function |
|---|---|---|
| 11 | EP4INTN | Setting the interrupt occur enable or disable when EP4INT bit is setting. <br> 0: Interrupt disabled <br> 1: Interrupt enabled |
| 10 | EP3NTN | Setting the interrupt occur enable or disable when EP3INT bit is setting. <br> 0: Interrupt disabled <br> 1: Interrupt enabled |
| 9 | EP2NTN | Setting the interrupt occur enable or disable when EP2INT bit is setting. <br> 0: Interrupt disabled <br> 1: Interrupt enabled |
| 8 | EP1NTN | Setting the interrupt occur enable or disable when EP1INT bit is setting. <br> 0: Interrupt disabled <br> 1: Interrupt enabled |
| 2 | EPCINT2BEN | Setting the interrupt occur enable or disable when EPCINT2BEN bit is setting. <br> 0: Interrupt disabled <br> 1: Interrupt enabled |
| 1 | EPCINT1BEN | Setting the interrupt occur enable or disable when EPCINT1BEN bit is setting. <br> 0: Interrupt disabled <br> 1: Interrupt enabled |
| 0 | EPCINT0BEN | Setting the interrupt occur enable or disable when EPCINT0BEN bit is setting. <br> 0: Interrupt disabled <br> 1: Interrupt enabled |

**(3) EPC macro control register (EPCCLT)**

The EPCCLT register controls the reset generator to the EPC macro.

The EPCCLT register can be read or written in 16-bit units.

After reset: 0000H    R/W    Address: 00200404H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| EPCCLT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | EPCRST |

| Bit position | Bit name | Function |
|---|---|---|
| 0 | EPCRST | Setting the reset occurs to EPC.<br>0: Reset released<br>1: Reset issued |

RENESAS

**(4)  CPU I/F bus control register (CPUBCTL)**

The CPUBCTL register controls the interface between bridge circuit and CPU.

The CPUBCTL register can be read or written in 16-bit units.

After reset: Undefined    R/W    Address: 00200408H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| CPUBCTL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | BULKWAIT | DATAWAIT | NOWAIT |

| Bit position | Bit name | Function |
|---|---|---|
| 2 | BULKWAIT | Forcibly inserting the 1 wait (bulk wait) when the bulk register is accessed.<br>0: No forcibly insert the bulk wait[Note] (default value)<br>1: Forcibly insert the bulk wait<br><br>**Note**  The setting is invalid in write accessing, the bulk wait is forcibly inserted. |
| 1 | DATAWAIT | Forcibly inserting the 1 wait (data wait) after the CPU bus cycle.<br>0: No forcibly insert the data wait (default value)<br>1: Forcibly insert the data wait |
| 0 | NOWAIT | Setting enables/disable the no wait operation of CPU bus cycle.<br>0: No wait disables[Note] (default value)<br>1: No wait enables<br><br>**Note**  1 wait or more is inserted. |

### 20.6.7 DMA register

**(1) EPn DMA control register 1 (UF0E1DC1 to UF0E4DC1)**

The UF0E1DC1 to UF0E4DC1 register controls the DMA transfer of end point n (EPn). (n = 1 to 4)

The UF0E1DC1 to UF0E4DC1 register can be read or written in 16-bit units.

(1/2)

After reset: 0000H    R/W    Address: 00200500H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0E1DC1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | EP1BULK2 | EP1BULK1 | EP1BULK0 | EP1STOP | EP1REQ | EP1DMAEN |

After reset: 0000H    R/W    Address: 00200504H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0E2DC1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | EP2BULK2 | EP2BULK1 | EP2BULK0 | EP2STOP | EP2REQ | EP2DMAEN |

After reset: 0000H    R/W    Address: 00200508H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0E3DC1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | EP3BULK2 | EP3BULK1 | EP3BULK0 | EP3STOP | EP3REQ | EP3DMAEN |

After reset: 0000H    R/W    Address: 0020050CH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0E4DC1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | EP4BULK2 | EP4BULK1 | EP4BULK0 | EP4STOP | EP4REQ | EP4DMAEN |

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 5 to 3 | EPnBULK2, EPnBULK1, EPnBULK0 | Shown the status the state machine "BIN_STATE" for bulk transfer of the internal bridge<br><br>| EPnBULK2 | EPnBULK1 | EPnBULK0 | "BIN_STATE" status |<br>|---|---|---|---|<br>| 0 | 0 | 0 | BIN_IDLE |<br>| 0 | 0 | 1 | BIN_CPU |<br>| 0 | 1 | 0 | BIN_EPC |<br>| 0 | 1 | 1 | BIN_CMP |<br>| 1 | 0 | 0 | BIN_END | |
| 2 | EPnSTOP | Showing the status (end factor of DMA transfer) of DMA transfer end from EPC<br>    0: End of DMA transfer by EPn_TCNT value "0"<br>    1: End of DMA transfer by negate of "EPC_DMARQ_EPnB"<br>Automatically clear (0) by setting next EP1_DMAEN to "1". |
| 1 | EPnREQ | Showing the status of "EPC_DMARQ_EPnB" signal from EPC<br>    0: No DMA request signal<br>    1: DMA request signal |
| 0 | EPnDMAEN | Setting the control of DMA request from EPC<br>    0: Masks DMA request<br>    1: Enables DMA request<br>Automatically clear (0) by complete number of packet transfer setting in EPn_TCNT, or complete the DMA transfer by the negate of DMARQ_EPnB.<br><br>**Caution  The setting value is not guaranteed in forcibly end.** |

**Remark**   n = 1 to 4

**(2) EPn DMA control register 2 (UF0E1DC2 to UF0E4DC2)**

The UF0E1DC2 to UF0E4DC2 register controls the DMA transfer of end point n (EPn). (n = 1 to 4)

The UF0E1DC2 to UF0E4DC2 register can be read or written in 16-bit units.

(1/2)

After reset: 0000H    R/W    Address: 00200502H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0E1DC2 | EP1 TCNT15 | EP1 TCNT14 | EP1 TCNT13 | EP1 TCNT12 | EP1 TCNT11 | EP1 TCNT10 | EP1 TCNT9 | EP1 TCNT8 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EP1 TCNT7 | EP1 TCNT6 | EP1 TCNT5 | EP1 TCNT4 | EP1 TCNT3 | EP1 TCNT2 | EP1 TCNT1 | EP1 TCNT0 |

After reset: 0000H    R/W    Address: 00200506H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0E2DC2 | EP2 TCNT15 | EP2 TCNT14 | EP2 TCNT13 | EP2 TCNT12 | EP2 TCNT11 | EP2 TCNT10 | EP2 TCNT9 | EP2 TCNT8 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EP2 TCNT7 | EP2 TCNT6 | EP2 TCNT5 | EP2 TCNT4 | EP2 TCNT3 | EP2 TCNT2 | EP2 TCNT1 | EP2 TCNT0 |

After reset: 0000H    R/W    Address: 0020050AH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0E3DC2 | EP3 TCNT15 | EP3 TCNT14 | EP3 TCNT13 | EP3 TCNT12 | EP3 TCNT11 | EP3 TCNT10 | EP3 TCNT9 | EP3 TCNT8 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EP3 TCNT7 | EP3 TCNT6 | EP3 TCNT5 | EP3 TCNT4 | EP3 TCNT3 | EP3 TCNT2 | EP3 TCNT1 | EP3 TCNT0 |

After reset: 0000H    R/W    Address: 0020050EH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0E4DC2 | EP4 TCNT15 | EP4 TCNT14 | EP4 TCNT13 | EP4 TCNT12 | EP4 TCNT11 | EP4 TCNT10 | EP4 TCNT9 | EP4 TCNT8 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EP4 TCNT7 | EP4 TCNT6 | EP4 TCNT5 | EP4 TCNT4 | EP4 TCNT3 | EP4 TCNT2 | EP4 TCNT1 | EP4 TCNT0 |

| Bit position | Bit name | Function |
|---|---|---|
| 15 to 0 | EPnTCNT15 to EPnTCNT0 | Setting the number of byte to DMA transfer in EPn.<br>End the DMA transfer after the value of EPn_TCNT is "0" to decrement each transfer.<br><br>**Cautions 1. Set this register when EPn_DMAEN = 0.**<br>　　　　　**2. Setting this register to "0" is prohibited.**<br>　　　　　**Be sure to set this register +1 value for the value of DMA transfer count register DBC0 to DBC3.**<br>　　　　　**3. The setting value of this register is reflected the counter BIN_TCNT for bulk transfer of the bridge inside.  And the value of BIN_TCNT is "0", EPn_TCN is "0", too.**<br>　　　　　**4. Update the value of the counter BIN_TCNT for bulk transfer is stopped when forcibly terminated.** |

**Remark**　n = 1 to 4

### 20.6.8  Bulk-in register

#### (1)  UF0 EP1 bulk-in transfer data register (UF0EP1BI)

The UF0EP1BI register writes the bulk-in transfer data of EP1.

The UF0EP1BI register can be read or written in 8-bit or 16-bit units.

After reset: 0000H    R/W    Address: 00201000H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0EP1BI | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EP1BI7 | EP1BI6 | EP1BI5 | EP1BI4 | EP1BI3 | EP1BI2 | EP1BI1 | EP1BI0 |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | EP1BI7 to EP1BI0 | Writing the bulk-out transfer data of EP1.<br>Data outputting to the EPC macro by writing data to this register.<br>If using this register, setting the address (00201000H) in DMA destination address register (DDAn (n = 0 to 3)) of DMAC. In addition, set the RQnUR1E (n = 0 to 3) bit of the UFDRQEN register to 1 to assign a DMA channel. |

#### (2)  UF0 EP3 bulk-in transfer data register (UF0EP3BI)

The UF0EP3BI register writes the bulk-in transfer data of EP1.

The UF0EP3BI register can be read or written in 8-bit or 16-bit units.

After reset: 0000H    R/W    Address: 00202000H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0EP3BI | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EP3BI7 | EP3BI6 | EP3BI5 | EP3BI4 | EP3BI3 | EP3BI2 | EP3BI1 | EP3BI0 |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | EP3BI7 to EP3BI0 | Writing the bulk-out transfer data of EP3.<br>Data outputting to the EPC macro by writing data to this register.<br>If using this register, setting the address (00202000H) in DMA destination address register (DDAn (n = 0 to 3)) of DMAC. In addition, set the RQnUR3E (n = 0 to 3) bit of the UFDRQEN register to 1 to assign a DMA channel. |

### 20.6.9 Bulk-out register

#### (1) UF0 EP2 bulk-out transfer data register (UF0EP2BO)

The UF0EP2BO register writes the bulk-out transfer data of EP2.

The UF0EP2BO register can be read or written in 8-bit or 16-bit units.

After reset: 0000H　R　Address: 00210000H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0EP2BO | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EP2BO7 | EP2BO6 | EP2BO5 | EP2BO4 | EP2BO3 | EP2BO2 | EP2BO1 | EP2BO0 |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | EP2BO7 to EP2BO0 | Reading the bulk-out transfer data of EP2.<br>Reading the input data from the EPC macro from this register.<br>If using this register, setting the address (00210000H) in DMA source address register (DSAn (n = 0 to 3)) of DMAC. In addition, set the RQnUR0E (n = 0 to 3) bit of the UFDRQEN register to 1 to assign a DMA channel. |

**Caution** **If either of the following operations is performed, the data stored in this register is read out and the next bulk-out transfer data is stored into this register.**

- **The UF0EP2BO register is read during program execution.**
- **The UF0EP2BO register is monitored on the memory window while the debugger is being used.**

**(2) UF0 EP4 bulk-out transfer data register (UF0EP4BO)**

The UF0EP4BO register writes the bulk-out transfer data of EP4.

The UF0EP4BO register can be read or written in 8-bit or 16-bit units.

After reset: 0000H    R    Address: 00220000H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UF0EP4BO | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EP4BO7 | EP4BO6 | EP4BO5 | EP4BO4 | EP4BO3 | EP4BO2 | EP4BO1 | EP4BO0 |

| Bit position | Bit name | Function |
|---|---|---|
| 7 to 0 | EP4BO7 to EP4BO0 | Reading the bulk-out transfer data of EP4.<br>Reading the input data from the EPC macro from this register.<br>If using this register, setting the address (00220000H) in DMA source address register (DSAn (n = 0 to 3)) of DMAC. In addition, set the RQnUR2E (n = 0 to 3) bit of the UFDRQEN register to 1 to assign a DMA channel. |

**Caution** **If either of the following operations is performed, the data stored in this register is read out and the next bulk-out transfer data is stored into this register.**

- **The UF0EP4BO register is read during program execution.**
- **The UF0EP4BO register is monitored on the memory window while the debugger is being used.**

### 20.6.10  Peripheral control registers

**(1) USBF DMA request enable register (UFDRQEN)**

The UFDRQEN register specifies the DMA channel to be used and the endpoint to be transferred.

The UFDRQEN register can be read or written in 8-bit or 16-bit units.

(1/2)

After reset: 0000H    R/W    Address: 00240000H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| UFDRQEN | RQ3UR3E | RQ2UR3E | RQ1UR3E | RQ0UR3E | RQ3UR2E | RQ2UR2E | RQ1UR2E | RQ0UR2E |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RQ3UR1E | RQ2UR1E | RQ1UR1E | RQ0UR1E | RQ3UR0E | RQ2UR0E | RQ1UR0E | RQ0UR0E |

| Bit position | Bit name | Function |
|---|---|---|
| 15, 11, 7, 3 | RQ3UR3E, RQ3UR2E, RQ3UR1E, RQ3UR0E | Specify the endpoint n (EPn) to be transferred by DMA channel 3. (n = 1 to 4) <br><br> |

| RQ3UR3E | RQ3UR2E | RQ3UR1E | RQ3UR0E | EP transferred by DMA3 |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | EP4 |
| 0 | 1 | 0 | 0 | EP3 |
| 0 | 0 | 1 | 0 | EP2 |
| 0 | 0 | 0 | 1 | EP1 |
| Other than above | | | | DMA3 does not transfer EPn (DMA3 not used) |

| Bit position | Bit name | Function |
|---|---|---|
| 14, 10, 6, 2 | RQ2UR3E, RQ2UR2E, RQ2UR1E, RQ2UR0E | Specify the endpoint n (EPn) to be transferred by DMA channel 2. (n = 1 to 4) |

| RQ2UR3E | RQ2UR2E | RQ2UR1E | RQ2UR0E | EP transferred by DMA2 |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | EP4 |
| 0 | 1 | 0 | 0 | EP3 |
| 0 | 0 | 1 | 0 | EP2 |
| 0 | 0 | 0 | 1 | EP1 |
| Other than above | | | | DMA2 does not transfer EPn (DMA2 not used) |

(2/2)

| Bit position | Bit name | Function | | | | |
|---|---|---|---|---|---|---|
| 13, 9, 5, 1 | RQ1UR3E, RQ1UR2E, RQ1UR1E, RQ1UR0E | Specify the endpoint n (EPn) to be transferred by DMA channel 1. (n = 1 to 4) | | | | |
| | | RQ1UR3E | RQ1UR2E | RQ1UR1E | RQ1UR0E | EP transferred by DMA2 |
| | | 1 | 0 | 0 | 0 | EP4 |
| | | 0 | 1 | 0 | 0 | EP3 |
| | | 0 | 0 | 1 | 0 | EP2 |
| | | 0 | 0 | 0 | 1 | EP1 |
| | | Other than above | | | | DMA1 does not transfer EPn (DMA1 not used) |
| 12, 8, 4, 0 | RQ0UR3E, RQ0UR2E, RQ0UR1E, RQ0UR0E | Specify the endpoint n (EPn) to be transferred by DMA channel 0. (n = 1 to 4) | | | | |
| | | RQ0UR3E | RQ0UR2E | RQ0UR1E | RQ0UR0E | EP transferred by DMA0 |
| | | 1 | 0 | 0 | 0 | EP4 |
| | | 0 | 1 | 0 | 0 | EP3 |
| | | 0 | 0 | 1 | 0 | EP2 |
| | | 0 | 0 | 0 | 1 | EP1 |
| | | Other than above | | | | DMA0 does not transfer EPn (DMA0 not used) |

**Cautions 1.** **Setting the same DMA transfer target to multiple DMA channels, and setting multiple DMA transfer targets to the same DMA channel are prohibited.**

**2.** **If using the function of this register, set the DMA trigger factor register (DTFRn (n = 0 to 3)) to disable DMA requests by interrupt (00H).**

The following flowcharts illustrate the program execution when the host is disconnected and then reconnected, and the program execution when power is supplied.

**Figure 20-12.  Flowchart of Program When Host Is Disconnected and Then Reconnected**
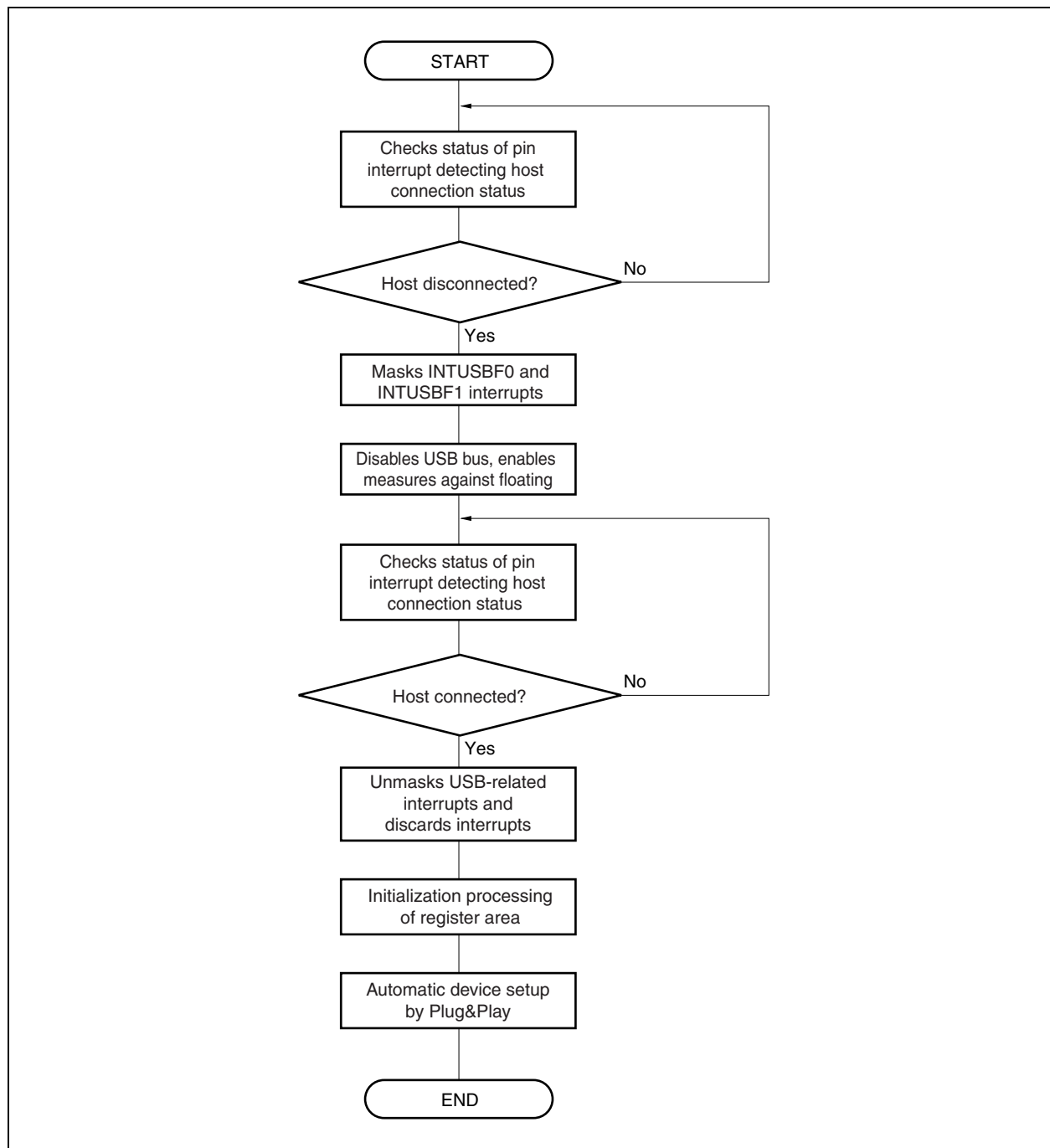
**Figure 20-13.  Flowchart of Program When Power Is Supplied**
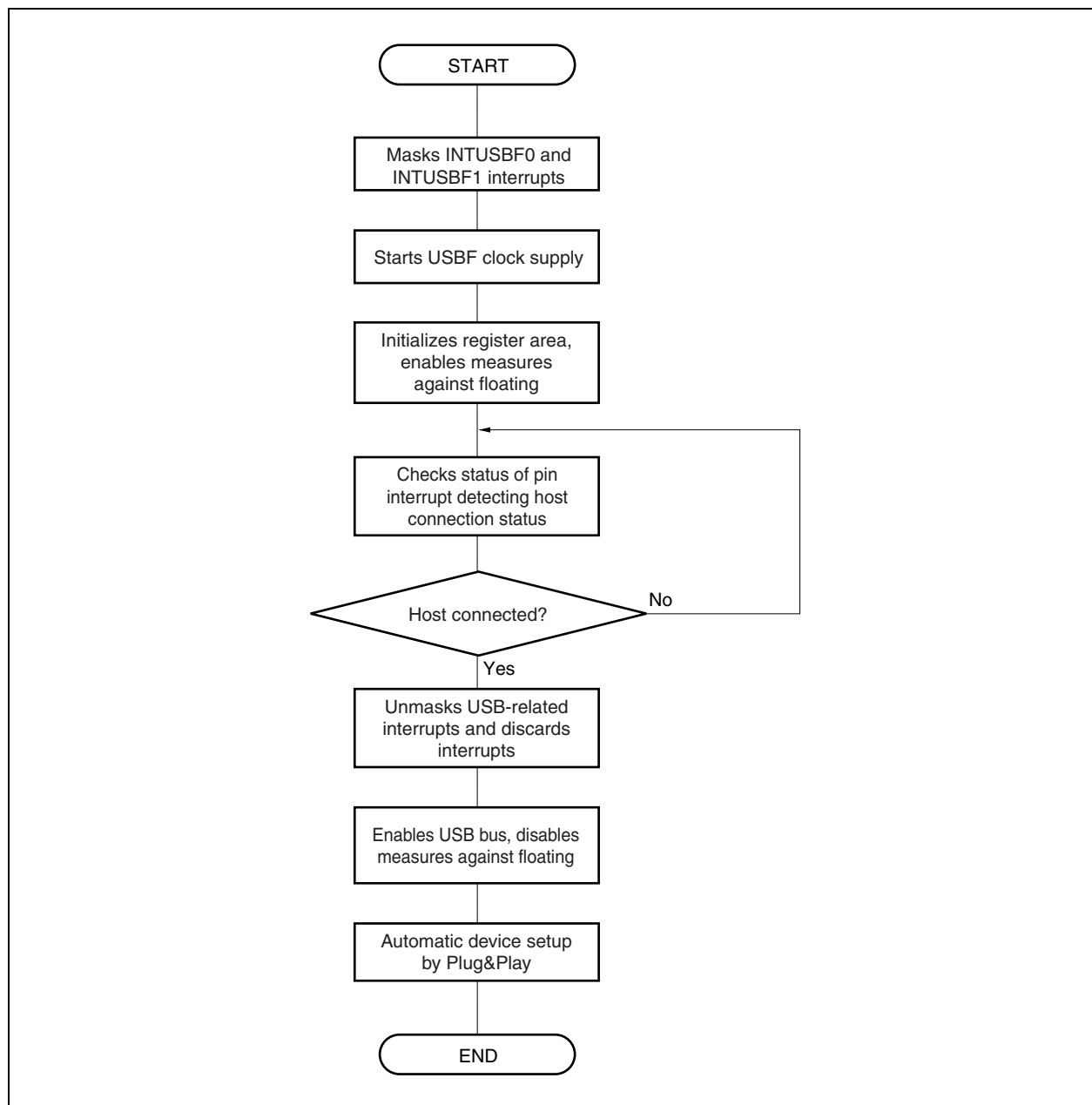
## 20.7  STALL Handshake or No Handshake

Errors of USBF are defined to be handled as follows.

| Transfer Type | Transaction | Target Packet | Error Type | Function Response | Processing |
|---|---|---|---|---|---|
| Control transfer/ bulk transfer/ interrupt transfer | IN/OUT/SETUP | Token | Endpoint not supported | No response | None |
| | | | Endpoint transfer direction mismatch | No response | None |
| | | | CRC error | No response | None |
| | | | Bit stuffing error | No response | None |
| Control transfer/ bulk transfer | OUT/SETUP | Data | Timeout | No response | None |
| | | | PID check error | No response | None |
| | | | Unsupported PID (other than Data PID) | No response | None |
| | | | CRC error | No response | Discard received data |
| | | | Bit stuffing error | No response | Discard received data |
| | OUT | Data | Data PID mismatch | ACK | Discard received data |
| Control transfer (SETUP stage) | SETUP | Data | Overrun | No response | Discard received data |
| Control transfer (data stage) | OUT | Data | Overrun | No response[Note 1] | Set SNDSTL bit of UF0SDS register to 1 and discard received data |
| Control transfer (status stage) | OUT | Data | Overrun | ACK or no response[Note 2] | Set SNDSTL bit of UF0SDS register to 1 and discard received data |
| Bulk transfer | OUT | Data | Overrun | No response[Note 1] | Set EnHALT bit of UF0EnSL register (n = 0 to 4, 7) to 1 |
| Control transfer/ bulk transfer/ interrupt transfer | IN | Handshake | PID check error | – | Hold transferred data and re-transfer data[Note 3] |
| | | | Unsupported PID (other than ACK PID) | – | Hold transferred data and re-transfer data[Note 3] |
| | | | Timeout | – | Hold transferred data and re-transfer data[Note 3] |

**Notes 1.** A STALL response is made to re-transfer by the host.

**2.** An ACK response is made if the transfer data is of less than MaxPacketSize and the data received in the status stage is discarded.  If MaxPacketSize is exceeded, no response is made, the SNDSTL bit of the UF0SDS register is set to 1, and the received data is discarded.

**3.** If an OUT transaction indicating a change from the data stage to the status stage is received during control transfer, an error is not handled and it is assumed that reception has been correctly completed.

**Cautions 1.  It is judged by the Alternative Setting number currently set whether the target Endpoint is valid or invalid.**

**2.  For the response to the request included in control transfer to/from Endpoint0, see 20.5  Requests.**

## 20.8  Register Values in Specific Status

**Table 20-8.  Register Values in Specific Status (1/2)**

| Register Name | After CPU Reset (RESET) | After Bus Reset |
|---|---|---|
| UF0E0N register | 00H | Value is held. |
| UF0E0NA register | 00H | Value is held. |
| UF0EN register | 00H | Value is held. |
| UF0ENM register | 00H | Value is held. |
| UF0SDS register | 00H | Value is held. |
| UF0CLR register | 00H | Value is held. |
| UF0SET register | 00H | Value is held. |
| UF0EPS0 register | 00H | Value is held. |
| UF0EPS1 register | 00H | Value is held. |
| UF0EPS2 register | 00H | Value is held. |
| UF0IS0 register | 00H | Value is held. |
| UF0IS1 register | 00H | Value is held. |
| UF0IS2 register | 00H | Value is held. |
| UF0IS3 register | 00H | Value is held. |
| UF0IS4 register | 00H | Value is held. |
| UF0IM0 register | 00H | Value is held. |
| UF0IM1 register | 00H | Value is held. |
| UF0IM2 register | 00H | Value is held. |
| UF0IM3 register | 00H | Value is held. |
| UF0IM4 register | 00H | Value is held. |
| UF0IC0 register | FFH | Value is held. |
| UF0IC1 register | FFH | Value is held. |
| UF0IC2 register | FFH | Value is held. |
| UF0IC3 register | FFH | Value is held. |
| UF0IC4 register | FFH | Value is held. |
| UF0IDR register | 00H | Value is held. |
| UF0DMS0 register | 00H | Value is held. |
| UF0DMS1 register | 00H | Value is held. |
| UF0FIC0 register | 00H | Value is held. |
| UF0FIC1 register | 00H | Value is held. |
| UF0DEND register | 00H | Value is held. |
| UF0GPR register | 00H | Value is held. |
| UF0MODC register | 00H | Value is held. |
| UF0MODS register | 00H | Bit 2 (CONF): Cleared (0), Other bits: Value is held. |
| UF0AIFN register | 00H | Value is held. |
| UF0AAS register | 00H | Value is held. |
| UF0ASS register | 00H | 00H |
| UF0E1IM register | 00H | Value is held. |
| UF0E2IM register | 00H | Value is held. |

**Table 20-8.  Register Values in Specific Status (2/2)**

| Register Name | After CPU Reset ($\overline{\text{RESET}}$) | After Bus Reset |
|---|---|---|
| UF0E3IM register | 00H | Value is held. |
| UF0E4IM register | 00H | Value is held. |
| UF0E7IM register | 00H | Value is held. |
| UF0E0R register | Undefined[Note 1] | Value is held. |
| UF0E0L register | 00H | Value is held. |
| UF0E0ST register | 00H | 00H |
| UF0E0W register | Undefined[Note 1] | Value is held. |
| UF0BO1 register | Undefined[Note 1] | Value is held. |
| UF0BO1L register | 00H | Value is held. |
| UF0BO2 register | Undefined[Note 1] | Value is held. |
| UF0BO2L register | 00H | Value is held. |
| UF0BI1 register | Undefined[Note 1] | Value is held. |
| UF0BI2 register | Undefined[Note 1] | Value is held. |
| UF0INT1 register | Undefined | Value is held. |
| UF0DSTL register | 00H | 00H |
| UF0E0SL register | 00H | 00H |
| UF0E1SL register | 00H | 00H |
| UF0E2SL register | 00H | 00H |
| UF0E3SL register | 00H | 00H |
| UF0E4SL register | 00H | 00H |
| UF0E7SL register | 00H | 00H |
| UF0ADRS register | 00H | 00H |
| UF0CNF register | 00H | 00H |
| UF0IF0 register | 00H | 00H |
| UF0IF1 register | 00H | 00H |
| UF0IF2 register | 00H | 00H |
| UF0IF3 register | 00H | 00H |
| UF0IF4 register | 00H | 00H |
| UF0DSCL register | 00H | Value is held. |
| UF0DDn register (n = 0 to 17) | **Note 2** | **Note 2** |
| UF0CIEn register (n = 0 to 255) | **Note 2** | **Note 2** |

**Notes 1.** This register can be cleared to 0 by the $\overline{\text{RESET}}$ signal because its write pointer, counter, and read pointer are cleared to 0 when the $\overline{\text{RESET}}$ signal becomes active, in the same manner as clearing by the UF0FICn register, as the register is controlled by FIFO.

**2.** This register cannot be cleared to 0.  Because data can be written to it by FW, however, any value can be written to the register (before doing so, however, be sure to set the EP0NKA bit of the UF0E0NA register to 1).

## 20.9 FW Processing

The following FW processing is performed.

- Setting processing on device side for the SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, and CLEAR_FEATURE requests during enumeration processing
- Analysis and processing of XXXXStandard, XXXXClass, and XXXXVendor requests not subject to automatic processing
- Reading data following bulk-transferred OUT token from receive buffer
- Writing data to be returned in response to bulk-transferred IN token
- Writing data to be returned in response to interrupt-transferred token

The following table lists the requests supported by FW.

**Table 20-9. FW-Supported Standard Requests**

| Request | Reception Side | Processing/ Frequency | Explanation |
|---|---|---|---|
| CLEAR_FEATURE | Interface | Automatic STALL response | It is considered that this request does not come to Interface because there is no function selector value, though it is reserved for bmRequestType.<br>When this request is received, the hardware makes an automatic STALL response. |
| SET_FEATURE | Interface | Automatic STALL response | It is considered that this request does not come to Interface because there is no function selector value, though it is reserved for bmRequestType.<br>When this request is received, the hardware makes an automatic STALL response. |
| GET_DESCRIPTOR | String | FW | Returns the string descriptor.<br>When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and writes the data to be returned to the host, to the UF0E0W register. |
| SET_DESCRIPTOR | Device | FW | Rewrites the device descriptor.<br>When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and the writes the data for the next control transfer (OUT) to the UF0DDn register (n = 0 to 17). |
| SET_DESCRIPTOR | Configuration | FW | Rewrites the configuration descriptor.<br>When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and the writes the data for the next control transfer (OUT) to the UF0CIEn register (n = 0 to 255). |
| SET_DESCRIPTOR | String | FW | Rewrites the string descriptor.<br>When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and loads the data for the next control transfer (OUT). |
| Other | NA | FW | When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and performs the necessary processing. |

### 20.9.1  Initialization processing

Initialization processing is executed in the following two ways.

- Initialization of request data register
- Setting of interrupt

When a request data register is initialized, data for the GET_XXXX request to which a value is to be automatically returned is written and an endpoint is allocated to an interface.  In the interrupt settings, the interrupt sources that do not have to be checked can be masked by using the UF0IMn register (n = 0 to 4).

The following flowcharts illustrate the above processing.

**Figure 20-14.  Initializing Request Data Register**



**Remark**   n = 0 to 255

**Figure 20-15. Initialization Settings of Request Data Register**



UF0DSTL register = 0XH — The value of 0XH depends on the power supply method.
• SFPW = 1: Self-powered
• SFPW = 0: Bus-powered

UF0EnSL register = 00H — n = 0 to 4, or 7. Setting is unnecessary if the target endpoint is not used.

Setting of UF0DSCL register — Input the total number of bytes of the UF0CIEa register.

Inputting UF0DDm register

Inputting UF0CIEa register — If the total number of bytes of the UF0CIEa register exceeds 256, set the UF0MODC register to 40H. No data has to be written to the UF0CIEa register.

**Remark** m = 0 to 17
a = 0 to 255

**Figure 20-16. Setting of Interface and Endpoint**



Setting of UF0AIFN register — ADDIF, IFNO1, IFNO0 = 000: Interface number 0 is valid.
ADDIF, IFNO1, IFNO0 = 100: Interface numbers 0 and 1 are valid.
ADDIF, IFNO1, IFNO0 = 101: Interface numbers 0 to 2 are valid.
ADDIF, IFNO1, IFNO0 = 110: Interface numbers 0 to 3 are valid.
ADDIF, IFNO1, IFNO0 = 111: Interface numbers 0 to 4 are valid.

Setting of UF0AAS register — Set Interface number(s) and a link with the 5- or 2-series Alternative Setting.

Setting of UF0EnIM register — Set a link between the target Interface of endpoint n and Alternative Setting. Set 00H if the target endpoint is not used.

**Remark** n = 1 to 4, 7

**Figure 20-17.  Setting of Interrupt**

```
                 ┌─────────────────────┐
                 │        START        │
                 └─────────────────────┘
                            │
         ┌──────────────────────────────┐   Mask the interrupt source to avoid issuance of an unnecessary
         │   Setting of UF0IMn register │   interrupt request (INTUSBF0).
         └──────────────────────────────┘
                            │
                 ┌─────────────────────┐
                 │         END         │
                 └─────────────────────┘


   Remark    n = 0 to 4
```
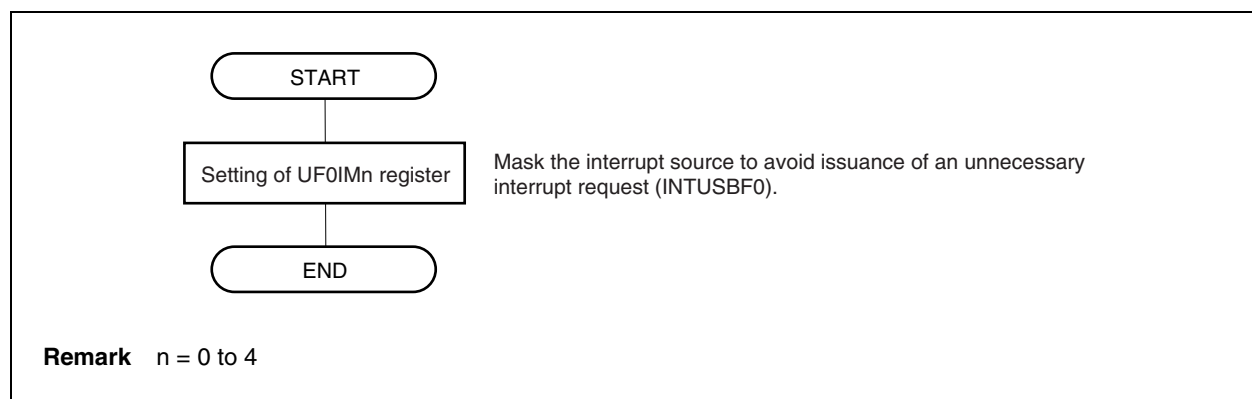
### 20.9.2 Interrupt servicing

The following flowchart illustrates how an interrupt is serviced.

**Figure 20-18. Interrupt Servicing**



Remark   ♦ : Processing by hardware
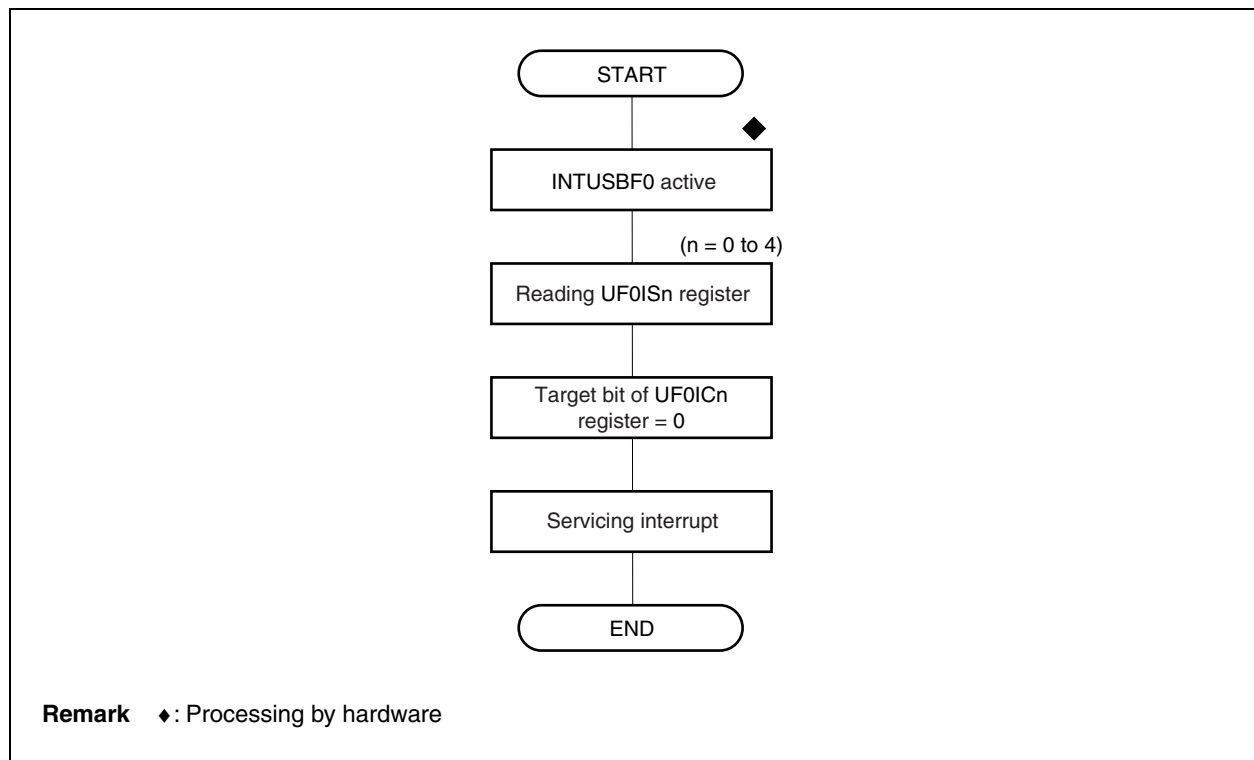
The following bits of the UF0ISn register are automatically cleared by hardware when a given condition is satisfied (n = 0 to 4).

- E0INDT, E0ODT, SUCES, STG, and CPUDEC bits of UF0IS1 register
- BKI2DT, BKI1DT, and IT1DT bits of UF0IS2 register
- BKO2FL, BKO2DT, BKO1FL, and BKO1DT bits of UF0IS3 register

Because clearing an interrupt source by the UF0ICn register is given a lower priority than setting an interrupt source by hardware, the interrupt source may not be cleared depending on the timing (n = 0 to 4).

### 20.9.3  USB main processing

USB main processing involves processing USB transactions. The types of transactions to be processed are as follows.

- Fully automatically processed request for control transfer
- Automatically processed requests for control transfer
  (SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, CLEAR_FEATURE)
- CPUDEC request for control transfer
- Processing for bulk transfer (IN)
- Processing for bulk transfer (OUT)
- Processing for interrupt transfer (IN)

Processing for endpoint n involves writing or reading for data transfer. The flowchart shown below is for PIO.

### (1)  Fully automatically processed request for control transfer

Because the fully automatically processed request for control transfer is executed by hardware, it cannot be referenced by FW. Therefore, FW does not have to perform any special processing for this request.

### (2)  Automatically processed requests for control transfer

(SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, CLEAR_FEATURE)

Processing to write a register for automatically processed requests for control transfer, such as SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, and CLEAR_FEATURE requests, is automatically executed by hardware, but an interrupt request is issued for recognition on the device side. This processing may be ignored if there is no special processing to be executed.

The flowcharts are shown below.

**Figure 20-19. Automatically Processed Requests for Control Transfer**



**Remark** ♦ : Processing by hardware
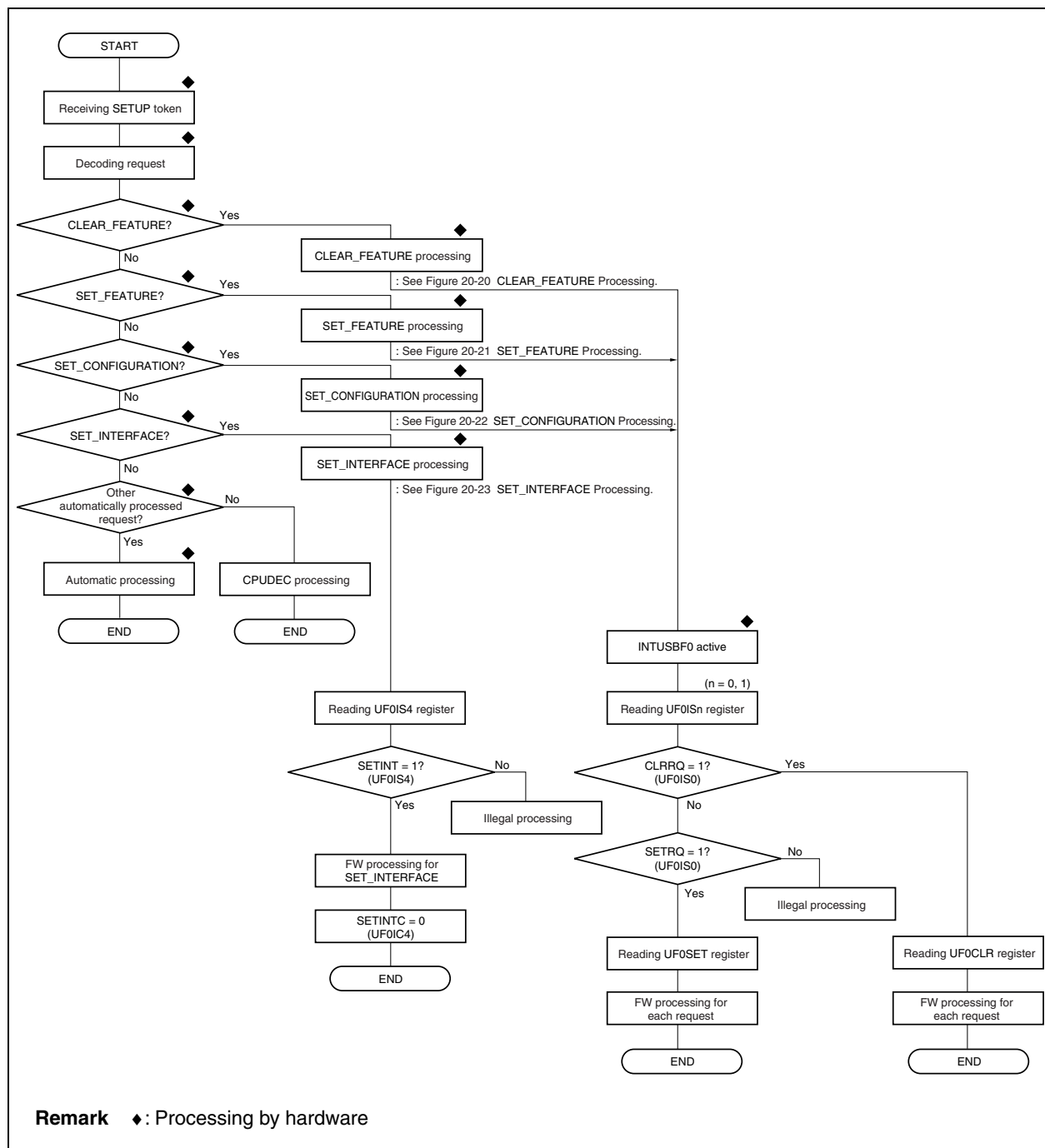
**Figure 20-20.  CLEAR_FEATURE Processing**



**Remarks 1.**   n = 0 to 4, 7
      **2.**   ◆ : Processing by hardware
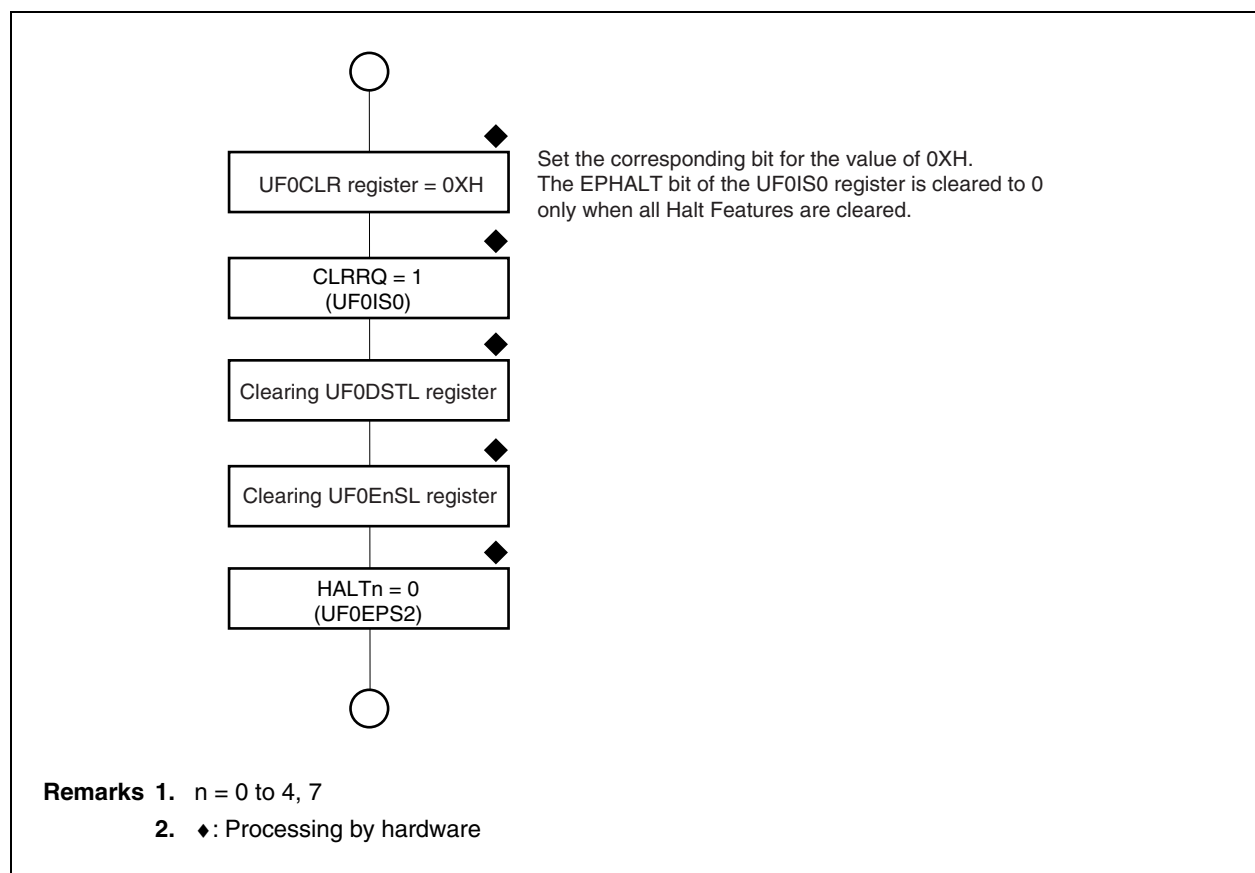
**Figure 20-21.  SET_FEATURE Processing**



Set the corresponding bit for the value of 0XH.
The EPHALT bit of the UF0IS0 register is not
set to 1 by setting the UF0DSTL register.

UF0SET register = 0XH

SETRQ = 1
(UF0IS0)

Setting UF0DSTL register

Setting UF0EnSL register

HALTn = 1
(UF0EPS2)

EPHALT = 1
(UF0IS0)

**Remarks 1.**  n = 0 to 4, 7

 **2.**  ♦: Processing by hardware
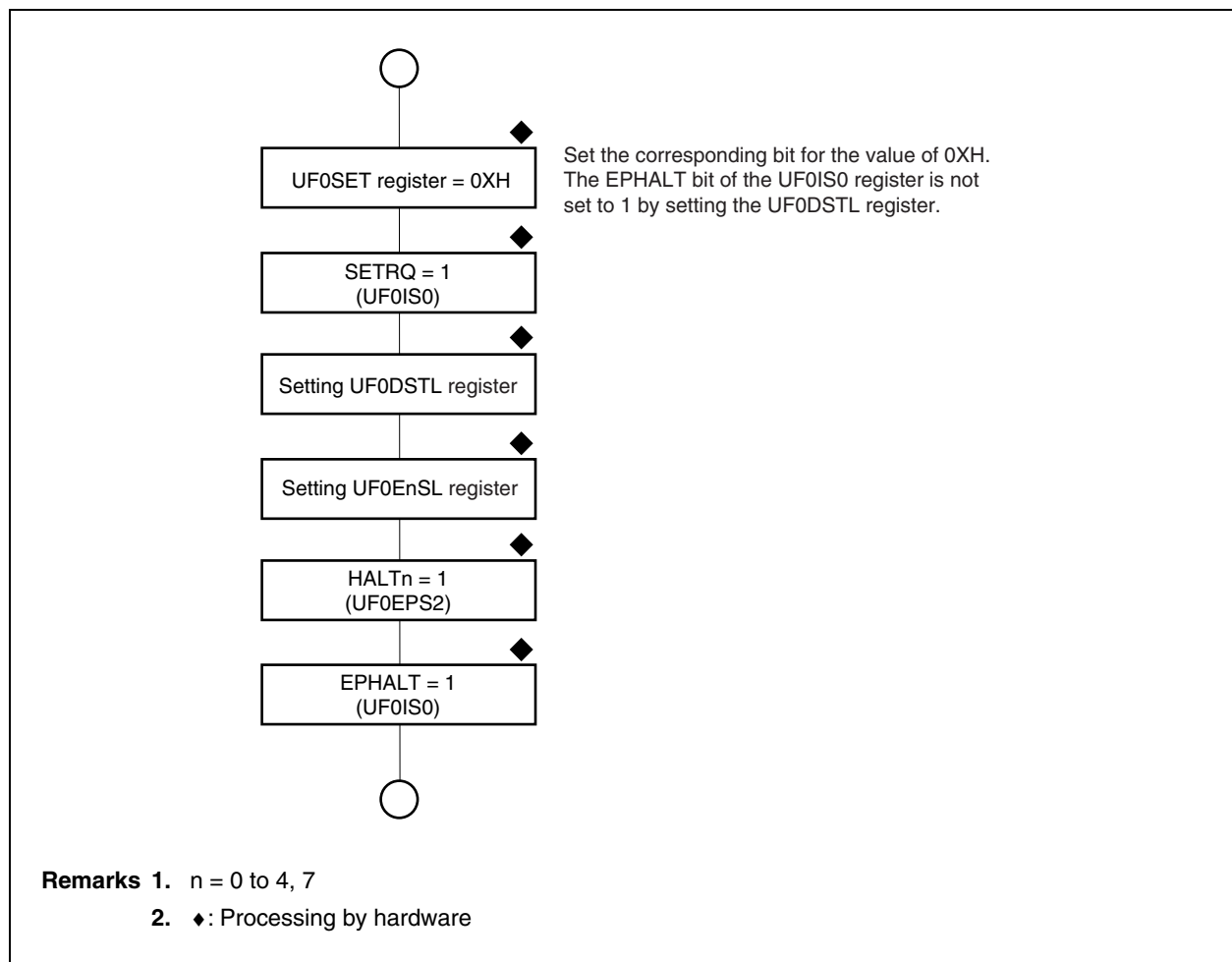
**Figure 20-22. SET_CONFIGURATION Processing**



**Remark** ♦: Processing by hardware
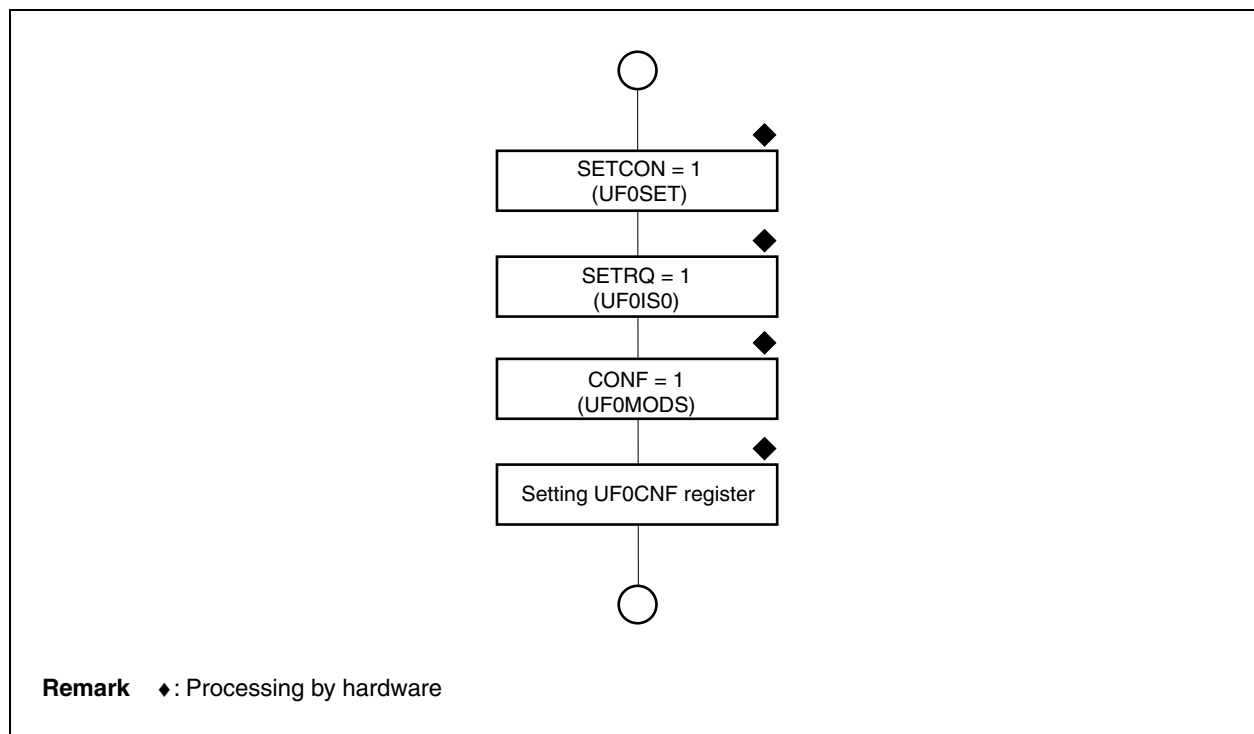
**Figure 20-23. SET_INTERFACE Processing**



**Remarks 1.** n = 0 to 4
      **2.** ♦: Processing by hardware
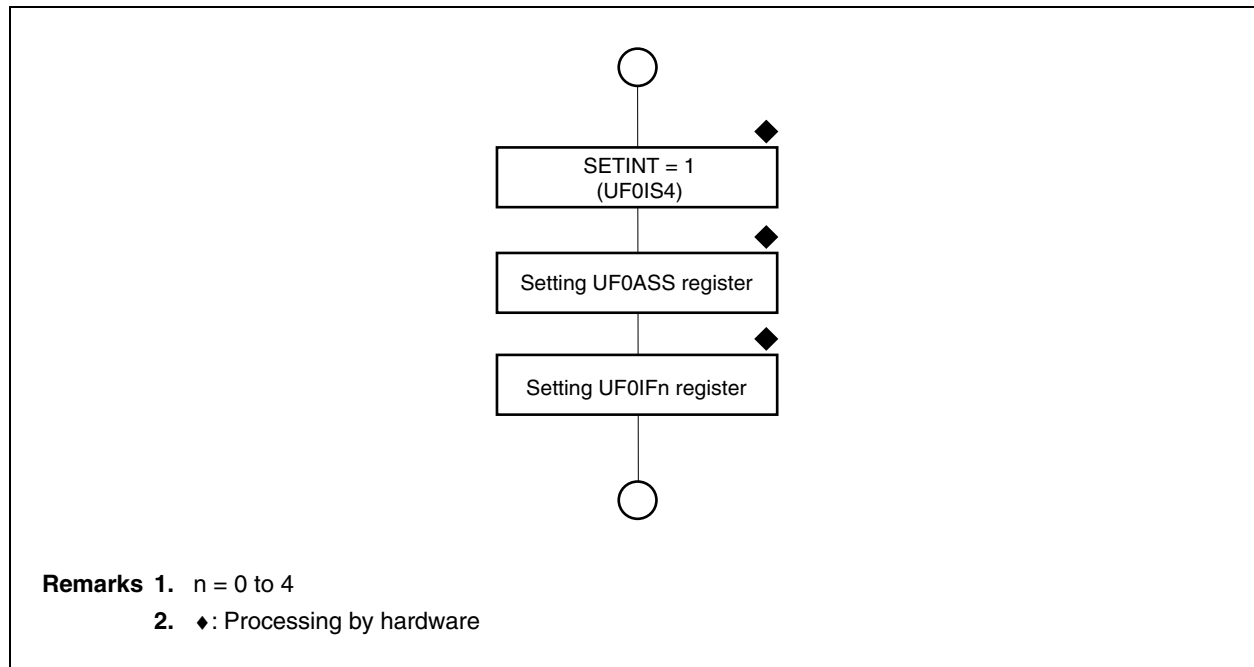
**(3) CPUDEC request for control transfer**

The CPUDEC request can be classified into three types of processing: control transfer (write), control transfer (read), and control transfer (without data).  Control transfer (write) indicates a request that uses the OUT transaction in the data stage (e.g., SET_DESCRIPTOR), and control transfer (read) indicates a request that uses the IN transaction in the data stage (e.g., GET_DESCRIPTOR). Control transfer (without data) indicates a request that has no data stage (e.g., SET_CONFIGURATION).

The flowcharts are shown below.

**Figure 20-24.  CPUDEC Request for Control Transfer (1/12)**



**(a)  Token phase (1/2)**

**Remarks 1.**  n = 0, 1
**2.**  ◆: Processing by hardware

**Figure 20-24.  CPUDEC Request for Control Transfer (2/12)**



(a)  Token phase (2/2)

It is judged whether the request decoded by the device is supported.

Request that uses control transfer (IN), such as GET_DESCRIPTOR String

Request that uses control transfer (OUT), such as SET_DESCRIPTOR String

In the case of an unsupported request for control transfer (write), clear the FIFO because data may be written to the FIFO as a result of OUT transfer before the STALL response is made.

**Remarks 1.** n = 0, 1
**2.**  ◆: Processing by hardware

**Figure 20-24.  CPUDEC Request for Control Transfer (3/12)**

**(b)  Control transfer (read) (1/4)**

```
                    ( B )
                      |
                      |←─────────────────────────┐
                      ▼                           │
              ╱───────────────╲                   │
             ╱  CPUDEC = 1?  ◆  ╲      No          │
             ╲   (UF0IS1)      ╱ ──────────────────┘
              ╲───────────────╱
                      │ Yes       ◆
                      ▼
             ┌───────────────────┐ ◆
             │  Transmitting NAK  │
             └───────────────────┘
                      │           ◆
                      ▼
             ┌───────────────────┐
             │     E0IN = 1       │
             │    (UF0IS1)        │
             └───────────────────┘
                      │    ◆
                      ▼
             ┌───────────────────┐
             │  INTUSBF0 active   │
             └───────────────────┘
                      │
                      ▼
             ┌───────────────────┐
             │ Reading UF0ISn     │
             │   register         │
             └───────────────────┘
                      │
                      ▼
              ╱───────────────╲        No
             ╱   E0IN = 1?      ╲ ──────────────┐
             ╲   (UF0IS1)      ╱                 ▼
              ╲───────────────╱          ┌──────────────────┐
                      │ Yes              │ Illegal processing│
                      ▼                  └──────────────────┘
             ┌───────────────────┐
             │    E0INM  = 1      │
             │    (UF0IM1)        │
             └───────────────────┘
                      │                  ( I )
                      │←──────────────────┘
                      ▼
             ┌───────────────────┐     If return data greater than the FIFO size exists,
             │  FW request decode │     it is divided into FIFO size units and sequentially
             └───────────────────┘     written, starting from the lowest data byte.
                      │
                      ▼
                    ( F )
```

**Remarks  1.**  n = 0, 1

   **2.**  ♦: Processing by hardware

**Figure 20-24 CPUDEC Request for Control Transfer (4/12)**

**(b) Control transfer (read) (2/4)**



**Remark** ♦ : Processing by hardware

**Figure 20-24.  CPUDEC Request for Control Transfer (5/12)**



**(b)  Control transfer (read) (3/4)**

Remarks 1.  n = 0, 1
2.  ♦: Processing by hardware

**Figure 20-24. CPUDEC Request for Control Transfer (6/12)**

**(b) Control transfer (read) (4/4)**



**Remarks 1.** n = 0, 1

     **2.** ♦: Processing by hardware

**Figure 20-24. CPUDEC Request for Control Transfer (7/12)**



**(c) Control transfer (write) (1/4)**

**Remarks 1.** n = 0, 1

      **2.** ♦ : Processing by hardware

**Figure 20-24.  CPUDEC Request for Control Transfer (8/12)**

**(c) Control transfer (write) (2/4)**



**Remark**  ♦: Processing by hardware

**Figure 20-24. CPUDEC Request for Control Transfer (9/12)**



(c) Control transfer (write) (3/4)

Remarks 1. n = 0, 1
 2. ♦: Processing by hardware

**Figure 20-24. CPUDEC Request for Control Transfer (10/12)**

**(c) Control transfer (write) (4/4)**



**Remarks 1.** n = 0, 1

**2.** ♦: Processing by hardware

**Figure 20-24.  CPUDEC Request for Control Transfer (11/12)**



**(d)  Control transfer (without data stage) (1/2)**

Remarks 1. n = 0, 1

      2.  ◆: Processing by hardware

**Figure 20-24.  CPUDEC Request for Control Transfer (12/12)**



**(d)  Control transfer (without data stage) (2/2)**

**Remarks 1.**  n = 0, 1

**2.**  ♦ : Processing by hardware

**(4) Processing for bulk transfer (IN)**

Bulk transfer (IN) is allocated to Endpoint1 and Endpoint3.  The flowchart shown below illustrates how Endpoint1 is controlled.  Endpoint3 can also be controlled in the same sequence.  To use this flowchart as the control flow of Endpoint3, therefore, read the bit names of Endpoint1 in the flowchart as those of Endpoint3.

**Figure 20-25.  Processing for Bulk Transfer (IN) (Endpoint1)**



Remarks **1.** n = 2, 3

**2.** ♦: Processing by hardware
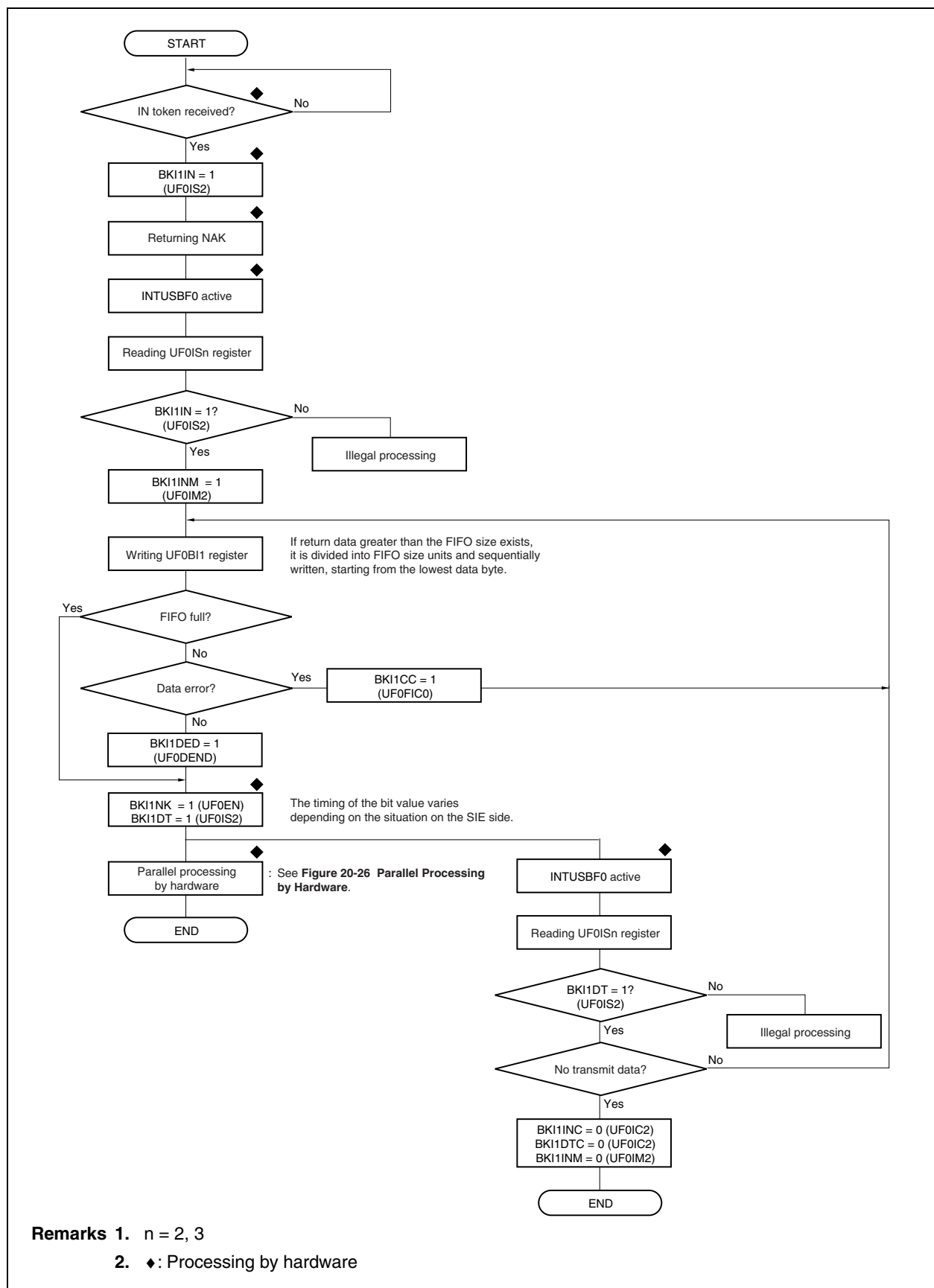
**Figure 20-26.  Parallel Processing by Hardware**
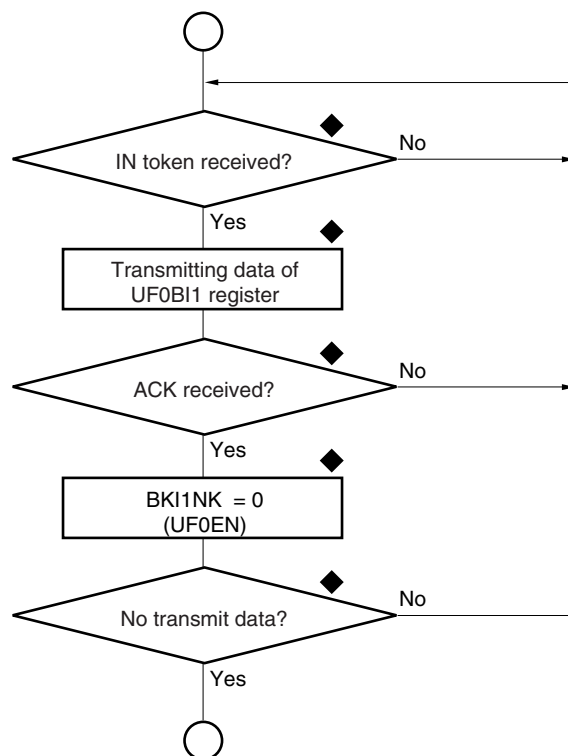


**Remark**   ♦ : Processing by hardware

**(5) Processing for bulk transfer (OUT)**

Bulk transfer (OUT) is allocated to Endpoint2 and Endpoint4. The flowchart shown below illustrates how Endpoint2 is controlled. Endpoint4 can also be controlled in the same sequence. To use this flowchart as the control flow of Endpoint4, therefore, read the bit names of Endpoint2 in the flowchart as those of Endpoint4.

**Figure 20-27.  Normal Processing for Bulk Transfer (OUT) (Endpoint2)**



**Remarks 1.** n = 2, 3

**2.** ◆: Processing by hardware
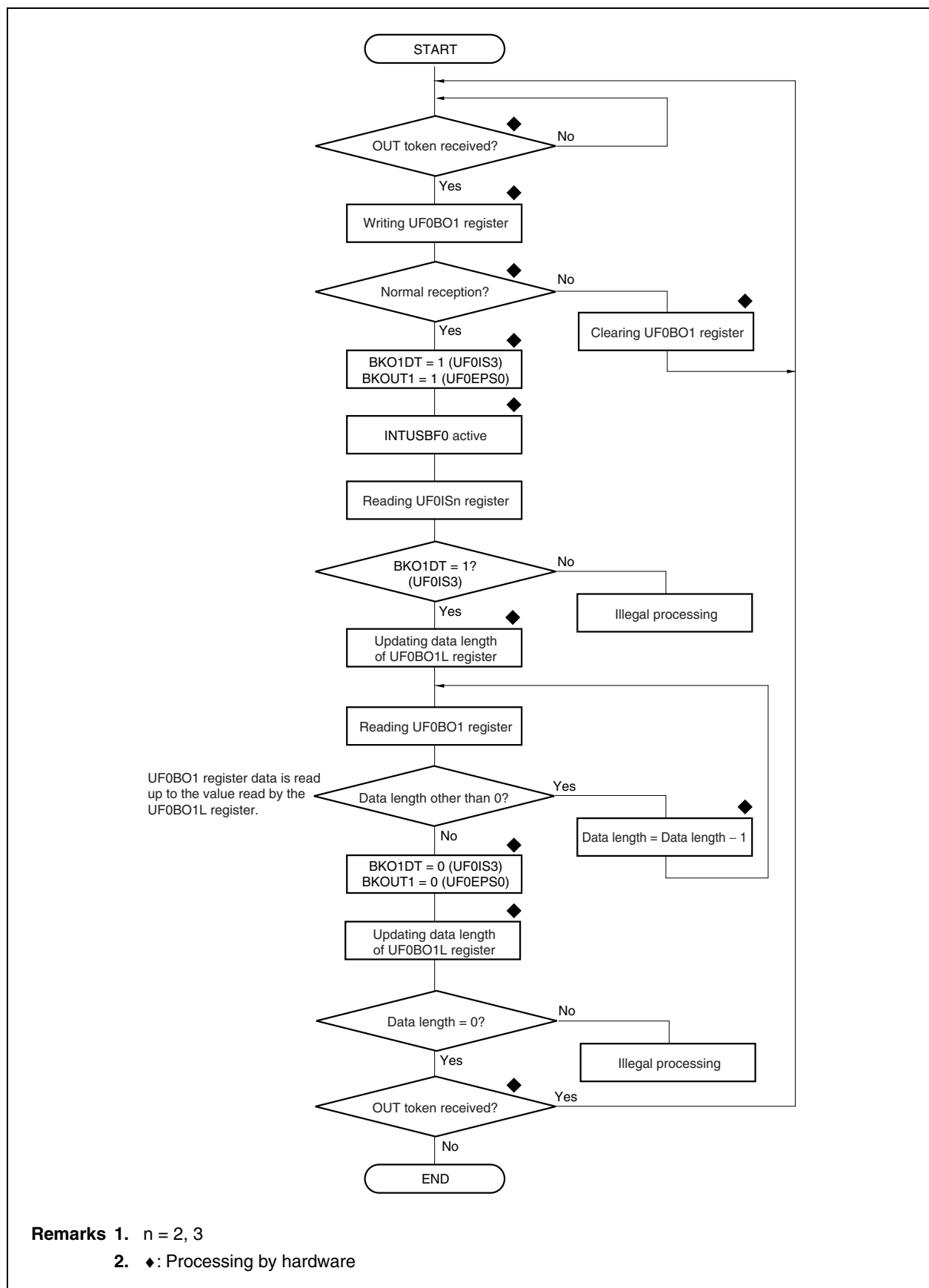
During bulk transfer (OUT), more data may be transmitted from the host than expected by the system.  Endpoint2 and Endpoint4 for bulk transfer (OUT) of the V850ES/JG3-U and V850ES/JH3-U consist of two 64-byte buffers so that NAK responses are suppressed as much as possible and data can be read from the CPU side even while the bus side is being accessed as the transfer rate of the USB bus increases.  Consequently, if the host sends more data than expected by the system, up to 128 bytes of extra data may be automatically received in the worst case. In this case, change the control flow from that of the normal processing of Endpoint2 and Endpoint4 to the flow illustrated below when the quantity of data expected by the system has decreased to two packets.  This flowchart illustrates how Endpoint2 is controlled.  Endpoint4 can also be controlled in the same sequence.  To use this flowchart as the control flow of Endpoint4, therefore, read the bit names of Endpoint2 in the flowchart as those of Endpoint4.

**Figure 20-28.  Processing If More Data Than Expected by System Is Transmitted (Endpoint2) (1/2)**
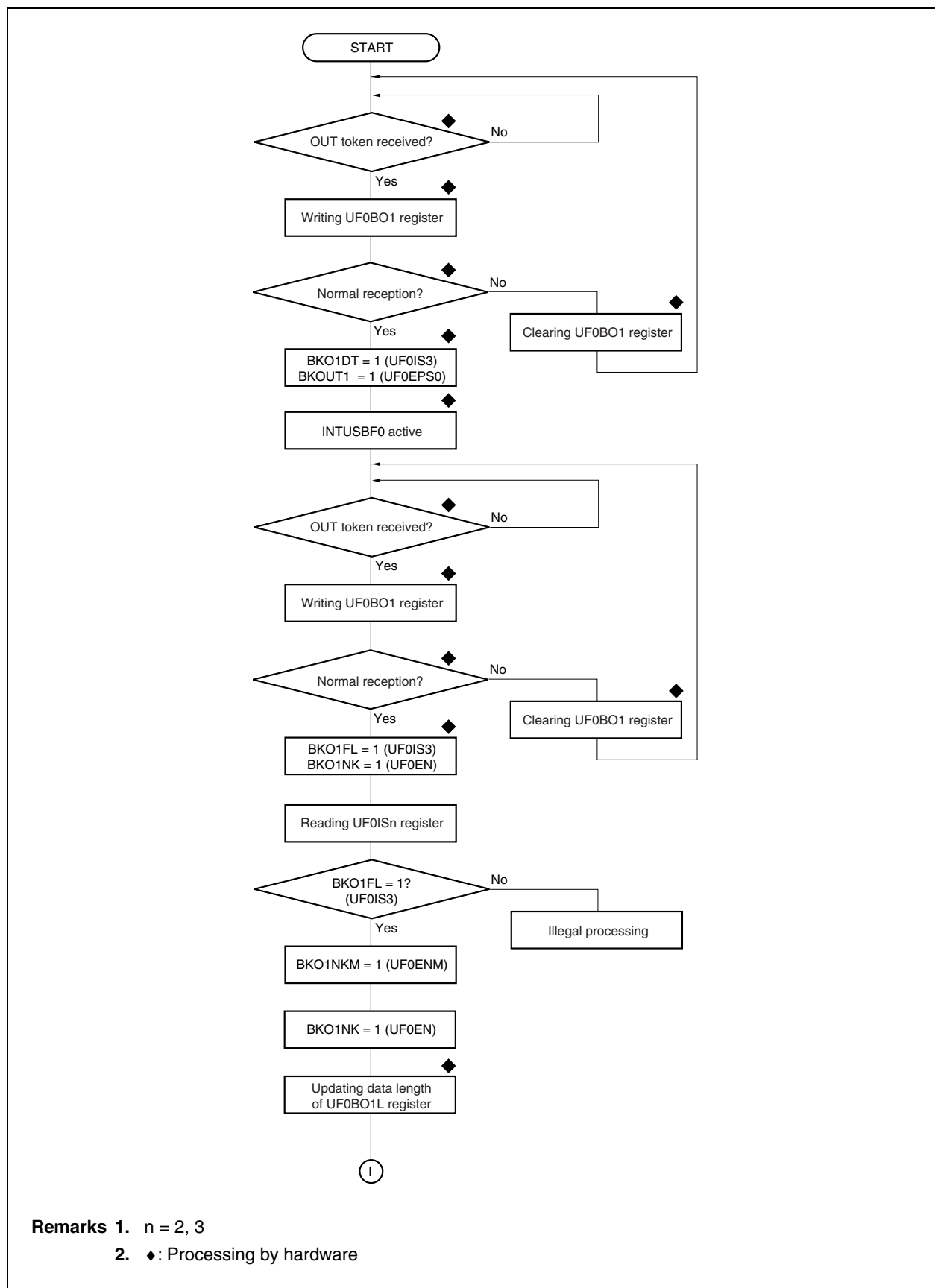


**Remarks 1.**  n = 2, 3

**2.**  ◆: Processing by hardware

**Figure 20-28.  Processing If More Data Than Expected by System Is Transmitted (Endpoint2) (2/2)**



**Remark**  ♦ : Processing by hardware
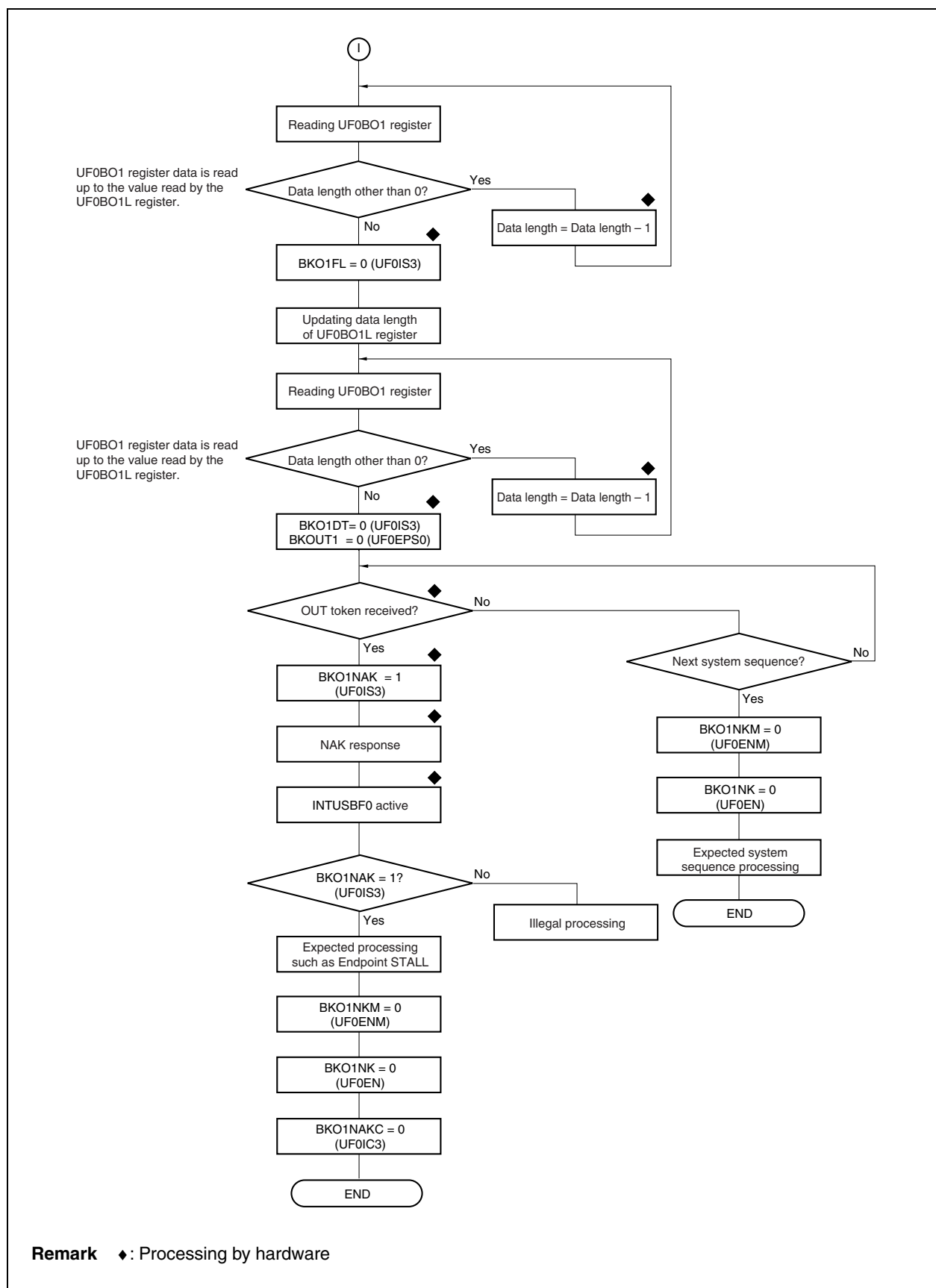
**(6) Processing for interrupt transfer (IN)**

Interrupt transfer (IN) is allocated to Endpoint7.  The flowchart is shown in **Figure 20-29**.

**Figure 20-29.  Processing for Interrupt Transfer (IN) (Endpoint7)**



Remarks **1.** n = 2, 3

**2.** ♦: Processing by hardware
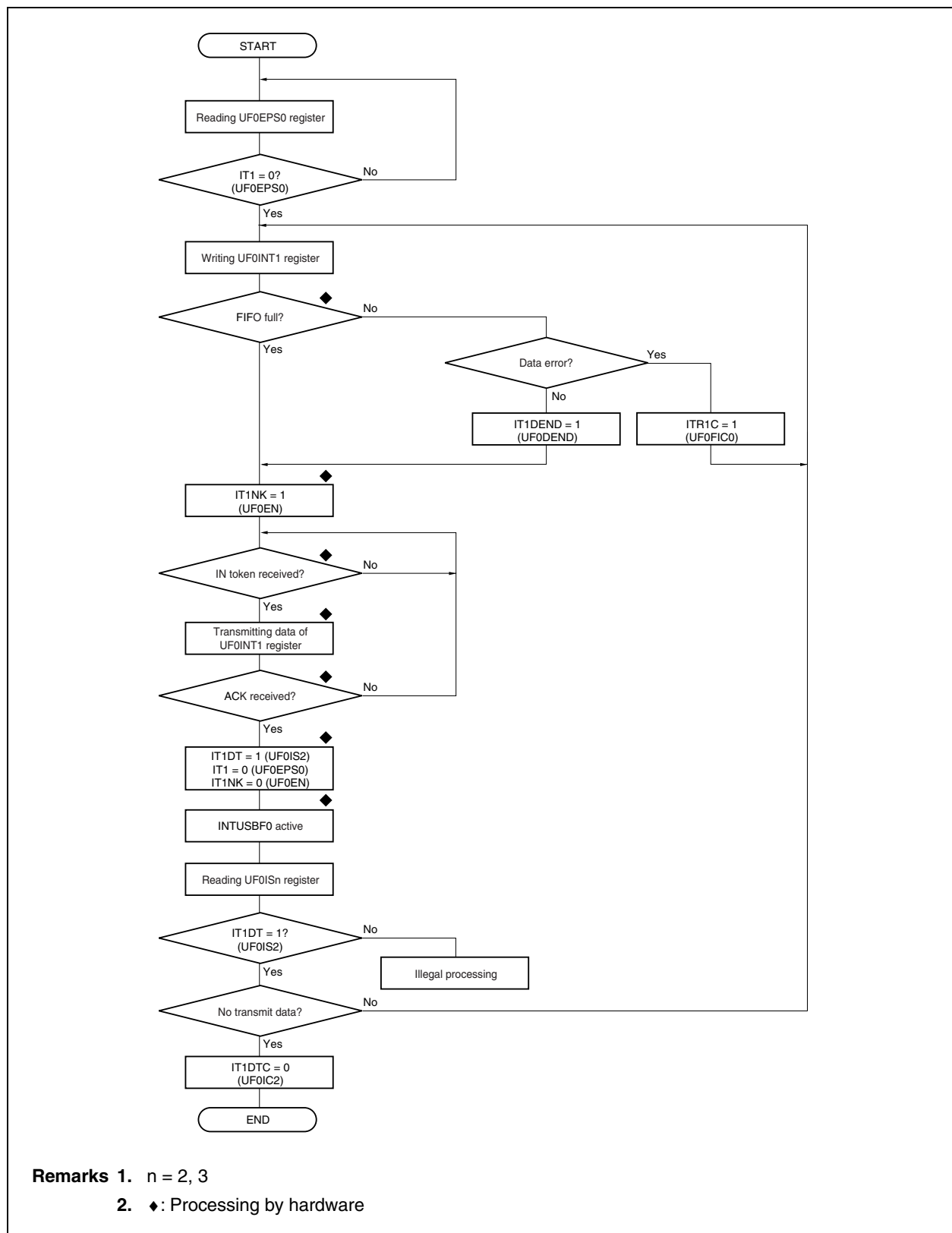
#### 20.9.4  Suspend/Resume processing

How Suspend/Resume processing is performed differs depending on the configuration of the system.  One example is given below.

**Figure 20-30.  Example of Suspend/Resume Processing (1/3)**



**Remarks 1.** n = 0, 1
   **2.** ♦: Processing by hardware

**Figure 20-30. Example of Suspend/Resume Processing (2/3)**

**(b) Example of Resume processing**

```
                          START
                            │
                            ▼
              ┌──────────◆──────────┐      No
              <   Resume detected?   >─────────────┐
              └──────────┬──────────┘              │
                     Yes │ ◆                        │
                         ▼                          │
              ┌──────────────────────┐             │
              │   RSUSPD = 1 (UF0IS0) │             │
              │   RSUM = 0 (UF0EPS1)  │             │
              └──────────┬───────────┘             │
                         │ ◆                        │
                         ▼                          │
              ┌──────────────────────┐             │
              │    INTUSBF0 active    │             │
              └──────────┬───────────┘             │
                         │                          │
                         ▼                          │
              ┌──────────────────────┐             │
              │  Reading UF0ISn register │          │
              └──────────┬───────────┘             │
                         │                          │
                         ▼                          │
              ┌──────────────────────┐     No       │
              <      RSUSPD = 1?      >──────────┐   │
              <       (UF0IS0)        >          │   │
              └──────────┬───────────┘          ▼   │
                     Yes │              ┌───────────────────┐
                         ▼              │ Illegal processing │
              ┌──────────────────────┐  └───────────────────┘
              │ Reading UF0EPS1 register│
              └──────────┬───────────┘
                         │
                         ▼
              ┌──────────────────────┐     No
              <      RSUM = 0?        >──────────┐
              <      (UF0EPS1)        >          ▼
              └──────────┬───────────┘   ┌───────────────────┐
                     Yes │               │ Illegal processing │
                         ▼               └───────────────────┘
              ┌──────────────────────┐
              │  FW Resume processing │
              └──────────┬───────────┘
                         ▼
              ┌──────────────────────┐
              │    RSUSPDC = 0        │
              │    (UF0IC0)           │
              └──────────┬───────────┘
                         ▼
                        END
```

**Remarks 1.** n = 0, 1

        **2.** ♦ : Processing by hardware

**Figure 20-30.  Example of Suspend/Resume Processing (3/3)**

**(c)  Example of Resume processing (when supply of USB clock to USBF is stopped)**

```
                    ┌─────────────┐
                    │    START    │
                    └──────┬──────┘
                           │ ◄──────────────┐
                           ▼                 │
                      ◆                       │
                 ╱───────────╲    No          │
                ╱ Resume      ╲───────────────┘
                ╲ detected?   ╱
                 ╲───────────╱
                      │ Yes
                      ▼
            ┌───────────────────┐
            │  INTUSBF1 active  │
            └─────────┬─────────┘
                      ▼
            ┌───────────────────────────┐
            │ Executing interrupt servicing │
            └─────────┬─────────────────┘
                      ▼
            ┌───────────────────┐
            │ Supplying USB clock │
            └─────────┬─────────┘
                      ▼
            ┌───────────────────────┐
            │ FW Resume processing  │
            └─────────┬─────────────┘
                      ▼
                ┌──────────┐
                │   END    │
                └──────────┘
```

**Remark**   ♦ : Processing by hardware

#### 20.9.5 Processing after power application

The processing to be performed after power application differs depending on the configuration of the system. One example is given below.

**Figure 20-31. Example of Processing After Power Application/Power Failure (1/3)**

**(a) Processing after power application (1/2)**

START
→ ◆ Pull-up processing of D+ inactive[Note 1]
→ ◆ Initialization of request data register : See **Figure 20-15 Initialization Settings of Request Data Register**.
→ ◆ Controlling port[Note 2]
→ ◆ Pull-up processing of D+ active[Note 1]
→ ◆ Connection
→ Resume detected?
  - No → (loop back)
  - Yes → ◆ BUSRST = 1 (UF0IS0) DFLT = 1 (UF0MODS)
→ (a)

START
→ Initialization of request data register : See **Figure 20-15 Initialization Settings of Request Data Register**.
→ Controlling port[Note 2]

**Notes 1.** Use one general-purpose port pin for the signal that controls switching of the pull-up resistor of the USB bus.

**2.** The input mode or control mode of the general-purpose port pin allocated in **Note 1** may be selected as the default value. Note the active level of pull-up processing of D+ on power application.

**Remark** ◆ : Processing by hardware

**Figure 20-31. Example of Processing After Power Application/Power Failure (2/3)**



**(a) Processing after power application (2/2)**
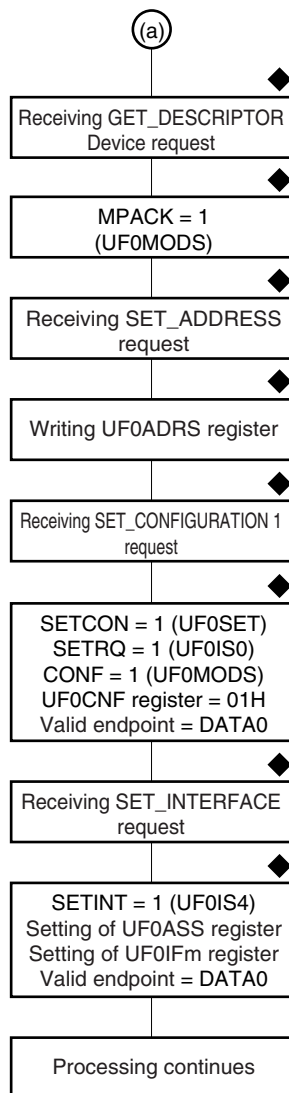
**Remarks 1.** m = 0 to 4

**2.** ♦: Processing by hardware

**Figure 20-31. Example of Processing After Power Application/Power Failure (3/3)**

**(b) Processing on power failure**
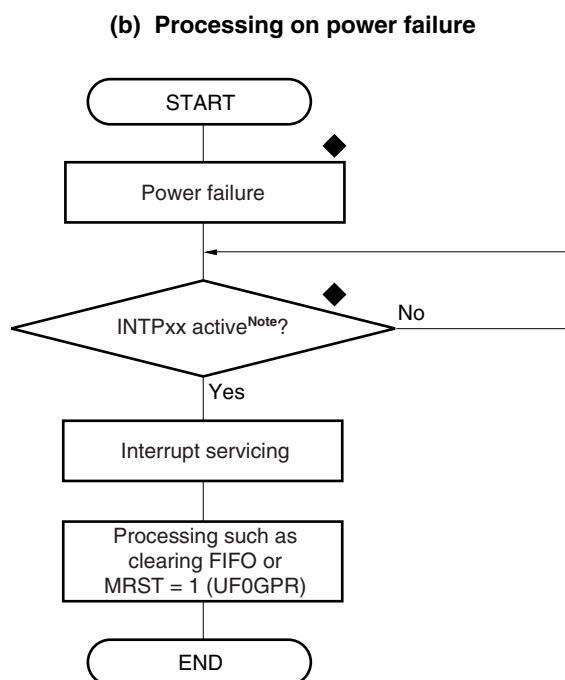


**Note** INTPxx indicates the external interrupt pins of the V850ES/JG3-U and V850ES/JH3-U (INTP00 to INTP18), and also indicates interrupts input by the external trigger pins (TIAA00, TAA01, TIAA10, TIAA11, TIAA20, TIAA21, TIAA30, TIAA31, TIAA50, TIAA51, TIAB00, TIAB10, TRGAB1, TIT00) of the timer.
Allocate one external interrupt pin to the following applications.

- Detecting disconnection of the connector in the case of self-powered mode (SFPW bit of UF0DSTL register = 1). In this case, monitor the VDD line of the USB connector, and input the result to the external interrupt pin at the edge. Note that the noise elimination time is that of the interrupt input pin, and that of each timer.
- Detecting turning off power from the HUB when the device is mounted on the same board as a HUB chip.

**Remark** ♦ : Processing by hardware

**20.9.6 Receiving data for bulk transfer (OUT) in DMA mode**

Bulk transfer (OUT) is allocated to Endpoint2 and Endpoint4. The flowchart shown below illustrates how Endpoint2 is controlled when DMA is used. Endpoint4 can also be controlled in the same sequence. To use this flowchart as the control flow of Endpoint4, therefore, read the bit names of Endpoint2 in the flowchart as those of Endpoint4. The control flowchart shown below illustrates how remaining data is read by the CPU.

If data for bulk transfer (OUT) has been correctly received by setting the DQBO1MS bit of the UF0IDR register to 1, the DMA request signal for Endpoint2, instead of an interrupt request (INTUSBF0), becomes active. This DMA request signal for Endpoint2 operates according to the setting of the MODEn bit of the UF0IDR register (n = 0, 1). If all the data stored in the UF0BO1 register has been read by DMA, the DMA request signal for Endpoint2 becomes inactive. In this status, if data for the next bulk transfer (OUT) has been correctly received, the DMA request signal for Endpoint2 becomes active again. If the data for bulk transfer (OUT) that has been received is equal to or less than the FIFO size, a Short interrupt request is issued and the INTUSBF0 (EP2_ENDINT) signal becomes active, as soon as reading the data by DMA is completed. To read data by DMA again, set the DQBO1MS bit to 1 again. If DMA is completed by the DMA end signal for Endpoint2, the DQBO1MS bit of the UF0IDR register is cleared to 0, and the DMA request signal for Endpoint2 becomes inactive. At the same time, the DMA_END interrupt request is issued. If data remains in the UF0BO1 register at this time, DMA can be started again by setting the DQBO1MS bit of the UF0IDR register again. However, the data for bulk transfer (OUT) is always equal to or less than the FIFO size. Consequently, a Short interrupt request is issued, the INTUSBF0 (EP2_ENDINT) signal becomes active, the DQBO1MS bit is cleared, and the DMA request signal for Endpoint2 becomes inactive, as soon as the data is read by DMA.

**Cautions 1. The DMA request signal for Endpoint n (n = 2, 4) becomes active in the demand mode (MODE1 and MODE0 bits of the UF0IDR register = 10), as long as there is data to be transferred.**

        **2. For a DMA transfer for which the data for a bulk transfer (OUT) is a Short packet (63 bytes or less), after the transfer finishes, clear the UF0IC0.SHORTC and UF0IS0.SHORT bits.**

        **If the SHORT bits are not cleared, the DMASTOP_EPnB signal is asserted and the next DMA transfer operation is not performed.**

RENESAS

**(1) Initial settings for a bulk transfer (OUT: EP2, EP4)**

**(a) Initial settings for DMAC**

- The DSAn registers (n = 0 to 3) are set to 00210000H (for EP2) or 00220000H (for EP4).
- The DADCn registers (n = 0 to 3) are set to 0080H.

  (8-bit transfer, transfer source address: fixed, transfer destination address: incremental)
- The DTFRn registers (n = 0 to 3) are set to 0000H.
- The UFDRQEN register is set up according to the DMA channel to be used.

  (For details, see 20.6.10 (1) USBF DMA request enable register (UFDRQEN).)

**(b) Initial settings for EPC**

- The UF0IDR register is set to 12H (for EP2) or 22H (for EP4) (demand mode).
- The UF0IM0.DMAEDM bit = 0
- The UF0IM3.BKO1NLM bit = 0 (for EP2)
- The UF0IM3.BKO1DTM bit = 0 (for EP2)
- The UF0IM3.BKO2NLM bit = 0 (for EP4)
- The UF0IM3.BKO2DTM bit = 0 (for EP4)

**Figure 20-32. DMA Processing by Bulk Transfer (OUT) (1/3)**
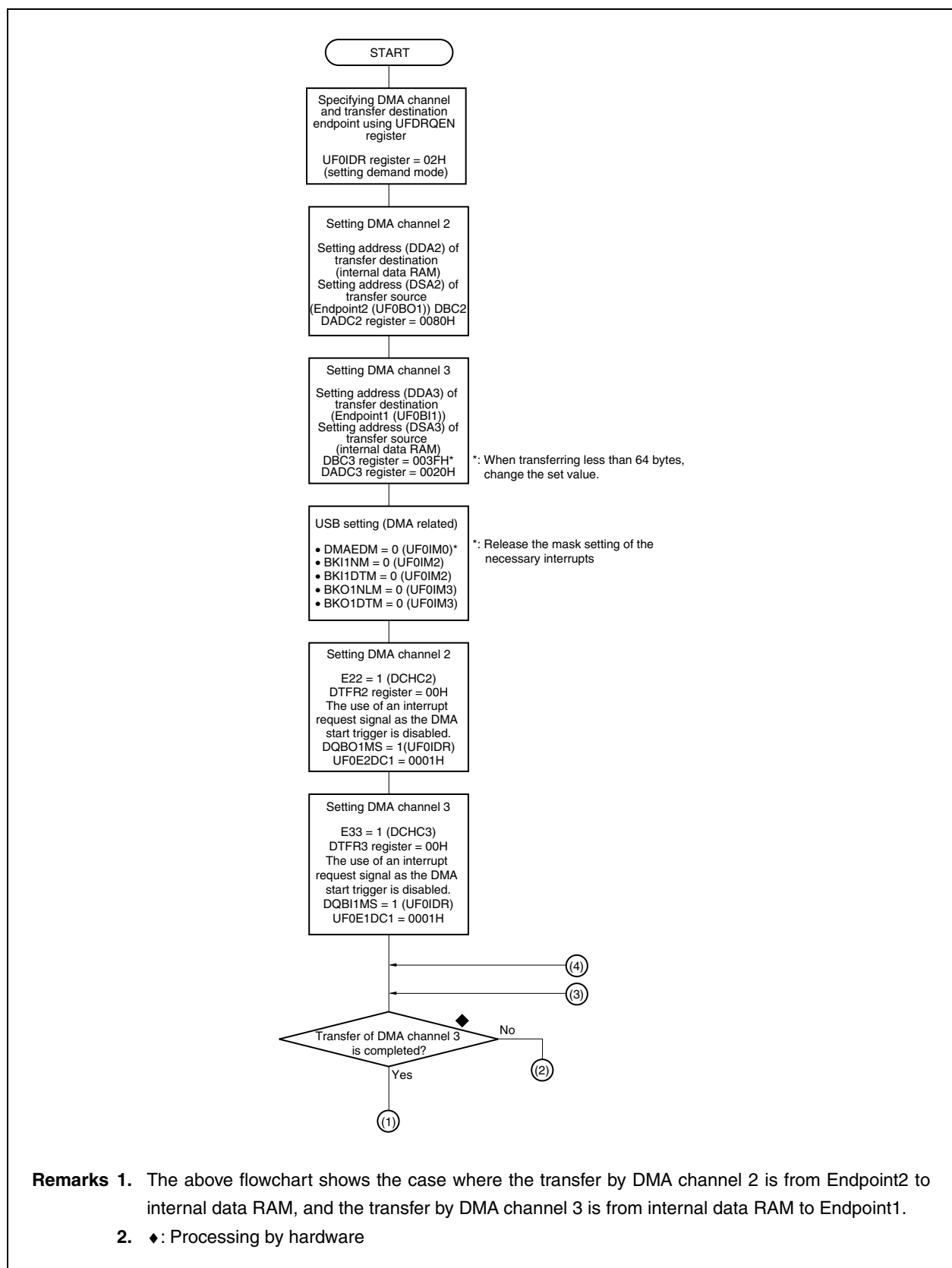


**Remarks 1.** The above flowchart shows the case where the transfer by DMA channel 2 is from Endpoint2 to internal data RAM, and the transfer by DMA channel 3 is from internal data RAM to Endpoint1.

**2.** ♦: Processing by hardware

**Figure 20-32. DMA Processing by Bulk Transfer (OUT) (2/3)**



Remarks 1. The above flowchart shows the case where the transfer by DMA channel 2 is from Endpoint2 to internal data RAM, and the transfer by DMA channel 3 is from internal data RAM to Endpoint1.

     2. n = 0, 1

     3. ◆: Processing by hardware
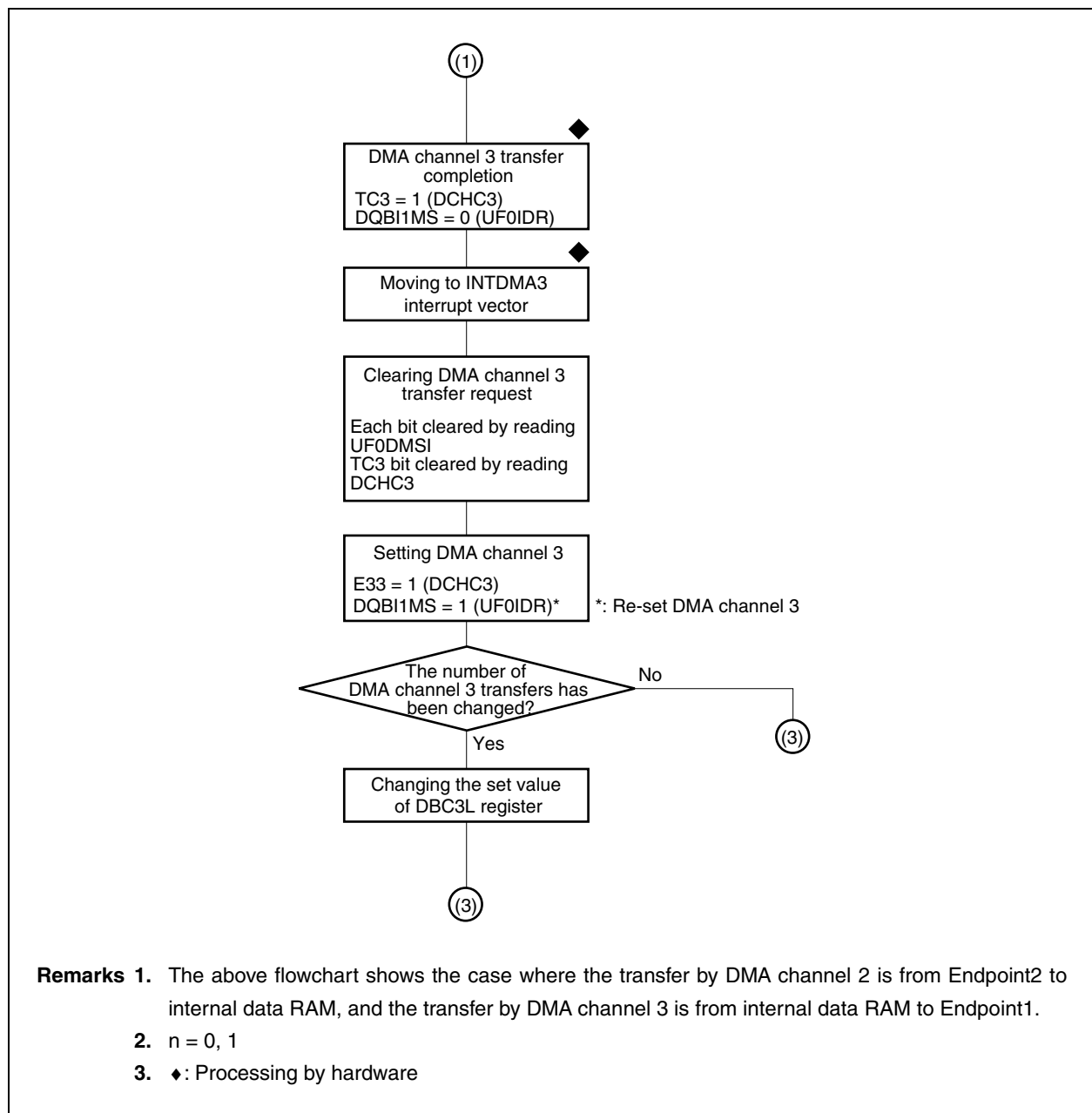
**Figure 20-32.  DMA Processing by Bulk Transfer (OUT) (3/3)**
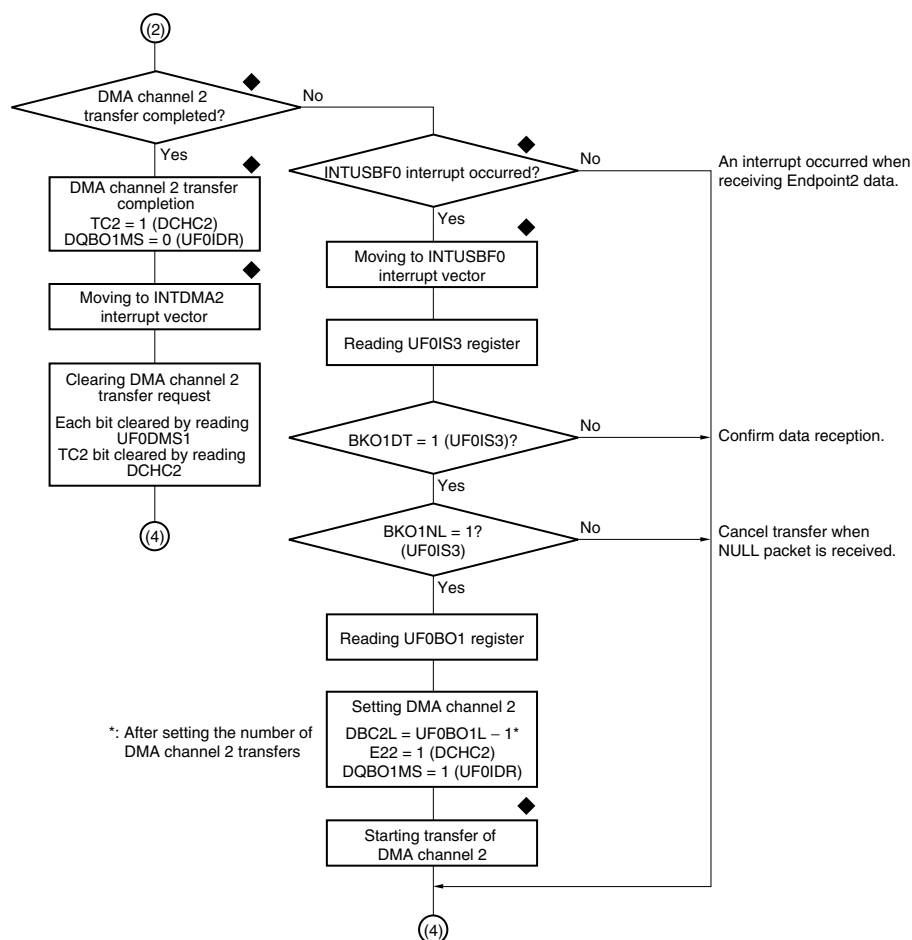


Remarks 1. The above flowchart shows the case where the transfer by DMA channel 2 is from Endpoint2 to internal data RAM, and the transfer by DMA channel 3 is from internal data RAM to Endpoint1.

2. n = 0, 1

   m = 2, 3

3. ♦: Processing by hardware

**20.9.7  Transmitting data for bulk transfer (IN) in DMA mode**

Bulk transfer (IN) is allocated to Endpoint1 and Endpoint3.  The flowchart shown below illustrates how Endpoint1 is controlled when DMA is used.  Endpoint3 can also be controlled in the same sequence.  To use this flowchart as the control flow of Endpoint3, therefore, read the bit names of Endpoint1 in the flowchart as those of Endpoint3.

If data for bulk transfer (IN) can be written by setting the DQBI1MS bit of the UF0IDR register to 1, the DMA request signal for Endpoint1, instead of an interrupt request (INTUSBF0), becomes active.  This DMA request signal for Endpoint1 operates according to the setting of the MODEn bit of the UF0IDR register (n = 0, 1).  If all the data that can be written to the UF0BI1 register has been written by DMA, the DMA request signal for Endpoint1 becomes inactive.  In this status, the toggle operation of the FIFO takes place and, if data for bulk transfer (IN) can be written, the DMA request signal for Endpoint1 becomes active again.  The automatic toggle operation of the FIFO is not executed even if the FIFO has become full as a result of DMA transfer, unless the BKI1T bit of the UF0DEND register is set to 1.  Therefore, be sure to set the BKI1DED bit of the UF0DEND register to 1 to transfer data.  If DMA is completed by the DMA end signal for Endpoint1, the DQBI1MS bit of the UF0IDR register is cleared to 0, and the DMA request signal for Endpoint1 becomes inactive.  At the same time, the DMA_END interrupt request is issued.  To transmit a short packet at this time when the FIFO is not full, set the BKI1DED bit of the UF0DEND register to 1.

**Caution   The DMA request signal for Endpoint n (n = 1, 3) becomes active in the demand mode (MODE1 and MODE0 bits of the UF0IDR register = 10), as long as data can be transferred.**

**(1) Initial settings for a bulk transfer (IN: EP1, EP3)**

   **(a) Initial settings for DMAC**
      - The DDAn registers (n = 0 to 3) are set to 00201000H (for EP1) or 00202000H (for EP3).
      - The DADCn registers (n = 0 to 3) are set to 0020H.
        (8-bit transfer, transfer source address: incremental, transfer destination address: fixed)
      - The DTFRn registers (n = 0 to 3) are set to 0000H.
      - The UFDRQEN register is set up according to the DMA channel to be used.
        (For details, see 20.6.10 (1) USBF DMA request enable register (UFDRQEN).)

   **(b) Initial settings for EPC**
      - The UF0IDR register is set to 42H (for EP1) or 82H (for EP3) (demand mode).
      - The UF0IM0.DMAEDM bit = 0
      - The UF0IM2.BKI1NLM bit = 0 (for EP1)
      - The UF0IM2.BKI1DTM bit = 0 (for EP1)
      - The UF0IM2.BKI2NLM bit = 0 (for EP3)
      - The UF0IM2.BKI2DTM bit = 0 (for EP3)

**Figure 20-33.  DMA Processing by Bulk Transfer (IN) (1/4)**
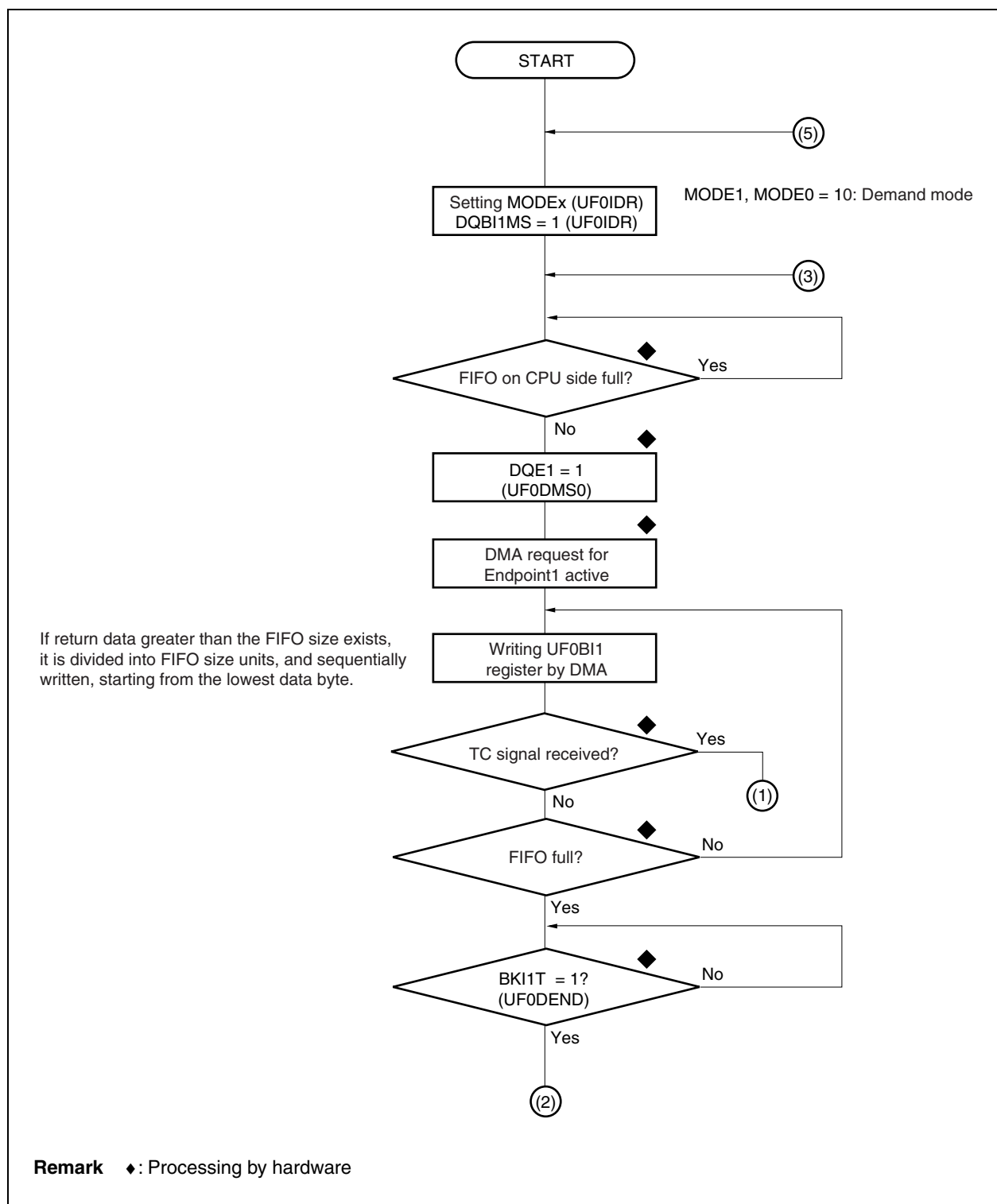


**Remark**   ♦ : Processing by hardware

**Figure 20-33. DMA Processing by Bulk Transfer (IN) (2/4)**



**Note** The timing of the bit value changes depending on the status on the SIE side.

**Remark** ♦ : Processing by hardware
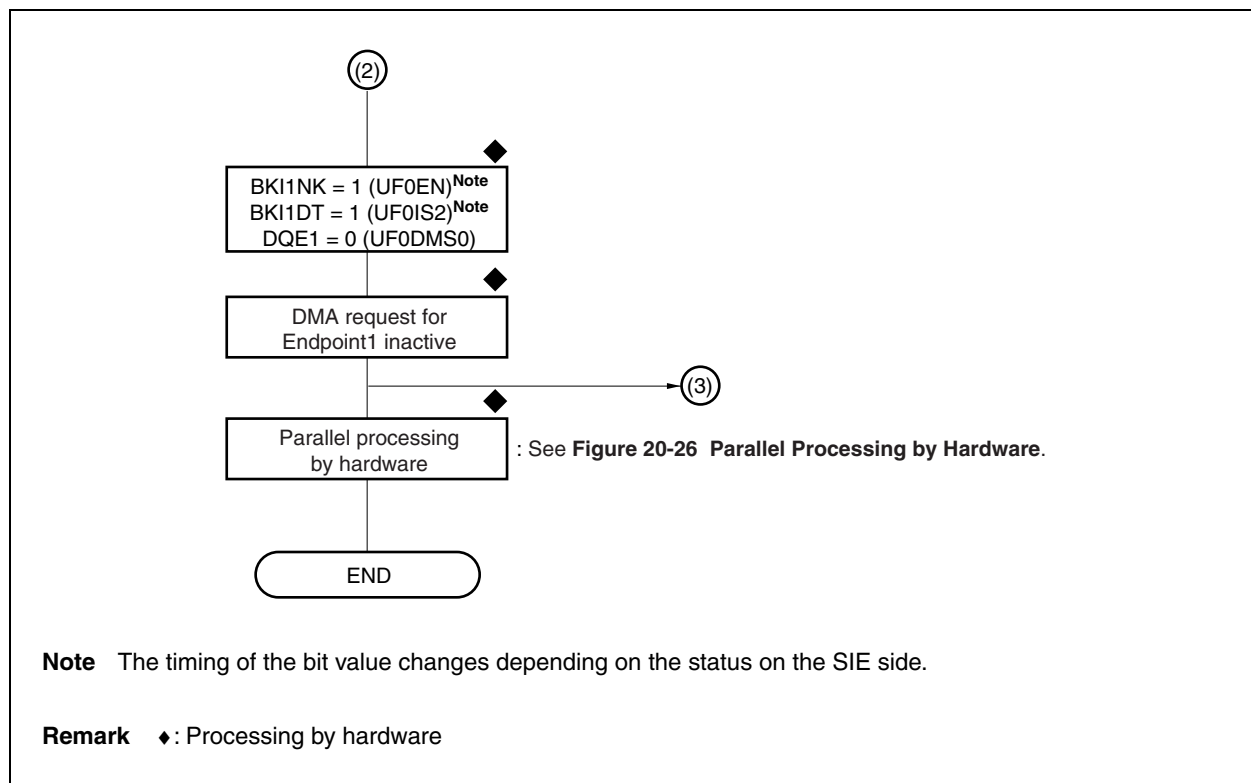
**Figure 20-33.  DMA Processing by Bulk Transfer (IN) (3/4)**
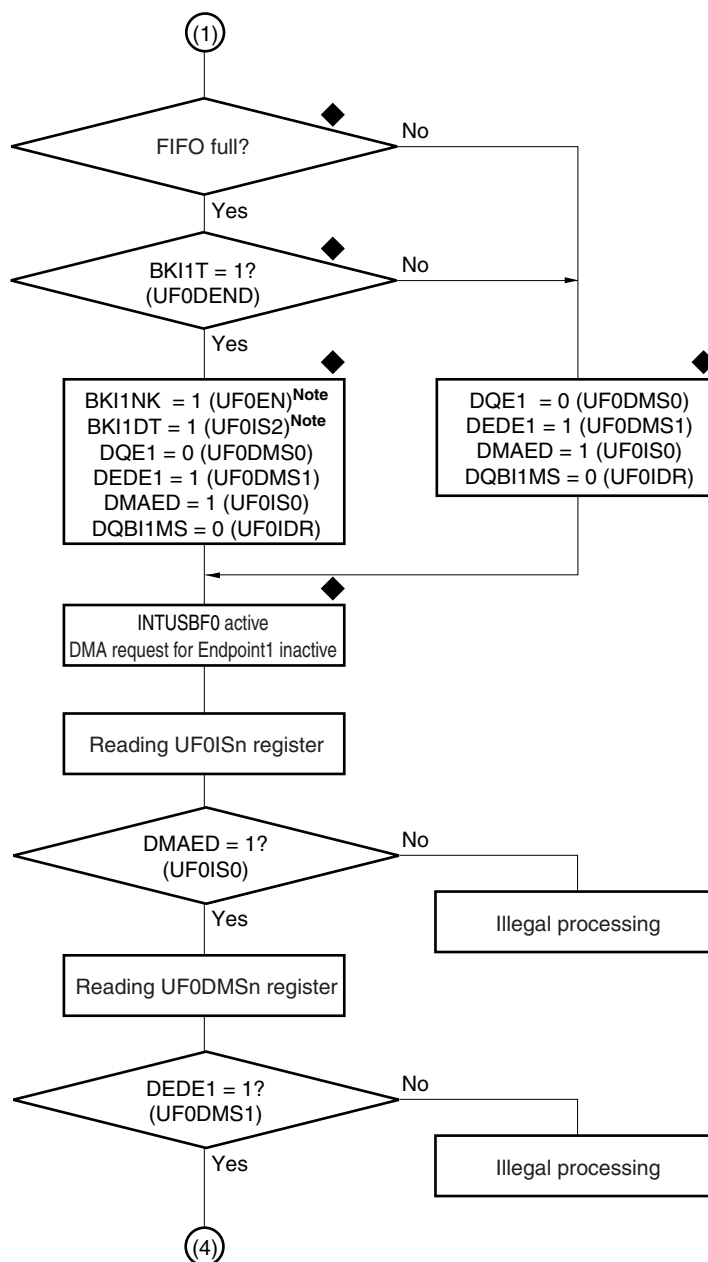


**Note** The timing of the bit value changes depending on the status on the SIE side.

**Remarks 1.** n = 0, 1

**2.** ♦: Processing by hardware

**Figure 20-33 DMA Processing by Bulk Transfer (IN) (4/4)**



Note The timing of the bit value changes depending on the status on the SIE side.

Remark ♦ : Processing by hardware
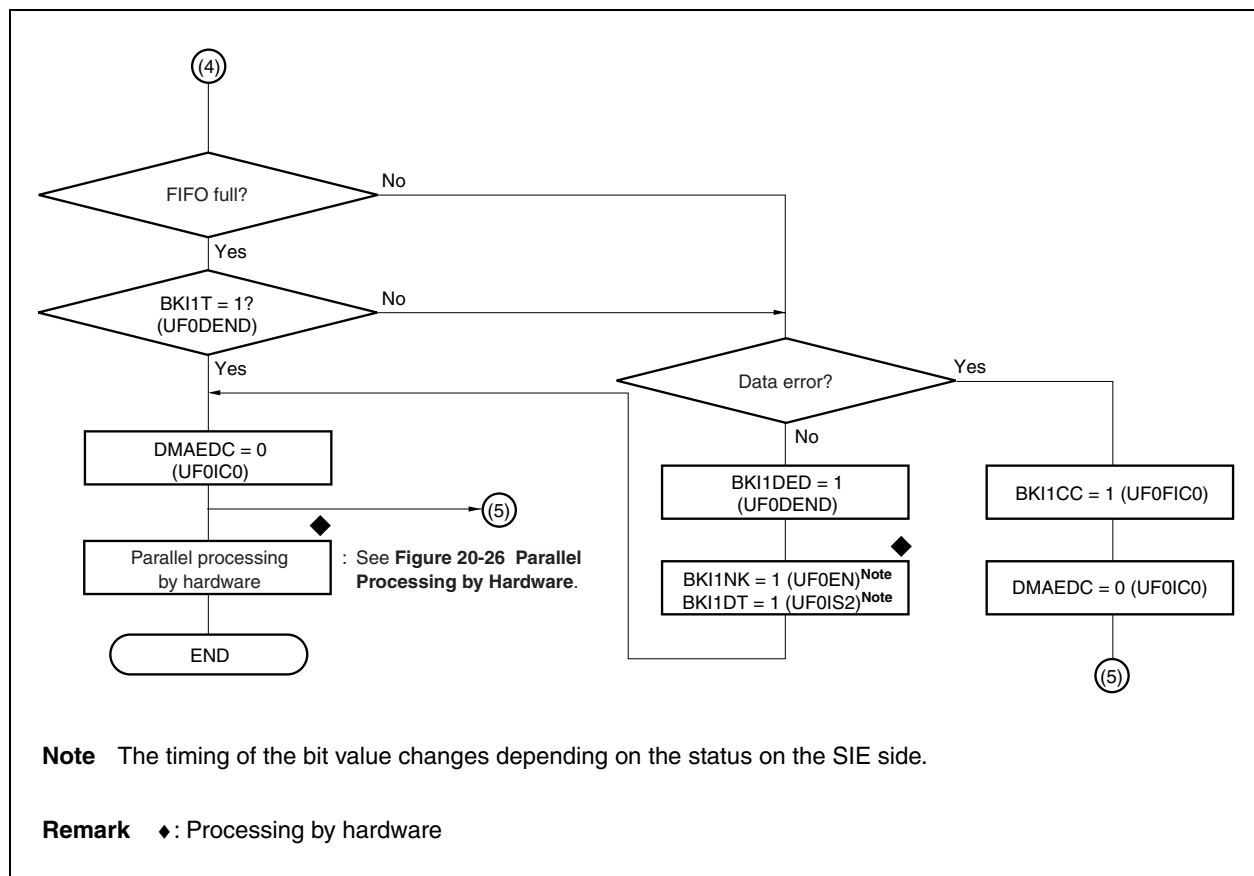
## CHAPTER 21  USB HOST CONTROLLER (USBH)

The V850ES/JG3-U and V850ES/JH3-U have an internal USB host controller (USBH) conforming to the Universal Serial Bus Specification.  Data communication using the polling method is performed between the USB host controller and external host device by using a token-based protocol.

### 21.1  Overview

- Conforms to the Universal Serial Bus Specification Revision
- Supports 12 Mbps (full-speed) transfer
- Supports Open Host Controller Interface (OHCI) (with restriction on control when transitioning the USB port state from Disable to Enable)
- Supports control transfer, bulk transfer, interrupt transfer, and isochronous transfer[Note]
- Has a built-in root hub function and is equipped with 1 downstream port channel
  Uses SRAM (8 KB) for shared memory
- Clock: Internal CLK (6 MHz $\times$ 8 = 48 MHz) or clock input from UCLK pin ($f_{USB}$ = 48 MHz)

**Note**  Isochronous transfer puts a heavy load on the system performance, so be sure to sufficiently evaluate performance with the target system in advance.

## 21.2 Configuration

### 21.2.1 Block diagram

**Figure 21-1. Block Diagram of USB Host Controller**



Remark     Inside the broken lines square: Functions included in the USB host controller
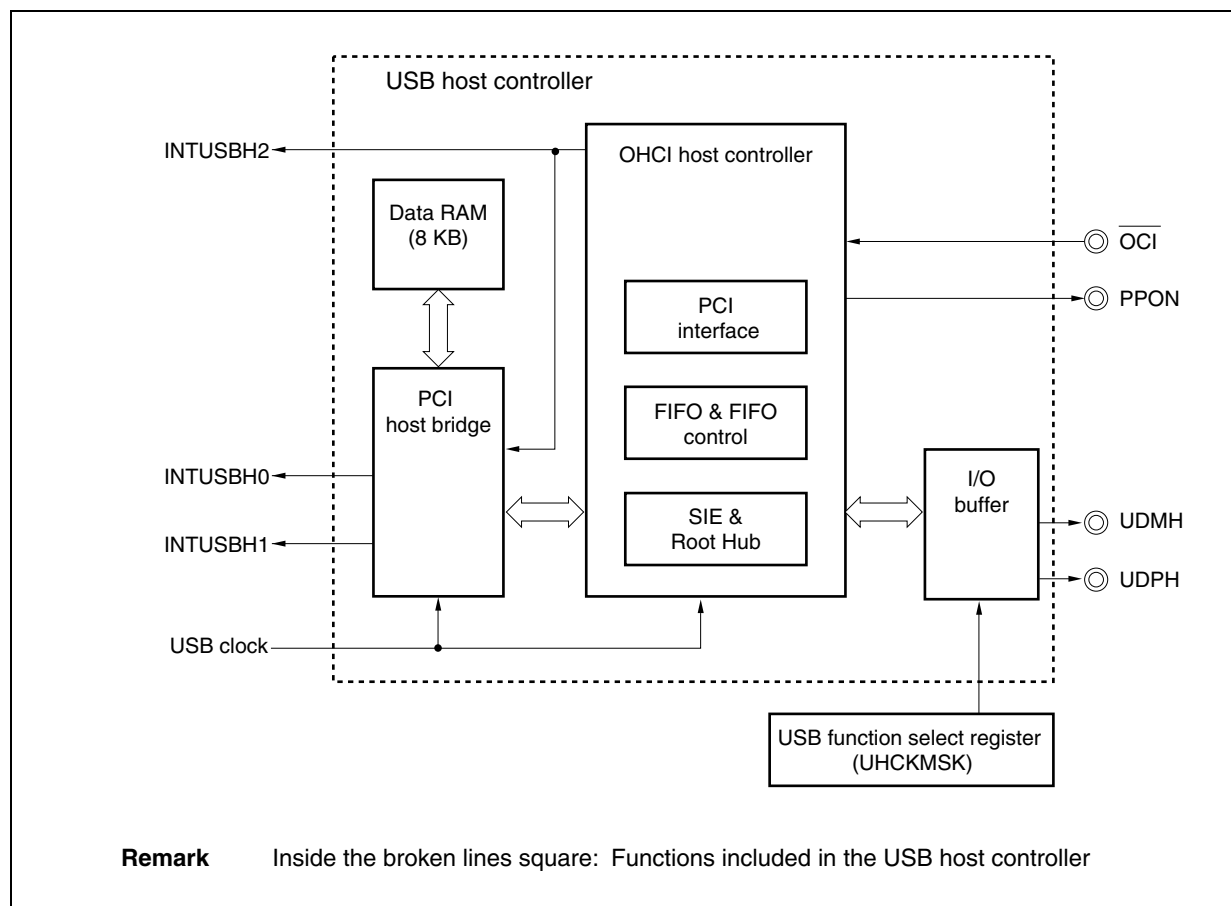
**21.2.2 USB host controller memory map**

The USB host controller is allocated in the $\overline{\text{CS1}}$ space internally in the microcontroller (external pin $\overline{\text{CS1}}$ does not exist). The OHCI host controller, which is the main function of the USB host controller, is connected via the SRAM interface of the $\overline{\text{CS1}}$ space and the internal PCI bus bridge. The CPU and internal PCI bus bridge share the same memory, and the CPU memory space and PCI memory space are linked. This relationship is defined by the PCI host bridge registers and the OHCI host configuration registers that are accessed via the PCI host bridge registers.

**Table 21-1. Division of CPU Memory Space**

| Address | Access | Area |
|---|---|---|
| 00280000H to 00281FFFH | 8 or 16 bits | SRAM area |
| 002E0000H to 002E0FFFH | 32 bits | OHCI area |
| 002E1000H to 002E10FFH | | PCI bus bridge register area |

**Figure 21-2. USB Host Controller Memory Map**
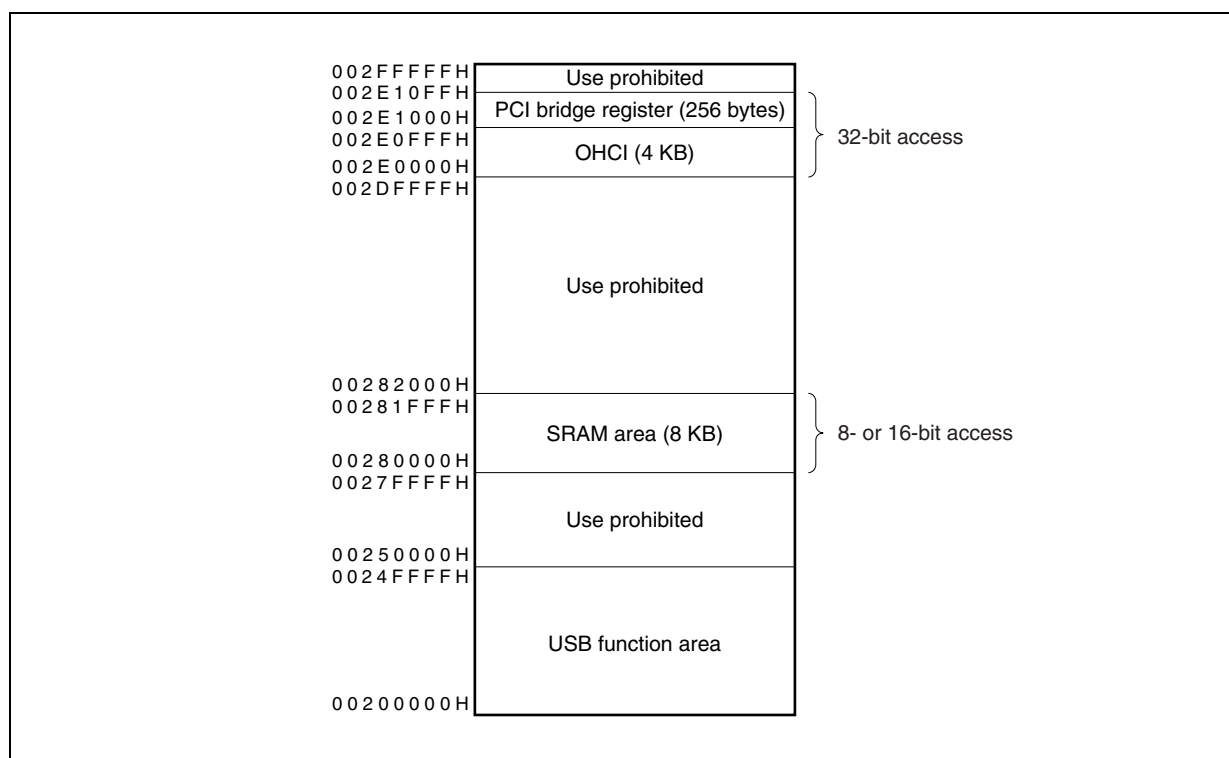
### 21.2.3  Cautions on data access

The V850ES/JG3-U and V850ES/JH3-U are connected to the USB host controller via the PCI bus bridge via a 16-bit bus.  The OHCI area and the PCI bus bridge register area are accessed in 32-bit units, and the USBH converts 16-bit cycles received from the CPU into 32-bit cycles to start the cycle for the target.

Be sure to access in 32-bit units in accordance with Table 21-2.  A data access causing an error results in a PCI cycle error (INTUSBH1).  Accessing misaligned addresses is prohibited since the address is regarded as being discontinuous.  Only accessing 32-bit-aligned continuous addresses is considered to be an access to a continuous address, and operates normally (see **Figure 21-3**).

**Table 21-2.  OHCI and PCI Bus Bridge Register Area (32-bit Access)**

|  | Access | Address | | Command | |
|---|---|---|---|---|---|
|  |  | 1st Cycle | 2nd Cycle | 1st Cycle | 2nd Cycle |
| <1> | Normal | Continuous | | Write | Write |
| <2> | Normal | Continuous | | Read | Read |
| <3> | Error | Discontinuous | | Write | Write |
| <4> | Error | Discontinuous | | Read | Read |
| <5> | Error | Continuous or discontinuous | | Write | Read |
| <6> | Error | Continuous or discontinuous | | Read | Write |
| <7> | Error | 32-bit access | None | Write (8 bits) | |
| <8> | Error | Continuous or discontinuous | | Write or read | Write (8 bits) |
| <9> | Error | Access prohibited area | | – | – |

**Figure 21-3.  Example of Accessing Misaligned Addresses**
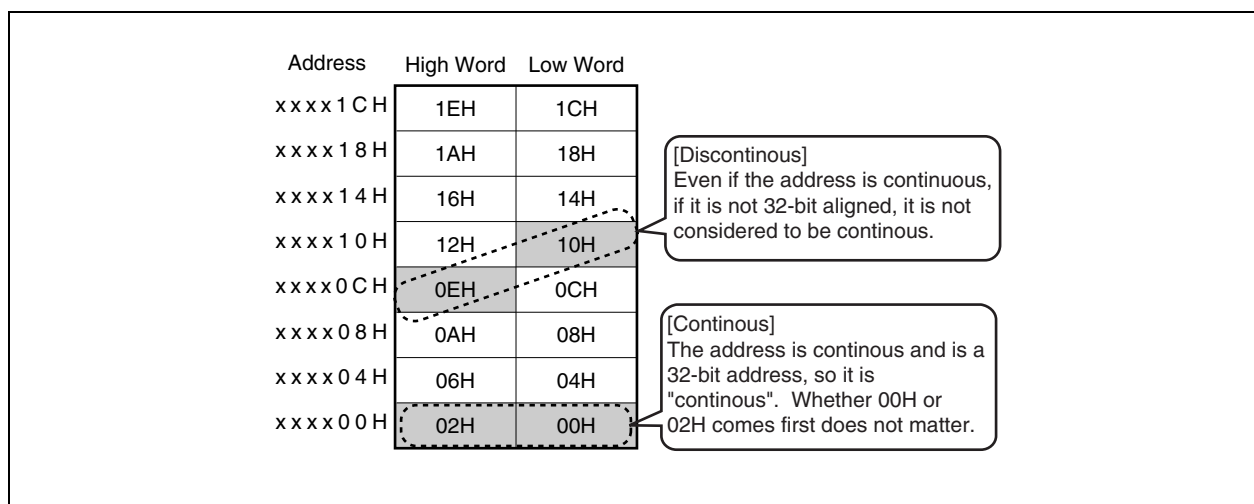
## 21.3  External Circuit Configuration

### 21.3.1  Overview

In USB transmission, when communication is performed with the host controller and function controller facing each other, pull-up/pull-down resistors must be connected to the USB signal (D+/D−) to identify the communication partner. Moreover, in the V850ES/JG3-U and V850ES/JH3-U, series resistors must also be connected.

Because the V850ES/JG3-U and V850ES/JH3-U do not include these pull-up/pull-down resistors and series resistors, be sure to connect them externally.

Figure 21-4 shows the schematic configuration of the USB transmission line.  For details of the external configuration, see **21.3.2  Connection configuration**.

**Figure 21-4.  Schemativ Configuration of Pull-up, Pull-down, Series Resistors in USB Transmission Line**
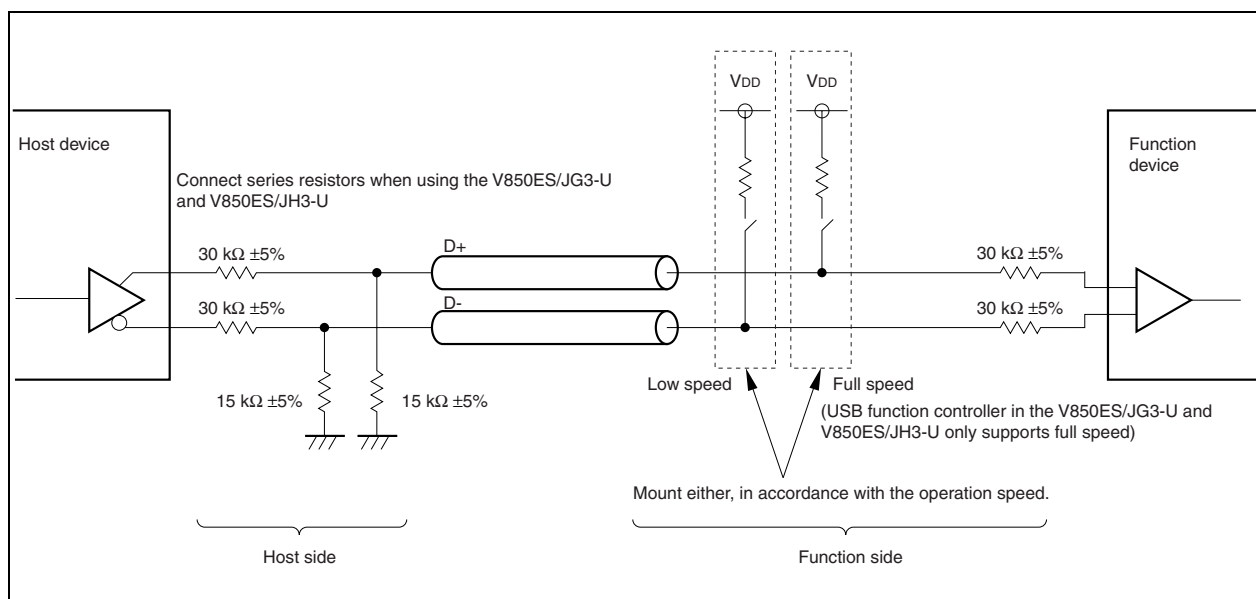
### 21.3.2  Connection configuration

**Figure 21-5.  Example of USB Host Controller Connection**
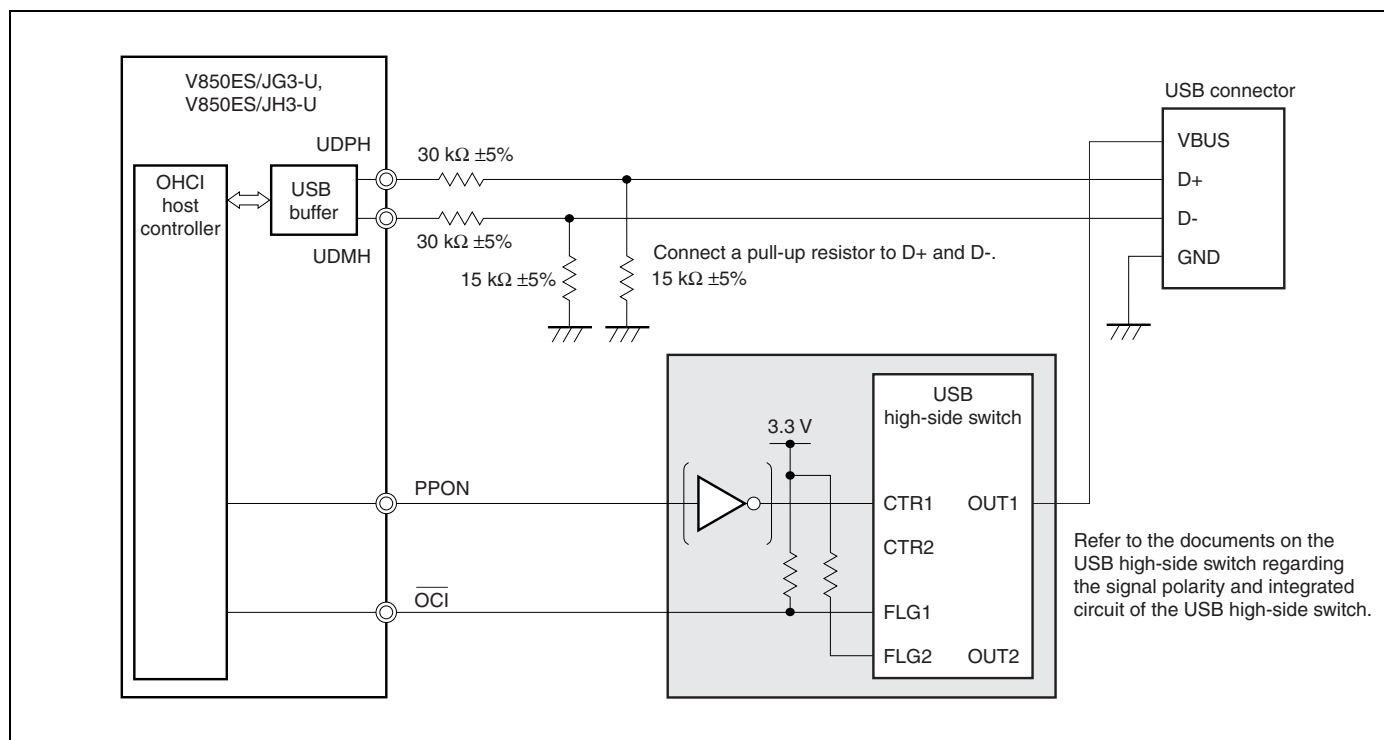


**(1)  Series resistor connection to D+/D–**

Connect series resistors of 30 Ω ±5% to the D+/D– pins (UDPH, UDMH) of the USB host controller in the V850ES/JG3-U and V850ES/JH3-U.  If they are not connected, the impedance rating cannot be satisfied and the output waveform may be disturbed.

Allocate the series resistors adjacent to the V850ES/JG3-U or V850ES/JH3-U, and make the length of the wiring between the series resistors and the USB connectors the same, to make the impedance of D+ and D– equal (a differential with 90 Ω ±5% is recommended).

**(2)  Pull-down control of D+/D–**

Be sure to pull down the D+/D– pins (UDPH, UDMH) by 15 kΩ ±5% to GND.

In this case, the configuration is the same as that when the function device is disconnected.

### 21.3.3  USB power supply

**(1)  Overcurrent detection and power control**

The V850ES/JG3-U and V850ES/JH3-U do not have a circuit for detecting overcurrent or controlling power to the USB port.

To support these functions in the system, use an external circuit such as a USB high-side switch to configure these functions and connect it with the OCI and $\overline{\text{PPON}}$ pins.

The following table shows the operations related to the $\overline{\text{OCI}}$ and PPON pins used for controlling the external circuit for detecting overcurrent or controlling power to the USB port.

**Table 21-3.  OCI and PPON Signals**

| Pin | I/O | Level | Function |
|------|--------|-------|-------------------------------|
| $\overline{\text{OCI}}$ | Input | 1 | Overcurrent not detected |
|  |  | 0 | Overcurrent detected |
| PPON | Output | 1 | Power supply to VBUS is on |
|  |  | 0 | Power supply to VBUS is off |

Figure 21-5 shows a connection example for overcurrent detection and power (VBUS) control.  When the USB bus is not used, power consumption can be reduced by stopping VBUS supply to the port, though its availability depends on the connection to the USB connector.  If the connected USB function device is buspowered, it is recommended to use a high-side switch as shown in the connection example.

**(2)  VBUS control specification according to root hub register setting conditions**

Control of the PPS bit for controlling VBUS varies depending on the setting of the HcRhDescriptorA and HcRhDescriptorB (Root Hub control) registers.

When controlling VBUS for each port, the HcRhDescriptorA.NPS and HcRhDescriptorA.PSM bits must be set to "1".

## 21.4  Cautions

**(1)  Clock accuracy**

To operate the USB host controller, the internal clock (6 MHz external clock $\times$ 8 internal clocks = 48 MHz internal clock) or external clock (external clock input to UCLK pin ($f_{USB}$ = 48 MHz)) must be used as the USB clock.  When the internal clock is used as the USB clock, use a resonator with an accuracy of 6 MHz $\pm$500 ppm (max.).  When the external clock is used, supply a clock with an accuracy of 48 MHz $\pm$500 ppm (max.) to the UCLK pin.  If the USB clock accuracy drops, the transmission data cannot satisfy the USB rating.

**(2)  HUB connection**

Use of the external clock (crystal resonator) is recommended for the hub connection.  When the internal clock is used, the USB rating may not be satisfied because of the effects of clock jitter in the internal circuit of the microcontroller.  When the internal clock is used, connecting the host device on a 1-to-1 basis is recommended.

## 21.5  Control Registers

### 21.5.1  USB control registers

**(1)  USB clock select register (UCKSEL)**

The UCKSEL register selects the operation clock of the USB controllers.

The UCKSEL register is also used by the USB function controller.  For details, see **20.6.1 (1) USB clock selection register (UCKSEL)**.

**(2)  USB function select register (UHCKMSK)**

The UHCKMSK register selects the functions of the USB controllers.

The UHCKMSK register is also used by the USB function controller.  For details, see **20.6.1 (3) USB function selection register (UHCKMSK)**.

## 21.6  PCI Host Bridge

### 21.6.1  PCI host bridge

The PCI host bridge is a bridge circuit that connects to the CPU system to the OHCI host controller, and is equipped with the following functions.

- PCI master cycle control

  Issues the following PCI cycles in response to requests for bus access from the CPU (MEMC).
    - PCI Configuration Register Read/Write Single Cycle
    - PCI Memory Read/Write Single Cycle

- PCI slave cycle control

  Acknowledges the PCI Memory Read/Write Cycle (up to 8 Dwords of burst transfer) in response to an access to the SDRAM or SRAM area from the PCI bus.

- PCI error handling

  Generates an error interrupt (INTUSBH0) upon Master Abort, Target Abort, $\overline{\text{PERR}}$ reception, and $\overline{\text{SERR}}$ reception. (Holds the address immediately before the address at which an error occurs.)

- PCI address conversion control

  The PCI window base address register, which is used to convert the physical addresses transferred from the CPU and output them to the PCI bus, is available.

- Control of memory controller bus incorporated in V850ES/JG3-U and V850ES/JH3-U

  Controls the bus cycles by using hardware waits (WAIT) for the accesses from the CPU via the memory controller bus.

- SRAM control

  An 8 KB SRAM is incorporated as a shared memory.  It is mainly used for allocating descriptors.  Controls and arbitrates accesses to the SRAM areas from the CPU (MEMC) and the PCI bus.

### 21.6.2 PCI host bridge registers

The PCI bridge has the following PCI host bridge registers.

These registers can be accessed in 32-bit units only.

**Table 21-4.  PCI Host Bridge Registers**

| Address | Register Name | Symbol | R/W | Manipulatable Bit Unit | | | | Initial Value |
|---|---|---|---|---|---|---|---|---|
| | | | | 1 | 8 | 16 | 32 | |
| 002E1000H | PCI configuration data register | PCI_CONFIG_DATA | R/W | | | | √ | Undefined |
| 002E1004H | PCI configuration address register | PCI_CONFIG_ADD | R/W | | | | √ | 00000000H |
| 002E1008H | PCI control register 1 | PCI_CONTROL1 | R/W | | | | √ | 07000300H |
| 002E100CH | PCI control register 2 | PCI_CONTROL2 | R/W | | | | √ | 00000077H |
| 002E1014H | PCI window base address register | PCI_WIN_BASE | R/W | | | | √ | 002E0000H |
| 002E1018H | PCI interrupt status register | PCI_INT_STATUS | R/W | | | | √ | 00000000H |
| 002E101CH | PCI interrupt control register | PCI_INT_CTL | R/W | | | | √ | 00000000H |
| 002E1020H | PCI bus error address register | PCI_ERR_ADD | R/W | | | | √ | 00000000H |
| 002E1024H | Bus bridge control register | BGR_CTL | R/W | | | | √ | 00000016H |
| 002E1050H | SRAM_BASE register | SRAM_BASE | R/W | | | | √ | 00280000H |

### (1)  PCI configuration data register (PCI_CONFIG_DATA)

After reset: Undefined　　R/W　　Address: 002E1000H

| | | |
|---|---|---|
| | 31 | 0 |
| PCI_CONFIG_DATA | cdata[31:0] | |

| Bit position | Bit name | Function |
|---|---|---|
| 31-0 | cdata[31:0] | Can be used to access OHCI host configuration registers. The PCI_CONFIG_ADD register must be set in advance. |

**(2) PCI configuration address register (PCI_CONFIG_ADD)**

After reset: 0000 0000H  R/W  Address: 002E1004H

| | 31 | 0 |
|---|---|---|
| PCI_CONFIG_ADD | cadd[31:0] | |

| Bit position | Bit name | Function |
|---|---|---|
| 31-0 | cadd[31:0] | Sets the address of the OHCI host configuration register subject to access.<br>[How to set OHCI configuration address register]<br><br>31　　　　　　　　　　1110　8 7　　2 1 0<br><br>IDSEL specification　　Function no.　Register no.<br><br>• Register no.:　Specifies the OHCI host configuration register number.<br>• Function no.:　Specifies the function number for the PCI multifunction device.<br>• IDSEL specification: Selects the IDSEL signal of the PCI device to be accessed.<br><br>Since the PCI host bridge substitutes AD[31:11] for the IDSEL signal, set any one bit from among bits corresponding to AD signals connected to IDSEL pins to perform a configuration access to PCI devices connected to the PCI bus.<br>In the V850ES/JG3-U and V850ES/JH3-U, the AD31 signal is connected to the IDSEL pin. Therefore, the PCI device (OHCI host controller) can be accessed by setting (1) the cadd31 bit.<br><br>[Method for accessing OHCI host configuration registers]<br>Configuration registers on the PCI bus can be accessed by using the following procedure.<br><br>• PCI_CONFIG_ADD register setting<br>　Set the address subject to the access during the PCI configuration cycle.<br>• PCI_CONFIG_DATA register setting<br>　The value set to the PCI_CONFIG_ADD register can be read from the PCI_CONFIG_DATA register.<br>　When a value is written to the PCI_CONFIG_DATA register, a write access to the corresponding OHCI host configuration register can then be performed. |

### (3) PCI control register 1 (PCI_CONTROL1)

After reset: 0700 0300H   R/W   Address: 002E1008H

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| PCI_CONTROL1 | pci_parkcnt7 | pci_parkcnt6 | pci_parkcnt5 | pci_parkcnt4 | pci_parkcnt3 | pci_parkcnt2 | pci_parkcnt1 | pci_parkcnt0 |

| | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | pci_bpmode |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | pci_req_en1 | pci_req_en0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | pci_pchken | pci_reset | sram_en | sdram_en | mem_en | 0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31-24 | pci_parkcnt [7:0] | PCI Bus Parking Timer<br>Sets the time required for shifting to Bus Parking mode.  Counting starts by setting $\overline{\text{FRAME}}$ and $\overline{\text{IRDY}}$ to "1".<br>Use this field with the initial setting. |
| 16 | pci_bpmode | PCI Bus Parking Mode<br>Only the V850ES/JG3-U and V850ES/JH3-U can serve as the bus parking master (initial value).<br>The bus parking master is the master that accessed the PCI bus last.<br>Use this field with the initial setting. |
| 9, 8 | pci_req_en [1:0] | PCI Request Enable<br>The pci_req_en0 bit is fixed to "1".<br>Disable PCI requests (initial value).<br>Enables PCI requests.<br>Set this bit (1) when issuing a PCI request. |
| 5 | pci_pchken | PCI Parity Check Enable<br>Disables parity check for PCI bus (initial value).<br>Enables parity check for PCI bus.<br>Use this field with the initial setting. |
| 4 | pci_reset | PCI Reset<br>The PCI bus is in the reset state (initial value).<br>The PCI bus is in the reset release state.<br>Set this bit (1) when accessing the OHCI host controller. |
| 3 | sram_en | SRAM Area Enable<br>Does not respond to access from PCI bus to SRAM area (initial value).<br>Responds to access from PCI bus to SRAM area.<br>Set this bit (1) when accessing SRAM from the OHCI host controller. |
| 2 | sdram_en | SDRAM Area Enable<br>Does not respond to access from PCI bus to SDRAM area (initial value).<br>Responds to access from PCI bus to SDRAM area.<br>Set this bit (1) when accessing SDRAM from the OHCI host controller. |
| 1 | mem_en | PCI Memory Area Enable<br>Disables access from CPU to PCI memory area (initial value).<br>Enables access from CPU to PCI memory area.<br>Set this bit (1) when accessing an OHCI host configuration register of the OHCI host controller. |

**(4) PCI control register 2 (PCI_CONTROL2)**

After reset: 0000 0077H   R/W   Address: 002E100CH

| PCI_CONTROL2 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | wbuf_busy |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | wburst_size2 | wburst_size1 | wburst_size0 | 0 | rburst_size2 | rburst_size1 | rburst_size0 |

| Bit position | Bit name | Function |
|---|---|---|
| 8 | wbuf_busy | PCI Write Buffer Busy<br>Indicates the status of data in the write buffer when writing to a PCI target (SDRAM).<br>　0: Data to be written to SDRAM/SRAM does not remain in the write buffer.<br>　1: Data to be written to SDRAM/SRAM remains in the write buffer.<br>This bit is read-only. |
| 6-4 | wburst_size [2:0] | PCI Write Burst Max Size<br>Sets the maximum burst length for writing to a PCI target (SDRAM).<br><br>Use this field with the initial setting. |
| 2-0 | rburst_size [2:0] | PCI Read Burst Max Size<br>Sets the maximum burst length for reading from a PCI target (SDRAM).<br><br>Use this field with the initial setting. |

wburst_size:

| 2 | 1 | 0 | Maximum Burst Length for Writing to PCI Target |
|---|---|---|---|
| 0 | 0 | 0 | Supports single transfer only. |
| 0 | 0 | 1 | 2 bursts |
| 0 | 1 | 0 | 3 bursts |
| 0 | 1 | 1 | 4 bursts |
| 1 | 0 | 0 | 5 bursts |
| 1 | 0 | 1 | 6 bursts |
| 1 | 1 | 0 | 7 bursts |
| 1 | 1 | 1 | 8 bursts (initial value) |

rburst_size:

| 2 | 1 | 0 | Maximum Burst Length for Reading from PCI Target |
|---|---|---|---|
| 0 | 0 | 0 | Supports single transfer only. |
| 0 | 0 | 1 | 2 bursts |
| 0 | 1 | 0 | 3 bursts |
| 0 | 1 | 1 | 4 bursts |
| 1 | 0 | 0 | 5 bursts |
| 1 | 0 | 1 | 6 bursts |
| 1 | 1 | 0 | 7 bursts |
| 1 | 1 | 1 | 8 bursts (initial value) |

**(5)  PCI window base address register (PCI_WIN_BASE)**

After reset: 002E 0000H    R/W    Address: 002E1014H

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| PCI_WIN_BASE | p_base23 | p_base22 | p_base21 | p_base20 | p_base19 | p_base18 | p_base17 | p_base16 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| p_base23 | p_base22 | p_base21 | p_base20 | p_base19 | p_base18 | p_base17 | p_base16 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| p_base15 | p_base14 | p_base13 | p_base12 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31-12 | p_base [31:12] | PCI Window Base Address Register<br>Sets the highest address (PCI window base address) of a PCI space to be accessed via a PCI window.<br>When accessing a PCI bus memory space via a PCI window, accessing any area in 4 KB PCI bus memory spaces is enabled according to the setting of this register. |

**(6) PCI interrupt status register (PCI_INT_STATUS)**

This register can be used to identify the source of the INTUSBH0 and INTUSBH1 signals that are output from the PCI host bridge.

The sources of interrupts inta, int_smmi and int_pme are cleared by the unit which issued the interrupts.

The sources of interrupts serr, perr, mabort and tabort are used during debugging only, and are not used in other operations.

See **6.7.5  PCI_INT_CTL register** for how to enable interrupt sources.

(1/2)

After reset: 0000 0000H    R/W    Address: 002E1018H

PCI_INT_STATUS

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | Int_pme | Int_smmi | 0 | Inta |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | serr | perr | mabort | tabort |

| Bit position | Bit name | Function |
|--------------|----------|----------|
| 11 | int_pme | PCI Interrupt PME Status<br>0:  No interrupt sources<br>1:  An interrupt by PME occurred.<br>The source of this interrupt is cleared (0) by the source which issued PME.<br>This bit is read-only. |
| 10 | Int_smmi | PCI Interrupt SMMI Status<br>0:  No interrupt sources<br>1:  An interrupt by SMMI occurred.<br>The source of this interrupt is cleared (0) by the source which issued SMMI.<br>This bit is read-only. |
| 8 | inta | PCI Interrupt INTA Status<br>0:  No interrupt sources<br>1:  An interrupt by INTA occurred.<br>The source of this interrupt is cleared (0) by the source which issued INTA.<br>This bit is read-only. |
| 3 | serr | PCI Host Bridge System Error Interrupt Status<br>0:  No interrupt sources<br>1:  A system error was detected.<br>The source of this interrupt is cleared (0) by writing "1" to this bit.<br>The System Error interrupt is issued during debugging and is not used during normal operation. |

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 2 | perr | PCI Host Bridge Parity Error Interrupt Status<br><br>  0:  No interrupt sources<br><br>  1:  A parity error was detected.<br><br>The source of this interrupt is cleared (0) by writing "1" to this bit.<br>The Parity Error interrupt is issued during debugging and is not used during normal operation. |
| 1 | mabort | PCI Host Bridge Master Abort Interrupt Status<br><br>  0:  No interrupt sources<br><br>  1:  Master abort was received.<br><br>The source of this interrupt is cleared (0) by writing "1" to this bit.<br>The Master Abort interrupt is issued during debugging and is not used during normal operation. |
| 0 | tabort | PCI Host Bridge Target Abort Interrupt Status<br><br>  0:  No interrupt sources<br><br>  1:  Target abort was received.<br><br>The source of this interrupt is cleared (0) by writing "1" to this bit.<br>The Target Abort interrupt is issued during debugging and is not used during normal operation. |

**(7) PCI interrupt control register (PCI_INT_CTL)**

After reset: 0000 0000H   R/W   Address: 002E101CH

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| PCI_INT_CTL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | int_pme_en | int_smmi_en | 0 | inta_en |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | serrint_en | perrint_en | mabortint_en | tabortint_en |

| Bit position | Bit name | Function |
|---|---|---|
| 11 | int_pme_en | PCI Interrupt PME Enable<br>0: Disables as interrupt source (initial value)<br>1: Enables as interrupt source |
| 10 | int_smmi_en | PCI Interrupt SMMI Enable<br>0: Disables as interrupt source (initial value)<br>1: Enables as interrupt source |
| 8 | inta_en | PCI Interrupt INTA Enable<br>0: Disables as interrupt source (initial value)<br>1: Enables as interrupt source |
| 3 | serrint_en | PCI Host Bridge System Error Interrupt Enable<br>0: Disables as interrupt source (initial value)<br>1: Enables as interrupt source<br>This interrupt is issued during debugging and is not used during normal operation. |
| 2 | perrint_en | PCI Host Bridge Parity Error Interrupt Enable<br>0: Disables as interrupt source (initial value)<br>1: Enables as interrupt source<br>This interrupt is issued during debugging and is not used during normal operation. |
| 1 | mabortint_en | PCI Host Bridge Master Abort Interrupt Enable<br>0: Disables as interrupt source (initial value)<br>1: Enables as interrupt source<br>This interrupt is issued during debugging and is not used during normal operation. |
| 0 | tabortint_en | PCI Host Bridge Target Abort Interrupt Enable<br>0: Disables as interrupt source (initial value)<br>1: Enables as interrupt source<br>This interrupt is issued during debugging and is not used during normal operation. |

**(8) PCI bus error address register (PCI_ERR_ADD)**

After reset: 0000 0000H   R/W   Address: 002E1020H

| | 31 | 0 |
|---|---|---|
| PCI_ERR_ADD | err_adr[31:0] | |

| Bit position | Bit name | Function |
|---|---|---|
| 31-0 | err_adr [31:0] | PCI Bus Error Address<br>This field retains the PCI address at which any of the following error sources occurs.<br>• $\overline{\text{SERR}}$ input<br>• Parity Error<br>• PCI Bus Master Abort<br>• PCI Bus Target Abort<br>All bits in this field are cleared (0) when they are read.  Once a bus error occurs and a value is set to this register, the value is retained until it is read or it is updated due to occurrence of a new bus error.  This register is used during debugging and is not used during normal operation. |

**(9) Bus bridge control register (BRG_CTL)**

After reset: 0000 0000H    R/W    Address: 002E1024H

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| BRG_CTL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | cnv_err |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | data_wait_en | no_wait_en |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | cnv_err_en |

| Bit position | Bit name | Function |
|---|---|---|
| 16 | cnv_err | 16-32 Bits Cycle Convert Error<br>This bit is set when an error occurs due to reception of an unexpected cycle during conversion from 16 to 32 bits.  This bit reports occurrence of the INTUSBH1 interrupt.  This bit will never be set unless the cnv_err_en bit is set to "1".<br>0:  No interrupt source has occurred.<br>1:  Cycle Convert Error detected.<br>The INTUSBH1 (PCI cycle error) interrupt is issued during debugging and is not used during normal operation. |
| 9 | data_wait_en | Data Wait Enable<br>Sets forcible insertion of 1 wait (data wait) to the end of a CPU bus cycle.<br>0:  Does not insert a data wait forcibly (initial value).<br>1:  Forcibly inserts a data wait. |
| 8 | no_wait_en | No Wait Enable<br>Sets whether to enable no wait operation of a CPU bus cycle.<br>0:  Disables no wait operation (at least 1 wait is always inserted) (initial value)<br>1:  Enables wait operation (recommended)<br>Before changing the value of this bit from "1" to "0", it is recommended to read this bit once and then write the value so as to make up the time lag until the change is reflected.  The read value at that time may not be correct because waits have not been inserted correctly, so discard the read value. |
| 0 | cnv_err_en | 16-32 Bits Cycle Convert Error Enable<br>This bit is used to set and clear interrupt sources.  When it is set to "0", the cnv_err bit is cleared and masks the interrupt source at the same time.<br>0:  Clears the interrupt source (initial value)<br>1:  Enables the interrupt source.<br>The INTUSBH1 (PCI cycle error) interrupt is issued during debugging and is not used during normal operation. |

**(10) SRAM_BASE register (SRAM_BASE)**

After reset: 0028 0000H    R/W    Address: 002E1050H

|  | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| SRAM_BASE | sram_base31 | sram_base30 | sram_base29 | sram_base28 | sram_base27 | sram_base26 | sram_base25 | sram_base24 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| sram_base23 | sram_base22 | sram_base21 | sram_base20 | sram_base19 | sram_base18 | sram_base17 | sram_base16 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31-16 | sram_base [31:16] | SRAM Area Base Address<br>Sets the base address for accessing an SRAM area.<br>Sets the base address in the SRAM area in a PCI space.<br>When a PCI target attempts to access a memory space, the address is compared with the one specified in this register.  If they match, the relevant response is returned. |

## 21.7 OHCI Host Controller

### 21.7.1 OHCI host controller functions

The OHCI host controller has the following functions.

- Conforms to OpenHCI Specification Release 1.0a
- Conforms to Universal Serial Bus Specification Revision 1.1
  Supports transfer at full-speed (12 Mbps)
- Incorporates 1 channel root hub and supports one downstream port
- USB clock: 48 MHz, PCI clock: 25 to 33 MHz
- Memory space
  4 KB PCI memory area (OHCI operational registers) allocated
  256-byte host controller communication area (HCCA) allocated
- Communication with CPU
  Performs communication via operational registers in the OHCI host controller and the host controller communication area (HCCA).

Two communication channels are connected between the CPU and the OHCI host controller. The communication channel consists of OHCI operational registers, and the OHCI host controller serves as a target (slave) for this communication. The Base Address register (10H) in the OHCI host configuration registers functions as a pointer to the OHCI operational registers.

The second communication channel is the host controller communication area (HCCA) in the OHCI operational registers, which functions as a pointer to the shared memory. The OHCI host controller serves as a master for this communication.

Descriptor information used for communication is managed by the OHCI operational registers and the HCCA area.

### 21.7.2  OHCI host configuration registers

The OHCI host configuration registers configure a 256-byte register space, and are incorporated in the OHCI host controller.  These registers are accessed from the CPU system via PCI host bridge registers (PCI_CONFIG_DATA, PCI_CONFIG_ADD).  For addresses, see **21.2.2 USB host controller memory map**.

**Table 22-4.  PCI Configuration Registers**

| Bit Address | 31 ... 24 | 23 ... 16 | 15 ... 8 | 7 ... 0 |
|---|---|---|---|---|
| 00H | Device ID | | Vendor ID | |
| 04H | Status | | Command | |
| 08H | Class Code | | | Revision ID |
| 0CH | BIST | Header Type | Latency Timer | Cache Line Size |
| 10H | Base Address | | | |
| 14H | Reserved | | | |
| 18H | | | | |
| 1CH | | | | |
| 20H | | | | |
| 24H | | | | |
| 28H | Reserved | | | |
| 2CH | Subsystem ID | | Subsystem Vendor ID | |
| 30H | Reserved | | | |
| 34H | Reserved | | | Cap_ptr |
| 38H | Reserved | | | |
| 3CH | Max_Latency | Min_Gnt | Interrupt Pin | Interrupt Line |
| 40H | PMC | | Next_Item_Ptr | Cap_ID |
| 44H | Data | DMCSR_BSE | PMCSR | |
| E0H | Reserved | | | |

**(1) Vendor ID and Device ID (Offset 00H)**

After reset: 0035 1033H    R    Address: 002E1050H
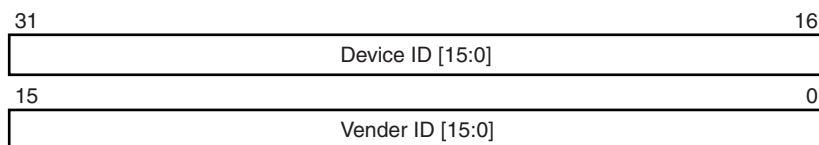
| 31 | 16 |
|---|---|
| Device ID [15:0] | |

| 15 | 0 |
|---|---|
| Vender ID [15:0] | |

| Bit position | Bit name | Function |
|---|---|---|
| 31-16 | Device ID [15:0] | Indicates the device type.<br>It is used to select the driver for driving a device according to the PCI Specification.<br>This field is fixed to 0035H. |
| 15-0 | Vender ID [15:0] | Indicates the device vendor.<br>It is used to select the driver for driving a device according to the PCI Specification.<br>This field is fixed to 1033H. |

## (2) Command and Status (Offset 04H)

After reset: 0210 0000H    R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Detected Parity Error | Signaled System Error | Received Master Abort | Received Target Abort | Signaled Target Abort | Devsel Timing1 | Devsel Timing0 | Data Parity Detected |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| Fast Back to Back Capable | 0 | 0 | Capabilities | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | Fast Back to Back Enable | SERR Enable |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Wait Cycle Control | Parity Error Response | VGA Pallet Snoop | Memory Write and Invalidate | Special Cycle | Bus Master | Memory Space | I/O Space |

| Bit position | Bit name | Function |
|---|---|---|
| 31 | Detected Parity Error | Indicates the parity error status. This bit is set when an address or data parity error is detected. This bit is cleared (0) when "1" is written via the PCI bus. |
| 30 | Signaled System Error | Indicates the SERR status. This bit is set when a system error occurs. This bit is cleared (0) when "1" is written via the PCI bus. |
| 29 | Received Master Abort | Indicates the master's master abort status. This bit is set when master operation is terminated by master abort. This bit is cleared (0) when "1" is written via the PCI bus. |
| 28 | Received Target Abort | Indicates the master's target abort status. This bit is set when master operation is terminated by target abort. This bit is cleared (0) when "1" is written via the PCI bus. |
| 27 | Signaled Target Abort | Indicates the slave's target abort status. This bit is set when slave operation is terminated by target abort. This bit is cleared (0) when "1" is written via the PCI bus. |
| 26-25 | Devsel Timing[1:0] | Indicates the DEVSEL response speed. This field is fixed to "01" because only the Medium mode is supported. These bits are read-only. |
| 24 | Data Parity Detected | This bit is set when a parity error is detected during master operation. This bit is cleared (0) when "1" is written via the PCI bus. This bit is fixed to "0" when the parity error response is disabled with the Parity Error Response bit (Command register). |
| 23 | Fast Back to Back Capable | Indicates the support status of Fast Back-to-Back. This bit is fixed to "0" because Fast Back-to-Back is not supported. This bit is read-only. |
| 22-21 | – | Reserved. (Be sure to write "0" to these bits.) |
| 20 | Capabilities | Indicates that the Power Management mode is supported. This bit is fixed to "1". This bit is read-only. |
| 19-10 | – | Reserved. (Be sure to write "0" to these bits.) |
| 9 | Fast Back to Back Enable | Fast Back-to-Back Enable Bit This bit is fixed to "0" because the USB host controller does not support Fast Back to Back. This bit is read-only. |

| Bit position | Bit name | Function |
|---|---|---|
| 8 | SERR Enable | SERR Enable Bit<br>Set this bit to "1" when sending system errors via the SERR signal. |
| 7 | Wait Cycle Control | Wait Cycle Control Enable Bit<br>This bit is fixed to "0" because the USB host controller does not support address/data stepping.<br>This bit is read-only. |
| 6 | Parity Error Response | Parity Error Response Enable Bit<br>Set this bit to "1" when checking parity errors. |
| 5 | VGA Pallet Snoop | VGA Palette Snoop Enable Bit<br>This bit is fixed to "0" because the USB host controller does not support VGA palette snoop.<br>This bit is read-only. |
| 4 | Memory Write and Invalidate | Memory Write and Invalidate Enable Bit<br>This bit is fixed to "0" because the USB host controller does not support Memory Write and Invalidate.<br>This bit is read-only. |
| 3 | Special Cycle | Special Cycle Enable Bit<br>This bit is fixed to "0" because the USB host controller does not support Special Cycle.<br>This bit is read-only. |
| 2 | Bus Master | Bus Master Enable Bit<br>Enables bus master accesses for the PCI bus and must be set to "1" before accessing SRAM via the system bus. Set this bit to "1" upon host controller initialization. |
| 1 | Memory Space | Memory Space Access Enable Bit<br>Enables memory accesses according to the PCI Specification, and must be set to "1" before accessing registers. Set this bit to "1" upon host controller initialization. |
| 0 | I/O Space | I/O Space Access Enable Bit<br>Enables I/O accesses defined in the PCI Specification, but this bit is fixed to "0" because the USB host controller does not use I/O accesses.<br>This bit is read-only. |

**(3)  Revision ID and Class Code (Offset 08H)**

After reset: 0C03 1042H    R

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Base Class7 | Base Class6 | Base Class5 | Base Class4 | Base Class3 | Base Class2 | Base Class1 | Base Class0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| Sub Class7 | Sub Class6 | Sub Class5 | Sub Class4 | Sub Class3 | Sub Class2 | Sub Class1 | Sub Class0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Program-ming I/F7 | Program-ming I/F6 | Program-ming I/F5 | Program-ming I/F4 | Program-ming I/F3 | Program-ming I/F2 | Program-ming I/F1 | Program-ming I/F0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Revision ID7 | Revision ID6 | Revision ID5 | Revision ID4 | Revision ID3 | Revision ID2 | Revision ID1 | Revision ID0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31-24 | Base Class [7:0] | Indicates the base class prescribed in the PCI Specification (Class Code). This field is fixed to 0CH, which indicates the serial peripheral bus controller. |
| 23-16 | Sub Class [7:0] | Indicates the subclass prescribed in the PCI Specification (Class Code). This field is fixed to 03H, which indicates the USB device. |
| 15-8 | Programming I/F[7:0] | Indicates the program interface prescribed in the PCI Specification (Class Code). This field is fixed to 10H, which indicates OHCI specification USB. |
| 7-0 | Revision ID [7:0] | Indicates the revision of the USB host controller (Class Code). This field is fixed to 42H. |

**(4)  Cache Line Size, Latency Timer, Header Type, and BIST (Offset 0CH)**

After reset: 0000 0800H    R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| BIST7 | BIST6 | BIST5 | BIST4 | BIST3 | BIST2 | BIST1 | BIST0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| Header Type7 | Header Type6 | Header Type5 | Header Type4 | Header Type3 | Header Type2 | Header Type1 | Header Type0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Latency Timer7 | Latency Timer6 | Latency Timer5 | Latency Timer4 | Latency Timer3 | Latency Timer2 | Latency Timer1 | Latency Timer0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Cache Line Size7 | Cache Line Size6 | Cache Line Size5 | Cache Line Size4 | Cache Line Size3 | Cache Line Size2 | Cache Line Size1 | Cache Line Size0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31-24 | BIST[7:0] | Field for self-testing.<br>This field is fixed to 00H.<br>These bits are read-only. |
| 23-16 | Header Type [7:0] | Field for reporting the header type to the system<br>This field is fixed to 00H because the device is a PCI device.<br>Particularly, bit 23 is fixed to 0 because multifunction is not supported.<br>These bits are read-only. |
| 15-8 | Latency Timer[7:0] | Field for reporting the latency timer to the system<br>The lower 2 bits are fixed to "00".<br>These bits are read-only. |
| 7-0 | Cache Line Size[7:0] | Field for reporting the cache line size to the system<br>This field is fixed to 00H.<br>These bits are read-only. |

**(5) Base Address (Offset 10H)**

After reset: 0000 0000H R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| OHCI Base Address27 | OHCI Base Address26 | OHCI Base Address25 | OHCI Base Address24 | OHCI Base Address23 | OHCI Base Address22 | OHCI Base Address21 | OHCI Base Address20 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| OHCI Base Address19 | OHCI Base Address18 | OHCI Base Address17 | OHCI Base Address16 | OHCI Base Address15 | OHCI Base Address14 | OHCI Base Address13 | OHCI Base Address12 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| OHCI Base Address11 | OHCI Base Address10 | OHCI Base Address9 | OHCI Base Address8 | OHCI Base Address7 | OHCI Base Address6 | OHCI Base Address5 | OHCI Base Address4 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OHCI Base Address3 | OHCI Base Address2 | OHCI Base Address1 | OHCI Base Address0 | Prefetchable | Type1 | Type0 | Memory Space Indicator |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 4 | OHCI Base Address [27:0] | Specifies the operational register address by using bits 31 to 12.<br>Set the operational register base address defined by the system during initialization.<br>Bits 11 to 4 are fixed to 00H.<br>Bits 11 to 4 are read-only. |
| 3 | Prefetchable | Field indicating that the field specified with Base Address is a memory space.<br>This bit is fixed to 0, which indicates that prefetching from the field is prohibited.<br>This bit is read-only. |
| 2, 1 | Type[1:0] | Field that indicates the base address type. This field is fixed to 00H. Indicates that the address specified with the OHCI Base Address field is a specific position in a 32-bit space.<br>These bits are read-only. |
| 0 | Memory Space Indicator | Field indicating that the field specified with Base Address is a memory space.<br>This bit is fixed to 0.<br>This bit is read-only. |

**(6) Subsystem Vendor ID and Subsystem ID (Offset 2CH)**

After reset: 0133 1033H    R

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| SubSystem ID15 | SubSystem ID14 | SubSystem ID13 | SubSystem ID12 | SubSystem ID11 | SubSystem ID10 | SubSystem ID9 | SubSystem ID8 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| SubSystem ID7 | SubSystem ID6 | SubSystem ID5 | SubSystem ID4 | SubSystem ID3 | SubSystem ID2 | SubSystem ID1 | SubSystem ID0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SubSystem Vender ID15 | SubSystem Vender ID14 | SubSystem Vender ID13 | SubSystem Vender ID12 | SubSystem Vender ID11 | SubSystem Vender ID10 | SubSystem Vender ID9 | SubSystem Vender ID8 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SubSystem Vender ID7 | SubSystem Vender ID6 | SubSystem Vender ID5 | SubSystem Vender ID4 | SubSystem Vender ID3 | SubSystem Vender ID2 | SubSystem Vender ID1 | SubSystem Vender ID0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 16 | SubSystem ID[15:0] | Indicates the device type.<br>It is used to select the driver for driving a device according to the PCI Specification.<br>This field is fixed to 0133H. |
| 15 to 0 | SubSystem Vender ID [15:0] | Indicates the device vendor.<br>It is used to select the driver for driving a device according to the PCI Specification.<br>This field is fixed to 1033H. |

**(7) Capability Pointer (Offset 34H)**

After reset: 0000 0040H    R

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|----|----|
| Capability Pointer7 | Capability Pointer6 | Capability Pointer5 | Capability Pointer4 | Capability Pointer3 | Capability Pointer2 | Capability Pointer1 | Capability Pointer0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 10 | – | Reserved. (Be sure to write "0" to these bits.) |
| 7 to 0 | Capability Pointer[7:0] | This field is fixed to 40H. |

**(8) Interrupt Line, Interrupt Pin, Min Gnt, Max Latency (Offset 3CH)**

After reset: 2A01 0100H　　R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Max Latency7 | Max Latency6 | Max Latency5 | Max Latency4 | Max Latency3 | Max Latency2 | Max Latency1 | Max Latency0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| Min Gnt7 | Min Gnt6 | Min Gnt5 | Min Gnt4 | Min Gnt3 | Min Gnt2 | Min Gnt1 | Min Gnt0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Interrupt Pin7 | Interrupt Pin6 | Interrupt Pin5 | Interrupt Pin4 | Interrupt Pin3 | Interrupt Pin2 | Interrupt Pin1 | Interrupt Pin0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Interrupt Line7 | Interrupt Line6 | Interrupt Line5 | Interrupt Line4 | Interrupt Line3 | Interrupt Line2 | Interrupt Line1 | Interrupt Line0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 24 | Max Latency[7:0] | Indicates the maximum latency. Fixed to 2AH.<br>These bits are read-only. |
| 23 to 16 | Min Gnt[7:0] | Indicates the minimum grant time. Fixed to 01H.<br>These bits are read-only. |
| 15 to 8 | Interrupt Pin[7:0] | Indicates the interrupt output pin. Fixed to 01H because INTA is used.<br>These bits are read-only. |
| 7 to 0 | Interrupt Line[7:0] | Indicates the interrupt line. Fixed to 00H. |

**(9) Capability Identifier, Next Item Pointer, Power Management Capabilities (Offset 40H)**

After reset: 7E02 0001H    R

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| PME Support4 | PME Support3 | PME Support2 | PME Support1 | PME Support0 | D2 Support | D1 Support | Aux Current2 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| Aux Current1 | Aux Current0 | DSI | 0 | PME CLK | Version2 | Version1 | Version0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Next Item Pointer7 | Next Item Pointer6 | Next Item Pointer5 | Next Item Pointer4 | Next Item Pointer3 | Next Item Pointer2 | Next Item Pointer1 | Next Item Pointer0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Capability Identifier7 | Capability Identifier6 | Capability Identifiere5 | Capability Identifier4 | Capability Identifier3 | Capability Identifier2 | Capability Identifier1 | Capability Identifier0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 | PME Support[4:0] | Indicates whether the D3 Cold state is supported (Power Management Capabilities). This bit is fixed to "0B" because the D3 Cold state is not supported. |
| 30 to 27 | | Indicates that PME (Power Management Capabilities) is supported in all PCI power states (D0 to D3). These bits are fixed to "1111". |
| 26 | D2 Support | Indicates that D2 of the PCI power states is supported (Power Management Capabilities). This bit is fixed to "1". |
| 25 | D1 Support | Indicates that D1 of the PCI power states is supported (Power Management Capabilities). This bit is fixed to "1'. |
| 24 to 22 | Aux Current [2:0] | Indicates that asserting PME interrupts in the D3 Cold state is not supported (Power Management Capabilities). These bits are therefore fixed to "000". |
| 21 | DSI | Indicates that no special initialization is required when using Power Management (Power Management Capabilities). This bit is fixed to "0". |
| 19 | PME CLK | Indicates that PCLK is not required for generating PME interrupts (Power Management Capabilities). This bit is fixed to "0". |
| 18 to 16 | Version[2:0] | Indicates the Power Management version (Power Management Capabilities). These bits are fixed to "010" according to the circuit configuration implemented in the USB host controller. |
| 15 to 8 | Next Item Pointer[7:0] | Indicates that the next item does not exist. These bits are fixed to "00H". |
| 7 to 0 | Capability Identifier[7:0] | Indicates the Power Management Register ID These bits are fixed to "01H". |

**(10) Power Management Control/Status, PMCSR Bridge Support Extensions, Data (Offset 44H)**

After reset: 0000 0000H    R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Data7 | Data6 | Data5 | Data4 | Data3 | Data2 | Data1 | Data0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| BPCC Enable | B2_B3 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| PME Status | Data Scale1 | Data Scale0 | Data Select3 | Data Select2 | Data Select1 | Data Select0 | PME Enable |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | Power State1 | Power State0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 24 | Data[7:0] | This is an optional field in the PCI Specification, and is not supported by the USB host controller (Data). <br> This field is fixed to "00". <br> These bits are read-only. |
| 23 | BPCC Enable | Bit for Bridge, which is not supported by the USB host controller (PMCSR Bridge Support Extensions). <br> This field is fixed to "0". <br> This bit is read-only. |
| 22 | B2_B3 | Bit for Bridge, which is not supported by the USB host controller (PMCSR Bridge Support Extensions). <br> This field is fixed to "0". <br> This bit is read-only. |
| 15 | PME Status | Indicates the PME interrupt status (Power Management Control/Status). <br> This bit is set to "1" when conditions for asserting PME are satisfied. <br> This bit is cleared to "0" when "1" is written via the PCI bus. |
| 14, 13 | Data Scale [1:0] | This field is optional in the PCI Specification, and is not supported by the host controller (Power Management Control/Status). <br> This field is fixed to "00". <br> These bits are read-only. |
| 12 to 9 | Data Select [3:0] | This field is optional in the PCI Specification, and is not supported by the host controller (Power Management Control/Status). <br> This field is fixed to "0000". <br> These bits are read-only. |
| 8 | PME Enable | Bit for setting whether to use external pin PME (Power Management Control/Status). <br> If this bit is set to 1B, a PME interrupt occurs when returning from Power Management mode. |

| Bit position | Bit name | Function |
|---|---|---|
| 1, 0 | Power State[1:0] | Indicates the PCI power status (Power Management Control/Status). The status varies depending on the setting of this field.<br><br>{TABLE}<br> |

{TABLE}:

| Power State1 | Power State0 | Field Status |
|---|---|---|
| 0 | 0 | D0 State |
| 0 | 1 | D1 State |
| 1 | 0 | D2 State |
| 1 | 1 | D3 hot State |

### 21.7.3 OHCI operational registers

The OHCI operational registers are incorporated in the OHCI host controller, and consist of the registers listed in Table 21-5. For details, see OpenHCI Specification Release 1.0a.

Since OpenHCI Specification Release 1.0a defines the port number as [1: number of ports], this section observes this rule. For example, port 1 corresponds to host channel 0 and port 2 corresponds to host channel 1. For the offset addresses, see **21.2.2 USB host controller memory map**.

**Table 21-5. OHCI Operational Registers**

| Bit\Address | 31      24 | 23      16 | 15      8 | 7      0 |
|---|---|---|---|---|
| 00H | HcRevision | | | |
| 04H | HcControl | | | |
| 08H | HcCommandStatus | | | |
| 0CH | HcInterruptStatus | | | |
| 10H | HcInterruptEnable | | | |
| 14H | HcInterruptDisable | | | |
| 18H | HcHCCA | | | |
| 1CH | HcPeriodCurrentED | | | |
| 20H | HcControlHeadED | | | |
| 24H | HcControlCurrentED | | | |
| 28H | HcBulkHeadED | | | |
| 2CH | HcBulkCurrentED | | | |
| 30H | HcDoneHead | | | |
| 34H | HcFmInterval | | | |
| 38H | HcFmRemaining | | | |
| 3CH | HcFmNumber | | | |
| 40H | HcPeriodicStart | | | |
| 44H | HcLSThreshold | | | |
| 48H | HcRhDescriptorA | | | |
| 4CH | HcRhDescriptorB | | | |
| 50H | HcRhStatus | | | |
| 54H | HcRhPortStatus1 | | | |
| 58H | HcRhPortStatus2 | | | |
| 5CH to FFH | Reserved | | | |

**Remark**    HC:      Host controller (OHCI host controller)
                  HCD:    Host controller driver
                  ED:      Endpoint descriptor
                  TD:      Transfer descriptor
                  EOP:    End of packet
                  SOF:    Start of frame

**(1) HcRevision register (Offset 00H)**

After reset: 0000 0010H    R

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | Legacy |

| 7 | | | | | | | 0 |
|---|---|---|---|---|---|---|---|
| Revision | | | | | | | |

| Bit position | Bit name | Function |
|--------------|----------|----------|
| 31 to 9 | – | Reserved.  (Be sure to write "0" to these bits.) |
| 8 | Legacy | Indicates whether legacy-support registers are implemented in the OHCI host controller. Fixed to "0" because the OHCI host controller does not support the legacy function. |
| 7 to 0 | Revision | Indicates the version of the OHCI Specification implemented in the OHCI host controller Fixed to 10H because the OHCI host controller is compliant with OHCI Specification 1.0a. |

**(2) HcControl register (Offset 04H)**

After reset: 0000 0000H  R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | RWE | RWC | IR |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|----|----|
| HCFS1 | HCFS0 | BLE | CLE | IE | PLE | CBSR1 | CBSR0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 11 | − | Reserved.  (Be sure to write "0" to these bits.) |
| 10 | RWE | Remote WakeUp Enable<br>Sets detection of upstream resume signals.<br>    1: Detects the Resume signal as Remote Wakeup.<br>    0: Does not detect the Resume signal as Remote Wakeup. |
| 9 | RWC | Remote WakeUp Connect<br>Indicates whether the OHCI host controller supports Remote Wakeup.<br>This bit must be set during initialization if remote wakeup is supported in the system.<br>    1: Supports Remote Wakeup.<br>    0: Does not support Remote Wakeup. |
| 8 | IR | Interrupt Routing<br>Indicates the route for outputting interrupts via the OHCI host controller.<br>Sets how to report to the HcInterruptStatus register the source of an interrupt that has occurred.<br>    1: Reports the occurrence of an interrupt via SMMI.<br>    0: Reports the occurrence of an interrupt INTA. |
| 7 to 6 | HCFS[1:0] | Host Controller Functional Status<br>Indicates the OHCI host controller operating status.<br><br>| HCFS | USB Status |<br>|---|---|<br>| 00 | USB Reset |<br>| 01 | USB Resume |<br>| 10 | USB Operational |<br>| 11 | USB Suspend |<br><br>When the OHCI host controller shifts to the USB Operational state, it starts management of frames in 1 ms units.<br>This status is always controlled by the host controller driver (HCD) except for when the status shifts to USB Resume by remote wakeup during USB Suspend.<br>This field is set to "00" after hardware reset, or "11" after software reset. |

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 5 | BLE | Bulk List Enable<br>Sets whether to perform bulk list processing.<br>  1: Performs bulk list processing.<br>  0: Does not perform bulk list processing.<br>The setting of this bit becomes valid from the next frame of bulk list processing.<br>This bit must be set to "0" when modifying a bulk list. |
| 4 | CLE | Control List Enable<br>Sets whether to perform control list processing.<br>  1: Performs control list processing.<br>  0: Does not perform control list processing.<br>The setting of this bit becomes valid from the next frame of control list processing.<br>This bit must be set to "0" when modifying a control list. |
| 3 | IE | Isochronous Enable<br>Sets whether to perform isochronous ED processing.<br>This bit is checked when an isochronous ED is detected during list processing, and whether to perform isochronous ED processing is determined.<br>  1: Performs isochronous transfer processing.<br>  0: Does not perform isochronous transfer processing.<br>The setting of this bit becomes valid from the next frame of isochronous transfer processing. |
| 2 | PLE | Periodic List Enable<br>Sets whether to perform periodic list processing.<br>  1: Performs periodic list processing.<br>  0: Does not perform periodic list processing.<br>The setting of this bit becomes valid from the next frame of periodic list processing. |
| 1, 0 | CBSR[1:0] | Control Bulk Service Ratio<br>Defines the ratio of control transfer and bulk transfer services.<br>The ratio defined in this field is kept during periodic list processing for performing transfer.<br><br>                                          <table><tr><td>CBSR</td><td>Service Ratio of Bulk ED and Control ED</td></tr><tr><td>00</td><td>1:1</td></tr><tr><td>01</td><td>2:1</td></tr><tr><td>10</td><td>3:1</td></tr><tr><td>11</td><td>4:1</td></tr></table> |

**(3) HcCommandStatus register (Offset 08H)**

After reset: 0000 0000H    R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | SOC1 | SOC0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | OCR | BLF | CLF | HCR |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 18 | _ | Reserved. (Be sure to write "0" to these bits.) |
| 17, 16 | SOC[1:0] | Scheduling Overrun Count<br>Counts the schedule overrun count. Counts up each time a schedule overrun occurs.<br>Counting up continues even if the SO bit of the HcInterruptStatus register is set to "1".<br>These bits are read-only. |
| 15 to 4 | _ | Reserved. (Be sure to write "0" to these bits.) |
| 3 | OCR | Ownership Change Request<br>Requests to change the control right of the OHCI host controller. |
| 2 | BLF | Bulk List Filled<br>Indicates whether a TD exists in a bulk list.<br>This bit is always set to "1" by the host controller driver (HCD) when a TD is added to the ED of a bulk list.<br>The OHCI host controller checks this bit when starting bulk list head processing.<br>List head processing is not started if a TD exists while this bit is set to "0". If this bit is set to "1", the OHCI host controller sets it to "0" and control ED processing starts. If a TD is detected in a bulk list, this bit is set to "1" again to continue bulk TD processing.<br>The driver must set this bit before reconfiguring the list, setting the BLE bit of the HcCommand register, and starting list processing. |
| 1 | CLF | Control List Filled<br>Indicates whether a control list exists.<br>This bit is always set to "1" by the host controller driver (HCD) when a TD is added to the ED of a control list.<br>The OHCI host controller checks this bit when starting control list head processing.<br>List head processing is not started while this bit is set to "0". If this bit is set to "1", the OHCI host controller sets it to "0" and control list processing starts. If a TD is detected in a control list, this bit is set to "1" again to continue control list processing.<br>The driver must set this bit before reconfiguring the list, setting the CLE bit of the HcCommand register, and starting list processing. |

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 0 | HCR | Host Controller Reset<br>Activates software reset via the OHCI host controller.<br>When this bit is set, the USB Suspend state is entered regardless of the OHCI host controller function state.<br>This bit is cleared (0) by the OHCI host controller upon completion of reset operation. |

**(4) HcInterruptStatus register (Offset 0CH)**

(1/2)

After reset: 0000 0000H R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0 | OC | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|------|-----|----|----|----|-----|----|
| 0 | RHSC | FNO | UE | RD | SF | WDH | SO |

| Bit position | Bit name | Function |
|--------------|----------|----------|
| 31 | _ | Reserved. (Be sure to write "0" to these bits.) |
| 30 | OC | Ownership Change<br>Indicates the occurrence of an Ownership Change interrupt.<br>This bit is set (1) when the OCR field of the HcCommandStatus register is set to "1".<br>An SMMI interrupt occurs if the interrupt source is not masked.<br> 1: An OC interrupt has occurred.<br> 0: An OC interrupt has not occurred.<br>Writing "1" to this bit clears the interrupt source. |
| 29 to 7 | _ | Reserved. (Be sure to write "0" to these bits.) |
| 6 | RHSC | Root Hub Status Change<br>Indicates that the HcRhPortStatus register setting has been changed.<br>This bit is set to "1" when the status of the HcRhPortStatus register has been changed due to hardware.<br> 1: An RHSC interrupt has occurred.<br> 0: An RHSC interrupt has not occurred.<br>Writing "1" to this bit clears the interrupt source. |
| 5 | FNO | Frame Number Overflow<br>Indicates that the MSB of a frame number has been changed.<br>This bit is set to "1" after HccaFrameNumber is updated in a frame whose frame number MSB changes from "0" to "1", or "1" to "0".<br> 1: An FNO interrupt has occurred.<br> 0: An FNO interrupt has not occurred.<br>Writing "1" to this bit clears the interrupt source. |
| 4 | UE | Unrecoverable Error<br>Indicates that a system error has been detected on the PCI bus not related to the USB.<br> 1: A UE interrupt has occurred.<br> 0: A UE interrupt has not occurred.<br>Writing "1" to this bit clears the interrupt source. |

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 3 | RD | Resume Detected<br>Indicates that Resume has been detected.<br>This bit is set to "1" when asserting of the resume signal by a device on the USB bus is detected. This bit is not set if USB Resume is issued by the host controller driver (HCD).<br>　1: An RD interrupt has occurred.<br>　0: An RD interrupt has not occurred.<br>Writing "1" to this bit clears the interrupt to "0". |
| 2 | SF | StartOfFrame<br>Indicates that HccaFrameNumber was updated at the beginning of a frame. The OHCI host controller transmits the SOF packet and updates HccaFrameNumber.<br>　1: An SF interrupt has occurred.<br>　0: An SF interrupt has not occurred.<br>Writing "1" to this bit clears the interrupt to "0". |
| 1 | WDH | Writeback Done Head<br>Indicates that the USB host controller updated HccDoneHead. The OHCI host controller sets this bit immediately to "1" after updating HccaDoneHead, and does not update HccaDoneHead until this bit is cleared to "0".<br>　1: A WDH interrupt has occurred.<br>　0: A WDH interrupt has not occurred.<br>Writing "1" to this bit clears the interrupt to "0". |
| 0 | SO | Scheduling Overrun<br>Indicates that the USB scheduling in a frame has overrun. This bit is set to "1" after updating of FrameNumberUpdate of the frame following the frame in which the USB scheduling overruns. When this bit is set to "1", the SOC field of the HcCommandStatus register is incremented.<br>　1: An SO interrupt has occurred.<br>　0: An SO interrupt has not occurred.<br>Writing "1" to this bit clears the interrupt to "0". |

**(5) HcInterruptEnable register (Offset 10H)**

(1/2)

After reset: 0000 0000H　R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| MIE | OCE | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 0 | RHSCE | FNOE | UEE | RDE | SFE | WDHE | SOE |

| Bit position | Bit name | Function |
|-----|-----|-----|
| 31 | MIE | Master Interrupt Enable<br>Specifies whether to enable interrupt sources set by bits 30 and 6 to 0.<br>　1: Enables all the specified interrupt sources.<br>　0: Ignored<br>To clear this bit to "0", write "1" to the corresponding bit of the HcInterruptDisable register. |
| 30 | OCE | Ownership Change Enable<br>Specifies whether to enable OC as an interrupt source.<br>　1: Enables OC as an interrupt source.<br>　0: Ignored<br>Writing "1" to this bit sets OC as an interrupt source.<br>To clear this bit to "0", write "1" to the corresponding bit of the HcInterruptDisable register. |
| 29 to 7 | – | Reserved.  (Be sure to write "0" to these bits.) |
| 6 | RHSCE | Root Hub Status Change Enable<br>Specifies whether to enable RHSC as an interrupt source.<br>　1: Enables RHSC as an interrupt source.<br>　0: Ignored<br>Writing "1" to this bit sets to RHSC as an interrupt source.<br>To clear this bit to "0", write "1" to the corresponding bit of the HcInterruptDisable register. |
| 5 | FNOE | Frame Number Overflow Enable<br>Specifies whether to enable FNO as an interrupt source.<br>　1: Enables FNO as an interrupt source.<br>　0: Ignored<br>Writing "1" to this bit sets FNO as an interrupt source.<br>To clear this bit to "0", write "1" to the corresponding bit of the HcInterruptDisable register. |
| 4 | UEE | Unrecoverable Error Enable<br>Specifies whether to enable UE as an interrupt source.<br>　1: Enables UE as an interrupt source.<br>　0: Ignored<br>Writing "1" to this bit sets UE as an interrupt source.<br>To clear this bit to "0", write "1" to the corresponding bit of the HcInterruptDisable register. |

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 3 | RDE | Resume Detected Enable<br>Specifies whether to enable RD as an interrupt source.<br> 1: Enables RD as an interrupt source.<br> 0: Ignored<br>Writing "1" to this bit sets RD as an interrupt source.<br>To clear this bit to "0", write "1" to the corresponding bit of the HcInterruptDisable register. |
| 2 | SFE | Start Of Frame Enable<br>Specifies whether to enable SF as an interrupt source.<br> 1: Enables SF as an interrupt source.<br> 0: Ignored<br>Writing "1" to this bit sets SF as an interrupt source.<br>To clear this bit to "0", write "1" to the corresponding bit of the HcInterruptDisable register. |
| 1 | WDHE | Writeback Done Head Enable<br>Specifies whether to enable WDH as an interrupt source.<br> 1: Enables WDH as an interrupt source.<br> 0: Ignored<br>Writing "1" to this bit sets WDH as an interrupt source.<br>To clear this bit to "0", write "1" to the corresponding bit of the HcInterruptDisable register. |
| 0 | SOE | Scheduling Overrun Enable<br>Specifies whether to enable SO as an interrupt source.<br> 1: Enables SO as an interrupt source.<br> 0: Ignored<br>Writing "1" to this bit sets SO as an interrupt source.<br>To clear this bit to "0", write "1" to the corresponding bit of the HcInterruptDisable register. |

**(6) HcInterruptDisable register (Offset 14H)**

After reset: 0000 0000H    R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| MID | OCD | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|----|----|
| 0 | RHSCD | FNOD | UED | RDD | SFD | WDHD | SOD |

| Bit position | Bit name | Function |
|---|---|---|
| 31 | MID | Master Interrupt Disable<br>Sets whether to disable interrupt sources set by bits 30 and 6 to 0 of the HcInterruptEnable register.<br>  1: Disables all the specified interrupt sources.<br>  0: Ignored<br>When this bit is read, the value of the corresponding bit of the HcInterruptEnable register is read.<br>To set this bit to "1", write "1" to the corresponding bit of the HcInterruptEnable register. |
| 30 | OCD | Ownership Change Disable<br>Sets whether to remove OC from the interrupt sources.<br>  1: Disables OC as an interrupt source.<br>  0: Ignored<br>When this bit is read, the value of the corresponding bit of the HcInterruptEnable register is read.  Writing "1" to this bit clears OC from the interrupt sources.<br>To set this bit to "1", write "1" to the corresponding bit of the HcInterruptEnable register. |
| 29 to 7 | – | Reserved.  (Be sure to write "0" to these bits.) |
| 6 | RHSCD | Root Hub Status Change Disable<br>Sets whether to remove RHSC from the interrupt sources.<br>  1: Disables RHSC as an interrupt source.<br>  0: Ignored<br>When this bit is read, the value of the corresponding bit of the HcInterruptEnable register is read.  Writing "1" to this bit clears RHSC from the interrupt sources.<br>To set this bit to "1", write "1" to the corresponding bit of the HcInterruptEnable register. |
| 5 | FNOD | Frame Number Overflow Disable<br>Sets whether to remove FNO from the interrupt sources.<br>  1: Disables FNO as an interrupt source.<br>  0: Ignored<br>When this bit is read, the value of the corresponding bit of the HcInterruptEnable register is read.  Writing "1" to this bit clears FNO from the interrupt sources.<br>To set this bit to "1", write "1" to the corresponding bit of the HcInterruptEnable register. |

(2/2)

| Bit position | Bit name | Function |
|---|---|---|
| 4 | UED | Unrecoverable Error Disable<br>Sets whether to remove UE from the interrupt sources.<br>  1: Disables UE as an interrupt source.<br>  0: Ignored<br>When this bit is read, the value of the corresponding bit of the HcInterruptEnable register is read. Writing "1" to this bit clears to 0 from the interrupt sources.<br>To set this bit to "1", write "1" to the corresponding bit of the HcInterruptEnable register. |
| 3 | RDD | Resume Detected Disable<br>Sets whether to remove RD from the interrupt sources.<br>  1: Disables RD as an interrupt source.<br>  0: Ignored<br>When this bit is read, the value of the corresponding bit of the HcInterruptEnable register is read. Writing "1" to this bit clears RD from the interrupt sources.<br>To set this bit to "1", write "1" to the corresponding bit of the HcInterruptEnable register. |
| 2 | SFD | StartOfFrame Disable<br>Sets removing of SF from interrupt sources.<br>  1: Disables SF as an interrupt source.<br>  0: Ignored<br>When this bit is read, the value of the corresponding bit of the HcInterruptEnable register is read. Writing "1" to this bit clears SF from the interrupt sources.<br>To set this bit to "1", write "1" to the corresponding bit of the HcInterruptEnable register. |
| 1 | WDHD | Writeback Done Head Disable<br>Sets whether to remove WDH from the interrupt sources.<br>  1: Disables WDH as an interrupt source.<br>  0: Ignored<br>When this bit is read, the value of the corresponding bit of the HcInterruptEnable register is read. Writing "1" to this bit clears WDH from the interrupt sources.<br>To set this bit to "1", write "1" to the corresponding bit of the HcInterruptEnable register. |
| 0 | SOD | Scheduling Overrun Disable<br>Sets whether to remove SO from interrupt sources.<br>  1: Disables SO as an interrupt source.<br>  0: Ignored<br>When this bit is read, the value of the corresponding bit of the HcInterruptEnable register is read. Writing "1" to this bit clears SO from the interrupt sources.<br>To set this bit to "1", write "1" to the corresponding bit of the HcInterruptEnable register. |

**(7)  HcHCCA register (Offset 18H)**

After reset: 0000 0000H    R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| HcHCCA23 | HcHCCA22 | HcHCCA21 | HcHCCA20 | HcHCCA19 | HcHCCA18 | HcHCCA17 | HcHCCA16 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| HcHCCA15 | HcHCCA14 | HcHCCA13 | HcHCCA12 | HcHCCA11 | HcHCCA10 | HcHCCA9 | HcHCCA8 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| HcHCCA7 | HcHCCA6 | HcHCCA5 | HcHCCA4 | HcHCCA3 | HcHCCA2 | HcHCCA1 | HcHCCA0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 8 | HcHCCA [23:0] | Sets the base address of the RAM to which the OHCI host controller communication area is allocated. This field must be set during initialization. The OHCI host controller requests the 256-byte area from the specified base address, as the HCCA. |

**(8)  HcPeriodCurrentED register (Offset 1CH)**

After reset: 0000 0000H    R

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Period CurrentED27 | Period CurrentED26 | Period CurrentED25 | Period CurrentED24 | Period CurrentED23 | Period CurrentED22 | Period CurrentED21 | Period CurrentED20 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| Period CurrentED19 | Period CurrentED18 | Period CurrentED17 | Period CurrentED16 | Period CurrentED15 | Period CurrentED14 | Period CurrentED13 | Period CurrentED12 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Period CurrentED11 | Period CurrentED10 | Period CurrentED9 | Period CurrentED8 | Period CurrentED7 | Period CurrentED6 | Period CurrentED5 | Period CurrentED4 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Period CurrentED3 | Period CurrentED2 | Period CurrentED1 | Period CurrentED0 | 0 | 0 | 0 | 0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 4 | PeriodCurre -ntED[27:0] | Indicates the processing address included in a periodic list. The OHCI host controller updates the value of this field each time a periodic list processing sequence ends. |

**(9) HcControlHeadED register (Offset 20H)**

After reset: 0000 0000H   R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Control HeadED27 | Control HeadED26 | Control HeadED25 | Control HeadED24 | Control HeadED23 | Control HeadED22 | Control HeadED21 | Control HeadED20 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| Control HeadED19 | Control HeadED18 | Control HeadED17 | Control HeadED16 | Control HeadED15 | Control HeadED14 | Control HeadED13 | Control HeadED12 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Control HeadED11 | Control HeadED10 | Control HeadED9 | Control HeadED8 | Control HeadED7 | Control HeadED6 | Control HeadED5 | Control HeadED4 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Control HeadED3 | Control HeadED2 | Control HeadED1 | Control HeadED0 | 0 | 0 | 0 | 0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 4 | ControlHead ED[27:0] | Specifies the start address of the list ED for control transfer.<br>To perform control transfer, this field must be set (1) before setting the CLE bit of the HcControl register. |

**(10) HcControlCurrentED register (Offset 24H)**

After reset: 0000 0000H   R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Control CurrentED27 | Control CurrentED26 | Control CurrentED25 | Control CurrentED24 | Control CurrentED23 | Control CurrentED22 | Control CurrentED21 | Control CurrentED20 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| Control CurrentED19 | Control CurrentED18 | Control CurrentED17 | Control CurrentED16 | Control CurrentED15 | Control CurrentED14 | Control CurrentED13 | Control CurrentED12 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Control CurrentED11 | Control CurrentED10 | Control CurrentED9 | Control CurrentED8 | Control CurrentED7 | Control CurrentED6 | Control CurrentED5 | Control CurrentED4 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Control CurrentED3 | Control CurrentED2 | Control CurrentED1 | Control CurrentED0 | 0 | 0 | 0 | 0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 4 | ControlCurr-entED[27:0] | Indicates the processing address included in a control list.<br>The OHCI host controller updates the value of this field each time a control ED processing sequence ends.  When configuring a new list, this field is set to 00000000H, which indicates the end of the list.<br>When suspending or resuming transfer, it must be guaranteed that the ED indicated by this field exists. |

**(11) HcBulkHeadED register (Offset 28H)**

After reset: 0000 0000H　R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| BulkHead ED27 | BulkHead ED26 | BulkHead ED25 | BulkHead ED24 | BulkHead ED23 | BulkHead ED22 | BulkHead ED21 | BulkHead ED20 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| BulkHead ED19 | BulkHead ED18 | BulkHead ED17 | BulkHead ED16 | BulkHead ED15 | BulkHead ED14 | BulkHead ED13 | BulkHead ED12 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| BulkHead ED11 | BulkHead ED10 | BulkHead ED9 | BulkHead ED8 | BulkHead ED7 | BulkHead ED6 | BulkHead ED5 | BulkHead ED4 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|----|----|
| BulkHead ED3 | BulkHead ED2 | BulkHead ED1 | BulkHead ED0 | 0 | 0 | 0 | 0 |

| Bit position | Bit name | Function |
|--------------|----------|----------|
| 31 to 4 | BulkHeadED [27:0] | Specifies the start address of a list ED for bulk transfer. To perform bulk transfer, this field must be set (1) before setting the BLE bit of the HcControl register. |

**(12) HcBulkCurrentED register (Offset 2CH)**

After reset: 0000 0000H　R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| BulkCurrent ED27 | BulkCurrent ED26 | BulkCurrent ED25 | BulkCurrent ED24 | BulkCurrent ED23 | BulkCurrent ED22 | BulkCurrent ED21 | BulkCurrent ED20 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| BulkCurrent ED19 | BulkCurrent ED18 | BulkCurrent ED17 | BulkCurrent ED16 | BulkCurrent ED15 | BulkCurrent ED14 | BulkCurrent ED13 | BulkCurrent ED12 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| BulkCurrent ED11 | BulkCurrent ED10 | BulkCurrent ED9 | BulkCurrent ED8 | BulkCurrent ED7 | BulkCurrent ED6 | BulkCurrent ED5 | BulkCurrent ED4 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|----|----|
| BulkCurrent ED3 | BulkCurrent ED2 | BulkCurrent ED1 | BulkCurrent ED0 | 0 | 0 | 0 | 0 |

| Bit position | Bit name | Function |
|--------------|----------|----------|
| 31 to 4 | BulkCurrent-ED[27:0] | Indicates the address at which bulk list processing is being executed. The OHCI host controller updates the value of this field each time a bulk ED processing sequence ends. When configuring a new list, this field is set to 00000000H, which indicates the end of the list. When suspending or resuming transfer, it must be guaranteed that the ED indicated by this field exists. |

**(13) HcDoneHead register (Offset 30H)**

After reset: 0000 0000H    R

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Done Head27 | Done Head26 | Done Head25 | Done Head24 | Done Head23 | Done Head22 | Done Head21 | Done Head20 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| Done Head19 | Done Head18 | Done Head17 | Done Head16 | Done Head15 | Done Head14 | Done Head13 | Done Head12 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Done Head11 | Done Head10 | Done Head9 | Done Head8 | Done Head7 | Done Head6 | Done Head5 | Done Head4 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Done Head3 | Done Head2 | Done Head1 | Done Head0 | 0 | 0 | 0 | 0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 4 | DoneHead [27:0] | Indicates the address of HcDoneHead of the OHCI host controller. |

**(14) HcFmInterval register (Offset 34H)**

After reset: 0000 2EDFH    R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| FIT | FSMPS14 | FSMPS13 | FSMPS12 | FSMPS11 | FSMPS10 | FSMPS9 | FSMPS8 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| FSMPS7 | FSMPS6 | FSMPS5 | FSMPS4 | FSMPS3 | FSMPS2 | FSMPS1 | FSMPS0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | FI13 | FI12 | FI11 | FI10 | FI9 | FI8 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FI7 | FI6 | FI5 | FI4 | FI3 | FI2 | FI1 | FI0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 | FIT | Frame Interval Toggle<br>Used to synchronize the frame setting values of the host controller driver (HCD) and OHCI host controller.<br>Toggle the setting of this bit when writing to the FI field via the HCD.<br>The OHCI host controller reflects the FIT value to the FRT bit of the HcFmRemaining register when loading the FI field.<br>The HCD can check whether the new FI field value is reflected, by comparing the FIT bit value that was set upon writing to the FI field and the FRT bit value that was read. |
| 30 to 16 | FSMPS [14:0] | FS Largest Data Packet<br>Sets the maximum data amount that can be transmitted or received without causing a schedule overrun. The current frame position and the set value are compared, and judged up to which position of a frame can be transferred. The result varies depending on the system bus performance, so this value is set by the host controller driver (HCD). |
| 15, 14 | – | Reserved (Be sure to write "0" to these bits.) |
| 13 to 0 | FI[13:0] | Frame Interval<br>Sets the bit time for the interval of two successive SOFs in Full-Speed mode.<br>This field must be set to 2EDFH so as to satisfy the length of one frame (= 1 ms) prescribed by the USB Specification. |

**(15) HcFmRemaining register (Offset 38H)**

After reset: 0000 2EDFH    R

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| FRT | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 0 | 0 | FR13 | FR12 | FR11 | FR10 | FR9 | FR8 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| FR7 | FR6 | FR5 | FR4 | FR3 | FR2 | FR1 | FR0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 | FRT | Frame Remaining Toggle<br>Used to synchronize the frame setting values of the host controller driver (HCD) and OHCI host controller.<br>The OHCI host controller copies the FIT field value to this bit when the FR field is set to "0H" and the FI field value is reloaded.<br>The HCD can check whether the value set to the FI field was set to the FR field by comparing the FIT bit and FRT bit values. |
| 13 to 0 | FR[13:0] | Frame Remaining<br>Indicates the current frame value.<br>The value of this field is counted down as time elapses.<br>Since the frame value is reloaded when the value of FI field becomes "0H", the FR field value is copied to this field and counting down starts again. |

**(16) HcFmNumber register (Offset 3CH)**

After reset: 0000 0000H    R

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| Frame Number15 | Frame Number14 | Frame Number13 | Frame Number12 | Frame Number11 | Frame Number10 | Frame Number9 | Frame Number8 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| Frame Number7 | Frame Number6 | Frame Number5 | Frame Number4 | Frame Number3 | Frame Number2 | Frame Number1 | Frame Number0 |

| Bit position | Bit name | Function |
|---|---|---|
| 15 to 0 | FrameNumber [15:0] | Indicates the number of frames that have been transferred.<br>This field is counted up when the FR field value becomes "0H". |

**(17) HcPeriodicStart register (Offset 40H)**

After reset: 0000 0000H    R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | Periodic Start13 | Periodic Start12 | Periodic Start11 | Periodic Start10 | Periodic Start9 | Periodic Start8 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|----|----|
| Periodic Start7 | Periodic Start6 | Periodic Start5 | Periodic Start4 | Periodic Start3 | Periodic Start2 | Periodic Start1 | Periodic Start0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 14 | − | Reserved.  (Be sure to write "0" to these bits.) |
| 13 to 0 | PeriodicStart [13:0] | Used to determine the transfer ratio of periodic and async lists in a PeriodicStart field frame.<br>This field must be set by the host controller driver (HCD) during initialization of the OHCI host controller.<br>If the value of the FR field is larger than the value set to this field, the non-periodic list takes precedence over the periodic list.<br>The OHCI Specification recommends setting this value to about 10% of the FI field value, and the relevant value is 3E67H. |

**(18) HcLSThreshold register (Offset 44H)**

After reset: 0000 0628H    R/W

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | HcLSThres-hold11 | HcLSThres-hold10 | HcLSThres-hold9 | HcLSThres-hold8 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|----|----|
| HcLSThres-hold7 | HcLSThres-hold6 | HcLSThres-hold5 | HcLSThres-hold4 | HcLSThres-hold3 | HcLSThres-hold2 | HcLSThres-hold1 | HcLSThres-hold0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 12 | − | Reserved.  (Be sure to write "0" to these bits.) |
| 11 to 0 | HcLSThres-hold[11:0] | Specifies the threshold value for whether transfer is available for the remaining time of frames during LS transfer.<br>If the FR field value is larger than the value set to this field, LS transfer can be started. |

**(19) HRDA (HcRhDescriptorA) register (Offset 48H)**

After reset: FF00 0902H    R/W

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| HRDA | POTPGT7 | POTPGT6 | POTPGT5 | POTPGT4 | POTPGT3 | POTPGT2 | POTPGT1 | POTPGT0 |

| | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | NOCP | OCPM | DT | NPS | PSM |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NDP7 | NDP6 | NDP5 | NDP4 | NDP3 | NDP2 | NDP1 | NDP0 |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 24 | POTPGT [7:0] | Power On To Power Good Time<br>Defines the wait time until the host controller driver (HCD) accesses a root hub powered on (unit: 2 ms). The wait time is calculated by POTPGT field value × 2 ms. |
| 23 to 13 | – | Reserved. (Be sure to write "0" to these bits.) |
| 12 | NOCP | No Overcurrent Protection<br>Defines whether to support the root hub overcurrent protect function.<br>    1: Does not support the overcurrent protect state.<br>    0: Supports the overcurrent protect state. |
| 11 | OCPM | Overcurrent Protection Mode<br>Defines how to report the overcurrent state of the root hub.<br>The mode set with this bit must be the same as the mode set with the PSM bit.<br>    1: Reports the overcurrent state in port units.<br>    0: Reports the overcurrent state of all ports simultaneously.<br>The setting of this bit becomes valid only when the NOCP bit is cleared to "0". |
| 10 | DT | Device Type<br>Indicates that the root hub is not a combined device.<br>This field always returns "0" because the root hub must not be a combined device.<br>This bit is read-only. |
| 9 | NPS | No Power Switching<br>Defines whether power switching is supported or the port is constantly powered on.<br>    1: Power is constantly on while the OHCI host controller is running.<br>    0: Power for ports can be switched. |

| Bit position | Bit name | Function |
|---|---|---|
| 8 | PSM | Power Switching Mode<br>Defines how to control root hub port power switching.<br>  1: Power for ports is controlled individually.<br>  0: Power for all ports is controlled simultaneously.<br>Ports are only controlled by the SPP and CPP bits of the HcRhPortStatus register if the PPCM field of the HcRhDescriptorB register is set (1). If it is cleared (0), ports are controlled by the SGP and CGP bits. The setting of this bit becomes valid only when the NPS bit is cleared to "0". |
| 7 to 0 | NDP[7:0] | Number Downstream Port<br>Defines the number of downstream ports supported by the root hub of the OHCI host controller.<br>This field is fixed to 02H because the OHCI host controller incorporates two downstream ports.<br>These bits are read-only. |

**(20) HcRhDescriptorB register (Offset 4CH)**

After reset: 0006 0000H    R/W

| 31 | | 16 |
|---|---|---|
| | PPCM [15:0] | |

| 15 | | 0 |
|---|---|---|
| | DR [15:0] | |

| Bit position | Bit name | Function |
|---|---|---|
| 31 to 16 | PPCM [15:0] | Port Power Control Mask<br>Indicates whether ports are controlled by the Set/ClearGlobalPower.<br>The setting of this bit becomes valid only when the PSM bit is set (1).<br>• Field<br><table><tr><td>Bit</td><td>Description</td></tr><tr><td>0</td><td>Reserved</td></tr><tr><td>1</td><td>Setting of device connected to port 1.</td></tr><tr><td>2</td><td>Setting of device connected to port 2.</td></tr><tr><td>15-3</td><td>Reserved</td></tr></table>• Value<br>1: Ports are only controlled by the Set/ClearPortPower.<br>0: Ports are controlled by the Set/ClearGlobalPower. |
| 15 to 0 | DR [15:0] | Device Removable<br>Indicates whether the device connected to the USB host controller port is removable.<br>• Field<br><table><tr><td>Bit</td><td>Description</td></tr><tr><td>0</td><td>Reserved</td></tr><tr><td>1</td><td>Setting of device connected to port 1.</td></tr><tr><td>2</td><td>Setting of device connected to port 2.</td></tr><tr><td>15-3</td><td>Reserved</td></tr></table>• Value<br>1: The connected device is not removable.<br>0: The connected device is removable. |

**(21) HcRhStatus register (Offset 50H)**

After reset: 0000 0000H R/W

**(a) When read**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| CRWE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | OCIC | LPSC |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| DRWE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | OCI | LPS |

**(b) When written**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| CRWE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | OCIC | SGP |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SRWE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | OCI | CGP |

| Bit position | R/W | Bit name | Function |
|---|---|---|---|
| 31 | – | CRWE | Clear Remote Wakeup Enable<br>Clears the DRWE bit to "0".<br>Writing "1" to this bit clears the DRWE bit to 0. Writing "0" to this bit is ignored.<br>Value "0" is always read from this bit.<br>This bit is write-only. |
| 30 to 18 | – | – | Reserved. (Be sure to write "0" to these bits.) |
| 17 | – | OCIC | Overcurrent Indicate Change<br>Reports the change in the OCI bit. This bit is set (1) when the OCI bit setting changes. This bit is cleared (0) when "1" is written while being set (1).<br>　1: The overcurrent state has changed.<br>　0: The overcurrent state has not changed. |
| 16 | R | LPSC | Local Power Status Change<br>This bit is fixed to "0" because the local power status is not supported. |
| | W | SGP | Set Global Power<br>Used to turn on power to all ports in global power mode. Setting this bit to "1" turns power on to all ports.<br>When the PSM bit of the HcRhDescriptorA register is set to "1", power is turned on to ports with which the PPCM field is cleared (0). |

| Bit position | R/W | Bit name | Function |
|---|---|---|---|
| 15 | R | DRWE | Device Remote Wakeup Enable<br>Sets whether the connect status change is included in remote wakeup events.<br>　1: The connect status change is a remote wakeup source.<br>　0: The connect status change is not a remote wakeup source.<br>If a connect status change event occurs while this bit is set (1), the state changes from USB Suspend to USB Resume, and a Resume Detected interrupt occurs. |
|  | W | SRWE | Set Remote Wakeup Enable<br>Sets the DRWE bit to 1.<br>Writing "1" to this bit sets the DRWE bit to "1".  Writing "0" to this bit is ignored. |
| 14 to 2 | – | – | Reserved.  (Be sure to write "0" to these bits.) |
| 1 | – | OCI | Overcurrent Indicator<br>Reports the overcurrent state in global overcurrent detection mode.<br>　1: Ports are in the overcurrent state.<br>　0: Ports are in the normal state.<br>This bit is fixed to "0" when overcurrent is reported in port units.<br>This bit is read-only. |
| 0 | R | LPS | Local Power Status<br>This bit is fixed to "0" because the local power status is not supported. |
|  | W | CGP | Clear Global Power<br>Used to turn off power for all ports in global power mode.<br>Setting this bit to "1" turns off power to all ports.<br>When the PSM bit of the HcRhDescriptorA register is set to "1", power is turned off to ports with which the PPCM field is cleared (0). |

**(22) HcRhPortStatus1/2 register (Offset 54H/58H)**

After reset: 0000 0000H    R/W

**(a) When read**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0  | 0  | 0  | PRSC | OCIC | PSSC | PESC | CSC |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|---|---|
| 0  | 0  | 0  | 0  | 0  | 0  | LSDA | PPS |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | PRS | POCI | PSS | PES | CSC |

**(b) When written**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----|----|----|----|----|----|----|----|
| 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----|----|----|----|----|----|----|----|
| 0  | 0  | 0  | PRSC | OCIC | PSSC | PESC | CSC |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----|----|----|----|----|----|---|---|
| 0  | 0  | 0  | 0  | 0  | 0  | CPP | SPP |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | SPR | CSS | SPS | SPE | CPE |

| Bit position | R/W | Bit name | Function |
|---|---|---|---|
| 30 to 21 | – | – | Reserved.  (Be sure to write "0" to these bits.) |
| 20 | – | PRSC | Port Reset Status Change<br>Indicates the completion of port reset.<br>  1: Port reset has completed.<br>  0: The port reset status has not changed.<br>This bit is set (1) when hardware reset of 10 ms is completed.<br>This bit is cleared (0) when "1" is written via the host controller driver (HCD). |
| 19 | – | OCIC | Overcurrent Indicate Change<br>This bit is set (1) when the overcurrent status is detected at a port.<br>  1: The overcurrent state has changed.<br>  0: The overcurrent state has not changed.<br>This bit is cleared (0) when "1" is written via the host controller driver (HCD). |
| 18 | – | PSSC | Port Suspend Status Change<br>Indicates that the RESUME sequence has completed.<br>  1: RESUME has completed.<br>  0: Port Suspend Status has not changed.<br>This bit is set when all of RESUME processing is completed by hardware.<br>This bit is cleared to 0 when "1" is written via the host controller driver (HCD). |

(2/4)

| Bit position | R/W | Bit name | Function |
|---|---|---|---|
| 17 | − | PESC | Port Enable Status Change<br>Indicates that the PES bit has been cleared (0).<br>　1: The PES bit status has changed. (PES cleared)<br>　0: The PES bit status has not changed.<br>This bit is set (1) along with the PES bit being cleared (0) because the port state is changed from Enable to Disable due to a hardware event such as an overcurrent state, device disconnection, power-off or bubble error.<br>This bit is cleared (0) when "1" is written via the host controller driver (HCD). |
| 16 | − | CSC | Connect Status Change<br>Indicates that the CCS bit has been changed.<br>　1: The CCS bit status has changed.<br>　0: The CCS bit status has not changed.<br>This bit is set (1) when the CCS bit status is changed due to connection or disconnection of a USB device.<br>This bit is set (1) when a request for port reset, port suspend or port enable is issued while a USB device is disconnected, so that the driver can re-evaluate the device connection.<br>This bit is cleared to 0 when "1" is written via the host controller driver (HCD). |
| 15 to 10 | − | − | Reserved. (Be sure to write "0" to these bits.) |
| 9 | R | LSDA | Low Speed Device Attached<br>Indicates the speed of the device connected to the USB port.<br>　1: A low-speed device is connected.<br>　0: A full-speed device is connected.<br>The bit is enabled only when the CCS bit is set. |
| | W | CPP | Clear Port Power<br>Turns port power off.<br>Writing "1" to this bit turns the port power off. Writing "0" to this bit is ignored. |
| 8 | R | PPS | Port Power Status<br>Indicates the port power status.<br>　1: Port power on<br>　0: Port power off<br>The control method varies depending on the power switch time. |
| | W | SPP | Set Port Power<br>Turns port power on when power control is performed in port units.<br>Writing "1" to this bit turns the port power on. Writing "0" to this bit is ignored. |
| 7-5 | − | − | Reserved. (Be sure to write "0" to these bits.) |

(3/4)

| Bit position | R/W | Bit name | Function |
|---|---|---|---|
| 4 | R | PRS | Port Reset Status<br>Indicates that reset is being issued for a downstream port.<br>  1: The port is being reset.<br>  0: The port is not being reset.<br>This bit is cleared (0) together with setting of the PRSC bit when port reset of 10 ms ends.<br>This bit cannot be set when the CSC bit has been cleared (0) (no device connected). |
| | W | SPR | Set Port Reset<br>Issues port reset for a downstream port.<br>Writing "1" to this bit starts port reset of 10 ms.  When "1" is written to this bit while the CCS bit is cleared (0), the CSC bit is set and it is reported to the host controller driver that an attempt was made to reset the disconnected port.  Writing "0" to this bit is ignored. |
| 3 | R | POCI | Port Overcurrent Indicator<br>Indicates the overcurrent status at a downstream port.<br>  1: The port is in the overcurrent state.<br>  0: The port is in the normal state. |
| | W | CPS | Clear Port Suspend<br>Ends Suspend mode and starts the resume sequence.<br>Writing "1" to this bit starts the resume sequence.  Writing "0" to this bit is ignored.<br>Resume starts only when the PSS bit has been set. |
| 2 | R | PSS | Port Suspend Status<br>Indicates that ports are in the Suspend state or the resume sequence is being executed.<br>  1: Ports are in the Suspend state.<br>  0: Ports are in the normal transfer state.<br>This bit cannot be set when the CCS bit has been cleared to 0 (no device connected).<br>Writing to the SPS bit sets ports to the Suspend state.  This bit is cleared (0) when resume ends, port reset ends, or when the state shifts to USB RESUME. |
| | W | SPS | Set Port Suspend<br>Shifts the port status to the Suspend state.<br>When "1" is written to this bit, ports enter the Suspend state.  Writing "0" to this bit is ignored.<br>If this bit is written when the CCS bit has been cleared to 0, the CSC bit is set and it is reported to the host controller driver that an attempt was made to suspend the disconnected port. |

(4/4)

| Bit position | R/W | Bit name | Function |
|---|---|---|---|
| 1 | R | PES | Port Enable Status<br>Indicates whether ports are enabled or disabled.<br>  1: Ports are enabled.<br>  0: Ports are disabled.<br>This bit cannot be set when the CSC bit has been cleared to 0 (no device connected).<br>Ports enter the Enable state when port reset is completed.<br>This bit is automatically cleared (0) by hardware when an overcurrent state, disconnect, power-off, or bubble error is detected. |
|   | W | SPE | Set Port Enable<br>Sets the PES bit to "1".<br>Writing "0" to this bit is ignored.<br>Change the port status by setting the SPR bit.  The OHCI host controller of the V850ES/JG3-U and V850ES/JH3-U does not support enabling of ports by using this bit because the USB Specification does not, although the OHCI Specification does. |
| 0 | R | CCS | Current Connect Status<br>Indicates the current connection status at a downstream port.<br>  1: A device is connected.<br>  0: No device is connected. |
|   | W | CPE | Clear Port Enable<br>Clears the CPE bit.<br>When "1" is written to this bit, ports enter the Disable state.  Writing "0" to this bit is ignored. |

### 21.7.4  Interruption from USB host controller

The USB host controller collects and sorts interrupts sent from the OHCI host controller and reports them to the system as one of the following three types of interrupts.

**Table 21-6.  Interruption from USB Host Controller**

| Interrupt Report Signal to System | Interrupt Report Signal Generated by OHCI Host Controller |
|---|---|
| INTUSBH0 | USBH status interrupt (INTA, SMMI or PME) |
| INTUSBH1 | USBH PCI cycle error |
| INTUSBH2 | USBH PME interrupt |

The details on each interrupt are as follows.

### (1)  USB status interrupt (INTUSBH0)

#### (a)  Interrupt routing

The OHCI host controller reports an occurrence of the INTA or SMMI interrupt to the system, according to the setting of the IR bit of the HcControl register in OHCI operational registers.  After reset, INTA is routed for reporting the interrupt by the initial setting of the IR bit.

INTA and SMMI are used as the sources of interruption, except for Ownership Change.

**Table 21-7.  Routing of Interrupts INTA and SMMI**

| IR Bit of HcControl Register | Interrupt Report Signal |
|---|---|
| 0 | INTA (initial value) |
| 1 | SMMI |

To use the INTA and SMMI interrupt, the inta_en and int_smmi_en bits of the PCI interrupt control register of PCI host bridge registers must be set (1).

### (2)  Interrupt sources

Interrupts defined in the OpenHCI Specification are supported.

Use the HcInterruptEnable register to set interrupt sources reported to the system.  Interrupts are reported via the route specified with the IR bit.

The following lists the interrupt sources.

**Table 21-8. Sources of Interrupt INTA and SMMI**

| Interrupt Source | Description |
|---|---|
| Scheduling Overrun | Indicates the occurrence of USB scheduling overrun in a frame. |
| Writeback DoneHead | Indicates the occurrence of writeback when the USB host controller ends TD. |
| Start Of Frame | Indicates the update of HccaFmNumber upon a start of a frame. |
| Resume Detected | Indicates the detection of resume signal sent from a USB device. |
| Unrecoverable Error | Indicates the detection of error not related to USB (PCI abort). |
| Frame Number Overflow | Indicates the change of FrameNumber15 bit of the HcFmNumber register (0 → 1 or 1 → 0) |
| Root Hub Status Change | Indicates the change of the HcRhStatus/HcRhPortStatus status. <br> This interrupt source is classified based on the following event details. <br><br> <table><tr><td>OverCurrent Indicate Change</td><td>Indicates the occurrence of overcurrent state.</td></tr><tr><td>Connect Status Change</td><td>Indicates the device connection/disconnection to the USB bus.</td></tr><tr><td>Port Enable Status Change</td><td>Indicates that the port is disabled due to a USB error.</td></tr><tr><td>Port Suspend Status Change</td><td>Indicates the completion of resume sequence</td></tr><tr><td>Port Reset Status Change</td><td>Indicates the completion of USB reset</td></tr></table> |
| Ownership Change | Indicates the occurrence of Ownership Request. <br> This interrupt is reported to SMMI only. |

**(2) PCI cycle error interrupt (INTUSBH1)**

The PCI cycle error interrupt (INTUSBH1) occurs when a data access that causes an error is performed for the OHCI or PCI bus bridge register area (in 32-bit access). This interrupt is used for debugging, and is not used during normal operation.

**(3) PME interrupt (INTUSBH2)**

The PME interrupt is an interrupt signal for power management, which can be used to report the USB bus status change to the system when PCLK is not supplied.

The following lists the events that may occur on the USB bus and support for interrupts.

**Table 21-9. PME Interrupt Sources**

| Event | Occurrence of Interrupts |
|---|---|
| Overcurrent Indicate | Does not occur. |
| Connect | Occurs. |
| Disconnect | Occurs. |
| Resume (RemoteWakeUp) | Occurs. |

To use the PME interrupt, the PME Enable bit of the Power Management Control/Status register in OHCI host configuration registers and the int_pme_en bit of the PCI interrupt control register in PCI host bridge registers must be set to "1" in advance.

## CHAPTER 22 DMA FUNCTION (DMA CONTROLLER)

The V850ES/JG3-U and V850ES/JH3-U include a direct memory access (DMA) controller (DMAC) that executes and controls DMA transfer.

The DMAC controls data transfer between memory and I/O, between memories, or between I/Os based on DMA requests issued by the on-chip peripheral I/O (serial interface, timer/counter, and A/D converter), interrupts from external input pins, or software triggers (memory refers to internal RAM or external memory).

## 22.1 Features

- 4 independent DMA channels
- Transfer unit: 8/16 bits
- Maximum transfer count: 65,536 ($2^{16}$)
- Transfer type: Two-cycle transfer
- Transfer mode: Single transfer mode
- Transfer requests
  - Request by interrupts from on-chip peripheral I/O (serial interface, timer/counter, A/D converter) or interrupts from external input pin
  - Requests by software trigger
- Transfer targets
  - Internal RAM $\leftrightarrow$ Peripheral I/O
  - Peripheral I/O $\leftrightarrow$ Peripheral I/O
  - Internal RAM $\leftrightarrow$ External memory
  - External memory $\leftrightarrow$ Peripheral I/O
  - External memory $\leftrightarrow$ External memory

## 22.2 Configuration



**Remark**    n = 0 to 3
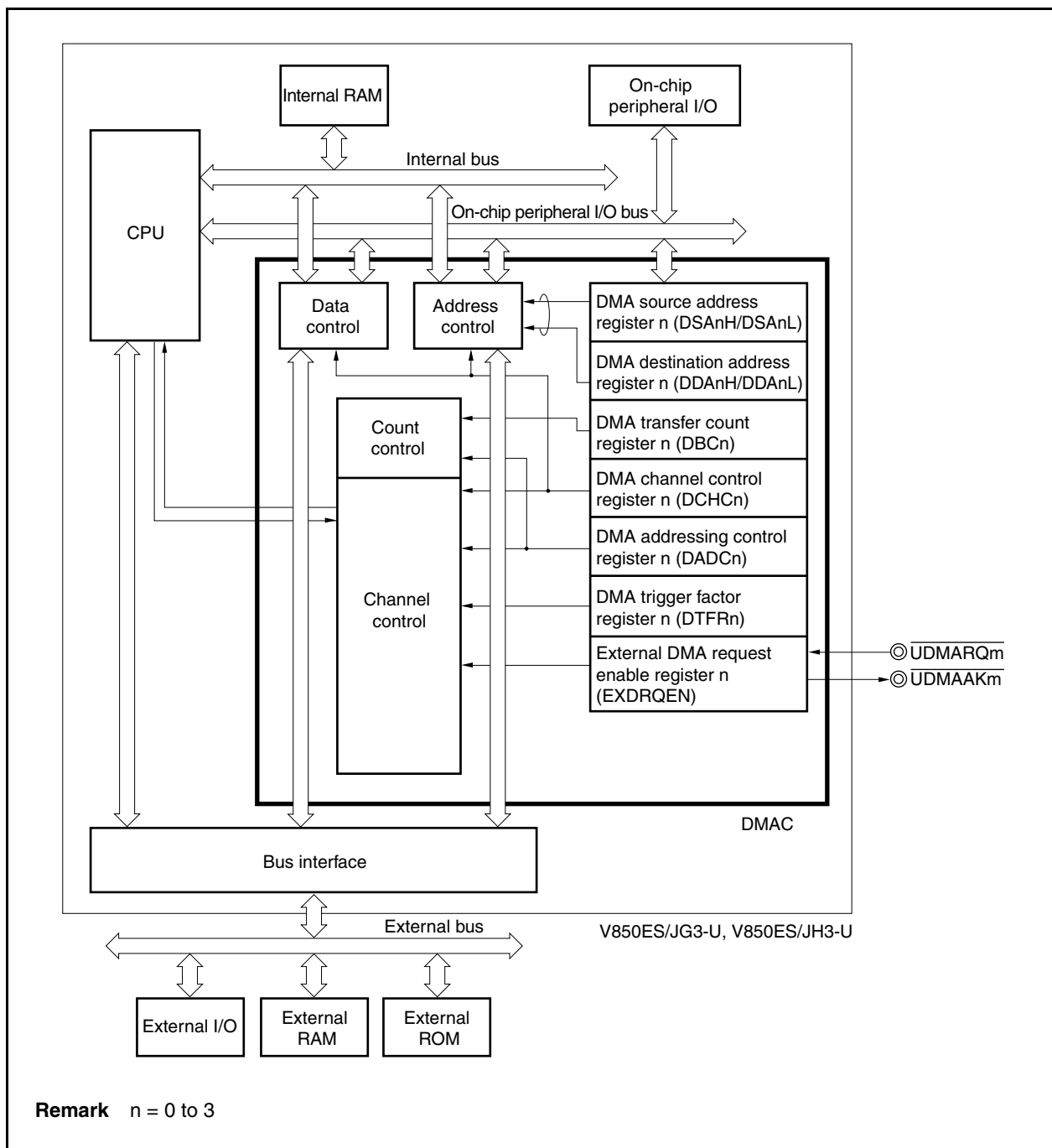
## 22.3  Registers

**(1)  DMA source address registers 0 to 3 (DSA0 to DSA3)**

The DSA0 to DSA3 registers set the DMA source addresses (26 bits each) for DMA channel n (n = 0 to 3).

These registers are divided into two 16-bit registers, DSAnH and DSAnL.

These registers can be read or written in 16-bit units.

After reset: Undefined     R/W     Address:   DSA0H FFFFF082H, DSA1H FFFFF08AH,

DSA2H FFFFF092H, DSA3H FFFFF09AH,

DSA0L FFFFF080H, DSA1L FFFFF088H,

DSA2L FFFFF090H, DSA3L FFFFF098H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DSAnH (n = 0 to 3) | IR | 0 | 0 | 0 | 0 | 0 | SA25 | SA24 | SA23 | SA22 | SA21 | SA20 | SA19 | SA18 | SA17 | SA16 |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DSAnL (n = 0 to 3) | SA15 | SA14 | SA13 | SA12 | SA11 | SA10 | SA9 | SA8 | SA7 | SA6 | SA5 | SA4 | SA3 | SA2 | SA1 | SA0 |

| IR | Specification of DMA transfer source |
|---|---|
| 0 | External memory or on-chip peripheral I/O |
| 1 | Internal RAM |

| SA25 to SA16 | Set the address (A25 to A16) of the DMA transfer source (default value is undefined). During DMA transfer, the next DMA transfer source address is held. When DMA transfer is completed, the DMA address set first is held. |
|---|---|

| SA15 to SA0 | Set the address (A15 to A0) of the DMA transfer source (default value is undefined). During DMA transfer, the next DMA transfer source address is held. When DMA transfer is completed, the DMA address set first is held. |
|---|---|

**Cautions 1.  Be sure to clear bits 14 to 10 of the DSAnH register to 0.**

**2.  Set the DSAnH and DSAnL registers at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).**

- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

**3.  When the value of the DSAn register is read, two 16-bit registers, DSAnH and DSAnL, are read.  If reading and updating conflict, the value being updated may be read (see 22.13 Cautions).**

**4.  Following reset, set the DSAnH, DSAnL, DDAnH, DDAnL, and DBCn registers before starting DMA transfer.  If these registers are not set, the operation when DMA transfer is started is not guaranteed.**

**(2) DMA destination address registers 0 to 3 (DDA0 to DDA3)**

The DDA0 to DDA3 registers set the DMA destination address (26 bits each) for DMA channel n (n = 0 to 3). These registers are divided into two 16-bit registers, DDAnH and DDAnL.

These registers can be read or written in 16-bit units.

---

After reset: Undefined     R/W     Address:    DDA0H FFFFF086H, DDA1H FFFFF08EH,

DA2H FFFFF096H, DDA3H FFFFF09EH,

DDA0L FFFFF084H, DDA1L FFFFF08CH,

DDA2L FFFFF094H, DDA3L FFFFF09CH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DDAnH (n = 0 to 3) | IR | 0 | 0 | 0 | 0 | 0 | DA25 | DA24 | DA23 | DA22 | DA21 | DA20 | DA19 | DA18 | DA17 | DA16 |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DDAnL (n = 0 to 3) | DA15 | DA14 | DA13 | DA12 | DA11 | DA10 | DA9 | DA8 | DA7 | DA6 | DA5 | DA4 | DA3 | DA2 | DA1 | DA0 |

| IR | Specification of DMA transfer destination |
|---|---|
| 0 | External memory or on-chip peripheral I/O |
| 1 | Internal RAM |

| DA25 to DA16 | Set an address (A25 to A16) of DMA transfer destination (default value is undefined). During DMA transfer, the next DMA transfer destination address is held. When DMA transfer is completed, the DMA transfer source address set first is held. |
|---|---|

| DA15 to DA0 | Set an address (A15 to A0) of DMA transfer destination (default value is undefined). During DMA transfer, the next DMA transfer destination address is held. When DMA transfer is completed, the DMA transfer source address set first is held. |
|---|---|

**Cautions 1. Be sure to clear bits 14 to 10 of the DDAnH register to 0.**

**2. Set the DDAnH and DDAnL registers at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).**
- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

**3. When the value of the DDAn register is read, two 16-bit registers, DDAnH and DDAnL, are read. If reading and updating conflict, a value being updated may be read (see 22.13 Cautions).**

**4. Following reset, set the DSAnH, DSAnL, DDAnH, DDAnL, and DBCn registers before starting DMA transfer. If these registers are not set, the operation when DMA transfer is started is not guaranteed.**

---

**(3) DMA transfer count registers 0 to 3 (DBC0 to DBC3)**

The DBC0 to DBC3 registers are 16-bit registers that set the transfer count for DMA channel n (n = 0 to 3). These registers hold the remaining transfer count during DMA transfer.

These registers are decremented by 1 per transfer regardless of the transfer data unit (8/16 bits), and the transfer is terminated if a borrow occurs.

These registers can be read or written in 16-bit units.

After reset: Undefined     R/W     Address: DBC0 FFFFF0C0H, DBC1 FFFFF0C2H,
DBC2 FFFFF0C4H, DBC3 FFFFF0C6H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DBCn (n = 0 to 3) | BC15 | BC14 | BC13 | BC12 | BC11 | BC10 | BC9 | BC8 | BC7 | BC6 | BC5 | BC4 | BC3 | BC2 | BC1 | BC0 |

| BC15 to BC0 | Transfer count setting or remaining transfer count during DMA transfer |
|---|---|
| 0000H | Transfer count of 1st transfer or remaining transfer count |
| 0001H | Transfer count of 2nd transfer or remaining transfer count |
| : | : |
| FFFFH | Transfer count of 65,536 ($2^{16}$)th transfer or remaining transfer count |
| | The number of transfer data set first is held when DMA transfer is complete. |

**Cautions 1. Set the DBCn register at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).**
- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

**2. Following reset, set the DSAnH, DSAnL, DDAnH, DDAnL, and DBCn registers before starting DMA transfer. If these registers are not set, the operation when DMA transfer is started is not guaranteed.**

**(4) DMA addressing control registers 0 to 3 (DADC0 to DADC3)**

The DADC0 to DADC3 registers are 16-bit registers that control the DMA transfer mode for DMA channel n (n = 0 to 3).

These registers can be read or written in 16-bit units.

Reset sets these registers to 0000H.

After reset: 0000H　　R/W　　Address: DADC0 FFFFF0D0H, DADC1 FFFFF0D2H,
　　　　　　　　　　　　　　　　　　　DADC2 FFFFF0D4H, DADC3 FFFFF0D6H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| DADCn | 0 | DS0 | 0 | 0 | 0 | 0 | 0 | 0 |

(n = 0 to 3)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SAD1 | SAD0 | DAD1 | DAD0 | 0 | 0 | 0 | 0 |

| DS0 | Setting of transfer data size |
|---|---|
| 0 | 8 bits |
| 1 | 16 bits |

| SAD1 | SAD0 | Setting of count direction of the transfer source address |
|---|---|---|
| 0 | 0 | Increment |
| 0 | 1 | Decrement |
| 1 | 0 | Fixed |
| 1 | 1 | Setting prohibited |

| DAD1 | DAD0 | Setting of count direction of the destination address |
|---|---|---|
| 0 | 0 | Increment |
| 0 | 1 | Decrement |
| 1 | 0 | Fixed |
| 1 | 1 | Setting prohibited |

**Cautions 1. Be sure to clear bits 15, 13 to 8, and 3 to 0 of the DADCn register to 0.**

**2. Set the DADCn register at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).**
- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

**3. The DS0 bit specifies the size of the transfer data, and does not control bus sizing. If 8-bit data (DS0 bit = 0) is set, therefore, the lower data bus is not always used.**

**4. If the transfer data size is set to 16 bits (DS0 bit = 1), transfer cannot be started from an odd address. Transfer is always started from an address with the first bit of the lower address aligned to 0.**

**5. If DMA transfer is executed on an on-chip peripheral I/O register (as the transfer source or destination), be sure to specify the same transfer size as the register size. For example, to execute DMA transfer on an 8-bit register, be sure to specify 8-bit transfer.**

**(5) DMA channel control registers 0 to 3 (DCHC0 to DCHC3)**

The DCHC0 to DCHC3 registers are 8-bit registers that control the DMA transfer operating mode for DMA channel n.

These registers can be read or written in 8-bit or 1-bit units. (However, bit 7 is read-only and bits 1 and 2 are write-only. If bit 1 or 2 is read, the read value is always 0.)

Reset sets these registers to 00H.

After reset: 00H    R/W    Address: DCHC0 FFFFF0E0H, DCHC1 FFFFF0E2H,
DCHC2 FFFFF0E4H, DCHC3 FFFFF0E6H

| | <7> | 6 | 5 | 4 | 3 | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| DCHCn | TCn[Note 1] | 0 | 0 | 0 | 0 | INITn[Note 2] | STGn[Note 2] | Enn |

(n = 0 to 3)

| TCn[Note 1] | Status flag indicates whether DMA transfer through DMA channel n has completed or not |
|---|---|
| 0 | DMA transfer had not completed. |
| 1 | DMA transfer had completed. |
| It is set to 1 on the last DMA transfer and cleared to 0 when it is read. | |

| INITn[Note 2] | If the INITn bit is set to 1 with DMA transfer disabled (Enn bit = 0), the DMA transfer status can be initialized. When re-setting the DMA transfer status (re-setting the DDAnH, DDAnL, DSAnH, DSAnL, DBCn, and DADCn registers) before DMA transfer is completed (before the TCn bit is set to 1), be sure to initialize the DMA channel. When initializing the DMA controller, however, be sure to observe the procedure described in **22.13 Cautions**. |
|---|---|

| STGn[Note 2] | This is a software startup trigger of DMA transfer. If this bit is set to 1 in the DMA transfer enable state (TCn bit = 0, Enn bit = 1), DMA transfer is started. |
|---|---|

| Enn | Setting of whether DMA transfer through DMA channel n is to be enabled or disabled |
|---|---|
| 0 | DMA transfer disabled |
| 1 | DMA transfer enabled |
| DMA transfer is enabled when the Enn bit is set to 1. When DMA transfer is completed (when a terminal count is generated), this bit is automatically cleared to 0. To abort DMA transfer, clear the Enn bit to 0 by software. To resume, set the Enn bit to 1 again. When aborting or resuming DMA transfer, however, be sure to observe the procedure described in **22.13 Cautions**. | |

**Notes 1.** The TCn bit is read-only.

**2.** The INITn and STGn bits are write-only.

**Cautions 1. Be sure to clear bits 6 to 3 of the DCHCn register to 0.**

**2. When DMA transfer is completed (when a terminal count is generated), the Enn bit is cleared to 0 and then the TCn bit is set to 1. If the DCHCn register is read while its bits are being updated, a value indicating "transfer not completed and transfer is disabled" (TCn bit = 0 and Enn bit = 0) may be read.**

**(6) DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)**

The DTFR0 to DTFR3 registers are 8-bit registers that control the DMA transfer start trigger via interrupt request signals from on-chip peripheral I/O.

The interrupt request signals set by these registers serve as DMA transfer start factors.

These registers can be read or written in 8-bit units.  However, DFn bit can be read or written in 1-bit units.

Reset sets these registers to 00H.

After reset: 00H     R/W     Address: DTFR0 FFFFF810H,  DTFR1 FFFFF812H,
                                     DTFR2 FFFFF814H,  DTFR3 FFFFF816H

| | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DTFRn | DFn | 0 | IFCn5 | IFCn4 | IFCn3 | IFCn2 | IFCn1 | IFCn0 |

(n = 0 to 3)

| DFn[Note] | DMA transfer request status flag |
|---|---|
| 0 | No DMA transfer request |
| 1 | DMA transfer request |

**Note**  Do not set the DFn bit to 1 by software.  Write 0 to this bit to clear a DMA transfer request if an interrupt that is specified as the DMA transfer start factor occurs while DMA transfer is disabled.

**Cautions 1.  Set the IFCn5 to IFCn0 bits at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).**
- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

**2.  An interrupt request that is generated in the standby mode (IDEL1, IDLE2, STOP, or sub-IDLE mode) does not start the DMA transfer cycle (nor is the DFn bit set to 1).**

**3.  If a DMA start factor is selected by the IFCn5 to IFCn0 bits, the DFn bit is set to 1 when an interrupt occurs from the selected on-chip peripheral I/O, regardless of whether the DMA transfer is enabled or disabled.  If DMA is enabled in this status, DMA transfer is immediately started.**

**4.  Be sure to follow the steps below when changing the DTFRn register settings.**
- **When the values to be set to bits IFCn5 to IFCn0 are not set to bits IFCm5 to IFCm0 of another channel (n = 0 to 3, m = 0 to 3, n ≠ m)**
  - **<1>  Stop the DMAn operation of the channel to be rewritten (DCHCn.Enn bit = 0).**
  - **<2>  Change the DTFRn register settings. (Be sure to set DFn bit = 0 and change the settings in the 8-bit manipulation.)**
  - **<3>  Confirm that DFn bit = 0. (Stop the interrupt generation source operation beforehand.)**
  - **<4>  Enable the DMAn operation (Enn bit = 1).**
- **When the values to be set to bits IFCn5 to IFCn0 are set to bits IFCm5 to IFCm0 of another channel (n = 0 to 3, m = 0 to 3, n ≠ m)**
  - **<1>  Stop the DMAn operation of the channel to be rewritten (DCHCn.Enn bit = 0).**
  - **<2>  Stop the DMAm operation of the channel where the same values are set to bits IFCm5 to IFCm0 as the values to be used to rewrite bits IFCn5 to IFCn0 (DCHCm.Emm bit = 0).**
  - **<3>  Change the DTFRn register settings. (Be sure to set DFn bit = 0 and change the settings in the 8-bit manipulation.)**
  - **<4>  Confirm that bits DFn and DFm = 0. (Stop the interrupt generation source operation beforehand.)**
  - **<5>  Enable the DMAn operation (bits Enn and Emm = 1).**

**Remark**  For the IFCn5 to IFCn0 bits, see **Table 22-1  DMA Start Factors**.

**Table 22-1.  DMA Start Factors (1/2)**

| IFCn5 | IFCn4 | IFCn3 | IFCn2 | IFCn1 | IFCn0 | Interrupt Source |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | DMA request by interrupt disabled |
| 0 | 0 | 0 | 0 | 0 | 1 | INTP02 |
| 0 | 0 | 0 | 0 | 1 | 0 | INTP05 |
| 0 | 0 | 0 | 0 | 1 | 1 | INTP09 |
| 0 | 0 | 0 | 1 | 0 | 0 | INTP10 |
| 0 | 0 | 0 | 1 | 0 | 1 | INTP13 |
| 0 | 0 | 0 | 1 | 1 | 0 | INTP16 |
| 0 | 0 | 0 | 1 | 1 | 1 | INTTAB0OV |
| 0 | 0 | 1 | 0 | 0 | 0 | INTTAB0CC0 |
| 0 | 0 | 1 | 0 | 0 | 1 | INTTAB0CC1 |
| 0 | 0 | 1 | 0 | 1 | 0 | INTTAB0CC2 |
| 0 | 0 | 1 | 0 | 1 | 1 | INTTAB0CC3 |
| 0 | 0 | 1 | 1 | 0 | 0 | INTTAB1OV_BASE[Note] |
| 0 | 0 | 1 | 1 | 0 | 1 | INTTAB1CC0 |
| 0 | 0 | 1 | 1 | 1 | 0 | INTTAB1CC1 |
| 0 | 0 | 1 | 1 | 1 | 1 | INTTAB1CC2 |
| 0 | 1 | 0 | 0 | 0 | 0 | INTTAB1CC3 |
| 0 | 1 | 0 | 0 | 0 | 1 | INTTT0OV |
| 0 | 1 | 0 | 0 | 1 | 0 | INTTT0CC0 |
| 0 | 1 | 0 | 0 | 1 | 1 | INTTT0CC1 |
| 0 | 1 | 0 | 1 | 0 | 0 | INTTAA0OV |
| 0 | 1 | 0 | 1 | 0 | 1 | INTTAA0CC0 |
| 0 | 1 | 0 | 1 | 1 | 0 | INTTAA0CC1 |
| 0 | 1 | 0 | 1 | 1 | 1 | INTTAA1OV |
| 0 | 1 | 1 | 0 | 0 | 0 | INTTAA1CC0 |
| 0 | 1 | 1 | 0 | 0 | 1 | INTTAA1CC1 |
| 0 | 1 | 1 | 0 | 1 | 0 | INTTAA2CC0 |
| 0 | 1 | 1 | 0 | 1 | 1 | INTTAA2CC1 |
| 0 | 1 | 1 | 1 | 0 | 0 | INTTAA3CC0 |
| 0 | 1 | 1 | 1 | 0 | 1 | INTTAA3CC1 |
| 0 | 1 | 1 | 1 | 1 | 0 | INTTAA4CC0 |
| 0 | 1 | 1 | 1 | 1 | 1 | INTTAA4CC1 |
| 1 | 0 | 0 | 0 | 0 | 0 | INTTAA5CC0 |
| 1 | 0 | 0 | 0 | 0 | 1 | INTTAA5CC1 |
| 1 | 0 | 0 | 0 | 1 | 0 | INTTM0EQ0 |
| 1 | 0 | 0 | 0 | 1 | 1 | INTTM1EQ0 |
| 1 | 0 | 0 | 1 | 0 | 0 | INTTM2EQ0 |
| 1 | 0 | 0 | 1 | 0 | 1 | INTTM3EQ0 |
| 1 | 0 | 0 | 1 | 1 | 0 | INTCF0R/INTIIC1 |
| 1 | 0 | 0 | 1 | 1 | 1 | INTCF0T |
| 1 | 0 | 1 | 0 | 0 | 0 | INTCF1R |
| 1 | 0 | 1 | 0 | 0 | 1 | INTCF1T |

**Note**  INTTAB1OV_BASE is the interrupt signal from before the overflow interrupt of TAB1 (INTTAB1OV) was culled by TMQOP.

**Table 22-1. DMA Start Factors (2/2)**

| IFCn5 | IFCn4 | IFCn3 | IFCn2 | IFCn1 | IFCn0 | Interrupt Source |
|-------|-------|-------|-------|-------|-------|------------------|
| 1 | 0 | 1 | 0 | 1 | 0 | INTCF2R |
| 1 | 0 | 1 | 0 | 1 | 1 | INTCF2T |
| 1 | 0 | 1 | 1 | 0 | 0 | INTCF3R |
| 1 | 0 | 1 | 1 | 0 | 1 | INTCF3T |
| 1 | 0 | 1 | 1 | 1 | 0 | INTCF4R |
| 1 | 0 | 1 | 1 | 1 | 1 | INTCF4T |
| 1 | 1 | 0 | 0 | 0 | 0 | INTUC0R |
| 1 | 1 | 0 | 0 | 0 | 1 | INTUC0T |
| 1 | 1 | 0 | 0 | 1 | 0 | INTUC1R/INTIIC2 |
| 1 | 1 | 0 | 0 | 1 | 1 | INTUC1T |
| 1 | 1 | 0 | 1 | 0 | 0 | INTUC2R |
| 1 | 1 | 0 | 1 | 0 | 1 | INTUC2T |
| 1 | 1 | 0 | 1 | 1 | 0 | INTUC3R/INTIIC0 |
| 1 | 1 | 0 | 1 | 1 | 1 | INTUC3T |
| 1 | 1 | 1 | 0 | 0 | 0 | INTUC4R |
| 1 | 1 | 1 | 0 | 0 | 1 | INTUC4T |
| 1 | 1 | 1 | 0 | 1 | 0 | INTAD |
| 1 | 1 | 1 | 0 | 1 | 1 | INTKR |
| 1 | 1 | 1 | 1 | 0 | 0 | INTRTC1 |

**Remark**　n = 0 to 3

**(7) External DMA request enable register (EXDRQEN)**

The EXDRQEN register sets the DMA request to each DMA channel when connecting the external USB device by using the $\overline{\text{UDMARQm}}$/$\overline{\text{UDMAAKm}}$ pin (m = 0, 1).

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

After reset: 00H     R/W     Address: FFFFFF60H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EXDRQEN | RQ3EX1E | RQ2EX1E | RQ1EX1E | RQ0EX1E | RQ3EX0E | RQ2EX0E | RQ1EX0E | RQ0EX0E |

| RQnEX1E | Assignment of DMA channel n (n = 0 to 3) |
|---|---|
| 0 | Does not assign DMA channel n to $\overline{\text{UDMARQ1}}$/$\overline{\text{UDMAAK1}}$ pin |
| 1 | Assigns DMA channel n to $\overline{\text{UDMARQ1}}$/$\overline{\text{UDMAAK1}}$ pin |

| RQnEX0E | Assignment of DMA channel n (n = 0 to 3) |
|---|---|
| 0 | Does not assign DMA channel n to $\overline{\text{UDMARQ0}}$/$\overline{\text{UDMAAK0}}$ pin |
| 1 | Assigns DMA channel n to $\overline{\text{UDMARQ0}}$/$\overline{\text{UDMAAK0}}$ pin |

Cautions 1. **Assigning multiple DMA channels to the $\overline{\text{UDMARQ1}}$/$\overline{\text{UDMAAK1}}$ pin is prohibited (setting the RQ3EX1E, RQ2EX1E, RQ1EX1E, and RQ0EX1E bits to the $\overline{\text{UDMARQ1}}$/$\overline{\text{UDMAAK1}}$ pin at the same time is prohibited).**

2. **Assigning multiple DMA channels to the $\overline{\text{UDMARQ0}}$/$\overline{\text{UDMAAK0}}$ pin is prohibited (setting the RQ3EX0E, RQ2EX0E, RQ1EX0E, and RQ0EX0E bits to the $\overline{\text{UDMARQ0}}$/$\overline{\text{UDMAAK0}}$ pin at the same time is prohibited).**

3. **Assigning both the $\overline{\text{UDMARQ1}}$/$\overline{\text{UDMAAK1}}$ pin and the $\overline{\text{UDMARQ0}}$/$\overline{\text{UDMAAK0}}$ pin to the same DMA channel is prohibited (setting the RQ3EX1E and RQ3EX0E, RQ2EX1E and RQ2EX0E, RQ1EX1E and RQ1EX0E, and RQ0EX1E and RQ0EX0E bits respectively at the same time is prohibited).**

4. **When using a DMA request from an external source by setting the EXDRQEN register, set the DTFRn.IFCn5-IFCn0 bit to 000000 (to prohibit a DMA request via an interrupt). For details, see 22.3 (6)  DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3).**

## 22.4 Transfer Targets

Table 22-2 shows the relationship between the transfer targets (√: Transfer enabled, ×: Transfer disabled).

**Table 22-2. Relationship Between Transfer Targets**

| | | Transfer Destination | | | |
|---|---|---|---|---|---|
| | | Internal ROM | On-Chip Peripheral I/O | Internal RAM | External Memory |
| Source | On-chip peripheral I/O | × | √ | √ | √ |
| | Internal RAM | × | √ | × | √ |
| | External memory | × | √ | √ | √ |
| | Internal ROM | × | × | × | × |

**Caution** **The operation is not guaranteed for combinations of transfer destination and source marked with "×" in Table 22-2.**

## 22.5 Transfer Modes

Single transfer is supported as the transfer mode.

In single transfer mode, the bus is released at each byte/halfword transfer. If there is a subsequent DMA transfer request, transfer is performed again once. This operation continues until a terminal count occurs.

When the DMAC has released the bus, if another higher priority DMA transfer request is issued, the higher priority DMA request always takes precedence.

If a new transfer request of the same channel and a transfer request of another channel with a lower priority are generated in a transfer cycle, DMA transfer of the channel with the lower priority is executed after the bus is released to the CPU (the new transfer request of the same channel is ignored in the transfer cycle).

## 22.6　Transfer Types

As a transfer type, the 2-cycle transfer is supported.

In two-cycle transfer, data transfer is performed in two cycles, a read cycle and a write cycle.

In the read cycle, the transfer source address is output and reading is performed from the source to the DMAC.  In the write cycle, the transfer destination address is output and writing is performed from the DMAC to the destination.

An idle cycle of one clock is always inserted between a read cycle and a write cycle.  If the data bus width differs between the transfer source and destination for DMA transfer of two cycles, the operation is performed as follows.

<16-bit data transfer>
　<1>　Transfer from 32-bit bus → 16-bit bus

　　　　A read cycle (the higher 16 bits are in a high-impedance state) is generated, followed by generation of a write cycle (16 bits).

　<2>　Transfer from 16-/32-bit bus to 8-bit bus

　　　　A 16-bit read cycle is generated once, and then an 8-bit write cycle is generated twice.

　<3>　Transfer from 8-bit bus to 16-/32-bit bus

　　　　An 8-bit read cycle is generated twice, and then a 16-bit write cycle is generated once.

　<4>　Transfer between 16-bit bus and 32-bit bus

　　　　A 16-bit read cycle is generated once, and then a 16-bit write cycle is generated once.

For DMA transfer executed to an on-chip peripheral I/O register (transfer source/destination), be sure to specify the same transfer size as the register size.  For example, for DMA transfer to an 8-bit register, be sure to specify byte (8-bit) transfer.

**Remark**　The bus width of each transfer target (transfer source/destination) is as follows.
- On-chip peripheral I/O:　16-bit bus width
- Internal RAM:　　　　　　32-bit bus width
- External memory:　　　　 8-bit or 16-bit bus width

## 22.7 DMA Channel Priorities

The DMA channel priorities are fixed as follows.

DMA channel 0 > DMA channel 1 > DMA channel 2 > DMA channel 3

The priorities are checked for every transfer cycle.

## 22.8 Time Related to DMA Transfer

The time required to respond to a DMA request, and the minimum number of clocks required for DMA transfer are shown below.

Single transfer: DMA response time (<1>) + Transfer source memory access (<2>) + $1^{\text{Note 1}}$ + Transfer destination memory access (<2>)

| DMA Cycle | | Minimum Number of Execution Clocks |
|---|---|---|
| <1> DMA request response time | | 4 clocks (MIN.) + Noise elimination time[Note 2] |
| <2> Memory access | External memory access | Depends on connected memory. |
| | Internal RAM access | 2 clocks[Note 3] |
| | Peripheral I/O register access | 3 clocks + Number of wait cycles specified by VSWC register[Note 4] |
| | USB register access | 4 clocks[Note 5] |
| | Data-only RAM access | 4 clocks[Note 5] |

Notes 1. One clock is always inserted between a read cycle and a write cycle in DMA transfer.
   2. If an external interrupt (INTPn) is specified as the trigger to start DMA transfer, noise elimination time is added (n = 00 to 18).
   3. Two clocks are required for a DMA cycle.
   4. More wait cycles are necessary for accessing a specific peripheral I/O register (for details, see **3.4.9 (2)**).
   5. This is the number of clocks required when the following wait specifications have been made: 1 data wait (set by the DWC0 register), 0 address waits (set by the AWC register), and 0 idle waits (set by the BCC register).

## 22.9  DMA Transfer Start Factors

There are two types of DMA transfer start factors, as shown below.

### (1)  Request by software

If the STGn bit is set to 1 while the DCHCn.TCn bit = 1 and Enn bit = 1 (DMA transfer enabled), DMA transfer is started.

To request the next DMA transfer cycle immediately after that, confirm, by using the DBCn register, that the preceding DMA transfer cycle has been completed, and set the STGn bit to 1 again (n = 0 to 3).

> TCn bit = 0, Enn bit = 1
>
> ↓
>
> STGn bit = 1 … Starts the first DMA transfer.
>
> ↓
>
> Confirm that the contents of the DBCn register have been updated.
>
> STGn bit = 1 … Starts the second DMA transfer.
>
> ↓
>
> :
>
> ↓
>
> Generation of terminal count … Enn bit = 0, TCn bit = 1, and INTDMAn signal is generated.
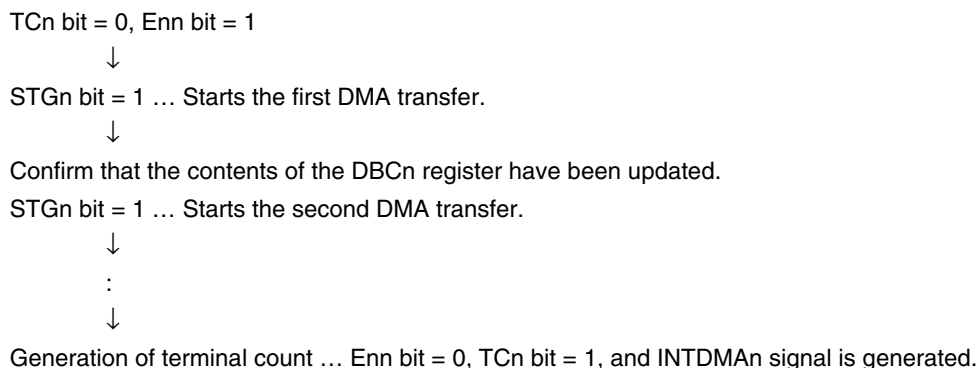
### (2)  Request by on-chip peripheral I/O

If an interrupt request is generated from the on-chip peripheral I/O set by the DTFRn register when the DCHCn.TCn bit = 0 and Enn bit = 1 (DMA transfer enabled), DMA transfer is started.

**Cautions 1.  Two start factors (software trigger and hardware trigger) cannot be used for one DMA channel. If two start factors are simultaneously generated for one DMA channel, only one of them is valid. The start factor that is valid cannot be identified.**

**2.  A new transfer request that is generated after the preceding DMA transfer request was generated or in the preceding DMA transfer cycle is ignored (cleared).**

**3.  The transfer request interval of the same DMA channel varies depending on the setting of bus wait in the DMA transfer cycle, the start status of the other channels, or the external bus hold request. In particular, as described in Caution 2, a new transfer request that is generated for the same channel before the DMA transfer cycle or during the DMA transfer cycle is ignored. Therefore, the transfer request intervals for the same DMA channel must be sufficiently separated by the system. When the software trigger is used, completion of the DMA transfer cycle that was generated before can be checked by updating the DBCn register.**

## 22.10  DMA Abort Factors

DMA transfer is aborted if a bus hold occurs.

The same applies if transfer is executed between the internal memory/on-chip peripheral I/O and internal memory/on-chip peripheral I/O.

When the bus hold is cleared, DMA transfer is resumed.

## 22.11  End of DMA Transfer

When DMA transfer has been completed the number of times set to the DBCn register and when the DCHCn.Enn bit is cleared to 0 and TCn bit is set to 1, a DMA transfer end interrupt request signal (INTDMAn) is generated for the interrupt controller (INTC) (n = 0 to 3).

The V850ES/JG3-U and V850ES/JH3-U do not output a terminal count signal to external devices. Therefore, confirm completion of DMA transfer by using the DMA transfer end interrupt or polling the TCn bit.

## 22.12  Operation Timing

Figures 22-1 to 22-4 show DMA operation timing.

**Figure 22-1. Priority of DMA (1)**



**Remarks 1.** Transfer in the order of DMA0 → DMA1 → DMA2

**2.** In the case of transfer between external memory spaces (multiplexed bus, no wait)
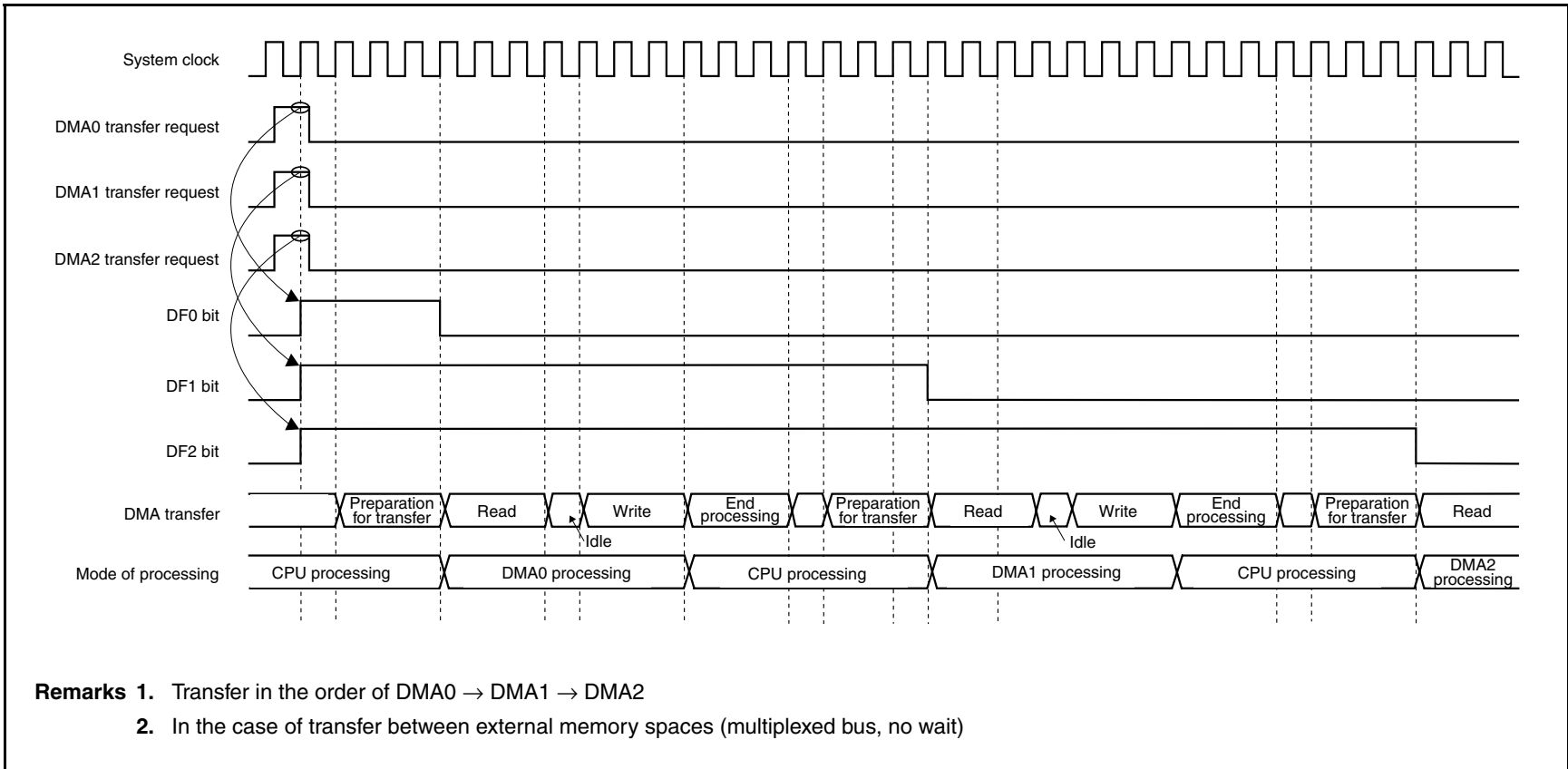
**Figure 22-2. Priority of DMA (2)**



**Remarks 1.** Transfer in the order of DMA0 → DMA1 → DMA0 (DMA2 is held pending.)

**2.** In the case of transfer between external memory spaces (multiplexed bus, no wait)
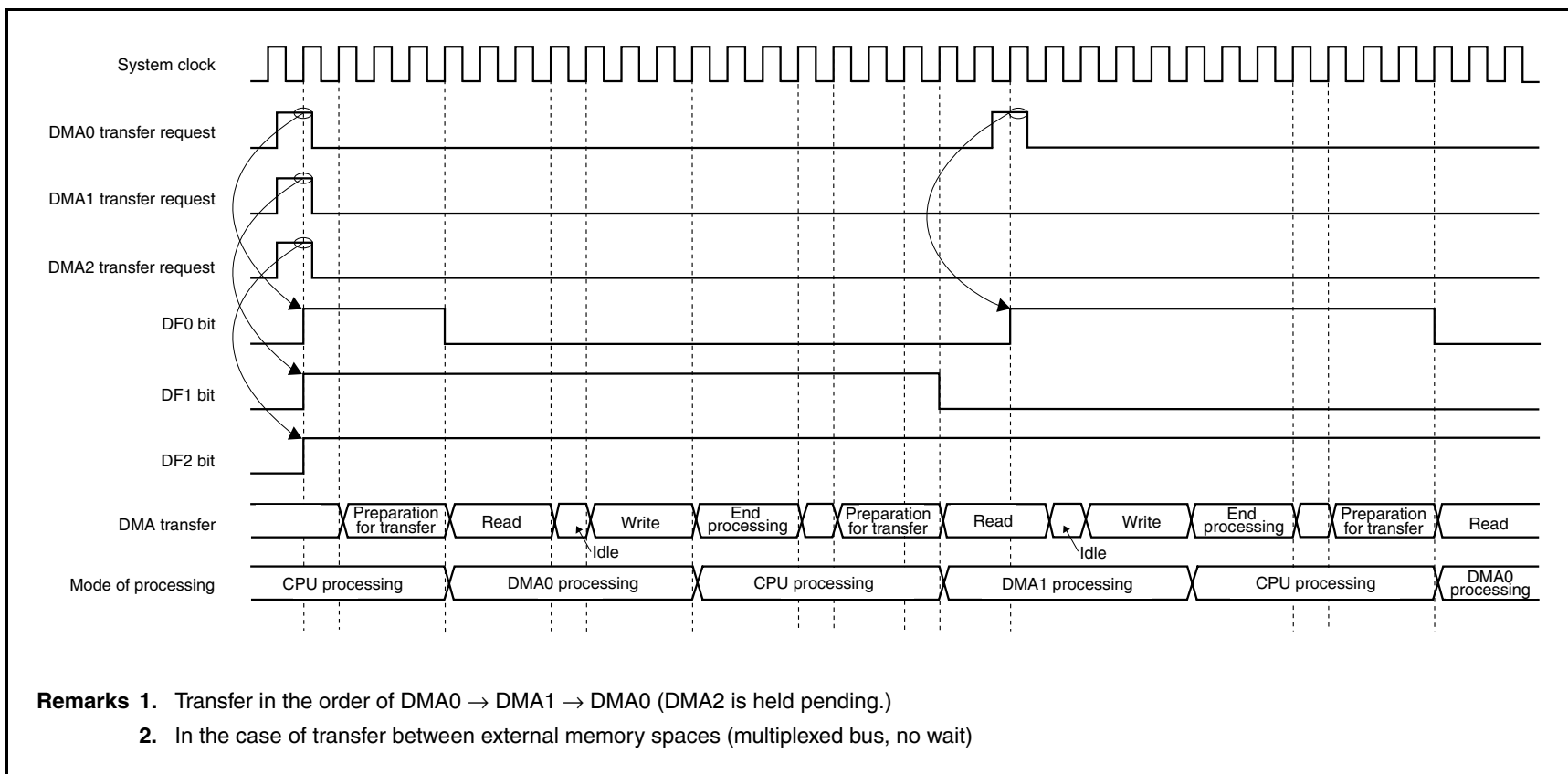
**Figure 22-3.  Period in Which DMA Transfer Request Is Ignored (1)**



**Notes 1.** Interrupt from on-chip peripheral I/O, or software trigger (STGn bit)

   **2.** New DMA request of the same channel is ignored between when the first request is generated and the end processing is complete.

**Remark** In the case of transfer between external memory spaces (multiplexed bus, no wait)
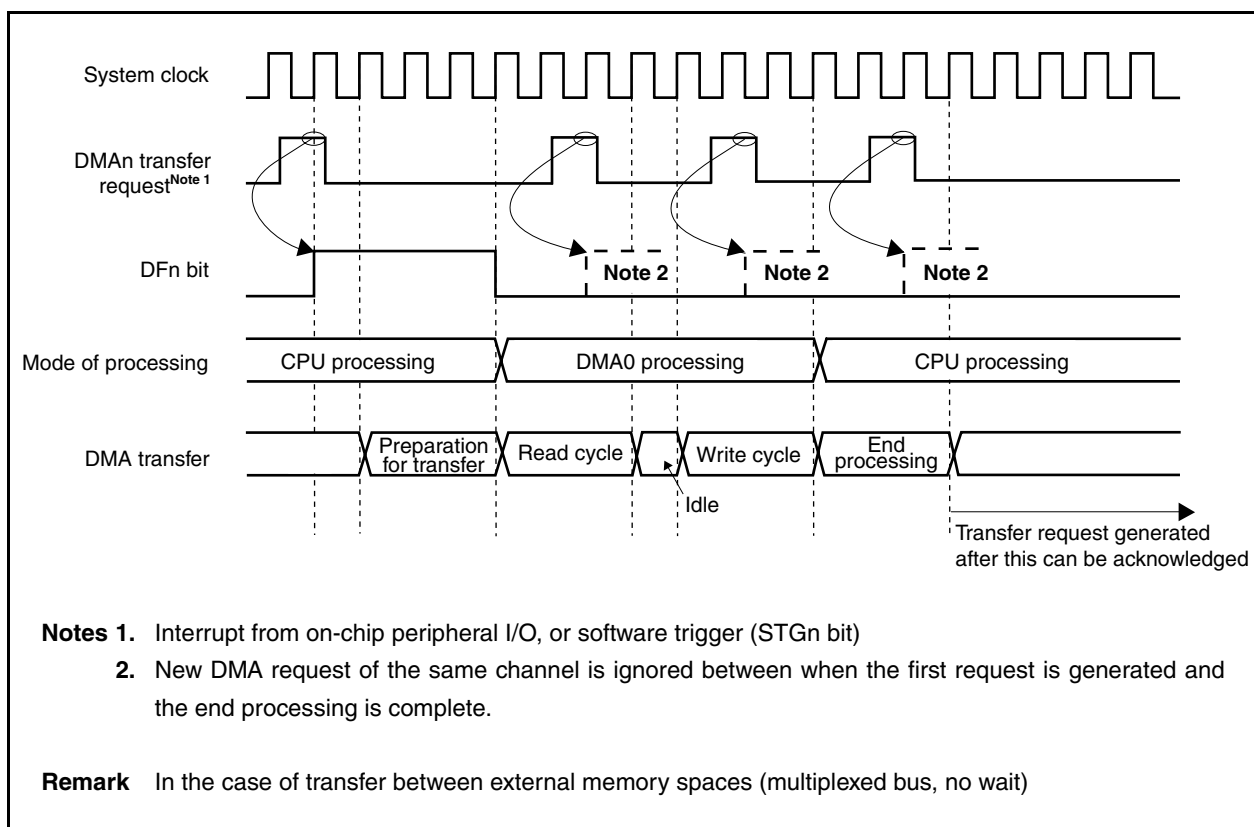
**Figure 22-4. Period in Which DMA Transfer Request Is Ignored (2)**



<1> DMA0 transfer request

<2> New DMA0 transfer request is generated during DMA0 transfer.

→ A DMA transfer request of the same channel is ignored during DMA transfer.

<3> Requests for DMA0 and DMA1 are generated at the same time.

→ DMA0 request is ignored (a DMA transfer request of the same channel during transfer is ignored).

→ DMA1 request is acknowledged.

<4> Requests for DMA0, DMA1, and DMA2 are generated at the same time.

→ DMA1 request is ignored (a DMA transfer request of the same channel during transfer is ignored).

→ DMA0 request is acknowledged according to priority. DMA2 request is held pending (transfer of DMA2 occurs next).

## 22.13 Cautions

**(1) Caution for VSWC register**

When using the DMAC, be sure to set an appropriate value, in accordance with the operating frequency, to the VSWC register.

When the default value (77H) of the VSWC register is used, or if an inappropriate value is set to the VSWC register, the operation is not correctly performed (for details of the VSWC register, see **3.4.8 (1) (a) System wait control register (VSWC)**).

**(2) Caution for reading DCHCn.TCn bit (n = 0 to 3)**

The TCn bit is cleared to 0 when it is read, but it is not automatically cleared even if it is read at a specific timing. To accurately clear the TCn bit, add the following processing.

**(a) When waiting for completion of DMA transfer by polling TCn bit**

Confirm that the TCn bit has been set to 1 (after TCn bit = 1 is read), and then read the TCn bit three more times.

**(b) When reading TCn bit in interrupt servicing routine**

Execute reading the TCn bit three times.

**(3) DMA transfer initialization procedure (setting DCHCn.INITn bit to 1)**

Even if the INITn bit is set to 1 when the channel executing DMA transfer is to be initialized, the channel may not be initialized. To accurately initialize the channel, execute either of the following two procedures.

**(a) Temporarily stop transfer of all DMA channels**

Initialize the channel executing DMA transfer using the procedure in <1> to <7> below.

Note, however, that TCn bit is cleared to 0 when step <5> is executed. Make sure that the other processing programs do not expect that the TCn bit is 1.

<1> Disable interrupts (DI).

<2> Read the DCHCn.Enn bit of DMA channels other than the one to be forcibly terminated, and transfer the value to a general-purpose register.

<3> Clear the Enn bit of the DMA channels used (including the channel to be forcibly terminated) to 0. To clear the Enn bit of the last DMA channel, execute the clear instruction twice. If the target of DMA transfer (transfer source/destination) is the internal RAM, execute the instruction three times.

Example: Execute instructions in the following order if channels 0, 1, and 2 are used (if the target of transfer is not the internal RAM).
- Clear DCHC0.E00 bit to 0.
- Clear DCHC1.E11 bit to 0.
- Clear DCHC2.E22 bit to 0.
- Clear DCHC2.E22 bit to 0 again.

<4> Set the INITn bit of the channel to be forcibly terminated to 1.

<5> Read the TCn bit of each channel not to be forcibly terminated. If both the TCn bit and the Enn bit read in <2> are 1 (logical product (AND) is 1), clear the saved Enn bit to 0.

<6> After the operation in <5>, write the Enn bit value to the DCHCn register.

<7> Enable interrupts (EI).

**Caution   Be sure to execute step <5> above to prevent illegal setting of the Enn bit of the channels whose DMA transfer has been normally completed between <2> and <3>.**

**(b) Repeatedly execute setting INITn bit until transfer is forcibly terminated correctly**

&lt;1&gt;  Suppress a request from the DMA request source of the channel to be forcibly terminated (stop operation of the on-chip peripheral I/O).

&lt;2&gt;  Check that the DMA transfer request of the channel to be forcibly terminated is not held pending, by using the DTFRn.DFn bit.  If a DMA transfer request is held pending, wait until execution of the pending request is completed.

&lt;3&gt;  When it has been confirmed that the DMA request of the channel to be forcibly terminated is not held pending, clear the Enn bit to 0.

&lt;4&gt;  Again, clear the Enn bit of the channel to be forcibly terminated.
If the target of transfer for the channel to be forcibly terminated (transfer source/destination) is the internal RAM, execute this operation once more.

&lt;5&gt;  Copy the initial number of transfers of the channel to be forcibly terminated to a general-purpose register.

&lt;6&gt;  Set the INITn bit of the channel to be forcibly terminated to 1.

&lt;7&gt;  Read the value of the DBCn register of the channel to be forcibly terminated, and compare it with the value copied in &lt;5&gt;.  If the two values do not match, repeat operations &lt;6&gt; and &lt;7&gt;.

Remarks **1.** When the value of the DBCn register is read in &lt;7&gt;, the initial number of transfers is read if forced termination has been correctly completed.  If not, the remaining number of transfers is read.

**2.** Note that method (b) may take a long time if the application frequently uses DMA transfer for a channel other than the DMA channel to be forcibly terminated.

**(4) Procedure of temporarily stopping DMA transfer (clearing Enn bit)**
Stop and resume the DMA transfer under execution using the following procedure.

&lt;1&gt;  Suppress a transfer request from the DMA request source (stop the operation of the on-chip peripheral I/O).

&lt;2&gt;  Check the DMA transfer request is not held pending, by using the DFn bit (check if the DFn bit = 0).
If a request is pending, wait until execution of the pending DMA transfer request is completed.

&lt;3&gt;  If it has been confirmed that no DMA transfer request is held pending, clear the Enn bit to 0 (this operation stops DMA transfer).

&lt;4&gt;  Set the Enn bit to 1 to resume DMA transfer.

&lt;5&gt;  Resume the operation of the DMA request source that has been stopped (start the operation of the on-chip peripheral I/O).

**(5) Memory boundary**
The operation is not guaranteed if the address of the transfer source or destination exceeds the area of the DMA target (external memory, internal RAM, or on-chip peripheral I/O) during DMA transfer.

**(6) Transferring misaligned data**
DMA transfer of misaligned data with a 16-bit bus width is not supported.
If an odd address is specified as the transfer source or destination, the least significant bit of the address is forcibly assumed to be 0.

**(7) Bus arbitration for CPU**
Because the DMA controller has a higher priority bus mastership than the CPU, a CPU access that takes place during DMA transfer is held pending until the DMA transfer cycle is completed and the bus is released to the CPU.
However, the CPU can access the external memory, internal peripheral I/O, and internal RAM for which DMA transfer is not being executed.

- The CPU can access the internal ROM and internal RAM when DMA transfer is being executed between the external memory and on-chip peripheral I/O.
- The CPU can access the internal ROM, and internal peripheral I/O when DMA transfer is being executed between external memories.

**(8) Registers/bits that must not be rewritten during DMA operation**

Set the following registers at the following timing when a DMA operation is not under execution.

[Registers]

- DSAnH, DSAnL, DDAnH, DDAnL, DBCn, and DADCn registers
- DTFRn.IFCn5 to DTFRn.IFCn0 bits

[Settable timing]

- Period from after reset to start of the first DMA transfer
- Time after channel initialization to start of DMA transfer
- Period from after completion of DMA transfer (TCn bit = 1) to start of the next DMA transfer

**(9) Be sure to set the following register bits to 0.**

- Bits 14 to 10 of DSAnH register
- Bits 14 to 10 of DDAnH register
- Bits 15, 13 to 8, and 3 to 0 of DADCn register
- Bits 6 to 3 of DCHCn register

**(10) DMA start factor**

Do not start two or more DMA channels with the same start factor. If two or more channels are started with the same factor, DMA for which a channel has already been set may be started or a DMA channel with a lower priority may be acknowledged earlier than a DMA channel with a higher priority. The operation cannot be guaranteed.

**(11) Read values of DSAn and DDAn registers**

Values in the middle of updating may be read from the DSAn and DDAn registers during DMA transfer (n = 0 to 3). For example, if the DSAnH register and then the DSAnL register are read when the DMA transfer source address (DSAn register) is 0000FFFFH and the count direction is incremental (DADCn.SAD1 and DADCn.SAD0 bits = 00), the value of the DSAn register differs as follows, depending on whether DMA transfer is executed immediately after the DSAnH register is read.

**(a) If DMA transfer does not occur while DSAn register is read**

<1> Read value of DSAnH register: DSAnH = 0000H
<2> Read value of DSAnL register: DSAnL = FFFFH

**(b) If DMA transfer occurs while DSAn register is read**

<1> Read value of DSAnH register: DSAnH = 0000H
<2> Occurrence of DMA transfer
<3> Incrementing DSAn register: DSAn = 00100000H
<4> Read value of DSAnL register: DSAnL = 0000H

# CHAPTER 23  INTERRUPT/EXCEPTION PROCESSING FUNCTION

The V850ES/JG3-U and V850ES/JH3-U are provided with a dedicated interrupt controller (INTC) for interrupt servicing and can process a total of 87 to 92 interrupt requests.

An interrupt is an event that occurs independently of program execution, and an exception is an event whose occurrence is dependent on program execution.

The V850ES/JG3-U and V850ES/JH3-U can process interrupt request signals from the on-chip peripheral hardware and external sources.  Moreover, exception processing can be started by the TRAP instruction (software exception) or by generation of an exception event (i.e. fetching of an illegal opcode) (exception trap).

## 23.1  Features

○ Interrupts

**Table 23-1.  Interrupts of V850ES/JG3-U and V850ES/JH3-U**

| | | Internal | | | External | | |
|---|---|---|---|---|---|---|---|
| | | Non-maskable | Maskable | Total | Non-maskable | Maskable | Total |
| V850ES/JG3-U | $\mu$PD70F3763 | 1 | 71 | 72 | 1 | 14 | 15 |
| | $\mu$PD70F3764 | 1 | 71 | 72 | 1 | 14 | 15 |
| V850ES/JH3-U | $\mu$PD70F3768 | 1 | 71 | 72 | 1 | 19 | 20 |
| | $\mu$PD70F3769 | 1 | 71 | 72 | 1 | 19 | 20 |

- 8 levels of programmable priorities (maskable interrupts)
- Multiple interrupt control according to priority
- Masks can be specified for each maskable interrupt request.
- Noise elimination, edge detection, and valid edge specification for external interrupt request signals.

○ Exceptions
- Software exceptions:  32 sources
- Exception trap:         2 sources (illegal opcode exception)

Interrupt/exception sources of the V850ES/JG3-U and V850ES/JH3-U are listed in **Tables 23-2** and **23-3**, respectively.

**Table 23-2.  V850ES/JG3-U Interrupt Sources (1/4)**

| Type | Classification | Default Priority | Name | Trigger | Generating Unit | Exception Code | Handler Address | Restored PC | Interrupt Control Register |
|------|---------------|------------------|------|---------|-----------------|----------------|-----------------|-------------|---------------------------|
| Reset | Interrupt | – | RESET | $\overline{\text{RESET}}$ pin input/ reset input by internal source | RESET | 0000H | 00000000H | Undefined | – |
| Non-maskable | Interrupt | – | NMI | NMI pin valid edge input | Pin | 0010H | 00000010H | nextPC | – |
| | | – | INTWDT2 | WDT2 overflow | WDT2 | 0020H | 00000020H | **Note 1** | – |
| Software exception | Exception | – | TRAP0n[Note 2] | TRAP instruction | – | 004nH[Note 2] | 00000040H | nextPC | – |
| | | – | TRAP1n[Note 2] | TRAP instruction | – | 005nH[Note 2] | 00000050H | nextPC | – |
| Exception trap | Exception | – | ILGOP/ DBG0 | Illegal opcode/DBTRAP instruction | – | 0060H | 00000060H | nextPC | – |
| Maskable | Interrupt | 0 | INTLVI | Low voltage detection | POCLVI | 0080H | 00000080H | nextPC | LVIIC |
| | | 3 | INTP02 | External interrupt pin input edge detection (INTP02) | Pin | 00B0H | 000000B0H | nextPC | PIC02 |
| | | 6 | INTP05 | External interrupt pin input edge detection (INTP05) | Pin | 00E0H | 000000E0H | nextPC | PIC05 |
| | | 8 | INTP07 | External interrupt pin input edge detection (INTP07) | Pin | 0100H | 00000100H | nextPC | PIC07 |
| | | 9 | INTP08 | External interrupt pin input edge detection (INTP08) | Pin | 0110H | 00000110H | nextPC | PIC08 |
| | | 10 | INTP09 | External interrupt pin input edge detection (INTP09) | Pin | 0120H | 00000120H | nextPC | PIC09 |
| | | 11 | INTP10 | External interrupt pin input edge detection (INTP10) | Pin | 0130H | 00000130H | nextPC | PIC10 |
| | | 12 | INTP11 | External interrupt pin input edge detection (INTP11) | Pin | 0140H | 00000140H | nextPC | PIC11 |
| | | 13 | INTP12 | External interrupt pin input edge detection (INTP12) | Pin | 0150H | 00000150H | nextPC | PIC12 |
| | | 14 | INTP13 | External interrupt pin input edge detection (INTP16) | Pin | 0160H | 00000160H | nextPC | PIC13 |
| | | 15 | INTP14 | External interrupt pin input edge detection (INTP14) | Pin | 0170H | 00000170H | nextPC | PIC14 |
| | | 16 | INTP15 | External interrupt pin input edge detection (INTP15) | Pin | 0180H | 00000180H | nextPC | PIC15 |
| | | 17 | INTP16 | External interrupt pin input edge detection (INTP16) | Pin | 0190H | 00000190H | nextPC | PIC16 |

**Notes 1.** For restoring in the case of INTWDT2, see **23.2.2 (2)  From INTWDT2 signal**.

   **2.** n = 0 to FH

**Table 23-2.  V850ES/JG3-U Interrupt Sources (2/4)**

| Type | Classification | Default Priority | Name | Trigger | Generating Unit | Exception Code | Handler Address | Restored PC | Interrupt Control Register |
|---|---|---|---|---|---|---|---|---|---|
| Maskable | Interrupt | 18 | INTP17 | External interrupt pin input edge detection (INTP17) | Pin | 01A0H | 000001A0H | nextPC | PIC17 |
| | | 19 | INTP18 | External interrupt pin input edge detection (INTP18) | Pin | 01B0H | 000001B0H | nextPC | PIC18 |
| | | 20 | INTTAB0OV | TAB0 overflow | TAB0 | 01C0H | 000001C0H | nextPC | TAB0OVIC |
| | | 21 | INTTAB0CC0 | TAB0 capture 0/ compare 0 match | TAB0 | 01D0H | 000001D0H | nextPC | TAB0CCIC0 |
| | | 22 | INTTAB0CC1 | TAB0 capture 1/ compare 1 match | TAB0 | 01E0H | 000001E0H | nextPC | TAB0CCIC1 |
| | | 23 | INTTAB0CC2 | TAB0 capture 2/ compare 2 match | TAB0 | 01F0H | 000001F0H | nextPC | TAB0CCIC2 |
| | | 24 | INTTAB0CC3 | TAB0 capture 3/ compare 3 match | TAB0 | 0200H | 00000200H | nextPC | TAB0CCIC3 |
| | | 25 | INTTAB1OV[Note 1] | TAB1 overflow | TAB1 | 0210H | 00000210H | nextPC | TAB1OVIC |
| | | 26 | INTTAB1CC0[Note 2] | TAB1 capture 0/ compare 0 match | TAB1 | 0220H | 00000220H | nextPC | TAB1CCIC0 |
| | | 27 | INTTAB1CC1 | TAB1 capture 1/ compare 1 match | TAB1 | 0230H | 00000230H | nextPC | TAB1CCIC1 |
| | | 28 | INTTAB1CC2 | TAB1 capture 2/ compare 2 match | TAB1 | 0240H | 00000240H | nextPC | TAB1CCIC2 |
| | | 29 | INTTAB1CC3 | TAB1 capture 3/ compare 3 match | TAB1 | 0250H | 00000250H | nextPC | TAB1CCIC3 |
| | | 30 | INTTT0OV | TMT0 overflow | TMT0 | 0260H | 00000260H | nextPC | TT0OVIC |
| | | 31 | INTTT0CC0 | TMT0 capture 0/ compare 0 match | TMT0 | 0270H | 00000270H | nextPC | TT0CCIC0 |
| | | 32 | INTTT0CC1 | TMT0 capture 1/ compare 1 match | TMT0 | 0280H | 00000280H | nextPC | TT0CCIC1 |
| | | 33 | INTTT0EC | TMT0 encoder input | TMT0 | 0290H | 00000290H | nextPC | TT0ECIC |
| | | 34 | INTTAA0OV | TAA0 overflow | TAA0 | 02A0H | 000002A0H | nextPC | TAA0OVIC |
| | | 35 | INTTAA0CC0 | TAA0 capture 0/ compare 0 match | TAA0 | 02B0H | 000002B0H | nextPC | TAA0CCIC0 |
| | | 36 | INTTAA0CC1 | TAA0 capture 1/ compare 1 match | TAA0 | 02C0H | 000002C0H | nextPC | TAA0CCIC1 |
| | | 37 | INTTAA1OV | TAA1 overflow | TAA1 | 02D0H | 000002D0H | nextPC | TAA1OVIC |
| | | 38 | INTTAA1CC0 | TAA1 capture 0/ compare 0 match | TAA1 | 02E0H | 000002E0H | nextPC | TAA1CCIC0 |
| | | 39 | INTTAA1CC1 | TAA1 capture 1/ compare 1 match | TAA1 | 02F0H | 000002F0H | nextPC | TAA1CCIC1 |
| | | 40 | INTTAA2OV | TAA2 overflow | TAA2 | 0300H | 00000300H | nextPC | TAA2OVIC |

**Notes  1.** When using TAB1 in the 6-phase PWM output mode, functions as the zero match interrupt (TAB1TIOD) request from TMQOP.

**2.** When using TAB1 in the 6-phase PWM output mode, functions as the compare match interrupt (TAB1TICD0) request from TMQOP.

**Table 23-2.  V850ES/JG3-U Interrupt Sources (3/4)**

| Type | Classification | Default Priority | Name | Trigger | Generating Unit | Exception Code | Handler Address | Restored PC | Interrupt Control Register |
|------|----------------|------------------|------|---------|-----------------|----------------|-----------------|-------------|----------------------------|
| Maskable | Interrupt | 41 | INTTAA2CC0 | TAA2 capture 0/ compare 0 match | TAA2 | 0310H | 00000310H | nextPC | TAA2CCIC0 |
| | | 42 | INTTAA2CC1 | TAA2 capture 1/ compare 1 match | TAA2 | 0320H | 00000320H | nextPC | TAA2CCIC1 |
| | | 43 | INTTAA3OV | TAA3 overflow | TAA3 | 0330H | 00000330H | nextPC | TAA3OVIC |
| | | 44 | INTTAA3CC0 | TAA3 capture 0/ compare 0 match | TAA3 | 0340H | 00000340H | nextPC | TAA3CCIC0 |
| | | 45 | INTTAA3CC1 | TAA3 capture 1/ compare 1 match | TAA3 | 0350H | 00000350H | nextPC | TAA3CCIC1 |
| | | 46 | INTTAA4OV | TAA4 overflow | TAA4 | 0360H | 00000360H | nextPC | TAA4OVIC |
| | | 47 | INTTAA4CC0 | TAA4 compare 0 match | TAA4 | 0370H | 00000370H | nextPC | TAA4CCIC0 |
| | | 48 | INTTAA4CC1 | TAA4 compare 1 match | TAA4 | 0380H | 00000380H | nextPC | TAA4CCIC1 |
| | | 49 | INTTAA5OV | TAA5 overflow | TAA5 | 0390H | 00000390H | nextPC | TAA5OVIC |
| | | 50 | INTTAA5CC0 | TAA5 capture 0/ compare 0 match | TAA5 | 03A0H | 000003A0H | nextPC | TAA5CCIC0 |
| | | 51 | INTTAA5CC1 | TAA5 capture 1/ compare 1 match | TAA5 | 03B0H | 000003B0H | nextPC | TAA5CCIC1 |
| | | 52 | INTTM0EQ0 | TMM0 compare match | TMM0 | 03C0H | 000003C0H | nextPC | TM0EQIC0 |
| | | 53 | INTTM1EQ0 | TMM1 compare match | TMM1 | 03D0H | 000003D0H | nextPC | TM1EQIC0 |
| | | 54 | INTTM2EQ0 | TMM2 compare match | TMM2 | 03E0H | 000003E0H | nextPC | TM2EQIC0 |
| | | 55 | INTTM3EQ0 | TMM3 compare match | TMM3 | 03F0H | 000003F0H | nextPC | TM3EQIC0 |
| | | 56 | INTCF0R /INTIIC1 | CSIF0 reception completion/ CSIF0 reception error/ IIC1 transfer completion | CSIF0/ IIC1 | 0400H | 00000400H | nextPC | CF0RIC/ IICIC1 |
| | | 57 | INTCF0T | CSIF0 consecutive transmission write enable | CSIF0 | 0410H | 00000410H | nextPC | CF0TIC |
| | | 58 | INTCF1R | CSIF1 reception completion/ CSIF1 reception error | CSIF1 | 0420H | 00000420H | nextPC | CF1RIC |
| | | 59 | INTCF1T | CSIF1 consecutive transmission write enable | CSIF1 | 0430H | 00000430H | nextPC | CF1TIC |
| | | 60 | INTCF2R | CSIF2 reception completion/ CSIF2 reception error | CSIF2 | 0440H | 00000440H | nextPC | CF2RIC |
| | | 61 | INTCF2T | CSIF2 consecutive transmission write enable | CSIF2 | 0450H | 00000450H | nextPC | CF2TIC |
| | | 62 | INTCF3R | CSIF3 reception completion/ CSIF3 reception error | CSIF3 | 0460H | 00000460H | nextPC | CF3RIC |
| | | 63 | INTCF3T | CSIF3 consecutive transmission write enable | CSIF3 | 0470H | 00000470H | nextPC | CF3TIC |
| | | 64 | INTCF4R | CSIF4 reception completion/ CSIF4 reception error | CSIF4 | 0480H | 00000480H | nextPC | CF4RIC |
| | | 65 | INTCF4T | CSIF4 consecutive transmission write enable | CSIF4 | 0490H | 00000490H | nextPC | CF4TIC |

**Table 23-2. V850ES/JG3-U Interrupt Sources (4/4)**

| Type | Classification | Default Priority | Name | Trigger | Generating Unit | Exception Code | Handler Address | Restored PC | Interrupt Control Register |
|------|---------------|-----------------|------|---------|----------------|---------------|-----------------|-------------|---------------------------|
| Maskable | Interrupt | 66 | INTUC0R | UARTC0 reception completion/UARTC0 reception error | UARTC0 | 04A0H | 000004A0H | nextPC | UC0RIC |
| | | 67 | INTUC0T | UARTC0 consecutive transmission enable | UARTC0 | 04B0H | 000004B0H | nextPC | UC0TIC |
| | | 68 | INTUC1R/ INTIIC2 | UARTC1 reception completion/UARTC1 reception error/IIC2 transfer completion | UARTC1/ IIC2 | 04C0H | 000004C0H | nextPC | UC1RIC/ IICIC2 |
| | | 69 | INTUC1T | UARTC1 consecutive transmission enable | UARTC1 | 04D0H | 000004D0H | nextPC | UC1TIC |
| | | 70 | INTUC2R | UARTC2 reception completion/UARTC2 reception error | UARTC2 | 04E0H | 000004E0H | nextPC | UC2RIC |
| | | 71 | INTUC2T | UARTC2 consecutive transmission enable | UARTC2 | 04F0H | 000004F0H | nextPC | UC2TIC |
| | | 72 | INTUC3R/ INTIIC0 | UARTC3 reception completion/UARTC0 reception error/IIC0 transfer completion | UARTC3/ IIC0 | 0500H | 00000500H | nextPC | UC3RIC/ IICIC0 |
| | | 73 | INTUC3T | UARTC3 consecutive transmission enable | UARTC3 | 0510H | 00000510H | nextPC | UC3TIC |
| | | 74 | INTUC4R | UARTC4 reception completion/UARTC4 reception error | UARTC4 | 0520H | 00000520H | nextPC | UC4RIC |
| | | 75 | INTUC4T | UARTC4 consecutive transmission enable | UARTC4 | 0530H | 00000530H | nextPC | UC4TIC |
| | | 76 | INTAD | A/D conversion completion | A/D | 0540H | 00000540H | nextPC | ADIC |
| | | 77 | INTDMA0 | DMA0 transfer completion | DMA | 0550H | 00000550H | nextPC | DMAIC0 |
| | | 78 | INTDMA1 | DMA1 transfer completion | DMA | 0560H | 00000560H | nextPC | DMAIC1 |
| | | 79 | INTDMA2 | DMA2 transfer completion | DMA | 0570H | 00000570H | nextPC | DMAIC2 |
| | | 80 | INTDMA3 | DMA3 transfer completion | DMA | 0580H | 00000580H | nextPC | DMAIC3 |
| | | 81 | INTKR | Key return interrupt | KR | 0590H | 00000590H | nextPC | KRIC |
| | | 82 | INTRTC0 | RTC constant cycle signal | RTC | 05A0H | 000005A0H | nextPC | RTC0IC |
| | | 83 | INTRTC1 | RTC alarm match | RTC | 05B0H | 000005B0H | nextPC | RTC1IC |
| | | 84 | INTRTC2 | RTC interval signal | RTC | 05C0H | 000005C0H | nextPC | RTC2IC |
| | | 89 | INTUSBH0 | USBH status interrupt | USBH | 0610H | 00000610H | nextPC | UHIC0 |
| | | 90 | INTUSBH1 | USBH PCI cycle error | USBH | 0620H | 00000620H | nextPC | UHIC1 |
| | | 91 | INTUSBH2 | USBH PME interrupt | USBH | 0630H | 00000630H | nextPC | UHIC2 |
| | | 92 | INTUSBF0 | USBF interrupt | USBF | 0640H | 00000640H | nextPC | UFIC0 |
| | | 93 | INTUSBF1 | USBF resume interrupt | USBF | 0650H | 00000650H | nextPC | UFIC1 |

**Remarks 1.** Default Priority: The priority order when two or more maskable interrupt requests occur at the same time. The highest priority is 0.

The priority order of non-maskable interrupt is INTWDT2 > NMI.

Restored PC: The value of the program counter (PC) saved to EIPC, FEPC, or DBPC when interrupt servicing is started.  Note, however, that the restored PC when a non-maskable or maskable interrupt is acknowledged while one of the following instructions is being executed does not become the nextPC (if an interrupt is acknowledged during interrupt execution, execution stops, and then resumes after the interrupt servicing has finished).

- Load instructions (SLD.B, SLD.BU, SLD.H, SLD.HU, SLD.W)
- Division instructions (DIV, DIVH, DIVU, DIVHU)
- PREPARE, DISPOSE instructions (only if an interrupt is generated before the stack pointer is updated)

nextPC:          The PC value that starts the processing following interrupt/exception processing.

**2.** The execution address of the illegal instruction when an illegal opcode exception occurs is calculated by (Restored PC − 4).

**Table 23-3.  V850ES/JH3-U Interrupt Sources (1/4)**

| Type | Classification | Default Priority | Name | Trigger | Generating Unit | Exception Code | Handler Address | Restored PC | Interrupt Control Register |
|---|---|---|---|---|---|---|---|---|---|
| Reset | Interrupt | – | RESET | $\overline{\text{RESET}}$ pin input/ Reset input by internal source | RESET | 0000H | 00000000H | Undefined | – |
| Non-maskable | Interrupt | – | NMI | NMI pin valid edge input | Pin | 0010H | 00000010H | nextPC | – |
| | | – | INTWDT2 | WDT2 overflow | WDT2 | 0020H | 00000020H | **Note 1** | – |
| Software exception | Exception | – | TRAP0n[Note 2] | TRAP instruction | – | 004nH[Note 2] | 00000040H | nextPC | – |
| | | – | TRAP1n[Note 2] | TRAP instruction | – | 005nH[Note 2] | 00000050H | nextPC | – |
| Exception trap | Exception | – | ILGOP/DBG0 | Illegal opcode/ DBTRAP instruction | – | 0060H | 00000060H | nextPC | – |
| Maskable | Interrupt | 0 | INTLVI | Low voltage detection | POCLVI | 0080H | 00000080H | nextPC | LVIIC |
| | | 1 | INTP00 | External interrupt pin input edge detection (INTP00) | Pin | 0090H | 00000090H | nextPC | PIC00 |
| | | 2 | INTP01 | External interrupt pin input edge detection (INTP01) | Pin | 00A0H | 000000A0H | nextPC | PIC01 |
| | | 3 | INTP02 | External interrupt pin input edge detection (INTP02) | Pin | 00B0H | 000000B0H | nextPC | PIC02 |
| | | 4 | INTP03 | External interrupt pin input edge detection (INTP03) | Pin | 00C0H | 000000C0H | nextPC | PIC03 |
| | | 5 | INTP04 | External interrupt pin input edge detection (INTP04) | Pin | 00D0H | 000000D0H | nextPC | PIC04 |
| | | 6 | INTP05 | External interrupt pin input edge detection (INTP05) | Pin | 00E0H | 000000E0H | nextPC | PIC05 |
| | | 7 | INTP06 | External interrupt pin input edge detection (INTP06) | Pin | 00F0H | 000000F0H | nextPC | PIC06 |
| | | 8 | INTP07 | External interrupt pin input edge detection (INTP07) | Pin | 0100H | 00000100H | nextPC | PIC07 |
| | | 9 | INTP08 | External interrupt pin input edge detection (INTP08) | Pin | 0110H | 00000110H | nextPC | PIC08 |
| | | 10 | INTP09 | External interrupt pin input edge detection (INTP09) | Pin | 0120H | 00000120H | nextPC | PIC09 |
| | | 11 | INTP10 | External interrupt pin input edge detection (INTP10) | Pin | 0130H | 00000130H | nextPC | PIC10 |
| | | 12 | INTP11 | External interrupt pin input edge detection (INTP11) | Pin | 0140H | 00000140H | nextPC | PIC11 |
| | | 13 | INTP12 | External interrupt pin input edge detection (INTP12) | Pin | 0150H | 00000150H | nextPC | PIC12 |
| | | 14 | INTP13 | External interrupt pin input edge detection (INTP16) | Pin | 0160H | 00000160H | nextPC | PIC13 |
| | | 15 | INTP14 | External interrupt pin input edge detection (INTP14) | Pin | 0170H | 00000170H | nextPC | PIC14 |

**Notes 1.** For restoring in the case of INTWDT2, see **23.2.2 (2)  From INTWDT2 signal**.

   **2.** n = 0 to FH

**Table 23-3. V850ES/JH3-U Interrupt Sources (2/4)**

| Type | Classification | Default Priority | Name | Trigger | Generating Unit | Exception Code | Handler Address | Restored PC | Interrupt Control Register |
|---|---|---|---|---|---|---|---|---|---|
| Maskable | Interrupt | 16 | INTP15 | External interrupt pin input edge detection (INTP15) | Pin | 0180H | 00000180H | nextPC | PIC15 |
| | | 17 | INTP16 | External interrupt pin input edge detection (INTP16) | Pin | 0190H | 00000190H | nextPC | PIC16 |
| | | 18 | INTP17 | External interrupt pin input edge detection (INTP17) | Pin | 01A0H | 000001A0H | nextPC | PIC17 |
| | | 19 | INTP18 | External interrupt pin input edge detection (INTP18) | Pin | 01B0H | 000001B0H | nextPC | PIC18 |
| | | 20 | INTTAB0OV | TAB0 overflow | TAB0 | 01C0H | 000001C0H | nextPC | TAB0OVIC |
| | | 21 | INTTAB0CC0 | TAB0 capture 0/ compare 0 match | TAB0 | 01D0H | 000001D0H | nextPC | TAB0CCIC0 |
| | | 22 | INTTAB0CC1 | TAB0 capture 1/ compare 1 match | TAB0 | 01E0H | 000001E0H | nextPC | TAB0CCIC1 |
| | | 23 | INTTAB0CC2 | TAB0 capture 2/ compare 2 match | TAB0 | 01F0H | 000001F0H | nextPC | TAB0CCIC2 |
| | | 24 | INTTAB0CC3 | TAB0 capture 3/ compare 3 match | TAB0 | 0200H | 00000200H | nextPC | TAB0CCIC3 |
| | | 25 | INTTAB1OV[Note 1] | TAB1 overflow | TAB1 | 0210H | 00000210H | nextPC | TAB1OVIC |
| | | 26 | INTTAB1CC0[Note 2] | TAB1 capture 0/ compare 0 match | TAB1 | 0220H | 00000220H | nextPC | TAB1CCIC0 |
| | | 27 | INTTAB1CC1 | TAB1 capture 1/ compare 1 match | TAB1 | 0230H | 00000230H | nextPC | TAB1CCIC1 |
| | | 28 | INTTAB1CC2 | TAB1 capture 2/ compare 2 match | TAB1 | 0240H | 00000240H | nextPC | TAB1CCIC2 |
| | | 29 | INTTAB1CC3 | TAB1 capture 3/ compare 3 match | TAB1 | 0250H | 00000250H | nextPC | TAB1CCIC3 |
| | | 30 | INTTT0OV | TMT0 overflow | TMT0 | 0260H | 00000260H | nextPC | TT0OVIC |
| | | 31 | INTTT0CC0 | TMT0 capture 0/ compare 0 match | TMT0 | 0270H | 00000270H | nextPC | TT0CCIC0 |
| | | 32 | INTTT0CC1 | TMT0 capture 1/ compare 1 match | TMT0 | 0280H | 00000280H | nextPC | TT0CCIC1 |
| | | 33 | INTTT0EC | TMT0 encoder input | TMT0 | 0290H | 00000290H | nextPC | TT0ECIC |
| | | 34 | INTTAA0OV | TAA0 overflow | TAA0 | 02A0H | 000002A0H | nextPC | TAA0OVIC |
| | | 35 | INTTAA0CC0 | TAA0 capture 0/ compare 0 match | TAA0 | 02B0H | 000002B0H | nextPC | TAA0CCIC0 |
| | | 36 | INTTAA0CC1 | TAA0 capture 1/ compare 1 match | TAA0 | 02C0H | 000002C0H | nextPC | TAA0CCIC1 |
| | | 37 | INTTAA1OV | TAA1 overflow | TAA1 | 02D0H | 000002D0H | nextPC | TAA1OVIC |

**Notes 1.** When using TAB1 in the 6-phase PWM output mode, functions as the zero match interrupt (TAB1TIOD) request from TMQOP.

    **2.** When using TAB1 in the 6-phase PWM output mode, functions as the compare match interrupt (TAB1TICD0) request from TMQOP.

**Table 23-3.  V850ES/JH3-U Interrupt Sources (3/4)**

| Type | Classification | Default Priority | Name | Trigger | Generating Unit | Exception Code | Handler Address | Restored PC | Interrupt Control Register |
|------|----------------|------------------|------|---------|-----------------|----------------|-----------------|-------------|---------------------------|
| Maskable | Interrupt | 38 | INTTAA1CC0 | TAA1 capture 0/ compare 0 match | TAA1 | 02E0H | 000002E0H | nextPC | TAA1CCIC0 |
| | | 39 | INTTAA1CC1 | TAA1 capture 1/ compare 1 match | TAA1 | 02F0H | 000002F0H | nextPC | TAA1CCIC1 |
| | | 40 | INTTAA2OV | TAA2 overflow | TAA2 | 0300H | 00000300H | nextPC | TAA2OVIC |
| | | 41 | INTTAA2CC0 | TAA2 capture 0/ compare 0 match | TAA2 | 0310H | 00000310H | nextPC | TAA2CCIC0 |
| | | 42 | INTTAA2CC1 | TAA2 capture 1/ compare 1 match | TAA2 | 0320H | 00000320H | nextPC | TAA2CCIC1 |
| | | 43 | INTTAA3OV | TAA3 overflow | TAA3 | 0330H | 00000330H | nextPC | TAA3OVIC |
| | | 44 | INTTAA3CC0 | TAA3 capture 0/ compare 0 match | TAA3 | 0340H | 00000340H | nextPC | TAA3CCIC0 |
| | | 45 | INTTAA3CC1 | TAA3 capture 1/ compare 1 match | TAA3 | 0350H | 00000350H | nextPC | TAA3CCIC1 |
| | | 46 | INTTAA4OV | TAA4 overflow | TAA4 | 0360H | 00000360H | nextPC | TAA4OVIC |
| | | 47 | INTTAA4CC0 | TAA4 compare 0 match | TAA4 | 0370H | 00000370H | nextPC | TAA4CCIC0 |
| | | 48 | INTTAA4CC1 | TAA4 compare 1 match | TAA4 | 0380H | 00000380H | nextPC | TAA4CCIC1 |
| | | 49 | INTTAA5OV | TAA5 overflow | TAA5 | 0390H | 00000390H | nextPC | TAA5OVIC |
| | | 50 | INTTAA5CC0 | TAA5 capture 0/ compare 0 match | TAA5 | 03A0H | 000003A0H | nextPC | TAA5CCIC0 |
| | | 51 | INTTAA5CC1 | TAA5 capture 1/ compare 1 match | TAA5 | 03B0H | 000003B0H | nextPC | TAA5CCIC1 |
| | | 52 | INTTM0EQ0 | TMM0 compare match | TMM0 | 03C0H | 000003C0H | nextPC | TM0EQIC0 |
| | | 53 | INTTM1EQ0 | TMM1 compare match | TMM1 | 03D0H | 000003D0H | nextPC | TM1EQIC0 |
| | | 54 | INTTM2EQ0 | TMM2 compare match | TMM2 | 03E0H | 000003E0H | nextPC | TM2EQIC0 |
| | | 55 | INTTM3EQ0 | TMM3 compare match | TMM3 | 03F0H | 000003F0H | nextPC | TM3EQIC0 |
| | | 56 | INTCF0R/ INTIIC1 | CSIF0 reception completion/ CSIF0 reception error/ IIC1 transfer completion | CSIF0/ IIC1 | 0400H | 00000400H | nextPC | CF0RIC/ IICIC1 |
| | | 57 | INTCF0T | CSIF0 consecutive transmission write enable | CSIF0 | 0410H | 00000410H | nextPC | CF0TIC |
| | | 58 | INTCF1R | CSIF1 reception completion/ CSIF1 reception error | CSIF1 | 0420H | 00000420H | nextPC | CF1RIC |
| | | 59 | INTCF1T | CSIF1 consecutive transmission write enable | CSIF1 | 0430H | 00000430H | nextPC | CF1TIC |
| | | 60 | INTCF2R | CSIF2 reception completion/ CSIF2 reception error | CSIF2 | 0440H | 00000440H | nextPC | CF2RIC |
| | | 61 | INTCF2T | CSIF2 consecutive transmission write enable | CSIF2 | 0450H | 00000450H | nextPC | CF2TIC |
| | | 62 | INTCF3R | CSIF3 reception completion/ CSIF3 reception error | CSIF3 | 0460H | 00000460H | nextPC | CF3RIC |

**Table 23-3.  V850ES/JH3-U Interrupt Sources (4/4)**

| Type | Classification | Default Priority | Name | Trigger | Generating Unit | Exception Code | Handler Address | Restored PC | Interrupt Control Register |
|------|----------------|------------------|------|---------|-----------------|----------------|-----------------|-------------|----------------------------|
| Maskable | Interrupt | 63 | INTCF3T | CSIF3 consecutive transmission write enable | CSIF3 | 0470H | 00000470H | nextPC | CF3TIC |
| | | 64 | INTCF4R | CSIF4 reception completion/ CSIF4 reception error | CSIF4 | 0480H | 00000480H | nextPC | CF4RIC |
| | | 65 | INTCF4T | CSIF4 consecutive transmission write enable | CSIF4 | 0490H | 00000490H | nextPC | CF4TIC |
| | | 66 | INTUC0R | UARTC0 reception completion/UARTC0 reception error | UARTC0 | 04A0H | 000004A0H | nextPC | UC0RIC |
| | | 67 | INTUC0T | UARTC0 consecutive transmission enable | UARTC0 | 04B0H | 000004B0H | nextPC | UC0TIC |
| | | 68 | INTUC1R /INTIIC2 | UARTC1 reception completion/UARTC1 reception error/IIC2 transfer completion | UARTC1/ IIC2 | 04C0H | 000004C0H | nextPC | UC1RIC/ IICIC2 |
| | | 69 | INTUC1T | UARTC1 consecutive transmission enable | UARTC1 | 04D0H | 000004D0H | nextPC | UC1TIC |
| | | 70 | INTUC2R | UARTC2 reception completion/UARTC2 reception error | UARTC2 | 04E0H | 000004E0H | nextPC | UC2RIC |
| | | 71 | INTUC2T | UARTC2 consecutive transmission enable | UARTC2 | 04F0H | 000004F0H | nextPC | UC2TIC |
| | | 72 | INTUC3R/ INTIIC0 | UARTC3 reception completion/UARTC0 reception error/IIC0 transfer completion | UARTC3/ IIC0 | 0500H | 00000500H | nextPC | UC3RIC/ IICIC0 |
| | | 73 | INTUC3T | UARTC3 consecutive transmission enable | UARTC3 | 0510H | 00000510H | nextPC | UC3TIC |
| | | 74 | INTUC4R | UARTC4 reception completion/UARTC4 reception error | UARTC4 | 0520H | 00000520H | nextPC | UC4RIC |
| | | 75 | INTUC4T | UARTC4 consecutive transmission enable | UARTC4 | 0530H | 00000530H | nextPC | UC4TIC |
| | | 76 | INTAD | A/D conversion completion | A/D | 0540H | 00000540H | nextPC | ADIC |
| | | 77 | INTDMA0 | DMA0 transfer completion | DMA | 0550H | 00000550H | nextPC | DMAIC0 |
| | | 78 | INTDMA1 | DMA1 transfer completion | DMA | 0560H | 00000560H | nextPC | DMAIC1 |
| | | 79 | INTDMA2 | DMA2 transfer completion | DMA | 0570H | 00000570H | nextPC | DMAIC2 |
| | | 80 | INTDMA3 | DMA3 transfer completion | DMA | 0580H | 00000580H | nextPC | DMAIC3 |
| | | 81 | INTKR | Key return interrupt | KR | 0590H | 00000590H | nextPC | KRIC |
| | | 82 | INTRTC0 | RTC constant cycle signal | RTC | 05A0H | 000005A0H | nextPC | RTC0IC |
| | | 83 | INTRTC1 | RTC alarm match | RTC | 05B0H | 000005B0H | nextPC | RTC1IC |
| | | 84 | INTRTC2 | RTC interval signal | RTC | 05C0H | 000005C0H | nextPC | RTC2IC |
| | | 89 | INTUSBH0 | USBH status interrupt | USBH | 0610H | 00000610H | nextPC | UHIC0 |
| | | 90 | INTUSBH1 | USBH PCI cycle error | USBH | 0620H | 00000620H | nextPC | UHIC1 |
| | | 91 | INTUSBH2 | USBH PME interrupt | USBH | 0630H | 00000630H | nextPC | UHIC2 |
| | | 92 | INTUSBF0 | USBF interrupt | USBF | 0640H | 00000640H | nextPC | UFIC0 |
| | | 93 | INTUSBF1 | USBF resume interrupt | USBF | 0650H | 00000650H | nextPC | UFIC1 |

**Remarks 1.** Default Priority: The priority order when two or more maskable interrupt requests occur at the same time. The highest priority is 0.

The priority order of non-maskable interrupt is INTWDT2 > NMI.

Restored PC: The value of the program counter (PC) saved to EIPC, FEPC, or DBPC when interrupt servicing is started.  Note, however, that the restored PC when a non-maskable or maskable interrupt is acknowledged while one of the following instructions is being executed does not become the nextPC (if an interrupt is acknowledged during interrupt execution, execution stops, and then resumes after the interrupt servicing has finished).

- Load instructions (SLD.B, SLD.BU, SLD.H, SLD.HU, SLD.W)

- Division instructions (DIV, DIVH, DIVU, DIVHU)

- PREPARE, DISPOSE instructions (only if an interrupt is generated before the stack pointer is updated)

nextPC: The PC value that starts the processing following interrupt/exception processing.

**2.** The execution address of the illegal instruction when an illegal opcode exception occurs is calculated by (Restored PC − 4).

## 23.2 Non-Maskable Interrupts

A non-maskable interrupt request signal is acknowledged unconditionally, even when interrupts are in the interrupt disabled (DI) status. An NMI is not subject to priority control and takes precedence over all the other interrupt request signals.

This product has the following two non-maskable interrupt request signals.

- NMI pin input (NMI)
- Non-maskable interrupt request signal generated by overflow of watchdog timer (INTWDT2)

The valid edge of the NMI pin can be selected from four types: "rising edge", "falling edge", "both edges", and "no edge detection".

The non-maskable interrupt request signal generated by overflow of watchdog timer 2 (INTWDT2) functions when the WDTM2.WDM21 and WDTM2.WDM20 bits are set to "01".

If two or more non-maskable interrupt request signals occur at the same time, the interrupt with the higher priority is serviced, as follows (the interrupt request signal with the lower priority is ignored).

INTWDT2 > NMI

If a new NMI or INTWDT2 request signal is issued while an NMI is being serviced, it is serviced as follows.

**(1) If new NMI request signal is issued while NMI is being serviced**
The new NMI request signal is held pending, regardless of the value of the PSW.NP bit. The pending NMI request signal is acknowledged after the NMI currently under execution has been serviced (after the RETI instruction has been executed).

**(2) If INTWDT2 request signal is issued while NMI is being serviced**
The INTWDT2 request signal is held pending if the NP bit remains set (1) while the NMI is being serviced. The pending INTWDT2 request signal is acknowledged after the NMI currently under execution has been serviced (after the RETI instruction has been executed).

If the NP bit is cleared (0) while the NMI is being serviced, the newly generated INTWDT2 request signal is executed (the NMI servicing is stopped).

**Caution   For the non-maskable interrupt servicing executed by the non-maskable interrupt request signal (INTWDT2), see 23.2.2 (2) From INTWDT2 signal.**

**Figure 23-1. Non-Maskable Interrupt Request Signal Acknowledgment Operation (1/2)**



(a) NMI and INTWDT2 request signals generated at the same time

**Figure 23-1.  Non-Maskable Interrupt Request Signal Acknowledgment Operation (2/2)**

**(b)  Non-maskable interrupt request signal generated during non-maskable interrupt servicing**

| Non-maskable interrupt being serviced | Non-maskable interrupt request signal generated during non-maskable interrupt servicing | |
|---|---|---|
| | NMI | INTWDT2 |
| NMI | • NMI request generated during NMI servicing<br><br>Main routine → NMI servicing<br>NMI request → (Held pending)<br>NMI request → Servicing of pending NMI | • INTWDT2 request generated during NMI servicing (NP bit = 1 retained before INTWDT2 request)<br><br>Main routine → NMI servicing<br>NMI request → INTWDT2 request → (Held pending)<br>INTWDT2 servicing → System reset<br><br>• INTWDT2 request generated during NMI servicing (NP bit = 0 set before INTWDT2 request)<br><br>Main routine → NMI servicing → INTWDT2 servicing<br>NMI request, INTWDT2 request, NP = 0 → System reset<br><br>• INTWDT2 request generated during NMI servicing (NP = 0 set after INTWDT2 request)<br><br>Main routine → NMI servicing (Held pending) → INTWDT2 servicing<br>NMI request, INTWDT2 request, NP = 0 → System reset |
| INTWDT2 | • NMI request generated during INTWDT2 servicing<br><br>Main routine → INTWDT2 servicing<br>INTWDT2 request → NMI request → (Invalid)<br>System reset | • INTWDT2 request generated during INTWDT2 servicing<br><br>Main routine → INTWDT2 servicing<br>INTWDT2 request → INTWDT2 request → (Invalid)<br>System reset |

### 23.2.1  Operation

If a non-maskable interrupt request signal is generated, the CPU performs the following processing, and transfers control to the handler routine.

<1>  Saves the restored PC to FEPC.
<2>  Saves the current PSW to FEPSW.
<3>  Writes exception code (0010H, 0020H) to the higher halfword (FECC) of ECR.
<4>  Sets the PSW.NP and PSW.ID bits to 1 and clears the PSW.EP bit to 0.
<5>  Sets the handler address (00000010H, 00000020H) corresponding to the non-maskable interrupt to the PC, and transfers control.

The servicing configuration of a non-maskable interrupt is shown below.

**Figure 23-2.  Servicing Configuration of Non-Maskable Interrupt**
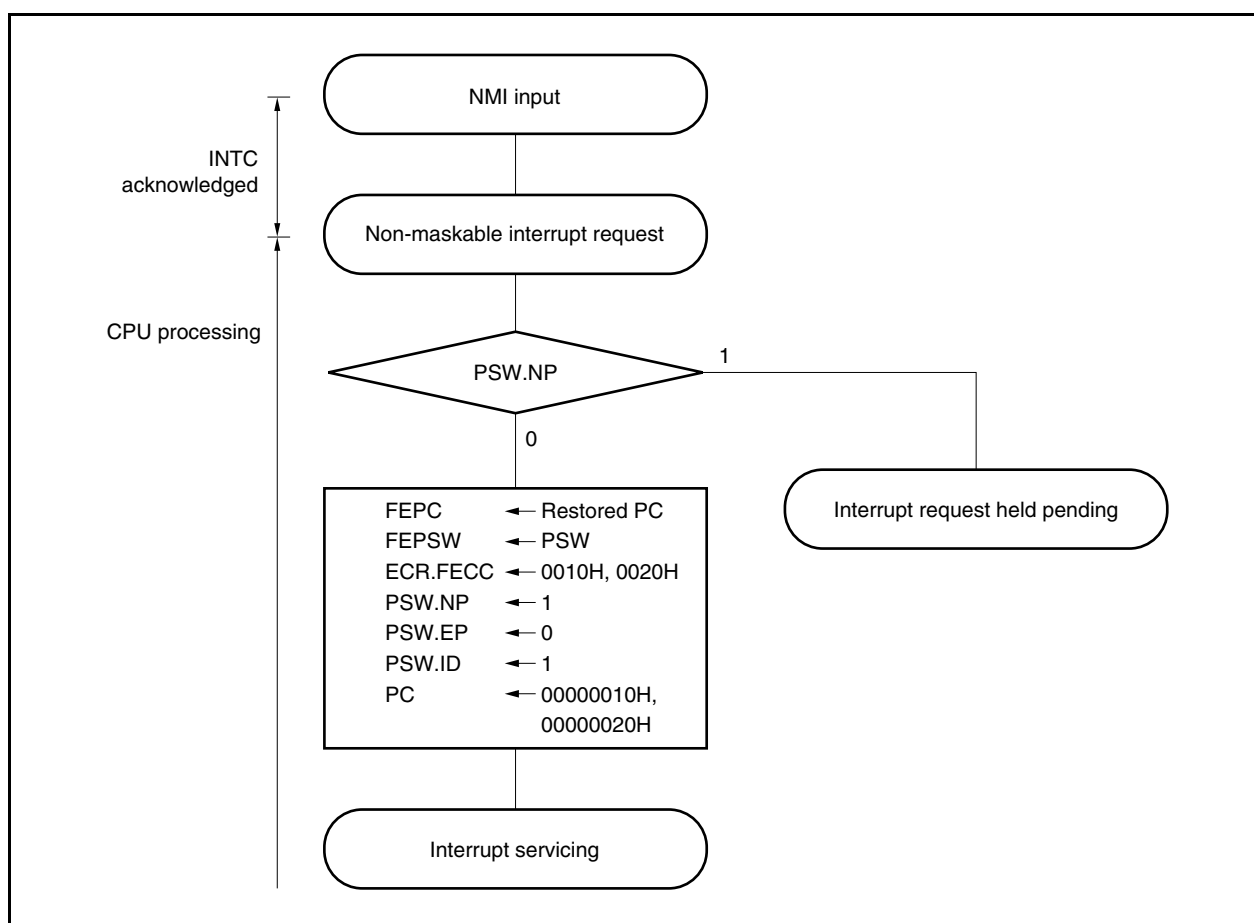
**23.2.2  Restore**

**(1)  From NMI pin input**

Execution is restored from the NMI servicing by the RETI instruction.

When the RETI instruction is executed, the CPU performs the following processing, and transfers control to the address of the restored PC.

<1>  Loads the restored PC and PSW from FEPC and FEPSW, respectively, because the PSW.EP bit is 0 and the PSW.NP bit is 1.

<2>  Transfers control back to the address of the restored PC and PSW.

The processing of the RETI instruction is shown below.

**Figure 23-3.  RETI Instruction Processing**



**Caution    When the EP and NP bits are changed by the LDSR instruction during non-maskable interrupt servicing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set the EP bit back to 0 and the NP bit back to 1 using the LDSR instruction immediately before the RETI instruction.**

**Remark    The solid line shows the CPU processing flow.**
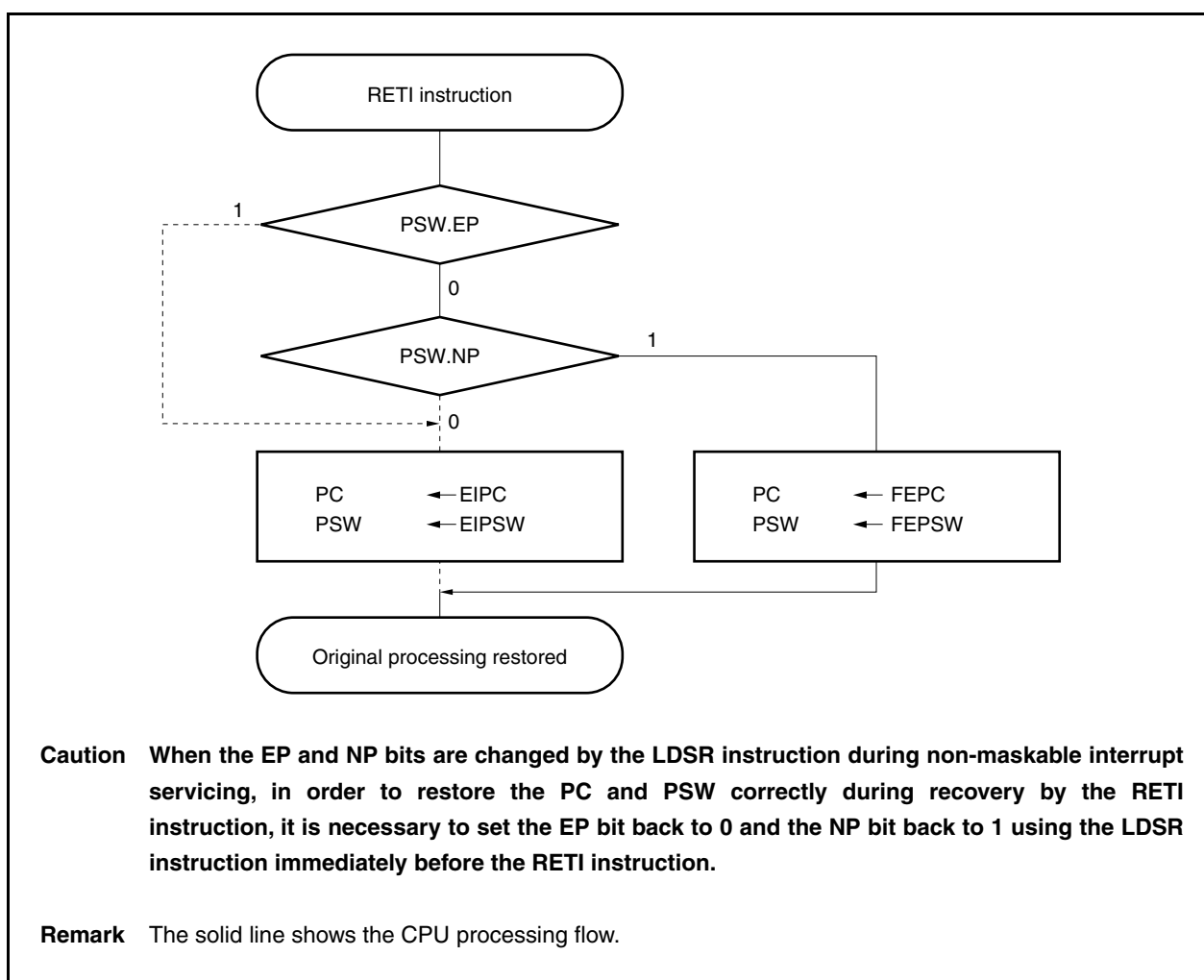
**(2)  From INTWDT2 signal**

Restoring from non-maskable interrupt servicing executed by the non-maskable interrupt request (INTWDT2) by using the RETI instruction is disabled.  Execute the following software reset processing.

**Figure 23-4.  Software Reset Processing**



```
                          ┌──────────────────────┐
                          │   INTWDT2 occurs.    │
                          └──────────────────────┘
                                     │
         ┌───────────────────────────────────────────┐
         │  FEPC ← Software reset processing address  │
         │  FEPSW ← Value that sets NP bit = 1, EP bit = 0 │    INTWDT2 servicing routine
         │                  ↓                         │
         │                RETI                        │
         └───────────────────────────────────────────┘
                                     │
         ┌───────────────────────────────────────────┐
         │  RETI 10 times (FEPC and FEPSW^Note must be set.) │
         │                  ↓                         │
         │  PSW ← PSW default value setting           │    Software reset processing routine
         │                  ↓                         │
         │  Initialization processing                 │
         └───────────────────────────────────────────┘
```
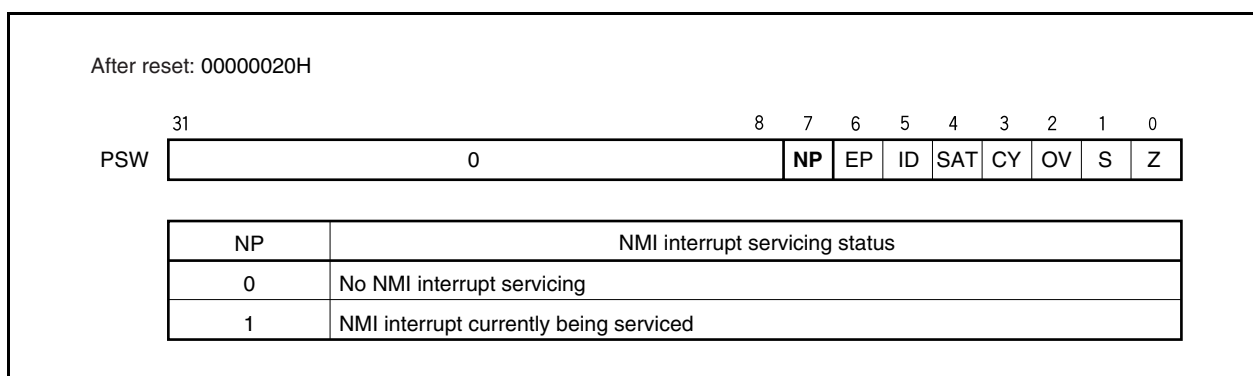
**Note**   FEPSW ← Value that sets NP bit = 1, EP bit = 0
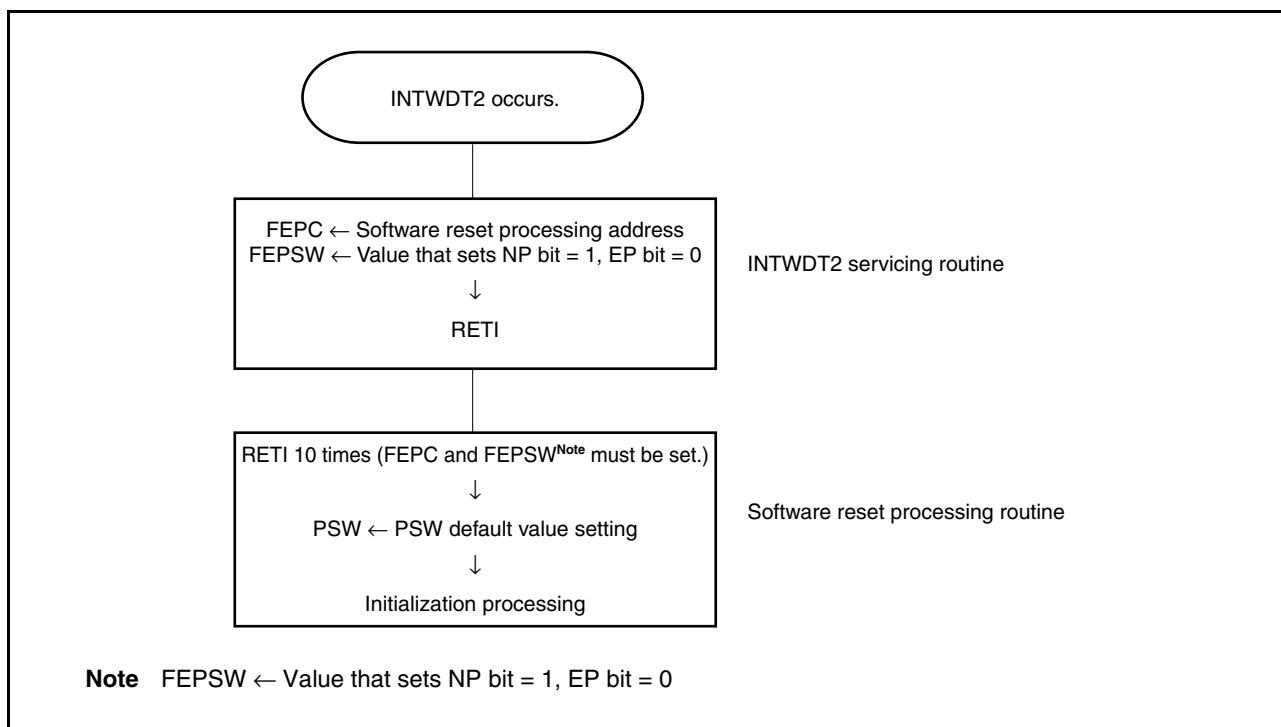
### 23.2.3  NP flag

The NP flag is a status flag that indicates that non-maskable interrupt servicing is under execution.

This flag is set when a non-maskable interrupt request signal has been acknowledged, and masks non-maskable interrupt requests to prohibit multiple interrupts from being acknowledged.

After reset: 00000020H

| PSW | 31                0 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|---------------------|---|---|---|---|---|---|---|---|---|
|     | 0                   |   | **NP** | EP | ID | SAT | CY | OV | S | Z |

| NP | NMI interrupt servicing status |
|----|--------------------------------|
| 0  | No NMI interrupt servicing |
| 1  | NMI interrupt currently being serviced |

## 23.3  Maskable Interrupts

Maskable interrupt request signals can be masked by interrupt control registers.  The V850ES/JG3-U and V850ES/JH3-U have 85 and 90 maskable interrupt sources.

If two or more maskable interrupt request signals are generated at the same time, they are acknowledged according to the default priority.  In addition to the default priority, eight levels of priorities can be specified by using the interrupt control registers (programmable priority control).

When an interrupt request signal has been acknowledged, the acknowledgment of other maskable interrupt request signals is disabled and the interrupt disabled (DI) status is set.

When the EI instruction is executed in an interrupt service routine, the interrupt enabled (EI) status is set, which enables servicing of interrupts having a higher priority than the interrupt request signal in progress (specified by the interrupt control register).  Note that only interrupts with a higher priority will have this capability; interrupts with the same priority level cannot be nested.

To enable multiple interrupts, however, save EIPC and EIPSW to memory or general-purpose registers before executing the EI instruction, and execute the DI instruction before the RETI instruction to restore the original values of EIPC and EIPSW.

### 23.3.1  Operation

If a maskable interrupt occurs, the CPU performs the following processing, and transfers control to a handler routine.

<1>  Saves the restored PC to EIPC.
<2>  Saves the current PSW to EIPSW.
<3>  Writes an exception code to the lower halfword of ECR (EICC).
<4>  Sets the PSW. ID bit to 1 and clears the PSW. EP bit to 0.
<5>  Sets the handler address corresponding to each interrupt to the PC, and transfers control.

The maskable interrupt request signal masked by INTC and the maskable interrupt request signal generated while another interrupt is being serviced (while the PSW.NP bit = 1 or the PSW.ID bit = 1) are held pending inside INTC.  In this case, servicing a new maskable interrupt is started in accordance with the priority of the pending maskable interrupt request signal if either the maskable interrupt is unmasked or the NP and ID bits are set to 0 by using the RETI or LDSR instruction.

How maskable interrupts are serviced is illustrated below.

**Figure 23-5.  Maskable Interrupt Servicing**



Note    For the ISPR register, see **23.3.6  In-service priority register (ISPR)**.
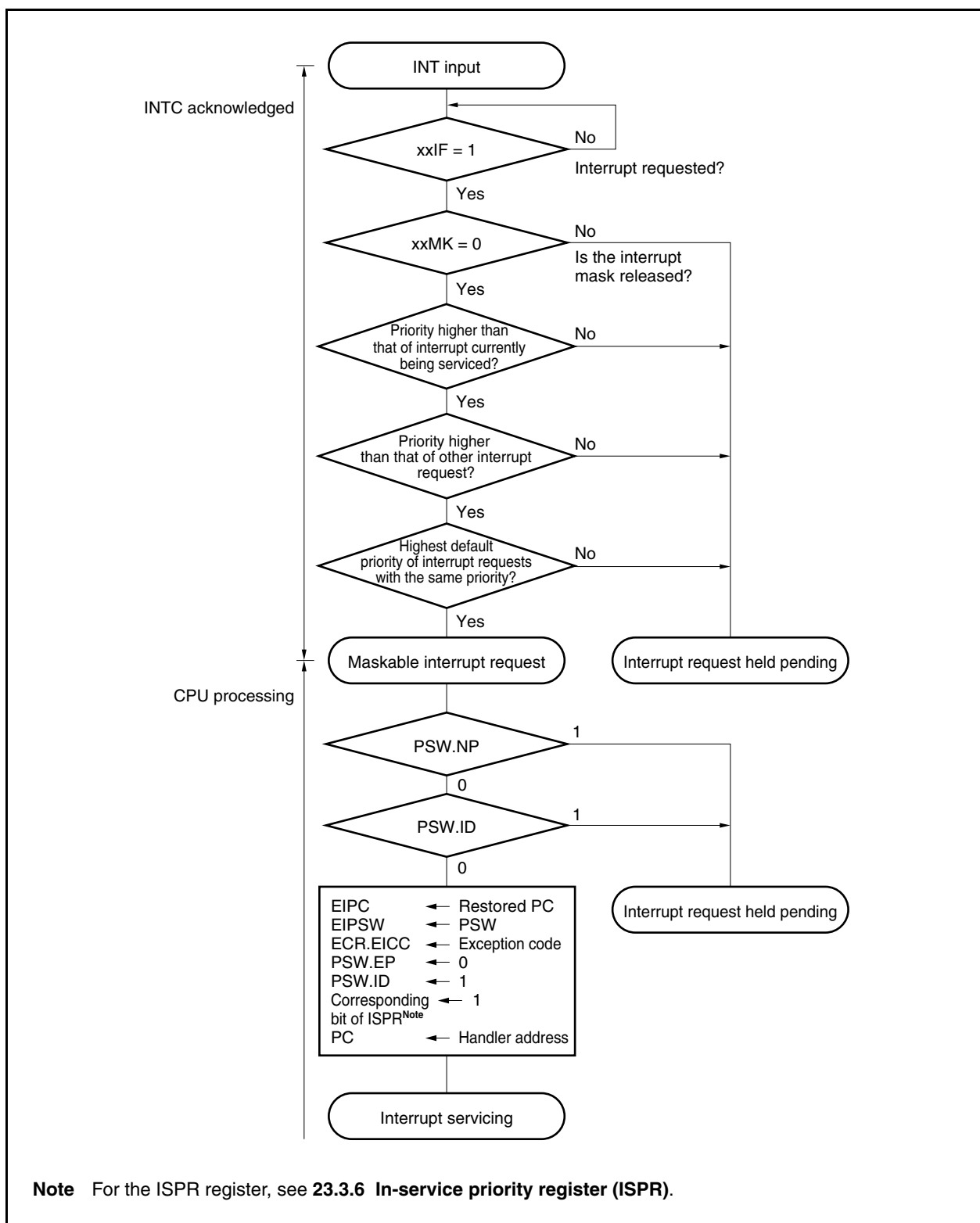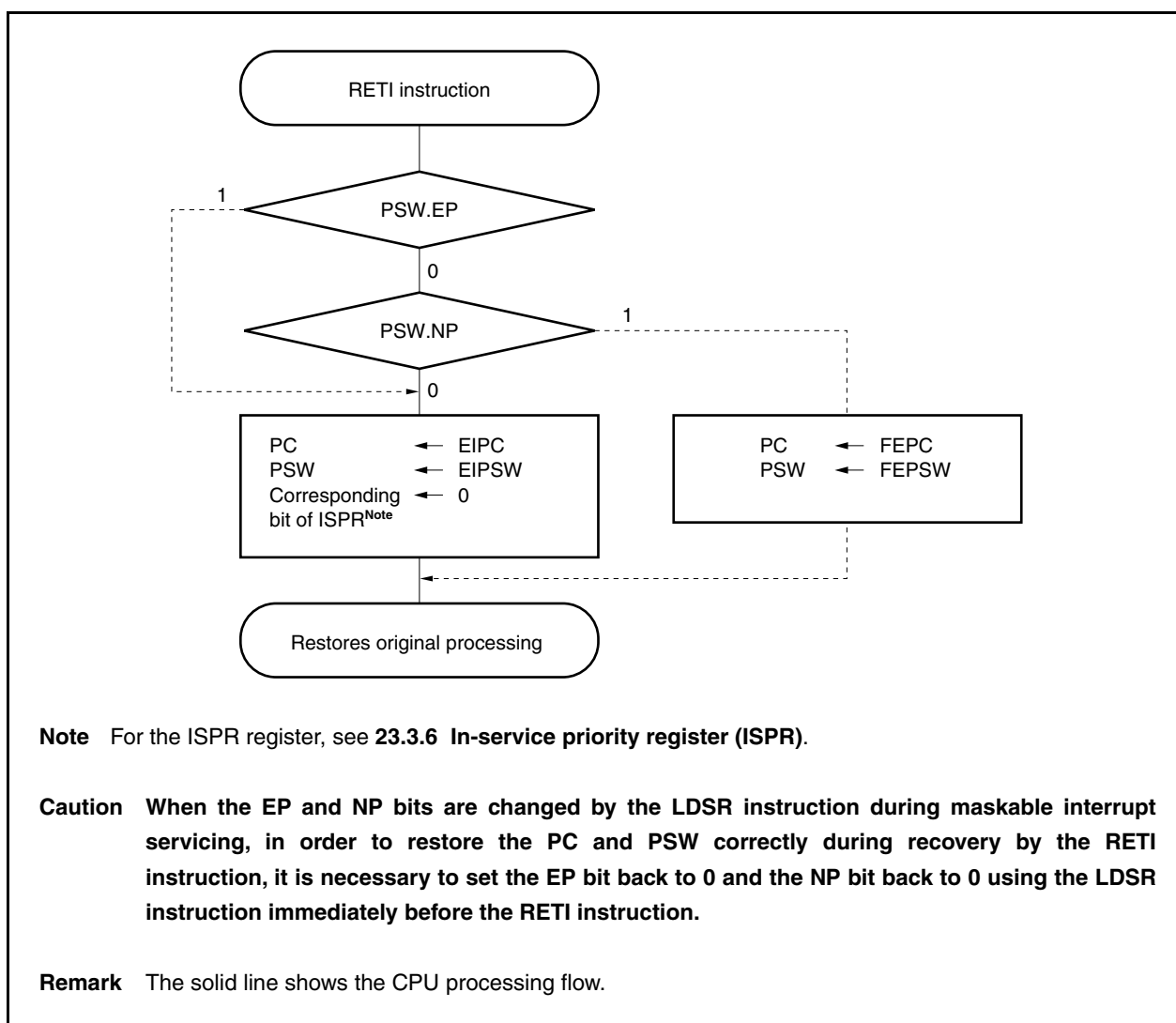
### 23.3.2  Restore

Recovery from maskable interrupt servicing is carried out by the RETI instruction.

When the RETI instruction is executed, the CPU performs the following processing, and transfers control to the address of the restored PC.

<1>  Loads the restored PC and PSW from EIPC and EIPSW because the PSW.EP bit is 0 and the PSW.NP bit is 0.

<2>  Transfers control back to the address of the restored PC and PSW.

The processing of the RETI instruction is shown below.

**Figure 23-6.  RETI Instruction Processing**



**Note**   For the ISPR register, see **23.3.6  In-service priority register (ISPR)**.

**Caution**   **When the EP and NP bits are changed by the LDSR instruction during maskable interrupt servicing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set the EP bit back to 0 and the NP bit back to 0 using the LDSR instruction immediately before the RETI instruction.**

**Remark**   The solid line shows the CPU processing flow.

**23.3.3  Priorities of maskable interrupts**

The INTC performs multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being serviced.  Multiple interrupts can be controlled by priority levels.

There are two types of priority level control:  control based on the default priority levels, and control based on the programmable priority levels that are specified by the interrupt priority level specification bit (xxPRn) of the interrupt control register (xxICn).  When two or more interrupts having the same priority level specified by the xxPRn bit are generated at the same time, interrupt request signals are serviced in order depending on the priority level allocated to each interrupt request type (default priority level) beforehand.  For more information, see **Table 23-2** and **Table 23-3**.  The programmable priority control customizes interrupt request signals into eight levels by setting the priority level specification flag.

Note that when an interrupt request signal is acknowledged, the PSW.ID flag is automatically set to 1.  Therefore, when multiple interrupts are to be used, clear the ID flag to 0 beforehand (for example, by placing the EI instruction in the interrupt service program) to set the interrupt enable mode.

**Remark**  xx: Identification name of each peripheral unit (see **Table 23-4  Interrupt Control Register (xxICn)**)
          n:  Peripheral unit number (see **Table 23-4  Interrupt Control Register (xxICn)**).

**Figure 23-7.  Example of Processing in Which Another Interrupt Request Signal Is Issued
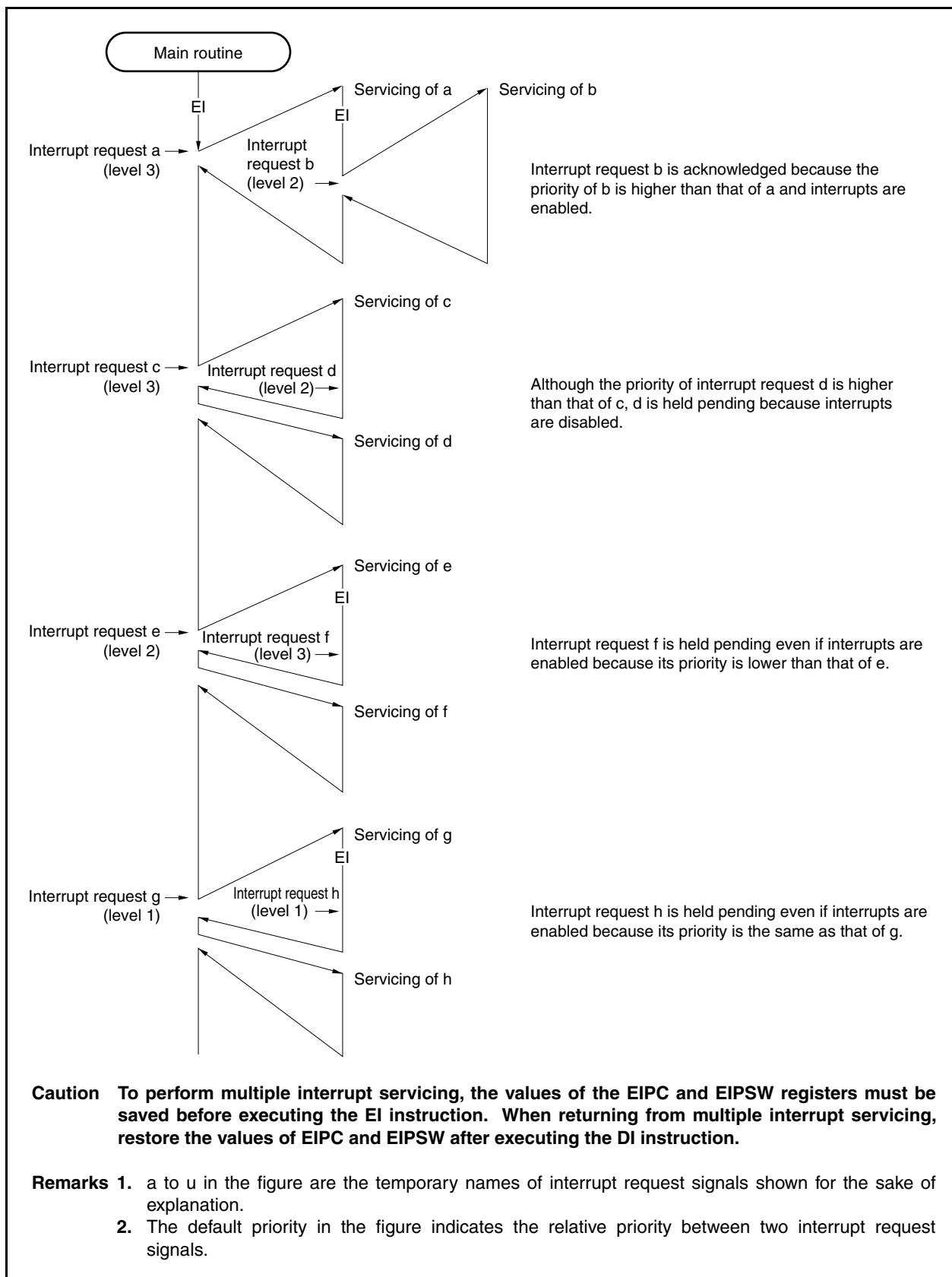While an Interrupt Is Being Serviced (1/2)**



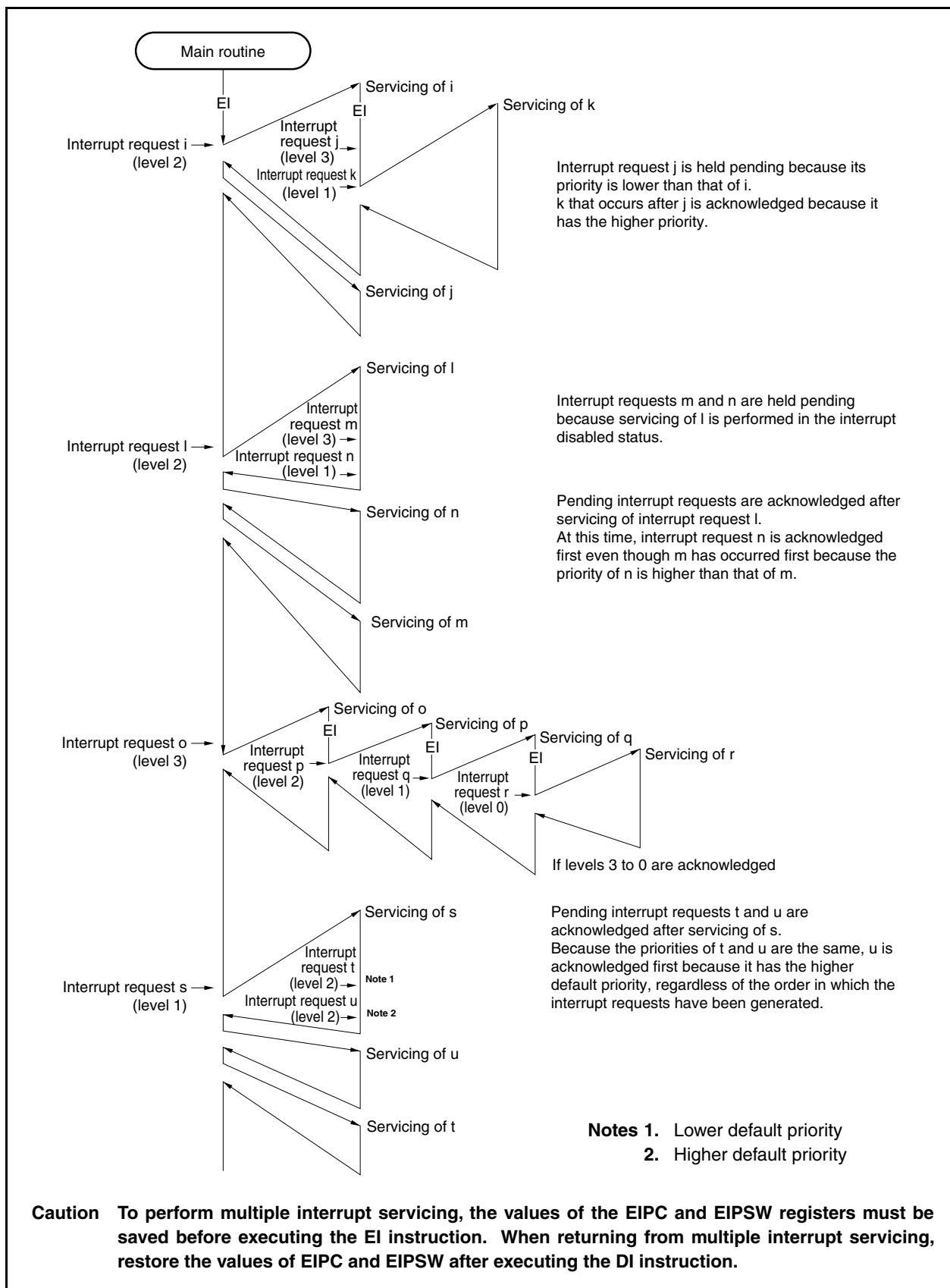**Caution   To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be
saved before executing the EI instruction.  When returning from multiple interrupt servicing,
restore the values of EIPC and EIPSW after executing the DI instruction.**

**Remarks 1.** a to u in the figure are the temporary names of interrupt request signals shown for the sake of
explanation.
**2.** The default priority in the figure indicates the relative priority between two interrupt request
signals.

**Figure 23-7.  Example of Processing in Which Another Interrupt Request Signal Is Issued
While an Interrupt Is Being Serviced (2/2)**



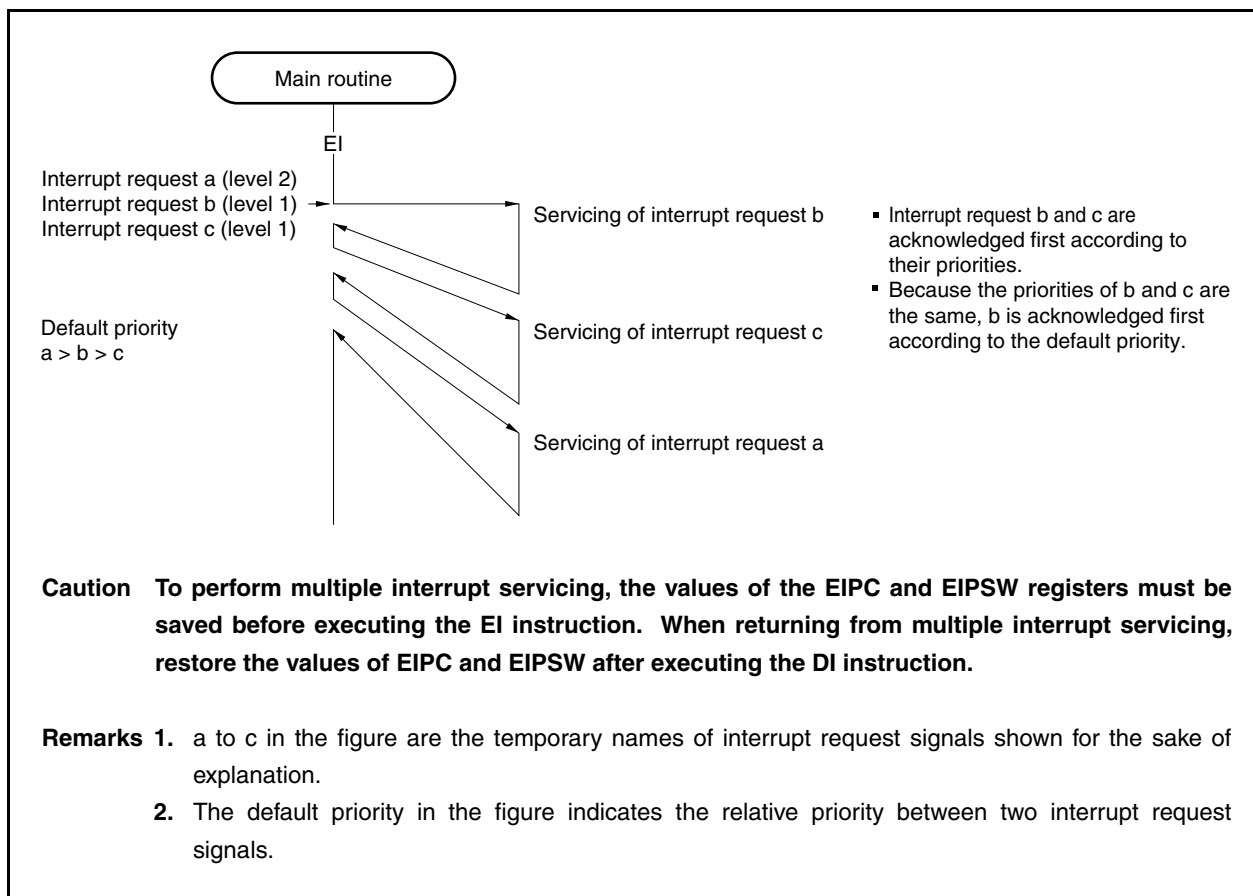Interrupt request j is held pending because its priority is lower than that of i.
k that occurs after j is acknowledged because it has the higher priority.

Interrupt requests m and n are held pending because servicing of l is performed in the interrupt disabled status.

Pending interrupt requests are acknowledged after servicing of interrupt request l.
At this time, interrupt request n is acknowledged first even though m has occurred first because the priority of n is higher than that of m.

If levels 3 to 0 are acknowledged

Pending interrupt requests t and u are acknowledged after servicing of s.
Because the priorities of t and u are the same, u is acknowledged first because it has the higher default priority, regardless of the order in which the interrupt requests have been generated.

**Notes 1.**  Lower default priority
    **2.**  Higher default priority

**Caution   To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction.  When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.**

**Figure 23-8. Example of Servicing Interrupt Request Signals Simultaneously Generated**



**Caution**  **To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.**

**Remarks 1.** a to c in the figure are the temporary names of interrupt request signals shown for the sake of explanation.

        **2.** The default priority in the figure indicates the relative priority between two interrupt request signals.

### 23.3.4  Interrupt control register (xxICn)

The xxICn register is assigned to each interrupt request signal (maskable interrupt) and sets the control conditions for each maskable interrupt request.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 47H.

**Caution   Disable interrupts (DI) or mask the interrupt to read the xxICn.xxIFn bit.  If the xxIFn bit is read while interrupts are enabled (EI) or while the interrupt is unmasked, the correct value may not be read when acknowledging an interrupt and reading the bit conflict.**

After reset: 47H       R/W      Address: FFFFF112H to FFFFF184H

| | <7> | <6> | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| xxICn | xxIFn | xxMKn | 0 | 0 | 0 | xxPRn2 | xxPRn1 | xxPRn0 |

| xxIFn | Interrupt request flag[Note] |
|---|---|
| 0 | Interrupt request not issued |
| 1 | Interrupt request issued |

| xxMKn | Interrupt mask flag |
|---|---|
| 0 | Interrupt servicing enabled |
| 1 | Interrupt servicing disabled (pending) |

| xxPRn2 | xxPRn1 | xxPRn0 | Interrupt priority specification bit |
|---|---|---|---|
| 0 | 0 | 0 | Specifies level 0 (highest). |
| 0 | 0 | 1 | Specifies level 1. |
| 0 | 1 | 0 | Specifies level 2. |
| 0 | 1 | 1 | Specifies level 3. |
| 1 | 0 | 0 | Specifies level 4. |
| 1 | 0 | 1 | Specifies level 5. |
| 1 | 1 | 0 | Specifies level 6. |
| 1 | 1 | 1 | Specifies level 7 (lowest). |

**Note**   The flag xxIFn is reset automatically by the hardware if an interrupt request signal is acknowledged.

**Remark**   xx: Identification name of each peripheral unit (see **Table 23-4  Interrupt Control Register (xxICn)**)
        n:  Peripheral unit number (see **Table 23-4  Interrupt Control Register (xxICn)**).

The addresses and bits of the interrupt control registers are as follows.

**Table 23-4. Interrupt Control Register (xxICn) (1/3)**

| Address | Register | Bit | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | <7> | <6> | 5 | 4 | 3 | 2 | 1 | 0 |
| FFFFF110H | LVIIC | LVIIF | LVIMK | 0 | 0 | 0 | LVIPR2 | LVIPR1 | LVIPR0 |
| FFFFF112H | PIC00[Note] | PIF00 | PMK00 | 0 | 0 | 0 | PPR002 | PPR001 | PPR000 |
| FFFFF114H | PIC01[Note] | PIF01 | PMK01 | 0 | 0 | 0 | PPR012 | PPR011 | PPR010 |
| FFFFF116H | PIC02 | PIF02 | PMK02 | 0 | 0 | 0 | PPR022 | PPR021 | PPR020 |
| FFFFF118H | PIC03[Note] | PIF03 | PMK03 | 0 | 0 | 0 | PPR032 | PPR031 | PPR030 |
| FFFFF11AH | PIC04[Note] | PIF04 | PMK04 | 0 | 0 | 0 | PPR042 | PPR041 | PPR040 |
| FFFFF11CH | PIC05 | PIF05 | PMK05 | 0 | 0 | 0 | PPR052 | PPR051 | PPR050 |
| FFFFF11EH | PIC06[Note] | PIF06 | PMK06 | 0 | 0 | 0 | PPR062 | PPR061 | PPR060 |
| FFFFF120H | PIC07 | PIF07 | PMK07 | 0 | 0 | 0 | PPR072 | PPR071 | PPR070 |
| FFFFF122H | PIC08 | PIF08 | PMK08 | 0 | 0 | 0 | PPR082 | PPR081 | PPR080 |
| FFFFF124H | PIC09 | PIF09 | PMK09 | 0 | 0 | 0 | PPR092 | PPR091 | PPR090 |
| FFFFF126H | PIC10 | PIF10 | PMK10 | 0 | 0 | 0 | PPR102 | PPR101 | PPR100 |
| FFFFF128H | PIC11 | PIF11 | PMK11 | 0 | 0 | 0 | PPR112 | PPR111 | PPR110 |
| FFFFF12AH | PIC12 | PIF12 | PMK12 | 0 | 0 | 0 | PPR122 | PPR121 | PPR120 |
| FFFFF12CH | PIC13 | PIF13 | PMK13 | 0 | 0 | 0 | PPR132 | PPR131 | PPR130 |
| FFFFF12EH | PIC14 | PIF14 | PMK14 | 0 | 0 | 0 | PPR142 | PPR141 | PPR140 |
| FFFFF130H | PIC15 | PIF15 | PMK15 | 0 | 0 | 0 | PPR152 | PPR151 | PPR150 |
| FFFFF132H | PIC16 | PIF16 | PMK16 | 0 | 0 | 0 | PPR162 | PPR161 | PPR160 |
| FFFFF134H | PIC17 | PIF17 | PMK17 | 0 | 0 | 0 | PPR172 | PPR171 | PPR170 |
| FFFFF136H | PIC18 | PIF18 | PMK18 | 0 | 0 | 0 | PPR182 | PPR181 | PPR180 |
| FFFFF138H | TAB0OVIC | TAB0OVIF | TAB0OVMK | 0 | 0 | 0 | TAB0OVPPR2 | TAB0OVPPR1 | TAB0OVPPR0 |
| FFFFF13AH | TAB0CCIC0 | TAB0CCIF0 | TAB0CCMK0 | 0 | 0 | 0 | TAB0CCPPR02 | TAB0CCPPR01 | TAB0CCPPR00 |
| FFFFF13CH | TAB0CCIC1 | TAB0CCIF1 | TAB0CCMK1 | 0 | 0 | 0 | TAB0CCPPR12 | TAB0CCPPR11 | TAB0CCPPR10 |
| FFFFF13EH | TAB0CCIC2 | TAB0CCIF2 | TAB0CCMK2 | 0 | 0 | 0 | TAB0CCPPR22 | TAB0CCPPR21 | TAB0CCPPR20 |
| FFFFF140H | TAB0CCIC3 | TAB0CCIF3 | TAB0CCMK3 | 0 | 0 | 0 | TAB0CCPPR32 | TAB0CCPPR31 | TAB0CCPPR30 |
| FFFFF142H | TAB1OVIC | TAB1OVIF | TAB1OVMK | 0 | 0 | 0 | TAB1OVPPR2 | TAB1OVPPR1 | TAB1OVPPR0 |
| FFFFF144H | TAB1CCIC0 | TAB1CCIF0 | TAB1CCMK0 | 0 | 0 | 0 | TAB1CCPPR02 | TAB1CCPPR01 | TAB1CCPPR00 |
| FFFFF146H | TAB1CCIC1 | TAB1CCIF1 | TAB1CCMK1 | 0 | 0 | 0 | TAB1CCPPR12 | TAB1CCPPR11 | TAB1CCPPR10 |
| FFFFF148H | TAB1CCIC2 | TAB1CCIF2 | TAB1CCMK2 | 0 | 0 | 0 | TAB1CCPPR22 | TAB1CCPPR21 | TAB1CCPPR20 |
| FFFFF14AH | TAB1CCIC3 | TAB1CCIF3 | TAB1CCMK3 | 0 | 0 | 0 | TAB1CCPPR32 | TAB1CCPPR31 | TAB1CCPPR30 |
| FFFFF14CH | TT0OVIC | TT0OVIF | TT0OVMK | 0 | 0 | 0 | TT0OVPPR2 | TT0OVPPR1 | TT0OVPPR0 |
| FFFFF14EH | TT0CCIC0 | TT0CCIF0 | TT0CCMK0 | 0 | 0 | 0 | TT0CCPPR02 | TT0CCPPR01 | TT0CCPPR00 |
| FFFFF150H | TT0CCIC1 | TT0CCIF1 | TT0CCMK1 | 0 | 0 | 0 | TT0CCPPR12 | TT0CCPPR11 | TT0CCPPR10 |
| FFFFF152H | TT0IECIC | TT0IECIF | TT0IECMK | 0 | 0 | 0 | TT0IECPPR2 | TT0IECPPR1 | TT0IECPPR0 |
| FFFFF154H | TAA0OVIC | TAA0OVIF | TAA0OVMK | 0 | 0 | 0 | TAA0OVPPR2 | TAA0OVPPR1 | TAA0OVPPR0 |
| FFFFF156H | TAA0CCIC0 | TAA0CCIF0 | TAA0CCMK0 | 0 | 0 | 0 | TAA0CCPPR02 | TAA0CCPPR01 | TAA0CCPPR00 |
| FFFFF158H | TAA0CCIC1 | TAA0CCIF1 | TAA0CCMK1 | 0 | 0 | 0 | TAA0CCPPR12 | TAA0CCPPR11 | TAA0CCPPR10 |
| FFFFF15AH | TAA1OVIC | TAA1OVIF | TAA1OVMK | 0 | 0 | 0 | TAA1OVPPR2 | TAA1OVPPR1 | TAA1OVPPR0 |
| FFFFF15CH | TAA1CCIC0 | TAA1CCIF0 | TAA1CCMK0 | 0 | 0 | 0 | TAA1CCPPR02 | TAA1CCPPR01 | TAA1CCPPR00 |
| FFFFF15EH | TAA1CCIC1 | TAA1CCIF1 | TAA1CCMK1 | 0 | 0 | 0 | TAA1CCPPR12 | TAA1CCPPR11 | TAA1CCPPR10 |

**Note** V850ES/JH3-U only

**Table 23-4.  Interrupt Control Register (xxICn) (2/3)**

| Address | Register | Bit | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | <7> | <6> | 5 | 4 | 3 | 2 | 1 | 0 |
| FFFFF160H | TAA2OVIC | TAA2OVIF | TAA2OVMK | 0 | 0 | 0 | TAA2OVPPR2 | TAA2OVPPR1 | TAA2OVPPR0 |
| FFFFF162H | TAA2CCIC0 | TAA2CCIF0 | TAA2CCMK0 | 0 | 0 | 0 | TAA2CCPPR02 | TAA2CCPPR01 | TAA2CCPPR00 |
| FFFFF164H | TAA2CCIC1 | TAA2CCIF1 | TAA2CCMK1 | 0 | 0 | 0 | TAA2CCPPR12 | TAA2CCPPR11 | TAA2CCPPR10 |
| FFFFF166H | TAA3OVIC | TAA3OVIF | TAA3OVMK | 0 | 0 | 0 | TAA3OVPPR2 | TAA3OVPPR1 | TAA3OVPPR0 |
| FFFFF168H | TAA3CCIC0 | TAA3CCIF0 | TAA3CCMK0 | 0 | 0 | 0 | TAA3CCPPR02 | TAA3CCPPR01 | TAA3CCPPR00 |
| FFFFF16AH | TAA3CCIC1 | TAA3CCIF1 | TAA3CCMK1 | 0 | 0 | 0 | TAA3CCPPR12 | TAA3CCPPR11 | TAA3CCPPR10 |
| FFFFF16CH | TAA4OVIC | TAA4OVIF | TAA4OVMK | 0 | 0 | 0 | TAA4OVPPR2 | TAA4OVPPR1 | TAA4OVPPR0 |
| FFFFF16EH | TAA4CCIC0 | TAA4CCIF0 | TAA4CCMK0 | 0 | 0 | 0 | TAA4CCPPR02 | TAA4CCPPR01 | TAA4CCPPR00 |
| FFFFF170H | TAA4CCIC1 | TAA4CCIF1 | TAA4CCMK1 | 0 | 0 | 0 | TAA4CCPPR12 | TAA4CCPPR11 | TAA4CCPPR10 |
| FFFFF172H | TAA5OVIC | TAA5OVIF | TAA5OVMK | 0 | 0 | 0 | TAA5OVPPR2 | TAA5OVPPR1 | TAA5OVPPR0 |
| FFFFF174H | TAA5CCIC0 | TAA5CCIF0 | TAA5CCMK0 | 0 | 0 | 0 | TAA5CCPPR02 | TAA5CCPPR01 | TAA5CCPPR00 |
| FFFFF176H | TAA5CCIC1 | TAA5CCIF1 | TAA5CCMK1 | 0 | 0 | 0 | TAA5CCPPR12 | TAA5CCPPR11 | TAA5CCPPR10 |
| FFFFF178H | TM0EQIC0 | TM0EQIF0 | TM0EQMK0 | 0 | 0 | 0 | TM0EQPR02 | TM0EQPR01 | TM0EQPR00 |
| FFFFF17AH | TM1EQIC0 | TM1EQIF0 | TM1EQMK0 | 0 | 0 | 0 | TM1EQPR02 | TM1EQPR01 | TM1EQPR00 |
| FFFFF17CH | TM2EQIC0 | TM2EQIF0 | TM2EQMK0 | 0 | 0 | 0 | TM2EQPR02 | TM2EQPR01 | TM2EQPR00 |
| FFFFF17EH | TM3EQIC0 | TM3EQIF0 | TM3EQMK0 | 0 | 0 | 0 | TM3EQPR02 | TM3EQPR01 | TM3EQPR00 |
| FFFFF180H | CF0RIC/<br>IICIC1 | CF0RIF/<br>IICIF1 | CF0RMK/<br>IICMK1 | 0 | 0 | 0 | CF0RPPR2/<br>IICPPR12 | CF0RPPR1/<br>IICPPR11 | CF0RPPR0/<br>IICPPR10 |
| FFFFF182H | CF0TIC | CF0TIF | CF0TMK | 0 | 0 | 0 | CF0TPPR2 | CF0TPPR1 | CF0TPPR0 |
| FFFFF184H | CF1RIC | CF1RIF | CF1RMK | 0 | 0 | 0 | CF1RPPR2 | CF1RPPR1 | CF1RPPR0 |
| FFFFF186H | CF1TIC | CF1TIF | CF1TMK | 0 | 0 | 0 | CF1TPPR2 | CF1TPPR1 | CF1TPPR0 |
| FFFFF188H | CF2RIC | CF2RIF | CF2RMK | 0 | 0 | 0 | CF2RPPR2 | CF2RPPR1 | CF2RPPR0 |
| FFFFF18AH | CF2TIC | CF2TIF | CF2TMK | 0 | 0 | 0 | CF2TPPR2 | CF2TPPR1 | CF2TPPR0 |
| FFFFF18CH | CF3RIC | CF3RIF | CF3RMK | 0 | 0 | 0 | CF3RPPR2 | CF3RPPR1 | CF3RPPR0 |
| FFFFF18EH | CF3TIC | CF3TIF | CF3TMK | 0 | 0 | 0 | CF3TPPR2 | CF3TPPR1 | CF3TPPR0 |
| FFFFF190H | CF4RIC | CF3RIF | CF3RMK | 0 | 0 | 0 | CF3RPPR2 | CF3RPPR1 | CF3RPPR0 |
| FFFFF192H | CF4TIC | CF3TIF | CF3TMK | 0 | 0 | 0 | CF3TPPR2 | CF3TPPR1 | CF3TPPR0 |
| FFFFF194H | UC0RIC | UC0RIF | UC0RMK | 0 | 0 | 0 | UC0RPPR2 | UC0RPPR1 | UC0RPPR0 |
| FFFFF196H | UC0TIC | UC0TIF | UC0TMK | 0 | 0 | 0 | UC0TPPR2 | UC0TPPR1 | UC0TPPR0 |
| FFFFF198H | UC1RIC/<br>IICIC2 | UC1RIF/<br>IICIF2 | UC1RMK/<br>IICMK2 | 0 | 0 | 0 | UC1RPPR2/<br>IICPPR22 | UC1RPPR1/<br>IICPPR21 | UC1RPPR0/<br>IICPPR20 |
| FFFFF19AH | UC1TIC | UC1TIF | UC1TMK | 0 | 0 | 0 | UC1TPPR2 | UC1TPPR1 | UC1TPPR0 |
| FFFFF19CH | UC2RIC | UC2RIF | UC2RMK | 0 | 0 | 0 | UC2RPPR2 | UC2RPPR1 | UC2RPPR0 |
| FFFFF19EH | UC2TIC | UC2TIF | UC2TMK | 0 | 0 | 0 | UC2TPPR2 | UC2TPPR1 | UC2TPPR0 |
| FFFFF1A0H | UC3RIC/<br>IICIC0 | UC3RIF/<br>IICIF0 | UC3RMK/<br>IICMK0 | 0 | 0 | 0 | UC3RPPR2/<br>IICPPR02 | UC3RPPR1/<br>IICPPR01 | UC3RPPR0/<br>IICPPR00 |
| FFFFF1A2H | UC3TIC | UC3TIF | UC3TMK | 0 | 0 | 0 | UC3TPPR2 | UC3TPPR1 | UC3TPPR0 |
| FFFFF1A4H | UC4RIC | UC4RIF | UC4RMK | 0 | 0 | 0 | UC4RPPR2 | UC4RPPR1 | UC4RPPR0 |
| FFFFF1A6H | UC4TIC | UC4TIF | UC4TMK | 0 | 0 | 0 | UC4TPPR2 | UC4TPPR1 | UC4TPPR0 |
| FFFFF1A8H | ADIC | ADIF | ADMK | 0 | 0 | 0 | ADPPR2 | ADPPR1 | ADPPR0 |
| FFFFF1AAH | DMAIC0 | DMAIC0 | DMAMK0 | 0 | 0 | 0 | DMAPPR02 | DMAPPR01 | DMAPPR00 |
| FFFFF1ACH | DMAIC1 | DMAIC1 | DMAMK1 | 0 | 0 | 0 | DMAPPR12 | DMAPPR11 | DMAPPR10 |

**Table 23-4.  Interrupt Control Register (xxICn) (3/3)**

| Address | Register | Bit | | | | | | | |
|---------|----------|-----|-----|-----|-----|-----|-----|-----|-----|
| | | <7> | <6> | 5 | 4 | 3 | 2 | 1 | 0 |
| FFFFF1AEH | DMAIC2 | DMAIC2 | DMAMK2 | 0 | 0 | 0 | DMAPPR22 | DMAPPR21 | DMAPPR20 |
| FFFFF1B0H | DMAIC3 | DMAIC3 | DMAMK3 | 0 | 0 | 0 | DMAPPR32 | DMAPPR31 | DMAPPR30 |
| FFFFF1B2H | KRIC | KRIF | KRMK | 0 | 0 | 0 | KRPPR2 | KRPPR1 | KRPPR0 |
| FFFFF1B4H | RTC0IC | RTC0IF | RTC0MK | 0 | 0 | 0 | RTC0PPR2 | RTC0PPR1 | RTC0PPR0 |
| FFFFF1B6H | RTC1IC | RTC1IF | RTC1MK | 0 | 0 | 0 | RTC1PPR2 | RTC1PPR1 | RTC1PPR0 |
| FFFFF1B8H | RTC2IC | RTC2IF | RTC2MK | 0 | 0 | 0 | RTC2PPR2 | RTC2PPR1 | RTC2PPR0 |
| FFFFF1C2H | UHIC0 | UHIC0 | UHMK0 | 0 | 0 | 0 | UHPPR02 | UHPPR01 | UHPPR00 |
| FFFFF1C4H | UHIC1 | UHIC1 | UHMK1 | 0 | 0 | 0 | UHPPR12 | UHPPR11 | UHPPR10 |
| FFFFF1C6H | UHIC2 | UHIC2 | UHMK2 | 0 | 0 | 0 | UHPPR22 | UHPPR21 | UHPPR20 |
| FFFFF1C8H | UFIC0 | UFIC0 | UFMK0 | 0 | 0 | 0 | UFPPR02 | UFPPR01 | UFPPR00 |
| FFFFF1CAH | UFIC1 | UFIC1 | UFMK1 | 0 | 0 | 0 | UFPPR12 | UFPPR11 | UFPPR10 |

### 23.3.5  Interrupt mask registers 0 to 5 (IMR0 to IMR5)

The IMR0 to IMR5 registers set the interrupt mask state for the maskable interrupts.  The xxMKn bit of the IMR0 to IMR5 registers is equivalent to the xxICn.xxMKn bit.

The IMRm register can be read or written in 16-bit units.

If the higher 8 bits of the IMRm register are used as an IMRmH register and the lower 8 bits as an IMRmL register, these registers can be read or written in 8-bit or 1-bit units (m = 0 to 5).

Reset sets these registers to FFFFH.

**Caution  The device file defines the xxICn.xxMKn bit as a reserved word.  If a bit is manipulated using the name of xxMKn, the contents of the xxICn register, instead of the IMRm register, are rewritten (as a result, the contents of the IMRm register are also rewritten).**

(1/2)

After reset: FFFFH    R/W    Address: IMR5 FFFFF10AH,
IMR5L FFFFF10AH, IMR5H FFFFF10BH

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| IMR5 (IMR5H**Note**) | 1 | 1 | UFMK1 | UFMK0 | UHMK2 | UHMK1 | UHMK0 | 1 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IMR5L | 1 | 1 | 1 | RTC2MK | RTC1MK | RTC0MK | KRMK | DMAMK3 |

After reset: FFFFH    R/W    Address: IMR4 FFFFF108H,
IMR4L FFFFF108H, IMR4H FFFFF109H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| IMR4 (IMR4H**Note**) | DMAMK2 | DMAMK1 | DMAMK0 | ADMK | UC4TMK | UC4RMK | UC3TMK | UC3RMK/IICMK0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IMR4L | UC2TMK | UC2RMK | UC1TMK | UC1RMK/IICMK2 | UC0TMK | UC0RMK | CF3TMK | CF3RMK |

After reset: FFFFH    R/W    Address: IMR3 FFFFF106H,
IMR3L FFFFF106H, IMR3H FFFFF107H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| IMR3 (IMR3H**Note**) | CF3TMK | CF3RMK | CF2TMK | CF2RMK | CF1TMK | CF1RMK | CF0TMK | CF0RMK/IICMK1 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IMR3L | TM3EQMK0 | TM2EQMK0 | TM1EQMK0 | TM0EQMK0 | TAA5CCMK1 | TAA5CCMK0 | TAA5OVMK | TAA4CCMK1 |

**Note**  To read bits 8 to 15 of the IMR3 to IMR5 registers in 8-bit or 1-bit units, specify them as bits 0 to 7 of IMR3H to IMR5H registers.

**Caution  Set bits 5 to 8, 14, 15 of the IMR5 register to 1.  If the setting of these bits is changed, the operation is not guaranteed.**

**Remark**  xx: Identification name of each peripheral unit (see **Table 23-4  Interrupt Control Register (xxICn)**).
n:  Peripheral unit number (see **Table 23-4  Interrupt Control Register (xxICn)**)

After reset: FFFFH    R/W    Address: IMR2 FFFFF104H,
IMR2L FFFFF104H, IMR2H FFFFF105H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| IMR2 (IMR2H[Note 1]) | TAA4CCMK0 | TAA4OVMK | TAA3CCMK1 | TAA3CCMK0 | TAA3OVMK | TAA2CCMK1 | TAA2CCMK0 | TAA2OVMK |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IMR2L | TAA1CCMK1 | TAA1CCMK0 | TAA1OVMK | TAA0CCMK1 | TAA0CCMK0 | TAA0OVMK | TMTIECMK | TMT0CCMK1 |

After reset: FFFFH    R/W    Address: IMR1 FFFFF102H,
IMR1L FFFFF102H, IMR1H FFFFF103H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| IMR1 (IMR1H[Note 1]) | TT0CCMK0 | TT0OVMK | TAB1CCMK3 | TAB1CCMK2 | TAB1CCMK1 | TAB1CCMK0 | TAB1OVMK | TAB0CCMK3 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IMR1L | TAB0CCMK2 | TAB0CCMK1 | TAB0CCMK0 | TAB0OVMK | PMK18 | PMK17 | PMK16 | PMK15 |

After reset: FFFFH    R/W    Address: IMR0 FFFFF100H,
IMR0L FFFFF100H, IMR0H FFFFF101H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| IMR0 (IMR0H[Note 1]) | PMK14 | PMK13 | PMK12 | PMK11 | PMK10 | PMK09 | PMK08 | PMK07 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IMR0L | PMK06[Note 2] | PMK05 | PMK04[Note 2] | PMK03[Note 2] | PMK02 | PMK01[Note 2] | PMK00[Note 2] | LVIMK |

| xxMKn | Setting of interrupt mask flag |
|---|---|
| 0 | Interrupt servicing enabled |
| 1 | Interrupt servicing disabled |

**Notes 1.** To read bits 8 to 15 of the IMR0 to IMR2 registers in 8-bit or 1-bit units, specify them as bits 0 to 7 of IMR0H to IMR2H registers.

**2.** V850ES/JH3-U only

**Remark** xx: Identification name of each peripheral unit (see **Table 23-4   Interrupt Control Register (xxICn)**).

n:  Peripheral unit number (see **Table 23-4  Interrupt Control Register (xxICn)**)

#### 23.3.6  In-service priority register (ISPR)

The ISPR register holds the priority level of the maskable interrupt currently acknowledged.  When an interrupt request signal is acknowledged, the bit of this register corresponding to the priority level of that interrupt request signal is set to 1 and remains set while the interrupt is serviced.

When the RETI instruction is executed, the bit corresponding to the interrupt request signal having the highest priority is automatically reset to 0 by hardware.  However, it is not reset to 0 when execution is returned from non-maskable interrupt servicing or exception processing.

This register is read-only, in 8-bit or 1-bit units.

Reset sets this register to 00H.

**Caution  If an interrupt is acknowledged while the ISPR register is being read in the interrupt enabled (EI) status, the value of the ISPR register after the bits of the register have been set by acknowledging the interrupt may be read.  To accurately read the value of the ISPR register before an interrupt is acknowledged, read the register while interrupts are disabled (DI).**

After reset: 00H    R    Address: FFFFF1FAH

|      | <7>   | <6>   | <5>   | <4>   | <3>   | <2>   | <1>   | <0>   |
|------|-------|-------|-------|-------|-------|-------|-------|-------|
| ISPR | ISPR7 | ISPR6 | ISPR5 | ISPR4 | ISPR3 | ISPR2 | ISPR1 | ISPR0 |

| ISPRn | Priority of interrupt currently acknowledged |
|-------|----------------------------------------------|
| 0 | Interrupt request signal with priority n not acknowledged |
| 1 | Interrupt request signal with priority n acknowledged |

**Remark**   n = 0 to 7 (priority level)

### 23.3.7 ID flag

This flag controls the maskable interrupt's operating state, and stores control information regarding enabling or disabling of interrupt request signals. An interrupt disable flag (ID) is assigned to the PSW.

Reset sets this flag to 00000020H.

After reset: 00000020H

| 31 | | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| PSW | 0 | | NP | EP | **ID** | SAT | CY | OV | S | Z |

| ID | Specification of maskable interrupt servicing**Note** |
|---|---|
| 0 | Maskable interrupt request signal acknowledgment enabled |
| 1 | Maskable interrupt request signal acknowledgment disabled |

**Note** Interrupt disable flag (ID) function

This bit is set to 1 by the DI instruction and cleared to 0 by the EI instruction. Its value is also modified by the RETI instruction or LDSR instruction when referencing the PSW.

Non-maskable interrupt request signals and exceptions are acknowledged regardless of this flag. When a maskable interrupt request signal is acknowledged, the ID flag is automatically set to 1 by hardware.

The interrupt request signal generated during the acknowledgment disabled period (ID flag = 1) is acknowledged when the xxICn.xxIFn bit is set to 1, and the ID flag is cleared to 0.

### 23.3.8 Watchdog timer mode register 2 (WDTM2)

This register can be read or written in 8-bit units (for details, see **CHAPTER 13 FUNCTIONS OF WATCHDOG TIMER 2**).

Reset sets this register to 67H.

After reset: 67H    R/W    Address: FFFFF6D0H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| WDTM2 | 0 | WDM21 | WDM20 | 0 | 0 | 0 | 0 | 0 |

| WDM21 | WDM20 | Selection of watchdog timer operation mode |
|---|---|---|
| 0 | 0 | Stops operation |
| 0 | 1 | Non-maskable interrupt request mode |
| 1 | × | Reset mode (initial value) |

## 23.4  Software Exception

A software exception is generated when the CPU executes the TRAP instruction, and can always be acknowledged.

### 23.4.1  Operation

If a software exception occurs, the CPU performs the following processing, and transfers control to the handler routine.

<1>  Saves the restored PC to EIPC.

<2>  Saves the current PSW to EIPSW.

<3>  Writes an exception code to the lower 16 bits (EICC) of ECR (interrupt source).

<4>  Sets the PSW.EP and PSW.ID bits to 1.

<5>  Sets the handler address (00000040H or 00000050H) corresponding to the software exception to the PC, and transfers control.

The processing of a software exception is shown below.

**Figure 23-9.  Software Exception Processing**



**Note**  TRAP instruction format: TRAP vector (the vector is a value from 00H to 1FH.)
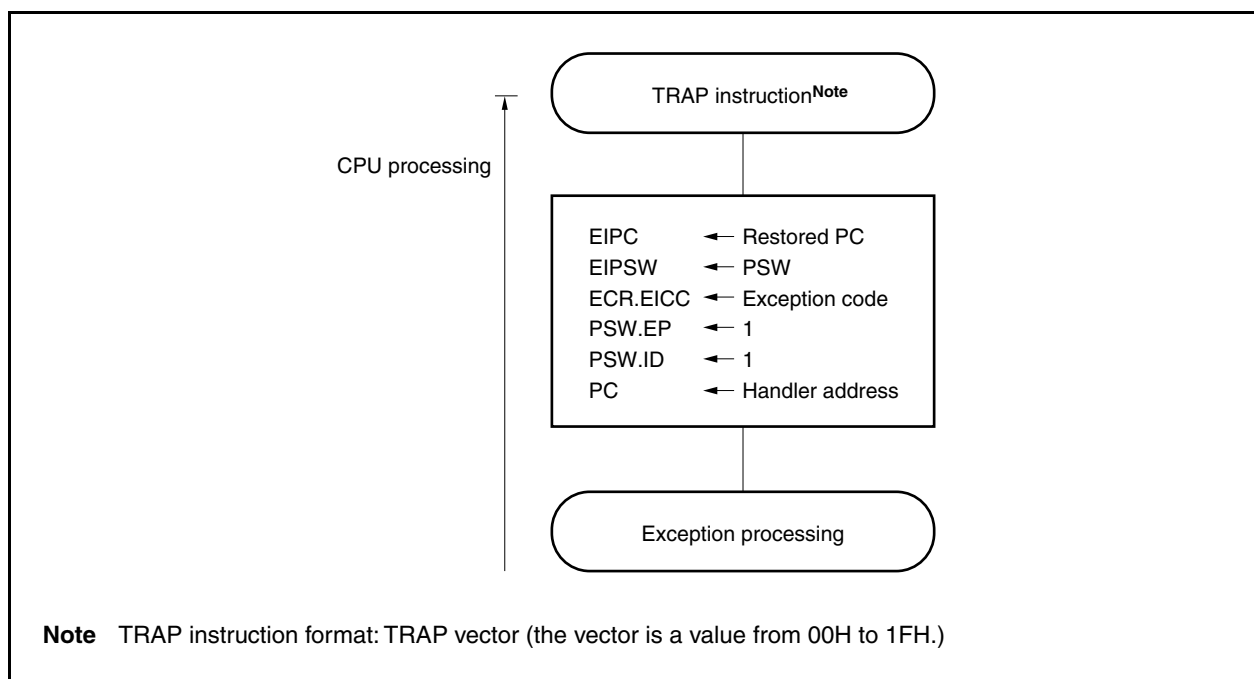
The handler address is determined by the TRAP instruction's operand (vector).  If the vector is 00H to 0FH, it becomes 00000040H, and if the vector is 10H to 1FH, it becomes 00000050H.

**23.4.2  Restore**

Restoration from software exception processing is carried out by the RETI instruction.

By executing the RETI instruction, the CPU carries out the following processing and shifts control to the restored PC's address.

<1>  Loads the restored PC and PSW from EIPC and EIPSW because the PSW.EP bit is 1.

<2>  Transfers control to the address of the restored PC and PSW.

The processing of the RETI instruction is shown below.

**Figure 23-10.  RETI Instruction Processing**



**Caution**  **When the EP and NP bits are changed by the LDSR instruction during the software exception processing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set the EP bit back to 1 and the NP bit back to 0 using the LDSR instruction immediately before the RETI instruction.**

**Remark**  The solid line shows the CPU processing flow.

### 23.4.3  EP flag

The EP flag is a status flag used to indicate that exception processing is in progress.  It is set when an exception occurs.

After reset: 00000020H

| | 31 | | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PSW | | 0 | | NP | **EP** | ID | SAT | CY | OV | S | Z |

| EP | Exception processing status |
|---|---|
| 0 | Exception processing not in progress. |
| 1 | Exception processing in progress. |
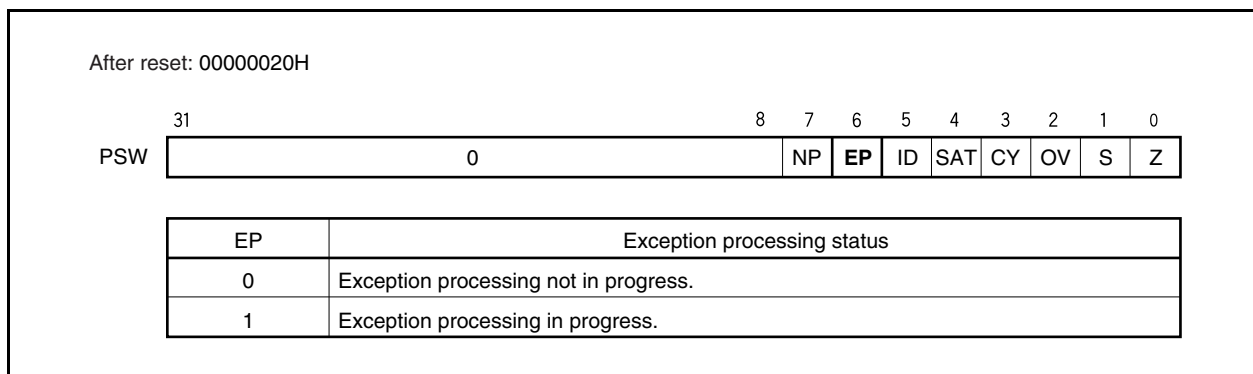
## 23.5  Exception Trap

An exception trap is an interrupt that is requested when the illegal execution of an instruction takes place.  In the V850ES/JG3-U and V850ES/JH3-U, an illegal opcode exception (ILGOP: Illegal Opcode Trap) is considered as an exception trap.

### 23.5.1  Illegal opcode

An illegal opcode is defined as an instruction with instruction opcode (bits 10 to 5) = 111111B, sub-opcode (bits 26 to 23) = 0111B to 1111B, and sub-opcode (bit 16) = 0B.  When such an instruction is executed, an exception trap is generated.

| 15 | 11 10 | 5 4 | 0 31 | 27 26 | 23 22 | 16 |
|----|-------|-----|------|-------|-------|----|
| × × × × × | 1 1 1 1 1 1 | × × × × × | × × × × × | 0 1 1 1 to 1 1 1 1 | × × × × × × | 0 |

×: Arbitrary
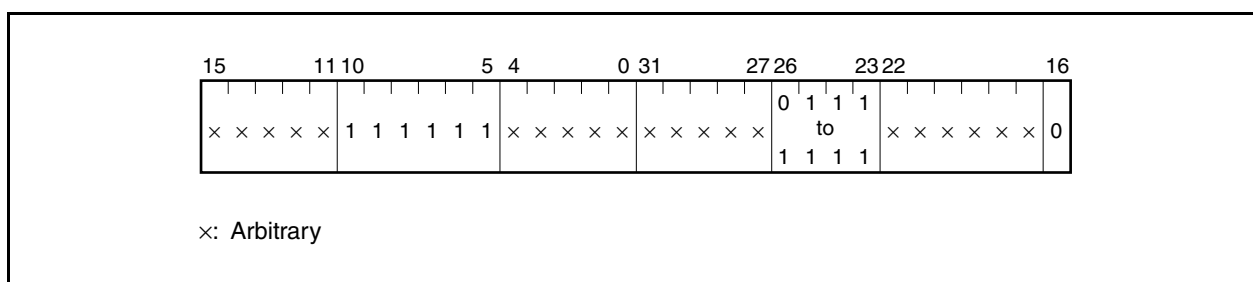
**Caution    It is recommended not to use an illegal opcode because instructions may newly be assigned in the future.**

### (1)  Operation

If an exception trap occurs, the CPU performs the following processing, and transfers control to the handler routine.

<1>  Saves the restored PC to DBPC.
<2>  Saves the current PSW to DBPSW.
<3>  Sets the PSW.NP, PSW.EP, and PSW.ID bits to 1.
<4>  Sets the handler address (00000060H) corresponding to the exception trap to the PC, and transfers control.

The processing of the exception trap is shown below.

**Figure 23-11.  Exception Trap Processing**
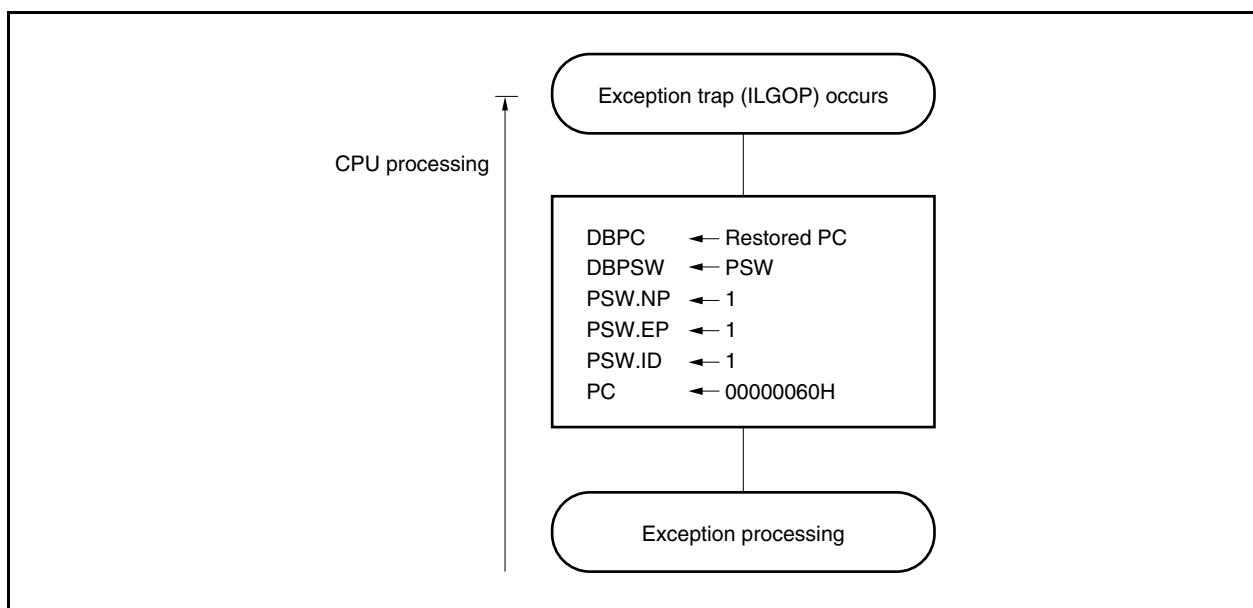


**(2)  Restoration**
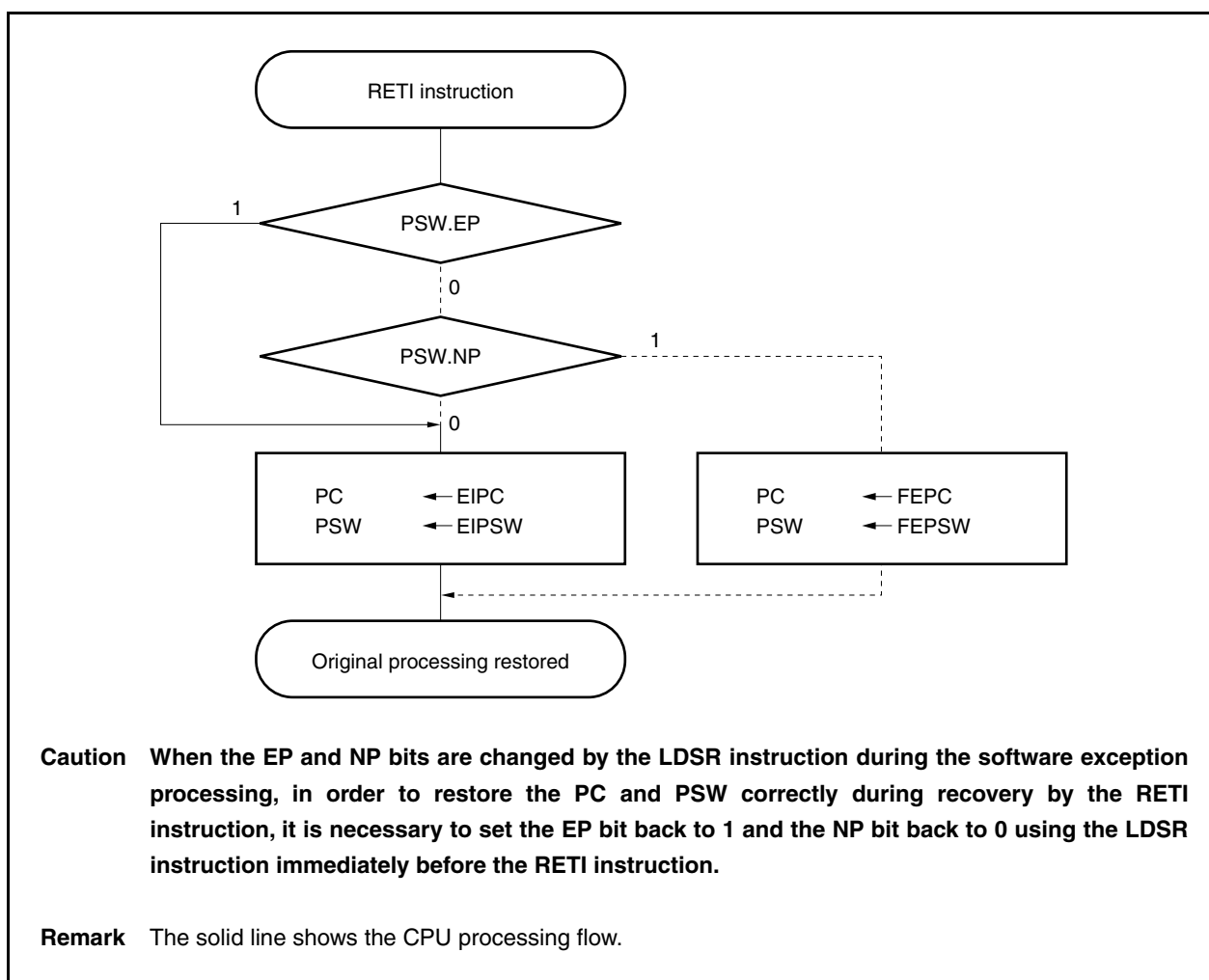
Restoration from an exception trap is carried out by the DBRET instruction.  By executing the DBRET instruction, the CPU carries out the following processing and controls the address of the restored PC.

<1> Loads the restored PC and PSW from DBPC and DBPSW.
<2> Transfers control to the address indicated by the restored PC and PSW.

**Caution   DBPC and DBPSW can be accessed only during the interval between the execution of an illegal opcode and DBRET instruction.**

Processing for restoring from an exception trap is shown below.

**Figure 23-12.  Processing for Restoring from Exception Trap**
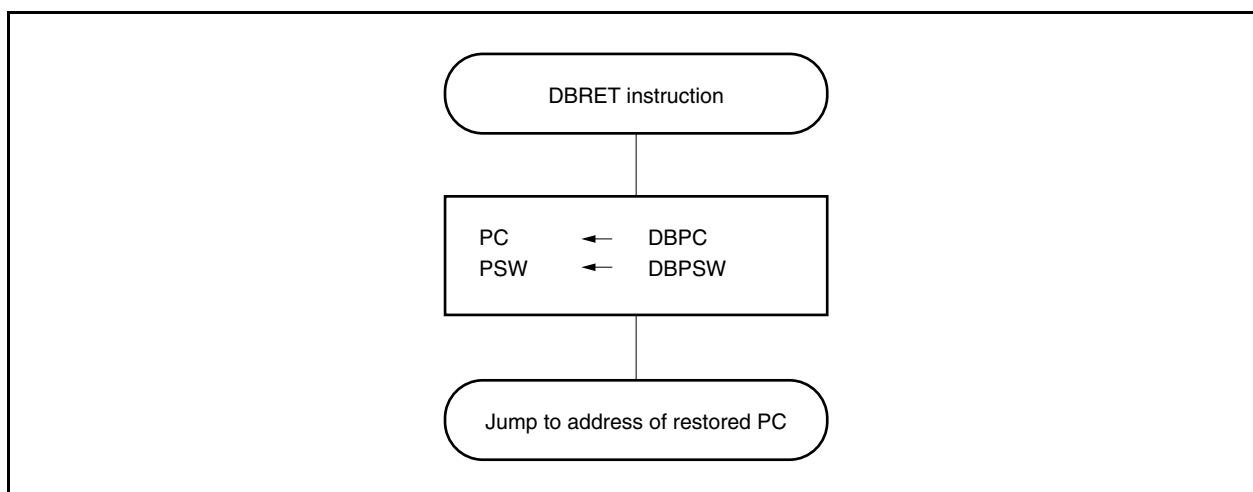
### 23.5.2  Debug trap

A debug trap is an exception that is generated when the DBTRAP instruction is executed and is always acknowledged.

#### (1)  Operation

Upon occurrence of a debug trap, the CPU performs the following processing.

<1> Saves restored PC to DBPC.

<2> Saves current PSW to DBPSW.

<3> Sets the PSW.NP, PSW.EP, and PSW.ID bits to 1.

<4> Sets handler address (00000060H) for debug trap to PC and transfers control.

The debug trap processing format is shown below.

**Figure 23-13.  Debug Trap Processing Format**
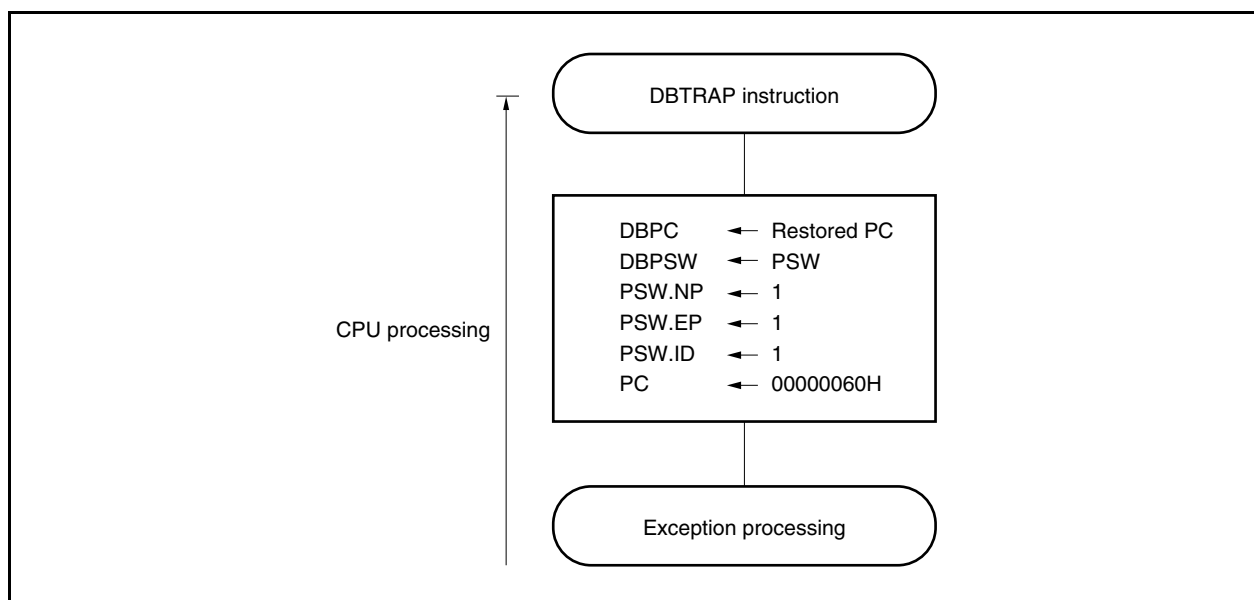
**(2) Restoration**

Restoration from a debug trap is executed with the DBRET instruction.

With the DBRET instruction, the CPU performs the following steps and transfers control to the address of the restored PC.

<1> The restored PC and PSW are read from DBPC and DBPSW.

<2> Control is transferred to the fetched address of the restored PC and PSW.

**Caution   DBPC and DBPSW can be accessed only during the interval between the execution of the DBTRAP instruction and DBRET instruction.**

The processing format for restoration from a debug trap is shown below.

**Figure 23-14.  Processing Format of Restoration from Debug Trap**
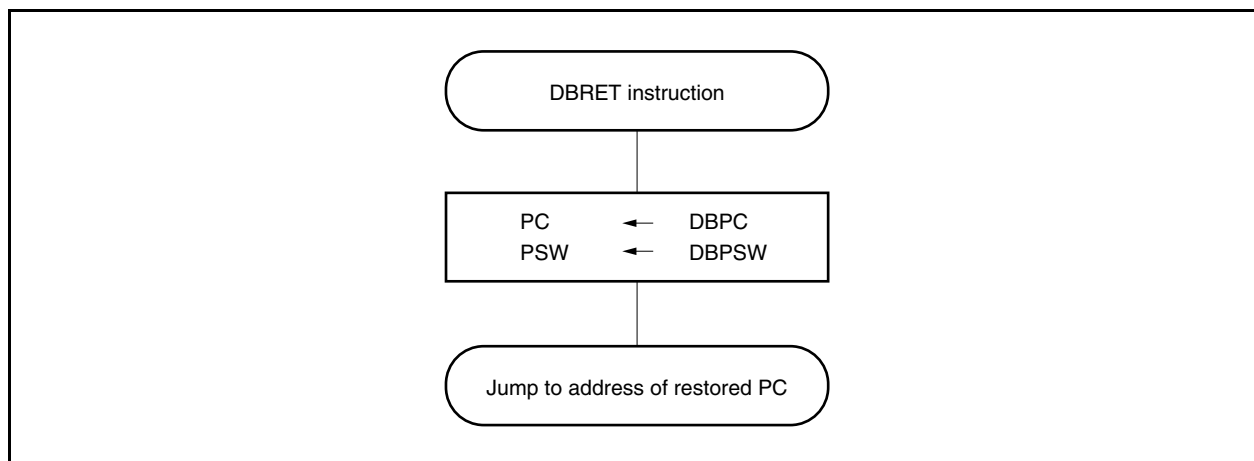
## 23.6  External Interrupt Request Input Pins (NMI and INTP00 to INTP18)

### 23.6.1  Noise elimination

**(1)  Eliminating noise on NMI pin**
The NMI pin has an internal noise elimination circuit that uses analog delay.  Therefore, the input level of the NMI pin is not detected as an edge unless it is maintained for a specific time or longer.  Therefore, an edge is detected after specific time.
The NMI pin can be used to release the STOP mode.  In the STOP mode, noise is not eliminated by using the system clock because the internal system clock is stopped.

**(2)  Eliminating noise on INTP00, INTP01, and INTP03 to INTP18 pins**
The INTP00, INTP01, and INTP03 to INTP18 pins have an internal noise elimination circuit that uses analog delay.  Therefore, the input level of the NMI pin is not detected as an edge unless it is maintained for a specific time or longer.  Therefore, an edge is detected after specific time.

**(3)  Eliminating noise on INTP02**
The INTP02 pin has an internal noise elimination circuit that uses analog delay and an internal digital noise elimination circuit.  Either can be selected by using the noise elimination control register (INTNFC) (see **23.6.2 (7)**).

### 23.6.2  Edge detection
The valid edge of each of the NMI and INTP00 to INTP18 pins can be selected from the following four.

- Rising edge
- Falling edge
- Both rising and falling edges
- No edge detected

The edge of the NMI pin is not detected after reset.  Therefore, the interrupt request signal is not acknowledged unless a valid edge is enabled by using the INTF0 and INTR0 register (the NMI pin functions as a normal port pin).

**(1) External interrupt falling, rising edge specification register 0 (INTF0, INTR0)**

The INTF0 and INTR0 registers are 8-bit registers that specify detection of the falling and rising edges of the NMI pin via bit 2 and the external interrupt pins (INTP00 to INTP04) via bits 0, 1, 3 to 5.

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

**Caution** **When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, set the INTF0n and INTR0n bits to 00, and then set the port mode.**

---

After reset: 00H    R/W    Address: INTF0 FFFFFC00H, INTR0 FFFFFC20H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTF0 | 0 | 0 | INTF05[Note] | INTF04[Note] | INTF03 | INTF02 | INTF01[Note] | INTF00[Note] |
| | | | INTP04 | INTP03 | INTP02 | NMI | INTP01 | INTP00 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTR0 | 0 | 0 | INTR05[Note] | INTR04[Note] | INTR03 | INTR02 | INTR01[Note] | INTR00[Note] |
| | | | INTP04 | INTP03 | INTP02 | NMI | INTP01 | INTP00 |

**Note** V850ES/JH3-U only

**Remark** For how to specify a valid edge, see **Table 23-5**.

---

**Table 23-5. Valid Edge Specification**

| INTF0n | INTR0n | Valid Edge Specification (n = 0 to 5) |
|---|---|---|
| 0 | 0 | No edge detected |
| 0 | 1 | Rising edge |
| 1 | 0 | Falling edge |
| 1 | 1 | Both rising and falling edges |

**Caution** **Be sure to set the INTF0n and INTR0n bits to 00 when these registers are not used as the NMI or INTP00 to INTP04 pins.**

**Remark** n = 0, 1:    Control of INTP00 and INTP01 pins
n = 2:       Control of NMI pin
n = 3 to 5: Control of INTP02 to INTP04 pins

**(2) External interrupt falling, rising edge specification register 2 (INTF2, INTR2) (V850ES/JH3-U only)**

The INTF2 and INTR2 registers are 8-bit registers that specify detection of the falling and rising edges of the external interrupt pins (INTP05 to INTP06).

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

**Caution   When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected.  Therefore, set the INTF2n and INTR2n bits to 00, and then set the port mode.**

After reset:  00H     R/W     Address:  INTF2 FFFFFC04H, INTR2 FFFFFC24H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTF2 | 0 | 0 | INTF25 | INTF24 | 0 | 0 | 0 | 0 |
| | | | INTP06 | INTP05 | | | | |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTR2 | 0 | 0 | INTR25 | INTR24 | 0 | 0 | 0 | 0 |
| | | | INTP06 | INTP05 | | | | |

**Remark**   For how to specify a valid edge, see **Table 23-6**.

**Table 23-6.  Valid Edge Specification**

| INTF2n | INTR2n | Valid Edge Specification (n = 4, 5) |
|---|---|---|
| 0 | 0 | No edge detected |
| 0 | 1 | Rising edge |
| 1 | 0 | Falling edge |
| 1 | 1 | Both rising and falling edges |

**Caution   Be sure to set the INTF2n and INTR2n bits to 00 when these registers are not used as the INTP05 and INTP06 pins.**

**Remark**   n = 4, 5: Control of INTP05 and INTP06 pins

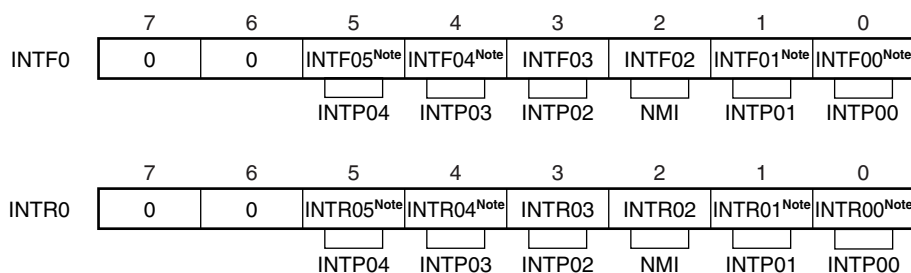**(3) External interrupt falling, rising edge specification register 3 (INTF3, INTR3)**

The INTF3 and INTR3 registers are 8-bit registers that specify detection of the falling and rising edges of the external interrupt pin (INTP07 to INTP09).

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

**Caution   When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected.  Therefore, set the INTF3n and INTR3n bits to 00, and then set the port mode.**

After reset: 00H     R/W     Address:  INTF3 FFFFFC06H, INTR3 FFFFFC26H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTF3 | 0 | 0 | 0 | INTF34 | 0 | 0 | INTF31 | INTF30 |
| | | | | INTP09 | | | INTP08 | INTP07 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTR3 | 0 | 0 | 0 | INTR34 | 0 | 0 | INTR31 | INTR30 |
| | | | | INTP09 | | | INTP08 | INTP07 |

**Remark**   For how to specify a valid edge, see **Table 23-7**.

**Table 23-7.  Valid Edge Specification**

| INTF3n | INTR3n | Valid Edge Specification |
|---|---|---|
| 0 | 0 | No edge detected |
| 0 | 1 | Rising edge |
| 1 | 0 | Falling edge |
| 1 | 1 | Both rising and falling edges |

**Caution   Be sure to set the INTF3n and INTR3n bits to 00 when these registers are not used as the INTP07 to INTP09 pin.**

**Remark**   n = 0, 1, 4: Control of INTP07 to INTP09 pins

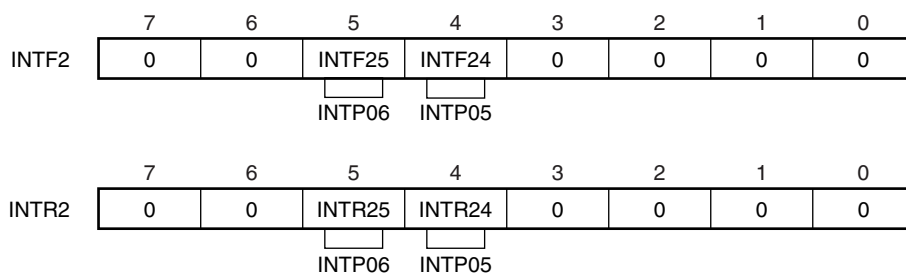**(4) External interrupt falling, rising edge specification registers 4 (INTF4, INTR4)**

The INTF4 and INTR4 registers are 8-bit registers that specify detection of the falling and rising edges of the external interrupt pin (INTP10).

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

**Caution** **When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected.  Therefore, set the INTF42 and INTR42 bits to 00, and then set the port mode.**

After reset:  00H    R/W    Address:  INTF4 FFFFFC08H, INTR4 FFFFFC28H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTF4 | 0 | 0 | 0 | 0 | 0 | INTF42 | 0 | 0 |

INTP10

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTR4 | 0 | 0 | 0 | 0 | 0 | INTR42 | 0 | 0 |

INTP10

**Remark**   For the valid edge specification combinations, see **Table 23-8**.

**Table 23-8.  Valid Edge Specification**

| INTF42 | INTR42 | Valid Edge Specification |
|---|---|---|
| 0 | 0 | No edge detected |
| 0 | 1 | Rising edge |
| 1 | 0 | Falling edge |
| 1 | 1 | Both rising and falling edges |

**Caution** **Be sure to set the INTF42 and INTR42 bits to 00 if the corresponding pin is not used as the INTP10 pin.**

**(5) External interrupt falling, rising edge specification registers 5 (INTF5, INTR5) (V850ES/JG3-U only)**

The INTF5 and INTR5 registers are 8-bit registers that specify detection of the falling and rising edges of the external interrupt pin (INTP05).

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

**Caution**   **When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected.  Therefore, set the INTF56 and INTR56 bits to 00, and then set the port mode.**
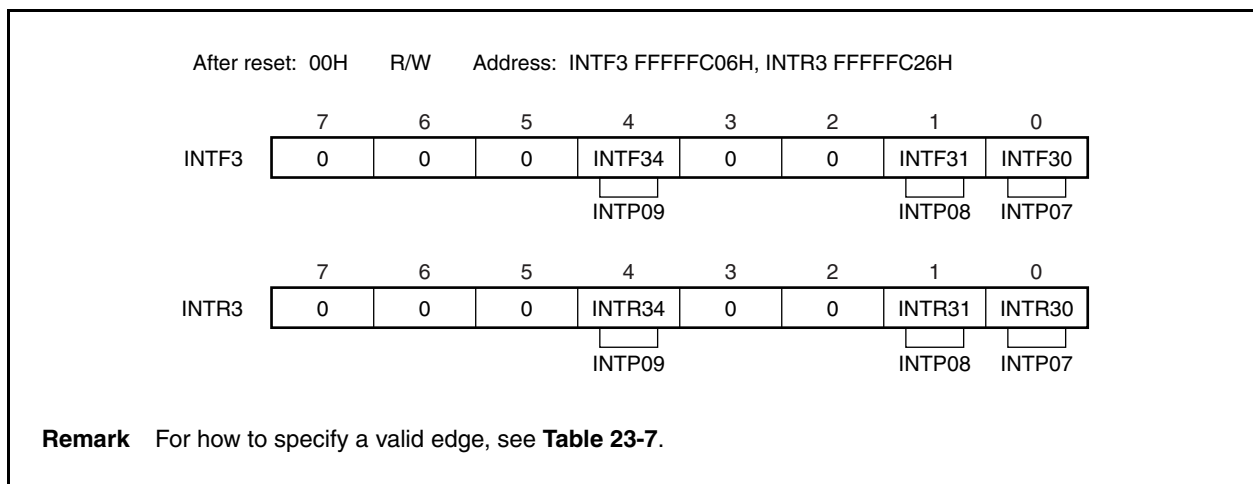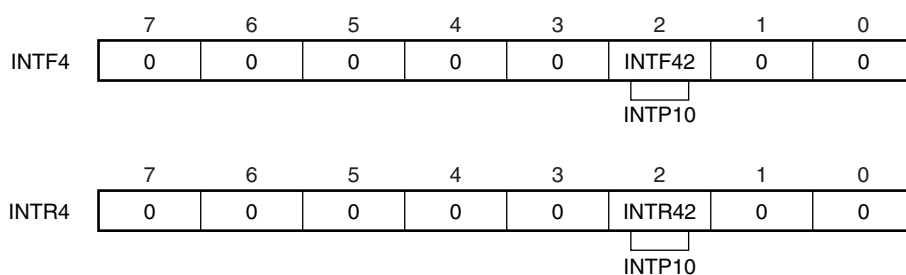
After reset: 00H      R/W      Address:  INTF5 FFFFFC0AH, INTR5 FFFFFC2AH

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTF5 | 0 | INTF56 | 0 | 0 | 0 | 0 | 0 | 0 |

INTP05

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTR5 | 0 | INTR56 | 0 | 0 | 0 | 0 | 0 | 0 |

INTP05

**Remark**   For the valid edge specification combinations, see **Table 23-9**.

**Table 23-9.  Valid Edge Specification**

| INTF56 | INTR56 | Valid Edge Specification |
|--------|--------|--------------------------|
| 0 | 0 | No edge detected |
| 0 | 1 | Rising edge |
| 1 | 0 | Falling edge |
| 1 | 1 | Both rising and falling edges |

**Caution**   **Be sure to set the INTF56 and INTR56 bits to 00 if the corresponding pin is not used as the INTP05 pin.**

**(6) External interrupt falling, rising edge specification register 9H (INTF9H, INTR9H)**
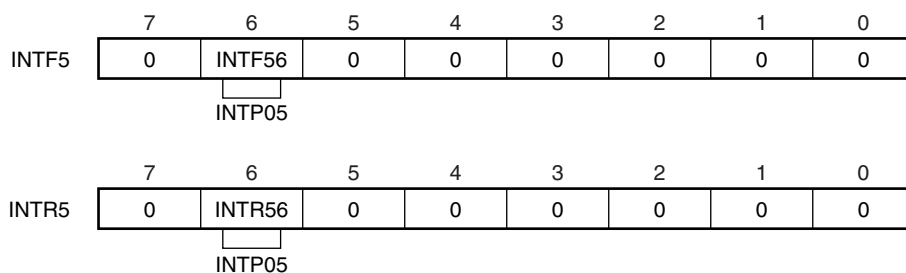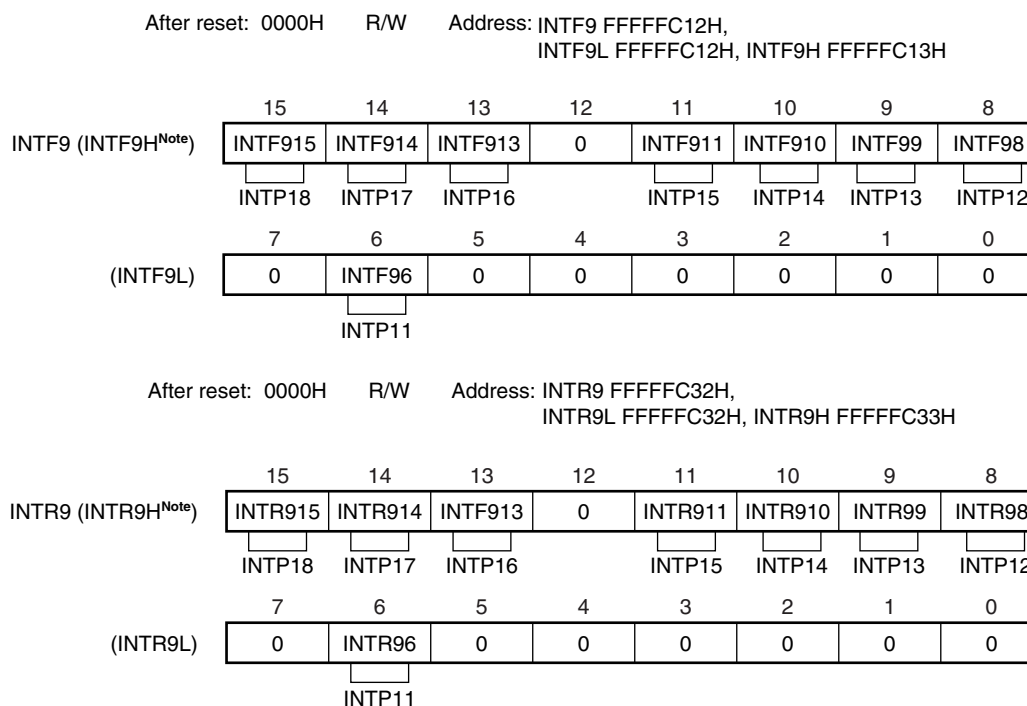
The INTF9H and INTR9H registers are 16-bit or 8-bit registers that specify detection of the falling and rising edges of the external interrupt pins (INTP11 to INTP18).

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 0000H/00H.

**Caution   When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected.  Therefore, clear the INTF9n and INTR9n bits to 0, and then set the port mode.**



After reset:  0000H     R/W     Address: INTF9 FFFFFC12H,
                                          INTF9L FFFFFC12H, INTF9H FFFFFC13H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| INTF9 (INTF9H**Note**) | INTF915 | INTF914 | INTF913 | 0 | INTF911 | INTF910 | INTF99 | INTF98 |
| | INTP18 | INTP17 | INTP16 | | INTP15 | INTP14 | INTP13 | INTP12 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (INTF9L) | 0 | INTF96 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | INTP11 | | | | | | |

After reset:  0000H     R/W     Address: INTR9 FFFFFC32H,
                                          INTR9L FFFFFC32H, INTR9H FFFFFC33H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| INTR9 (INTR9H**Note**) | INTR915 | INTR914 | INTF913 | 0 | INTR911 | INTR910 | INTR99 | INTR98 |
| | INTP18 | INTP17 | INTP16 | | INTP15 | INTP14 | INTP13 | INTP12 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (INTR9L) | 0 | INTR96 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | INTP11 | | | | | | |

**Note**   To read bits 8 to 15 of the INTF9 and INTR9 registers in 8-bit or 1-bit units, specify them as bits 0 to 7 of the INTF9H and INTR9H registers.

**Remark**   For how to specify a valid edge, see **Table 23-10**.

**Table 23-10.  Valid Edge Specification**

| INTF9n | INTR9n | Valid Edge Specification (n = 11 to 18) |
|---|---|---|
| 0 | 0 | No edge detected |
| 0 | 1 | Rising edge |
| 1 | 0 | Falling edge |
| 1 | 1 | Both rising and falling edges |

**Caution   Be sure to clear the INTF9n and INTR9n bits to 00 when these registers are not used as the INTP11 to INTP18 pins.**

**Remark**   n = 6, 8 to 11, 13 to 15: Control of INTP11 to INTP18 pins

**(7) Noise elimination control register (INTNFC)**

Analog noise elimination and digital noise elimination can be selected for the INTP02 pin. The noise elimination settings are performed using the INTNFC register.

When analog noise elimination is selected, the input level of the pin is detected as an edge by maintaining it for a specific time or longer.

When digital noise elimination is selected, the sampling clock for digital sampling can be selected from among $f_{XX}/64$, $f_{XX}/128$, $f_{XX}/256$, $f_{XX}/512$, $f_{XX}/1,024$, and $f_{XT}$. Sampling is performed 3 times.

Even when digital noise elimination is selected, using $f_{XT}$ as the sampling clock makes it possible to use the INTP02 interrupt request signal to release the IDLE1, IDLE2, and STOP modes.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

**Caution  After the sampling clock has been changed, it takes 3 sampling clocks to initialize the digital noise eliminator.  Therefore, if an INTP02 valid edge is input within these 3 sampling clocks after the sampling clock has been changed, an interrupt request signal may be generated.  Therefore, be careful about the following points when using the interrupt and DMA functions.**

- **When using the interrupt function, after the 3 sampling clocks have elapsed, enable interrupts after the interrupt request flag (PIC2.PIF2 bit) has been cleared.**
- **When using the DMA function (started by INTP02), enable DMA after 3 sampling clocks have elapsed.**

After reset:  00H     R/W     Address:  FFFFF728H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| INTNFC | INTNFEN | 0 | 0 | 0 | 0 | INTNFC2 | INTNFC1 | INTNFC0 |

| INTNFEN | Settings of INTP02 pin noise elimination |
|---|---|
| 0 | Analog noise elimination (60 ns (TYP.)) |
| 1 | Digital noise elimination |

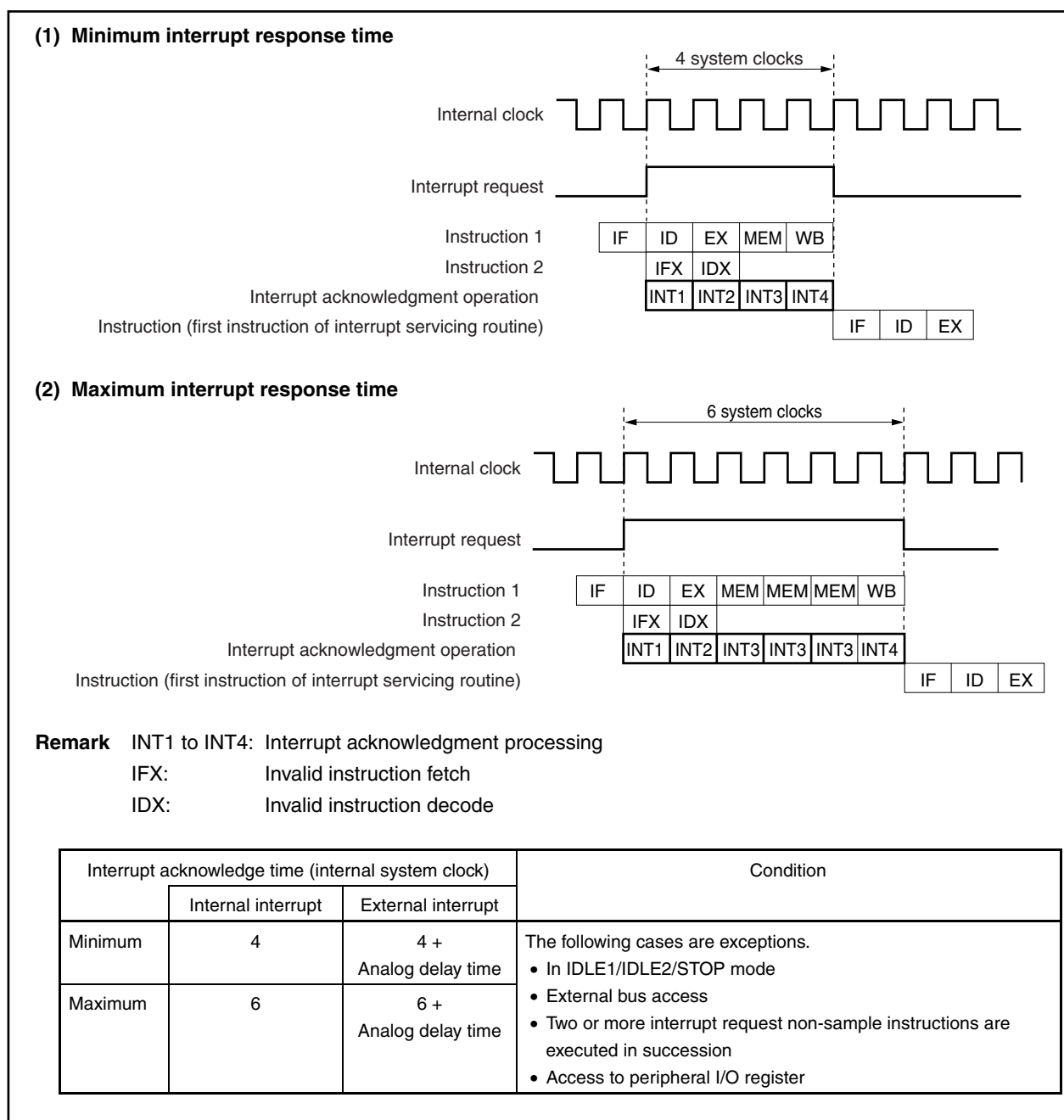| INTNFC2 | INTNFC1 | INTNFC0 | Digital sampling clock |
|---|---|---|---|
| 0 | 0 | 0 | $f_{XX}/64$ |
| 0 | 0 | 1 | $f_{XX}/128$ |
| 0 | 1 | 0 | $f_{XX}/256$ |
| 0 | 1 | 1 | $f_{XX}/512$ |
| 1 | 0 | 0 | $f_{XX}/1,024$ |
| 1 | 0 | 1 | $f_{XT}$ (subclock) |
| Other than above | | | Setting prohibited |

**Remarks 1.**  Since sampling is performed 3 times, the reliably eliminated noise width is 2 sampling clocks.
**2.**  In the case of noise with a width smaller than 2 sampling clocks, an interrupt request signal is generated if noise synchronized with the sampling clock is input.

## 23.7  Interrupt Acknowledge Time of CPU

Except the following cases, the interrupt acknowledge time of the CPU is 4 clocks minimum.  To input interrupt request signals successively, input the next interrupt request signal at least 5 clocks after the preceding interrupt.

- In IDLE1/IDLE2/STOP mode
- When the external bus is accessed
- When interrupt request non-sampling instructions are successively executed (see **23.8  Periods in Which Interrupts Are Not Acknowledged by CPU**.)
- When the interrupt control register is accessed

**Figure 23-15.  Pipeline Operation at Interrupt Request Signal Acknowledgment (Outline)**



**Remark**   INT1 to INT4:  Interrupt acknowledgment processing
         IFX:          Invalid instruction fetch
         IDX:          Invalid instruction decode

| Interrupt acknowledge time (internal system clock) | | | Condition |
|---|---|---|---|
| | Internal interrupt | External interrupt | |
| Minimum | 4 | 4 + Analog delay time | The following cases are exceptions. • In IDLE1/IDLE2/STOP mode • External bus access • Two or more interrupt request non-sample instructions are executed in succession • Access to peripheral I/O register |
| Maximum | 6 | 6 + Analog delay time | |

## 23.8  Periods in Which Interrupts Are Not Acknowledged by CPU

An interrupt is acknowledged by the CPU while an instruction is being executed.  However, no interrupt will be acknowledged between an interrupt request non-sample instruction and the next instruction (interrupt is held pending).

The interrupt request non-sample instructions are as follows.


- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (for PSW)
- The store instruction for the PRCMD register
- The store, SET1, NOT1, or CLR1 instructions for the following registers.
  - Interrupt-related registers:
    Interrupt control register (xxICn), interrupt mask registers 0 to 5 (IMR0 to IMR5)
  - Power save control register (PSC)
  - On-chip debug mode register (OCDM)


**Remark**  xx: Identification name of each peripheral unit (see **Table 23-4  Interrupt Control Register (xxICn)**)
        n: Peripheral unit number (see **Table 23-4  Interrupt Control Register (xxICn)**).

## 23.9  Cautions

The NMI pin alternately functions as the P02 pin, and functions as a normal port pin after being reset.  To enable the NMI pin, validate the NMI pin with the PMC0 register.  The initial setting of the NMI pin is "No edge detected".  Select the NMI pin valid edge using the INTF0 and INTR0 registers.

# CHAPTER 24  KEY INTERRUPT FUNCTION

## 24.1  Function

A key interrupt request signal (INTKR) can be generated by inputting a falling edge to the eight key input pins (KR0 to KR7) by setting the KRM register.

**Table 24-1.  Assignment of Key Return Detection Pins**

| Flag | Pin Description |
|------|-----------------|
| KRM0 | Controls KR0 signal in 1-bit units |
| KRM1 | Controls KR1 signal in 1-bit units |
| KRM2 | Controls KR2 signal in 1-bit units |
| KRM3 | Controls KR3 signal in 1-bit units |
| KRM4 | Controls KR4 signal in 1-bit units |
| KRM5 | Controls KR5 signal in 1-bit units |
| KRM6 | Controls KR6 signal in 1-bit units |
| KRM7 | Controls KR7 signal in 1-bit units |

**Figure 24-1.  Key Return Block Diagram**
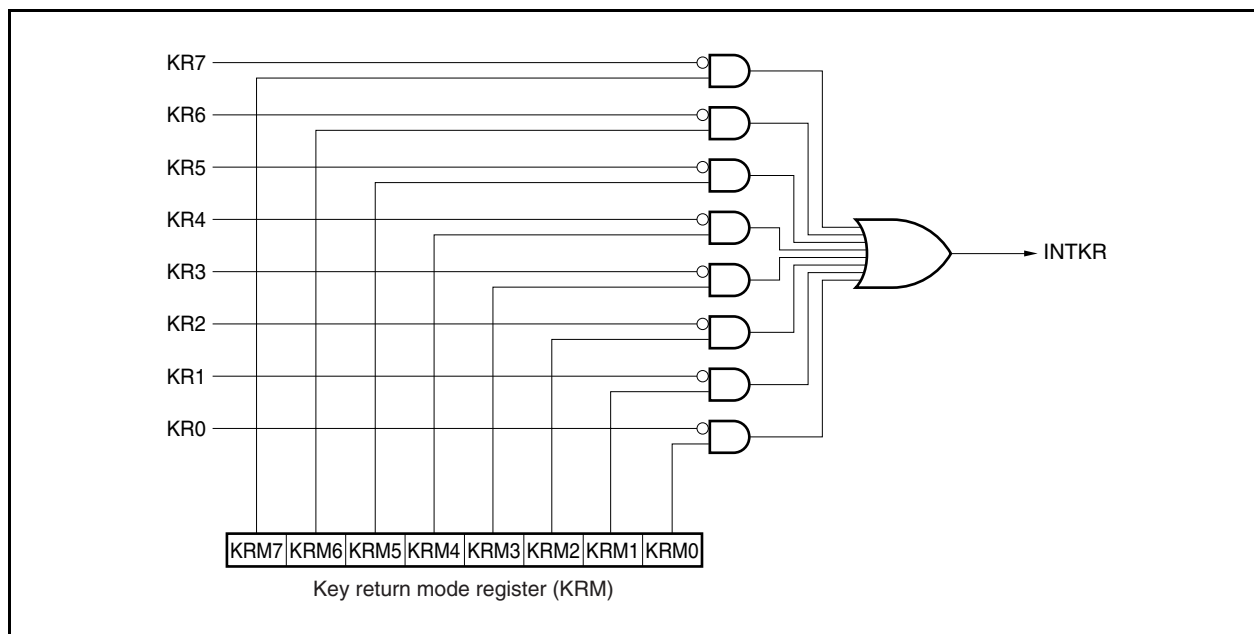
## 24.2  Register

**(1)  Key return mode register (KRM)**

The KRM register controls the KRM0 to KRM7 bits using the KR0 to KR7 signals.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H     R/W     Address: FFFFF300H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| KRM | KRM7 | KRM6 | KRM5 | KRM4 | KRM3 | KRM2 | KRM1 | KRM0 |

| KRMn | Control of key return mode |
|---|---|
| 0 | Does not detect key return signal |
| 1 | Detects key return signal |

**Caution  Rewrite the KRM register after once setting the KRM register to 00H.**

**Remark**  For the alternate-function pin settings, see **Table 4-20  Settings When Port Pins Are Used for Alternate Functions**.

## 24.3  Cautions

(1)  If a low level is input to any of the KR0 to KR7 pins, the INTKR signal is not generated even if the falling edge of another pin is input.

(2)  If the KRM register is changed, an interrupt request signal (INTKR) may be generated.  To prevent this, change the KRM register after disabling interrupts (DI) or masking, then clear the interrupt request flag (KRIC.KRIF bit) to 0, and enable interrupts (EI) or clear the mask.

(3)  To use the key interrupt function, be sure to set the port pin to the key return pin and then enable the operation with the KRM register.  To switch from the key return pin to the port pin, disable the operation with the KRM register and then set the port pin.

## CHAPTER 25  STANDBY FUNCTION

### 25.1  Overview

The power consumption of the system can be effectively reduced by using the standby modes in combination and selecting the appropriate mode for the application.  The available standby modes are listed in Table 25-1.

**Table 25-1.  Standby Modes**

| Mode | Functional Outline |
|---|---|
| HALT mode | Mode in which only the operating clock of the CPU is stopped |
| IDLE1 mode | Mode in which all the operations of the internal circuits except the oscillator, PLL[Note], and flash memory are stopped |
| IDLE2 mode | Mode in which all the operations of internal circuits except the oscillator are stopped |
| STOP mode | Mode in which all the operations of internal circuits except the subclock oscillator are stopped |
| Subclock operation mode | Mode in which the subclock is used as the internal system clock |
| Sub-IDLE mode | Mode in which all the operations of internal circuits except the oscillator are stopped, in the subclock operation mode |

**Note**   The PLL holds the previous operating status.

**Figure 25-1.  Status Transition**



**Note**  If a WDT overflow occurs during an oscillation stabilization time, the CPU operates on the internal oscillation clock.

**Remark**  fx: Main clock oscillation frequency
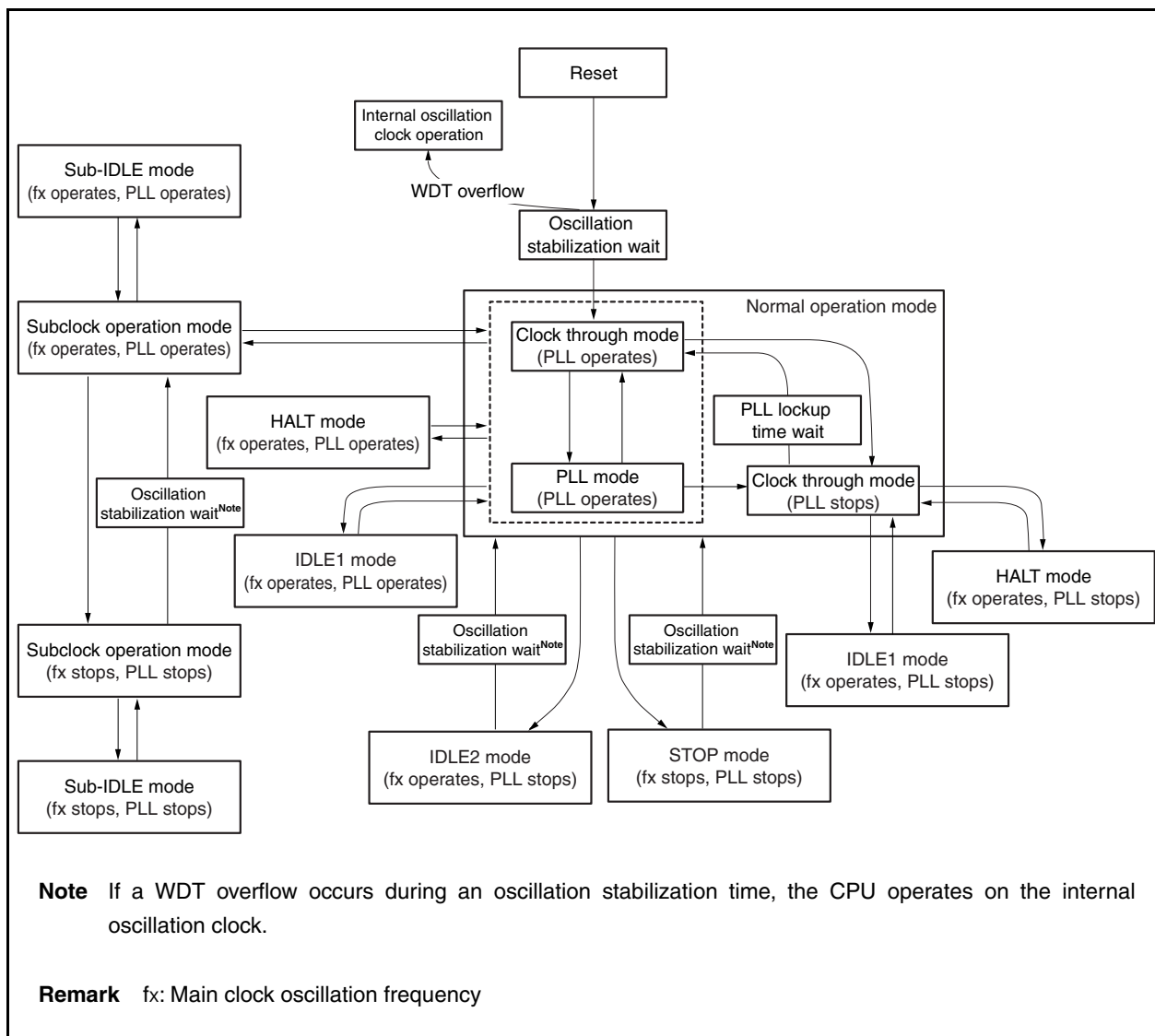
## 25.2  Registers

**(1) Power save control register (PSC)**

The PSC register is an 8-bit register that controls the standby function.  The STP bit of this register is used to specify the Standby mode.  This register is a special register that can be written only by the special sequence combinations (see **3.4.7  Special registers**).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H      R/W      Address:  FFFFF1FEH

| | 7 | <6> | <5> | <4> | 3 | 2 | <1> | 0 |
|---|---|---|---|---|---|---|---|---|
| PSC | 0 | NMI1M | NMI0M | INTM | 0 | 0 | STP | 0 |

| NMI1M | Control of releasing standby mode by INTWDT2 signal |
|---|---|
| 0 | Releasing standby mode by INTWDT2 signal enabled |
| 1 | Releasing standby mode by INTWDT2 signal disabled |

| NMI0M | Control of releasing standby mode by NMI pin input |
|---|---|
| 0 | Releasing standby mode by NMI pin input enabled |
| 1 | Releasing standby mode by NMI pin input disabled |

| INTM | Control of releasing standby mode by maskable interrupt request signals |
|---|---|
| 0 | Releasing standby mode by maskable interrupt request signals enabled |
| 1 | Releasing standby mode by maskable interrupt request signals disabled |

| STP | Standby mode[Note] setting |
|---|---|
| 0 | Normal mode |
| 1 | Standby mode |

**Note**   Standby mode set by STP bit: IDLE1, IDLE2, STOP, or sub-IDLE mode

**Cautions  1.** Before setting the IDLE1, IDLE2, STOP, or sub-IDLE mode, set the PSMR.PSM1 and PSMR.PSM0 bits and then set the STP bit.

**2.** Settings of the NMI1M, NMI0M, and INTM bits are invalid when HALT mode is released.

**3.** If the NMI1M, NMI0M, or INTM bit is set to 1 at the same time the STP bit is set to 1, the setting of NMI1M, NMI0M, or INTM bit becomes invalid.  If there is an unmasked interrupt request signal being held pending when the IDLE1/IDLE2/STOP mode is set, set the bit corresponding to the interrupt request signal (NMI1M, NMI0M, or INTM) to 1, and then set the STP bit to 1.

**(2) Power save mode register (PSMR)**

The PSMR register is an 8-bit register that controls the operation status in the power save mode and the clock operation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H     R/W     Address:  FFFFF820H

| | 7 | 6 | 5 | 4 | 3 | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PSMR | 0 | 0 | 0 | 0 | 0 | 0 | PSM1 | PSM0 |

| PSM1 | PSM0 | Specification of operation in software standby mode |
|---|---|---|
| 0 | 0 | IDLE1, sub-IDLE modes |
| 0 | 1 | STOP mode |
| 1 | 0 | IDLE2, sub-IDLE modes |
| 1 | 1 | STOP mode |

**Cautions 1.  Be sure to set bits 2 to 7 to "0".**

**2.  The PSM0 and PSM1 bits are valid only when the PSC.STP bit is 1.**

**Remark**  IDLE1:   In this mode, all operations except the oscillator operation and some other circuits (flash memory and PLL) are stopped.

After the IDLE1 mode is released, the normal operation mode is restored without needing to secure the oscillation stabilization time, like the HALT mode.

IDLE2:   In this mode, all operations except the oscillator operation are stopped.

After the IDLE2 mode is released, the normal operation mode is restored following the lapse of the setup time specified by the OSTS register (flash memory and PLL).

STOP:   In this mode, all operations except the subclock oscillator operation are stopped.

After the STOP mode is released, the normal operation mode is restored following the lapse of the oscillation stabilization time specified by the OSTS register.

Sub-IDLE:  In this mode, all other operations are halted except for the oscillator.  After the IDLE mode has been released by the interrupt request signal, the subclock operation mode will be restored after 12 cycles of the subclock have been secured.

**(3) Oscillation stabilization time select register (OSTS)**

The wait time until the oscillation stabilizes after the STOP mode is released or the wait time until the on-chip flash memory stabilizes after the IDLE2 mode is released is controlled by the OSTS register.

This register can be read or written in 8-bit units.

Reset sets this register to 06H.

After reset:  06H    R/W    Address:  FFFFF6C0H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSTS | 0 | 0 | 0 | 0 | 0 | OSTS2 | OSTS1 | OSTS0 |

| OSTS2 | OSTS1 | OSTS0 | Selection of oscillation stabilization time/setup time[Note] | fx | |
|---|---|---|---|---|---|
| | | | | 3 MHz | 6 MHz |
| 0 | 0 | 0 | $2^{10}/f_X$ | 0.341 ms | 0.171 ms |
| 0 | 0 | 1 | $2^{11}/f_X$ | 0.683 ms | 0.341 ms |
| 0 | 1 | 0 | $2^{12}/f_X$ | 1.365 ms | 0.683 ms |
| 0 | 1 | 1 | $2^{13}/f_X$ | 2.730 ms | 1.365 ms |
| 1 | 0 | 0 | $2^{14}/f_X$ | 5.461 ms | 2.731 ms |
| 1 | 0 | 1 | $2^{15}/f_X$ | 10.923 ms | 5.461 ms |
| 1 | 1 | 0 | $2^{16}/f_X$ | 21.85 ms | 10.92 ms |
| 1 | 1 | 1 | Setting prohibited | | |

**Note** The oscillation stabilization time and setup time are required when the STOP mode and IDLE2 mode are released, respectively.

**Cautions 1. The wait time following release of the STOP mode does not include the time until the clock oscillation starts ("a" in the figure below) following release of the STOP mode, regardless of whether the STOP mode is released by reset or the occurrence of an interrupt request signal.**
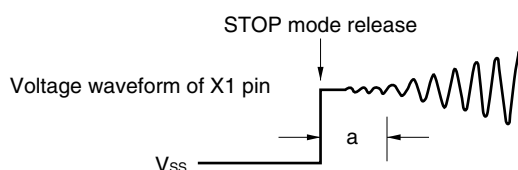


**2. Be sure to set bits 3 to 7 to "0".**

**3. The oscillation stabilization time following reset release is $2^{16}/f_X$ (because the initial value of the OSTS register = 06H).**

**Remark**  $f_X$ = Main clock oscillation frequency

## 25.3  HALT Mode

### 25.3.1  Setting and operation status

The HALT mode is set when a dedicated instruction (HALT) is executed in the normal operation mode.

In the HALT mode, the clock oscillator continues operating.  Only clock supply to the CPU is stopped; clock supply to the other on-chip peripheral functions continues.

As a result, program execution is stopped, and the internal RAM retains the contents before the HALT mode was set. The on-chip peripheral functions that are independent of instruction processing by the CPU continue operating.

Table 25-3 shows the operating status in the HALT mode.

The average current consumption of the system can be reduced by using the HALT mode in combination with the normal operation mode for intermittent operation.

**Cautions 1.  Insert five or more NOP instructions after the HALT instruction.**

**2.  If the HALT instruction is executed while an unmasked interrupt request signal is being held pending, the status shifts to HALT mode, but the HALT mode is then released immediately by the pending interrupt request.**

### 25.3.2  Releasing HALT mode

The HALT mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP00 to INTP18 pin input), unmasked internal interrupt request signal from a peripheral function operable in the HALT mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)).

After the HALT mode has been released, the normal operation mode is restored.

**(1)  Releasing HALT mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal**

The HALT mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal.  If the HALT mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

(a)  If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the HALT mode is released, but that interrupt request signal is not acknowledged.  The interrupt request signal itself is retained.

(b)  If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the HALT mode is released and that interrupt request signal is acknowledged.

**Table 25-2.  Operation After Releasing HALT Mode by Interrupt Request Signal**

| Release Source | Interrupt Enabled (EI) Status | Interrupt Disabled (DI) Status |
|---|---|---|
| Non-maskable interrupt request signal | Execution branches to the handler address. | |
| Maskable interrupt request signal | Execution branches to the handler address or the next instruction is executed. | The next instruction is executed. |

**(2)  Releasing HALT mode by reset**

The same operation as the normal reset operation is performed.

**Table 25-3.  Operating Status in HALT Mode**

| Setting of HALT Mode / Item | | Operating Status | |
|---|---|---|---|
| | | When Subclock Is Not Used | When Subclock Is Used |
| Main clock oscillator ($f_X$) | | Oscillation enabled | |
| Subclock oscillator ($f_{XT}$) | | – | Oscillation enabled |
| Internal oscillator ($f_R$) | | Oscillation enabled | |
| PLL | | Operable | |
| CPU | | Stops operation | |
| DMA controller | | Operable | |
| Interrupt controller | | Operable | |
| Timer | TAA0 to TAA5 | Operable | |
| | TAB0, TAB1 | Operable | |
| | TMM0 to TMM3 | Operable when a clock other than $f_{XT}$ is selected as the count clock | Operable |
| | TMT0 | Operable | |
| Real-time counter (RTC) | | Operable when $f_X$ (divided BRG) is selected as the count clock | Operable |
| Watchdog timer (WDT2) | | Operable when a clock other than $f_{XT}$ is selected as the count clock | Operable |
| Serial interface | CSIF0 to CSIF4 | Operable | |
| | I²C00 to I²C02 | Operable | |
| | UARTC0 to UARTC4 | Operable | |
| A/D converter | | Operable | |
| D/A converter | | Operable | |
| Real-time output function (RTO) | | Operable | |
| Key interrupt function (KR) | | Operable | |
| CRC operation circuit | | Operable (No data input to the CRCIN register because the CPU is stopped) | |
| External bus interface | | See **CHAPTER 5  BUS CONTROL FUNCTION**. | |
| Port function | | Retains status before HALT mode was set | |
| Internal data | | The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the HALT mode was set. | |
| USB function | | Operable | |
| USB host | | Operable | |

## 25.4  IDLE1 Mode

### 25.4.1  Setting and operation status

The IDLE1 mode is set by setting the PSMR.PSM1 and PSMR.PSM0 bits to 00 and setting the PSC.STP bit to 1 in the normal operation mode.

In the IDLE1 mode, the clock oscillator, PLL, and flash memory continue operating but clock supply to the CPU and other on-chip peripheral functions stops.

As a result, program execution stops and the contents of the internal RAM before the IDLE1 mode was set are retained. The CPU and other on-chip peripheral functions stop operating.  However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Table 25-5 shows the operating status in the IDLE1 mode.

The IDLE1 mode can reduce the power consumption more than the HALT mode because it stops the operation of the on-chip peripheral functions.  The main clock oscillator does not stop, so the normal operation mode can be restored without waiting for the oscillation stabilization time after the IDLE1 mode has been released, in the same manner as when the HALT mode is released.

**Cautions 1.  Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the IDLE1 mode.**

**2.  If the IDLE1 mode is set while an unmasked interrupt request signal is being held pending, the IDLE1 mode is released immediately by the pending interrupt request.**

### 25.4.2  Releasing IDLE1 mode

The IDLE1 mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP00 to INTP18 pin input), unmasked internal interrupt request signal from a peripheral function operable in the IDLE1 mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)).

After the IDLE1 mode has been released, the normal operation mode is restored.

**(1)  Releasing IDLE1 mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal**

The IDLE1 mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal.  If the IDLE1 mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is processed as follows.

(a)  If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the IDLE1 mode is released, but that interrupt request signal is not acknowledged.  The interrupt request signal itself is retained.

(b)  If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the IDLE1 mode is released and that interrupt request signal is acknowledged.

**Caution  An interrupt request signal that is disabled by setting the PSC.NMI1M, PSC.NMI0M, and PSC.INTM bits to 1 becomes invalid and IDLE1 mode is not released.**

**Table 25-4.  Operation After Releasing IDLE1 Mode by Interrupt Request Signal**

| Release Source | Interrupt Enabled (EI) Status | Interrupt Disabled (DI) Status |
| --- | --- | --- |
| Non-maskable interrupt request signal | Execution branches to the handler address after securing the prescribed setup time. | |
| Maskable interrupt request signal | Execution branches to the handler address or the next instruction is executed after securing the prescribed setup time. | The next instruction is executed after securing the prescribed setup time. |

**(2)  Releasing IDLE1 mode by reset**

The same operation as the normal reset operation is performed.

**Table 25-5.  Operating Status in IDLE1 Mode**

| Setting of IDLE1 Mode Item | | Operating Status | |
|---|---|---|---|
| | | When Subclock Is Not Used | When Subclock Is Used |
| Main clock oscillator ($f_X$) | | Oscillation enabled | |
| Subclock oscillator ($f_{XT}$) | | – | Oscillation enabled |
| Internal oscillator ($f_R$) | | Oscillation enabled | |
| PLL | | Operable | |
| CPU | | Stops operation | |
| DMA controller | | Stops operation | |
| Interrupt controller | | Stops operation (but standby mode release is possible) | |
| Timer | TAA0 to TAA5 | Stops operation | |
| | TAB0, TAB1 | Stops operation | |
| | TMM0 to TMM3 | Operable when $f_R/8$ is selected as the count clock | Operable when $f_R/8$ or $f_{XT}$ is selected as the count clock |
| | TMT0 | Stops operation | |
| Real-time counter (RTC) | | Operable when $f_X$ (divided BRG) is selected as the count clock | Operable |
| Watchdog timer (WDT2) | | Operable when $f_R$ is selected as the count clock | Operable when $f_R$ or $f_{XT}$ is selected as the count clock |
| Serial interface | CSIF0 to CSIF4 | Operable when the $\overline{\text{SCKFn}}$ input clock is selected as the count clock (n = 0 to 4) | |
| | I²C00 to I²C02 | Stops operation | |
| | UARTC0 to UARTC4 | Stops operation (but UARTC0 is operable when the ASCKC0 input clock is selected) | |
| A/D converter | | Holds operation (conversion result held)**Note** | |
| D/A converter | | Holds operation (output held**Note**) | |
| Real-time output function (RTO) | | Stops operation (output held) | |
| Key interrupt function (KR) | | Operable | |
| CRC operation circuit | | Stops operation | |
| External bus interface | | See **CHAPTER 5  BUS CONTROL FUNCTION**. | |
| Port function | | Retains status before IDLE1 mode was set | |
| Internal data | | The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the IDLE1 mode was set. | |
| USB function | | Operable when the UCLK input is selected as the operation clock or when the PLL is operating. **Note** | |
| USB host | | Operable when the PLL is operating. **Note** | |

**Note**  To realize low power consumption, stop the A/D converter, D/A converter, USB function controller and USB host controller before shifting to the IDLE1 mode.

## 25.5 IDLE2 Mode

### 25.5.1 Setting and operation status

The IDLE2 mode is set by setting the PSMR.PSM1 and PSMR.PSM0 bits to 10 and setting the PSC.STP bit to 1 in the normal operation mode.

In the IDLE2 mode, the clock oscillator continues operation but clock supply to the CPU, PLL, flash memory, and other on-chip peripheral functions stops.

As a result, program execution stops and the contents of the internal RAM before the IDLE2 mode was set are retained. The CPU, PLL, and other on-chip peripheral functions stop operating. However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Table 25-7 shows the operating status in the IDLE2 mode.

The IDLE2 mode can reduce the power consumption more than the IDLE1 mode because it stops the operations of the on-chip peripheral functions, PLL, and flash memory. However, because the PLL and flash memory are stopped, a setup time for the PLL and flash memory is required when IDLE2 mode is released.

**Cautions 1. Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the IDLE2 mode.**

**2. If the IDLE2 mode is set while an unmasked interrupt request signal is being held pending, the IDLE2 mode is released immediately by the pending interrupt request.**

**25.5.2  Releasing IDLE2 mode**

The IDLE2 mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP00 to INTP18 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the IDLE2 mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)).  The PLL returns to the operating status it was in before the IDLE2 mode was set.

After the IDLE2 mode has been released, the normal operation mode is restored.

**(1)  Releasing IDLE2 mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal**

The IDLE2 mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal.  If the IDLE2 mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is processed as follows.

(a)  If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the IDLE2 mode is released, but that interrupt request signal is not acknowledged.  The interrupt request signal itself is retained.

(b)  If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the IDLE2 mode is released and that interrupt request signal is acknowledged.

**Caution   The interrupt request signal that is disabled by setting the PSC.NMI1M, PSC.NMI0M, and PSC.INTM bits to 1 becomes invalid and IDLE2 mode is not released.**

**Table 25-6.  Operation After Releasing IDLE2 Mode by Interrupt Request Signal**

| Release Source | Interrupt Enabled (EI) Status | Interrupt Disabled (DI) Status |
|---|---|---|
| Non-maskable interrupt request signal | Execution branches to the handler address after securing the prescribed setup time. | |
| Maskable interrupt request signal | Execution branches to the handler address or the next instruction is executed after securing the prescribed setup time. | The next instruction is executed after securing the prescribed setup time. |

**(2)  Releasing IDLE2 mode by reset**

The same operation as the normal reset operation is performed.

**Table 25-7.  Operating Status in IDLE2 Mode**

| Setting of IDLE2 Mode / Item | | Operating Status | |
|---|---|---|---|
| | | When Subclock Is Not Used | When Subclock Is Used |
| Main clock oscillator ($f_X$) | | Oscillation enabled | |
| Subclock oscillator ($f_{XT}$) | | – | Oscillation enabled |
| Internal oscillator ($f_R$) | | Oscillation enabled | |
| PLL | | Stops operation | |
| CPU | | Stops operation | |
| DMA controller | | Stops operation | |
| Interrupt controller | | Stops operation | |
| Timer | TAA0 to TAA5 | Stops operation | |
| | TAB0, TAB1 | Stops operation | |
| | TMM0 to TMM3 | Operable when $f_R$/8 is selected as the count clock | Operable when $f_R$/8 or $f_{XT}$ is selected as the count clock |
| | TMT0 | Stops operation | |
| Real-time counter (RTC) | | Operable when $f_X$ (divided BRG) is selected as the count clock | Operable |
| Watchdog timer (WDT2) | | Operable when $f_R$ is selected as the count clock | Operable when $f_R$ or $f_{XT}$ is selected as the count clock |
| Serial interface | CSIF0 to CSIF4 | Operable when the $\overline{SCKFn}$ input clock is selected as the count clock (n = 0 to 4) | |
| | I²C00 to I²C02 | Stops operation | |
| | UARTC0 to UARTC4 | Stops operation (but UARTC0 is operable when the ASCKC0 input clock is selected) | |
| A/D converter | | Holds operation (conversion result held)[Note] | |
| D/A converter | | Holds operation (output held[Note]) | |
| Real-time output function (RTO) | | Stops operation (output held) | |
| Key interrupt function (KR) | | Operable | |
| CRC operation circuit | | Stops operation | |
| External bus interface | | See **CHAPTER 5  BUS CONTROL FUNCTION**. | |
| Port function | | Retains status before IDLE2 mode was set | |
| Internal data | | The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the IDLE2 mode was set. | |
| USB function | | Operable when the UCLK input is selected as the operation clock or when the PLL is operating.[Note] | |
| USB host | | Stops operation | |

**Note**  To realize low power consumption, stop the A/D converter, D/A converter, USB function controller before shifting to the IDLE2 mode.

### 25.5.3  Securing setup time when releasing IDLE2 mode

Secure the setup time for the flash memory after releasing the IDLE2 mode because the operation of the blocks other than the main clock oscillator stops after the IDLE2 mode is set.

**(1)  Releasing IDLE2 mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal**

Secure the specified setup time by setting the OSTS register.

When the releasing source is generated, the dedicated internal timer starts counting according to the OSTS register setting.  When it overflows, the normal operation mode is restored.
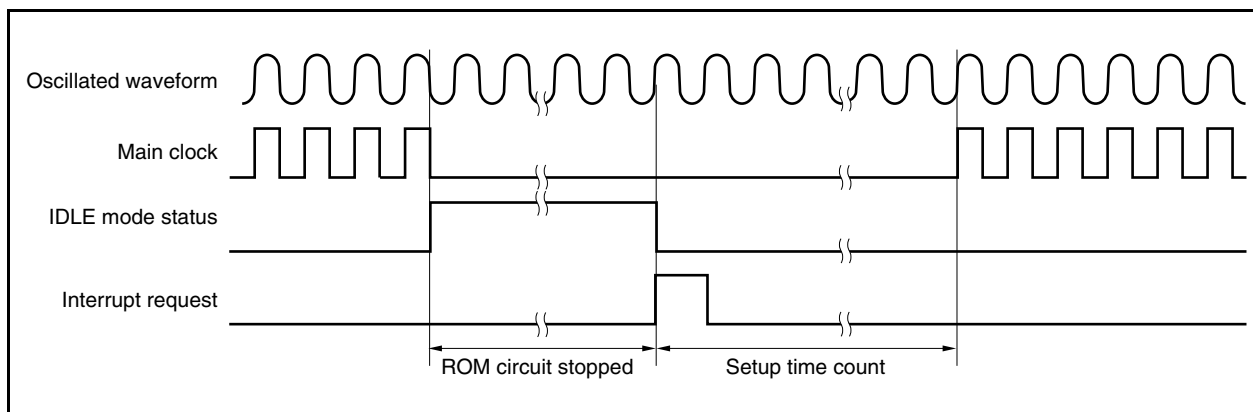


**(2)  Release by reset ($\overline{\text{RESET}}$ pin input, WDT2RES generation)**

This operation is the same as that of a normal reset.

The oscillation stabilization time is the initial value of the OSTS register, $2^{16}/f_X$.

## 25.6  STOP Mode

### 25.6.1  Setting and operation status

The STOP mode is set by setting the PSMR.PSM1 and PSMR.PSM0 bits to 01 or 11 and setting the PSC.STP bit to 1 in the normal operation mode.

In the STOP mode, the subclock oscillator continues operating but the main clock oscillator stops.  Clock supply to the CPU and the on-chip peripheral functions is stopped.

As a result, program execution stops, and the contents of the internal RAM before the STOP mode was set are retained.  The on-chip peripheral functions that operate with the clock oscillated by the subclock oscillator or an external clock continue operating.

Table 25-9 shows the operating status in the STOP mode.

Because the STOP mode stops operation of the main clock oscillator, it reduces the power consumption to a level lower than the IDLE2 mode.  If the subclock oscillator, internal oscillator, and external clock are not used, the power consumption can be minimized with only leakage current flowing.

**Cautions 1.  Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the STOP mode.**

**2.  If the STOP mode is set while an unmasked interrupt request signal is being held pending, the STOP mode is released immediately by the pending interrupt request.**

### 25.6.2  Releasing STOP mode

The STOP mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP00 to INTP18 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the STOP mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, or low-voltage detector (LVI)).

After the STOP mode has been released, the normal operation mode is restored after the oscillation stabilization time has been secured.

**(1) Releasing STOP mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal**

The STOP mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal.  If the STOP mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

(a)  If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the STOP mode is released, but that interrupt request signal is not acknowledged.  The interrupt request signal itself is retained.

(b)  If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the STOP mode is released and that interrupt request signal is acknowledged.

**Caution   The interrupt request that is disabled by setting the PSC.NMI1M, PSC.NMI0M, and PSC.INTM bits to 1 becomes invalid and STOP mode is not released.**

**Table 25-8.  Operation After Releasing STOP Mode by Interrupt Request Signal**

| Release Source | Interrupt Enabled (EI) Status | Interrupt Disabled (DI) Status |
|---|---|---|
| Non-maskable interrupt request signal | Execution branches to the handler address after securing the oscillation stabilization time. | |
| Maskable interrupt request signal | Execution branches to the handler address or the next instruction is executed after securing the oscillation stabilization time. | The next instruction is executed after securing the oscillation stabilization time. |

**(2)  Releasing STOP mode by reset**

   The same operation as the normal reset operation is performed.

**Table 25-9.  Operating Status in STOP Mode**

| Setting of STOP Mode Item | | Operating Status | |
|---|---|---|---|
| | | When Subclock Is Not Used | When Subclock Is Used |
| Main clock oscillator ($f_X$) | | Stops oscillation | |
| Subclock oscillator ($f_{XT}$) | | – | Oscillation enabled |
| Internal oscillator ($f_R$) | | Oscillation enabled | |
| PLL | | Stops operation | |
| CPU | | Stops operation | |
| DMA controller | | Stops operation | |
| Interrupt controller | | Stops operation | |
| Timer | TAA0 to TAA5 | Stops operation | |
| | TAB0, TAB1 | Stops operation | |
| | TMM0 to TMM3 | Operable when $f_R/8$ is selected as the count clock | Operable when $f_R/8$ or $f_{XT}$ is selected as the count clock |
| | TMT0 | Stops operation | |
| Real-time counter (RTC) | | Stops operation | Operable when $f_{XT}$ is selected as the count clock |
| Watchdog timer (WDT2) | | Operable when $f_R$ is selected as the count clock | Operable when $f_R$ or $f_{XT}$ is selected as the count clock |
| Serial interface | CSIF0 to CSIF4 | Operable when the $\overline{SCKFn}$ input clock is selected as the count clock (n = 0 to 4) | |
| | I²C00 to I²C02 | Stops operation | |
| | UARTC0 to UARTC4 | Stops operation (but UARTC0 is operable when the ASCKC0 input clock is selected) | |
| A/D converter | | Stops operation (conversion result undefined)[Notes 1, 2] | |
| D/A converter | | Stops operation[Notes 3, 4] (high impedance is output) | |
| Real-time output function (RTO) | | Stops operation (output held) | |
| Key interrupt function (KR) | | Operable | |
| CRC operation circuit | | Stops operation | |
| External bus interface | | See **CHAPTER 5  BUS CONTROL FUNCTION**. | |
| Port function | | Retains status before STOP mode was set | |
| Internal data | | The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the STOP mode was set. | |
| USB function | | Stops operation | |
| USB host | | Stops operation | |

**Notes 1.** If the STOP mode is set while the A/D converter is operating, the A/D converter is automatically stopped and starts operating again after the STOP mode is released.  However, in that case, the A/D conversion results after the STOP mode is released are invalid.  All the A/D conversion results before the STOP mode is set are invalid.

**2.** Even if the STOP mode is set while the A/D converter is operating, the power consumption is reduced equivalently to when the A/D converter is stopped before the STOP mode is set.

**3.** If the STOP mode is set while the D/A converter is operating, the D/A converter is automatically stopped and the pin status becomes high impedance.  After the STOP mode is released, D/A conversion resumes, the setting time elapses, and the status returns to the output level before the STOP mode was set.

**4.** Even if the STOP mode is set while the D/A converter is operating, the power consumption is reduced equivalently to when the D/A converter is stopped before the STOP mode is set.

**25.6.3 Securing oscillation stabilization time when releasing STOP mode**

Secure the oscillation stabilization time for the main clock oscillator after releasing the STOP mode because the operation of the main clock oscillator stops after STOP mode is set.

**(1) Releasing STOP mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal**

Secure the oscillation stabilization time by setting the OSTS register.

When the releasing source is generated, the dedicated internal timer starts counting according to the OSTS register setting. When it overflows, the normal operation mode is restored.
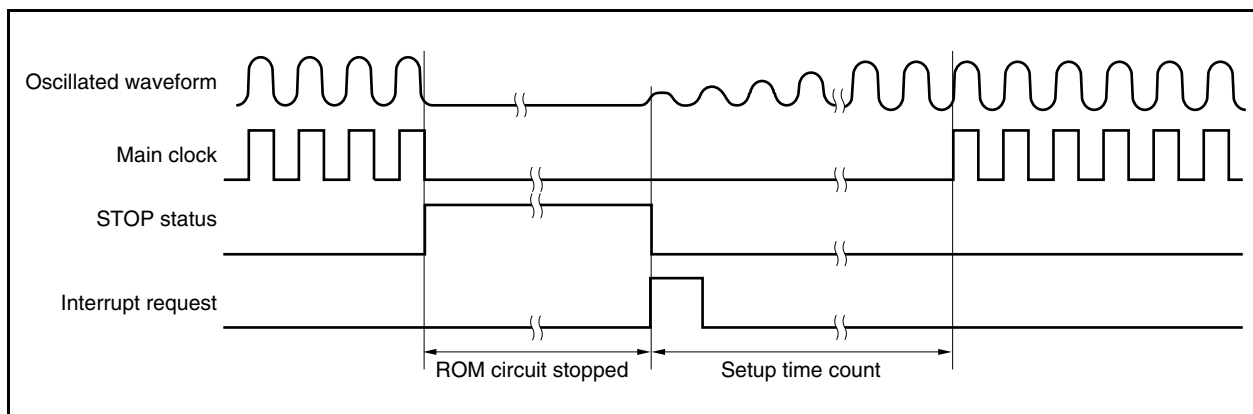


**(2) Release by reset**

This operation is the same as that of a normal reset.

The oscillation stabilization time is the initial value of the OSTS register, $2^{16}/f_X$.

## 25.7  Subclock Operation Mode

### 25.7.1  Setting and operation status

The subclock operation mode is set by setting the PCC.CK3 bit to 1 in the normal operation mode.

When the subclock operation mode is set, the internal system clock is changed from the main clock to the subclock. Check whether the clock has been switched by using the PCC.CLS bit.

When the PCC.MCK bit is set to 1, the operation of the main clock oscillator is stopped. As a result, the system operates only on the subclock.

In the subclock operation mode, the power consumption can be reduced to a level lower than in the normal operation mode because the subclock is used as the internal system clock. In addition, the power consumption can be further reduced to the level of the STOP mode by stopping the operation of the main clock oscillator.

Table 25-10 shows the operating status in subclock operation mode.

**Cautions 1.** **When manipulating the CK3 bit, do not change the set values of the PCC.CK2 to PCC.CK0 bits (using a bit manipulation instruction to manipulate the bit is recommended). For details of the PCC register, see 6.3 (1)  Processor clock control register (PCC).**

**2.** **If the following conditions are not satisfied, change the CK2 to CK0 bits so that the conditions are satisfied and set the subclock operation mode.**

**Internal system clock ($f_{CLK}$) > Subclock ($f_{XT}$ = 32.768 kHz) $\times$ 4**

**Remark** Internal system clock ($f_{CLK}$): Clock generated from main clock ($f_{XX}$) in accordance with the settings of the CK2 to CK0 bits

### 25.7.2  Releasing subclock operation mode

The subclock operation mode is released by a reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)) when the CK3 bit is set to 0.

If the main clock is stopped (MCK bit = 1), set the MCK bit to 1, secure the oscillation stabilization time of the main clock by software, and set the CK3 bit to 0.

The normal operation mode is restored when the subclock operation mode is released.

**Caution** **When manipulating the CK3 bit, do not change the set values of the CK2 to CK0 bits (using a bit manipulation instruction to manipulate the bit is recommended).**
**For details of the PCC register, see 6.3 (1)  Processor clock control register (PCC).**

**Table 25-10.  Operating Status in Subclock Operation Mode**

| Setting of Subclock Operation Mode / Item | | Operating Status | |
|---|---|---|---|
| | | When Main Clock Is Oscillating | When Main Clock Is Stopped |
| Subclock oscillator ($f_{XT}$) | | Oscillation enabled | |
| Internal oscillator ($f_R$) | | Oscillation enabled | |
| PLL | | Operable | Stops operation[Note] |
| CPU | | Operable | |
| DMA controller | | Operable | |
| Interrupt controller | | Operable | |
| Timer | TAA0 to TAA5 | Operable | Stops operation |
| | TAB0, TAB1 | Operable | Stops operation |
| | TMM0 to TMM3 | Operable | Operable when $f_R/8$ or $f_{XT}$ is selected as the count clock |
| | TMT0 | | |
| Real-time counter (RTC) | | Operable | Operable when $f_{XT}$ is selected as the count clock |
| Watchdog timer (WDT2) | | Operable | Operable when $f_R$ or $f_{XT}$ is selected as the count clock |
| Serial interface | CSIF0 to CSIF4 | Operable | Operable when the $\overline{SCKFn}$ input clock is selected as the count clock (n = 0 to 4) |
| | I²C00 to I²C02 | Operable | Stops operation |
| | UARTC0 to UARTC4 | Operable | Stops operation (but UARTC0 is operable when the ASCKC0 input clock is selected) |
| A/D converter | | Operable | Stops operation |
| D/A converter | | Operable | |
| Real-time output function (RTO) | | Operable | Stops operation (output held) |
| Key interrupt function (KR) | | Operable | |
| CRC operation circuit | | Operable | |
| External bus interface | | See **CHAPTER 5  BUS CONTROL FUNCTION**. | |
| Port function | | Settable | |
| Internal data | | Settable | |
| USB function | | Operable | Stops operation |
| USB host | | Operable | Stops operation |

**Note**   Be sure to stop the PLL (PLLCTL.PLLON bit = 0) before stopping the main clock.

**Caution   When the CPU is operating on the subclock and main clock oscillation is stopped, accessing a register in which a wait occurs is disabled.  If a wait is generated, it can be released only by reset (see 3.4.8 (2)).**

## 25.8  Sub-IDLE Mode

### 25.8.1  Setting and operation status

The sub-IDLE mode is set by setting the PSMR.PSM1 and PSMR.PSM0 bits to 00 or 10 and setting the PSC.STP bit to 1 in the subclock operation mode.

In this mode, the clock oscillator continues operating but clock supply to the CPU, flash memory, and the other on-chip peripheral functions is stopped.

As a result, program execution stops and the contents of the internal RAM before the sub-IDLE mode was set are retained.  The CPU and the other on-chip peripheral functions are stopped.  However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Because the sub-IDLE mode stops operation of the CPU, flash memory, and other on-chip peripheral functions, it can reduce the power consumption more than the subclock operation mode.  If the sub-IDLE mode is set after the main clock has been stopped, the current consumption can be reduced to a level as low as that in the STOP mode.

Table 25-12 shows the operating status in the sub-IDLE mode.

**Cautions 1.  Following the store instruction to the PSC register for setting the sub-IDLE mode, insert the five or more NOP instructions.**

**2.  If the sub-IDLE mode is set while an unmasked interrupt request signal is being held pending, the sub-IDLE mode is then released immediately by the pending interrupt request.**

### 25.8.2  Releasing sub-IDLE mode

The sub-IDLE mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP00 to INTP18 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the sub-IDLE mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)).  The PLL returns to the operating status it was in before the sub-IDLE mode was set.

When the sub-IDLE mode is released by an interrupt request signal, the subclock operation mode is set.

**(1)  Releasing sub-IDLE mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal**

The sub-IDLE mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal.

If the sub-IDLE mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

(a)  If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the sub-IDLE mode is released, but that interrupt request signal is not acknowledged.  The interrupt request signal itself is retained.

(b)  If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the sub-IDLE mode is released and that interrupt request signal is acknowledged.

**Cautions 1.  The interrupt request signal that is disabled by setting the PSC.NMI1M, PSC.NMI0M, and PSC.INTM bits to 1 becomes invalid and sub-IDLE mode is not released.**

**2.  When the sub-IDLE mode is released, 12 cycles of the subclock (about 366 $\mu$s) elapse from when the interrupt request signal that releases the sub-IDLE mode is generated to when the mode is released.**

**Table 25-11. Operation After Releasing Sub-IDLE Mode by Interrupt Request Signal**

| Release Source | Interrupt Enabled (EI) Status | Interrupt Disabled (DI) Status |
|---|---|---|
| Non-maskable interrupt request signal | Execution branches to the handler address. | |
| Maskable interrupt request signal | Execution branches to the handler address or the next instruction is executed. | The next instruction is executed. |

**(2) Releasing sub-IDLE mode by reset**

The same operation as the normal reset operation is performed.

**Table 25-12. Operating Status in Sub-IDLE Mode**

| Setting of Sub-IDLE Mode / Item | | Operating Status | |
|---|---|---|---|
| | | When Main Clock Is Oscillating | When Main Clock Is Stopped |
| Subclock oscillator ($f_{XT}$) | | Oscillation enabled | |
| Internal oscillator ($f_R$) | | Oscillation enabled | |
| PLL | | Operable | Stops operation[Note 1] |
| CPU | | Stops operation | |
| DMA controller | | Stops operation | |
| Interrupt controller | | Stops operation | |
| Timer | TAA0 to TAA5 | Stops operation | |
| | TAB0, TAB1 | Stops operation | |
| | TMM0 to TMM3 | Operable when $f_R/8$ or $f_{XT}$ is selected as the count clock | |
| | TMT0 | Stops operation | |
| Real-time counter (RTC) | | Operable | Operable when $f_{XT}$ is selected as the count clock |
| Watchdog timer (WDT2) | | Operable when $f_R$ or $f_{XT}$ is selected as the count clock | |
| Serial interface | CSIF0 to CSIF4 | Operable when the $\overline{SCKFn}$ input clock is selected as the count clock (n = 0 to 4) | |
| | I²C00 to I²C02 | Stops operation | |
| | UARTC0 to UARTC4 | Stops operation (but UARTC0 is operable when the ASCKC0 input clock is selected) | |
| A/D converter | | Holds operation (conversion result held)[Note 2] | |
| D/A converter | | Holds operation (output held[Note 2]) | |
| Real-time output function (RTO) | | Stops operation (output held) | |
| Key interrupt function (KR) | | Operable | |
| CRC operation circuit | | Stops operation | |
| External bus interface | | See **CHAPTER 5 BUS CONTROL FUNCTION**. (same operation status as IDLE mode). | |
| Port function | | Retains status before sub-IDLE mode was set | |
| Internal data | | The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the sub-IDLE mode was set. | |
| USB function | | Stops operation | |
| USB host | | Stops operation | |

**Notes 1.** Be sure to stop the PLL (PLLCTL.PLLON bit = 0) before stopping the main clock.

**2.** To realize low power consumption, stop the A/D and D/A converters before shifting to the sub-IDLE mode.

# CHAPTER 26  RESET FUNCTIONS

## 26.1  Overview

The following reset functions are available.

(1)  Four kinds of reset sources
- External reset input via the $\overline{\text{RESET}}$ pin
- Reset via the watchdog timer 2 (WDT2) overflow (WDT2RES)
- System reset via the comparison of the low-voltage detector (LVI) supply voltage and detected voltage
- System reset via the detecting clock monitor (CLM) oscillation stop

After a reset is released, the source of the reset can be confirmed with the reset source flag register (RESF).

(2)  Emergency operation mode
If the WDT2 overflows during the main clock oscillation stabilization time inserted after reset, a main clock oscillation anomaly is judged and the CPU starts operating on the internal oscillation clock.

**Caution   In emergency operation mode, do not access on-chip peripheral I/O registers other than registers used for interrupts, port function, WDT2, or timer M, each of which can operate with the internal oscillation clock.  In addition, operation of CSIF0 to CSIF4 and UARTC0 using the externally input clock is also prohibited in this mode.**

**Figure 26-1.  Block Diagram of Reset Function**



**Caution   An LVI circuit internal reset does not reset the LVI circuit.**

**Remark**   LVIM: Low-voltage detection register
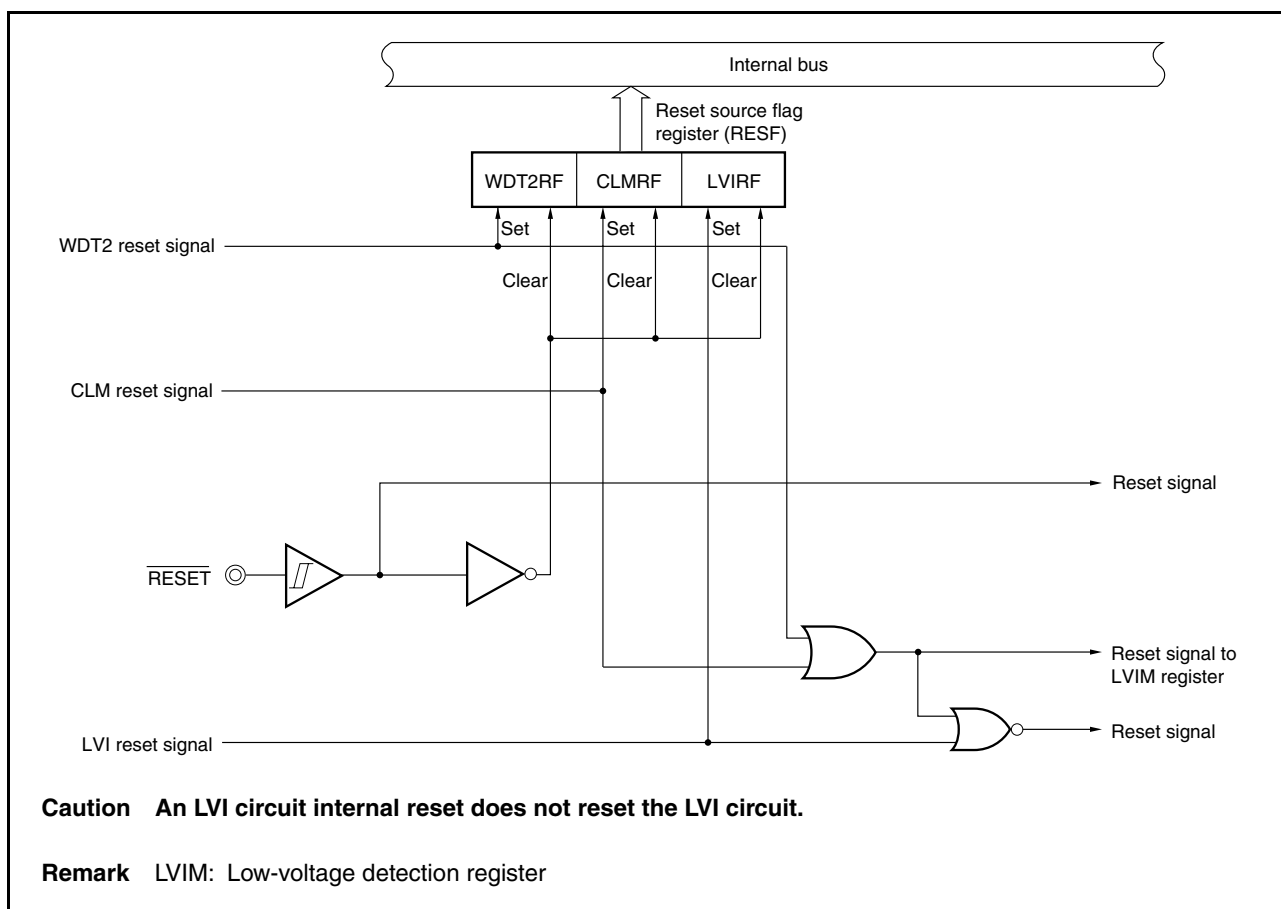
## 26.2 Registers to Check Reset Source

The V850ES/JG3-U and V850ES/JH3-U have four kinds of reset sources. After a reset has been released, the source of the reset that occurred can be checked with the reset source flag register (RESF).

### (1) Reset source flag register (RESF)

The RESF register is a special register that can be written only by a combination of specific sequences (see **3.4.7 Special registers**).

The RESF register indicates the source from which a reset signal is generated.

This register can be read or written in 8-bit or 1-bit units.

$\overline{\text{RESET}}$ pin input sets this register to 00H. The initial value differs if the source of reset is other than the $\overline{\text{RESET}}$ pin signal.

After reset: 00H**Note**    R/W    Address: FFFFF888H

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RESF | 0 | 0 | 0 | WDT2RF | 0 | 0 | CLMRF | LVIRF |

| WDT2RF | Reset signal from WDT2 |
|---|---|
| 0 | Not generated |
| 1 | Generated |

| CLMRF | Reset signal from CLM |
|---|---|
| 0 | Not generated |
| 1 | Generated |

| LVIRF | Reset signal from LVI |
|---|---|
| 0 | Not generated |
| 1 | Generated |

**Note** The value of the RESF register is set to 00H when a reset is executed via the $\overline{\text{RESET}}$ pin. When a reset is executed by the watchdog timer 2 (WDT2), low-voltage detector (LVI), or clock monitor (CLM), the reset flags of this register (WDT2RF bit, CLMRF bit, and LVIRF bit) are set. However, other sources are retained.

**Caution** Only "0" can be written to each bit of this register. If writing "0" conflicts with setting the flag (occurrence of reset), setting the flag takes precedence.

## 26.3 Operation

### 26.3.1 Reset operation via $\overline{\text{RESET}}$ pin

When a low level is input to the $\overline{\text{RESET}}$ pin, the system is reset, and each hardware unit is initialized.

When the level of the $\overline{\text{RESET}}$ pin is changed from low to high, the reset status is released.

**Table 26-1. Hardware Status on $\overline{\text{RESET}}$ Pin Input**

| Item | During Reset | After Reset |
|---|---|---|
| Main clock oscillator ($f_X$) | Oscillation stops | Oscillation starts |
| Subclock oscillator ($f_{XT}$) | Oscillation continues | |
| Internal oscillator | Oscillation stops | Oscillation starts |
| Peripheral clock ($f_X$ to $f_X/1,024$) | Operation stops | Operation starts after securing oscillation stabilization time |
| Internal system clock ($f_{CLK}$), CPU clock ($f_{CPU}$) | Operation stops | Operation starts after securing oscillation stabilization time (initialized to fXX/8) |
| CPU | Initialized | Program execution starts after securing oscillation stabilization time |
| Watchdog timer 2 | Operation stops (initialized to 0) | Counts up from 0 with internal oscillation clock as source clock. |
| Internal RAM | Undefined if power-on reset or CPU access and reset input conflict (data is damaged). Otherwise value immediately after reset input is retained. | |
| I/O lines (ports/alternate-function pins) | High impedance[Note] | |
| On-chip peripheral I/O registers | Initialized to specified status, OCDM register is set (01H). | |
| Other on-chip peripheral functions | Operation stops | Operation can be started after securing oscillation stabilization time |

**Note** When the power is turned on, the following pin may output an undefined level temporarily, even during reset.

- P10/ANO0 pin
- P11/ANO1 pin
- DDO pin (V850ES/JH3-U only)
- P53/SIF2/TIAB00/KR3/TOAB00/RTP03/DDO pin (V850ES/JG3-U only)

**Caution** **The OCDM register is initialized by the $\overline{\text{RESET}}$ pin input. Therefore, note with caution that, if a high level is input to the P56/INTP05/$\overline{\text{DRST}}$ pin after a reset release before the OCDM.OCDM0 bit is cleared, the on-chip debug mode is entered. For details, see CHAPTER 4 PORT FUNCTIONS.**

**Figure 26-2. Timing of Reset Operation by $\overline{\text{RESET}}$ Pin Input**
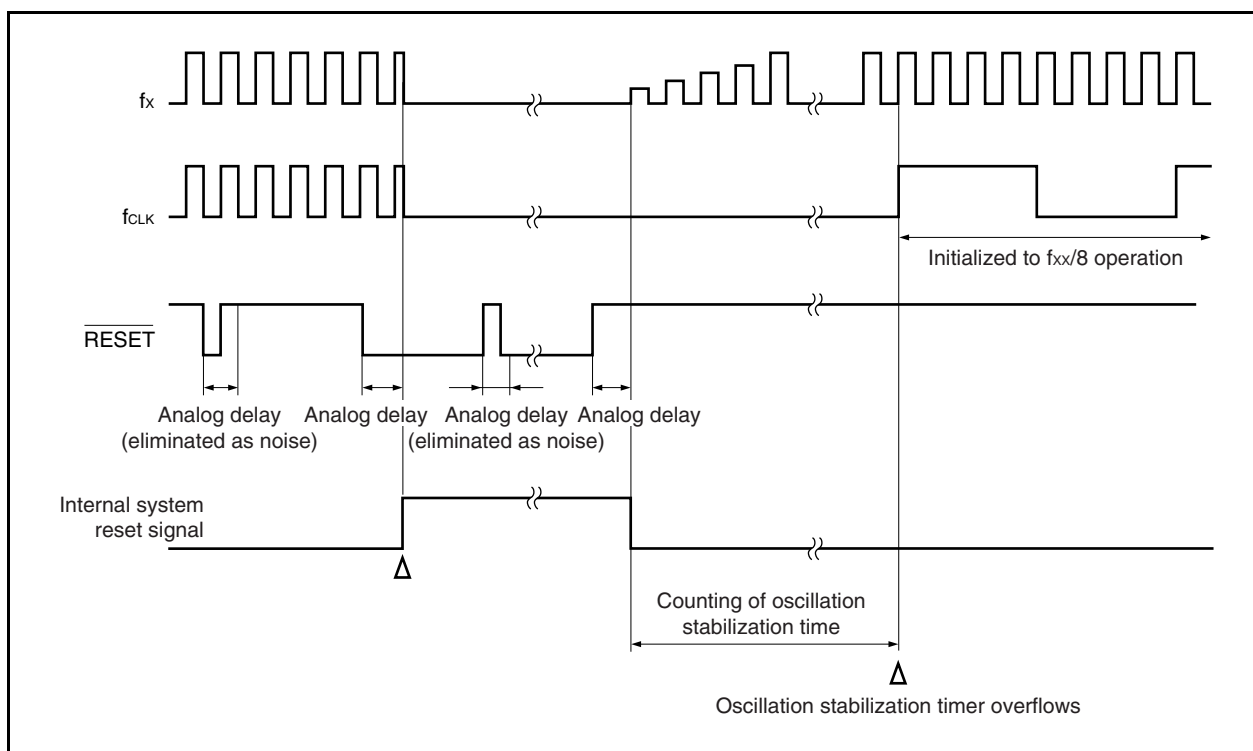


**Figure 26-3. Timing of Power-on Reset Operation**
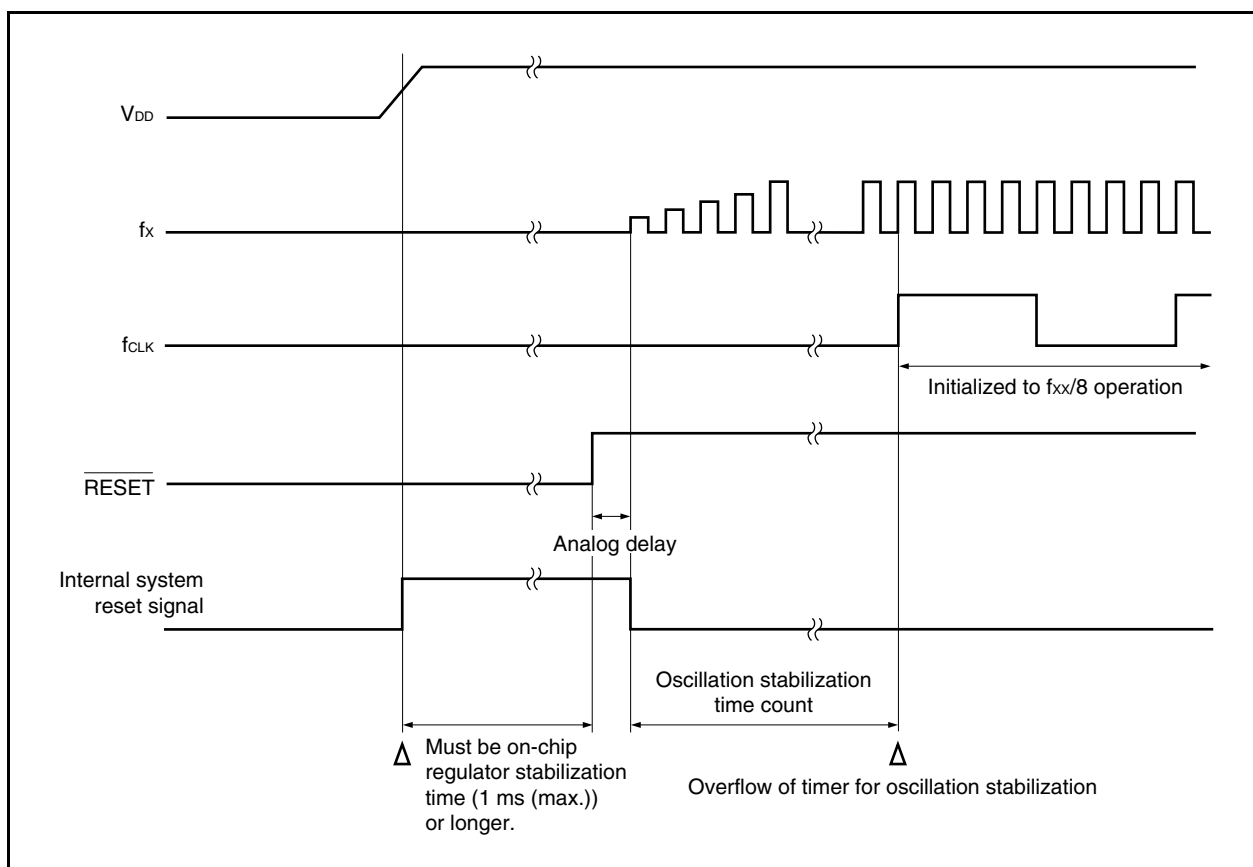
### 26.3.2 Reset operation by watchdog timer 2

When watchdog timer 2 is set to the reset operation mode due to overflow, upon watchdog timer 2 overflow (WDT2RES signal generation), a system reset is executed and the hardware is initialized to the initial status.

Following watchdog timer 2 overflow, the reset status is entered and lasts the predetermined time (analog delay), and the reset status is then automatically released.

The main clock oscillator is stopped during the reset period.

**Table 26-2. Hardware Status During Watchdog Timer 2 Reset Operation**

| Item | During Reset | After Reset |
|---|---|---|
| Main clock oscillator ($f_X$) | Oscillation stops | Oscillation starts |
| Subclock oscillator ($f_{XT}$) | Oscillation continues | |
| Internal oscillator | Oscillation stops | Oscillation starts |
| Peripheral clock ($f_{XX}$ to $f_{XX}$/1,024) | Operation stops | Operation starts after securing oscillation stabilization time |
| Internal system clock ($f_{XX}$), CPU clock ($f_{CPU}$) | Operation stops | Operation starts after securing oscillation stabilization time (initialized to $f_{XX}$/8) |
| CPU | Initialized | Program execution after securing oscillation stabilization time |
| Watchdog timer 2 | Operation stops (initialized to 0) | Counts up from 0 with internal oscillation clock as source clock. |
| Internal RAM | Undefined if power-on reset or CPU access and reset input conflict (data is damaged). Otherwise value immediately after reset input is retained. | |
| I/O lines (ports/alternate-function pins) | High impedance | |
| On-chip peripheral I/O register | Initialized to specified status, OCDM register retains its value. | |
| On-chip peripheral functions other than above | Operation stops | Operation can be started after securing oscillation stabilization time. |

**Figure 26-4.  Timing of Reset Operation by WDT2RES Signal Generation**
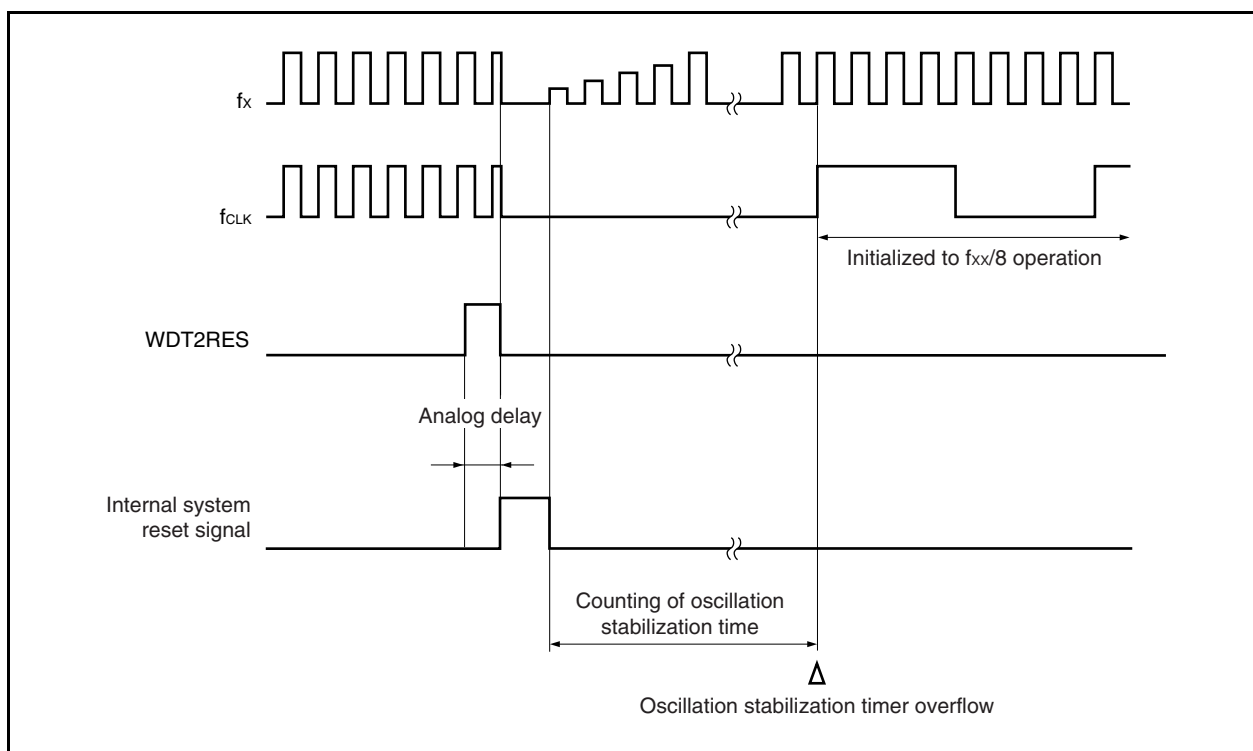
### 26.3.3 Reset operation by low-voltage detector

If the supply voltage falls below the voltage detected by the low-voltage detector when LVI operation is enabled, a system reset is executed (when the LVIM.LVIMD bit is set to 1), and the hardware is initialized to the initial status.

The reset status lasts from when a supply voltage drop has been detected until the supply voltage rises above the LVI detection voltage.

The main clock oscillator is stopped during the reset period.

When the LVIMD bit = 0, an interrupt request signal (INTLVI) is generated if a low voltage is detected.

**Table 26-3. Hardware Status During Reset Operation by Low-Voltage Detector**

| Item | During Reset | After Reset |
|------|--------------|-------------|
| Main clock oscillator ($f_X$) | Oscillation stops | Oscillation starts |
| Subclock oscillator ($f_{XT}$) | Oscillation continues | |
| Internal oscillator | Oscillation stops | Oscillation starts |
| Peripheral clock ($f_{XX}$ to $f_{XX}/1,024$) | Operation stops | Operation starts after securing oscillation stabilization time |
| Internal system clock ($f_{XX}$), CPU clock ($f_{CPU}$) | Operation stops | Operation starts after securing oscillation stabilization time (initialized to $f_{XX}/8$) |
| CPU | Initialized | Program execution starts after securing oscillation stabilization time |
| WDT2 | Operation stops (initialized to 0) | Counts up from 0 with internal oscillation clock as source clock. |
| Internal RAM | Undefined if power-on reset or CPU access and reset input conflict (data is damaged). Otherwise value immediately after reset input is retained. | |
| I/O lines (ports/alternate-function pins) | High impedance | |
| On-chip peripheral I/O register | Initialized to specified status, OCDM register retains its value. | |
| LVI | Operation stops | |
| On-chip peripheral functions other than above | Operation stops | Operation can be started after securing oscillation stabilization time. |

**Remark** For the reset timing of the low-voltage detector, see **CHAPTER 28 LOW-VOLTAGE DETECTOR (LVI)**.

### 26.3.4 Operation after reset release

After the reset is released, the main clock starts oscillation and oscillation stabilization time (OSTS register initial value: $2^{16}/f_X$) is secured, and the CPU starts program execution.

WDT2 immediately begins to operate after a reset has been released using the internal oscillation clock as a source clock.

**Figure 26-5. Operation After Reset Release**
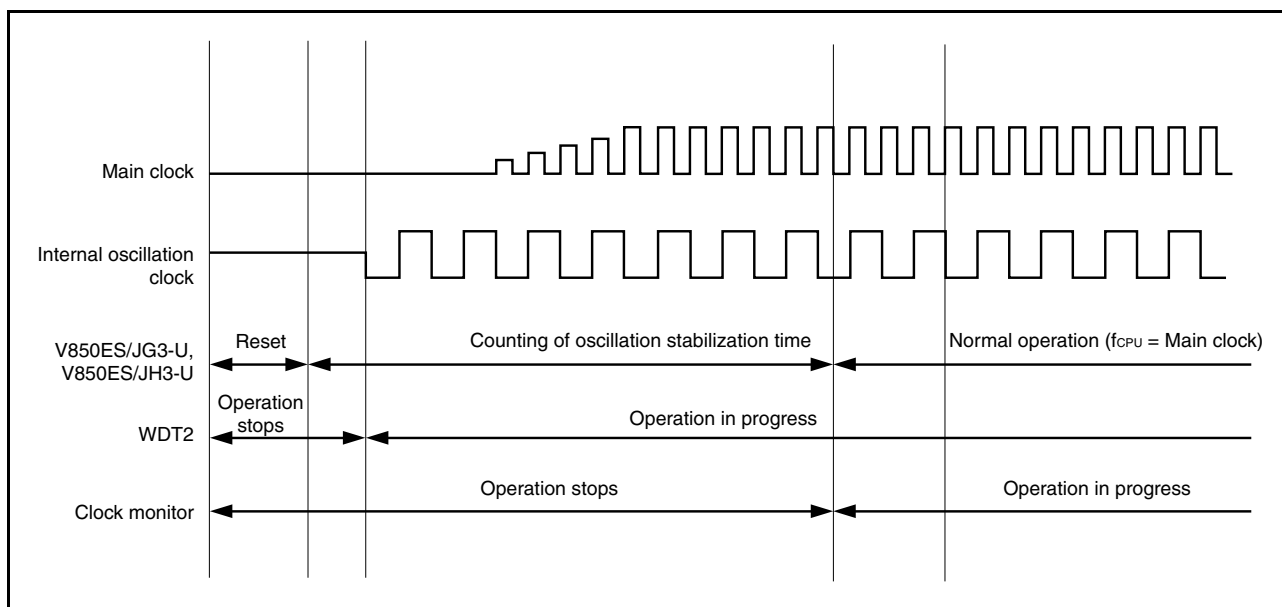


**(1) Emergent operation mode**

If an anomaly occurs in the main clock before oscillation stabilization time is secured, WDT2 overflows before executing the CPU program. At this time, the CPU starts program execution by using the internal oscillation clock as the source clock.

**Figure 26-6. Operation After Reset Release**
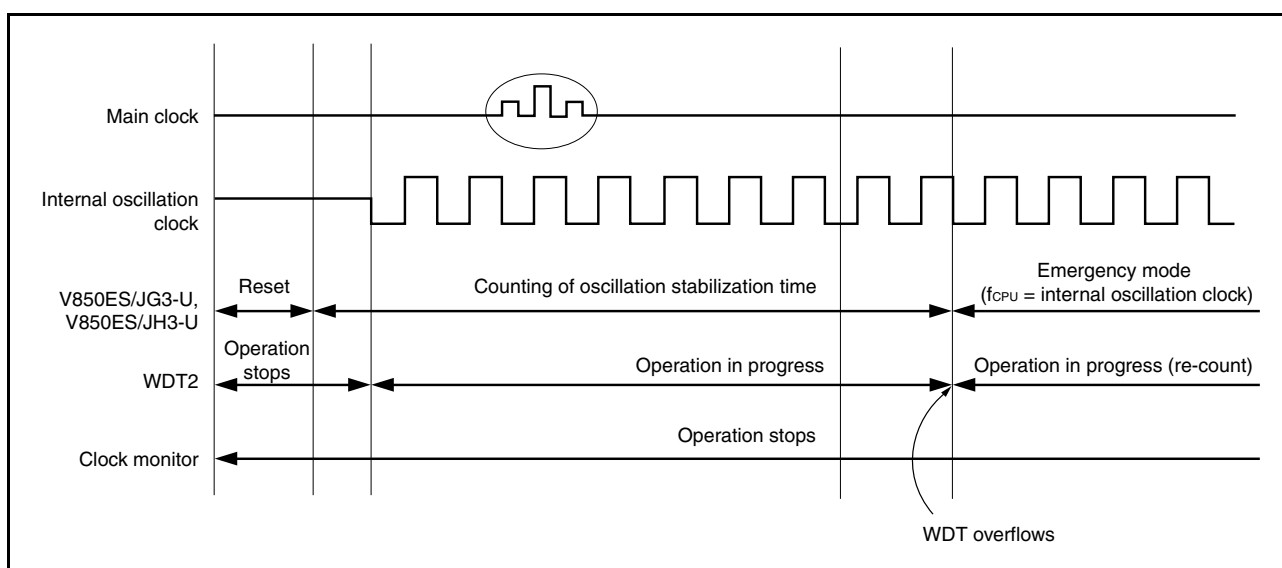


The CPU operation clock states can be checked with the CPU operation clock status register (CCLS).

### 26.3.5 Reset function operation flow
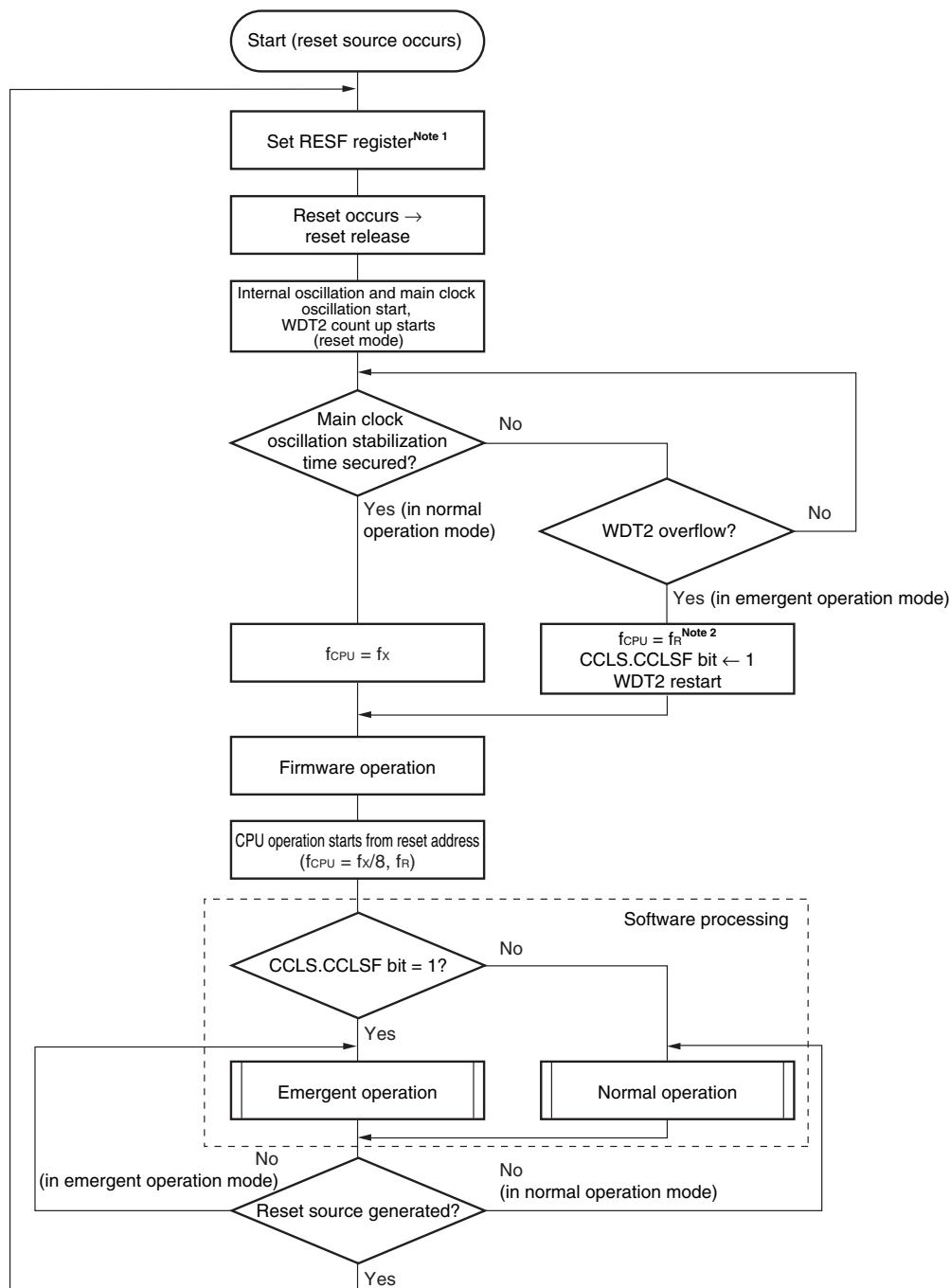


**Notes 1.** Bit to be set differs depending on the reset source.

| Reset Source | WDT2RF Bit | CRMRF Bit | LVIRF Bit |
|---|---|---|---|
| $\overline{\text{RESET}}$ pin | 0 | 0 | 0 |
| WDT2 | 1 | Value before reset is retained. | Value before reset is retained. |
| CLM | Value before reset is retained. | 1 | Value before reset is retained. |
| LVI | Value before reset is retained. | Value before reset is retained. | 1 |

**2.** The internal oscillator cannot be stopped.

# CHAPTER 27  CLOCK MONITOR

## 27.1  Functions

The clock monitor samples the main clock by using the internal oscillation clock and generates a reset request signal when oscillation of the main clock is stopped.

Once the operation of the clock monitor has been enabled by an operation enable flag, it cannot be cleared to 0 by any means other than reset.

When a reset by the clock monitor occurs, the RESF.CLMRF bit is set.  For details on the RESF register, see **26.2 Registers to Check Reset Source**.

The clock monitor automatically stops under the following conditions.

- During oscillation stabilization time after STOP mode is released
- When the main clock is stopped (from when the PCC.MCK bit = 1 during subclock operation, until the PCC.CLS bit = 0 during main clock operation)
- When the sampling clock (internal oscillation clock) is stopped
- When the CPU operates with the internal oscillation clock

## 27.2  Configuration

The clock monitor includes the following hardware.

**Table 27-1.  Configuration of Clock Monitor**

| Item | Configuration |
|---|---|
| Control register | Clock monitor mode register (CLM) |

**Figure 27-1.  Timing of Reset via $\overline{\text{RESET}}$ Pin Input**
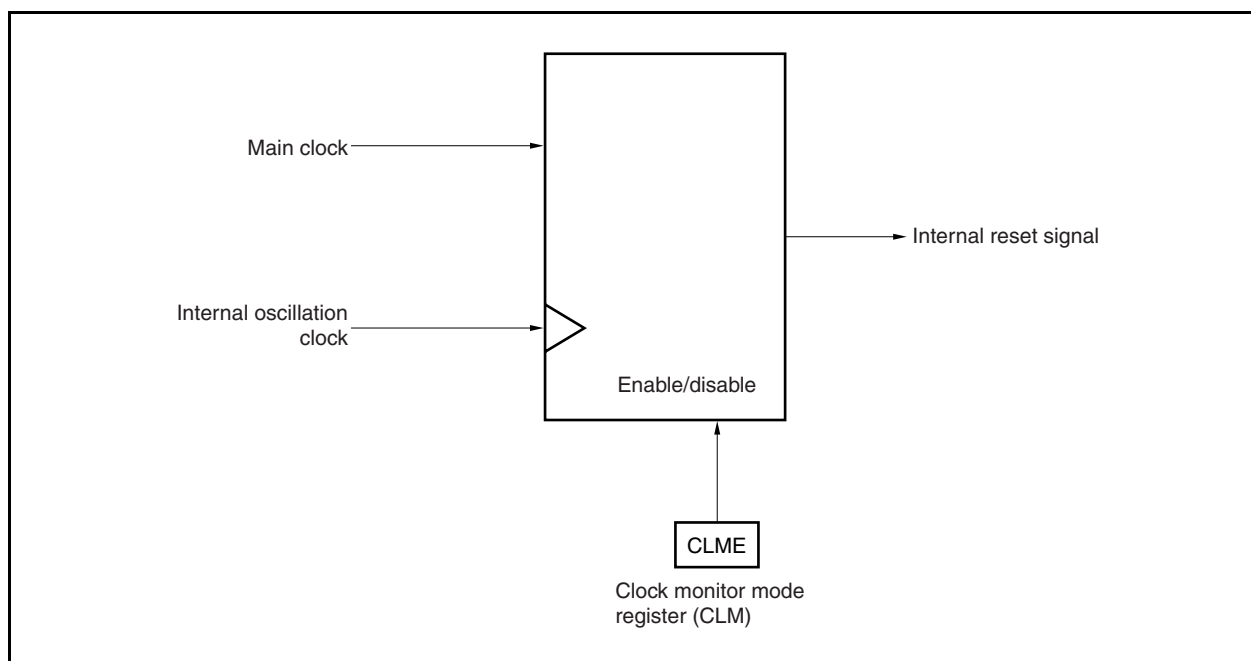
## 27.3  Register

The clock monitor is controlled by the clock monitor mode register (CLM).

**(1)  Clock monitor mode register (CLM)**

The CLM register is a special register.  This can be written only in a special combination of sequences (see **3.4.7 Special registers**).

This register is used to set the operation mode of the clock monitor.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H    R/W    Address: FFFFF870H

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| CLM | 0 | 0 | 0 | 0 | 0 | 0 | 0 | CLME |

| CLME | Clock monitor operation enable or disable |
|---|---|
| 0 | Disable clock monitor operation. |
| 1 | Enable clock monitor operation. |

**Cautions  1.  Once the CLME bit has been set to 1, it cannot be cleared to 0 by any means other than reset.**

**2.  When a reset by the clock monitor occurs, the CLME bit is cleared to 0 and the RESF.CLMRF bit is set to 1.**

**3.  Be sure to set bits 7 to 1 to "0".**

## 27.4  Operation

This section explains the functions of the clock monitor.  The start and stop conditions are as follows.

<Start condition>
    Enabling operation by setting the CLM.CLME bit to 1

<Stop conditions>
- While oscillation stabilization time is being counted after STOP mode is released
- When the main clock is stopped (from when PCC.MCK bit = 1 during subclock operation to when PCC.CLS bit = 0 during main clock operation)
- When the sampling clock (internal oscillation clock) is stopped
- When the CPU operates using the internal oscillation clock

**Table 27-2.  Operation Status of Clock Monitor**
**(When CLM.CLME Bit = 1, During Internal Oscillation Clock Operation)**

| CPU Operating Clock | Operation Mode | Status of Main Clock | Status of Internal Oscillation Clock | Status of Clock Monitor |
|---|---|---|---|---|
| Main clock | HALT mode | Oscillates | Oscillates[Note 1] | Operates[Note 2] |
| | IDLE1, IDLE2 modes | Oscillates | Oscillates[Note 1] | Operates[Note 2] |
| | STOP mode | Stops | Oscillates[Note 1] | Stops |
| Subclock (PCC.MCK = 0) | Sub-IDLE mode | Oscillates | Oscillates[Note 1] | Operates[Note 2] |
| Subclock (PCC.MCK = 1) | Sub-IDLE mode | Stops | Oscillates[Note 1] | Stops |
| Internal oscillation clock | – | Stops | Oscillates[Note 3] | Stops |
| During reset | – | Stops | Stops | Stops |

**Notes 1.** The internal oscillator can be stopped by setting the RCM.RSTOP bit to 1.
   **2.** The clock monitor is stopped while the internal oscillator is stopped.
   **3.** The internal oscillator cannot be stopped by software.

**(1) Operation when main clock oscillation is stopped (CLME bit = 1)**

If oscillation of the main clock is stopped when the CLME bit = 1, an internal reset signal is generated as shown in Figure 27-2.

**Figure 27-2.  Reset Period Due to That Oscillation of Main Clock Is Stopped**
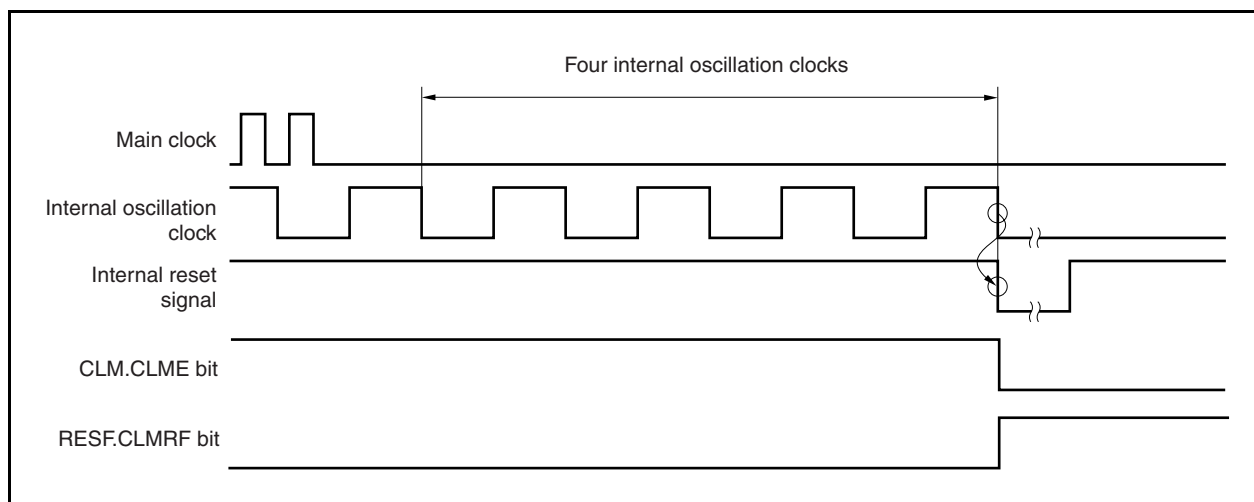


**(2) Clock monitor status after $\overline{\text{RESET}}$ input**

$\overline{\text{RESET}}$ input clears the CLM.CLME bit to 0 and stops the clock monitor operation.  When CLME bit is set to 1 by software at the end of the oscillation stabilization time of the main clock, monitoring is started.

**Figure 27-3.  Clock Monitor Status After $\overline{\text{RESET}}$ Input**
**(CLM.CLME bit = 1 is set after $\overline{\text{RESET}}$ input and at the end of main clock oscillation stabilization time)**
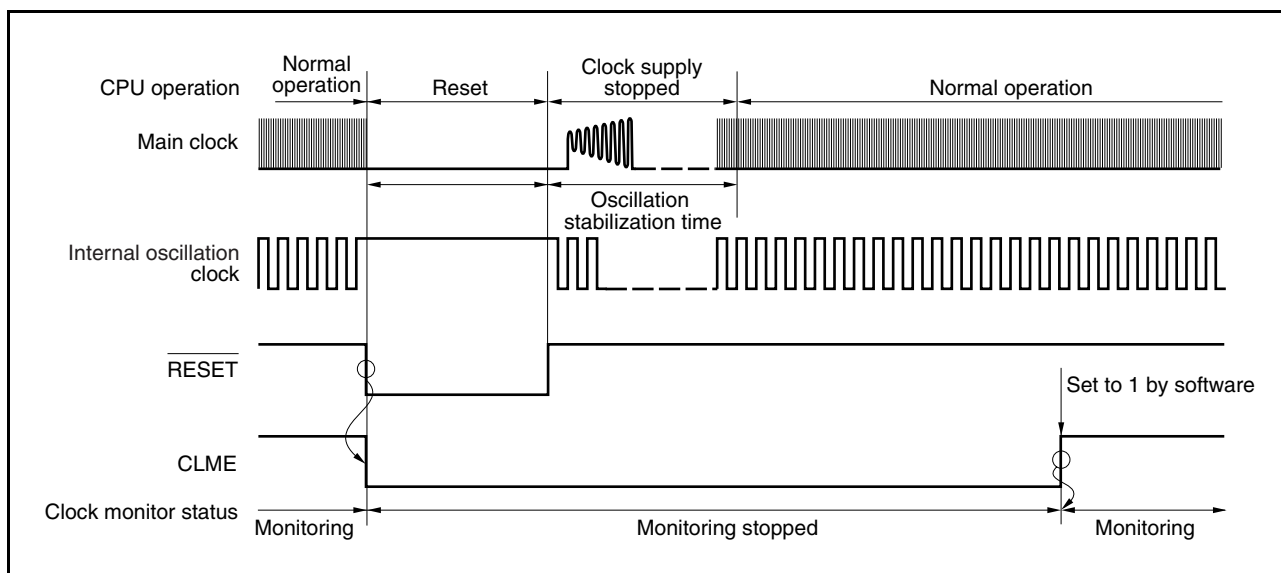
**(3) Operation in STOP mode or after STOP mode is released**

If the STOP mode is set with the CLM.CLME bit = 1, the monitor operation is stopped in the STOP mode and while the oscillation stabilization time is being counted. After the oscillation stabilization time, the monitor operation is automatically started.

**Figure 27-4. Operation in STOP Mode or After STOP Mode Is Released**
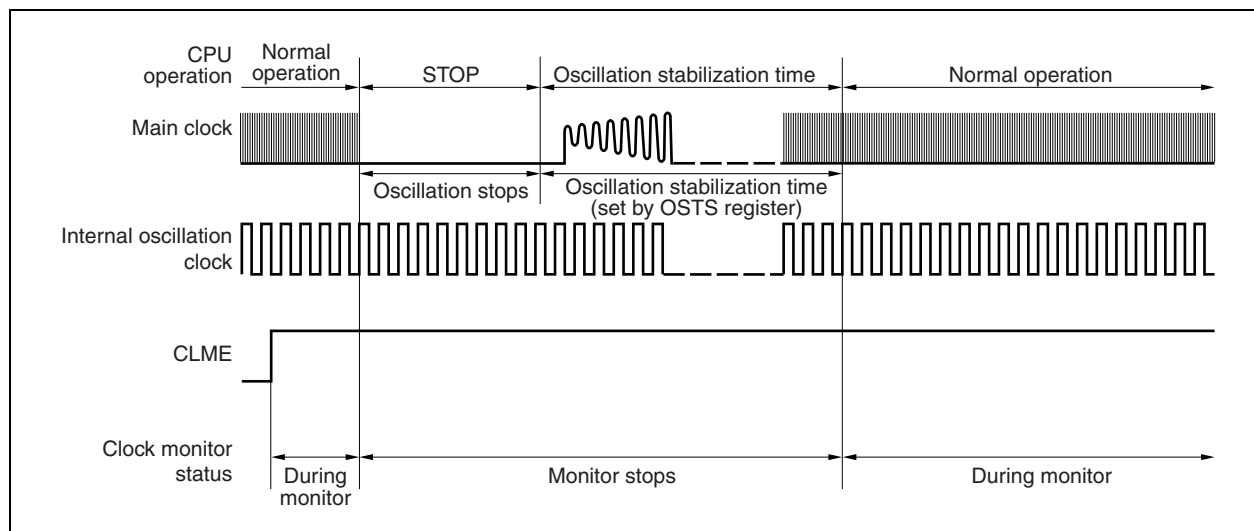


**(4) Operation when main clock is stopped (arbitrary)**

During subclock operation (PCC.CLS bit = 1) or when the main clock is stopped by setting the PCC.MCK bit to 1, the monitor operation is stopped until the main clock operation is started (PCC.CLS bit = 0). The monitor operation is automatically started when the main clock operation is started.

**Figure 27-5. Operation When Main Clock Is Stopped (Arbitrary)**
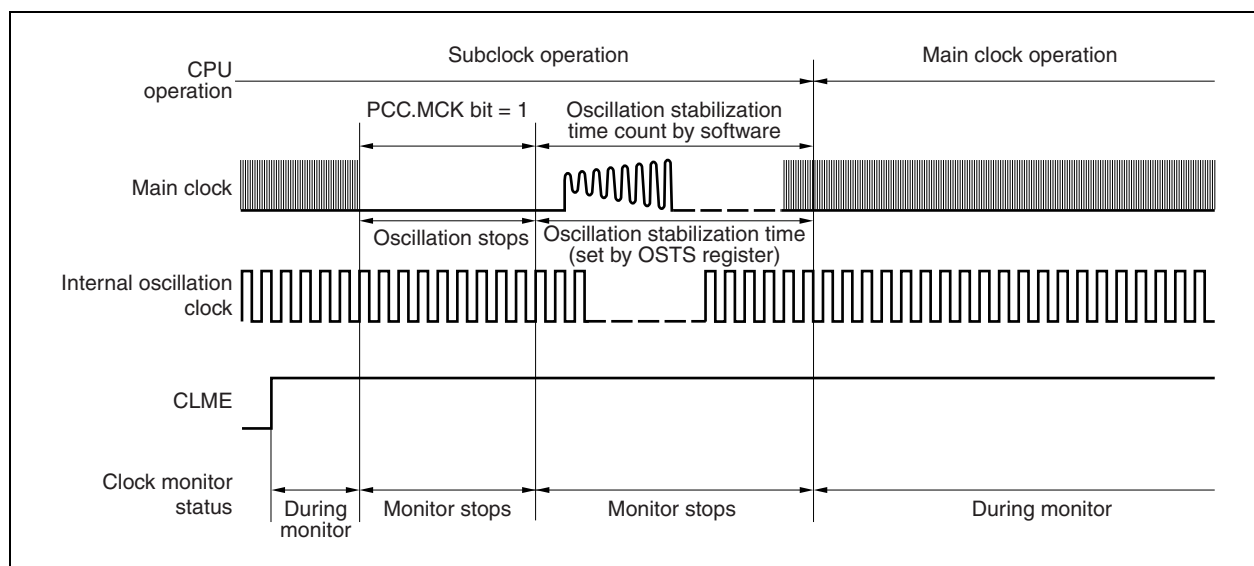


**(5) Operation while CPU is operating on internal oscillation clock (CCLS.CCLSF bit = 1)**

The monitor operation is not stopped when the CCLSF bit is 1, even if the CLME bit is set to 1.

## CHAPTER  28  LOW-VOLTAGE  DETECTOR  (LVI)

### 28.1  Functions

The low-voltage detector (LVI) has the following functions.

- If the interrupt occurrence at low voltage detection is selected, the low-voltage detector continuously compares the supply voltage ($V_{DD}$) and the detected voltage ($V_{LVI}$), and generates an internal interrupt signal when the supply voltage drops or rises across the detected voltage.
- If the reset occurrence at low voltage detection is selected, the low-voltage detector generates an interrupt reset signal when the supply voltage ($V_{DD}$) drops across the detected voltage ($V_{LVI}$).
- Interrupt or reset signal can be selected by software.
- Can operate in STOP mode.

If the low-voltage detector is used to generate a reset signal, the RESF.LVIRF bit is set to 1 when the reset signal is generated.  For details of RESF register, see **26.2  Registers to Check Reset Source**.

### 28.2  Configuration

The block diagram of the low-voltage detector is shown below.

**Figure 28-1.  Block Diagram of Low-Voltage Detector**
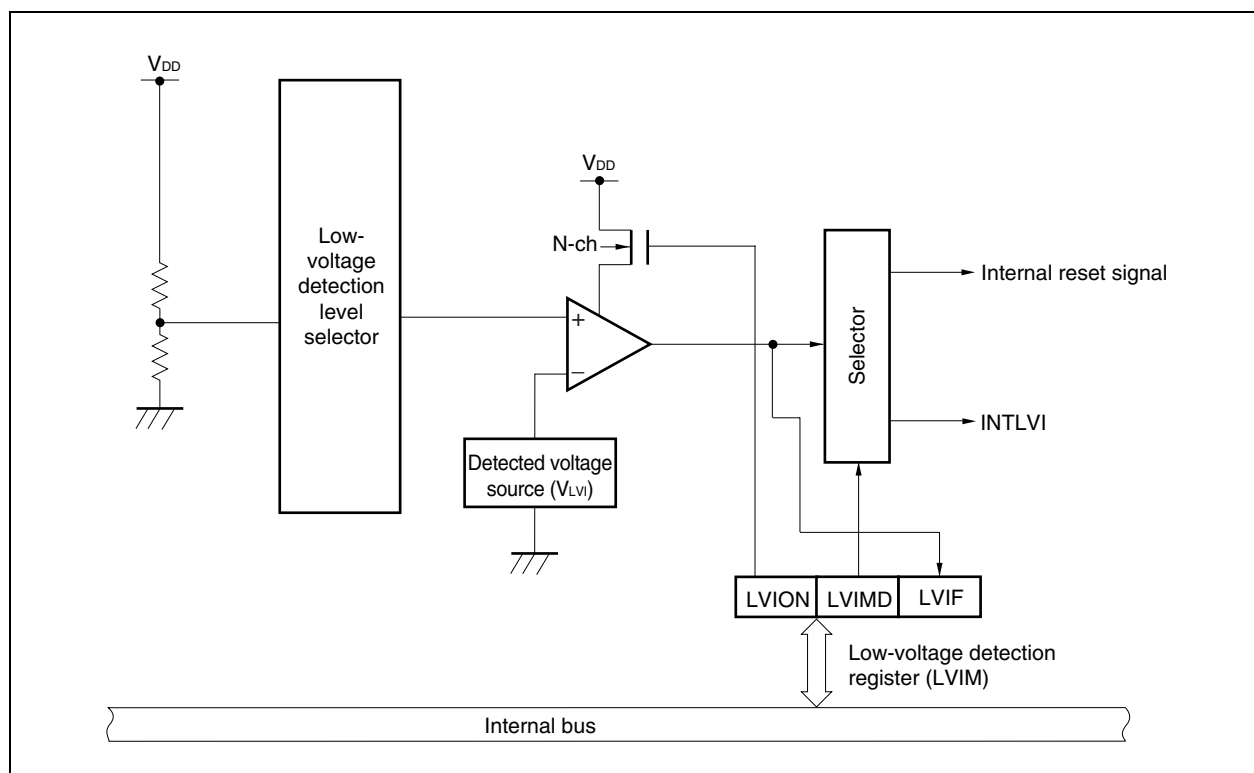
## 28.3  Registers

The low-voltage detector is controlled by the following registers.

- Low-voltage detection register (LVIM)
- Internal RAM data status register (RAMS)

**(1)  Low-voltage detection register (LVIM)**

The LVIM register is a special register.  This can be written only in the special combination of the sequences (see **3.4.7  Special registers**).

The LVIM register is used to enable or disable low-voltage detection, and to set the operation mode of the low-voltage detector.

This register can be read or written in 8-bit or 1-bit units.  However, the LVIF bit is read-only.

After reset: 00H[Note 1]    R/W    Address: FFFFF890H

|  | <7> | 6 | 5 | 4 | 3 | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| LVIM | LVION | 0 | 0 | 0 | 0 | 0 | LVIMD | LVIF |

| LVION | Low-voltage detection operation enable or disable |
|---|---|
| 0 | Disable operation. |
| 1 | Enable operation. |

| LVIMD | Selection of operation mode of low-voltage detection |
|---|---|
| 0 | Generates interrupt signal INTLVI when the supply voltage drops or rises across the detection voltage value. |
| 1 | Generates internal reset signal LVIRES when the supply voltage drops across the detected voltage value. |

| LVIF[Note 2] | Low-voltage detection flag |
|---|---|
| 0 | When supply voltage > detected voltage, or when operation is disabled |
| 1 | Supply voltage of connected power supply < detected voltage |

**Notes 1.**  Reset by low-voltage detection: 82H

Reset due to other source: 00H

**2.**  After the LVI operation has started (LVION bit = 1) or when INTLVI has occurred, confirm the supply voltage state using the LVIF bit.

**Cautions 1.  When the LVION and LVIMD bits to 1, the low-voltage detector cannot be stopped until the reset request due to other than the low-voltage detection is generated.**

**2.  When the LVION bit is set to 1, the comparator in the LVI circuit starts operating. Wait 0.2 ms or longer by software before checking the voltage at the LVIF bit after the LVION bit is set.**

**3.  Be sure to set bits 6 to 2 to "0".**

**(2)  Internal RAM data status register (RAMS)**

The RAMS register is a special register.  This can be written only in a special combination of sequences (see **3.4.7 Special registers**).

This register is a flag register that indicates whether the internal RAM is valid or not.

This register can be read or written in 8-bit or 1-bit units.

The set/clear conditions for the RAMF bit are shown below.

- Setting conditions:   Detection of voltage lower than specified level
                                 Set by instruction
- Clearing condition:  Writing of 0 in specific sequence

After reset: 01H[Note]        R/W        Address: FFFFF892H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| RAMS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RAMF |

| RAMF | Internal RAM voltage detection |
|---|---|
| 0 | Voltage lower than RAM retention voltage is not detected. |
| 1 | Voltage lower than RAM retention voltage is detected. |

**Note**   This register is reset only when a voltage drop below the RAM retention voltage is detected.

## 28.4 Operation

Depending on the setting of the LVIM.VIMD bit, an interrupt signal (INTLVI) or an internal reset signal is generated. How to specify each operation is described below, together with timing charts.

### 28.4.1 To use for internal reset signal

<To start operation>
<1> Mask the interrupt of LVI.
<2> Set the LVIM.LVION bit to 1 (to enable operation).
<3> Insert a wait cycle of 0.2 ms (max.) or more by software.
<4> By using the LVIM.LVIF bit, check if the supply voltage > detected voltage.
<5> Set the LVIMD bit to 1 (to generate an internal reset signal).

**Caution   If the LVIMD bit is set to 1, the content of the LVIM register cannot be changed until a reset request other than LVI is generated.**

**Figure 28-2.  Operation Timing of Low-Voltage Detector (LVIMD Bit = 1)**
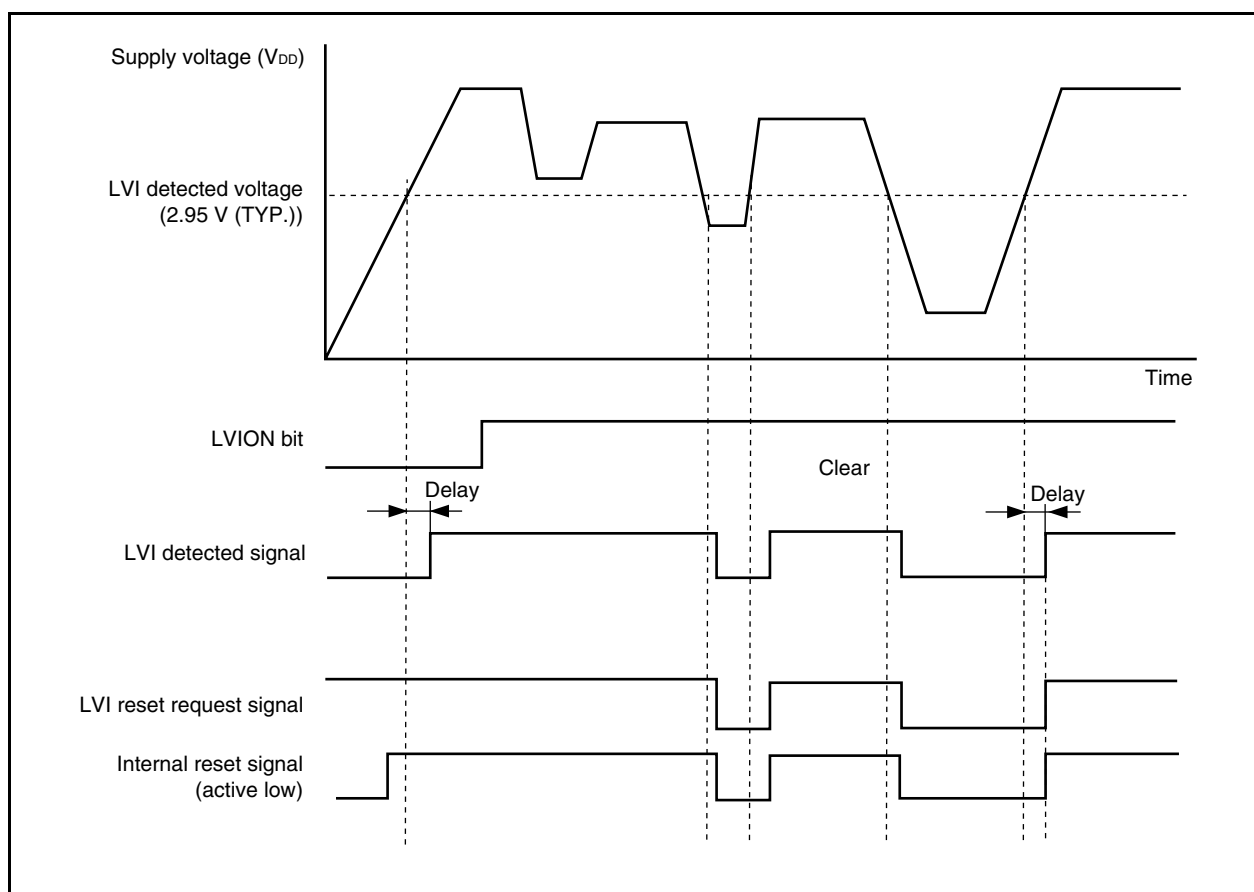
### 28.4.2  To use for interrupt

<To start operation>

<1> Mask the interrupt of LVI.

<2> Set the LVIM.LVION bit to 1 (to enable operation).

<3> Insert a wait cycle of 0.2 ms (max.) or more by software.

<4> By using the LVIM.LVIF bit, check if the supply voltage > detected voltage.

<5> Clear the interrupt request flag of LVI.

<6> Unmask the interrupt of LVI.

<To stop operation>

Clear the LVION bit to 0.

**Figure 28-3.  Operation Timing of Low-Voltage Detector (LVIMD Bit = 0)**



**Note**  Since the LVION bit is the initial value (operation disabled) due to the external reset input, no INTLVI interrupts occur.

**Caution**  **When the INTLVI signal is generated, confirm, using the LVIM.LVIF bit, whether the INTLVI signal is generated due to a supply voltage drop or rise across the detected voltage.**
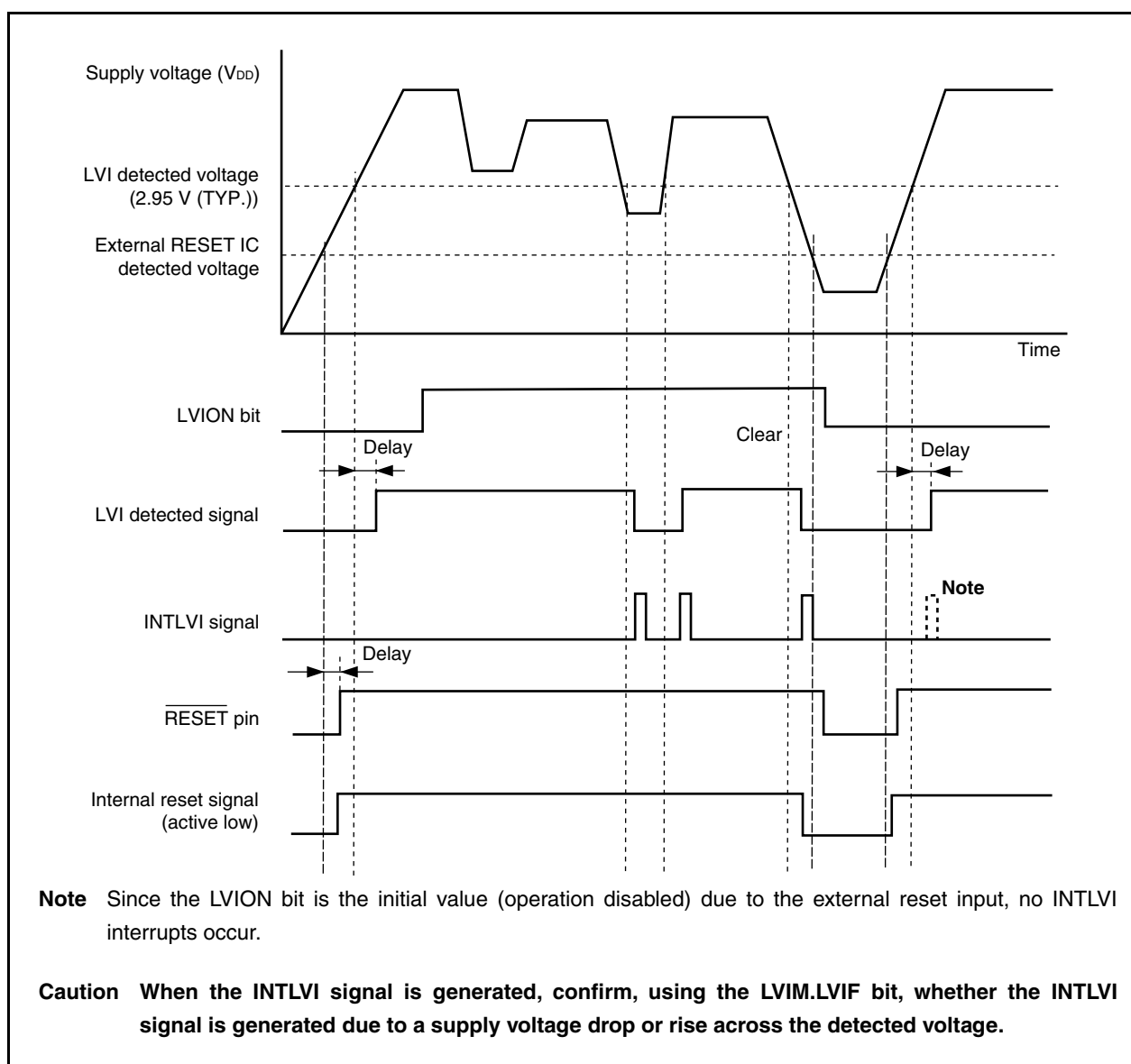
## 28.5  RAM Retention Voltage Detection Operation

The supply voltage and detected voltage are compared.  When the supply voltage drops below the detected voltage (including on power application), the RAMS.RAMF bit is set to 1.

**Figure 28-4.  Operation Timing of RAM Retention Voltage Detection Function**



**Remarks 1.** The RAMF bit is set to 1 if the supply voltage drops under the minimum RAM retention voltage (2.0 V (TYP.)).

**2.** The RAMF bit operates regardless of the $\overline{\text{RESET}}$ pin status.
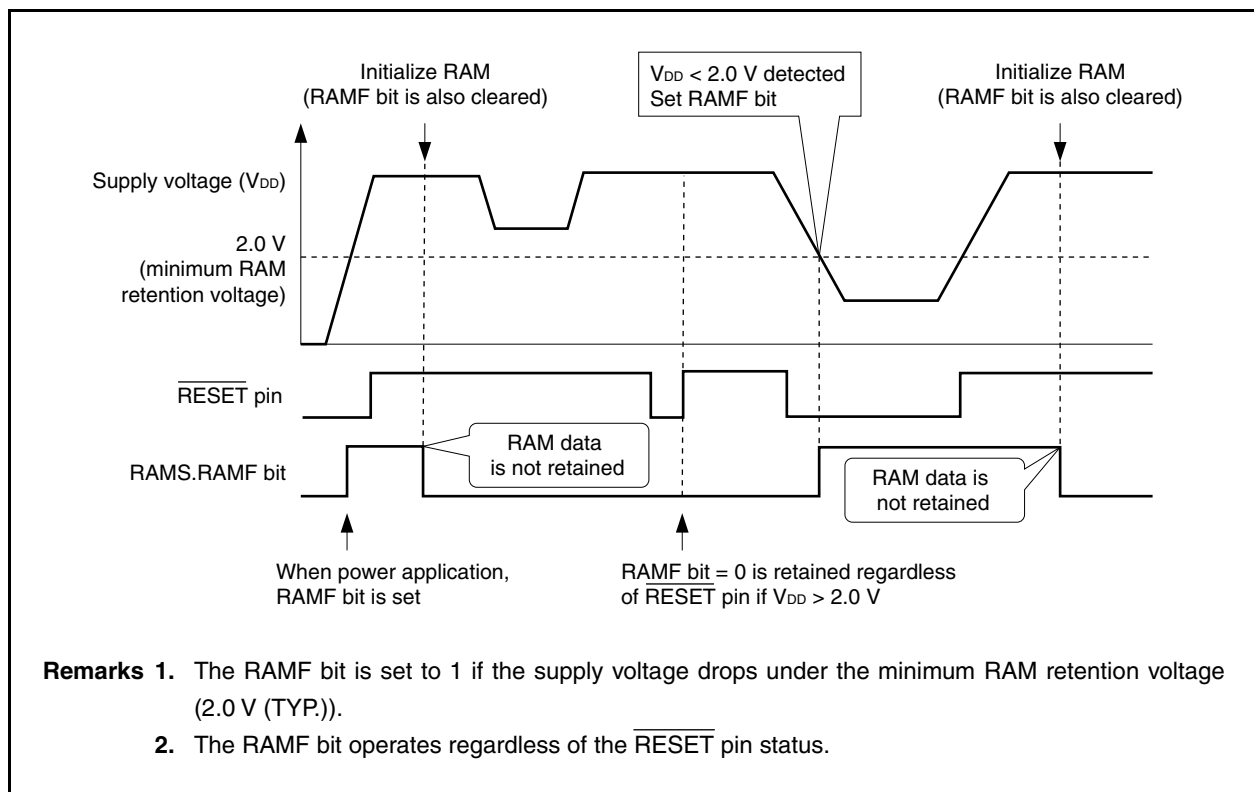
# CHAPTER 29  CRC FUNCTION

## 29.1  Functions

- CRC operation circuit for detection of data block errors
- Generation of 16-bit CRC code using a CRC-CCITT ($X^{16} + X^{12} + X^5 + 1$) generation polynomial for blocks of data of any length in 8-bit units
- CRC code is set to the CRCD data register each time 1-byte data is transferred to the CRCIN register, after the initial value is set to the CRCD register.

## 29.2  Configuration

The CRC function includes the following hardware.

**Table 29-1.  CRC Configuration**

| Item | Configuration |
|---|---|
| Control registers | CRC input register (CRCIN) |
| | CRC data register (CRCD) |

**Figure 29-1.  Block Diagram of CRC Register**
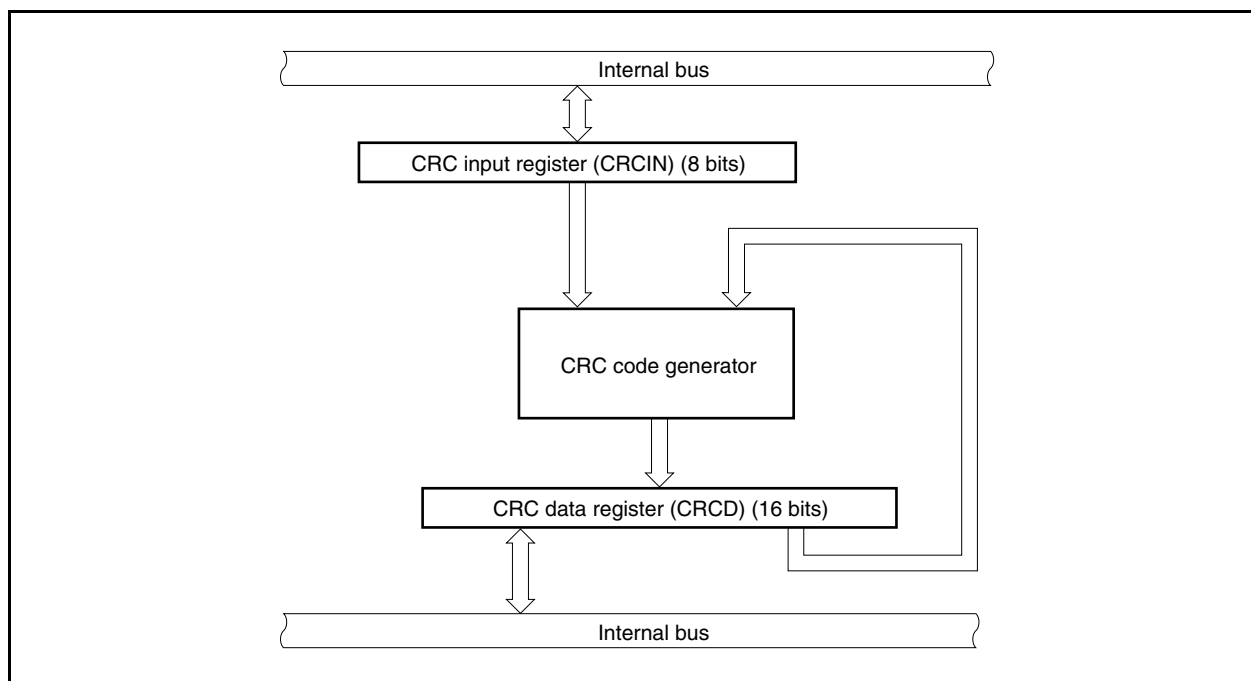
## 29.3  Registers

**(1)  CRC input register (CRCIN)**

The CRCIN register is an 8-bit register for setting data.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

After reset:  00H    R/W    Address:  FFFFF310H

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CRCIN | | | | | | | | |
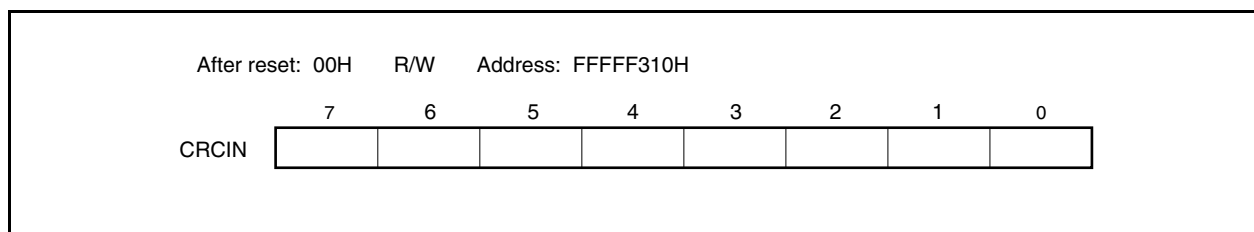
**(2)  CRC data register (CRCD)**

The CRCD register is a 16-bit register that stores the CRC-CCITT operation results.

This register can be read or written in 16-bit units.

Reset sets this register to 0000H.

**Caution   Accessing the CRCD register is prohibited in the following statuses.  For details, see 3.4.8 (2) Accessing specific on-chip peripheral I/O registers.**

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

After reset:  0000H    R/W    Address:  FFFFF312H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CRCD | | | | | | | | | | | | | | | | |
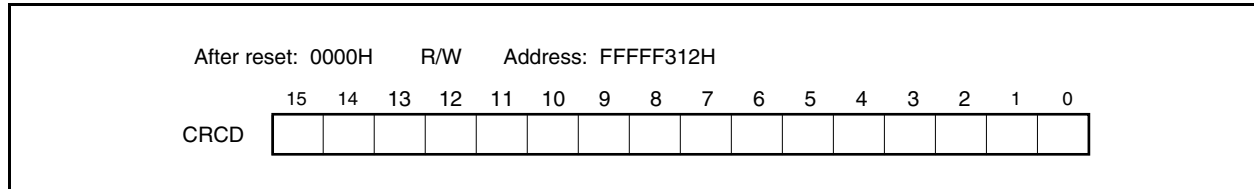
## 29.4  Operation

An example of the CRC operation circuit is shown below.

**Figure 29-2.  CRC Operation Circuit Operation Example (LSB First)**
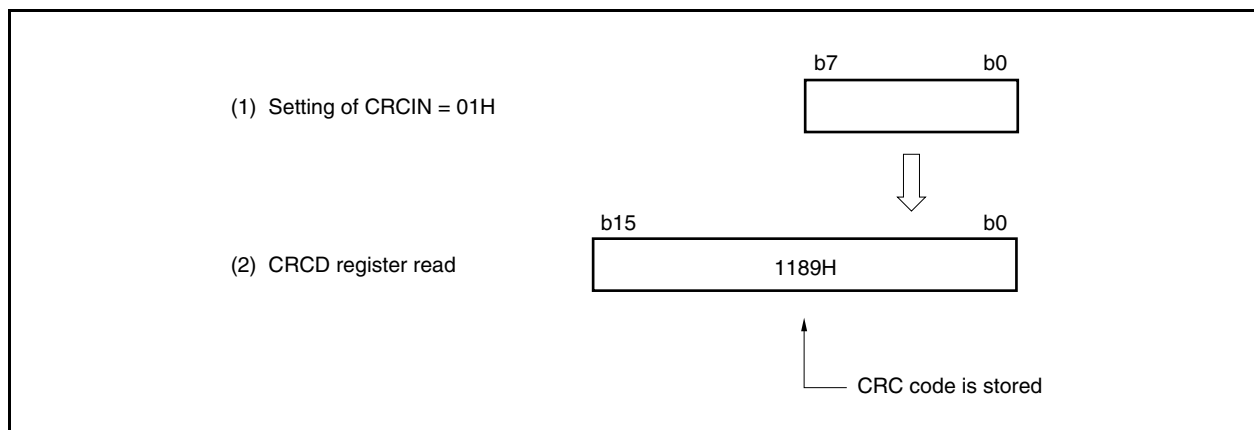


The code when 01H is sent LSB first is (1000 0000).  Therefore, the CRC code from generation polynomial $X^{16} + X^{12} + X^5 + 1$ becomes the remainder when (1000 0000) $X^{16}$ is divided by (1 0001 0000 0010 0001) using the modulo-2 operation formula.

The modulo-2 operation is performed based on the following formula.

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 0$$
$$-1 = 1$$



Therefore, the CRC code becomes $\underbrace{1001}_{9}\ \underbrace{0001}_{8}\ \underbrace{1000}_{1}\ \underbrace{1000}_{1}$ since LSB first is used, this corresponds to 1189H in hexadecimal notation.

## 29.5  Usage Method

How to use the CRC logic circuit is described below.

**Figure 29-3.  CRC Operation Flow**



[Basic usage method]
    <1> Write 0000H to the CRCD register.
    <2> Write the required quantity of data to the CRCIN register.
    <3> Read the CRCD register.
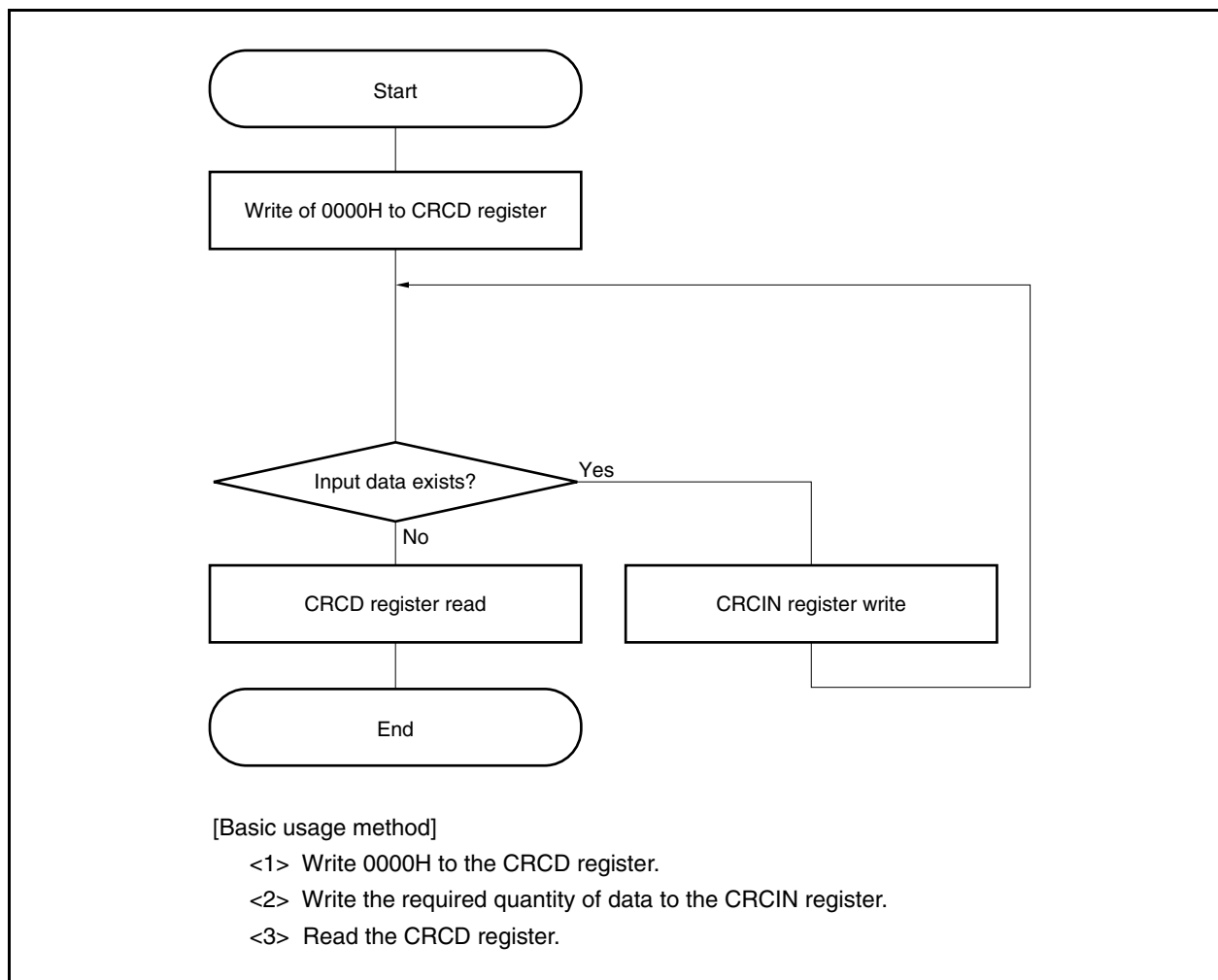
Communication errors can easily be detected if the CRC code is transmitted/received along with transmit/receive data when transmitting/receiving data consisting of several bytes.

The following is an illustration using the transmission of 12345678H (0001 0010 0011 0100 0101 0110 0111 1000B) LSB-first as an example.

**Figure 29-4. CRC Transmission Example**



Transmit/receive data
(12345678H)

CRC code
(08F6H)
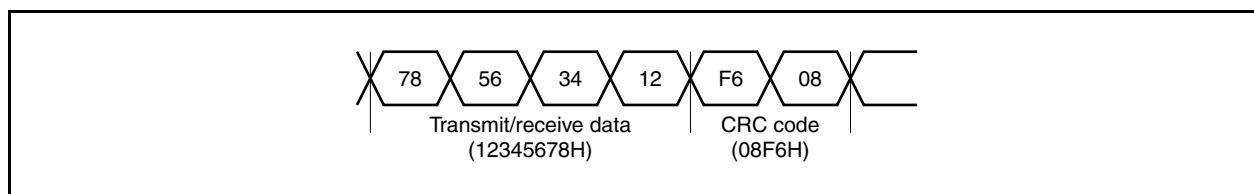
Setting procedure on transmitting side
 <1> Write the initial value 0000H to the CRCD register.
 <2> Write the 1 byte of data to be transmitted first to the transmit buffer register.  (At this time, also write the same data to the CRCIN register.)
 <3> When transmitting several bytes of data, write the same data to the CRCIN register each time transmit data is written to the transmit buffer register.
 <4> After all the data has been transmitted, write the contents of the CRCD register (CRC code) to the transmit buffer register and transmit them.  (Since this is LSB first, transmit the data starting from the lower bytes, then the higher bytes.)

Setting procedure on receiving side
 <1> Write the initial value 0000H to the CRCD register.
 <2> When reception of the first 1 byte of data is complete, write that receive data to the CRCIN register.
 <3> If receiving several bytes of data, write the receive data to the CRCIN register upon every reception completion.  (In the case of normal reception, when all the receive data has been written to the CRCIN register, the contents of the CRCD register on the receiving side and the contents of the CRCD register on the transmitting side are the same.)
 <4> Next, the CRC code is transmitted from the transmitting side, so write this data to the CRCIN register similarly to receive data.
 <5> When reception of all the data, including the CRC code, has been completed, reception was normal if the contents of the CRCD register are 0000H.  If the contents of the CRCD register are other than 0000H, this indicates a communication error, so transmit a resend request to the transmitting side.

# CHAPTER 30  REGULATOR

## 30.1  Overview

The V850ES/JG3-U and V850ES/JH3-U include a regulator to reduce power consumption and noise.

This regulator supplies a stepped-down $V_{DD}$ power supply voltage to the oscillator block and internal logic circuits (except the A/D converter, D/A converter, and output buffers).

**Figure 30-1.  Regulator**



**Caution    Be sure to use in $V_{DD}$ = $UV_{DD}$ = $EV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$.**
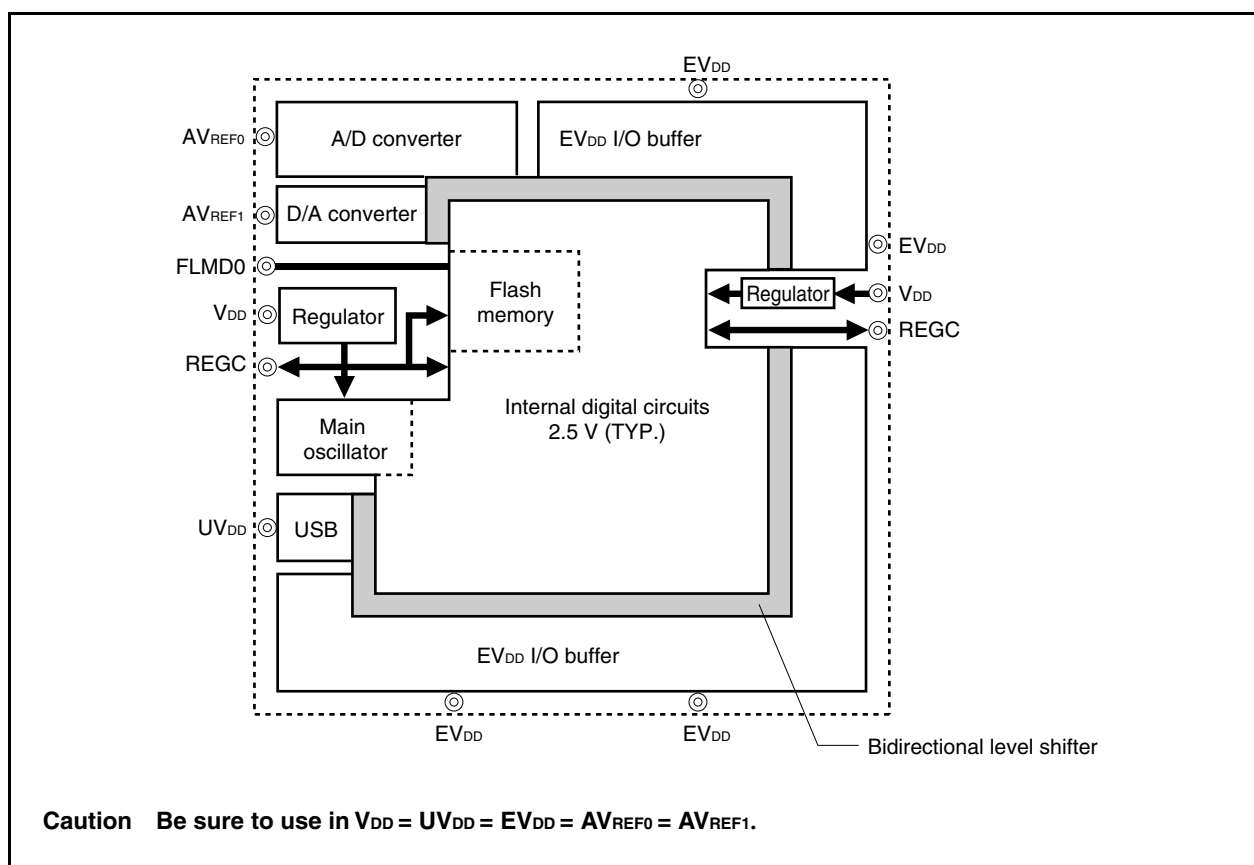
## 30.2 Operation

The regulators of the V850ES/JG3-U and V850ES/JH3-U always operate in any mode (normal operation mode, HALT mode, IDLE1 mode, IDLE2 mode, STOP mode, subclock operation mode, sub IDLE mode, or during reset).

Be sure to connect a capacitor (4.7 $\mu$F (recommended value)) to the REGC pin[Note] to stabilize the regulator output.

A diagram of the regulator pin connection method is shown below.

**Note** There are two REGC pins.

**Figure 30-2. REGC Pin Connection**



**Caution** **The V850ES/JG3-U and V850ES/JH3-U have two regulators. Therefore, connect each of the two REGC pins to a capacitor.**
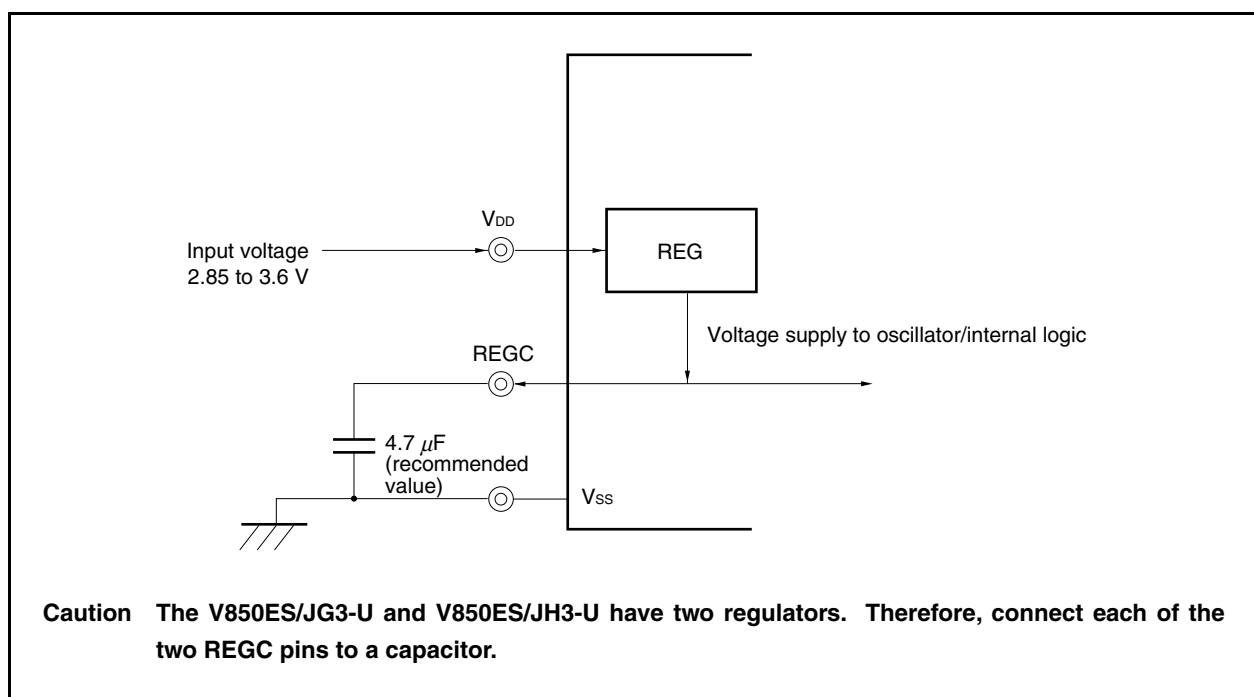
**CHAPTER  31   FLASH  MEMORY**

The V850ES/JG3-U and V850ES/JH3-U incorporate a flash memory.

- $\mu$PD70F3763, 70F3768:  384 KB flash memory
- $\mu$PD70F3764, 70F3769:  512 KB flash memory

Flash memory versions offer the following advantages for development environments and mass production applications.

○ For altering software after the V850ES/JG3-U and V850ES/JH3-U are soldered onto the target system.
○ For data adjustment when starting mass production.
○ For differentiating software according to the specification in small scale production of various models.
○ For facilitating inventory management.
○ For updating software after shipment.

## 31.1  Features

○ 4-byte/1-clock access (when instruction is fetched)
○ Capacity: 384/512 KB
○ Rewrite voltage: Erase/write with a single power supply
○ Rewriting method
  - Rewriting by communication with flash memory programmer via serial interface (on-board/off-board programming)
  - Rewriting flash memory by user program (self programming)
○ Flash memory rewrite prohibit function supported (security function)
○ Safe rewriting of entire flash memory area by self programming using boot swap function
○ Interrupts can be acknowledged during self programming.

## 31.2  Memory Configuration

The internal flash memory area of the V850ES/JG3-U and V850ES/JH3-U is divided into 96 or 128 blocks and can be programmed/erased in block units.  All the blocks can also be erased at once.

When the boot swap function is used, the physical memory located at the addresses of blocks 0 to 15 is replaced by the physical memory located at the addresses of blocks 16 to 31.  For details of the boot swap function, see **31.5  Rewriting by Self Programming**.

**Figure 31-1.  Flash Memory Mapping**



μPD70F3763, 70F3768
(384 KB)

μPD70F3764, 70F3769
(512 KB)

**Notes 1.**  Area to be replaced with the boot area by the boot swap function
   **2.**  Boot area
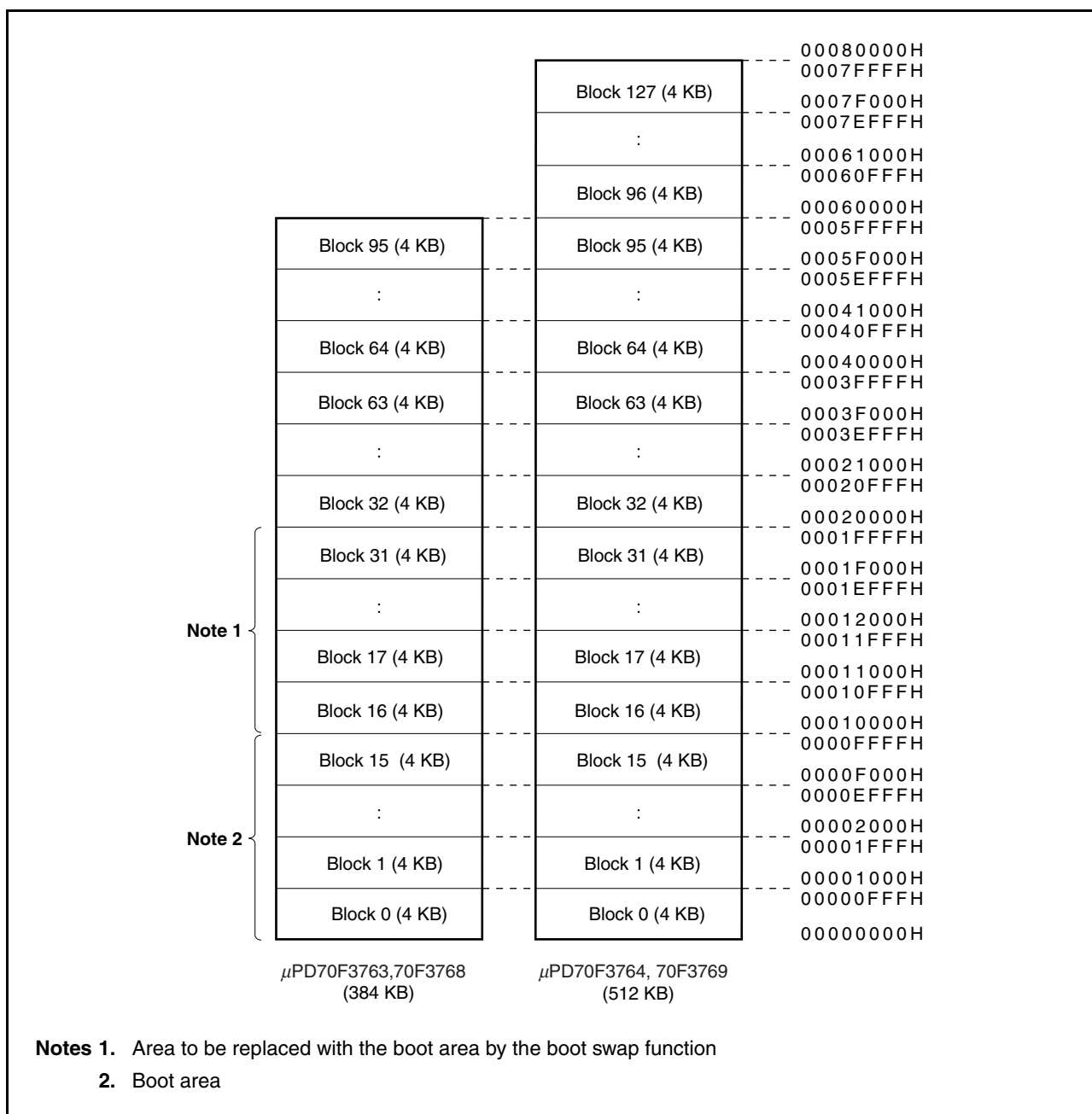
## 31.3  Functional Overview

The internal flash memory of the V850ES/JG3-U and V850ES/JH3-U can be rewritten by using the rewrite function of the flash memory programmer, regardless of whether the V850ES/JG3-U and V850ES/JH3-U have already been mounted on the target system or not (off-board/on-board programming).

In addition, a security function that prohibits rewriting the user program written to the internal flash memory is also supported, so that the program cannot be changed by an unauthorized person.

The rewrite function using the user program (self programming) is ideal for an application where it is assumed that the program is changed after production/shipment of the target system.  A boot swap function that rewrites the entire flash memory area safely is also supported.  In addition, interrupt servicing is supported during self programming, so that the flash memory can be rewritten under various conditions, such as while communicating with an external device.

**Table 31-1.  Rewrite Method**

| Rewrite Method | Functional Overview | Operation Mode |
|---|---|---|
| On-board programming | Flash memory can be rewritten after the device is mounted on the target system, by using a flash memory programmer. | Flash memory programming mode |
| Off-board programming | Flash memory can be rewritten before the device is mounted on the target system, by using a flash memory programmer and a dedicated program adapter board (FA series). | |
| Self programming | Flash memory can be rewritten by executing a user program that has been written to the flash memory in advance by means of off-board/on-board programming.  (During self-programming, instructions cannot be fetched from or data access cannot be made to the internal flash memory area.  Therefore, the rewrite program must be transferred to the internal RAM or external memory in advance). | Normal operation mode |

**Remark**   The FA series is a product of Naito Densei Machida Mfg. Co., Ltd.

**Table 31-2. Basic Functions**

| Function | Functional Outline | Support (√: Supported, ×: Not supported) | |
|---|---|---|---|
| | | On-Board/Off-Board Programming | Self Programming |
| Blank check | The erasure status of the entire memory is checked. | √ | √ |
| Chip erasure | The contents of the entire memory area are erased all at once. | √ | ×**Note** |
| Block erasure | The contents of specified memory blocks are erased. | √ | √ |
| Program | Writing to specified addresses, and a verify check to see if the write level is secured, are performed. | √ | √ |
| Verify/checksum | Data read from the flash memory is compared with data transferred from the flash memory programmer. | √ | × (Can be read by user program) |
| Read | Data written to the flash memory is read. | √ | × |
| Security setting | Use of the block erase command, chip erase command, program command, and read command, and boot area rewrite, are prohibited. | √ | × (Supported only when setting is changed from enable to disable) |

**Note** This is possible by selecting the entire memory area for the block erase function.

The following table lists the security functions. The block erase command prohibit, chip erase command prohibit, and program command prohibit functions are enabled by default after shipment, and security can be set by rewriting via on-board/off-board programming. Each security function can be used in combination with the others at the same time.

**Table 31-3. Security Functions**

| Function | Function Outline |
|---|---|
| Block erase command prohibit | Execution of a block erase command on all blocks is prohibited. Setting of prohibition can be initialized by execution of a chip erase command. |
| Chip erase command prohibit | Execution of block erase and chip erase commands on all the blocks is prohibited. Once prohibition is set, setting of prohibition cannot be initialized because the chip erase command cannot be executed. |
| Program command prohibit | Execution of program and block erase commands on all the blocks is prohibited. Setting of prohibition can be initialized by execution of the chip erase command. |
| Read command prohibit | Execution of a read command on all of the blocks is prohibited. Setting of the prohibition can be initialized by execution of a chip erase command. |
| Boot area rewrite prohibit | Execution of write, block erase, and chip erase commands on the boot area is prohibited. Setting of the prohibition of rewriting the boot area cannot be initialized after it is once set. |

**Table 31-4.  Security Setting**

| Function | Erase, Write, Read Operations When Each Security Is Set (√: Executable, ×: Not Executable, −: Not Supported) | | Notes on Security Setting | |
|---|---|---|---|---|
| | On-Board/ Off-Board Programming | Self Programming | On-Board/ Off-Board Programming | Self Programming |
| Block erase command prohibit | Block erase command: × <br> Chip erase command: √ <br> Program command: √ <br> Read command: √ | Block erasure (FlashBlockErase): √ <br> Chip erasure: − <br> Write (FlashWordWrite): √ <br> Read (FlashWordRead): √ | Setting of prohibition can be initialized by chip erase command. | Supported only when setting is changed from enable to prohibit |
| Chip erase command prohibit | Block erase command: × <br> Chip erase command: × <br> Program command: √[Note 1] <br> Read command: √ | Block erasure (FlashBlockErase): √ <br> Chip erasure: − <br> Write (FlashWordWrite): √ <br> Read (FlashWordRead): √ | Setting of prohibition cannot be initialized. | |
| Program command prohibit | Block erase command: × <br> Chip erase command: √ <br> Program command: × <br> Read command: √ | Block erasure (FlashBlockErase): √ <br> Chip erasure: − <br> Write (FlashWordWrite): √ <br> Read (FlashWordRead): √ | Setting of prohibition can be initialized by chip erase command. | |
| Read command prohibit | Block erase command: √ <br> Chip erase command: √ <br> Program command: √ <br> Read command: × | Block erasure (FlashBlockErase): √ <br> Chip erasure: − <br> Write (FlashWordWrite): √ <br> Read (FlashWordRead): √ | Setting of prohibition can be initialized by chip erase command. | |
| Boot area rewrite prohibit | Block erase command: ×[Note 2] <br> Chip erase command: × <br> Program command: ×[Note 2] <br> Read command: √ | Block erasure (FlashBlockErase): ×[Note 2] <br> Chip erasure: − <br> Write (FlashWordWrite): ×[Note 2] <br> Read (FlashWordRead): √ | Setting of prohibition cannot be initialized. | Supported only when setting is changed from enable to prohibit[Note 3] |

**Notes 1.** In this case, since the erase command is invalid, data different from the data already written in the flash memory cannot be written.

   **2.** Executable except in boot area.

   **3.** The boot area rewrite prohibit function becomes effective after the reset input.

## 31.4  Rewriting by Flash Memory Programmer

The flash memory can be rewritten by using a flash memory programmer after the V850ES/JG3-U and V850ES/JH3-U are mounted on the target system (on-board programming).  The flash memory can also be rewritten before the device is mounted on the target system (off-board programming) by using a dedicated program adapter (FA series).

### 31.4.1  Programming environment

The following shows the environment required for writing programs to the flash memory of the V850ES/JG3-U and V850ES/JH3-U.

**Figure 31-2.  Environment Required for Writing Programs to Flash Memory**



**Note**  Connect the FLMD1 pin to the flash memory programmer or connect to GND via a pull-down resistor on the board.
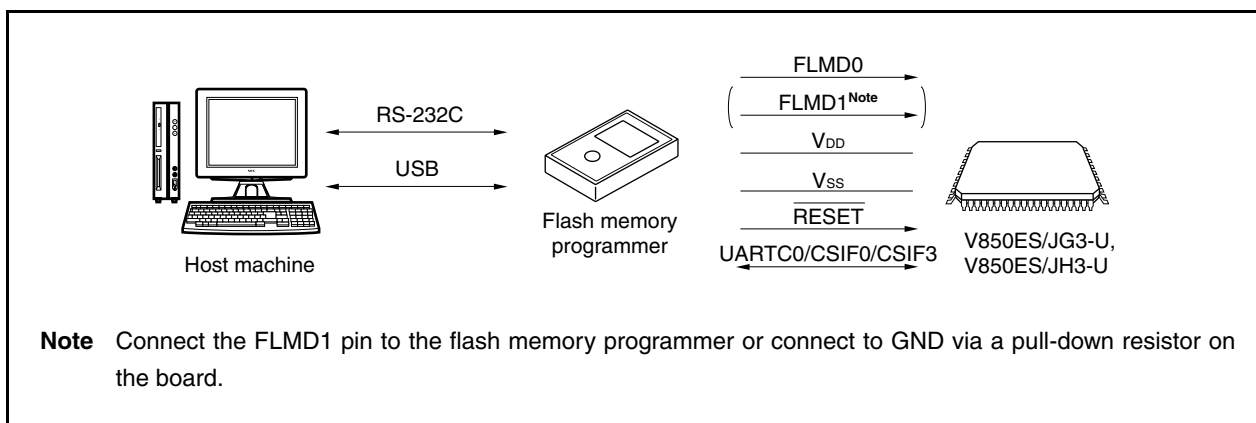
A host machine is required for controlling the flash memory programmer.

UARTC0, CSIF0, or CSIF3 is used for the interface between the flash memory programmer and the V850ES/JG3-U and V850ES/JH3-U to perform writing, erasing, etc.  A dedicated program adapter (FA series) required for off-board writing.

**Remark**  The FA series is a product of Naito Densei Machida Mfg. Co., Ltd.

### 31.4.2 Communication mode

Communication between the flash memory programmer and the V850ES/JG3-U or V850ES/JH3-U is performed by serial communication using the UARTC0, CSIF0, or CSIF3 interface of the V850ES/JG3-U or V850ES/JH3-U.

**(1) UARTC0**

Transfer rate: 9,600 to 153,600 bps

**Figure 31-3. Communication with Flash Memory Programmer (UARTC0)**



**Note** Connect the FLMD1 pin to the flash memory programmer or connect to GND via a pull-down resistor on the board.
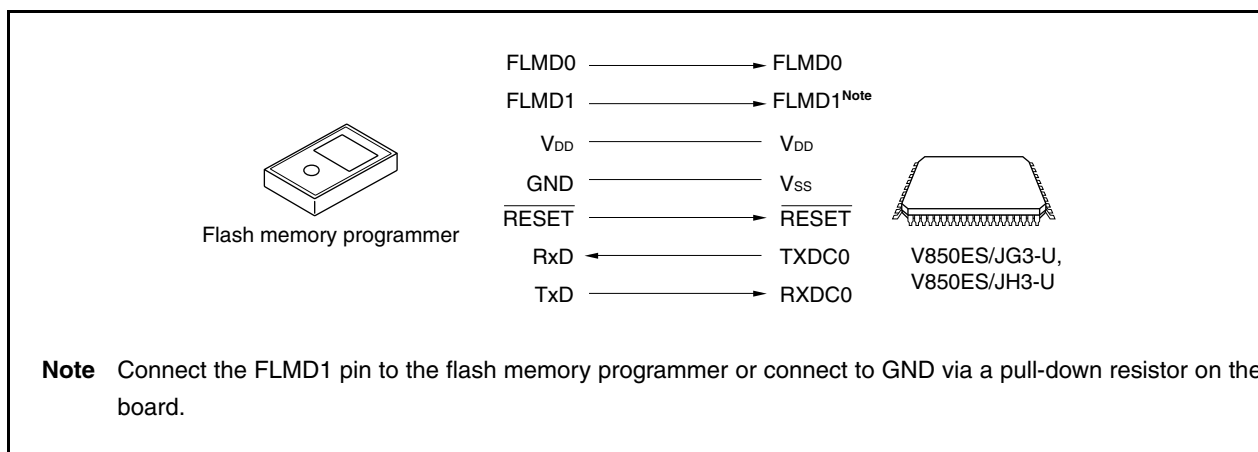
**(2) CSIF0, CSIF3**

Serial clock: 5 MHz or less (MSB first)

**Figure 31-4. Communication with Flash Memory Programmer (CSIF0, CSIF3)**



**Note** Connect the FLMD1 pin to the flash memory programmer or connect to GND via a pull-down resistor on the board.
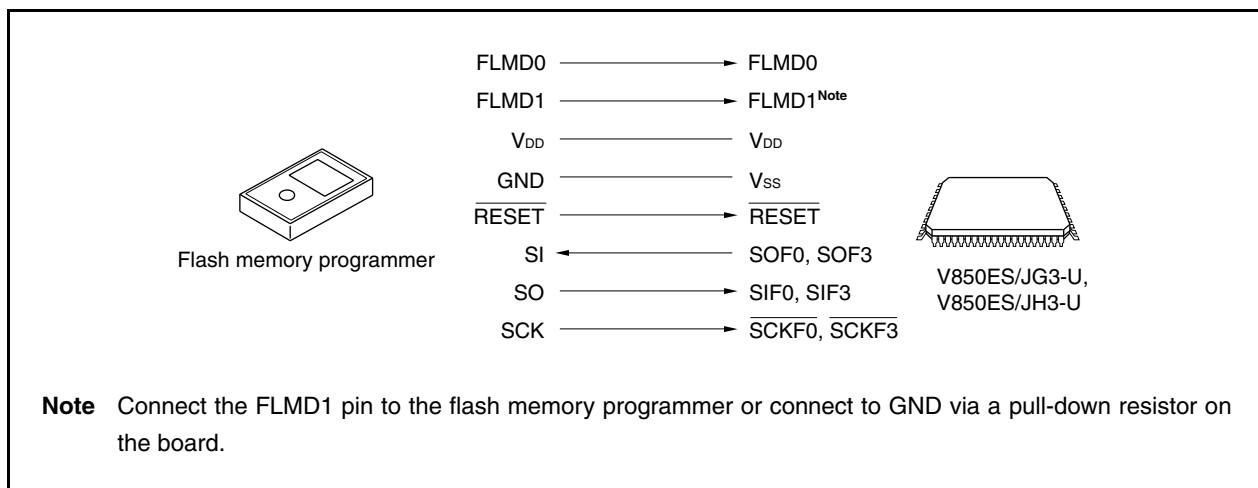
**(3)  CSIF0 + HS, CSIF3 + HS**

Serial clock:  5 MHz or less (MSB first)

**Figure 31-5.  Communication with Flash Memory Programmer (CSIF0 + HS, CSIF3 + HS)**



| | | |
|---|---|---|
| | FLMD0 ────────────► FLMD0 | |
| | FLMD1 ────────────► FLMD1[Note] | |
| | V$_{DD}$ ──────────── V$_{DD}$ | |
| Flash memory programmer | GND ──────────── V$_{SS}$ | V850ES/JG3-U, V850ES/JH3-U |
| | $\overline{RESET}$ ────────────► $\overline{RESET}$ | |
| | SI ◄──────────── SOF0, SOF3 | |
| | SO ────────────► SIF0, SIF3 | |
| | SCK ────────────► $\overline{SCKF0}$, $\overline{SCKF3}$ | |
| | HS ◄──────────── P913 | |

**Note**  Connect the FLMD1 pin to the flash memory programmer or connect to GND via a pull-down resistor on the board.
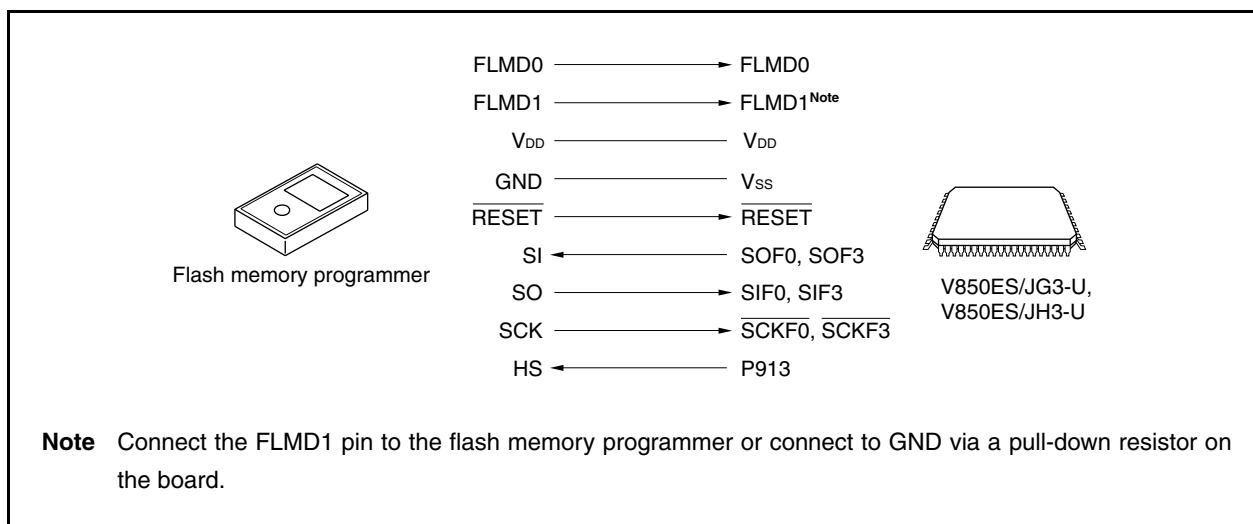
The flash memory programmer outputs the transfer clock, and the V850ES/JG3-U and V850ES/JH3-U operate as a slave.

When the PG-FP5 is used as the flash memory programmer, it generates the following signals to the V850ES/JG3-U and V850ES/JH3-U.  For details, refer to the **PG-FP5 User's Manual (U18865E)**.

**Table 31-5.  Signal Connections of Flash Memory Programmer (PG-FP5)**

| PG-FP5 | | | V850ES/JG3-U, V850ES/JH3-U | Processing for Connection | | |
|---|---|---|---|---|---|---|
| Signal Name | I/O | Pin Function | Pin Name | UARTC0 | CSIF0, CSIF3 | CSIF0 + HS, CSIF3 + HS |
| FLMD0 | Output | Write enable/disable | FLMD0 | ◎ | ◎ | ◎ |
| FLMD1 | Output | Write enable/disable | FLMD1 | ◎Note 1 | ◎Note 1 | ◎Note 1 |
| VDD | – | $V_{DD}$ voltage generation/voltage monitor | $V_{DD}$ | ◎ | ◎ | ◎ |
| GND | – | Ground | $V_{SS}$ | ◎ | ◎ | ◎ |
| CLK | Output | Clock output to V850ES/JG3-U and V850ES/JH3-U | X1, X2 | ×Note 2 | ×Note 2 | ×Note 2 |
| $\overline{RESET}$ | Output | Reset signal | $\overline{RESET}$ | ◎ | ◎ | ◎ |
| SI/RxD | Input | Receive signal | SOF0, SOF3/ TXDC0 | ◎ | ◎ | ◎ |
| SO/TxD | Output | Transmit signal | SIF0, SIF3/ RXDC0 | ◎ | ◎ | ◎ |
| SCK | Output | Transfer clock | $\overline{SCKF0}$, $\overline{SCKF3}$ | × | ◎ | ◎ |
| HS | Input | Handshake signal for CSIF0 + HS, CSIF3 + HS communication | P913 | × | × | ◎ |

**Notes 1.** Wire these pins as shown in Figures 31-6 and 31-7, or connect then to GND via pull-down resistor on board.

　　　**2.** Clock cannot be supplied via the CLK pin of the flash memory programmer.  Create an oscillator on board and supply the clock.

**Remark** ◎: Must be connected.

　　　　　×: Does not have to be connected.

**Table 31-6.  Wiring of V850ES/JG3-U Flash Writing Adapters (1/3)**

| Pin No. | Pin Name | Recommended Connection |
|---|---|---|
| 1 | AV$_{REF0}$ | Connect to VDD pin of the programmer |
| 2 | AV$_{SS}$ | Connect to GND pin of the programmer |
| 3 | P10/ANO0 | - |
| 4 | P11/ANO1 | - |
| 5 | AV$_{REF1}$ | Connect to VDD pin of the programmer |
| 6 | P02/NMI | - |
| 7 | P03/INTP02/ADTRG/UCLK | - |
| 8 | FLMD0 | Connect to FLMD0 (output) pin of the programmer |
| 9 | V$_{DD}$ | Connect to VDD pin of the programmer |
| 10 | REGC | Connect the REGC pin to GND via 4.7 $\mu$F capacitor |
| 11 | V$_{SS}$ | Connect to GND pin of the programmer |
| 12 | X1 | Connect to 3 to 6 MHz Resonator |
| 13 | X2 | Connect to 3 to 6 MHz Resonator |
| 14 | $\overline{\text{RESET}}$ | Connect to RESET (output) pin of the programmer |
| 15 | XT1 | Connect to GND pin of the programmer |
| 16 | XT2 | - |
| 17 | UDMF | - |
| 18 | UDPF | - |
| 19 | UV$_{DD}$ | Connect to VDD pin of the programmer |
| 20 | P04/INTP03 | - |
| 21 | P05/INTP04 | - |
| 22 | P40/SIF0/TXDC4/SDA01 | When CSI (CSIF0) is used : connect to SO (output) pin of the programmer<br>When CSI (CSIF0)is not used : pull-down[Note] |
| 23 | P41/SOF0/RXDC4/SCL01 | When CSI (CSIF0)is used : connect to SI (input) pin of the programmer<br>When CSI (CSIF0)is not used : pull-down[Note] |
| 24 | P42/$\overline{\text{SCKF0}}$/INTP10 | When CSI(CSIF0)is used : connect to SCK (output) pin of the programmer<br>When CSI (CSIF0)is not used : pull-down[Note] |
| 25 | P30/TXDC0/SOF4/INTP07 | When UART (UARTD0) is used : connect to RxD (input) pin of the programmer<br>When UART (UARTD0) is not used : pull-down[Note] |
| 26 | P31/RXDC0/SIF4/INTP08 | When UART (UARTD0) is used : connect to TxD (output) pin of the programmer<br>When UART (UARTD0) is not used : pull-down[Note] |
| 27 | P32/ASCKC0/$\overline{\text{SCKF4}}$/TIAA00/TOAA00 | - |
| 28 | P33/TIAA01/TOAA01/RTCDIV/RTCCL | - |
| 29 | P34/TIAA10/TOAA10/TOAA1OFF/INTP09 | - |
| 30 | P35/TIAA11/TOAA11/RTC1HZ | - |

**Note**  Independently connect to EV$_{SS}$ or V$_{DD}$ via a resistor.

**Table 31-6.  Wiring of V850ES/JG3-U Flash Writing Adapters (2/3)**

| Pin No. | Pin Name | Recommended Connection |
|---|---|---|
| 31 | P36/TXDC3/SCL00/$\overline{\text{UDMARQ0}}$ | - |
| 32 | P37/RXDC3/SDA00/$\overline{\text{UDMAAK0}}$ | - |
| 33 | EV$_{SS}$ | Connect to GND pin of the programmer |
| 34 | EV$_{DD}$ | Connect to VDD pin of the programmer |
| 35 | P50/TIAB01/KR0/TOAB01/RTP00/$\overline{\text{UDMARQ1}}$ | - |
| 36 | P51/TIAB02/KR1/TOAB02/RTP01/$\overline{\text{UDMAAK1}}$ | - |
| 37 | P52/TIAB03/KR2/TOAB03/RTP02/DDI | - |
| 38 | P53/SIF2/TIAB00/KR3/TOAB00/RTP03/DDO | - |
| 39 | P54/SOF2/KR4/RTP04/DCK | - |
| 40 | P55/$\overline{\text{SCKF2}}$/KR5/RTP05/DMS | - |
| 41 | P56/INTP05/$\overline{\text{DRST}}$ | - |
| 42 | P90/KR6/TXDC1/SDA02 | - |
| 43 | P91/KR7/RXDC1/SCL02 | - |
| 44 | P92/TENC01/TIT01/TOT01 | - |
| 45 | P93/TECR0/TIT00/TOT00 | - |
| 46 | P94/TIAA31/TOAA31/TENC00/EVTT00 | - |
| 47 | P95/TIAA30/TOAA30 | - |
| 48 | P96/TIAA21/TOAA21/INTP11 | - |
| 49 | P97/SIF1/TIAA20/TOAA20 | - |
| 50 | P98/SOF1/INTP12 | - |
| 51 | P99/SCKF1/INTP13/A9 | - |
| 52 | P910/SIF3/TXDC2/INTP14/A10 | When CSI (CSIF3) is used : connect to SO (output) pin of the programmer<br>When CSI (CSIF3) is not used : pull-down[Note] |
| 53 | P911/SOF3/RXDC2/INTP15/A11 | When CSI (CSIF3) is used : connect to SI (input) pin of the programmer<br>When CSI (CSIF3) is not used : pull-down[Note] |
| 54 | P912/$\overline{\text{SCKF3}}$/A12 | When CSI (CSIF3) is used : connect to SCK (output) pin of the programmer<br>When CSI (CSIF3) is not used : pull-down[Note] |
| 55 | P913/TOAB1OFF/INTP16/A13   HS | When CSI-HS (CSIF3) is used : connect to H/S (input) pin of the programmer<br>When CSI-HS (CSIF3) is not used : pull-down[Note] |
| 56 | P914/TIAA51/TOAA51/INTP17/A14 | - |
| 57 | P915/TIAA50/TOAA50/INTP18/A15 | - |
| 58 | PCT0/$\overline{\text{WR0}}$ | - |
| 59 | PCT1/$\overline{\text{WR1}}$ | - |
| 60 | V$_{DD}$ | Connect to VDD pin of the programmer |

**Note**  Independently connect to EV$_{SS}$ or V$_{DD}$ via a resistor.

**Table 31-6.  Wiring of V850ES/JG3-U Flash Writing Adapters (3/3)**

| Pin No. | Pin Name | Recommended Connection |
|---|---|---|
| 61 | REGC | Connect the REGC pin to GND via 4.7 $\mu$F capacitor |
| 62 | EV$_{SS}$ | Connect to GND pin of the programmer |
| 63 | EV$_{DD}$ | Connect to VDD pin of the programmer |
| 64 | PCM1/CLKOUT | - |
| 65 | P60/TOAB1T1/TOAB11/TIAB11/$\overline{\text{WAIT}}$ | - |
| 66 | P61/TOAB1B1/TIAB10/TOAB10/$\overline{\text{RD}}$ | - |
| 67 | P62/TOAB1T2/TOAB12/TIAB12/ASTB | - |
| 68 | P63/TOAB1B2/TRGAB1/$\overline{\text{CS0}}$ | - |
| 69 | P64/TOAB1T3/TOAB13/TIAB13/$\overline{\text{CS2}}$ | - |
| 70 | P65/TOAB1B3/EVTAB1/$\overline{\text{CS3}}$ | Connect to VDD pin of the programmer |
| 71 | PDL0/AD0 | Connect to VDD pin of the programmer |
| 72 | PDL1/AD1 | Connect to VDD pin of the programmer |
| 73 | PDL2/AD2 | Connect to VDD pin of the programmer |
| 74 | PDL3/AD3 | Connect to VDD pin of the programmer |
| 75 | PDL4/AD4 | Connect to VDD pin of the programmer |
| 76 | PDL5/AD5/FLMD1 | Connect to FLMD1 (output) pin of the programmer |
| 77 | PDL6/AD6 | - |
| 78 | PDL7/AD7 | - |
| 79 | EVSS | Connect to GND pin of the programmer |
| 80 | EVDD | Connect to VDD pin of the programmer |
| 81 | PDL8/AD8 | - |
| 82 | PDL9/AD9 | - |
| 83 | PDL10/AD10 | - |
| 84 | PDL11/AD11 | - |
| 85 | PDL12/AD12 | - |
| 86 | PDL13/AD13 | - |
| 87 | PDL14/AD14 | - |
| 88 | PDL15/AD15 | - |
| 89 | P711/ANI11 | - |
| 90 | P710/ANI10 | - |
| 91 | P79/ANI9 | - |
| 92 | P78/ANI8 | - |
| 93 | P77/ANI7 | - |
| 94 | P76/ANI6 | - |
| 95 | P75/ANI5 | - |
| 96 | P74/ANI4 | - |
| 97 | P73/ANI3 | - |
| 98 | P72/ANI2 | - |
| 99 | P71/ANI1 | - |
| 100 | P70/ANI0 | - |

**Figure 31-6. Wiring Example of V850ES/JG3-U Flash Writing Adapter (FA-100GC-UEU-B)**
**(In CSIF0 + HS Mode) (1/2)**

**Figure 31-6.  Wiring Example of V850ES/JG3-U Flash Writing Adapter (FA-100GC-UEU-B)**
**(In CSIF0 + HS Mode) (2/2)**

**Notes 1.** Corresponding pins when CSIF3 is used.

**2.** Wire the FLMD1 pin as shown below, or connect it to GND on board via a pull-down resistor.

**3.** Create an oscillator on the flash writing adapter (shown in broken lines) and supply a clock.
Here is an example of the oscillator.

**Example:**
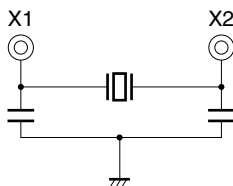


**4.** Corresponding pins when UARTC0 is used.

**Caution**   **Do not input a high level to the $\overline{\text{DRST}}$ pin.**

**Remarks 1.** Process the pins not shown in accordance with the handling of unused pins (see **2.3   Pin I/O Circuit Types, I/O Buffer Power Supplies, and Connection of Unused Pins**).

**2.** This adapter is for the 100-pin plastic LQFP package.

**Table 31-7.  Wiring of V850ES/JH3-U Flash Writing Adapters (1/4)**

| Pin No. | Pin Name | Recommended Connection |
|---|---|---|
| 1 | AV$_{REF0}$ | Connect to VDD pin of the programmer |
| 2 | AV$_{SS}$ | Connect to GND pin of the programmer |
| 3 | P10/ANO0 | - |
| 4 | P11/ANO1 | - |
| 5 | AV$_{REF1}$ | Connect to VDD pin of the programmer |
| 6 | P02/NMI | - |
| 7 | P03/INTP02/ADTRG/UCLK | - |
| 8 | P00/INTP00 | - |
| 9 | P01/INTP01 | - |
| 10 | PCM2/$\overline{\text{HLDAK}}$ | - |
| 11 | PCM3/$\overline{\text{HLDRQ}}$ | - |
| 12 | FLMD0 | Connect to FLMD0 (output) pin of the programmer |
| 13 | V$_{DD}$ | Connect to VDD pin of the programmer |
| 14 | REGC | Connect the REGC pin to GND via 4.7 $\mu$F capacitor |
| 15 | V$_{SS}$ | Connect to GND pin of the programmer |
| 16 | X1 | Connect to 3 to 6 MHz Resonator |
| 17 | X2 | Connect to 3 to 6 MHz Resonator |
| 18 | $\overline{\text{RESET}}$ | Connect to RESET (output) pin of the programmer |
| 19 | XT1 | Connect to GND pin of the programmer |
| 20 | XT2 | - |
| 21 | UDMF | - |
| 22 | UDPF | - |
| 23 | UV$_{DD}$ | Connect to VDD pin of the programmer |
| 24 | P04/INTP03 | - |
| 25 | P05/INTP04 | - |
| 26 | P04/INTP03 | - |
| 27 | P05/INTP04 | - |
| 28 | P25/INTP06 | - |
| 29 | P40/SIF0/TXDC4/SDA01 | When CSI (CSIF0) is used : connect to SO (output) pin of the programmer<br>When CSI (CSIF0)is not used : pull-down[Note] |
| 30 | P41/SOF0/RXDC4/SCL01 | When CSI (CSIF0)is used : connect to SI (input) pin of the programmer<br>When CSI (CSIF0)is not used : pull-down[Note] |
| 31 | P42/$\overline{\text{SCKF0}}$/INTP10 | When CSI(CSIF0)is used : connect to SCK (output) pin of the programmer<br>When CSI (CSIF0)is not used : pull-down[Note] |
| 32 | P20/TIAB03/KR2/TOAB03/RTP02 | - |
| 33 | P21/SIF2/TIAB00/KR3/TOAB00/RTP03 | - |

**Note**  Independently connect to EV$_{SS}$ or V$_{DD}$ via a resistor.

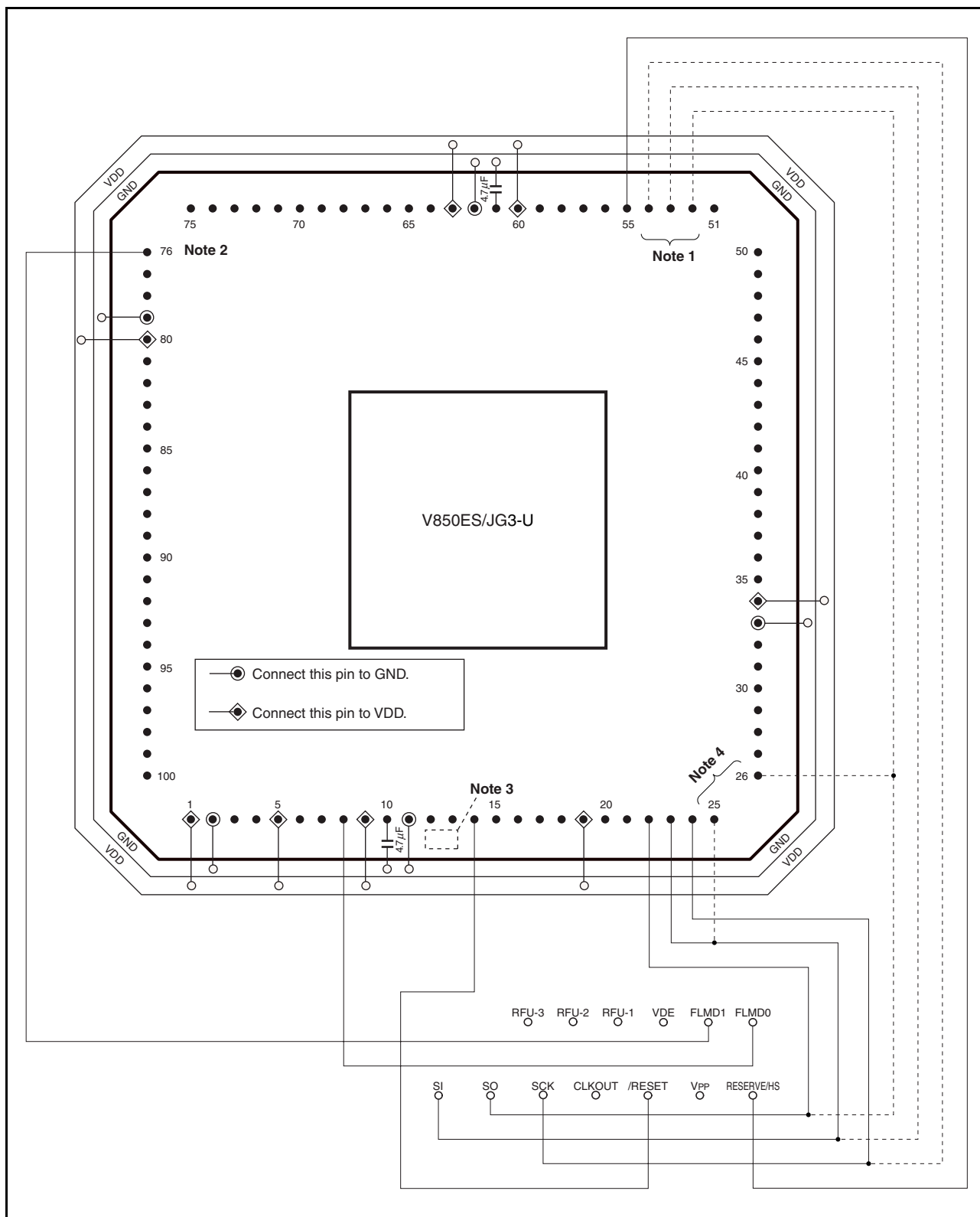**Table 31-7. Wiring of V850ES/JH3-U Flash Writing Adapters (2/4)**

| Pin No. | Pin Name | Recommended Connection |
|---------|----------|------------------------|
| 34 | P22/SOF2/KR4/RTP04 | - |
| 35 | P23/SCKF2/KR5/RTP05 | - |
| 36 | P24/INTP05 | - |
| 37 | P30/TXDC0/SOF4/INTP07 | When UART (UARTD0) is used : connect to RxD (input) pin of the programmer<br>When UART (UARTD0) is not used : pull-down[Note1] |
| 38 | P31/RXDC0/SIF4/INTP08 | When UART (UARTD0) is used : connect to TxD (output) pin of the programmer<br>When UART (UARTD0) is not used : pull-down[Note1] |
| 39 | P32/ASCKC0/$\overline{\text{SCKF4}}$/TIAA00/TOAA00 | - |
| 40 | P33/TIAA01/TOAA01/RTCDIV/RTCCL | - |
| 41 | P34/TIAA10/TOAA10/TOAA1OFF/INTP09 | - |
| 42 | P35/TIAA11/TOAA11/RTC1HZ | - |
| 43 | P36/TXDC3/SCL00/CTXD0[Note2]/$\overline{\text{UDMARQ0}}$ | - |
| 44 | P37/RXDC3/SDA00/CRXD0[Note2]/$\overline{\text{UDMAAK0}}$ | - |
| 45 | EV$_{SS}$ | Connect to GND pin of the programmer |
| 46 | EV$_{DD}$ | Connect to VDD pin of the programmer |
| 47 | P50/TIAB01/KR0/TOAB01/RTP00/$\overline{\text{UDMARQ1}}$ | - |
| 48 | P51/TIAB02/KR1/TOAB02/RTP01/$\overline{\text{UDMAAK1}}$ | - |
| 49 | DDI | - |
| 50 | DDO | - |
| 51 | DCK | - |
| 52 | DMS | - |
| 53 | $\overline{\text{DRST}}$ | Connect to GND pin of the programmer |
| 54 | P90/KR6/TXDC1/SDA02/A0 | - |
| 55 | P91/KR7/RXDC1/SCL02/A1 | - |
| 56 | P92/TENC01/TIT01/TOT01/A2 | - |
| 57 | P93/TECR0/TIT00/TOT00/A3 | - |
| 58 | P94/TIAA31/TOAA31/TENC00/EVTT00/A4 | - |
| 59 | P95/TIAA30/TOAA30/A5 | - |
| 60 | EV$_{SS}$ | Connect to GND pin of the programmer |
| 61 | EV$_{DD}$ | Connect to VDD pin of the programmer |
| 62 | P96/TIAA21/TOAA21/INTP11/A6 | - |
| 63 | P97/SIF1/TIAA20/TOAA20/A7 | - |
| 64 | P98/SOF1/INTP12/A8 | - |
| 65 | P99/SCKF1/INTP13/A9 | - |

**Note 1.** Independently connect to EV$_{SS}$ or V$_{DD}$ via a resistor.

    **2.** $\mu$PD70F3370, 70F3371 only

**Table 31-7. Wiring of V850ES/JH3-U Flash Writing Adapters (3/4)**

| Pin No. | Pin Name | Recommended Connection |
|---------|----------|------------------------|
| 66 | P910/SIF3/TXDC2/INTP14/A10 | When CSI (CSIF3) is used : connect to SO (output) pin of the programmer<br>When CSI (CSIF3) is not used : pull-down<sup>Note</sup> |
| 67 | P911/SOF3/RXDC2/INTP15/A11 | When CSI (CSIF3) is used : connect to SI (input) pin of the programmer<br>When CSI (CSIF3) is not used : pull-down<sup>Note</sup> |
| 68 | P912/$\overline{\text{SCKF3}}$/A12 | When CSI (CSIF3) is used : connect to SCK (output) pin of the programmer<br>When CSI (CSIF3) is not used : pull-down<sup>Note</sup> |
| 69 | P913/TOAB1OFF/INTP16/A13   HS | When CSI-HS (CSIF3) is used : connect to H/S (input) pin of the programmer<br>When CSI-HS (CSIF3) is not used : pull-down<sup>Note</sup> |
| 70 | P914/TIAA51/TOAA51/INTP17/A14 | - |
| 71 | P915/TIAA50/TOAA50/INTP18/A15 | - |
| 72 | PDH0/A16 | - |
| 73 | PDH1/A17 | - |
| 74 | PDH2/A18 | - |
| 75 | PDH3/A19 | - |
| 76 | PCT0/$\overline{\text{WR0}}$ | - |
| 77 | PCT1/$\overline{\text{WR1}}$ | - |
| 78 | PDH4/A20 | - |
| 79 | PDH5/A21 | - |
| 80 | PDH6/A22 | - |
| 81 | PDH7/A23 | - |
| 82 | V$_{DD}$ | Connect to VDD pin of the programmer |
| 83 | REGC | Connect the REGC pin to GND via 4.7 $\mu$F (recommended value) capacitor |
| 84 | EV$_{SS}$ | Connect to GND pin of the programmer |
| 85 | EV$_{DD}$ | Connect to VDD pin of the programmer |
| 86 | PCM1/CLKOUT | - |
| 87 | PCT4/$\overline{\text{RD}}$ | - |
| 88 | PCT6/ASTB | - |
| 89 | PCM0/$\overline{\text{WAIT}}$ | - |
| 90 | P60/TOAB1T1/TOAB11/TIAB11 | - |
| 91 | P61/TOAB1B1/TIAB10/TOAB10 | - |
| 92 | P62/TOAB1T2/TOAB12/TIAB12 | - |
| 93 | P63/TOAB1B2/TRGAB1 | - |
| 94 | P64/TOAB1T3/TOAB13/TIAB13 | - |
| 95 | P65/TOAB1B3/EVTAB1 | Connect to GND pin of the programmer |

**Note** Independently connect to EV$_{SS}$ or V$_{DD}$ via a resistor.

**Table 31-7.  Wiring of V850ES/JH3-U Flash Writing Adapters (4/4)**

| Pin No. | Pin Name | Recommended Connection |
|---|---|---|
| 96 | PCS0/$\overline{\text{CS0}}$ | - |
| 97 | PCS2/$\overline{\text{CS2}}$ | - |
| 98 | PDL0/AD0 | Connect to GND pin of the programmer |
| 99 | PDL1/AD1 | Connect to GND pin of the programmer |
| 100 | PDL2/AD2 | Connect to GND pin of the programmer |
| 101 | PDL3/AD3 | Connect to GND pin of the programmer |
| 102 | PDL4/AD4 | Connect to GND pin of the programmer |
| 103 | PDL5/AD5/FLMD1 | Connect to FLMD1 (output) pin of the programmer |
| 104 | PDL6/AD6 | - |
| 105 | PDL7/AD7 | - |
| 106 | EVSS | Connect to GND pin of the programmer |
| 107 | EVDD | Connect to VDD pin of the programmer |
| 108 | PDL8/AD8 | - |
| 109 | PDL9/AD9 | - |
| 110 | PDL10/AD10 | - |
| 111 | PDL11/AD11 | - |
| 112 | PDL12/AD12 | - |
| 113 | PDL13/AD13 | - |
| 114 | PDL14/AD14 | - |
| 115 | PDL15/AD15 | - |
| 116 | PCS3/$\overline{\text{CS3}}$ | - |
| 117 | P711/ANI11 | - |
| 118 | P710/ANI10 | - |
| 119 | P79/ANI9 | - |
| 120 | P78/ANI8 | - |
| 121 | P77/ANI7 | - |
| 122 | P76/ANI6 | - |
| 123 | P75/ANI5 | - |
| 124 | P74/ANI4 | - |
| 125 | P73/ANI3 | - |
| 126 | P72/ANI2 | - |
| 127 | P71/ANI1 | - |
| 128 | P70/ANI0 | - |

**Figure 31-7.  Wiring Example of V850ES/JH3-U Flash Writing Adapter (FA-128GF-GAT-B)**
**(In CSIF0 + HS Mode) (1/2)**

**Figure 31-7.  Wiring Example of V850ES/JH3-U Flash Writing Adapter (FA-128GF-GAT-B)**
**(In CSIF0 + HS Mode) (2/2)**

**Notes 1**. Corresponding pins when CSIF3 is used.

      **2.** Wire the FLMD1 pin as shown below, or connect it to GND on board via a pull-down resistor.

      **3.** Create an oscillator on the flash writing adapter (shown in broken lines) and supply a clock.
Here is an example of the oscillator.

           **Example:**
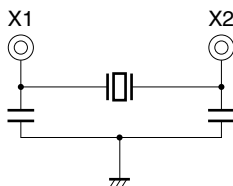


      **4.** Corresponding pins when UARTC0 is used.

**Caution**   **Do not input a high level to the $\overline{\text{DRST}}$ pin.**

**Remarks 1.** Process the pins not shown in accordance with the handling of unused pins (see **2.3  Pin I/O Circuit Types, I/O Buffer Power Supplies, and Connection of Unused Pins**).

      **2.** This adapter is for the 128-pin plastic LQFP package.

### 31.4.3  Flash memory control

The following shows the procedure for manipulating the flash memory.

**Figure 31-8.  Procedure for Manipulating Flash Memory**
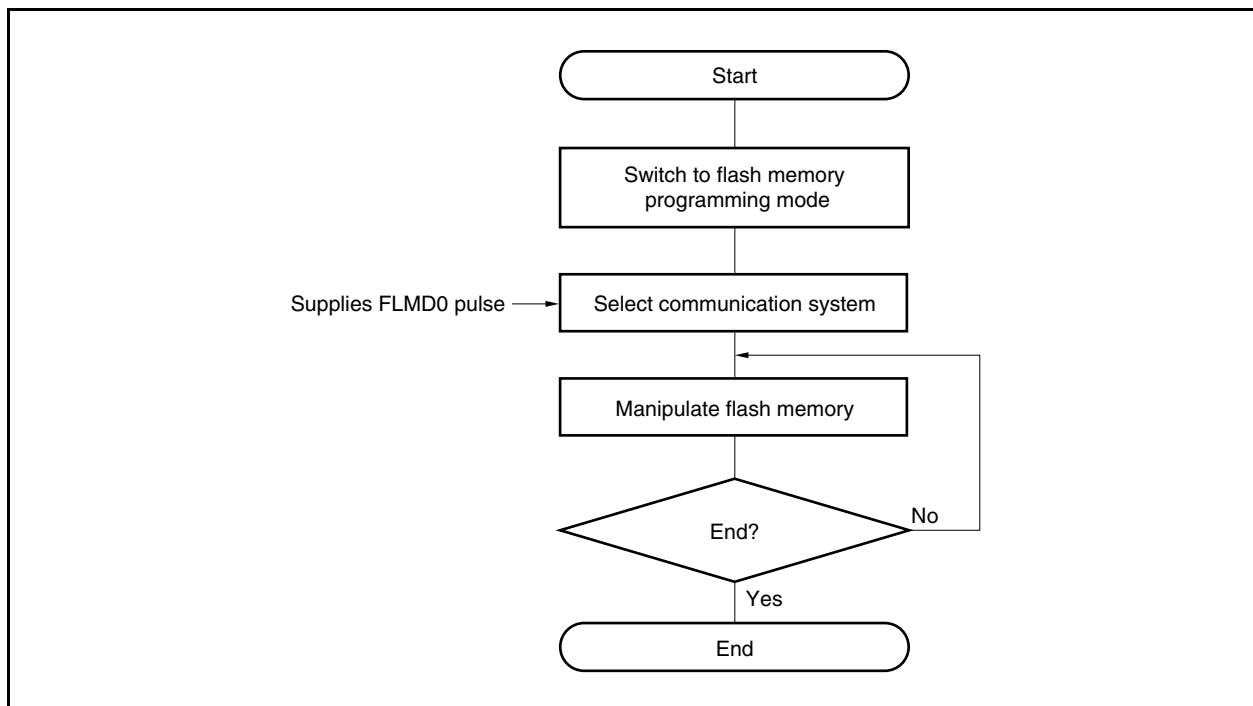
### 31.4.4  Selection of communication mode

In the V850ES/JG3-U and V850ES/JH3-U, the communication mode is selected by inputting pulses (12 pulses max.) to the FLMD0 pin after switching to the flash memory programming mode.  The FLMD0 pulse is generated by the dedicated flash memory programmer.

The following shows the relationship between the number of pulses and the communication mode.

**Figure 31-9.  Selection of Communication Mode**



**Note**   The number of clocks is as follows depending on the communication mode.

| FLMD0 Pulse | Communication Mode | Remarks |
|---|---|---|
| 0 | UARTC0 | Communication rate:  9,600 bps (after reset), LSB first |
| 8 | CSIF0 | V850ES/Jx3-U performs slave operation, MSB first |
| 9 | CSIF3 | V850ES/Jx3-U performs slave operation, MSB first |
| 11 | CSIF0 + HS | V850ES/Jx3-U performs slave operation, MSB first |
| 12 | CSIF3 + HS | V850ES/Jx3-U performs slave operation, MSB first |
| Other | RFU | Setting prohibited |

**Caution**    **When UARTC0 is selected, the receive clock is calculated based on the reset command sent from the flash memory programmer after receiving the FLMD0 pulse.**

### 31.4.5 Communication commands

The V850ES/JG3-U and V850ES/JH3-U communicate with the flash memory programmer by means of commands. The signals sent from the flash memory programmer to the V850ES/JG3-U and V850ES/JH3-U are called "commands". The response signals sent from the V850ES/JG3-U and V850ES/JH3-U to the flash memory programmer are called "response commands".

**Figure 31-10. Communication Commands**
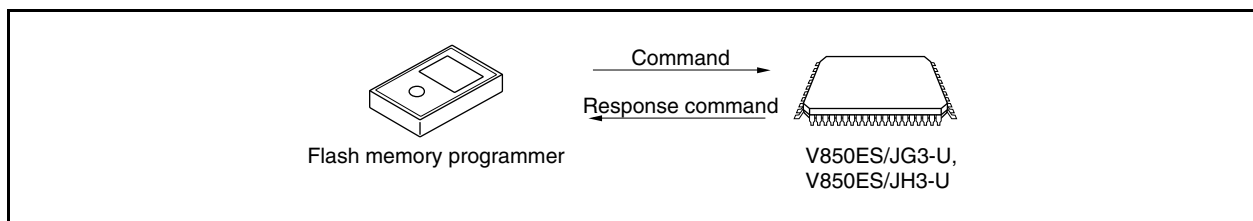


The following shows the commands for flash memory control in the V850ES/JG3-U and V850ES/JH3-U. All of these commands are issued from the dedicated flash programmer, and the V850ES/JG3-U and V850ES/JH3-U perform the processing corresponding to the commands.

**Table 31-8. Flash Memory Control Commands**

| Classification | Command Name | Support | | | Function |
|---|---|---|---|---|---|
| | | CSIF0, CSIF3 | CSIF0 + HS, CSIF3 + HS | UARTC0 | |
| Blank check | Block blank check command | √ | √ | √ | Checks if the contents of the memory in the specified block have been correctly erased. |
| Erase | Chip erase command | √ | √ | √ | Erases the contents of the entire memory. |
| | Block erase command | √ | √ | √ | Erases the contents of the memory of the specified block. |
| Write | Program command | √ | √ | √ | Writes the specified address range, and executes a contents verify check. |
| Verify | Verify command | √ | √ | √ | Compares the contents of memory in the specified address range with data transferred from the flash memory programmer. |
| | Checksum command | √ | √ | √ | Reads the checksum in the specified address range. |
| Read | Read command | √ | √ | √ | Reads the data written to flash memory. |
| System setting, control | Silicon signature command | √ | √ | √ | Reads silicon signature information. |
| | Security setting command | √ | √ | √ | Prohibits the chip erase command, block erase command, program command, read command, and boot area rewrite. |

### 31.4.6  Pin connection

When performing on-board writing, mount a connector on the target system to connect to the flash memory programmer.  Also, incorporate a function on-board to switch from the normal operation mode to the flash memory programming mode.

In the flash memory programming mode, all the pins not used for flash memory programming become the same status as that immediately after reset.  Therefore, pin handling is required when the external device does not acknowledge the status immediately after a reset.

### (1)  FLMD0 pin

In the normal operation mode, input a voltage of $V_{SS}$ level to the FLMD0 pin.  In the flash memory programming mode, supply a write voltage of $V_{DD}$ level to the FLMD0 pin.

Because the FLMD0 pin serves as a write protection pin in the self programming mode, a voltage of $V_{DD}$ level must be supplied to the FLMD0 pin via port control, etc., before writing to the flash memory.  For details, see **31.5.5 (1) FLMD0 pin**.

**Figure 31-11.  FLMD0 Pin Connection Example**
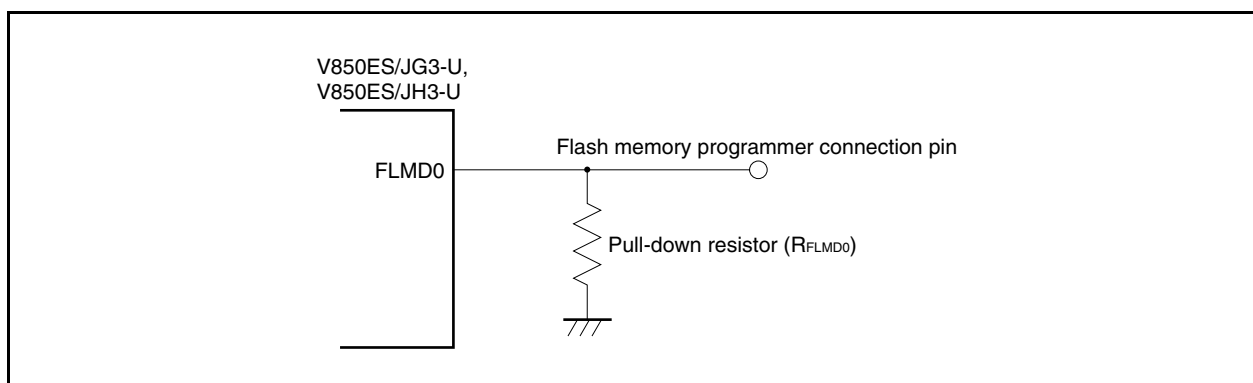


### (2)  FLMD1 pin

When 0 V is input to the FLMD0 pin, the FLMD1 pin does not function.  When $V_{DD}$ is supplied to the FLMD0 pin, the flash memory programming mode is entered, so 0 V must be input to the FLMD1 pin.  The following shows an example of the connection of the FLMD1 pin.

**Figure 31-12.  FLMD1 Pin Connection Example**



**Caution   If the $V_{DD}$ signal is input to the FLMD1 pin from another device during on-board writing and immediately after reset, isolate this signal.**
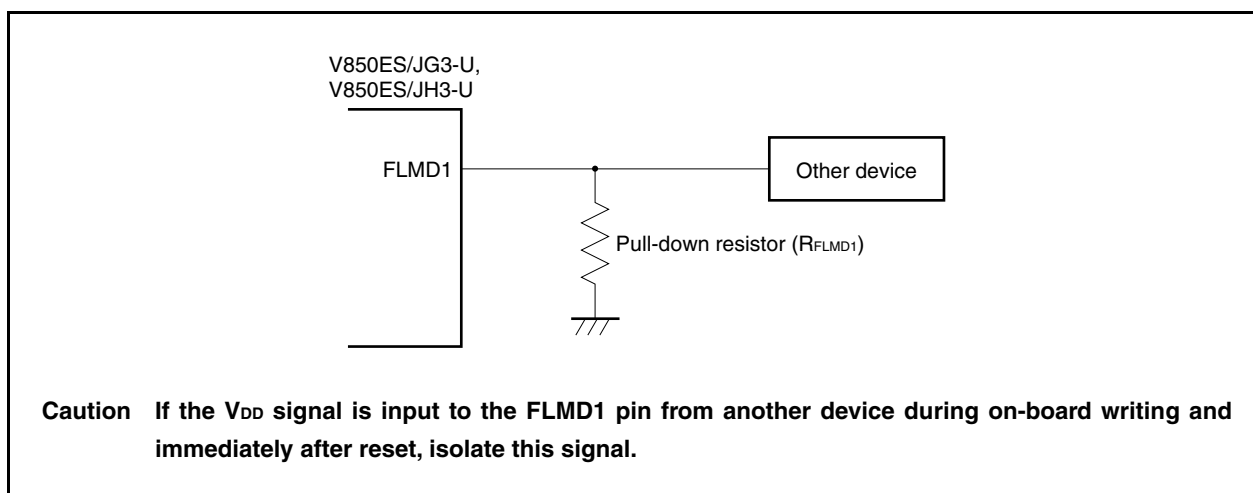
**Table 31-9.  Relationship Between FLMD0 and FLMD1 Pins and Operation Mode When Reset Is Released**

| FLMD0 | FLMD1 | Operation Mode |
|---|---|---|
| 0 | Don't care | Normal operation mode |
| V$_{DD}$ | 0 | Flash memory programming mode |
| V$_{DD}$ | V$_{DD}$ | Setting prohibited |

**(3)  Serial interface pin**

The following shows the pins used by each serial interface.

**Table 31-10.  Pins Used by Serial Interfaces**

| Serial Interface | Pins Used |
|---|---|
| UARTC0 | TXDC0, RXDC0 |
| CSIF0 | SOF0, SIF0, $\overline{SCKF0}$ |
| CSIF3 | SOF3, SIF3, $\overline{SCKF3}$ |
| CSIF0 + HS | SOF0, SIF0, $\overline{SCKF0}$, P913 |
| CSIF3 + HS | SOF3, SIF3, $\overline{SCKF3}$, P913 |

When connecting a flash memory programmer to a serial interface pin that is connected to another device on-board, care should be taken to avoid conflict of signals and malfunction of the other device.

**(a)  Conflict of signals**

When the flash memory programmer (output) is connected to a serial interface pin (input) that is connected to another device (output), a conflict of signals occurs.  To avoid the conflict of signals, isolate the connection to the other device or set the other device to the output high-impedance status.

**Figure 31-13.  Conflict of Signals (Serial Interface Input Pin)**
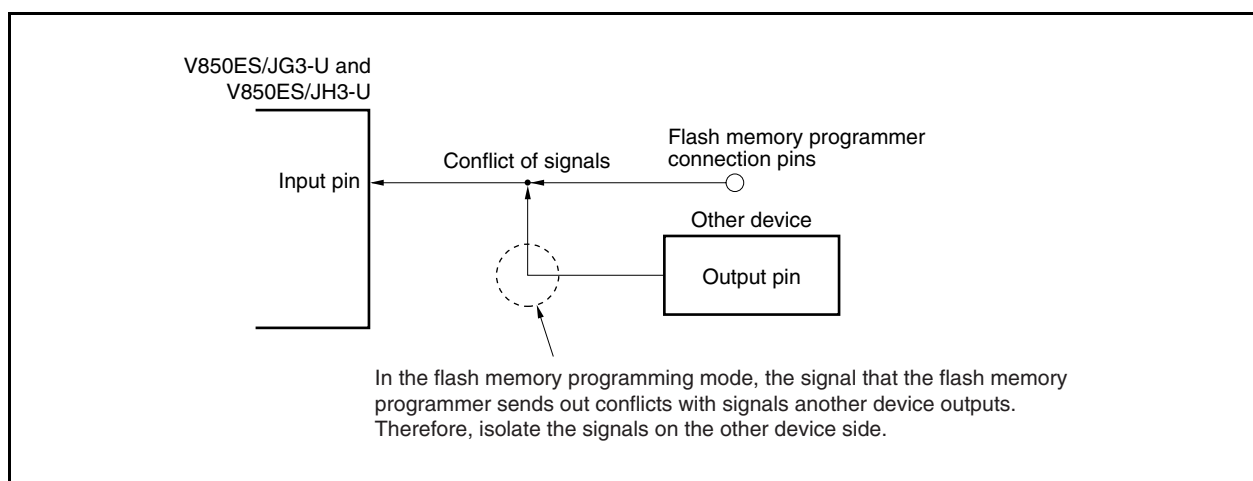
**(b) Malfunction of other device**

When the flash memory programmer (output or input) is connected to a serial interface pin (input or output) that is connected to another device (input), the signal is output to the other device, causing the device to malfunction. To avoid this, isolate the connection to the other device.

**Figure 31-14. Malfunction of Other Device**
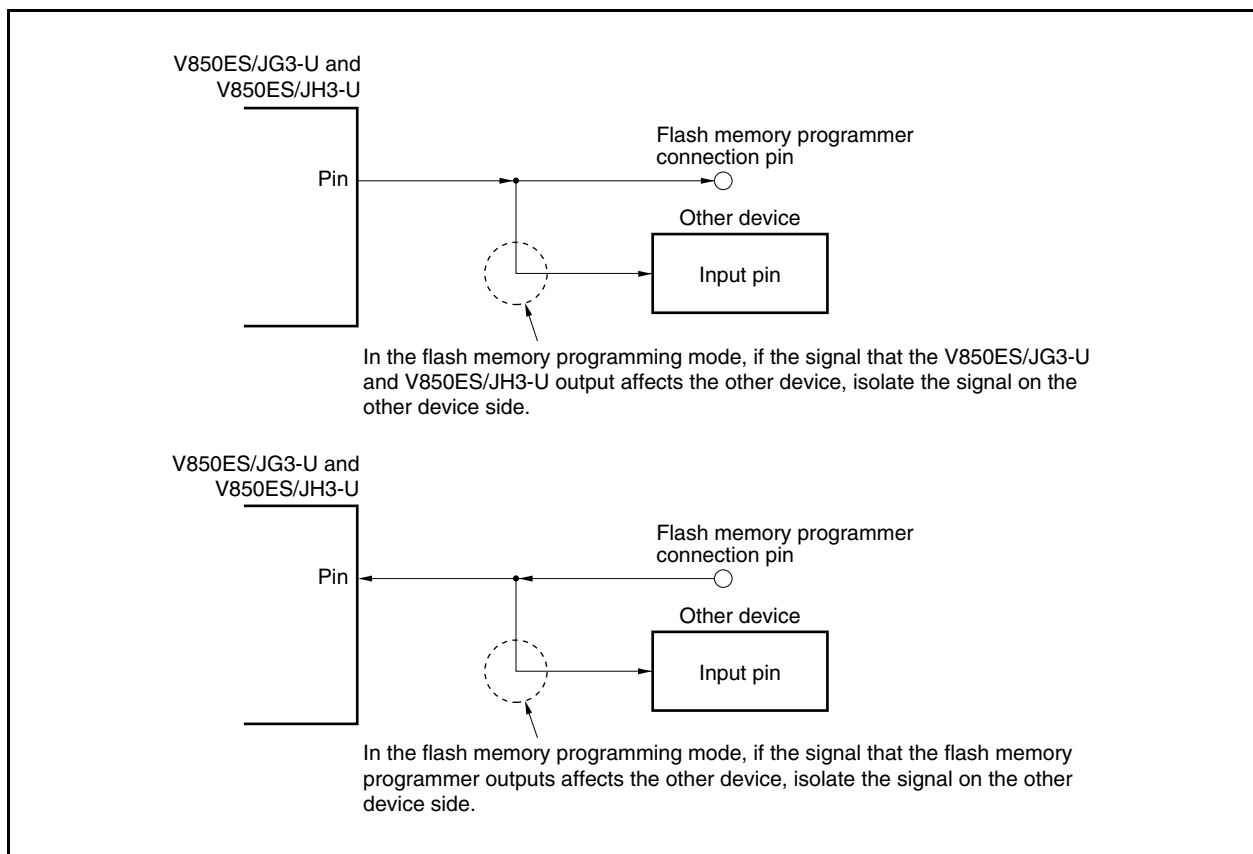
**(4)  $\overline{\text{RESET}}$ pin**

When the reset signals of the flash memory programmer are connected to the $\overline{\text{RESET}}$ pin that is connected to the reset signal generator on-board, a conflict of signals occurs.  To avoid the conflict of signals, isolate the connection to the reset signal generator.

When a reset signal is input from the user system in the flash memory programming mode, the programming operation will not be performed correctly.  Therefore, do not input signals other than the reset signals from the flash memory programmer.

**Figure 31-15.  Conflict of Signals ($\overline{\text{RESET}}$ Pin)**
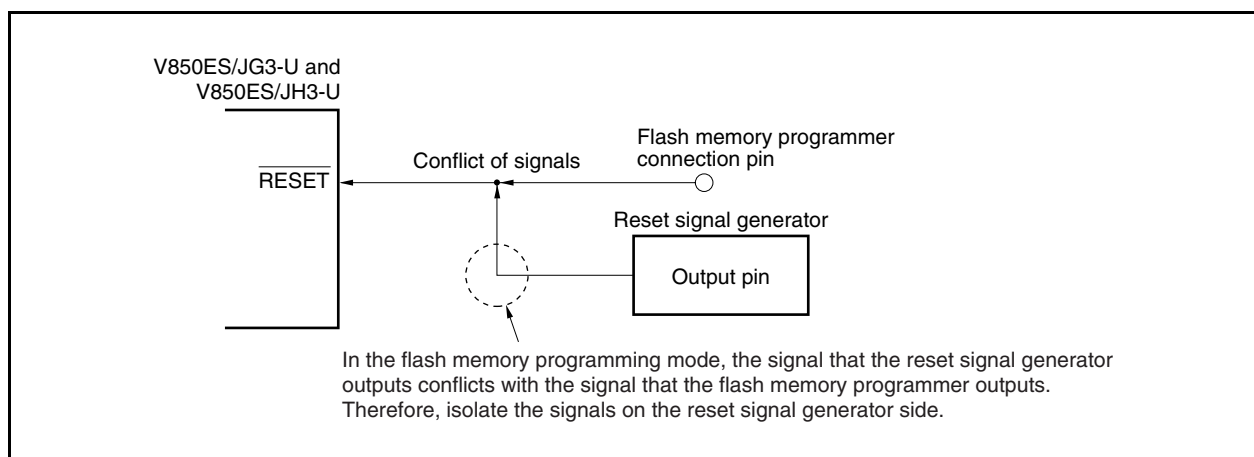


**(5)  Port pins (including NMI)**

When the system shifts to the flash memory programming mode, all the pins that are not used for flash memory programming are in the same status as that immediately after reset.  If the external device connected to each port does not recognize the status of the port immediately after reset, pins require appropriate processing, such as connecting to $V_{DD}$ via a resistor or connecting to $V_{SS}$ via a resistor.

**(6)  Other signal pins**

Connect X1, X2, XT1, XT2, and REGC in the same status as that in the normal operation mode.

During flash memory programming, input a low level to the $\overline{\text{DRST}}$ pin or leave it open.  Do not input a high level.

**(7)  Power supply**

Supply the same power ($V_{DD}$, $V_{SS}$, $EV_{DD}$, $UV_{DD}$, $AV_{REF0}$, $AV_{REF1}$, $AV_{SS}$) as in normal operation mode.

## 31.5 Rewriting by Self Programming

### 31.5.1 Overview

The V850ES/JG3-U and V850ES/JH3-U support a flash macro service that allows the user program to rewrite the internal flash memory by itself. By using this interface and a self programming library that is used to rewrite the flash memory with a user application program, the flash memory can be rewritten by a user application transferred in advance to the internal RAM or external memory. Consequently, the user program can be upgraded and constant data[Note] can be rewritten in the field.

**Note** Be sure not to allocate the program code to the block where the constant data of rewriting target is allocated. See **31.2 Memory Configuration** for the block configuration.

**Figure 31-16. Concept of Self Programming**
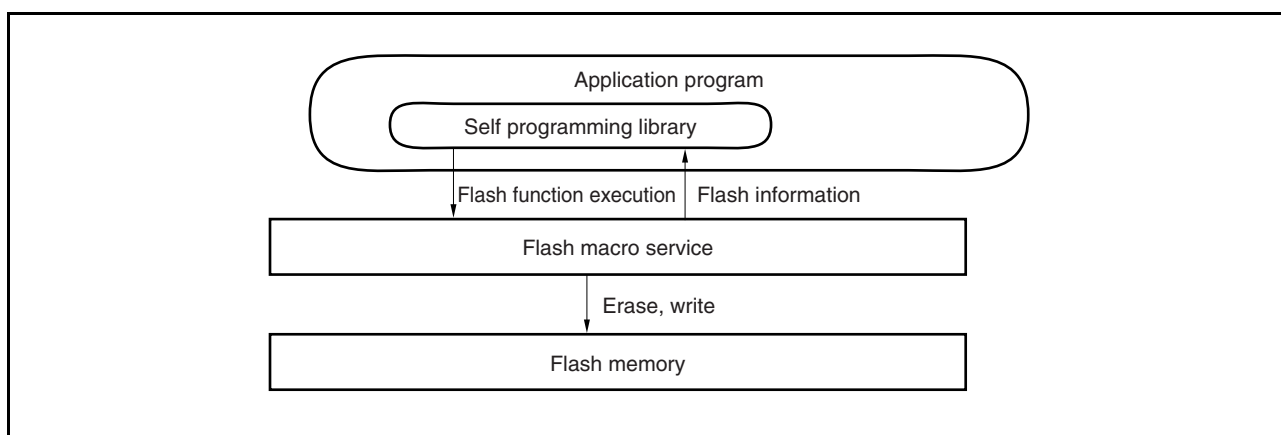
### 31.5.2  Features

**(1) Secure self programming (boot swap function)**

The V850ES/JG3-U and V850ES/JH3-U support a boot swap function that can exchange the physical memory of blocks 0 to 15 with the physical memory of blocks 16 to 31.  By writing the start program to be rewritten to blocks 16 to 31 in advance and then swapping the physical memory, the entire area can be safely rewritten even if a power failure occurs during rewriting because the correct user program always exists in blocks 0 to 15.

**Figure 31-17.  Rewriting Entire Memory Area (Boot Swap)**
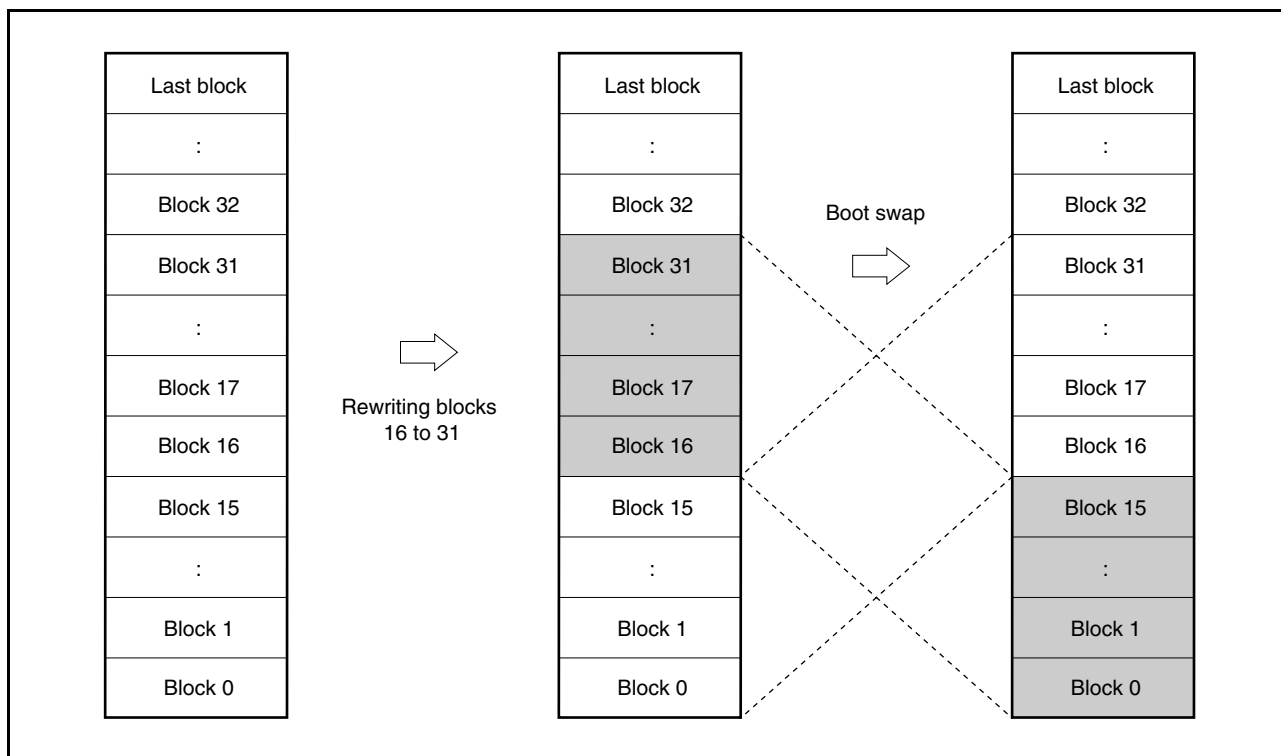


**(2) Interrupt support**

Instructions cannot be fetched from the flash memory during self-programming.  Consequently, a user handler written to the flash memory could not be used even if an interrupt has occurred.

Therefore, in the V850ES/JG3-U and V850ES/JH3-U, to use an interrupt during self-programming, processing transits to the specific address[Note] in the internal RAM.  Allocate the jump instruction that transits processing to the user interrupt servicing at the specific address[Note] in the internal RAM.

**Note**  NMI interrupt:          Start address of internal RAM
Maskable interrupt:  Start address of internal RAM + 4 addresses

### 31.5.3  Standard self programming flow

The entire processing to rewrite the flash memory by flash self programming is illustrated below.

**Figure 31-18.  Standard Self Programming Flow**
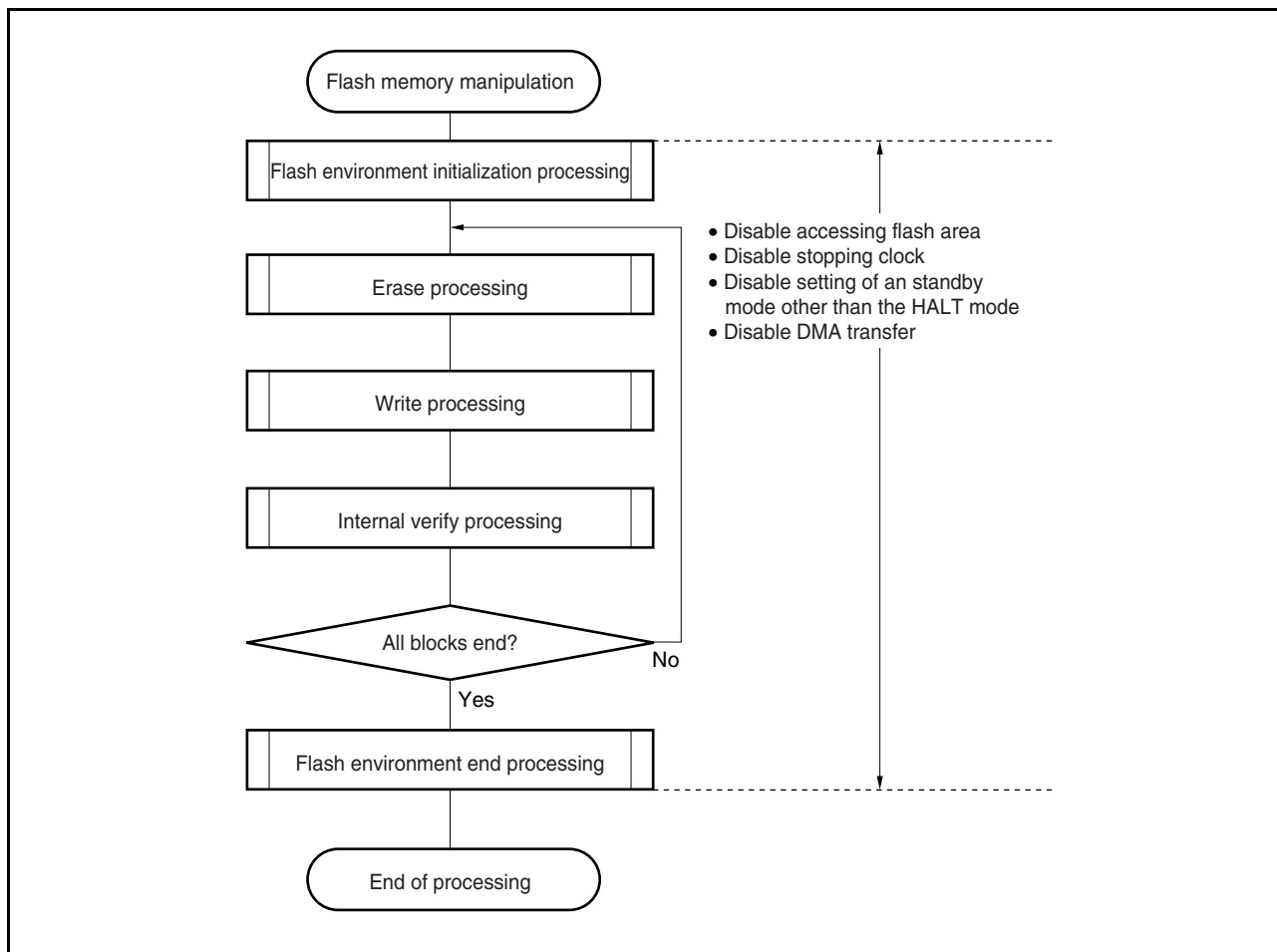
### 31.5.4  Flash functions

**Table 31-10.  Flash Function List**

| Function Name | Outline | Support |
|---|---|---|
| FlashInit | Self-programming library initialization | √ |
| FlashEnv | Flash environment start/end | √ |
| FlashFLMDCheck | FLMD pin check | √ |
| FlashStatusCheck | Hardware processing execution status check | √ |
| FlashBlockErase | Block erase | √ |
| FlashWordWrite | Data write | √ |
| FlashBlockIVerify | Internal verification of block | √ |
| FlashBlockBlankCheck | Blank check of block | √ |
| FlashSetInfo | Flash information setting | √ |
| FlashGetInfo | Flash information acquisition | √ |
| FlashBootSwap | Boot swap execution | √ |

### 31.5.5  Pin processing

#### (1)  FLMD0 pin

The FLMD0 pin is used to set the operation mode when reset is released and to protect the flash memory from being written during self rewriting.  It is therefore necessary to keep the voltage applied to the FLMD0 pin at 0 V when reset is released and a normal operation is executed.  It is also necessary to apply a voltage of $V_{DD}$ level to the FLMD0 pin during the self programming mode period via port control before the memory is rewritten.

When self programming has been completed, the voltage on the FLMD0 pin must be returned to 0 V.

**Figure 31-19.  Mode Change Timing**
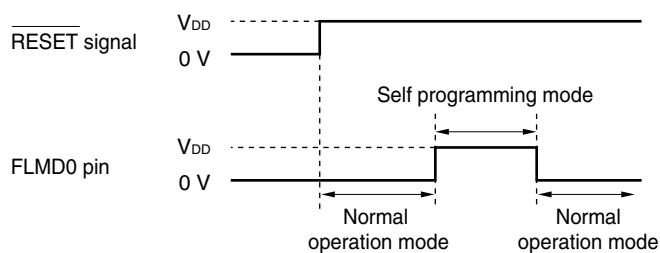


**Caution    Make sure that the FLMD0 pin is at 0 V when reset is released.**

### 31.5.6 Internal resources used

The following table lists the internal resources used for self programming. These internal resources can also be used freely for purposes other than self programming.

**Table 31-11. Internal Resources Used**

| Resource Name | Description |
|---|---|
| Stack area | An extension of the stack used by the user is used by the library (can be used in both the internal RAM and external RAM). |
| Library code[Note] | Program entity of library (can be used anywhere other than the flash memory block to be manipulated). |
| Application program | Executed as user application.<br>Calls flash functions. |
| Maskable interrupt | Can be used in the user application execution status or self-programming status. To use this interrupt in the self-programming status, since the processing transits to the address of the internal RAM start address + 4 addresses, allocate the jump instruction that transits the processing to the user interrupt servicing at the address of the internal RAM start address + 4 addresses in advance. |
| NMI interrupt | Can be used in the user application execution status or self-programming status. To use this interrupt in the self-programming status, since the processing transits to the address of the internal RAM start address, allocate the jump instruction that transits the processing to the user interrupt servicing at the internal RAM start address in advance. |

**Note** About resources used, refer to the **Flash Memory Self-Programming Library User's Manual**.

## 31.6 Creating ROM code to place order for previously written product

Before placing an order with Renesas Electronics for a previously written product, the ROM code for the order must be created.

To create the ROM code, use the Hex Consolidation Utility (hereafter abbreviated to HCU) on the finished programs (hex files) and optional data (such as security settings for flash memory programs).

The HCU is a software tool that includes functions required for creating ROM code.

The HCU can be downloaded at the Renesas Electronics website.

**(1) Website**

http://www2.renesas.com/micro/en/ods/     → Click Version-up Service.

**(2) Downloading the HCU**

To download the HCU, click Software for previously written flash products and then HCU_GUI.

**Remark** For details about how to install and use the HCU, see the materials (the user's manual) that comes with the HCU at the above website.

### 31.6.1 Procedure for using ROM code to place an order

Use the HCU to create the ROM code by following the procedure below, and then place your order with Renesas Electronics. For details, see the ROM Code Ordering Method Information (C10302J).



| Customer | Renesas Electronics |
|---|---|
| Decide which product to order. | |

Send the order information.

Renesas Electronics processes the product name and number and creates a record of the transaction.

Create the ROM code[Note]

Check the ROM order details and generate the required data.

Renesas Electronics sends the order number and other order-related information.

Send the data required for the ROM order.

Renesas Electronics processes the ROM code.

**Note** Use the HCU to create the ROM code for the order.

# CHAPTER 32  ON-CHIP  DEBUG  FUNCTION

The on-chip debug function of the V850ES/JG3-U and V850ES/JH3-U can be implemented by the following two methods.

- Using the DCU (debug control unit)
  On-chip debug function is implemented by the on-chip DCU in the V850ES/JG3-U and V850ES/JH3-U, with using the $\overline{\text{DRST}}$, DCK, DMS, DDI, and DDO pins as the debug interface pins.
- Not using the DCU
  On-chip debug function is implemented by MINICUBE2 or the like, using the user resources, instead of the DCU.

The following table shows the features of the two on-chip debug functions.

**Table 32-1.  On-Chip Debug Function Features**

| | Debugging Using DCU | Debugging Without Using DCU |
|---|---|---|
| Debug interface pins | $\overline{\text{DRST}}$, DCK, DMS, DDI, DDO | • When UARTC0 is used<br>  RXDC0, TXDC0<br><br>• When CSIF0 is used<br>  SIF0, SOF0, $\overline{\text{SCKF0}}$, HS (P913)<br><br>• When CSIF3 is used<br>  SIF3, SOF3, $\overline{\text{SCKF3}}$, HS (P913) |
| Securement of user resources | Not required | Required |
| Hardware break function | 2 points | 2 points |
| Software break function — Internal ROM area | 4 points | 4 points |
| Software break function — Internal RAM area | 2000 points | 2000 points |
| Real-time RAM monitor function[Note 1] | Available | Available |
| Dynamic memory modification (DMM) function[Note 2] | Available | Available |
| Mask function | Reset, NMI, INTWDT2, $\overline{\text{HLDRQ}}$, $\overline{\text{WAIT}}$ | $\overline{\text{RESET}}$ pin |
| ROM security function | 10-byte ID code authentication | 10-byte ID code authentication |
| Hardware used | NINICUBE, etc. | NINICUBE2, etc. |
| Trace function | Not supported. | Not supported. |
| Debug interrupt interface function (DBINT) | Not supported. | Not supported. |

**Notes  1.** This is a function which reads out memory contents during program execution.
**2.** This is a function which rewrites RAM contents during program execution.

## 32.1  Debugging with DCU

Programs can be debugged using the debug interface pins ($\overline{\text{DRST}}$, DCK, DMS, DDI, and DDO) to connect the on-chip debug emulator (MINICUBE).

### 32.1.1  Connection circuit example

**Figure 32-1.  Circuit Connection Example When Debug Interface Pins Are Used for Communication Interface**



Notes 1.  Example of pin connection when MINICUBE is not connected
      2.  A pull-down resistor is provided on chip.
      3.  For flash memory rewriting
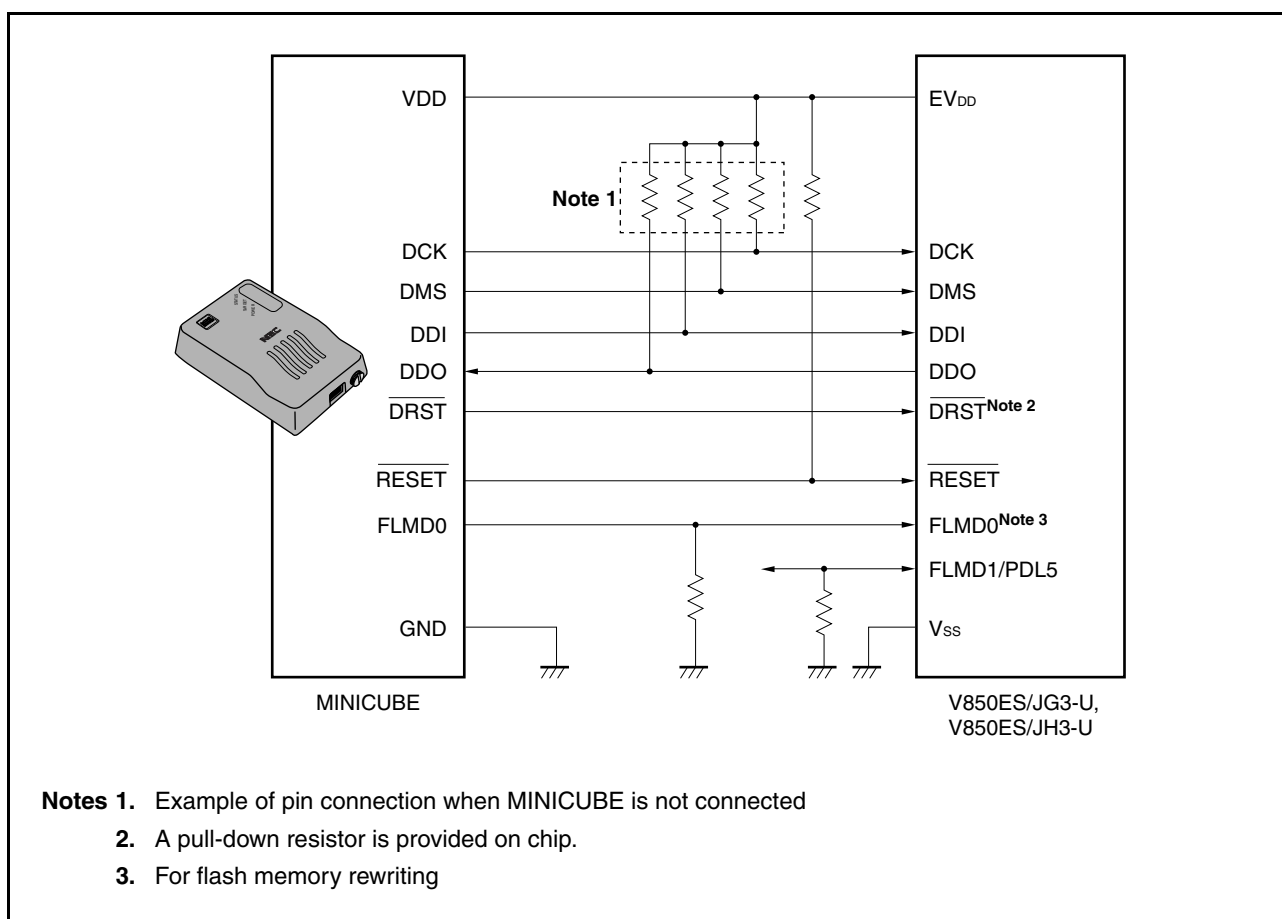
### 32.1.2  Interface signals
The interface signals are described below.

**(1)  $\overline{\text{DRST}}$**

This is a reset input signal for the on-chip debug unit.  It is a negative-logic signal that asynchronously initializes the debug control unit.

MINICUBE raises the $\overline{\text{DRST}}$ signal when it detects $V_{DD}$ of the target system after the integrated debugger is started, and starts the on-chip debug unit of the device.

When the $\overline{\text{DRST}}$ signal goes high, a reset signal is also generated in the CPU.

When starting debugging by starting the integrated debugger, a CPU reset is always generated.

**(2)  DCK**

This is a clock input signal.  It supplies a 20 MHz or 10 MHz clock from MINICUBE.  In the on-chip debug unit, the DMS and DDI signals are sampled at the rising edge of the DCK signal, and the data DDO is output at its falling edge.

**(3)  DMS**

This is a transfer mode select signal.  The transfer status in the debug unit changes depending on the level of the DMS signal.

**(4)  DDI**

This is a data input signal.  It is sampled in the on-chip debug unit at the rising edge of DCK.

**(5)  DDO**

This is a data output signal.  It is output from the on-chip debug unit at the falling edge of the DCK signal.

**(6)  EV$_{DD}$**

This signal is used to detect VDD of the target system.  If VDD from the target system is not detected, the signals output from MINICUBE ($\overline{DRST}$, DCK, DMS, DDI, FLMD0, and $\overline{RESET}$) go into a high-impedance state.

**(7)  FLMD0**

The flash self programming function is used for the function to download data to the flash memory via the integrated debugger.  During flash self programming, the FLMD0 pin must be kept high.  In addition, connect a pull-down resistor to the FLMD0 pin.
The FLMD0 pin can be controlled in either of the following two ways.

<1>  To control from MINICUBE
Connect the FLMD0 signal of MINICUBE to the FLMD0 pin.
In the normal mode, nothing is driven by MINICUBE (high impedance).
During a break, MINICUBE raises the FLMD0 pin to the high level when the download function of the integrated debugger is executed.

<2>  To control from port
Connect any port of the device to the FLMD0 pin.
The same port as the one used by the user program to realize the flash self programming function may be used.
On the console of the integrated debugger, make a setting to raise the port pin to high level before executing the download function, or lower the port pin after executing the download function.
For details, refer to the **ID850QB Ver. 3.40 Integrated Debugger Operation User's Manual  (U18604E)**.

**(8)  $\overline{RESET}$**

This is a system reset input pin.  If the $\overline{DRST}$ pin is made invalid by the value of the OCDM0 bit of the OCDM register set by the user program, on-chip debugging cannot be executed.  Therefore, reset is effected by MINICUBE, using the $\overline{RESET}$ pin, to make the $\overline{DRST}$ pin valid (initialization).

### 32.1.3  Maskable functions

Reset, NMI, INTWDT2, $\overline{\text{WAIT}}$, and $\overline{\text{HLDRQ}}$ signals can be masked.

The maskable functions with the debugger (ID850QB) and the corresponding V850ES/JG3-U and V850ES/JH3-U functions are listed below.

**Table 32-2.  Maskable Functions**

| Maskable Functions with ID850QB | Corresponding V850ES/JG3-U and V850ES/JH3-U Functions |
|---|---|
| NMI0 | NMI pin input |
| NMI2 | Non-maskable interrupt request signal (INTWDT2) generation |
| STOP | – |
| HOLD | $\overline{\text{HLDRQ}}$ pin input |
| RESET | Reset signal generation by $\overline{\text{RESET}}$ pin input, low-voltage detector, clock monitor, or watchdog timer (WDT2) overflow |
| $\overline{\text{WAIT}}$ | $\overline{\text{WAIT}}$ pin input |

### 32.1.4  Register

**(1)  On-chip debug mode register (OCDM) (V850ES/JG3-U only)**

The OCDM register is used to select the normal operation mode or on-chip debug mode.  This register is a special register and can be written only in a combination of specific sequences (see **3.4.7  Special registers**).

This register is also used to specify whether a pin provided with an on-chip debug function is used as an on-chip debug pin or as an ordinary port/peripheral function pin.  It also is used to disconnect the internal pull-down resistor of the P56/INTP05/$\overline{\text{DRST}}$ pin.

The OCDM register can be written only while a low level is input to the $\overline{\text{DRST}}$ pin.

This register can be read or written in 8-bit or 1-bit units.

After reset: 01H[Note]     R/W     Address: FFFFF9CH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| OCDM | 0 | 0 | 0 | 0 | 0 | 0 | 0 | OCDM0 |

| OCDM0 | Operation mode |
|---|---|
| 0 | Selects normal operation mode (in which a pin that functions alternately as on-chip debug function pin is used as a port/peripheral function pin) and disconnects the on-chip pull-down resistor of the P56/INTP05/$\overline{\text{DRST}}$ pin. |
| 1 | When $\overline{\text{DRST}}$ pin is low:<br>Normal operation mode (in which a pin that functions alternately as an on-chip debug function pin is used as a port/peripheral function pin)<br>When $\overline{\text{DRST}}$ pin is high:<br>On-chip debug mode (in which a pin that functions alternately as an on-chip debug function pin is used as an on-chip debug mode pin) |

**Note** $\overline{\text{RESET}}$ input sets this register to 01H. After reset by the WDT2RES signal, clock monitor (CLM), or low-voltage detector (LVI), however, the value of the OCDM register is retained.

**Cautions 1. When using the DDI, DDO, DCK, and DMS pins not as on-chip debug pins but as port pins after external reset, any of the following actions must be taken.**

- **Input a low level to the P56/INTP05/$\overline{\text{DRST}}$ pin.**
- **Set the OCDM0 bit. In this case, take the following actions.**
  **<1> Clear the OCDM0 bit to 0.**
  **<2> Fix the P56/INTP05/$\overline{\text{DRST}}$ pin to low level until <1> is completed.**

**2. The $\overline{\text{DRST}}$ pin has an on-chip pull-down resistor. This resistor is disconnected when the OCDM0 flag is set to 0.**
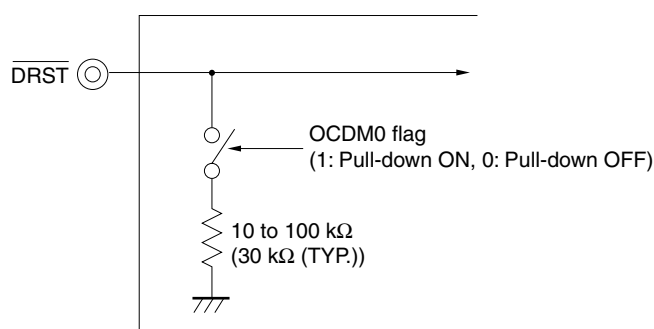
### 32.1.5  Operation

The on-chip debug function is made invalid under the conditions shown in the table below.

When this function is not used, keep the $\overline{\text{DRST}}$ pin low until the OCDM.OCDM0 flag is cleared to 0.

| OCDM0 Flag / $\overline{\text{DRST}}$ Pin | 0 | 1 |
|---|---|---|
| L | Invalid | Invalid |
| H | Invalid | Valid |

**Remark**   L: Low-level input

H: High-level input

**Figure 32-2.  Timing When On-Chip Debug Function Is Not Used**
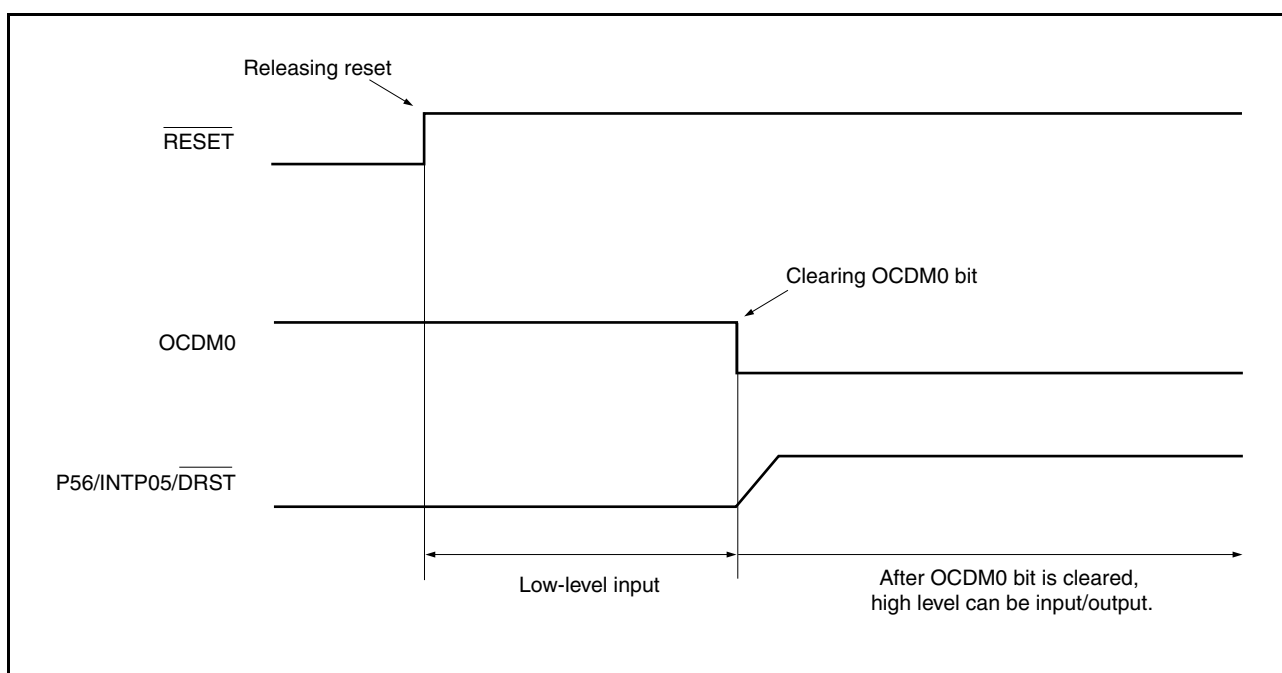


### 32.1.6  Cautions

(1) If a reset signal is input (from the target system or a reset signal from an internal reset source) during RUN (program execution), the break function may malfunction.

(2) Even if the reset signal is masked by the mask function, the I/O buffer (port pin) may be reset if a reset signal is input from a pin.

(3) Pin reset during a break is masked and the CPU and peripheral I/O are not reset.  If pin reset or internal reset is generated as soon as the flash memory is rewritten by DMM or read by the RAM monitor function while the user program is being executed, the CPU and peripheral I/O may not be correctly reset.

(4) In the on-chip debug mode, the DDO pin is forcibly set to the high-level output.

## 32.2  Debugging Without Using DCU

The following describes how to implement an on-chip debug function using MINICUBE2 with pins for UARTC0 (RXDC0 and TXDC0), pins for CSIF0 (SIF0, SOF0, $\overline{\text{SCKF0}}$, and HS (P913)), or pins for CSIF3 (SIF3, SOF3, $\overline{\text{SCKF3}}$, and HS (P913)) as debug interfaces, without using the DCU.

### 32.2.1  Circuit connection examples

**Figure 32-3.  Circuit Connection Example When UARTC0/CSIF0/CSIF3 Is Used for Communication Interface**



**Notes 1.** Connect TXDC0/SOF0/SOF3 (transmit side) of the V850ES/JG3-U or V850ES/JH3-U to RXD/SI (receive side) of the target connector, and TXD/SO (transmit side) of the target connector to RXDC0/SIF0/SIF3 (receive side) of the V850ES/JG3-U or V850ES/JH3-U.

   **2.** This pin may be used to supply a clock from MINICUBE2 during flash memory programming. For details, refer to **CHAPTER 31  FLASH MEMORY**.

   **3.** Because this pin is an (unused) input during debugging, its alternate function can be used. Note that, within MINICUBE2, a 100 kΩ pull-down resistor is connected.

   **4.** This connection is designed assuming that the $\overline{\text{RESET}}$ signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less).

   **5.** The circuit enclosed by a dashed line is designed for flash self programming, which controls the FLMD0 pin via ports. Use the port for inputting or outputting the high level. When flash self programming is not performed, a pull-down resistance for the FLMD0 pin can be within 1 to 10 kΩ.

**Remark** Refer to **Table 32-3** for pins used when UARTC0, CSIF0, or CSIF3 is used for communication interface.

**Table 32-3.  Wiring Between V850ES/JG3-U and MINICUBE2**

| Pin Configuration of MINICUBE2 (QB-MINI2) | | | With CSIF0-HS | | With CSIF3-HS | | With UARTC0 | |
|---|---|---|---|---|---|---|---|---|
| Signal Name | I/O | Pin Function | Pin Name | Pin No. | Pin Name | Pin No. | Pin Name | Pin No. |
| SI/RxD | Input | Pin to receive commands and data from V850ES/JG3-U | P41/SOF0 | 23 | P911/SOF3 | 53 | P30/TXDC0 | 25 |
| SO/TxD | Output | Pin to transmit commands and data to V850ES/JG3-U | P40/SIF0 | 22 | P910/SIF3 | 52 | P31/RXDC0 | 26 |
| SCK | Output | Clock output pin for 3-wire serial communication | P42/$\overline{\text{SCKF0}}$ | 24 | P912/$\overline{\text{SCKF3}}$ | 54 | Not needed | – |
| CLK[Note] | Output | Clock output pin to V850ES/JG3-U | Not needed[Note] | – | Not needed[Note] | – | Not needed[Note] | – |
| RESET_OUT | Output | Reset output pin to V850ES/JG3-U | $\overline{\text{RESET}}$ | 14 | $\overline{\text{RESET}}$ | 14 | $\overline{\text{RESET}}$ | 14 |
| FLMD0 | Output | Output pin to set V850ES/JG3-U to debug mode or programming mode | FLMD0 | 8 | FLMD0 | 8 | FLMD0 | 8 |
| FLMD1 | Output | Output pin to set programming mode | PDL5/FLMD1 | 76 | PDL5/FLMD1 | 76 | PDL5/FLMD1 | 76 |
| HS | Input | Handshake signal for CSI0 + HS communication | P913 | 85 | P913 | 85 | Not needed | – |
| GND | – | Ground | Vss | 11, 33, 62, 79 | Vss | 11, 33, 62, 79 | Vss | 11, 33, 62, 79 |
| | | | AVss | 2 | AVss | 2 | AVss | 2 |
| RESET_IN | Input | Reset input pin on the target system | | | | | | |

**Note**  It is used as the clock output of the flash programmer for MINICUBE2.  For details, refer to **CHAPTER 31 FLASH MEMORY**.

**Table 32-4.  Wiring Between V850ES/JH3-U and MINICUBE2**

| Pin Configuration of MINICUBE2 (QB-MINI2) | | | With CSIF0-HS | | With CSIF3-HS | | With UARTC0 | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Signal Name | I/O | Pin Function | Pin Name | Pin No. | Pin Name | Pin No. | Pin Name | Pin No. |
| SI/RXD | Input | Pin to receive commands and data from V850ES/JH3-U | P41/SOF0 | 30 | P911/SOF3 | 67 | P30/TXDC0 | 37 |
| SO/TXD | Output | Pin to transmit commands and data to V850ES/JH3-U | P40/SIF0 | 29 | P910/SIF3 | 66 | P31/RXDC0 | 31 |
| $\overline{SCK}$ | Output | Clock output pin for 3-wire serial communication | P42/$\overline{SCKF0}$ | 31 | P912/$\overline{SCKF3}$ | 68 | Not needed | – |
| CLK**Note** | Output | Clock output pin to V850ES/JH3-U | Not needed**Note** | – | Not needed**Note** | – | Not needed**Note** | – |
| RESET_OUT | Output | Reset output pin to V850ES/JH3-U | $\overline{RESET}$ | 18 | $\overline{RESET}$ | 18 | $\overline{RESET}$ | 18 |
| FLMD0 | Output | Output pin to set V850ES/JH3-U to debug mode or programming mode | FLMD0 | 12 | FLMD0 | 12 | FLMD0 | 12 |
| FLMD1 | Output | Output pin to set programming mode | PDL5/FLMD1 | 103 | PDL5/FLMD1 | 103 | PDL5/FLMD1 | 103 |
| HS | Input | Handshake signal for CSI0 + HS communication | P913 | 69 | P913 | 69 | Not needed | – |
| GND | – | Ground | V$_{SS}$ | 15, 45, 84, 106 | V$_{SS}$ | 15, 45, 84, 106 | V$_{SS}$ | 15, 45, 84, 106 |
| | | | AV$_{SS}$ | 2 | AV$_{SS}$ | 2 | AV$_{SS}$ | 2 |
| RESET_IN | Input | Reset input pin on the target system | | | | | | |

**Note**  It is used as the clock output of the flash programmer for MINICUBE2.  For details, refer to **CHAPTER 31  FLASH MEMORY**.

### 32.2.2 Maskable functions

Only reset signals can be masked.

The functions that can be masked in the debugger (ID850QB) and the corresponding functions of the V850ES/JG3-U and V850ES/JH3-U are listed below.

**Table 32-5. Maskable Functions**

| Functions Maskable in ID850QB | Corresponding Functions of V850ES/JG3-U and V850ES/JH3-U |
|---|---|
| NMI0 | – |
| NMI1 | – |
| NMI2 | – |
| STOP | – |
| HOLD | – |
| RESET | Reset signal generation by $\overline{\text{RESET}}$ pin input |
| WAIT | – |

**32.2.3  Securement of user resources**

The user must prepare the following to perform communication between MINICUBE2 and the target device and implement each debug function. These items need to be set in the user program or using the compiler options.

**(1)  Securement of memory space**

The shaded portions in Figure 32-4 are the areas reserved for placing the debug monitor program, so user programs and data cannot be allocated in these spaces.  These spaces must be secured so as not to be used by the user program.

**(2)  Security ID setting**

The ID code must be embedded in the area between 0000070H and 0000079H in Figure 32-4, to prevent the memory from being read by an unauthorized person.  For details, refer to **32.3  ROM Security Function**.

**Figure 32-4.  Memory Spaces Where Debug Monitor Programs Are Allocated**



**Notes 1.**  Address values vary depending on the product.

|  | Internal ROM Size | Address Value |
|---|---|---|
| μPD70F3763, 70F3768 | 384 KB | 005F800H to 003FFFFH |
| μPD70F3764, 70F3769 | 512 KB | 007F800H to 007FFFFH |

**2.**  This is the address when CSIF0 is used.  It starts at 0000406H when CSIF3 is used, and at 00004A0H when UARTC0 is used.

**3.**  Address values vary depending on the product.

|  | Internal RAM Size | Address Value |
|---|---|---|
| μPD70F3763, 70F3768 | 40 KB | 3FF5000H |
| μPD70F3764, 70F3769 | 48 KB | 3FF3000H |

**(3)  Reset vector**

A reset vector includes the jump instruction for the debug monitor program.

[How to secure areas]

It is not necessary to secure this area intentionally.  When downloading a program, however, the debugger rewrites the reset vector in accordance with the following cases.  If the rewritten pattern does not match the following cases, the debugger generates an error (F0C34 when using the ID850QB).

**(a)  When two nop instructions are placed in succession from address 0**

Before rewriting        After rewriting

0x0 nop      →      Jumps to debug monitor program at 0x0

0x2 nop              0x4  xxxx

0x4 xxxx

**(b)  When two 0xFFFF are successively placed from address 0 (already erased device)**

Before rewriting        After rewriting

0x0 0xFFFF    →    Jumps to debug monitor program at 0x0

0x2 0xFFFF         0x4  xxxx

0x4 xxxx

**(c)  The *jr* instruction is placed at address 0 (when using CA850)**

Before rewriting        After rewriting

0x0 jr disp22   →    Jumps to debug monitor program at 0x0

                       0x4 jr disp22 - 4

**(d)  *mov32* and *jmp* are placed in succession from address 0 (when using IAR compiler ICCV850)**

Before rewriting        After rewriting

0x0 mov imm32,reg1 → Jumps to debug monitor program at 0x0

0x6 jmp [reg1]     0x4  mov imm32,reg1

                      0xa jmp [reg1]

**(e)  The jump instruction for the debug monitor program is placed at address 0**

Before rewriting                      After rewriting

Jumps to debug monitor program at 0x0   →    No change

**(4)  Securement of area for debug monitor program**

The shaded portions in Figure 32-4 are the areas where the debug monitor program is allocated. The monitor program performs initialization processing for debug communication interface and RUN or break processing for the CPU.  The internal ROM area must be filled with 0xFF.  This area must not be rewritten by the user program.

[How to secure areas]

It is not necessarily required to secure this area if the user program does not use this area.

To avoid problems that may occur during the debugger startup, however, it is recommended to secure this area in advance, using the compiler.

The following shows examples for securing the area, using the Renesas Electronics compiler CA850.  Add the assemble source file and link directive code, as shown below.

• Assemble source (Add the following code as an assemble source file.)

```
 -- Secures 2 KB space for monitor ROM section
.section "MonitorROM", const
.space   0x800, 0xff

 -- Secures interrupt vector for debugging
.section "DBG0"
.space   4, 0xff

 -- Secures interrupt vector for serial communication
 -- Change the section name according to the serial communication mode used
.section "INTCF0R"
.space   4, 0xff

 -- Secures 16-byte space for monitor RAM section
.section "MonitorRAM", bss
.lcomm   monitorramsym, 16, 4    -- defines symbol monitorramsym
```

• Link directive (Add the following code to the link directive file.)

The following shows an example when the internal ROM has 512 KB (end address is 007FFFFFH) and internal RAM has 56 KB (end address is 3FFEFFFFH).

```
 MROMSEG      : !LOAD ?R V0x07f800{
              MonitorROM  = $PROGBITS  ?A MonitorROM;
 };

 MRAMSEG      : !LOAD ?RW V0x03ffeff0{
              MonitorRAM  = $NOBITS    ?AW MonitorRAM;
 };
```

**(5) Securement of communication serial interface**

UARTC0, CSIF0, or CSIF3 is used for communication between MINICUBE2 and the target system.  The settings related to the serial interface modes are performed by the debug monitor program, but if the setting is changed by the user program, a communication error may occur.

To prevent such a problem from occurring, communication serial interface must be secured in the user program.

[How to secure communication serial interface]

- On-chip debug mode register (OCDM)

  For the on-chip debug function using the UARTC0, CSIF0, or CSIF3, set the OCDM register functions to normal mode.  Be sure to set as follows.

  - Input low level to the P56/INTP05/$\overline{\text{DRST}}$ pin.
  - Set the OCDM0 bit as shown below.
    - <1> Clear the OCDM0 bit to 0.
    - <2> Fix the P56/INTP05/$\overline{\text{DRST}}$ pin input to low level until the processing of <1> is complete.

- Serial interface registers

  Do not set the registers related to CSIF0, CSIF3, or UARTC0 in the user program.

- Interrupt mask register

  When CSIF0 is used, do not mask the transfer end interrupt (INTCF0R).  When CSIF3 is used, do not mask the transfer end interrupt (INTCF3R).  When UARTC0 is used, do not mask the reception completion interrupt (INTUC0R).

---

**(a)  When CSIF0 is used**

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CF0RIC | × | 0 | × | × | × | × | × | × |

**(b)  When CSIF3 is used**

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CF3RIC | × | 0 | × | × | × | × | × | × |

**(C)  When UARTC0 is used**

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| UC0RIC | × | 0 | × | × | × | × | × | × |

**Remark**   ×: don't care

---

• Port registers when UARTC0 is used

When UARTC0 is used, port registers are set to make the TXDC0 and RXDC0 pins valid by the debug monitor program.  Do not change the following register settings with the user program during debugging. (The same value can be overwritten.)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC3 | × | × | × | × | × | × | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE3 | × | × | × | × | × | × | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC3 | × | × | × | × | × | × | 1 | 1 |

**Remark**  ×: don't care

• Port registers when CSIF0 is used

When CSIF0 is used, port registers are set to make the SIF0, SOF0, $\overline{SCKF0}$, and HS (P913) pins valid by the debug monitor program.  Do not change the following register settings with the user program during debugging. (The same value can be overwritten.)

**(a)  SIF0, SOF0, and $\overline{SCKF0}$ settings**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC4 | × | × | × | × | × | 1 | 1 | 1 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFC4 | × | × | × | × | × | 0 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFCE4 | × | × | × | × | × | × | 0 | 0 |

**(b)  HS (P913 pin) settings**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PM9H | × | × | 0 | × | × | × | × | × |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| P9H | × | × | Note | × | × | × | × | × |

**Note**  Writing to this bit is prohibited.

The port values corresponding to the HS pin are changed by the monitor program according to the debugger status.  To perform port register settings in 8-bit units, the user program can usually use read-modify-write. If an interrupt for debugging occurs before writing, however, an unexpected operation may be performed.

**Remark**  ×: don't care

- Port registers when CSIF3 is used

  When CSIF3 is used for communication, port registers are set to make the SIF3, SOF3, $\overline{SCKF3}$, and HS (P913) pins valid by the debug monitor program.  Do not change the following register settings with the user program during debugging. (The same value can be overwritten.)

---

**(a)  SIF3, SOF3, and $\overline{SCKF3}$ settings**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PMC9H | × | × | × | 1 | 1 | 1 | × | × |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFC9H | × | × | × | 0[Note1] | 0 | 0 | × | × |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PFCE9H | × | × | × | × | 0 | 0 | × | × |

**(b)  HS (P913 pin) settings**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PM9H | × | × | 0 | × | × | × | × | × |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| P9H | × | × | Note2 | × | × | × | × | × |

**Notes1.** V850ES/JH3-U only

**2.** Writing to this bit is prohibited.

The port values corresponding to the HS pin are changed by the monitor program according to the debugger status.  To perform port register settings in 8-bit units, the user program can usually use read-modify-write. If an interrupt for debugging occurs before writing, however, an unexpected operation may be performed.

**Remark**   ×: don't care

---

**32.2.4 Cautions**

**(1) Handling of device that was used for debugging**

Do not mount a device that was used for debugging on a mass-produced product, because the flash memory was rewritten during debugging and the number of rewrites of the flash memory cannot be guaranteed. Moreover, do not embed the debug monitor program into mass-produced products.

**(2) When breaks cannot be executed**

Forced breaks cannot be executed if one of the following conditions is satisfied.

- Interrupts are disabled (DI)
- Interrupts issued for the serial interface, which is used for communication between MINICUBE2 and the target device, are masked
- Standby mode is entered while standby release by a maskable interrupt is prohibited
- Mode for communication between MINICUBE2 and the target device is UARTC0, and the main clock has been stopped

**(3) When pseudo real-time RAM monitor (RRM) function and DMM function do not operate**

The pseudo RRM function and DMM function do not operate if one of the following conditions is satisfied.

- Interrupts are disabled (DI)
- Interrupts issued for the serial interface, which is used for communication between MINICUBE2 and the target device, are masked
- Standby mode is entered while standby release by a maskable interrupt is prohibited
- Mode for communication between MINICUBE2 and the target device is UARTC0, and the main clock has been stopped
- Mode for communication between MINICUBE2 and the target device is UARTC0, and a clock different from the one specified in the debugger is used for communication

**(4) Standby release with pseudo RRM and DMM functions enabled**

The standby mode is released by the pseudo RRM function and DMM function if one of the following conditions is satisfied.

- Mode for communication between MINICUBE2 and the target device is CSIF0 or CSIF3
- Mode for communication between MINICUBE2 and the target device is UARTC0, and the main clock has been supplied.

**(5) Rewriting to peripheral I/O registers that requires a specific sequence, using DMM function**

Peripheral I/O registers that requires a specific sequence cannot be rewritten with the DMM function.

**(6) Flash self programming**

If a space where the debug monitor program is allocated is rewritten by flash self programming, the debugger can no longer operate normally.

## 32.3  ROM Security Function

### 32.3.1  Security ID

The flash memory versions of the V850ES/JG3-U and V850ES/JH3-U perform authentication using a 10-byte ID code to prevent the contents of the flash memory from being read by an unauthorized person during on-chip debugging by the on-chip debug emulator.

Set the ID code in the 10-byte on-chip flash memory area from 0000070H to 0000079H to allow the debugger perform ID authentication.

If the IDs match, the security is released and reading flash memory and using the on-chip debug emulator are enabled.

- Set the 10-byte ID code to 0000070H to 0000079H.
- Bit 7 of 0000079H is the on-chip debug emulator enable flag.
  (0: Disable, 1: Enable)
- When the on-chip debug emulator is started, the debugger requests ID input.  When the ID code input on the debugger and the ID code set in 0000070H to 0000079H match, the debugger starts.
- Debugging cannot be performed if the on-chip debug emulator enable flag is 0, even if the ID codes match.

**Figure 32-5.  Security ID Area**



**Caution    After the flash memory is erased, 1 is written to the entire area.**
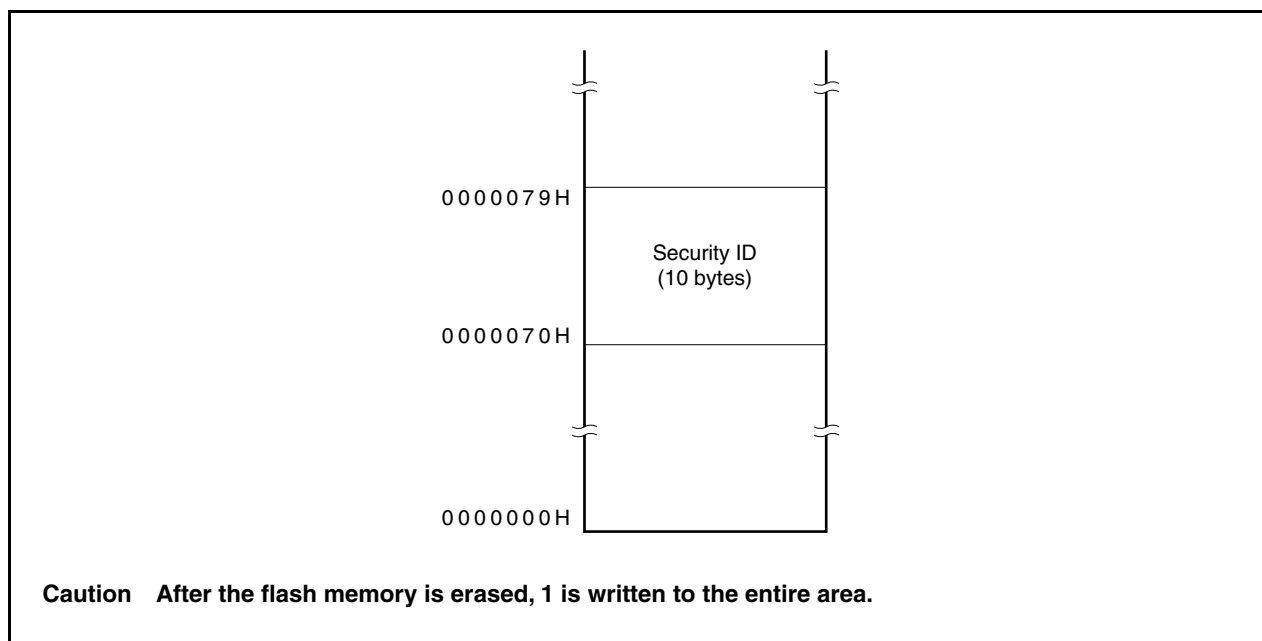
**32.3.2  Setting**

The following shows how to set the ID code as shown in Table 32-6.

When the ID code is set as shown in Table 32-6, the ID code input in the configuration dialog box of the ID850QB is "123456789ABCDEF123D4" (the ID code is case-insensitive).

**Table 32-6.  ID Code**

| Address | Value |
|---------|-------|
| 0x70 | 0x12 |
| 0x71 | 0x34 |
| 0x72 | 0x56 |
| 0x73 | 0x78 |
| 0x74 | 0x9A |
| 0x75 | 0xBC |
| 0x76 | 0xDE |
| 0x77 | 0XF1 |
| 0x78 | 0x23 |
| 0x79 | 0xD4 |

The ID code can be specified in the Compiler Common Options dialog box in PM+ if a device file that supports CA850 Ver. 3.10 and later and the security ID is used.
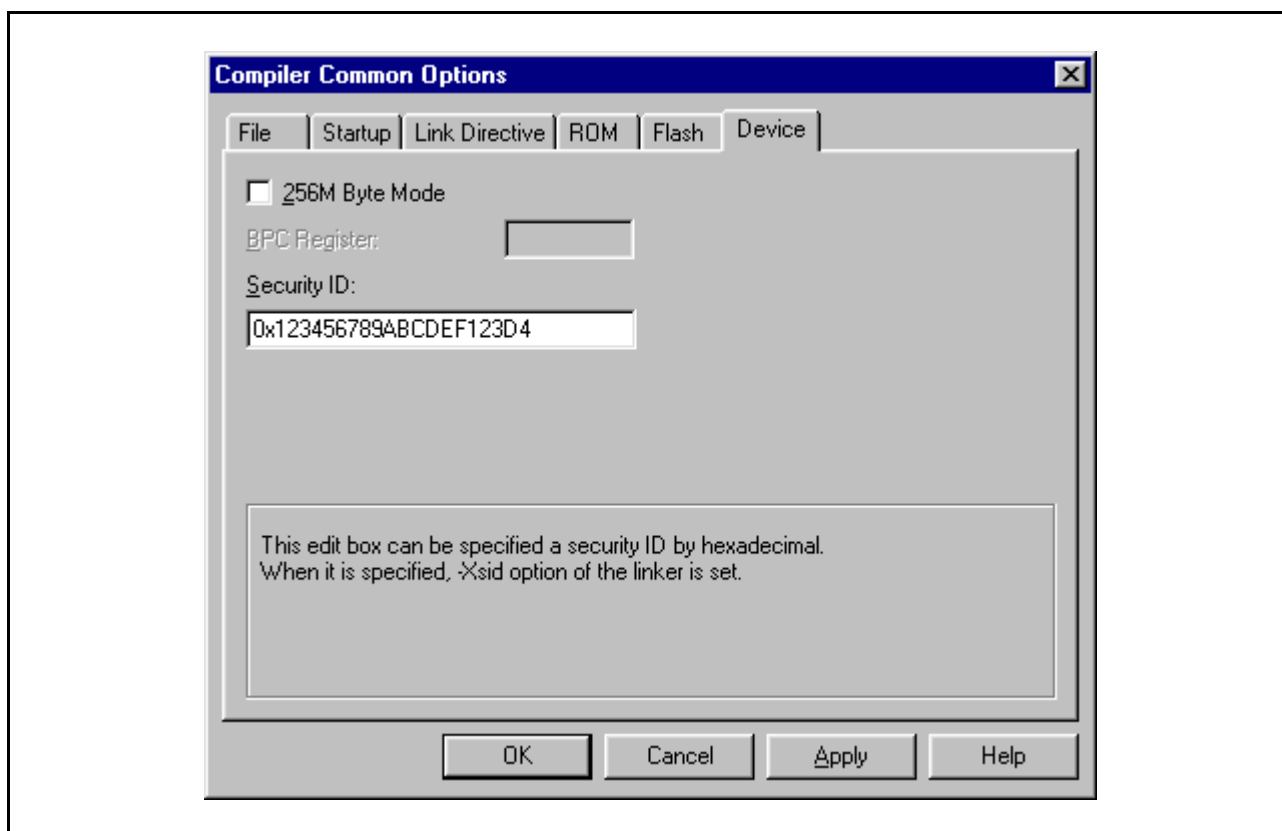
**[Program example (when using CA850 Ver. 3.10 or later)]**

```
#------------------------------------
#       SECURITYID
#------------------------------------
        .section    "SECURITY_ID"   --Interrupt handler address 0x70
        .word       0x78563412      --0-3 byte code
        .word       0xF1DEBC9A      --4-7 byte code
        .hword      0xD423          --8-9 byte code


   Remark   Add the above program example to the startup files.
```

# CHAPTER 33 ELECTRICAL SPECIFICATIONS

## 33.1 Absolute Maximum Ratings

**($T_A$ = 25°C) (1/2)**

| Parameter | Symbol | Conditions | Ratings | Unit |
|---|---|---|---|---|
| Supply voltage | $V_{DD}$ | $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$ | −0.5 to +4.6 | V |
| | $EV_{DD}$ | $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$ | −0.5 to +4.6 | V |
| | $UV_{DD}$ | $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$ | −0.5 to +4.6 | V |
| | $AV_{REF0}$ | $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$ | −0.5 to +4.6 | V |
| | $AV_{REF1}$ | $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$ | −0.5 to +4.6 | V |
| | $V_{SS}$ | $V_{SS}$ = $AV_{SS}$ | −0.5 to +0.5 | V |
| | $AV_{SS}$ | $V_{SS}$ = $AV_{SS}$ | −0.5 to +0.5 | V |
| Input voltage | $V_{I1}$ | P60 to P65, P90 to P915, PCM0, PCM1, PCS0, PCS2, PCS3, PCT0, PCT1, PCT4, PCT6, PDL0 to PDL15, PDH0 to PDH7, $\overline{RESET}$, FLMD0 | −0.5 to $EV_{DD}$ + 0.5[Note 1] | V |
| | $V_{I2}$ | UDMF, UDPF, UDMH, UDPH | −0.5 to $UV_{DD}$ + 0.5[Note 1] | V |
| | $V_{I3}$ | P10, P11 | −0.5 to $AV_{REF1}$ + 0.5[Note 1] | V |
| | $V_{I4}$ | X1, X2, XT1, XT2 | −0.5 to $V_{RO}$[Note 2] + 0.5[Note 1] | V |
| | $V_{I5}$ | P00 to P05, P20 to P25, P30 to P37, P40 to P42 P50 to P56, PCM2, PCM3 | −0.5 to +6.0 | V |
| Analog input voltage | $V_{IAN}$ | P70 to P711 | −0.5 to $AV_{REF0}$ + 0.5[Note 1] | V |

**Cautions 1. Do not directly connect the output (or I/O) pins of IC products to each other, or to $V_{DD}$, $V_{CC}$, and GND. Open-drain pins or open-collector pins, however, can be directly connected to each other. Direct connection of the output pins between an IC product and an external circuit is possible, if the output pins can be set to the high-impedance state and the output timing of the external circuit is designed to avoid output conflict.**

**2. Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage. Therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.**

**The ratings and conditions indicated for DC characteristics and AC characteristics represent the quality assurance range during normal operation.**

**Notes 1.** Be sure not to exceed the absolute maximum ratings (MAX. value) of each supply voltage.

**2.** On-chip regulator output voltage (2.5 V (TYP.))

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

**($T_A$ = 25°C) (2/2)**

| Parameter | Symbol | Conditions | | Ratings | Unit |
|---|---|---|---|---|---|
| Output current, low | $I_{OL}$ | P00 to P05, P20 to P25, P30 to P37, P40 to P42, P50 to P56, P60 to P65, P90 to P915, PCM0 to PCM3, PCS0, PCS2, PCS3, PCT0, PCT1, PCT4, PCT6, PDL0 to PDL15, PDH0 to PDH7 | Per pin | 4 | mA |
| | | | Total of all pins | 50 | mA |
| | | UDMF, UDPF | Per pin | 4 | mA |
| | | | Total of all pins | 8 | mA |
| | | UDMH, UDPH | Per pin | 4 | mA |
| | | | Total of all pins | 8 | mA |
| | | P10, P11 | Per pin | 4 | mA |
| | | | Total of all pins | 8 | mA |
| | | P70 to P711 | Per pin | 4 | mA |
| | | | Total of all pins | 20 | mA |
| Output current, high | $I_{OH}$ | P00 to P05, P20 to P25, P30 to P37, P40 to P42, P50 to P56, P60 to P65, P90 to P915, PCM0 to PCM3, PCS0, PCS2, PCS3, PCT0, PCT1, PCT4, PCT6, PDL0 to PDL15, PDH0 to PDH7 | Per pin | −4 | mA |
| | | | Total of all pins | −50 | mA |
| | | UDMF, UDPF, UDMH, UDPH | Per pin | −4 | mA |
| | | | Total of all pins | −8 | mA |
| | | P10, P11 | Per pin | −4 | mA |
| | | | Total of all pins | −8 | mA |
| | | P70 to P711 | Per pin | −4 | mA |
| | | | Total of all pins | −20 | mA |
| Operating ambient temperature | $T_A$ | In normal operation | | −40 to +85 | °C |
| | | In flash memory programming | | −40 to +85 | °C |
| Storage temperature | $T_{stg}$ | | | −40 to +125 | °C |

**Cautions 1.** Do not directly connect the output (or I/O) pins of IC products to each other, or to $V_{DD}$, $V_{CC}$, and GND. Open-drain pins or open-collector pins, however, can be directly connected to each other.  Direct connection of the output pins between an IC product and an external circuit is possible, if the output pins can be set to the high-impedance state and the output timing of the external circuit is designed to avoid output conflict.

      **2.** Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter.  That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage.  Therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

      The ratings and conditions indicated for DC characteristics and AC characteristics represent the quality assurance range during normal operation.

**Remark**    Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

## 33.2  Capacitance

($T_A$ = 25°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| I/O capacitance | $C_{IO}$ | $f_X$ = 1 MHz<br>Measured pins returned to 0 V | | | 10 | pF |

## 33.3  Operating Conditions

($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)
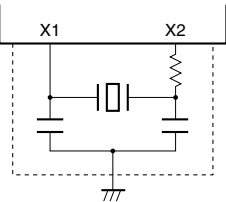
| Internal System Clock Frequency | Conditions | Supply voltage | | | | Unit |
|---|---|---|---|---|---|---|
| | | $V_{DD}$ | $EV_{DD}$ | $UV_{DD}$ | $AV_{REF0}$, $AV_{REF1}$ | |
| $f_{XX}$ = 3 to 6 MHz (clock-through operation)<br>$f_{XX}$ = 24 to 48 MHz (during PLL operation) | C = 4.7 $\mu$F,<br>A/D converter stopped,<br>D/A converter stopped,<br>USB stopped | 2.85 to 3.6 | 2.85 to 3.6 | 2.85 to 3.6 | 2.85 to 3.6 | V |
| | C = 4.7 $\mu$F,<br>A/D converter operating,<br>D/A converter operating,<br>USB operating | 3.0 to 3.6 | 3.0 to 3.6 | 3.0 to 3.6 | 3.0 to 3.6 | V |
| $f_{XT}$ = 32.768 kHz | C = 4.7 $\mu$F,<br>**Note** | 2.85 to 3.6 | 2.85 to 3.6 | 2.85 to 3.6 | 2.85 to 3.6 | V |

**Note**   When the system is operating on the subclock ($f_{XT}$ = 32.768 kHz), the A/D converter, D/A converter, and USB controller do not operate.

## 33.4  Oscillator Characteristics

### 33.4.1  Main clock oscillator characteristics

($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)

| Resonator | Circuit Example | Parameter | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Ceramic resonator/ crystal resonator | X1   X2 | Oscillation frequency ($f_X$)[Note 1] | | 3 | | 6 | MHz |
| | | Oscillation stabilization time[Note 2] | After reset is released | | $2^{16}/f_X$ | | s |
| | | | After STOP mode is released | | **Note 3** | | ms |
| | | | After IDLE2 mode is released | | **Note 3** | | $\mu$s |

Notes **1.** The oscillation frequency shown above indicates only oscillator characteristics. Use the V850ES/JG3-U and V850ES/JH3-U so that the internal operation conditions do not exceed the ratings shown in **AC Characteristics** and **DC Characteristics**.

**2.** Time required from start of oscillation until the resonator stabilizes.

**3.** The value varies depending on the setting of the OSTS register.

Cautions **1.** **When using the USB controller, be sure to use a ceramic resonator or crystal resonator with an accuracy of 6 MHz ±500 ppm or less when using the internal clock as the USB clock.**

**When using the external clock input by the UCLK pin, be sure to supply a clock with an accuracy of 48 MHz ±500 ppm or less.**
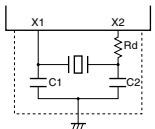
**If the USB clock accuracy drops, the transmission/reception data cannot satisfy the USB specification.**

**2.** **When using the main clock oscillator, wire as follows in the area enclosed by the broken lines in the above figure to avoid an adverse effect from wiring capacitance.**

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as $V_{SS}$.**
- **Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

**3.** **When the main clock is stopped and the device is operating on the subclock, wait until the oscillation stabilization time has been secured by the program before switching back to the main clock.**

**(1)  KYOCERA KINSEKI CORPORATION: Crystal resonator**
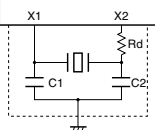
| Type | Circuit Example | Part Number | Oscillation Frequency fx (MHz) | Recommended Circuit Constant | | | Oscillation Voltage Range | | Oscillation Stabilization Time |
|---|---|---|---|---|---|---|---|---|---|
| | | | | C1 (pF) | C2 (pF) | Rd (Ω) | MIN. (V) | MAX. (V) | MAX. (ms) |
| Surface mounting |  | CX49GFWB04000D0PPTZ1 | 4.000 | 10 | 10 | 1000 | 2.85 | 3.6 | 14.86 |
| | | CX49GFWB05000D0PPTZ1 | 5.000 | 10 | 10 | 1000 | 2.85 | 3.6 | 13.98 |
| | | CX49GFWB06000D0PPTZ1 | 6.000 | 10 | 10 | 1000 | 2.85 | 3.6 | 12.8 |

**Caution   This oscillator constant is a reference value based on evaluation under a specific environment by the resonator manufacturer.**
**If optimization of oscillator characteristics is necessary in the actual application, apply to the resonator manufacturer for evaluation on the implementation circuit.**
**The oscillation voltage and oscillation frequency indicate only oscillator characteristics.  Use the V850ES/Jx3-U so that the internal operating conditions are within the specifications of the DC and AC characteristics.**

**(2)  KYOCERA CORPORATION:  Ceramic resonator**
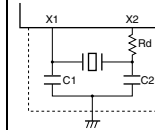
| Type | Circuit Example | Part Number | Oscillation Frequency fx (MHz) | Recommended Circuit Constant | | | Oscillation Voltage Range | | Oscillation stabilization time |
|---|---|---|---|---|---|---|---|---|---|
| | | | | C1 (pF) | C2 (pF) | Rd (Ω) | MIN. (V) | MAX. (V) | MAX. (ms) |
| Surface mounting |  | PBRC3.00HR10X000 | 3.000 | 30 | 30 | 1000 | 2.85 | 3.6 | 0.01 |
| | | PBRC4.00MR10X000 | 4.000 | 15 | 15 | 1000 | 2.85 | 3.6 | 0.01 |
| | | PBRC5.00MR10X000 | 5.000 | 15 | 15 | 1000 | 2.85 | 3.6 | 0.01 |
| | | PBRC6.00MR10X000 | 6.000 | 15 | 15 | 1000 | 2.85 | 3.6 | 0.01 |

**Caution   This oscillator constant is a reference value based on evaluation under a specific environment by the resonator manufacturer.**
**If optimization of oscillator characteristics is necessary in the actual application, apply to the resonator manufacturer for evaluation on the implementation circuit.**
**The oscillation voltage and oscillation frequency indicate only oscillator characteristics.  Use the V850ES/Jx3-U so that the internal operating conditions are within the specifications of the DC and AC characteristics.**

**(3)  Toyama Murata Mfg. Co. Ltd.:  Ceramic resonator**

| Type | Circuit Example | Part Number | Oscillation Frequency fx (MHz) | Recommended Circuit Constant | | | Oscillation Voltage Range | | Oscillation Stabilization Time |
|---|---|---|---|---|---|---|---|---|---|
| | | | | C1 (pF) | C2 (pF) | Rd (Ω) | MIN. (V) | MAX. (V) | MAX. (ms) |
| Surface mounting |  | CSTCR4M00GH5L99 | 4.000 | (39) | (39) | 1000 | 2.85 | 3.6 | 0.01 |
| | | CSTCR5M00GH5L99 | 5.000 | (39) | (39) | 1000 | 2.85 | 3.6 | 0.01 |
| | | CSTCR6M00GH5L99 | 6.000 | (39) | (39) | 680 | 2.85 | 3.6 | 0.01 |

**Caution   This oscillator constant is a reference value based on evaluation under a specific environment by the resonator manufacturer.**
**If optimization of oscillator characteristics is necessary in the actual application, apply to the resonator manufacturer for evaluation on the implementation circuit.**

The oscillation voltage and oscillation frequency indicate only oscillator characteristics. Use the V850ES/Jx3-U so that the internal operating conditions are within the specifications of the DC and AC characteristics.

**Remark** Figures in parentheses in columns C1 and C2 indicate the capacitance incorporated in the resonator.

### 33.4.2 Subclock oscillator characteristics

($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)

| Resonator | Circuit Example | Parameter | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Crystal resonator | XT1   XT2 | Oscillation frequency ($f_{XT}$)[Note 1] | | 32 | 32.768 | 35 | kHz |
| | | Oscillation stabilization time[Note 2] | | | | 10 | s |
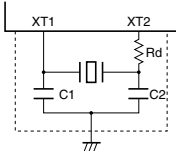
**Notes 1.** The oscillation frequency shown above indicates only oscillator characteristics. Use the V850ES/JG3-U and V850ES/JH3-U so that the internal operation conditions do not exceed the ratings shown in **AC Characteristics** and **DC Characteristics**.

**2.** Time required from when $V_{DD}$ reaches the oscillation voltage range (2.85 V (MIN.)) to when the crystal resonator stabilizes.

**Cautions 1.** When using the subclock oscillator, wire as follows in the area enclosed by the broken lines in the above figure to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as $V_{SS}$.**
- **Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

**2.** The subclock oscillator is designed as a low-amplitude circuit for reducing power consumption, and is more prone to malfunction due to noise than the main clock oscillator. Particular care is therefore required with the wiring method when the subclock is used.

**3.** For the resonator selection and oscillator constant, customers are requested to either evaluate the oscillation themselves or apply to the resonator manufacturer for evaluation.

**(1) Seiko Instruments Inc.: Crystal resonator**

   **Oscillation frequency: $f_{XT}$ = 32.768 kHz**

| Type | Circuit Example | Part Number | Recommended Circuit Constant | | | Oscillation Voltage Range | | Oscillation Stabilization Time |
|------|-----------------|-------------|---------|---------|--------|----------|----------|----------|
| | | | C1 (pF) | C2 (pF) | Rd (Ω) | MIN. (V) | MAX. (V) | MAX. (ms) |
| Surface mounting | XT1  XT2  Rd  C1  C2 | SSP-T7 | 7 | 7 | 0 | 2.85 | 3.6 | 1.4 |

**Caution   This oscillator constant is a reference value based on evaluation under a specific environment by the resonator manufacturer.**

**If optimization of oscillator characteristics is necessary in the actual application, apply to the resonator manufacturer for evaluation on the implementation circuit.**

**The oscillation voltage and oscillation frequency indicate only oscillator characteristics.  Use the V850ES/Jx3-U so that the internal operating conditions are within the specifications of the DC and AC characteristics.**

### 33.4.3  PLL characteristics

(T$_A$ = –40 to +85°C, V$_{DD}$ = EV$_{DD}$ = UV$_{DD}$ = AV$_{REF0}$ = AV$_{REF1}$, V$_{SS}$ = AV$_{SS}$ = 0 V, C$_L$ = 50 pF)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Input frequency | f$_X$ | | 3 | | 6 | MHz |
| Output frequency | f$_{XX}$ | Clock-through mode | 3 | | 6 | MHz |
| | | PLL mode (×8) | 24 | | 48 | MHz |
| Lock time | t$_{PLL}$ | | | | 800 | $\mu$s |

### 33.4.4  Internal oscillator characteristics

(T$_A$ = –40 to +85°C, V$_{DD}$ = EV$_{DD}$ = UV$_{DD}$ = AV$_{REF0}$ = AV$_{REF1}$, V$_{SS}$ = AV$_{SS}$ = 0 V, C$_L$ = 50 pF)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Output frequency | f$_R$ | | 100 | 220 | 400 | kHz |

## 33.5 DC Characteristics

### 33.5.1 I/O level

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Input voltage, high | $V_{IH1}$ | $\overline{RESET}$, FLMD0, P60 to P65, P90 to P915 | 0.8$EV_{DD}$ | | $EV_{DD}$ | V |
| | $V_{IH2}$ | P00 to P05, P20 to P25, P30 to P35, P42, P50 to P56 | 0.8$EV_{DD}$ | | 5.5 | V |
| | $V_{IH3}$ | P36, P37, P40, P41, PCM2, PCM3 | 0.7$EV_{DD}$ | | 5.5 | V |
| | $V_{IH4}$ | PDL0 to PDL15, PDH0 to PDH7, PCM0, PCM1, PCS0, PCS2, PCS3, PCT0, PCT1, PCT4, PCT6 | 0.7$EV_{DD}$ | | $EV_{DD}$ | V |
| | $V_{IH5}$ | UDPF, UDMF, UDPH, UDMH | 2.0 | | $UV_{DD}$ | V |
| | $V_{IH6}$ | P70 to P711 | 0.7$AV_{REF0}$ | | $AV_{REF0}$ | V |
| | $V_{IH7}$ | P10, P11 | 0.7$AV_{REF1}$ | | $AV_{REF1}$ | V |
| Input voltage, low | $V_{IL1}$ | $\overline{RESET}$, FLMD0, P60 to P65, P90 to P915 | $V_{SS}$ | | 0.2$EV_{DD}$ | V |
| | $V_{IL2}$ | P00 to P05, P20 to P25, P30 to P35, P42, P50 to P56 | $V_{SS}$ | | 0.2$EV_{DD}$ | V |
| | $V_{IL3}$ | P36, P37, P40, P41, PCM2, PCM3 | $V_{SS}$ | | 0.3$EV_{DD}$ | V |
| | $V_{IL4}$ | PDL0 to PDL15, PDH0 to PDH7, PCM0, PCM1, PCS0, PCS2, PCS3, PCT0, PCT1, PCT4, PCT6 | $V_{SS}$ | | 0.3$EV_{DD}$ | V |
| | $V_{IL5}$ | UDPF, UDMF, UDPH, UDMH | $V_{SS}$ | | 0.8 | V |
| | $V_{IL6}$ | P70 to P711 | $AV_{SS}$ | | 0.3$AV_{REF0}$ | V |
| | $V_{IL7}$ | P10, P11 | $AV_{SS}$ | | 0.3$AV_{REF1}$ | V |
| Input leakage current, high | $I_{LIH}$ | $V_I$ = $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$ | | | 5 | $\mu$A |
| Input leakage current, low | $I_{LIL}$ | $V_I$ = 0 V | | | –5 | $\mu$A |
| Output leakage current, high | $I_{LOH}$ | $V_O$ = $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$ | | | 5 | $\mu$A |
| Output leakage current, low | $I_{LOL}$ | $V_O$ = 0 V | | | –5 | $\mu$A |

**Remark**　Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

**($T_A$ = −40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output voltage, high | $V_{OH1}$ | **Note 1** | Per pin $I_{OH}$ = −1.0 mA | $EV_{DD}$ − 1.0 | | $EV_{DD}$ | V |
| | | | Per pin $I_{OH}$ = −100 $\mu$A | $EV_{DD}$ − 0.5 | | $EV_{DD}$ | V |
| | $V_{OH2}$ | P70 to P711 | Per pin $I_{OH}$ = −0.4 mA | $AV_{REF0}$ − 1.0 | | $AV_{REF0}$ | V |
| | | | Per pin $I_{OH}$ = −100 $\mu$A | $AV_{REF0}$ − 0.5 | | $AV_{REF0}$ | V |
| | $V_{OH3}$ | P10, P11 | Per pin $I_{OH}$ = −0.4 mA | $AV_{REF1}$ − 1.0 | | $AV_{REF1}$ | V |
| | | | Per pin $I_{OH}$ = −100 $\mu$A | $AV_{REF1}$ − 0.5 | | $AV_{REF1}$ | V |
| | $V_{OH4}$ | UDPF, UDMF, UDPH, UDMH | $P_L$ = 15 kΩ (Pull-up) | 2.8 | | | V |
| Output voltage, low | $V_{OL1}$ | **Note 2** | Per pin $I_{OL}$ = 1.0 mA | 0 | | 0.4 | V |
| | $V_{OL2}$ | P36, P37, P40, P41, P90, P91 | Per pin $I_{OL}$ = 3.0 mA | 0 | | 0.4 | V |
| | $V_{OL3}$ | P70 to P711 | Per pin $I_{OL}$ = 1.0 mA | 0 | | 0.4 | V |
| | $V_{OL4}$ | P10, P11 | Per pin $I_{OL}$ = 0.4 mA | 0 | | 0.4 | V |
| | $V_{OL5}$ | UDPF, UDMF, UDPH, UDMH | $P_L$ = 1.5 kΩ (Pull-down) | 0 | | 0.3 | V |
| Software pull-down resistor | $R_1$ | P56 | $V_I$ = $V_{DD}$ | 10 | 30 | 100 | kΩ |

**Notes 1.** P00 to P05, P20 to P25, P30 to P37, P40 to P42, P50 to P56, P60 to P65, P90 to P915, PCM0 to PCM3, PCS0, PCS2, PCS3

PCT0, PCT1, PCT4, PCT6, PDH0 to PDH7, PDL0 to PDL15

**2.** P00 to P05, P20 to P25, P30 to P35, P42, P50 to P56, P60 to P65, P92 to P915, PCM0 to PCM3, PCS0, PCS2, PCS3, PCT0, PCT1, PCT4, PCT6, PDH0 to PDH7, PDL0 to PDL15

**Remarks 1.** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

**2.** When the $I_{OH}$ and $I_{OL}$ conditions are not satisfied for one pin but the total value of all pins is satisfied, only that pin is deemed to not satisfy the DC characteristics.

### 33.5.2  Supply current

**($T_A$ = −40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| Supply current[Notes 1, 2] | $I_{DD1}$ | Normal operation | $f_{XX}$ = 48 MHz ($f_X$ = 6 MHz) Peripheral function operating | | | | 120 | mA |
| | | | $f_{XX}$ = 48 MHz ($f_X$ = 6 MHz) USBF operating | | | 54 | | mA |
| | $I_{DD2}$ | HALT mode | $f_{XX}$ = 48 MHz ($f_X$ = 6 MHz) Peripheral function operating | | | 42 | 60 | mA |
| | $I_{DD3}$ | IDLE1 mode | $f_{XX}$ = 48 MHz ($f_X$ = 6 MHz), PLL on | | | 4 | 7 | mA |
| | $I_{DD4}$ | IDLE2 mode | $f_{XX}$ = 6 MHz ($f_X$ = 6 MHz), PLL off | | | 0.5 | 1 | mA |
| | $I_{DD5}$ | Subclock operation mode | $f_{XT}$ = 32.768 kHz, main clock stopped, internal oscillator stopped | | | 120 | 600 | μA |
| | $I_{DD6}$ | Sub-IDLE mode | $f_{XT}$ = 32.768 kHz, main clock stopped, internal oscillator stopped | −40≤$T_A$≤+25°C | | 13 | 25 | μA |
| | | | | 25≤$T_A$≤85°C | | | 95 | μA |
| | $I_{DD7}$ | STOP mode | Subclock stopped, internal oscillator stopped | −40≤$T_A$≤+25°C | | 10 | 20 | μA |
| | | | | 25≤$T_A$≤85°C | | | 90 | μA |
| | | | Subclock operating, internal oscillator stopped | −40≤$T_A$≤+25°C | | 13 | 25 | μA |
| | | | | 25≤$T_A$≤85°C | | | 95 | μA |
| | $I_{DD8}$ | Flash memory programming mode | $f_{XX}$ = 48 MHz ($f_X$ = 6 MHz) | | | 65 | 130 | mA |

**Notes 1.** Total of $V_{DD}$, $EV_{DD}$, and $UV_{DD}$ currents.  Currents flowing through the output buffers, A/D converter, D/A converter, and on-chip pull-down resistor are not included.

**2.** The $V_{DD}$ of the TYP. value is 3.3 V.

## 33.6 Data Retention Characteristics

### (1) In STOP mode

($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Data retention voltage | $V_{DDDR}$ | STOP mode (all functions stopped) | 1.9 | | 3.6 | V |
| Data retention current | $I_{DDDR}$ | STOP mode (all functions stopped), $V_{DDDR}$ = 2.0 V | | 10 | 90 | $\mu$A |
| Supply voltage rise time | $t_{RVD}$ | | 200 | | | $\mu$s |
| Supply voltage fall time | $t_{FVD}$ | | 200 | | | $\mu$s |
| Supply voltage retention time | $t_{HVD}$ | After STOP mode setting | 0 | | | ms |
| STOP release signal input time | $t_{DREL}$ | After $V_{DD}$ reaches 2.85 V (MIN.) | 0 | | | ms |
| Data retention input voltage, high | $V_{IHDR}$ | $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $V_{DDDR}$ | $0.9V_{DDDR}$ | | $V_{DDDR}$ | V |
| Data retention input voltage, low | $V_{ILDR}$ | $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $V_{DDDR}$ | 0 | | $0.1V_{DDDR}$ | V |

**Caution  Shifting to STOP mode and restoring from STOP mode must be performed within the rated operating range.**
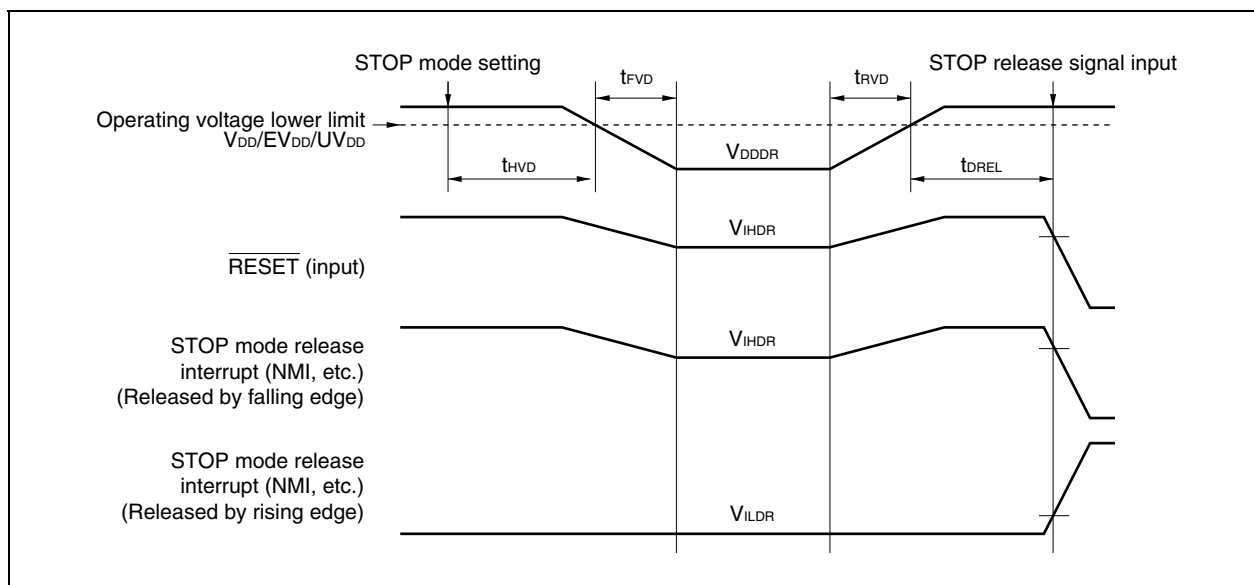
## 33.7  AC Characteristics

### (1)  AC test input measurement points ($V_{DD}$, $AV_{REF0}$, $AV_{REF1}$, $EV_{DD}$)
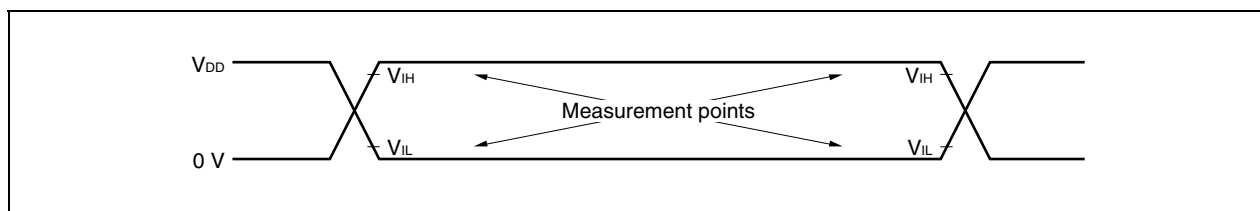


### (2)  AC test output measurement points



### (3)  Load conditions
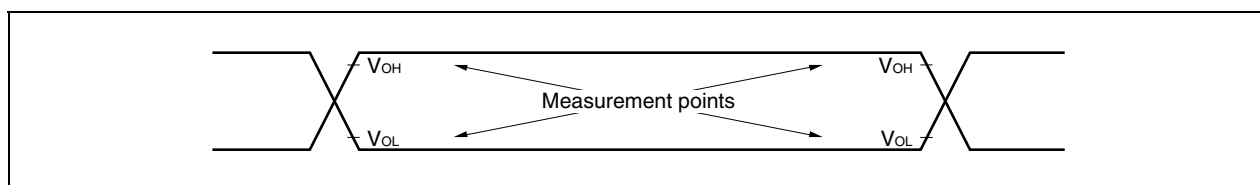


**Caution**     **If the load capacitance exceeds 50 pF due to the circuit configuration, bring the load capacitance of the device to 50 pF or less by inserting a buffer or by some other means.**
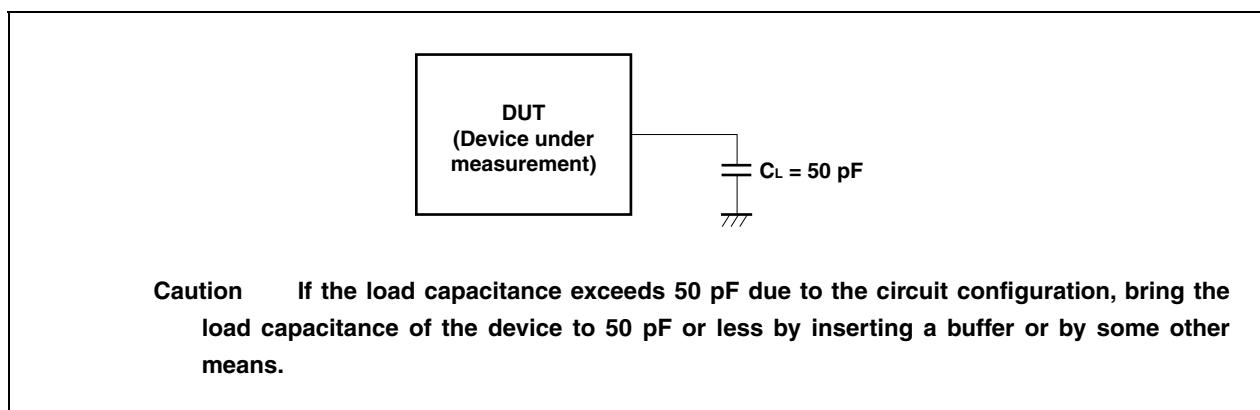
### 33.7.1  CLKOUT output timing

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Output cycle | $t_{CYK}$ | <1> | | 20.83 ns | 31.25 $\mu$s | |
| High-level width | $t_{WKH}$ | <2> | | $t_{CYK}/2 - 6$ | | ns |
| Low-level width | $t_{WKL}$ | <3> | | $t_{CYK}/2 - 6$ | | ns |
| Rise time | $t_{KR}$ | <4> | | | 6 | ns |
| Fall time | $t_{KF}$ | <5> | | | 6 | ns |

**Clock Timing**
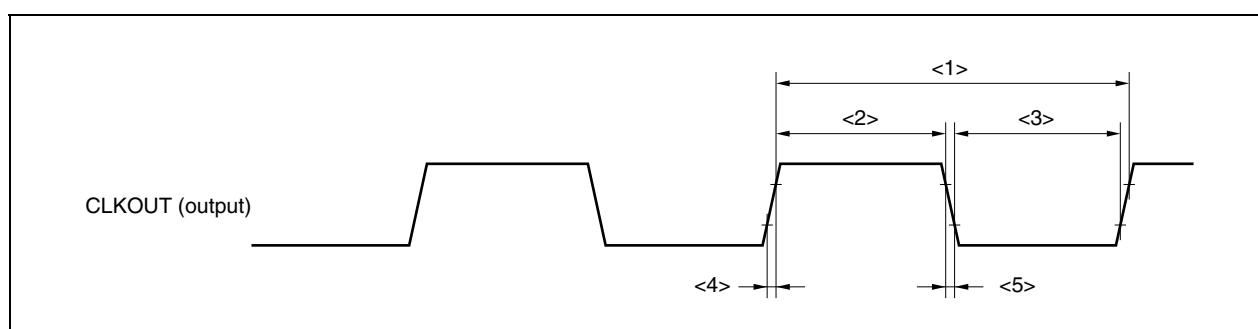
### 33.7.2  Bus timing

#### (1)  In multiplexed bus mode/separate bus mode

##### (a)  Read/write cycle (CLKOUT asynchronous)

($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)

| Parameter | Symbol | | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Address setup time (to ASTB↓) | $t_{DAST}$ | <6> | | $(0.5 + t_{ASw})T - 9$ | | ns |
| Address hold time (from ASTB↓) | $t_{HSTA}$ | <7> | | $(0.5 + t_{AHw})T - 8$ | | ns |
| Delay time from $\overline{RD}$↓ to address float | $t_{FRDA}$ | <8> | | | 5 | ns |
| Data input setup time from address | $t_{DAID}$ | <9> | | | $(2 + n + t_{ASw} + t_{AHw})T - 25$ | ns |
| Data input setup time from $\overline{RD}$↓ | $t_{DRDID2}$ | <10> | | | $(1 + n)T - 15$ | ns |
| Delay time from ASTB↓ to $\overline{RD}$↓ | $t_{DSTRD}$ | <11> | | $(0.5 + t_{AHw})T - 4$ | | ns |
| Delay time from ASTB↓ to $\overline{WRm}$↓ | $t_{DSTWR}$ | | | | | |
| Data input hold time (from $\overline{RD}$↑) | $t_{HRDID}$ | <12> | | 0 | | ns |
| Address output delay time from $\overline{RD}$↑ | $t_{DRDOD}$ | <13> | | $(1 + i)T - 3$ | | ns |
| Delay time from $\overline{RD}$↑ to ASTB↑ | $t_{DRDST}$ | <14> | | $0.5T - 5$ | | ns |
| Delay time from $\overline{WRm}$↑ to ASTB↑ | $t_{DWRST}$ | | | | | |
| Delay time from $\overline{RD}$↑ to ASTB↓ | $t_{DRDST}$ | <15> | | $(1.5 + i + t_{ASw})T - 4$ | | ns |
| $\overline{RD}$ low-level width | $t_{WRDL}$ | <16> | | $(1 + n)T - 10$ | | ns |
| $\overline{WRm}$ low-level width | $t_{WWRL}$ | | | | | |
| ASTB high-level width | $t_{WSTH}$ | <17> | | $(1 + i + t_{ASw})T - 10$ | | ns |
| Data output delay time from $\overline{WRm}$↓ | $t_{DWROD}$ | <18> | | | 9 | ns |
| Data output delay time (from $\overline{WRm}$↑) | $t_{DODWR}$ | <19> | | $(1 + n)T - 11$ | | ns |
| Data output hold time (from $\overline{WRm}$↑) | $t_{HWROD}$ | <20> | | $T - 3$ | | ns |
| $\overline{WAIT}$ setup time (to address) | $t_{SAWT1}$ | <21> | n ≥ 1 | | $(1.5 + t_{ASw} + t_{AHw})T - 25$ | ns |
| | $t_{SAWT2}$ | <22> | | | $(1.5 + n + t_{ASw} + t_{AHw})T - 25$ | ns |
| $\overline{WAIT}$ hold time (from address) | $t_{HAWT1}$ | <23> | n ≥ 1 | $(0.5 + n + t_{ASw} + t_{AHw})T$ | | ns |
| | $t_{HAWT2}$ | <24> | | $(1.5 + n + t_{ASw} + t_{AHw})T$ | | ns |
| $\overline{WAIT}$ setup time (to ASTB↓) | $t_{SSTWT1}$ | <25> | n ≥ 1 | | $(1 + t_{AHw})T - 15$ | ns |
| | $t_{SSTWT2}$ | <26> | | | $(1 + n + t_{AHw})T - 15$ | ns |
| $\overline{WAIT}$ hold time (from ASTB↓) | $t_{HSTWT1}$ | <27> | n ≥ 1 | $(n + t_{AHw})T - 2$ | | ns |
| | $t_{HSTWT2}$ | <28> | | $(1 + n + t_{AHw})T - 2$ | | ns |
| Address hold time from $\overline{RD}$↑ | $t_{HRDA2}$ | <29> | | $(1 + i)T - 5$ | | ns |
| Address hold time from $\overline{WRm}$↑ | $t_{HWRA2}$ | <30> | | $T - 5$ | | ns |
| Hold time from $\overline{RD}$↑ to $\overline{CSn}$ | $t_{HRDC2}$ | <31> | i ≥ 1 | $T - 5$ | | ns |
| Hold time from $\overline{WRm}$↑ to $\overline{CSn}$ | $t_{HWRC2}$ | <32> | | $T - 5$ | | ns |

**Remarks 1.** $t_{ASW}$: Number of address setup wait clocks

   $t_{AHW}$: Number of address hold wait clocks

**2.** $T = 1/f_{CPU}$ ($f_{CPU}$: CPU operating clock frequency)

**3.** n: Number of wait clocks inserted in the bus cycle

   The sampling timing changes when a programmable wait is inserted.

**4.** m = 0, 1

**5.** i: Number of idle states inserted after a read cycle (0 or 1)

**6.** The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

**Read Cycle (CLKOUT Asynchronous): In Multiplexed Bus Mode/Separate Bus Mode**
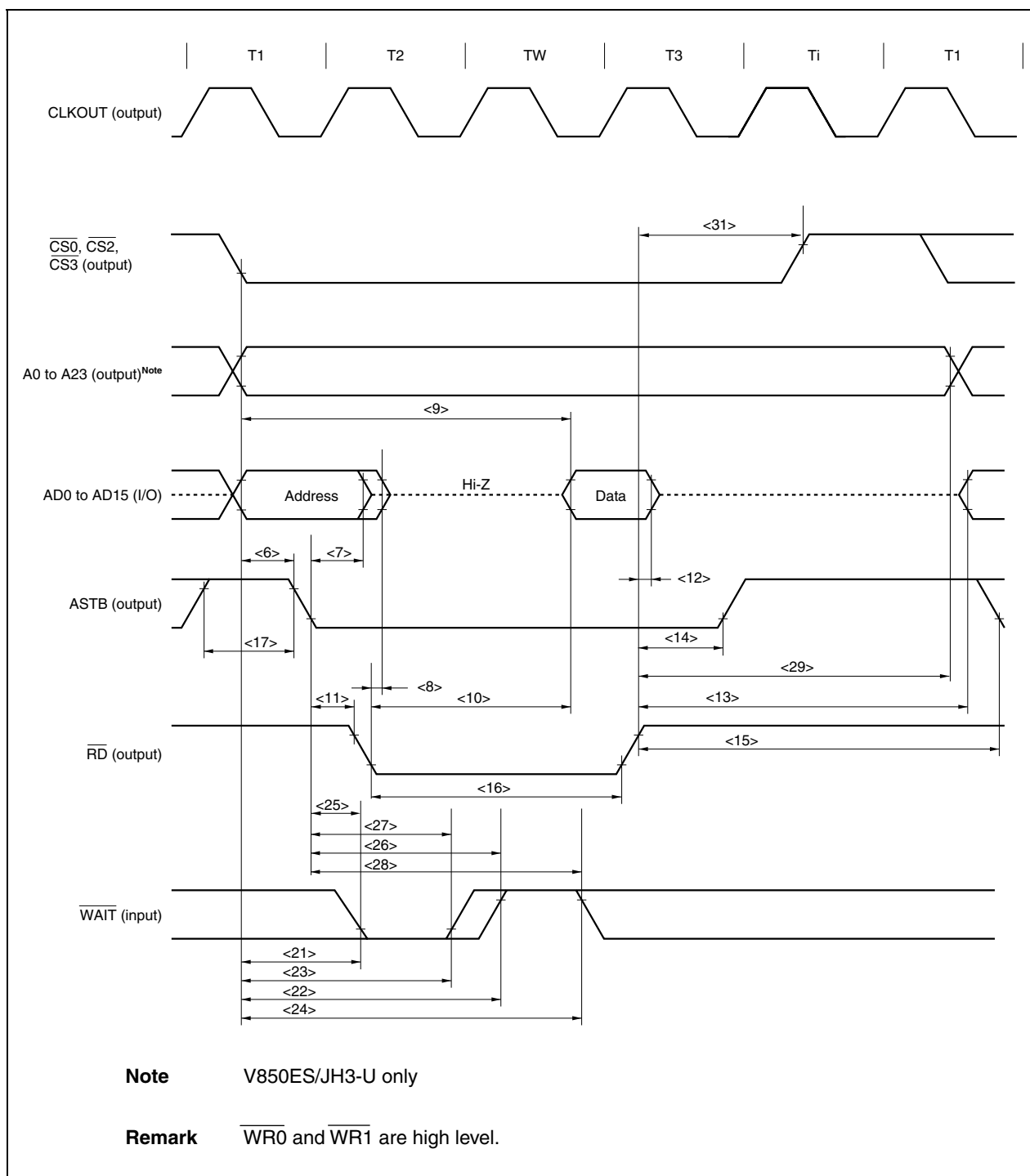


**Note** V850ES/JH3-U only

**Remark** WR0 and WR1 are high level.

**Write Cycle (CLKOUT Asynchronous): In Multiplexed Bus Mode/Separate Bus Mode**



**Note** V850ES/JH3-U only

**Remark** $\overline{\text{RD}}$ is high level.

**(b)  Read/write cycle (CLKOUT synchronous): In multiplexed bus mode/separate bus mode**

$(T_A = -40 \text{ to } +85°C, V_{DD} = EV_{DD} = UV_{DD} = AV_{REF0} = AV_{REF1}, V_{SS} = AV_{SS} = 0 \text{ V}, C_L = 50 \text{ pF})$

| Parameter | Symbol | | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Delay time from CLKOUT↑ to address | $t_{DKA}$ | <33> | | 0 | 17 | ns |
| Delay time from CLKOUT↑ to address float | $t_{FKA}$ | <34> | | 0 | 15 | ns |
| Delay time from CLKOUT↓ to ASTB | $t_{DKST}$ | <35> | | 0 | 12 | ns |
| Delay time from CLKOUT↑ to $\overline{RD}$, $\overline{WRm}$ | $t_{DKRDWR}$ | <36> | | 0 | 12 | ns |
| Data input setup time (to CLKOUT↑) | $t_{SIDK}$ | <37> | | 16 | | ns |
| Data input hold time (from CLKOUT↑) | $t_{HKID}$ | <38> | | 0 | | ns |
| Data output delay time from CLKOUT↑ | $t_{DKOD}$ | <39> | | | 17 | ns |
| $\overline{WAIT}$ setup time (to CLKOUT↓) | $t_{SWTK}$ | <40> | | 16 | | ns |
| $\overline{WAIT}$ hold time (from CLKOUT↓) | $t_{HKWT}$ | <41> | | 0 | | ns |
| Address hold time from CLKOUT↑ | $t_{HKA2}$ | <42> | | 0 | | ns |
| Data output hold time from CLKOUT↑ | $t_{HKOD2}$ | <43> | | 0 | | ns |

**Remarks 1.**   m = 0, 1
**2.**   The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

**Read Cycle (CLKOUT Synchronous): In Multiplexed Bus Mode/Separate Bus Mode**
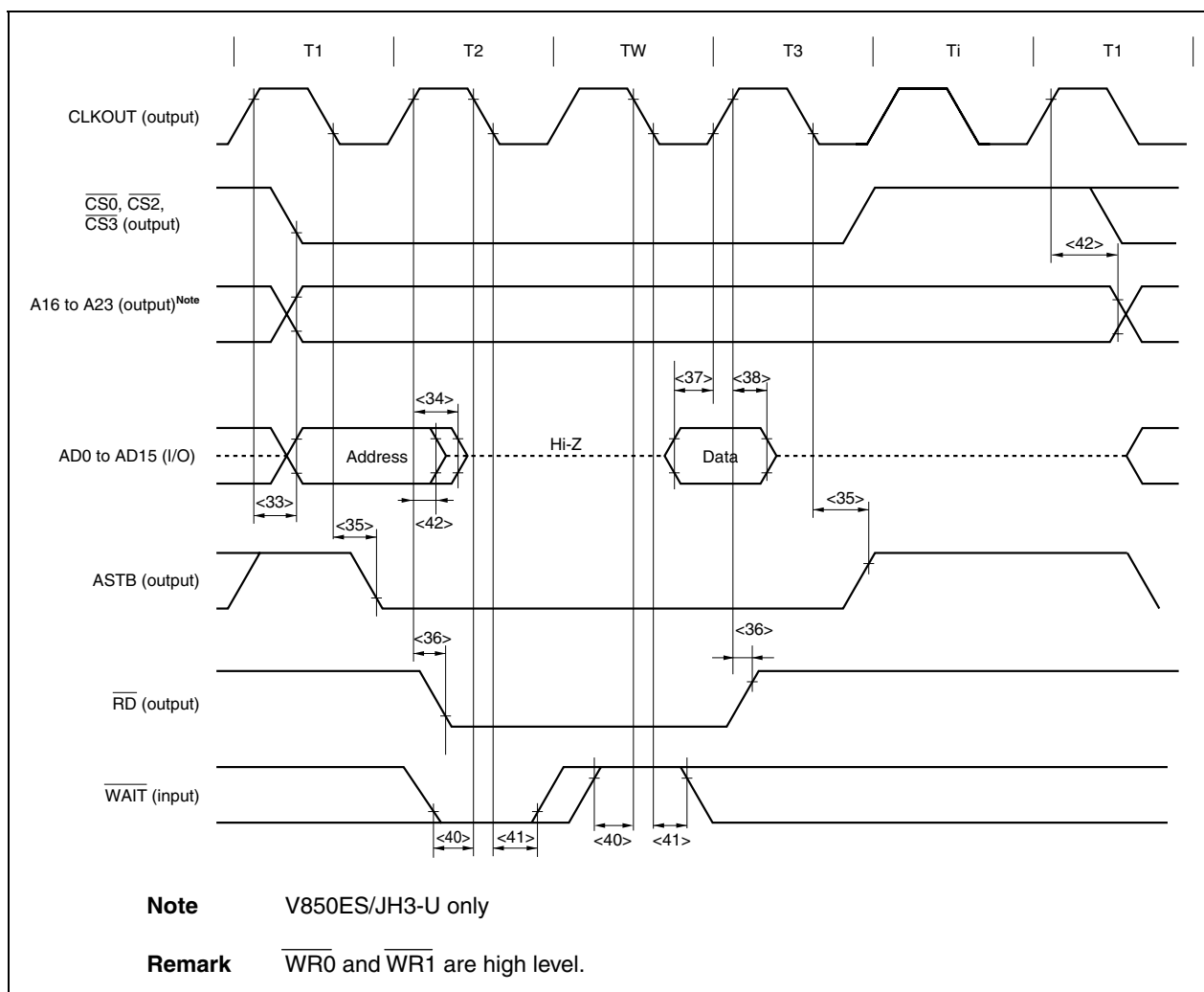


**Note**     V850ES/JH3-U only

**Remark**   $\overline{WR0}$ and $\overline{WR1}$ are high level.

**Write Cycle (CLKOUT Synchronous): In Multiplexed Bus Mode/Separate Bus Mode**



**Note**    V850ES/JH3-U only

**Remark**  $\overline{RD}$ is high level.
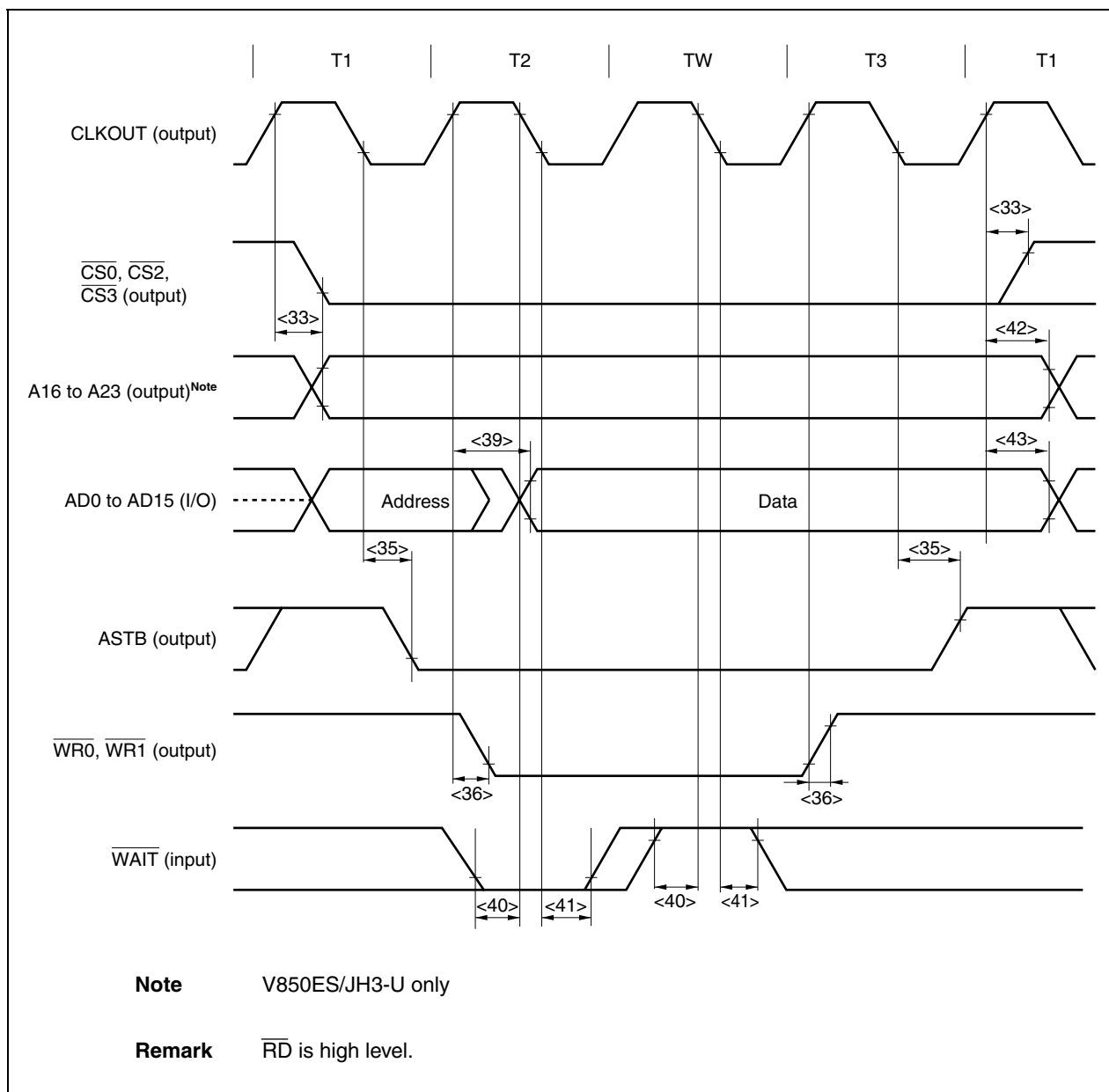
**(2) During bus hold (V850ES/JH3-U only)**

**(a) CLKOUT asynchronous**

($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)

| Parameter | Symbol | | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|---|
| $\overline{HLDRQ}$ high-level width | $t_{WHQH}$ | <44> | | T + 16 | | ns |
| $\overline{HLDAK}$ low-level width | $t_{WHAL}$ | <45> | | T – 10 | | ns |
| Delay time from $\overline{HLDAK}$↑ to bus output | $t_{DHAC}$ | <46> | | –7 | | ns |
| Delay time from $\overline{HLDRQ}$↓ to $\overline{HLDAK}$↓ | $t_{DHQHA1}$ | <47> | | 2.5T | | ns |
| Delay time from $\overline{HLDRQ}$↑ to $\overline{HLDAK}$↑ | $t_{DHQHA2}$ | <48> | | 0.5T + 17 | 1.5T + 31 | ns |
| Delay time from bus float to HLDAK↓ | $t_{DFHA}$ | <49> | | 0 | | ns |

**Remarks 1.** T = 1/$f_{CPU}$ ($f_{CPU}$: CPU operating clock frequency)

   **2.** n: Number of wait clocks inserted in the bus cycle

   The sampling timing changes when a programmable wait is inserted.

   **3.** The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

**Bus Hold (CLKOUT Asynchronous)**
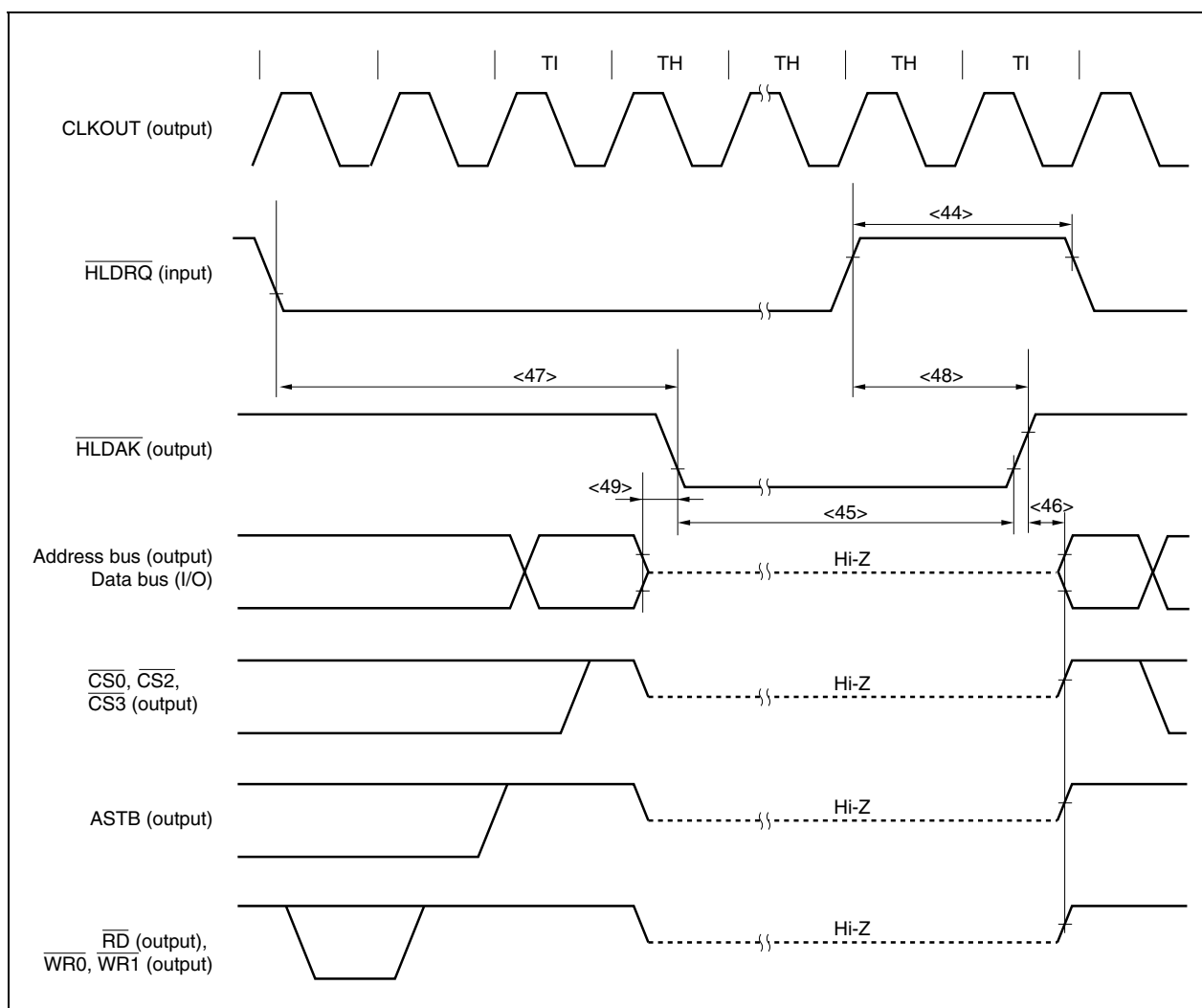
### (b)  CLKOUT synchronous

($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)
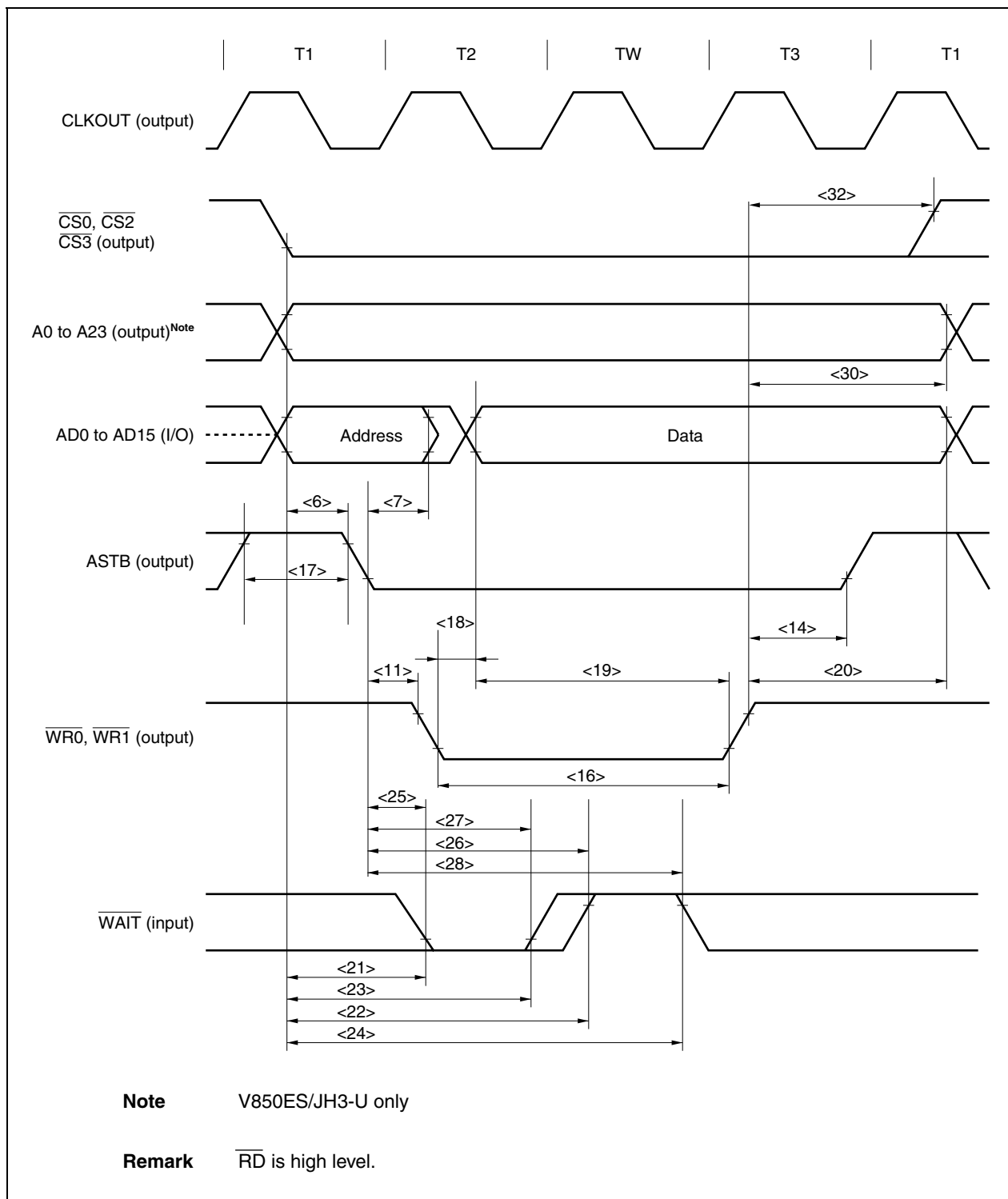
| Parameter | Symbol | | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|---|
| $\overline{HLDRQ}$ setup time (to CLKOUT↓) | $t_{SHQK}$ | <50> | | 16 | | ns |
| $\overline{HLDRQ}$ hold time (from CLKOUT↓) | $t_{HKHQ}$ | <51> | | 0 | | ns |
| Delay time from CLKOUT↑ to bus float | $t_{DKF}$ | <52> | | | 15 | ns |
| Delay time from CLKOUT↑ to $\overline{HLDAK}$↓ | $t_{DKHA1}$ | <53> | | 1 | 15 | ns |
| Delay time from CLKOUT↑ to $\overline{HLDAK}$↑ | $t_{DKHA2}$ | <54> | | 1 | 15 | ns |
| Delay time from CLKOUT↑ to data output | $t_{DKBO}$ | <55> | | 1 | 17 | ns |

**Remark**   The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

### Bus Hold (CLKOUT Synchronous)
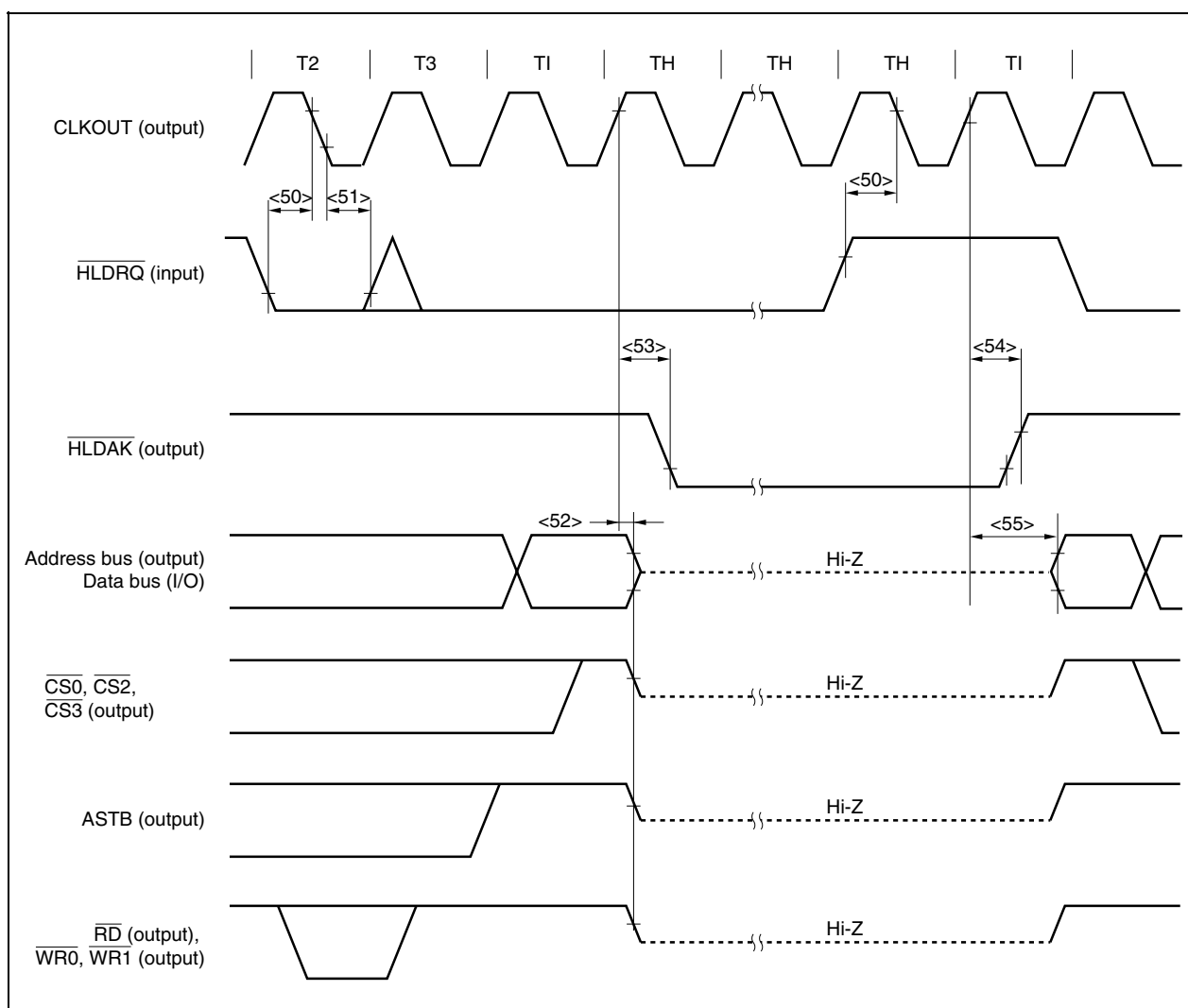
## 33.8  Basic Operation

### (1)  Power on/power off/reset timing

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Time from $EV_{DD}$, $UV_{DD}$↑ to $V_{DD}$↑ | $t_{REL}$ | <56> | | 0 | | ns |
| Time from $EV_{DD}$, $UV_{DD}$↑ to $AV_{REF0}$, $AV_{REF1}$↑ | $t_{REA}$ | <57> | | 0 | $t_{REL}$ | ns |
| Time from $V_{DD}$↑ to $\overline{RESET}$↑ | $t_{RER}$ | <58> | | 500 + $t_{REG}$**Note** | | ns |
| $\overline{RESET}$ low-level width | $t_{WRSL}$ | <59> | Analog noise elimination (during flash erase/writing) | 500 | | ns |
| | | | Analog noise elimination | 500 | | ns |
| Time from $\overline{RESET}$↓ to $V_{DD}$↓ | $t_{FRE}$ | <60> | | 500 | | ns |
| Time from $V_{DD}$↓ to $EV_{DD}$, $UV_{DD}$↓ | $t_{FEL}$ | <61> | | 0 | | ns |
| Time from $AV_{REF0}$, $AV_{REF1}$↓ to $EV_{DD}$, $UV_{DD}$↓ | $t_{FEA}$ | <62> | | 0 | $t_{FEL}$ | ns |

**Note**  Depends on the on-chip regulator characteristics.
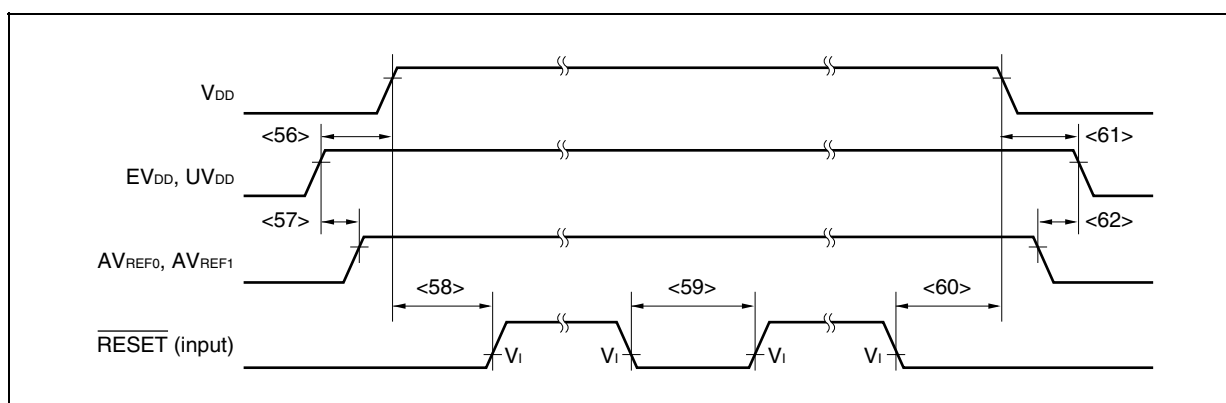
**(2)  Reset, interrupt, FLMD0 pin timing**

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|
| RESET input low-level width | $t_{WRSL}$ | | 500 | | ns |
| NMI high-level width | $t_{WNIH}$ | Analog noise elimination | 500 | | ns |
| NMI low-level width | $t_{WNIL}$ | Analog noise elimination | 500 | | ns |
| INTPn high-level width | $t_{WITH}$ | n = 0 to 18 (Analog noise elimination) | 500 | | ns |
| | | n = 2 (Digital noise elimination) | $3T_{SMP}$ + 20 | | ns |
| INTPn low-level width | $t_{WITL}$ | n = 0 to 18 (Analog noise elimination) | 500 | | ns |
| | | n = 2 (Digital noise elimination) | $3T_{SMP}$ + 20 | | ns |

**Remark**   $T_{SMP}$:  Set by the noise elimination control register (INTNFC).  Selectable from $f_{XX}/64$, $f_{XX}/128$, $f_{XX}/256$, $f_{XX}/512$, and $f_{XX}/1024$.

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|
| KRn high-level width | $t_{WKRH}$ | Analog noise elimination | 500 | | ns |
| KRn low-level width | $t_{WKRL}$ | Analog noise elimination | 500 | | ns |

**Remark**   n = 0 to 7

**(4) Timer timing**

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|
| TI high-level width | $t_{TIH}$ | TAB00 to TAB03,TAB10 to TAB13, EVTAB1, TRGAB1 | $12T + 20$ | | ns |
| | | TIAA00, TIAA01, TIAA10, TIAA11, TIAA20, TIAA21, TIAA30, TIAA31, TIAA50, TIAA51, | $3T_{SMP1} + 20$ | | ns |
| TI low-level width | $t_{TIL}$ | TAB00 to TAB03, TAB10 to TAB13, EVTAB1, TRGAB1 | $12T + 20$ | | ns |
| | | TIAA00, TIAA01, TIAA10, TIAA11, TIAA20, TIAA21, TIAA30, TIAA31, TIAA50, TIAA51, | $3T_{SMP1} + 20$ | | ns |
| TENCn high-level width | $t_{WENCHn}$ | n = 0, 1 | $3T_{SMP2} + 20$ | | ns |
| TENCn low-level width | $t_{WENCLn}$ | n = 0, 1 | $3T_{SMP2} + 20$ | | ns |
| TECR0 high-level width | $t_{WCRH0}$ | | $3T_{SMP2} + 20$ | | ns |
| TECR0 low-level width | $t_{WCRL0}$ | | $3T_{SMP2} + 20$ | | ns |
| TITn high-level width | $t_{WTITHn}$ | n = 0, 1 | $3T_{SMP2} + 20$ | | ns |
| TITn low-level width | $t_{WTITLn}$ | n = 0, 1 | $3T_{SMP2} + 20$ | | ns |
| EVTT0 high-level width | $t_{WTITH0}$ | | $3T_{SMP2} + 20$ | | ns |
| EVTT0 low-level width | $t_{WTITL0}$ | | $3T_{SMP2} + 20$ | | ns |
| TENCn input time difference | $t_{PHUD}$ | n = 0, 1 | $3T_{SMP2} + 20$ | | ns |

**Remarks 1.** $T = 1/f_{XX}$

    **2.** $T_{SMP1}$: Set by the noise elimination control register (TANFC). Selectable from $f_{XX}$ and $f_{XX}/4$.

    **3.** $T_{SMP2}$: Set by the noise elimination control register (TTNFC). Selectable from $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, and $f_{XX}/64$.

    **4.** The specifications above show the pulse widths that can be accurately detected as valid edges. Therefore, even if a pulse width less than the above specifications is input, it may be detected as a valid edge.
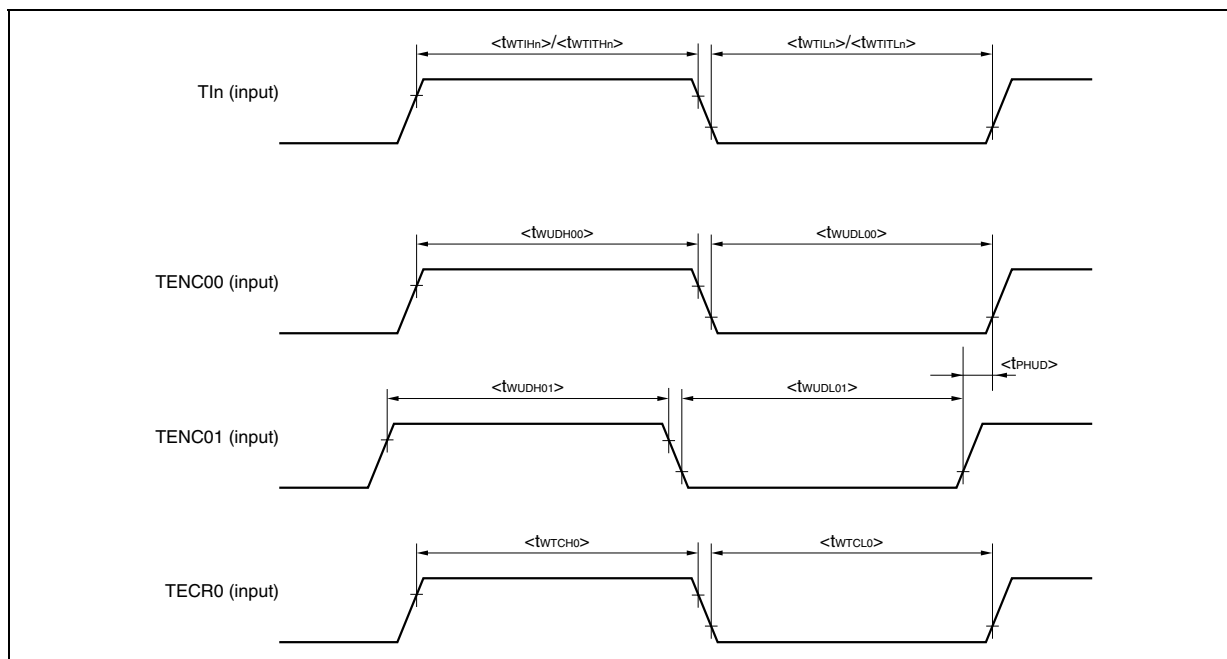
**(5) UARTC timing**

**($T_A$ = −40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | Conditions | MIN. | MAX. | Unit |
|-----------|--------|------------|------|------|------|
| Transmit rate | | | | 3.0 | Mbps |
| ASCK0 cycle time | | | | 10 | MHz |

**(6) CSIF timing**

**(a) Master mode**

**[When using CSI0 to CSIF2, or CSIF4]**

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|---|
| SCKFn cycle time | $t_{KCY1}$ | <63> | | 125 | | ns |
| SCKFn high-level width | $t_{KH1}$ | <64> | | $t_{KCY1}/2 - 8$ | | ns |
| SCKFn low-level width | $t_{KL1}$ | | | $t_{KCY1}/2 - 8$ | | ns |
| SIFn setup time (to SCKFn↑) | $t_{SIK1}$ | <65> | | 27 | | ns |
| SIFn setup time (to SCKFn↓) | | | | 27 | | ns |
| SIFn hold time (from SCKFn↑) | $t_{KSI1}$ | <66> | | 27 | | ns |
| SIFn hold time (from SCKFn↓) | | | | 27 | | ns |
| SOFn output delay time (from SCKFn↑) | $t_{KSO1}$ | <67> | | | 27 | ns |
| SOFn output delay time (from SCKFn↓) | | | | | 27 | ns |
| SOFn output hold time (from SCKFn↑) | $t_{HSO1}$ | <68> | | $t_{KCY1}/2 - 10$ | | ns |
| SOFn output hold time (from SCKFn↓) | | | | $t_{KCY1}/2 - 10$ | | ns |

**Remark** 0 to 2, 4

**[When using CSI3]**

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|---|
| SCKF3 cycle time | $t_{KCYM}$ | <63> | | 83.3 | | ns |
| SCKF3 high-level width | $t_{KHM}$ | <64> | | $t_{KCYM}/2 - 8$ | | ns |
| SCKF3 low-level width | | | | $t_{KCYM}/2 - 8$ | | ns |
| SIF3 setup time (to SCKF3↑) | $t_{SIKM}$ | <65> | | 16 | | ns |
| SIF3 setup time (to SCKF3↓) | | | | 16 | | ns |
| SIF3 hold time (from SCKF3↑) | $t_{KSIM}$ | <66> | | 16 | | ns |
| SIF3 hold time (from SCKF3↓) | | | | 16 | | ns |
| SOF3 output delay time (from SCKF3↑) | $t_{KSOM}$ | <67> | | | 16 | ns |
| SOF3 output delay time (from SCKF3↓) | | | | | 16 | ns |
| SOF3 output hold time (from SCKF3↑) | $t_{HSOM}$ | <68> | | $t_{KCYM}/2 - 10$ | | ns |
| SOF3 output hold time (from SCKF3↓) | | | | $t_{KCYM}/2 - 10$ | | ns |

**(b) Slave mode**

**[When using CSI0 to CSIF2, or CSIF4]**

$(T_A = -40$ to $+85°C, V_{DD} = EV_{DD} = UV_{DD} = AV_{REF0} = AV_{REF1}, V_{SS} = AV_{SS} = 0$ V, $C_L = 50$ pF$)$

| Parameter | Symbol | | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|---|
| $\overline{SCKFn}$ cycle time | $t_{KCY2}$ | <63> | | 125 | | ns |
| $\overline{SCKFn}$ high-level width | $t_{KH2}$ | <64> | | $t_{KCYn}/2 - 8$ | | ns |
| $\overline{SCKFn}$ low-level width | $t_{KL2}$ | | | $t_{KCYn}/2 - 8$ | | ns |
| SIFn setup time (to $\overline{SCKFn}$↑) | $t_{SIK2}$ | <65> | | 27 | | ns |
| SIFn setup time (from $\overline{SCKFn}$↓) | | | | 27 | | ns |
| SIFn hold time (to $\overline{SCKFn}$↑) | $t_{KSI2}$ | <66> | | 27 | | ns |
| SIFn hold time (from $\overline{SCKFn}$↓) | | | | 27 | | ns |
| SOFn output delay time (to $\overline{SCKFn}$↑) | $t_{KSO2}$ | <67> | | | 27 | ns |
| SOFn output delay time (from $\overline{SCKFn}$↓) | | | | | 27 | ns |
| SOFn output delay time (to $\overline{SCKFn}$↑) | $t_{HSO2}$ | <68> | | $t_{KCYn}/2 - 10$ | | ns |
| SOFn output delay time (from $\overline{SCKFn}$↓) | | | | $t_{KCYn}/2 - 10$ | | ns |

**Remark** n = 0 to 4

**(a) CFnCTL1.CFnCKP, CFnDAP bits = 00 or 11**

SCKFn (I/O)

SIFn (input)    Input data

SOFn (output)    Output data
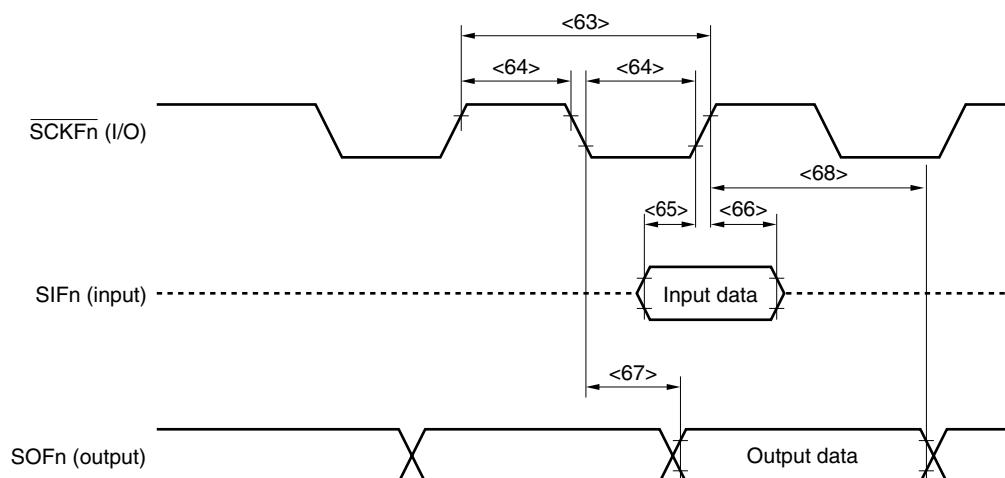
**(b) CFnCTL1.CFnCKP, CFnDAP bits = 10 or 01**

SCKFn (I/O)

SIFn (input)    Input data

SOFn (output)    Output data
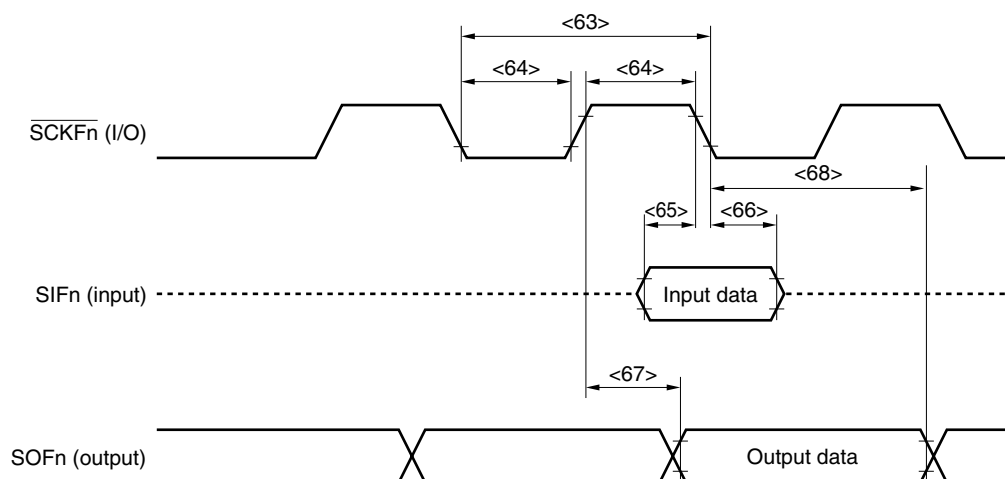
**Remark**  n = 0 to 4

**(7)  I²C bus mode**

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V)**

| Parameter | | Symbol | | Normal Mode | | High-Speed Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCL0n clock frequency | | $f_{CLK}$ | | 0 | 100 | 0 | 400 | kHz |
| Bus free time (Between start and stop conditions) | | $t_{BUF}$ | <69> | 4.7 | – | 1.3 | – | $\mu$s |
| Hold time[Note 1] | | $t_{HD:STA}$ | <70> | 4.0 | – | 0.6 | – | $\mu$s |
| SCL0n clock low-level width | | $t_{LOW}$ | <71> | 4.7 | – | 1.3 | – | $\mu$s |
| SCL0n clock high-level width | | $t_{HIGH}$ | <72> | 4.0 | – | 0.6 | – | $\mu$s |
| Setup time for start/restart conditions | | $t_{SU:STA}$ | <73> | 4.7 | – | 0.6 | – | $\mu$s |
| Data hold time | CBUS compatible master | $t_{HD:DAT}$ | <74> | 5.0 | – | – | – | $\mu$s |
| | I²C mode | | | 0[Note 2] | – | 0[Note 2] | 0.9[Note 3] | $\mu$s |
| Data setup time | | $t_{SU:DAT}$ | <75> | 250 | – | 100[Note 4] | – | ns |
| SDA0n and SCL0n signal rise time | | $t_R$ | <76> | – | 1000 | 20 + 0.1Cb[Note 5] | 300 | ns |
| SDA0n and SCL0n signal fall time | | $t_F$ | <77> | – | 300 | 20 + 0.1Cb[Note 5] | 300 | ns |
| Stop condition setup time | | $t_{SU:STO}$ | <78> | 4.0 | – | 0.6 | – | $\mu$s |
| Pulse width of spike suppressed by input filter | | $t_{SP}$ | <79> | – | – | 0 | 50 | ns |
| Capacitive load of each bus line | | Cb | | – | 400 | – | 400 | pF |

**Notes 1.** When the start condition is satisfied, the first clock pulse is generated after the hold time.

   **2.** The system requires a minimum of 300 ns hold time internally for the SDA0n signal (at $V_{IHmin.}$ of the SCL0n signal) in order to occupy the undefined area at the falling edge of SCL0n.

   **3.** If the system does not extend the SCL0n signal low hold time ($t_{LOW}$), only the maximum data hold time ($t_{HD:DAT}$) needs to be satisfied.

   **4.** The high-speed mode I²C bus can be used in a normal-mode I²C bus system. In this case, set the high-speed mode I²C bus so that it meets the following conditions.

   - If the system does not extend the SCL0n signal low hold time:
     $t_{SU:DAT} \geq$ 250 ns

   - If the system extends the SCL0n signal low hold time:
     Output the next data bit to the SDA0n line before the SCL0n line is released ($t_{Rmax.}$ + $t_{SU:DAT}$ = 1,000 + 250 = 1,250 ns: Normal mode I²C bus specification).

   **5.** Cb: Total capacitance of one bus line (unit: pF)

**Remark**    n = 0 to 2

**Remark**    n = 0 to 2
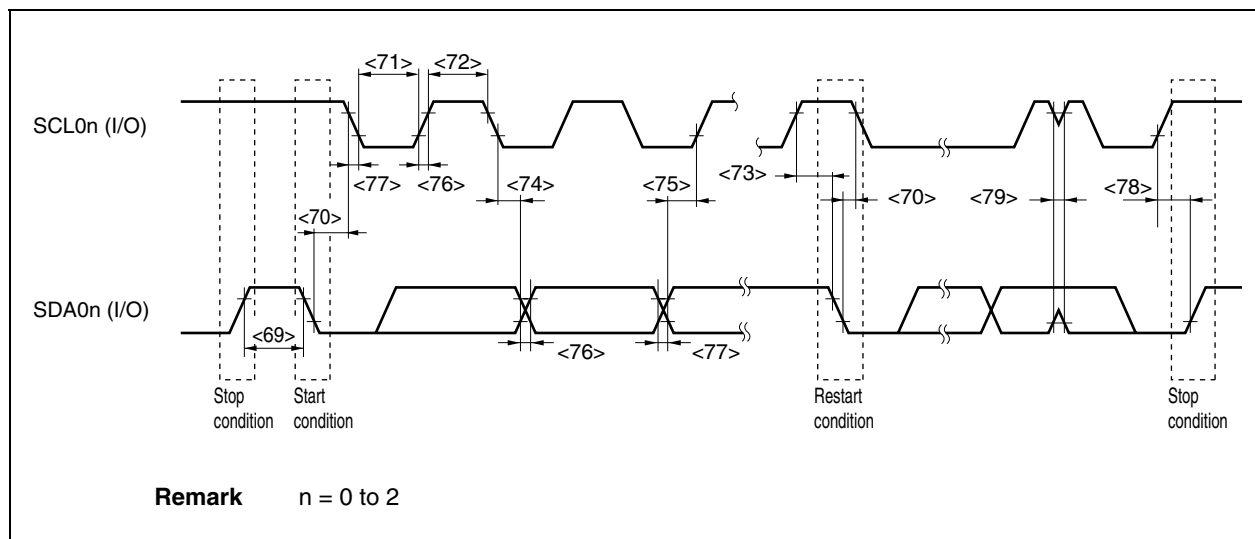
**(8) High-impedance control timing**

($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)

| Parameter | Symbol | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|
| Time from oscillator stop to timer output high impedance | $t_{CLM}$ | Clock monitor operating | | 65 | $\mu$s |
| Time from TOAB1OFF input → timer output high impedance | $t_{HTQn}$ | | | 300 | ns |
| Time from TOAA1OFF input → timer output high impedance | $t_{HTP2}$ | | | 300 | ns |

**(9) A/D converter**

($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, 3.0 V ≤ $AV_{REF0}$ ≤ 3.6 V, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Resolution | | | | | 10 | bit |
| Overall error[Note] | | 3.0 ≤ $AV_{REF0}$ ≤ 3.6 V | | | ±0.6 | %FSR |
| Conversion time | $t_{CONV}$ | | 2.17 | | 24 | $\mu$s |
| Zero scale error | | | | | ±0.5 | %FSR |
| Full scale error | | | | | ±0.5 | %FSR |
| Non-linearity error | | | | | ±4.0 | LSB |
| Differential linearity error | | | | | ±4.0 | LSB |
| Analog input voltage | $V_{IAN}$ | | $AV_{SS}$ | | $AV_{REF0}$ | V |
| Reference voltage | $AV_{REF0}$ | | 3.0 | | 3.6 | V |
| $AV_{REF0}$ current | $AI_{REF0}$ | Normal conversion mode | | 3 | 6.5 | mA |
| | | High-speed conversion mode | | 4 | 10 | mA |
| | | When A/D converter unused | | | 5 | $\mu$A |

**Note** Excluding quantization error (±0.05 %FSR).

**Caution Do not set (read/write) alternate-function ports during A/D conversion; otherwise the conversion resolution may be degraded.**

**Remark** LSB: Least Significant Bit
FSR: Full Scale Range

### (10)  D/A converter

**(T$_A$ = –40 to +85°C, V$_{DD}$ = EV$_{DD}$ = UV$_{DD}$ = AV$_{REF0}$ = AV$_{REF1}$, 3.0 V ≤ AV$_{REF1}$ ≤ 3.6 V, V$_{SS}$ = AV$_{SS}$ = 0 V, C$_L$ = 50 pF)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Resolution | | | | | 8 | bit |
| Overall error[Note 1] | | R = 2 MΩ | | | ±1.2 | %FSR |
| Settling time | | C = 20 pF | | | 3 | $\mu$s |
| Output resistor | R$_O$ | Output data 55H | | 6.42 | | kΩ |
| Reference voltage | AV$_{REF1}$ | | 3.0 | | 3.6 | V |
| AV$_{REF1}$ current[Note 2] | AI$_{REF1}$ | D/A conversion operating | | 1 | 2.5 | mA |
| | | D/A conversion stopped | | | 5 | $\mu$A |

**Notes 1.** Excluding quantization error (±0.5 %LSB).

　　　**2.** Value of 1 channel of D/A converter

**Remark**　R is the output pin load resistance and C is the output pin load capacitance.

### (11)  LVI circuit characteristics

**(T$_A$ = –40 to +85°C, V$_{DD}$ = EV$_{DD}$ = UV$_{DD}$ = AV$_{REF0}$ = AV$_{REF1}$, V$_{SS}$ = AV$_{SS}$ = 0 V, C$_L$ = 50 pF)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Detection voltage | V$_{LVI0}$ | | 2.85 | 2.95 | 3.05 | V |
| Response time[Note] | t$_{LD}$ | After V$_{DD}$ reaches V$_{LVI0}$ (MAX.), or after V$_{DD}$ has dropped to V$_{LVI0}$ (MAX.) | | 0.2 | 2.0 | ms |
| Minimum pulse width | t$_{LW}$ | | 0.2 | | | ms |
| Reference voltage stabilization wait time | t$_{LWAIT}$ | After V$_{DD}$ reaches 2.85 V(MIN.) | | 0.1 | 0.2 | ms |

**Note**　Time required to detect the detection voltage and output an interrupt or reset signal.
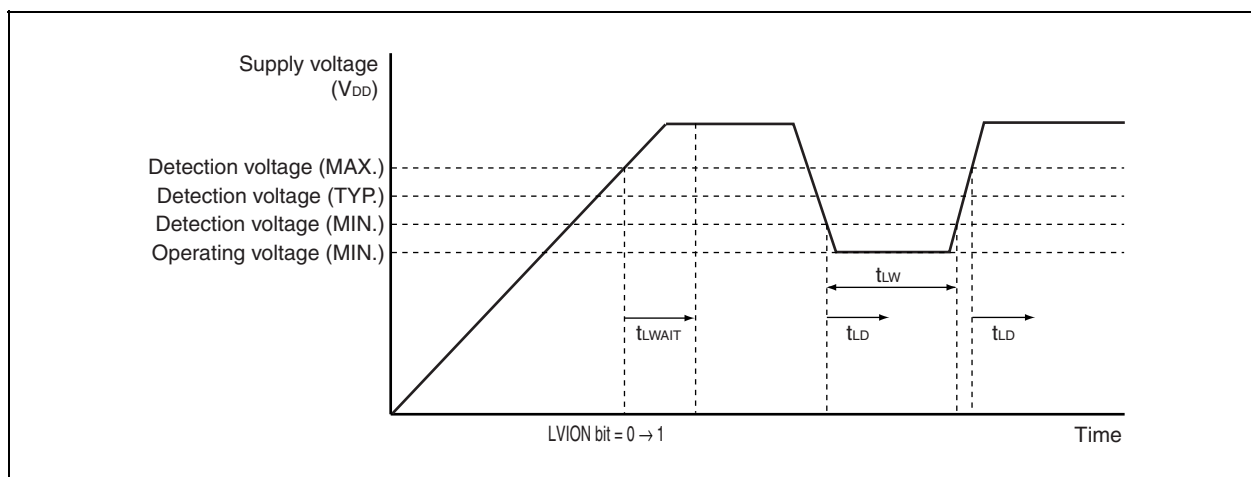
**(12) RAM retention detection**

**($T_A$ = –40 to +85°C, $V_{DD}$ = $EV_{DD}$ = $UV_{DD}$ = $AV_{REF0}$ = $AV_{REF1}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $C_L$ = 50 pF)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Detection voltage | $V_{RAMH}$ | | 1.9 | 2.0 | 2.1 | V |
| Supply voltage rise time | $t_{RAMHTH}$ | $V_{DD}$ = 0 to 2.85 V | 0.002 | | | ms |
| Response time[Note] | $t_{RAMHD}$ | After $V_{DD}$ reaches 2.1 V | | 0.2 | 3.0 | ms |
| Minimum pulse width | $t_{RAMHW}$ | | 0.2 | | | ms |

**Note** Time required to detect the detection voltage and set the RAMS.RAMF bit.
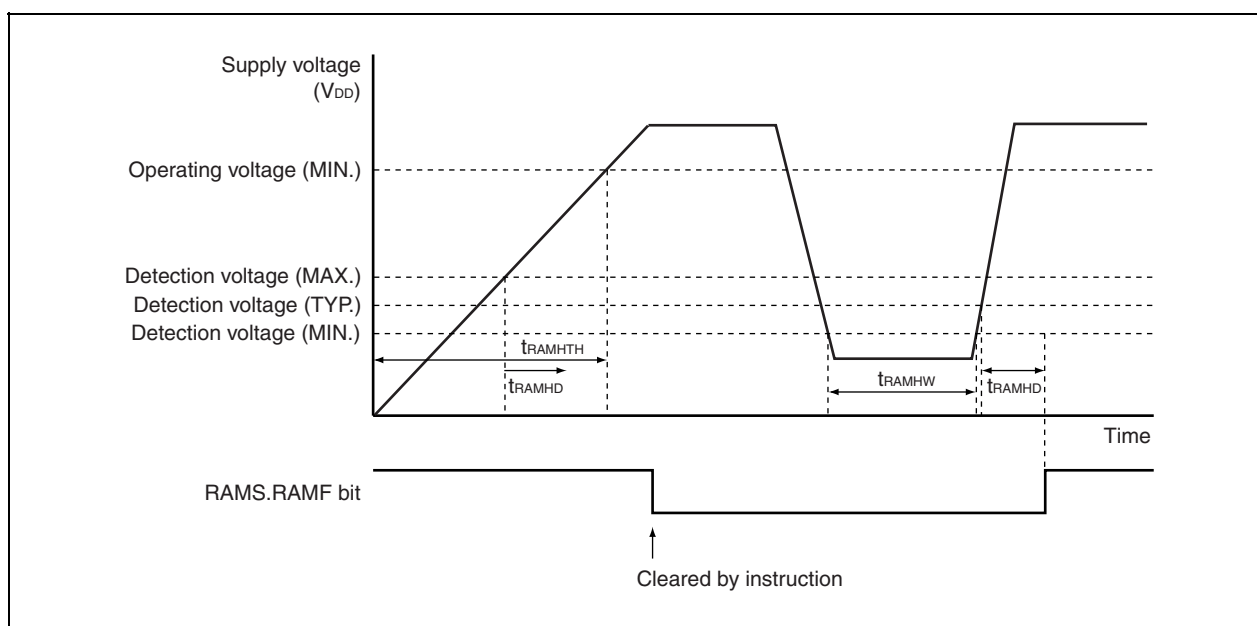
## 33.9  Flash Memory Programming Characteristics

$(T_A = -40$ to $+85°C$, $V_{DD} = EV_{DD} = UV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = AV_{SS} = 0$ V, $C_L = 50$ pF)

### (1)  Basic characteristics

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Operating frequency | $f_{CPU}$ | | | 24 | | 48 | MHz |
| Supply voltage | $V_{DD}$ | | | 2.85 | | 3.6 | V |
| Number of rewrites | $C_{WRT}$ | Used for updating programs When using flash memory programmer and Renesas Electronics self programming library | Retained for 15 years | 1,000 | | | times |
| | | Used for updating data When using Renesas Electronics EEPROM emulation library (usable ROM size: 12 KB of 3 consecutive blocks) | Retained for 5 years | 10,000 | | | times |
| Programming temperature | $t_{PRG}$ | | | −40 | | +85 | °C |

### (2)  Serial write operation characteristics

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| FLMD0, FLMD1 setup time | $t_{MDSET}$ | | 2 | | 3000 | ms |
| FLMD0 count start time from $\overline{RESET}\uparrow$ | $t_{RFCF}$ | $f_X = 3$ to 6 MHz | 800 | | | $\mu$s |
| FLMD0 counter high-level width/ low-level width | $t_{CH}/t_{CL}$ | | 10 | | 100 | $\mu$s |
| FLMD0 counter rise time/fall time | $t_R/t_F$ | | | | 1 | $\mu$s |

**Flash write mode setup timing**
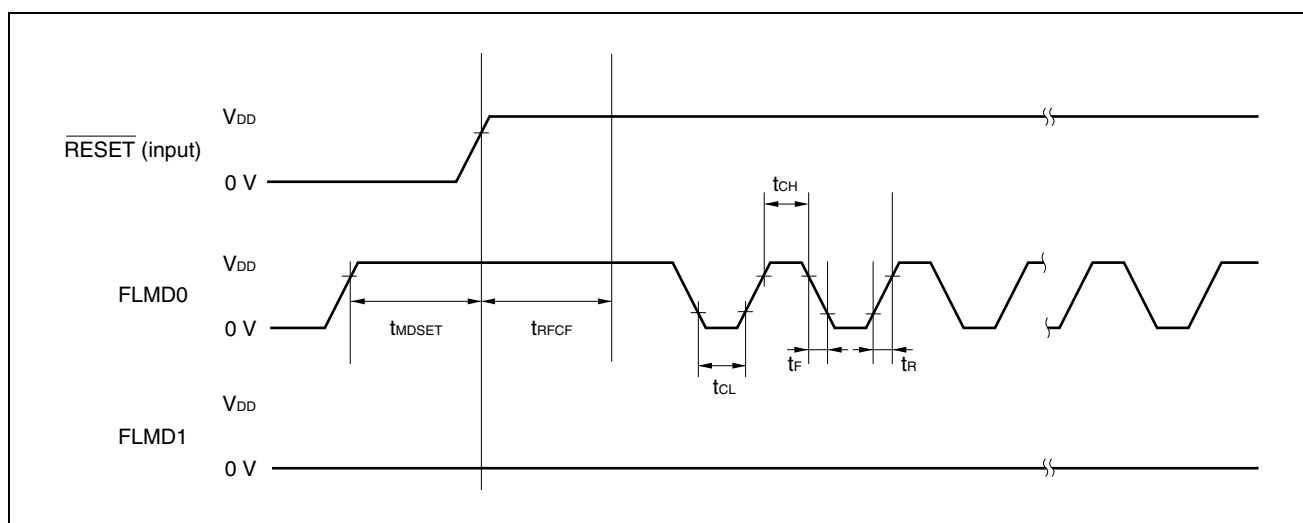
**(3)  Programming characteristics**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Chip erase time | | $f_{XX}$ = 48 MHz, batch erasure | | 105 | | ms |
| Write time per 256 bytes | | $f_{XX}$ = 48 MHz | | 2.0 | | ms |
| Block internal verify time | | $f_{XX}$ = 48 MHz | | 10 | | ms |
| Block blank check time | | $f_{XX}$ = 48 MHz | | 0.5 | | ms |
| Flash memory information setting time | | $f_{XX}$ = 48 MHz | | 30 | | ms |

**Caution**  **When writing initially to shipped products, it is counted as one rewrite for both "erase to write" and "write only".**

        **Example (P: Write, E: Erase)**

                **Shipped product  $\longrightarrow$ P $\rightarrow$ E $\rightarrow$ P $\rightarrow$ E $\rightarrow$ P:  3 rewrites**

                **Shipped product  $\rightarrow$ E $\rightarrow$ P $\rightarrow$ E $\rightarrow$ P $\rightarrow$ E $\rightarrow$ P:  3 rewrites**

**Remark**  The block size is 4 KB.

## CHAPTER 34  PACKAGE DRAWINGS

### 100-PIN PLASTIC LQFP (FINE PITCH) (14x14)

detail of lead end

| ITEM | DIMENSIONS |
|------|------------|
| | (UNIT:mm) |
| D | 14.00±0.20 |
| E | 14.00±0.20 |
| HD | 16.00±0.20 |
| HE | 16.00±0.20 |
| A | 1.60 MAX. |
| A1 | 0.10±0.05 |
| A2 | 1.40±0.05 |
| A3 | 0.25 |
| b | $0.20^{+0.07}_{-0.03}$ |
| c | $0.125^{+0.075}_{-0.025}$ |
| L | 0.50 |
| Lp | 0.60±0.15 |
| L1 | 1.00±0.20 |
| θ | $3°^{+5°}_{-3°}$ |
| e | 0.50 |
| x | 0.08 |
| y | 0.08 |
| ZD | 1.00 |
| ZE | 1.00 |

**P100GC-50-UEU-1**
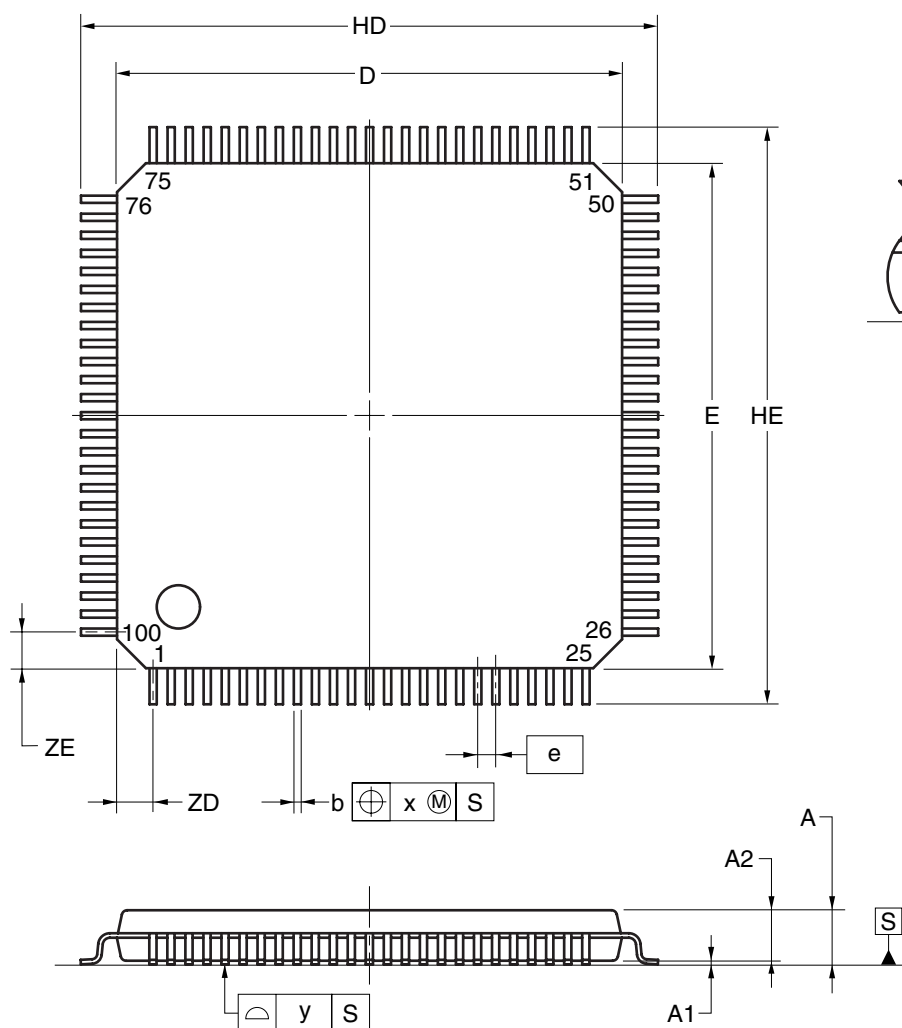
© NEC Electronics Corporation 2007

## 128-PIN PLASTIC LQFP (FINE PITCH) (14x20)

detail of lead end

NOTE
Each lead centerline is located within 0.08 mm of
its true position at maximum material condition.
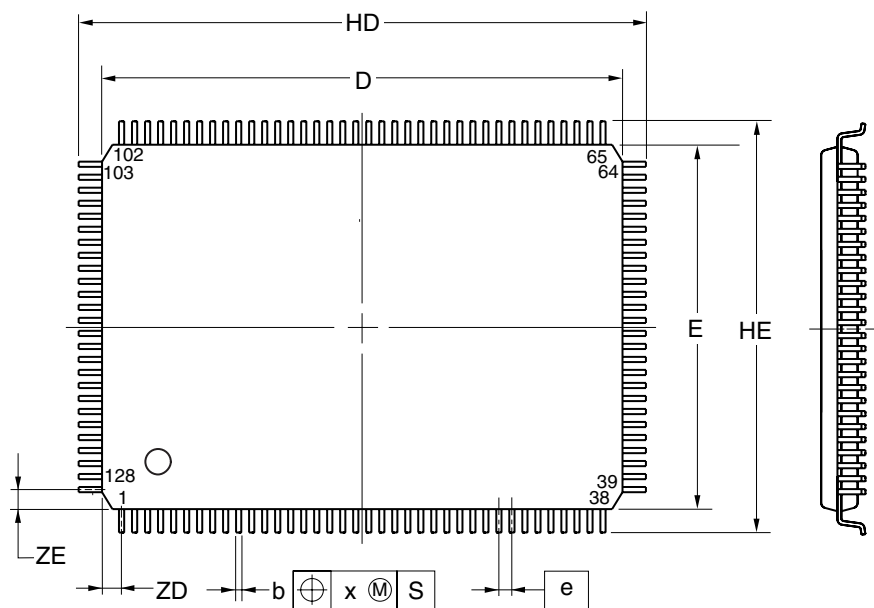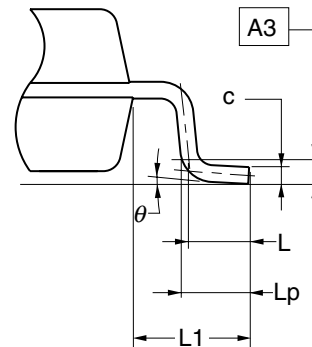
(UNIT:mm)

| ITEM | DIMENSIONS |
|------|-----------|
| D | 20.00±0.20 |
| E | 14.00±0.20 |
| HD | 22.00±0.20 |
| HE | 16.00±0.20 |
| A | 1.60 MAX. |
| A1 | 0.10±0.05 |
| A2 | 1.40±0.05 |
| A3 | 0.25 |
| b | $0.20^{+0.07}_{-0.03}$ |
| c | $0.125^{+0.075}_{-0.025}$ |
| L | 0.50 |
| Lp | 0.60±0.15 |
| L1 | 1.00±0.20 |
| θ | $3°^{+5°}_{-3°}$ |
| e | 0.50 |
| x | 0.08 |
| y | 0.08 |
| ZD | 0.75 |
| ZE | 0.75 |

**P128GF-50-GAT**

© NEC Electronics Corporation 2006

# CHAPTER 35  RECOMMENDED SOLDERING CONDITIONS

These products should be soldered and mounted under the following recommended conditions.

For soldering methods and conditions other than those recommended below, please contact an Renesas Electronics sales representative.

For technical information, see the following website.

Semiconductor Device Mount Manual (http://www2.renesas.com/pkg/en/mount/index.html)

**Remark** Evaluation of the soldering conditions for the (A) standard products is incomplete because these products are under development.

**Table 35-1.  Surface Mounting Type Soldering Conditions (1/2)**

$\mu$PD70F3763GC-UEU-AX: 100-pin plastic LQFP (fine pitch) (14 $\times$ 14 mm)

$\mu$PD70F3764GC-UEU-AX: 100-pin plastic LQFP (fine pitch) (14 $\times$ 14 mm)

| Soldering Method | Soldering Conditions | Recommended Condition Symbol |
|---|---|---|
| Infrared reflow | Package peak temperature: 260°C, Time: 60 seconds max. (at 220°C or higher), Count: 3 times or less, Exposure limit: 7 days[Note] (after that, prebake at 125°C for 20 to 72 hours) | IR60-207-3 |
| Partial heating | Pin temperature: 350°C max., Time: 3 seconds max. (per pin row) | – |

**Note** After opening the dry pack, store it at 25°C or less and 65% RH or less for the allowable storage period.

**Caution   Do not use different soldering methods together (except for partial heating).**

**Remarks 1.** Products with –AX at the end of the part number are lead-free products.
       **2.** For soldering methods and conditions other than those recommended, please contact an Renesas Electronics sales representative.

**Table 35-1.  Surface Mounting Type Soldering Conditions (2/2)**

$\mu$PD70F3768GF-GAT-AX: 128-pin plastic LQFP (fine pitch) (14 $\times$ 20 mm)

$\mu$PD70F3769GF-GAT-AX: 128-pin plastic LQFP (fine pitch) (14 $\times$ 20 mm)

| Soldering Method | Soldering Conditions | Recommended Condition Symbol |
|---|---|---|
| Infrared reflow | Package peak temperature: 260°C, Time: 60 seconds max. (at 220°C or higher), Count: 3 times or less, Exposure limit: 7 days[Note] (after that, prebake at 125°C for 20 to 72 hours) | IR60-207-3 |
| Partial heating | Pin temperature: 350°C max., Time: 3 seconds max. (per pin row) | − |

**Note**    After opening the dry pack, store it at 25°C or less and 65% RH or less for the allowable storage period.

**Caution    Do not use different soldering methods together (except for partial heating).**

**Remarks 1.**    Products with −AX at the end of the part number are lead-free products.

**2.**    For soldering methods and conditions other than those recommended, please contact an Renesas Electronics sales representative.

## APPENDIX  A  DEVELOPMENT  TOOLS

The following development tools are available for the development of systems that employ the V850ES/JG3-U or V850ES/JH3-U.

Figure A-1 shows the development tool configuration.

- **Support for PC98-NX series**

  Unless otherwise specified, products supported by IBM PC/AT™ compatibles are compatible with PC98-NX series computers.  When using PC98-NX series computers, refer to the explanation for IBM PC/AT compatibles.

- **Windows®**

  Unless otherwise specified, "Windows" means the following OSs.
  - Windows 98, 2000
  - Windows Me
  - Windows XP
  - Windows NT® Ver. 4.0

**Figure A-1.  Development Tool Configuration**



**Notes 1.**  Project manager PM+ is included in the C compiler package.

PM+ is only used in Windows.

   **2.**  The QB-V850MINI, QB-MINI2, and QB-V850ESJX3H support the USB interface only.

   **3.**  The QB-V850MINI is supplied with the ID850QB, USB interface cable, OCD cable, self-check board, KEL adapter, and KEL connector.  All other products are optional.

   **4.**  The QB-MINI2 is supplied with USB interface cable, 16-pin target cable, 10-pin target cable, and 78K0-OCD board (integrated debugger is not supplied.)  All other products are optional.

   **5.**  The QB-V850ESJX3H is supplied with the ID850QB, flash memory programmer (MINICUBE2), power supply unit, and USB interface adapter.  All other products are optional.

## A.1  Software Package

| SP850<br>Software package for V850<br>microcontrollers | Development tools (software) commonly used with V850 microcontrollers are included this package. |
|---|---|
| | Part number:  $\mu$S$\times\times\times\times$SP850 |

**Remark**   $\times\times\times\times$ in the part number differs depending on the host machine and OS used.

$\mu$S$\times\times\times\times$SP850

| $\times\times\times\times$ | Host Machine | OS | Supply Medium |
|---|---|---|---|
| AB17 | PC-9800 series,<br>IBM PC/AT compatibles | Windows (Japanese version) | CD-ROM |
| BB17 | | Windows (English version) | |

## A.2  Language Processing Software

| CA850<br>C compiler package | This compiler converts programs written in C into object codes executable with a microcontroller.  This compiler is started from project manager PM+. |
|---|---|
| | Part number:  $\mu$S$\times\times\times\times$CA703000 |
| DF703771<br>Device file | This file contains information peculiar to the device.<br>This device file should be used in combination with a tool (CA850 or ID850QB).<br>The corresponding OS and host machine differ depending on the tool to be used. |

**Remark**   $\times\times\times\times$ in the part number differs depending on the host machine and OS used.

$\mu$S$\times\times\times\times$CA703000

| $\times\times\times\times$ | Host Machine | OS | Supply Medium |
|---|---|---|---|
| AB17 | PC-9800 series,<br>IBM PC/AT compatibles | Windows (Japanese version) | CD-ROM |
| BB17 | | Windows (English version) | |
| 3K17 | SPARCstation$^{TM}$ | SunOS$^{TM}$ (Rel. 4.1.4),<br>Solaris$^{TM}$ (Rel. 2.5.1) | |

## A.3  Control Software

| PM+<br>Project manager | This is control software designed to enable efficient user program development in the Windows environment.  All operations used in development of a user program, such as starting the editor, building, and starting the debugger, can be performed from PM+.<br>**<Caution>**<br>PM+ is included in C compiler package CA850.<br>It can only be used in Windows. |
|---|---|

## A.4  Debugging Tools (Hardware)

### A.4.1  When using IECUBE QB-V850ESJX3H

The system configuration when connecting the QB-V850ESJX3H to the host machine (PC-9821 series, PC/AT compatible) is shown below.  Even if optional products are not prepared, connection is possible.

**Figure A-2.  System Configuration (When Using QB-V850ESJX3H) (1/2)**
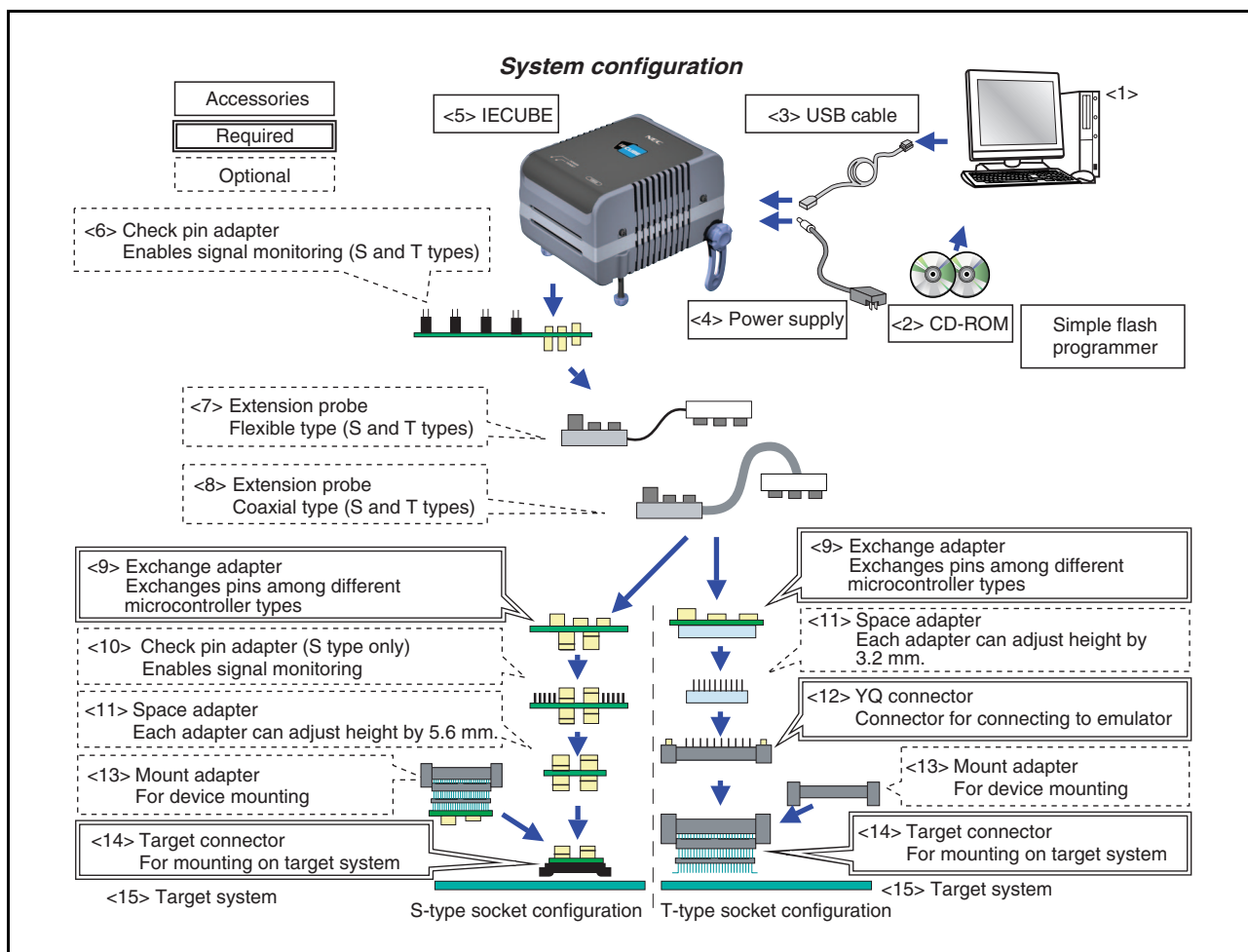
**Figure A-2.   System Configuration (When Using QB-V850ESJX3H) (2/2)**

<1>  Host machine (PC-9821 series, IBM-PC/AT compatibles)

<2>  Debugger, USB driver, manuals, etc. (ID850QB Disk, Accessory Disk[Note 1])

<3>  USB interface cable

<4>  AC adapter

<5>  In-circuit emulator (QB-V850ESJX3H)

<6>  Check pin adapter (S and T types) (QB-144-CA-01) (optional)

<7>  Extension probe (flexible type) (S and T types) (QB-144-EP-02S) (optional)

<8>  Extension probe (coaxial type) (S and T types) (QB-144-EP-01S) (optional)

<9>  Exchange adapter[Note 2] (S type: QB-100GC-EA-04S (GC package), QB-128GF-EA-01S (GF package), T type: QB-100GC-EA-05T (GC package), QB-128GF-EA-02T (GF package))

<10> Check pin adapter[Note 3] (S type only: QB-100-CA-01S (GC package), QB-128GF-CA-01S (GF package)) (optional)

<11> Space adapter[Note 3] (S type: QB-100-SA-01S (GC package), QB-144-SA-01S (GF package), T type: QB-100GC-YS-01T (GC package), QB-128GF-YS-01T (GF package) (optional)

<12> YQ connector[Note 2] (T type only) (QB-100GC-YQ-01T) (GC package), QB-128GF-YQ-01T (GF package)

<13> Mount adapter (S type: QB-100GC-MA-01S (GC package), QB-128GF-MA-01S (GF package), T type: QB-100GC-HQ-01T (GC package), QB-128GF-HQ-01T (GF package)) (optional)

<14> Target connector[Note 2] (S type: QB-100GC-TC-01S (GC package), QB-128GF-TC-01S (GF package), T type: QB-100GC-NQ-01T (GC package), QB-128GF-NQ-01T (GF package))

<15> Target system

**Notes 1.**  Download the device file from the Renesas Electronics website.

   http://www2.renesas.com/micro/en/ods/

**2.**  Supplied with the device depending on the ordering number.

- When QB-V850ESJX3H-ZZZ is ordered

   The exchange adapter and the target connector are not supplied.

- When QB-V850ESJX3H-S100GC is ordered

   The QB-100GC-EA-04S and QB-100GC-TC-01S are supplied.

- When QB-V850ESJX3H-S128GF is ordered

   The QB-128GF-EA-01S and QB-128GF-TC-01S are supplied.

- When QB-V850ESJX3H-T100GC is ordered

   The QB-100GC-EA-05T, QB-100GC-YQ-01T, and QB-100GC-NQ-01T are supplied.

- When QB-V850ESJX3H-T128GF is ordered

   The QB-1028GF-EA-02T, QB-128GF-YQ-01T, and QB-128GF-NQ-01T are supplied.

**3.**  When using both <9> and <10>, the order between <9> and <10> is not cared.

| | |
|---|---|
| <5> QB-V850ESJX3H<sup>Note</sup><br>In-circuit emulator | The in-circuit emulator serves to debug hardware and software when developing application systems using the V850ES/JG3-U or V850ES/JH3-U. It supports the integrated debugger ID850QB. This emulator should be used in combination with a power supply unit and emulation probe. Use the USB interface cable to connect this emulator to the host machine. |
| <3> USB interface cable | Cable to connect the host machine and the QB-V850ESJX3H. |
| <4> AC adapter | 100 to 240 V can be supported by replacing the AC plug. |
| <9> QB-100GC-EA-04S<br>QB-128GF-EA-01S<br>QB-100GC-EA-05T<br>QB-128GF-EA-02T<br>Exchange adapter | Adapter to perform pin conversion.<br>• QB-100GC-EA-04S: 100-pin plastic LQFP (GC-UEU type)<br>• QB-128GF-EA-01S: 128-pin plastic LQFP (GF-GAT type)<br>• QB-100GC-EA-05T: 100-pin plastic LQFP (GC-UEU type)<br>• QB-128GF-EA-02T: 128-pin plastic LQFP (GC-GAT type) |
| <10> QB-100-CA-01S<br>QB-128-CA-01S<br>(S type only)<br>Check pin adapter | Adapter used in waveform monitoring using the oscilloscope, etc.<br>• QB-100-CA-01S: 100-pin plastic LQFP (GC-UEU type)<br>• QB-128-CA-01S: 128-pin plastic LQFP (GF-GAT type) |
| <11> QB-100-SA-01S<br>QB-144-SA-01S<br>QB-100GC-YS-01T<br>QB-128GF-YS-01T<br>Space adapter | Adapter to adjust the height.<br>• QB-100-SA-01S: 100-pin plastic LQFP (GC-UEU type)<br>• QB-144-SA-01S: 128-pin plastic LQFP (GF-GAT type)<br>• QB-100GC-YS-01T: 100-pin plastic LQFP (GC-UEU type)<br>• QB-128GF-YS-01T: 128-pin plastic LQFP (GF-GAT type) |
| <12> QB-100GC-YQ-01T<br>QB-128GF-YQ-01T<br>(T type only)<br>YQ connector | Conversion adapter to connect target connector and exchange adapter<br>• QB-100GC-YQ-01T: 100-pin plastic LQFP (GC-UEU type)<br>• QB-128GF-YQ-01T: 128-pin plastic LQFP (GF-GAT type) |
| <13> QB-100GC-MA-01S<br>QB-128GF-MA-01S<br>QB-100GC-HQ-01T<br>QB-128GF-HQ-01T<br>Mount adapter | Adapter to mount the V850ES/JG3-U or V850ES/JH3-U on a socket.<br>• QB-100GC-MA-01S: 100-pin plastic LQFP (GC-UEU type)<br>• QB-128GF-MA-01S: 128-pin plastic LQFP (GF-GAT type)<br>• QB-100GC-HQ-01T: 100-pin plastic LQFP (GC-UEU type)<br>• QB-128GF-HQ-01T: 128-pin plastic LQFP (GF-GAT type) |
| <14> QB-100GC-TC-01S<br>QB-128GF-TC-01S<br>QB-100GC-NQ-01T<br>QB-128GF-NQ-01T<br>Target connector | Connector to solder on the target system.<br>• QB-100GC-TC-01S: 100-pin plastic LQFP (GC-UEU type)<br>• QB-128GF-TC-01S: 128-pin plastic LQFP (GF-GAT type)<br>• QB-100GC-NQ-01T: 100-pin plastic LQFP (GC-UEU type)<br>• QB-128GF-NQ-01T: 128-pin plastic LQFP (GF-GAT type) |

**Note** The QB-V850ESJX3H is supplied with a power supply unit, USB interface cable, and flash memory programmer (MINICUBE2). It is also supplied with integrated debugger ID850QB as control software.

**Remark** The numbers in the angle brackets correspond to the numbers in Figure A-2.

### A.4.2   When using MINICUBE QB-V850MINI

#### (1)  On-chip emulation using MINICUBE

The system configuration when connecting MINICUBE to the host machine (PC-9821 series, PC/AT compatible) is shown below.

**Figure A-3.  On-Chip Emulation System Configuration**



| <1> | Host machine | PC with USB ports |
|---|---|---|
| <2> | CD-ROM[Note 1] | Contents such as integrated debugger ID850QB, N-Wire Checker, device driver, and documents are included in CD-ROM.  It is supplied with MINICUBE. |
| <3> | USB interface cable | USB cable to connect the host machine and MINICUBE.  It is supplied with MINICUBE.  The cable length is approximately 2 m. |
| <4> | MINICUBE<br>On-chip debug emulator | This on-chip debug emulator serves to debug hardware and software when developing application systems using the V850ES/JG3-U or V850ES/JH3-U.  It supports integrated debugger ID850QB. |
| <5> | OCD cable | Cable to connect MINICUBE and the target system.<br>It is supplied with MINICUBE.  The cable length is approximately 20 cm. |
| <6> | Connector conversion board<br>KEL adapter | This conversion board is supplied with MINICUBE. |
| <7> | MINICUBE connector<br>KEL connector[Note 2] | 8830E-026-170S (supplied with MINICUBE)<br>8830E-026-170L (sold separately) |

**Notes  1.** Download the device file from the Renesas Electronics website.

http://www2.renesas.com/micro/en/ods/index.html

   **2.** Product of KEL Corporation

**Remark**   The numbers in the angular brackets correspond to the numbers in Figure A-3.

### A.4.3 When using MINICUBE2 QB-MINI2

The system configuration when connecting MINICUBE2 to the host machine (PC-9821 series, PC/AT compatible) is shown below.

**Figure A-4. System Configuration of On-Chip Emulation System**



| <1> Host machine | PC with USB ports |
|---|---|
| <2> Software | The integrated debugger ID850QB, device file, etc.<br>Download the device file from the Renesas Electronics website.<br>  http://www2.renesas.com/micro/en/ods/index.html |
| <3> USB interface cable | USB cable to connect the host machine and MINICUBE. It is supplied with MINICUBE. The cable length is approximately 2 m. |
| <4> MINICUBE2<br>On-chip debug emulator | This on-chip debug emulator serves to debug hardware and software when developing application systems using the V850ES/JG3-U or V850ES/JH3-U. It supports integrated debugger ID850QB. |
| <5> 16-pin target cable | Cable to connect MINICUBE2 and the target system.<br>It is supplied with MINICUBE. The cable length is approximately 15 cm. |
| <6> Target connector (sold separately) | Use a 16-pin general-purpose connector with 2.54 mm pitch. |

**Remark** The numbers in the angular brackets correspond to the numbers in Figure A-4.

## A.5  Debugging Tools (Software)

| ID850QB<br>Integrated debugger | This debugger supports the in-circuit emulators for V850 microcontrollers.  The ID850QB is Windows-based software.<br>It has improved C-compatible debugging functions and can display the results of tracing with the source program using a window integration function that associates the source program, disassemble display, and memory display with the trace result.<br>It should be used in combination with the device file. |
|---|---|
| | Part number:  $\mu$S$\times\times\times\times$ ID703000-QB (ID850QB) |

**Remark**   $\times\times\times\times$ in the part number differs depending on the host machine and OS used.

$\mu$S<u>$\times\times\times\times$</u>ID703000-QB

| $\times\times\times\times$ | Host Machine | OS | Supply Medium |
|---|---|---|---|
| AB17 | PC-9800 series, | Windows (Japanese version) | CD-ROM |
| BB17 | IBM PC/AT compatibles | Windows (English version) | |

## A.6  Embedded Software

| RX850, RX850 Pro<br>Real-time OS | The RX850 and RX850 Pro are real-time OSs conforming to $\mu$ITRON 3.0 specifications.<br>A tool (configurator) for generating multiple information tables is supplied.<br>RX850 Pro has more functions than the RX850. |
|---|---|
| | Part number:  $\mu$S××××RX703000-ΔΔΔΔ (RX850)<br>$\mu$S××××RX703100-ΔΔΔΔ (RX850 Pro) |
| RX-FS850<br>(File system) | This is a FAT file system function.<br>It is a file system that supports the CD-ROM file system function.<br>This file system is used with the real-time OS RX850 Pro. |

**Caution  To purchase the RX850 or RX850 Pro, first fill in the purchase application form and sign the license agreement.**

**Remark**  ×××× and ΔΔΔΔ in the part number differ depending on the host machine and OS used.

$\mu$S××××RX703000-ΔΔΔΔ

$\mu$S××××RX703100-ΔΔΔΔ

| ΔΔΔΔ | Product Outline | Maximum Number for Use in Mass Production |
|---|---|---|
| 001 | Evaluation object | Do not use for mass-produced product. |
| 100K | Mass-production object | 0.1 million units |
| 001M | | 1 million units |
| 010M | | 10 million units |
| S01 | Source program | Object source program for mass production |

| ×××× | Host Machine | OS | Supply Medium |
|---|---|---|---|
| AB17 | PC-9800 series,<br>IBM PC/AT compatibles | Windows (Japanese version) | CD-ROM |
| BB17 | | Windows (English version) | |
| 3K17 | SPARCstation | Solaris (Rel. 2.5.1) | |

## A.7  Flash Memory Writing Tools

| Flashpro V (part number: PG-FP5)<br>Flash memory programmer | Flash memory programmer dedicated to microcontrollers with internal flash memory. |
|---|---|
| QB-MINI2 (MINICUBE2) | On-chip debug emulator with programming function. |
| FA-100GC-UEU-B<br>FA-128GF-GAT-B<br>Flash memory writing adapter | Flash memory writing adapter (not wired) used by connecting to the Flashpro IV, Flashpro V, etc.<br>• FA-100GC-UEU-B:  100-pin plastic LQFP (GC-UEU type)<br>• FA-128GF-GAT-B:  100-pin plastic LQFP (GF-GAT type) |

**Remark**  FA-100GC-UEU-B and FA-128GF-GAT-B are products of Naito Densei Machida Mfg. Co., Ltd.
TEL: +81-42-750-4172

## APPENDIX  B   MAJOR  DIFFERENCES  BETWEEN  V850ES/Jx3-U  AND  V850ES/Jx3

**Table B-1.  Major Differences Between V850ES/Jx3-U and V850ES/Jx3**

| Major Difference | V850ES/Jx3-U | V850ES/Jx3 |
|---|---|---|
| Minimum instruction execution time | 20.8 ns (48 MHz operation) | 31.25 ns (32 MHz operation) |
| 16-bit timer | TAA (high-performance type of TMP) | TMP |
|  | TAB (high-performance type of TMQ) | TMQ |
|  | TMT (encoder timer) | None |
| Watch timer function | RTC<br>(Hardware counter included) | WT<br>(Hardware counter not included) |
| Motor control function | Available | None |
| USB interface | Function | None |
| Asynchronous serial interface | UARTC (high-performance type of UARTA) | UARTA |
| Package | 100-pin LQFP<br>128-pin LQFP | 100-pin LQFP<br>144-pin LQFP |

**APPENDIX  C  REGISTER  INDEX**

(1/25)

| Symbol | Name | Unit | Page |
|---|---|---|---|
| ADA0CR0 | A/D conversion result register 0 | ADC | 691 |
| ADA0CR0H | A/D conversion result register 0H | ADC | 691 |
| ADA0CR1 | A/D conversion result register 1 | ADC | 691 |
| ADA0CR1H | A/D conversion result register 1H | ADC | 691 |
| ADA0CR2 | A/D conversion result register 2 | ADC | 691 |
| ADA0CR2H | A/D conversion result register 2H | ADC | 691 |
| ADA0CR3 | A/D conversion result register 3 | ADC | 691 |
| ADA0CR3H | A/D conversion result register 3H | ADC | 691 |
| ADA0CR4 | A/D conversion result register 4 | ADC | 691 |
| ADA0CR4H | A/D conversion result register 4H | ADC | 691 |
| ADA0CR5 | A/D conversion result register 5 | ADC | 691 |
| ADA0CR5H | A/D conversion result register 5H | ADC | 691 |
| ADA0CR6 | A/D conversion result register 6 | ADC | 691 |
| ADA0CR6H | A/D conversion result register 6H | ADC | 691 |
| ADA0CR7 | A/D conversion result register 7 | ADC | 691 |
| ADA0CR7H | A/D conversion result register 7H | ADC | 691 |
| ADA0CR8 | A/D conversion result register 8 | ADC | 691 |
| ADA0CR8H | A/D conversion result register 8H | ADC | 691 |
| ADA0CR9 | A/D conversion result register 9 | ADC | 691 |
| ADA0CR9H | A/D conversion result register 9H | ADC | 691 |
| ADA0CR10 | A/D conversion result register 10 | ADC | 691 |
| ADA0CR10H | A/D conversion result register 10H | ADC | 691 |
| ADA0CR11 | A/D conversion result register 11 | ADC | 691 |
| ADA0CR11H | A/D conversion result register 11H | ADC | 691 |
| ADA0M0 | A/D converter mode register 0 | ADC | 684 |
| ADA0M1 | A/D converter mode register 1 | ADC | 686 |
| ADA0M2 | A/D converter mode register 2 | ADC | 689 |
| ADA0PFM | Power-fail compare mode register | ADC | 693 |
| ADA0PFT | Power-fail compare threshold value register | ADC | 694 |
| ADA0S | Analog input channel specification register | ADC | 690 |
| ADIC | Interrupt control register | INTC | 1194 |
| AWC | Address wait control register | BCU | 189 |
| BCC | Bus cycle control register | BCU | 190 |
| BGR_CTL | Bus bridge control register | USBH | 1091 |
| BIST | BIST | USBH | 1108 |
| BRGINTT | Bridge interrupt control register | USBF | 1015 |
| BSC | Bus size configuration register | BCU | 178 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| Cache Line Size | Cache Line Size | USBH | 1108 |
| Capability Identifier | Capability Identifier | USBH | 1114 |
| Capability Pointer | Capability Pointer | USBH | 1111 |
| CCLS | CPU operation clock status register | CG | 204 |
| CF0CTL0 | CSIF0 control register 0 | CSIF | 770 |
| CF0CTL1 | CSIF0 control register 1 | CSIF | 773 |
| CF0CTL2 | CSIF0 control register 2 | CSIF | 774 |
| CF0RIC | Interrupt control register | INTC | 1194 |
| CF0RX | CSIF0 receive data register | CSIF | 768 |
| CF0RXL | CSIF0 receive data register L | CSIF | 768 |
| CF0STR | CSIF0 status register | CSIF | 776 |
| CF0TIC | Interrupt control register | INTC | 1194 |
| CF0TX | CSIF0 transmit data register | CSIF | 769 |
| CF0TXL | CSIF0 transmit data register L | CSIF | 769 |
| CF1CTL0 | CSIF1 control register 0 | CSIF | 770 |
| CF1CTL1 | CSIF1 control register 1 | CSIF | 773 |
| CF1CTL2 | CSIF1 control register 2 | CSIF | 774 |
| CF1RIC | Interrupt control register | INTC | 1194 |
| CF1RX | CSIF1 receive data register | CSIF | 768 |
| CF1RXL | CSIF1 receive data register L | CSIF | 768 |
| CF1STR | CSIF1 status register | CSIF | 776 |
| CF1TIC | Interrupt control register | INTC | 1194 |
| CF1TX | CSIF1 transmit data register | CSIF | 769 |
| CF1TXL | CSIF1 transmit data register L | CSIF | 769 |
| CF2CTL0 | CSIF2 control register 0 | CSIF | 770 |
| CF2CTL1 | CSIF2 control register 1 | CSIF | 773 |
| CF2CTL2 | CSIF2 control register 2 | CSIF | 774 |
| CF2RIC | Interrupt control register | INTC | 1194 |
| CF2RX | CSIF2 receive data register | CSIF | 768 |
| CF2RXL | CSIF2 receive data register L | CSIF | 768 |
| CF2STR | CSIF2 status register | CSIF | 776 |
| CF2TIC | Interrupt control register | INTC | 1194 |
| CF2TX | CSIF2 transmit data register | CSIF | 769 |
| CF2TXL | CSIF2 transmit data register L | CSIF | 769 |
| CF3CTL0 | CSIF3 control register 0 | CSIF | 770 |
| CF3CTL1 | CSIF3 control register 1 | CSIF | 773 |
| CF3CTL2 | CSIF3 control register 2 | CSIF | 774 |
| CF3RIC | Interrupt control register | INTC | 1194 |
| CF3RX | CSIF3 receive data register | CSIF | 768 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| CF3RXL | CSIF3 receive data register L | CSIF | 768 |
| CF3STR | CSIF3 status register | CSIF | 776 |
| CF3TIC | Interrupt control register | INTC | 1194 |
| CF3TX | CSIF3 transmit data register | CSIF | 769 |
| CF3TXL | CSIF3 transmit data register L | CSIF | 769 |
| CF4CTL0 | CSIF4 control register 0 | CSIF | 770 |
| CF4CTL1 | CSIF4 control register 1 | CSIF | 773 |
| CF4CTL2 | CSIF4 control register 2 | CSIF | 774 |
| CF4RIC | Interrupt control register | INTC | 1194 |
| CF4RX | CSIF4 receive data register | CSIF | 768 |
| CF4RXL | CSIF4 receive data register L | CSIF | 768 |
| CF4STR | CSIF4 status register | CSIF | 776 |
| CF4TIC | Interrupt control register | INTC | 1194 |
| CF4TX | CSIF4 transmit data register | CSIF | 769 |
| CF4TXL | CSIF4 transmit data register L | CSIF | 769 |
| CKC | Clock control register | CG | 207 |
| Class Code | Class Code | USBH | 1107 |
| CLM | Clock monitor mode register | CLM | 1251 |
| Command | Command | USBH | 1105 |
| CPUBCTL | CPU I/F bus control register | USBF | 1017 |
| CRCD | CRC data register | CRC | 1262 |
| CRCIN | CRC input register | CRC | 1262 |
| CTBP | CALLT base pointer | CPU | 62 |
| CTPC | CALLT execution status saving register | CPU | 61 |
| CTPSW | CALLT execution status saving register | CPU | 61 |
| DA0CS0 | D/A conversion value setting register 0 | DAC | 718 |
| DA0CS1 | D/A conversion value setting register 1 | DAC | 718 |
| DA0M | D/A converter mode register | DAC | 712 |
| DADC0 | DMA addressing control register 0 | DMAC | 1151 |
| DADC1 | DMA addressing control register 1 | DMAC | 1151 |
| DADC2 | DMA addressing control register 2 | DMAC | 1151 |
| DADC3 | DMA addressing control register 3 | DMAC | 1151 |
| DBC0 | DMA transfer count register 0 | DMAC | 1150 |
| DBC1 | DMA transfer count register 1 | DMAC | 1150 |
| DBC2 | DMA transfer count register 2 | DMAC | 1150 |
| DBC3 | DMA transfer count register 3 | DMAC | 1150 |
| DBPC | Exception/debug trap status saving register | CPU | 62 |
| DBPSW | Exception/debug trap status saving register | CPU | 62 |
| DCHC0 | DMA channel control register 0 | DMAC | 1152 |
| DCHC1 | DMA channel control register 1 | DMAC | 1152 |
| DCHC2 | DMA channel control register 2 | DMAC | 1152 |
| DCHC3 | DMA channel control register 3 | DMAC | 1152 |

<div align="right">(4/25)</div>

| Symbol | Name | Unit | Page |
|---|---|---|---|
| DDA0H | DMA destination address register 0H | DMAC | 1149 |
| DDA0L | DMA destination address register 0L | DMAC | 1149 |
| DDA1H | DMA destination address register 1H | DMAC | 1149 |
| DDA1L | DMA destination address register 1L | DMAC | 1149 |
| DDA2H | DMA destination address register 2H | DMAC | 1149 |
| DDA2L | DMA destination address register 2L | DMAC | 1149 |
| DDA3H | DMA destination address register 3H | DMAC | 1149 |
| DDA3L | DMA destination address register 3L | DMAC | 1149 |
| Device ID | Device ID | USBH | 1104 |
| DMAIC0 | Interrupt control register | INTC | 1194 |
| DMAIC1 | Interrupt control register | INTC | 1194 |
| DMAIC2 | Interrupt control register | INTC | 1195 |
| DMAIC3 | Interrupt control register | INTC | 1195 |
| DSA0H | DMA source address register 0H | DMAC | 1148 |
| DSA0L | DMA source address register 0L | DMAC | 1148 |
| DSA1H | DMA source address register 1H | DMAC | 1148 |
| DSA1L | DMA source address register 1L | DMAC | 1148 |
| DSA2H | DMA source address register 2H | DMAC | 1148 |
| DSA2L | DMA source address register 2L | DMAC | 1148 |
| DSA3H | DMA source address register 3H | DMAC | 1148 |
| DSA3L | DMA source address register 3L | DMAC | 1148 |
| DTFR0 | DMA trigger factor register 0 | DMAC | 1153 |
| DTFR1 | DMA trigger factor register 1 | DMAC | 1153 |
| DTFR2 | DMA trigger factor register 2 | DMAC | 1153 |
| DTFR3 | DMA trigger factor register 3 | DMAC | 1153 |
| DWC0 | Data wait control register 0 | BCU | 186 |
| ECR | Interrupt source register | CPU | 59 |
| EIPC | Interrupt status saving register | CPU | 58 |
| EIPSW | Interrupt status saving register | CPU | 58 |
| EPCCLT | EPC macro control register | USBF | 1018 |
| EXDRQEN | External DMA request enable register | DMA | 1156 |
| FEPC | NMI status saving register | CPU | 59 |
| FEPSW | NMI status saving register | CPU | 59 |
| HcBulkCurrent ED Register | HcBulkCurrent ED Register | USBH | 1130 |
| HcBulkHead ED Register | HcBulkHead ED Register | USBH | 1130 |
| HcCommand Status Register | HcCommand Status Register | USBH | 1120 |
| HcControl Register | HcControl Register | USBH | 1129 |

(5/25)

| Symbol | Name | Unit | Page |
|---|---|---|---|
| HcControlCurrent ED Register | HcControlCurrent ED Register | USBH | 1129 |
| HcControlHead ED Register | HcControlHead ED Register | USBH | 1129 |
| HcDoneHead Register | HcDoneHead Register | USBH | 1131 |
| HcFmInterval Register | HcFmInterval Register | USBH | 1132 |
| HcFmNumber Register | HcFmNumber Register | USBH | 1133 |
| HcFmRemaining Register | HcFmRemaining Register | USBH | 1133 |
| HcHCCA Register | HcHCCA Register | USBH | 1128 |
| HcInterruptDisable Register | HcInterruptDisable Register | USBH | 1126 |
| HcInterruptEnable Register | HcInterruptEnable Register | USBH | 1124 |
| HcInterruptStatus Register | HcInterruptStatus Register | USBH | 1122 |
| HcLSThreshold Register | HcLSThreshold Register | USBH | 1134 |
| HcPeriodCurrentED Register | HcPeriodCurrentED Register | USBH | 1128 |
| HcPeriodicStart Register | HcPeriodicStart Register | USBH | 1134 |
| HcRevision Register | HcRevision Register | USBH | 1117 |
| HcRhDescriptorB Register | HcRhDescriptorB Register | USBH | 1137 |
| HcRhPortStatus1/ 2 Register | HcRhPortStatus1/2 Register | USBH | 1140 |
| HcRhStatus Register | HcRhStatus Register | USBH | 1138 |
| Header Type | Header Type | USBH | 1108 |
| HRDA | HcRhDescriptorA | USBH | 1135 |
| HZA0CTL0 | High-impedance output control register 0 | Motor | 587 |
| HZA0CTL1 | High-impedance output control register 1 | Motor | 587 |
| IIC0 | IIC shift register 0 | I²C | 836 |
| IIC1 | IIC shift register 1 | I²C | 836 |
| IIC2 | IIC shift register 2 | I²C | 836 |
| IICC0 | IIC control register 0 | I²C | 823 |
| IICC1 | IIC control register 1 | I²C | 823 |

RENESAS

(6/25)

| Symbol | Name | Unit | Page |
|--------|------|------|------|
| IICC2 | IIC control register 2 | I²C | 823 |
| IICCL0 | IIC clock select register 0 | I²C | 833 |
| IICCL1 | IIC clock select register 1 | I²C | 833 |
| IICCL2 | IIC clock select register 2 | I²C | 831 |
| IICF0 | IIC flag register 0 | I²C | 831 |
| IICF1 | IIC flag register 1 | I²C | 831 |
| IICF2 | IIC flag register 2 | I²C | 831 |
| IICIC0 | Interrupt control register | INTC | 1195 |
| IICIC1 | Interrupt control register | INTC | 1195 |
| IICIC2 | Interrupt control register | INTC | 1195 |
| IICS0 | IIC status register 0 | I²C | 828 |
| IICS1 | IIC status register 1 | I²C | 828 |
| IICS2 | IIC status register 2 | I²C | 828 |
| IICX0 | IIC function expansion register 0 | I²C | 834 |
| IICX1 | IIC function expansion register 1 | I²C | 834 |
| IICX2 | IIC function expansion register 2 | I²C | 834 |
| IMR0 | Interrupt mask register 0 | INTC | 1196 |
| IMR0H | Interrupt mask register 0H | INTC | 1196 |
| IMR0L | Interrupt mask register 0L | INTC | 1196 |
| IMR1 | Interrupt mask register 1 | INTC | 1196 |
| IMR1H | Interrupt mask register 1H | INTC | 1196 |
| IMR1L | Interrupt mask register 1L | INTC | 1196 |
| IMR2 | Interrupt mask register 2 | INTC | 1196 |
| IMR2H | Interrupt mask register 2H | INTC | 1196 |
| IMR2L | Interrupt mask register 2L | INTC | 1196 |
| IMR3 | Interrupt mask register 3 | INTC | 1196 |
| IMR3H | Interrupt mask register 3H | INTC | 1196 |
| IMR3L | Interrupt mask register 3L | INTC | 1196 |
| IMR4 | Interrupt mask register 4 | INTC | 1196 |
| IMR4H | Interrupt mask register 4H | INTC | 1196 |
| IMR4L | Interrupt mask register 4L | INTC | 1196 |
| IMR5 | Interrupt mask register 5 | INTC | 1196 |
| IMR5H | Interrupt mask register 5H | INTC | 1196 |
| IMR5L | Interrupt mask register 5L | INTC | 1196 |
| Interrupt Line | Interrupt Line | USBH | 1112 |
| Interrupt Pin | Interrupt Pin | USBH | 1112 |
| INTF0 | External falling edge specification register 0 | INTC | 1208 |
| INTF2 | External falling edge specification register 2 | INTC | 1209 |
| INTF3 | External falling edge specification register 3 | INTC | 1210 |
| INTF4 | External falling edge specification register 4 | INTC | 1211 |
| INTF5 | External falling edge specification register 5 | INTC | 1212 |
| INTF9 | External falling edge specification register 9 | INTC | 1213 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| INTF9H | External falling edge specification register 9H | INTC | 1213 |
| INTF9L | External falling edge specification register 9L | INTC | 1213 |
| INTNFC | Noise elimination control register | INTC | 1214 |
| INTR0 | External rising edge specification register 0 | INTC | 1208 |
| INTR2 | External rising edge specification register 2 | INTC | 1209 |
| INTR3 | External rising edge specification register 3 | INTC | 1210 |
| INTR4 | External rising edge specification register 4 | INTC | 1211 |
| INTR5 | External rising edge specification register 5 | INTC | 1212 |
| INTR9 | External rising edge specification register 9 | INTC | 1214 |
| INTR9H | External rising edge specification register 9H | INTC | 1214 |
| INTR9L | External rising edge specification register 9L | INTC | 1214 |
| ISPR | In-service priority register | INTC | 1198 |
| KRIC | Interrupt control register | INTC | 1195 |
| KRM | Key return mode register | KR | 1218 |
| Latency Timer | Latency Timer | USBH | 1108 |
| LOCKR | Lock register | CG | 208 |
| LVIIC | Interrupt control register | INTC | 1193 |
| LVIM | Low-voltage detection register | LVI | 1256 |
| Max Latency | Max Latency | USBH | 1112 |
| Min gnt | Min gnt | USBH | 1112 |
| Next Item Pointer | Next Item Pointer | USBH | 1113 |
| OCDM | On-chip debug mode register | DCU | 1304 |
| OCKS0 | IIC division clock select register 0 | I²C | 836 |
| OCKS1 | IIC division clock select register 1 | I²C | 836 |
| OHCI Base Address | OHCI Base Address | USBH | 1109 |
| OSTS | Oscillation stabilization time select register | Standby | 1223 |
| P0 | Port 0 register | Port | 108 |
| P1 | Port 1 register | Port | 113 |
| P2 | Port 2 register | Port | 114 |
| P3 | Port 3 register | Port | 118 |
| P4 | Port 4 register | Port | 123 |
| P5 | Port 5 register | Port | 127 |
| P6 | Port 6 register | Port | 133 |
| P7H | Port 7 register H | Port | 137 |
| P7L | Port 7 register L | Port | 137 |
| P9 | Port 9 register | Port | 140 |
| P9H | Port 9 register H | Port | 140 |
| P9L | Port 9 register L | Port | 140 |
| PC | Program counter | CPU | 56 |
| PCC | Processor clock control register | CG | 203 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| PCI_CONFIG_ADD | PCI configuration address register | USBH | 1092 |
| PCI_CONFIG_DATA | PCI configuration data register | USBH | 1091 |
| PCI_CONTROL1 | PCI control register 1 | USBH | 1093 |
| PCI_CONTROL2 | PCI control register 2 | USBH | 1094 |
| PCI_ERR_ADD | PCI bus error address register | USBH | 1109 |
| PCI_INT_CTL | PCI interrupt control register | USBH | 1108 |
| PCI_INT_STATUS | PCI interrupt status register | USBH | 1106 |
| PCI_WIN_BASE | PCI windows base address register | USBH | 1095 |
| PCM | Port CM register | Port | 149 |
| PCS | Port CS register | Port | 152 |
| PCT | Port CT register | Port | 154 |
| PDH | Port DH register | Port | 158 |
| PDL | Port DL register | Port | 160 |
| PDLH | Port DL register H | Port | 160 |
| PDLL | Port DL register L | Port | 160 |
| PF0 | Port 0 function register | Port | 112 |
| PF2 | Port 2 function register | Port | 117 |
| PF3 | Port 3 function register | Port | 122 |
| PF4 | Port 4 function register | Port | 125 |
| PF5 | Port 5 function register | Port | 132 |
| PF9 | Port 9 function register | Port | 148 |
| PF9L | Port 9 function register L | Port | 148 |
| PFC0 | Port 0 function control register | Port | 111 |
| PFC2 | Port 2 function control register | Port | 115 |
| PFC3 | Port 3 function control register | Port | 120 |
| PFC4 | Port 4 function control register | Port | 124 |
| PFC5 | Port 5 function control register | Port | 130 |
| PFC6 | Port 6 function control register | Port | 135 |
| PFC9 | Port 9 function control register | Port | 143 |
| PFC9H | Port 9 function control register H | Port | 143 |
| PFC9L | Port 9 function control register L | Port | 143 |
| PFCE0 | Port 0 function control expansion register | Port | 112 |
| PFCE2 | Port 2 function control expansion register | Port | 116 |
| PFCE3 | Port 3 function control expansion register | Port | 120 |
| PFCE4 | Port 4 function control expansion register | Port | 124 |
| PFCE5 | Port 5 function control expansion register | Port | 130 |
| PFCE6 | Port 6 function control expansion register | Port | 135 |
| PFCE9 | Port 9 function control expansion register | Port | 144 |
| PFCE9H | Port 9 function control expansion register H | Port | 144 |

| Symbol | Name | Unit | Page |
|--------|------|------|------|
| PFCE9L | Port 9 function control expansion register L | Port | 144 |
| PIC00 | Interrupt control register | INTC | 1193 |
| PIC01 | Interrupt control register | INTC | 1193 |
| PIC02 | Interrupt control register | INTC | 1193 |
| PIC03 | Interrupt control register | INTC | 1193 |
| PIC04 | Interrupt control register | INTC | 1193 |
| PIC05 | Interrupt control register | INTC | 1193 |
| PIC06 | Interrupt control register | INTC | 1193 |
| PIC07 | Interrupt control register | INTC | 1193 |
| PIC08 | Interrupt control register | INTC | 1193 |
| PIC09 | Interrupt control register | INTC | 1193 |
| PIC10 | Interrupt control register | INTC | 1193 |
| PIC11 | Interrupt control register | INTC | 1193 |
| PIC12 | Interrupt control register | INTC | 1193 |
| PIC13 | Interrupt control register | INTC | 1193 |
| PIC14 | Interrupt control register | INTC | 1193 |
| PIC15 | Interrupt control register | INTC | 1193 |
| PIC16 | Interrupt control register | INTC | 1193 |
| PIC17 | Interrupt control register | INTC | 1193 |
| PIC18 | Interrupt control register | INTC | 1193 |
| PLLCTL | PLL control register | CG | 206 |
| PLLS | PLL lockup time specification register | CG | 209 |
| PM0 | Port 0 mode register | Port | 109 |
| PM1 | Port 1 mode register | Port | 113 |
| PM2 | Port 2 mode register | Port | 114 |
| PM3 | Port 3 mode register | Port | 118 |
| PM4 | Port 4 mode register | Port | 123 |
| PM5 | Port 5 mode register | Port | 128 |
| PM6 | Port 6 mode register | Port | 134 |
| PM7H | Port 7 mode register H | Port | 138 |
| PM7L | Port 7 mode register L | Port | 138 |
| PM9 | Port 9 mode register | Port | 140 |
| PM9L | Port 9 mode register L | Port | 140 |
| PMC0 | Port 0 mode control register | Port | 110 |
| PMC2 | Port 2 mode control register | Port | 115 |
| PMC3 | Port 3 mode control register | Port | 119 |
| PMC4 | Port 4 mode control register | Port | 124 |
| PMC5 | Port 5 mode control register | Port | 129 |
| PMC6 | Port 6 mode control register | Port | 134 |
| PMC9 | Port 9 mode control register | Port | 141 |
| PMC9H | Port 9 mode control register H | Port | 141 |
| PMC9L | Port 9 mode control register L | Port | 141 |

APPENDIX C REGISTER INDEX

(10/25)

| Symbol | Name | Unit | Page |
|--------|------|------|------|
| PMCCM | Port CM mode control register | Port | 151 |
| PMCCS | Port CS mode control register | Port | 153 |
| PMCCT | Port CT mode control register | Port | 156 |
| PMCDH | Port DH mode control register | Port | 158 |
| PMCDL | Port DL mode control register | Port | 161 |
| PMCDLH | Port DL mode control register H | Port | 161 |
| PMCDLL | Port DL mode control register L | Port | 161 |
| PMCM | Port CM mode register | Port | 150 |
| PMCS | Port CS mode register | Port | 152 |
| PMCSR    Bridge Support Extensions | PMCSR Bridge Support Extensions | USBH | 1114 |
| PMCT | Port CT mode register | Port | 155 |
| PMDH | Port DH mode register | Port | 158 |
| PMDL | Port DL mode register | Port | 160 |
| PMDLH | Port DL mode register H | Port | 160 |
| PMDLL | Port DL mode register L | Port | 160 |
| Power Management Capabilities | Power Management Capabilities | USBH | 1113 |
| Power Management Control/Status | Power Management Control/Status | USBH | 1114 |
| PRCMD | Command register | CPU | 91 |
| PRSCM0 | Prescaler compare register 0 | BRG | 655 |
| PRSCM1 | Prescaler compare register 1 | BRG | 813 |
| PRSCM2 | Prescaler compare register 2 | BRG | 813 |
| PRSCM3 | Prescaler compare register 3 | BRG | 813 |
| PRSM0 | Prescaler mode register 0 | BRG | 654 |
| PRSM1 | Prescaler mode register 1 | BRG | 812 |
| PRSM2 | Prescaler mode register 2 | BRG | 812 |
| PRSM3 | Prescaler mode register 3 | BRG | 812 |
| PSC | Power save control register | CG | 1221 |
| PSMR | Power save mode register | CG | 1222 |
| PSW | Program status word | CPU | 60 |
| r0-r31 | General-purpose registers | CPU | 56 |
| RAMS | Internal RAM data status register | LVI | 1257 |
| RC1ALH | Alarm hour setting register | RTC | 651 |
| RC1ALM | Alarm minute setting register | RTC | 651 |
| RC1ALW | Alarm day-of week setting register | RTC | 652 |
| RC1CC0 | Real-time counter control register 0 | RTC | 640 |
| RC1CC1 | Real-time counter control register 1 | RTC | 641 |
| RC1CC2 | Real-time counter control register 2 | RTC | 642 |

| Symbol | Name | Unit | Page |
|--------|------|------|------|
| RC1CC3 | Real-time counter control register 3 | RTC | 643 |
| RC1DAY | Day count register | RTC | 647 |
| RC1HOUR | Hour count register | RTC | 45 |
| RC1MIN | Minute count register | RTC | 645 |
| RC1MONTH | Month count register | RTC | 649 |
| RC1SEC | Second count register | RTC | 643 |
| RC1SUBC | Sub-count register | RTC | 643 |
| RC1SUBU | Watch error correction register | RTC | 650 |
| RC1WEEK | Day-of-week count register | RTC | 648 |
| RC1YEAR | Year count register | RTC | 649 |
| RCM | Internal oscillation mode register | CG | 204 |
| RESF | Reset source flag register | Reset | 1242 |
| Revision ID | Revision ID | USBH | 1107 |
| RTBH0 | Real-time output buffer register 0H | RTO | 675 |
| RTBL0 | Real-time output buffer register 0L | RTO | 675 |
| RTC0IC | Interrupt control register | INTC | 1195 |
| RTC1IC | Interrupt control register | INTC | 1195 |
| RTC2IC | Interrupt control register | INTC | 1195 |
| RTPC0 | Real-time output port control register 0 | RTO | 677 |
| RTPM0 | Real-time output port mode register 0 | RTO | 676 |
| SELCNT0 | Selector operation control register 0 | Timer | 326 |
| SRAM_BASE | SRAM_BASE register | USBH | 1101 |
| Status | Status | USBH | 1105 |
| SubSystem Vender ID | SubSystem Vender ID | USBH | 1110 |
| SubSystem ID | SubSystem ID | USBH | 1110 |
| SVA0 | Slave address register 0 | I$^2$C | 837 |
| SVA1 | Slave address register 1 | I$^2$C | 837 |
| SVA2 | Slave address register 2 | I$^2$C | 837 |
| SYS | System status register | CPU | 92 |
| TAA0CCIC0 | Interrupt control register | INTC | 1193 |
| TAA0CCIC1 | Interrupt control register | INTC | 1193 |
| TAA0CCR0 | TAA0 capture/compare register 0 | Timer | 224 |
| TAA0CCR1 | TAA0 capture/compare register 1 | Timer | 226 |
| TAA0CNT | TAA0 counter read buffer register | Timer | 228 |
| TAA0CTL0 | TAA0 control register 0 | Timer | 215 |
| TAA0CTL1 | TAA0 control register 1 | Timer | 216 |
| TAA0IOC0 | TAA0 I/O control register 0 | Timer | 218 |
| TAA0IOC1 | TAA0 I/O control register 1 | Timer | 219 |
| TAA0IOC2 | TAA0 I/O control register 2 | Timer | 220 |
| TAA0IOC4 | TAA0 I/O control register 4 | Timer | 221 |
| TAA0OPT0 | TAA0 option register 0 | Timer | 222 |

(12/25)

| Symbol | Name | Unit | Page |
|---|---|---|---|
| TAA0OPT1 | TAA0 option register 1 | Timer | 223 |
| TAA0OVIC | Interrupt control register | INTC | 1193 |
| TAA1CCIC0 | Interrupt control register | INTC | 1193 |
| TAA1CCIC1 | Interrupt control register | INTC | 1193 |
| TAA1CCR0 | TAA1 capture/compare register 0 | Timer | 224 |
| TAA1CCR1 | TAA1 capture/compare register 1 | Timer | 226 |
| TAA1CNT | TAA1 counter read buffer register | Timer | 228 |
| TAA1CTL0 | TAA1 control register 0 | Timer | 215 |
| TAA1CTL1 | TAA1 control register 1 | Timer | 216 |
| TAA1IOC0 | TAA1I/O control register 0 | Timer | 218 |
| TAA1IOC1 | TAA1I/O control register 1 | Timer | 219 |
| TAA1IOC2 | TAA1I/O control register 2 | Timer | 220 |
| TAA1IOC4 | TAA1I/O control register 4 | Timer | 221 |
| TAA1OPT0 | TAA1 option register 0 | Timer | 222 |
| TAA1OVIC | Interrupt control register | INTC | 1193 |
| TAA2CCIC0 | Interrupt control register | INTC | 1193 |
| TAA2CCIC1 | Interrupt control register | INTC | 1193 |
| TAA2CCR0 | TAA2 capture/compare register 0 | Timer | 224 |
| TAA2CCR1 | TAA2 capture/compare register 1 | Timer | 226 |
| TAA2CNT | TAA2 counter read buffer register | Timer | 228 |
| TAA2CTL0 | TAA2 control register 0 | Timer | 215 |
| TAA2CTL1 | TAA2 control register 1 | Timer | 216 |
| TAA2IOC0 | TAA2 I/O control register 0 | Timer | 218 |
| TAA2IOC1 | TAA2 I/O control register 1 | Timer | 219 |
| TAA2IOC2 | TAA2 I/O control register 2 | Timer | 220 |
| TAA2IOC4 | TAA2 I/O control register 4 | Timer | 221 |
| TAA2OPT0 | TAA2 option register 0 | Timer | 222 |
| TAA2OPT1 | TAA2 option register 1 | Timer | 223 |
| TAA2OVIC | Interrupt control register | INTC | 1193 |
| TAA3CCIC0 | Interrupt control register | INTC | 1193 |
| TAA3CCIC1 | Interrupt control register | INTC | 1193 |
| TAA3CCR0 | TAA3 capture/compare register 0 | Timer | 1193 |
| TAA3CCR1 | TAA3 capture/compare register 1 | Timer | 224 |
| TAA3CNT | TAA3 counter read buffer register | Timer | 226 |
| TAA3CTL0 | TAA3 control register 0 | Timer | 228 |
| TAA3CTL1 | TAA3 control register 1 | Timer | 215 |
| TAA3IOC0 | TAA3 I/O control register 0 | Timer | 216 |
| TAA3IOC1 | TAA3 I/O control register 1 | Timer | 218 |
| TAA3IOC2 | TAA3 I/O control register 2 | Timer | 219 |
| TAA3IOC4 | TAA3 I/O control register 4 | Timer | 220 |
| TAA3OPT0 | TAA3 option register 0 | Timer | 221 |
| TAA3OVIC | Interrupt control register | INTC | 1193 |

| Symbol | Name | Unit | Page |
|--------|------|------|------|
| TAA4CCIC0 | Interrupt control register | INTC | 1193 |
| TAA4CCIC1 | Interrupt control register | INTC | 1193 |
| TAA4CCR0 | TAA4 capture/compare register 0 | Timer | 224 |
| TAA4CCR1 | TAA4 capture/compare register 1 | Timer | 226 |
| TAA4CNT | TAA4 counter read buffer register | Timer | 228 |
| TAA4CTL0 | TAA4 control register 0 | Timer | 215 |
| TAA4CTL1 | TAA4 control register 1 | Timer | 216 |
| TAA4OVIC | Interrupt control register | INTC | 218 |
| TAA5CCIC0 | Interrupt control register | INTC | 1193 |
| TAA5CCIC1 | Interrupt control register | INTC | 1193 |
| TAA5CCR0 | TAA5 capture/compare register 0 | Timer | 224 |
| TAA5CCR1 | TAA5 capture/compare register 1 | Timer | 226 |
| TAA5CNT | TAA5 counter read buffer register | Timer | 228 |
| TAA5CTL0 | TAA5 control register 0 | Timer | 215 |
| TAA5CTL1 | TAA5 control register 1 | Timer | 216 |
| TAA5IOC0 | TAA5 I/O control register 0 | Timer | 218 |
| TAA5IOC1 | TAA5 I/O control register 1 | Timer | 219 |
| TAA5IOC2 | TAA5 I/O control register 2 | Timer | 220 |
| TAA5IOC4 | TAA5 I/O control register 4 | Timer | 221 |
| TAA5OPT0 | TAA5 option register 0 | Timer | 222 |
| TAA5OVIC | Interrupt control register | INTC | 1193 |
| TAB0CCIC0 | Interrupt control register | INTC | 1194 |
| TAB0CCIC1 | Interrupt control register | INTC | 1194 |
| TAB0CCIC2 | Interrupt control register | INTC | 1194 |
| TAB0CCIC3 | Interrupt control register | INTC | 1194 |
| TAB0CCR0 | TAB0 capture/compare register 0 | Timer | 340 |
| TAB0CCR1 | TAB0 capture/compare register 1 | Timer | 342 |
| TAB0CCR2 | TAB0 capture/compare register 2 | Timer | 344 |
| TAB0CCR3 | TAB0 capture/compare register 3 | Timer | 346 |
| TAB0CNT | TAB0 counter read buffer register | Timer | 348 |
| TAB0CTL0 | TAB0 control register 0 | Timer | 333 |
| TAB0CTL1 | TAB0 control register 1 | Timer | 334 |
| TAB0IOC0 | TAB0 I/O control register 0 | Timer | 335 |
| TAB0IOC1 | TAB0 I/O control register 1 | Timer | 336 |
| TAB0IOC2 | TAB0 I/O control register 2 | Timer | 337 |
| TAB0OIC4 | TAB0 I/O control register 4 | Timer | 338 |
| TAB0OPT0 | TAB0 option register 0 | Timer | 339 |
| TAB0OVIC | Interrupt control register | INTC | 1193 |
| TAB1CCIC0 | Interrupt control register | INTC | 1194 |
| TAB1CCIC1 | Interrupt control register | INTC | 1194 |
| TAB1CCIC2 | Interrupt control register | INTC | 1194 |
| TAB1CCIC3 | Interrupt control register | INTC | 1194 |
| TAB1CCR0 | TAB1 capture/compare register 0 | Timer | 340 |

RENESAS

| Symbol | Name | Unit | Page |
|--------|------|------|------|
| TAB1CCR1 | TAB1 capture/compare register 1 | Timer | 342 |
| TAB1CCR2 | TAB1 capture/compare register 2 | Timer | 344 |
| TAB1CCR3 | TAB1 capture/compare register 3 | Timer | 346 |
| TAB1CNT | TAB1 counter read buffer register | Timer | 348 |
| TAB1CTL0 | TAB1 control register 0 | Timer | 333 |
| TAB1CTL1 | TAB1 control register 1 | Timer | 334 |
| TAB1DTC | TAB1 dead-time compare register 1 | Timer | 581 |
| TAB1IOC0 | TAB1 I/O control register 0 | Timer | 335 |
| TAB1IOC1 | TAB1 I/O control register 1 | Timer | 336 |
| TAB1IOC2 | TAB1 I/O control register 2 | Timer | 337 |
| TAB1IOC3 | TAB1 I/O control register 3 | Timer | 585 |
| TAB1OIC4 | TAB1 I/O control register 4 | Timer | 338 |
| TAB1OPT0 | TAB1 option register 0 | Timer | 339 |
| TAB1OPT1 | TAB1 option register 1 | Timer | 582 |
| TAB1OPT2 | TAB1 option register 2 | Timer | 583 |
| TAB1OVIC | Interrupt control register | INTC | 1194 |
| TANFC | Noise elimination control register | Timer | 229 |
| TM0CMP0 | TMM0 compare register 0 | Timer | 569 |
| TM0CTL0 | TMM0 control register 0 | Timer | 570 |
| TM0EQIC0 | Interrupt control register | INTC | 1194 |
| TM1CMP0 | TMM1 compare register 0 | Timer | 569 |
| TM1CTL0 | TMM1 control register 0 | Timer | 570 |
| TM1EQIC0 | Interrupt control register | INTC | 1194 |
| TM2CMP0 | TMM2 compare register 0 | Timer | 569 |
| TM2CTL0 | TMM2 control register 0 | Timer | 570 |
| TM2EQIC0 | Interrupt control register | INTC | 1194 |
| TM3CMP0 | TMM3 compare register 0 | Timer | 569 |
| TM3CTL0 | TMM3 control register 0 | Timer | 570 |
| TM3EQIC0 | Interrupt control register | INTC | 1194 |
| TTNFC | Noise elimination control register | Timer | 456 |
| TT0CCIC0 | Interrupt control register | INTC | 1194 |
| TT0CCIC1 | Interrupt control register | INTC | 1194 |
| TT0CCR0 | TMT0 capture/compare register 0 | Timer | 451 |
| TT0CCR1 | TMT0 capture/compare register 1 | Timer | 453 |
| TT0CNT | TMT0 counter read buffer register | Timer | 455 |
| TT0CTL0 | TMT0 control register 0 | Timer | 437 |
| TT0CTL1 | TMT0 control register 1 | Timer | 438 |
| TT0CTL2 | TMT0 control register 2 | Timer | 440 |
| TT0IECIC | Interrupt control register | INTC | 1194 |
| TT0IOC0 | TMT0 I/O control register 0 | Timer | 442 |
| TT0IOC1 | TMT0 I/O control register 1 | Timer | 444 |
| TT0IOC2 | TMT0 I/O control register 2 | Timer | 445 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| TT0IOC3 | TMT0 I/O control register 3 | Timer | 446 |
| TT0OPT0 | TMT0 option register 0 | Timer | 448 |
| TT0OPT1 | TMT0 option register 1 | Timer | 449 |
| TT0OVIC | Interrupt control register | INTC | 1194 |
| TT0TCW | TMT0 count write register | Timer | 455 |
| TTNFC | Noise elimination control register | Timer | 456 |
| UC0CTL0 | UARTC0 control register 0 | UARTC | 729 |
| UC0CTL1 | UARTC0 control register 1 | UARTC | 755 |
| UC0CTL2 | UARTC0 control register 2 | UARTC | 756 |
| UC0OPT0 | UARTC0 option control register 0 | UARTC | 731 |
| UC0OPT1 | UARTC0 option control register 1 | UARTC | 733 |
| UC0RIC | Interrupt control register | INTC | 1194 |
| UC0RX | UARTC0 receive data register | UARTC | 737 |
| UC0RXL | UARTC0 receive data register L | UARTC | 737 |
| UC0STR | UARTC0 status register | UARTC | 735 |
| UC0TIC | Interrupt control register | INTC | 1194 |
| UC0TX | UARTC0 transmit data register | UARTC | 738 |
| UC0TXL | UARTC0 transmit data register L | UARTC | 738 |
| UC1CTL0 | UARTC1 control register 0 | UARTC | 729 |
| UC1CTL1 | UARTC1 control register 1 | UARTC | 755 |
| UC1CTL2 | UARTC1 control register 2 | UARTC | 756 |
| UC1OPT0 | UARTC1 option control register 0 | UARTC | 731 |
| UC1OPT1 | UARTC1 option control register 1 | UARTC | 733 |
| UC1RIC | Interrupt control register | INTC | 1194 |
| UC1RX | UARTC1 receive data register | UARTC | 737 |
| UC1RXL | UARTC1 receive data register L | UARTC | 737 |
| UC1STR | UARTC1 status register | UARTC | 735 |
| UC1TIC | Interrupt control register | INTC | 1194 |
| UC1TX | UARTC1 transmit data register | UARTC | 738 |
| UC1TXL | UARTC1 transmit data register L | UARTC | 738 |
| UC2CTL0 | UARTC2 control register 0 | UARTC | 729 |
| UC2CTL1 | UARTC2 control register 1 | UARTC | 755 |
| UC2CTL2 | UARTC2 control register 2 | UARTC | 756 |
| UC2OPT0 | UARTC2 option control register 0 | UARTC | 731 |
| UC2OPT1 | UARTC2 option control register 1 | UARTC | 733 |
| UC2RIC | Interrupt control register | INTC | 1194 |
| UC2RX | UARTC2 receive data register | UARTC | 737 |
| UC2RXL | UARTC2 receive data register L | UARTC | 737 |
| UC2STR | UARTC2 status register | UARTC | 735 |
| UC2TIC | Interrupt control register | INTC | 1194 |
| UC2TX | UARTC2 transmit data register | UARTC | 738 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| UC2TXL | UARTC2 transmit data register L | UARTC | 738 |
| UC3CTL0 | UARTC3 control register 0 | UARTC | 729 |
| UC3CTL1 | UARTC3 control register 1 | UARTC | 755 |
| UC3CTL2 | UARTC3 control register 2 | UARTC | 756 |
| UC3OPT0 | UARTC3 option control register 0 | UARTC | 731 |
| UC3OPT1 | UARTC3 option control register 1 | UARTC | 733 |
| UC3RIC | Interrupt control register | INTC | 1194 |
| UC3RX | UARTC3 receive data register | UARTC | 737 |
| UC3RXL | UARTC3 receive data register L | UARTC | 737 |
| UC3STR | UARTC3 status register | UARTC | 735 |
| UC3TIC | Interrupt control register | INTC | 1194 |
| UC3TX | UARTC3 transmit data register | UARTC | 738 |
| UC3TXL | UARTC3 transmit data register L | UARTC | 738 |
| UC4CTL0 | UARTC4 control register 0 | UARTC | 729 |
| UC4CTL1 | UARTC4 control register 1 | UARTC | 755 |
| UC4CTL2 | UARTC4 control register 2 | UARTC | 756 |
| UC4OPT0 | UARTC4 option control register 0 | UARTC | 731 |
| UC4OPT1 | UARTC4 option control register 1 | UARTC | 733 |
| UC4RIC | Interrupt control register | INTC | 1194 |
| UC4RX | UARTC4 receive data register | UARTC | 737 |
| UC4RXL | UARTC4 receive data register L | UARTC | 737 |
| UC4STR | UARTC4 status register | UARTC | 735 |
| UC4TIC | Interrupt control register | INTC | 1194 |
| UC4TX | UARTC4 transmit data register | UARTC | 738 |
| UC4TXL | UARTC4 transmit data register L | UARTC | 738 |
| UCKSEL | USB clock select register | USB | 908 |
| UF0AAS | UF0 active alternative setting register | USBF | 971 |
| UF0ADRS | UF0 address register | USBF | 1008 |
| UF0AIFN | UF0 active interface number register | USBF | 970 |
| UF0ASS | UF0 alternative setting status register | USBF | 972 |
| UF0BI1 | UF0 bulk-in 1 register | USBF | 991 |
| UF0BI2 | UF0 bulk-in 2 register | USBF | 995 |
| UF0BO1 | UF0 bulk-out 1 register | USBF | 984 |
| UF0BO1L | UF0 bulk-out 1 length register | USBF | 987 |
| UF0BO2 | UF0 bulk-out 2 register | USBF | 988 |
| UF0BO2L | UF0 bulk-out 2 length register | USBF | 991 |
| UF0CIE0 | UF0 configuration/interface/endpoint descriptor register 0 | USBF | 1014 |
| UF0CIE1 | UF0 configuration/interface/endpoint descriptor register 1 | USBF | 1014 |
| UF0CIE2 | UF0 configuration/interface/endpoint descriptor register 2 | USBF | 1014 |
| UF0CIE3 | UF0 configuration/interface/endpoint descriptor register 3 | USBF | 1014 |
| UF0CIE4 | UF0 configuration/interface/endpoint descriptor register 4 | USBF | 1014 |
| UF0CIE5 | UF0 configuration/interface/endpoint descriptor register 5 | USBF | 1014 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| UF0CIE6 | UF0 configuration/interface/endpoint descriptor register 6 | USBF | 1014 |
| UF0CIE7 | UF0 configuration/interface/endpoint descriptor register 7 | USBF | 1014 |
| UF0CIE8 | UF0 configuration/interface/endpoint descriptor register 8 | USBF | 1014 |
| UF0CIE9 | UF0 configuration/interface/endpoint descriptor register 9 | USBF | 1014 |
| UF0CIE10 | UF0 configuration/interface/endpoint descriptor register 10 | USBF | 1014 |
| UF0CIE11 | UF0 configuration/interface/endpoint descriptor register 11 | USBF | 1014 |
| UF0CIE12 | UF0 configuration/interface/endpoint descriptor register 12 | USBF | 1014 |
| UF0CIE13 | UF0 configuration/interface/endpoint descriptor register 13 | USBF | 1014 |
| UF0CIE14 | UF0 configuration/interface/endpoint descriptor register 14 | USBF | 1014 |
| UF0CIE15 | UF0 configuration/interface/endpoint descriptor register 15 | USBF | 1014 |
| UF0CIE16 | UF0 configuration/interface/endpoint descriptor register 16 | USBF | 1014 |
| UF0CIE17 | UF0 configuration/interface/endpoint descriptor register 17 | USBF | 1014 |
| UF0CIE18 | UF0 configuration/interface/endpoint descriptor register 18 | USBF | 1014 |
| UF0CIE19 | UF0 configuration/interface/endpoint descriptor register 19 | USBF | 1014 |
| UF0CIE20 | UF0 configuration/interface/endpoint descriptor register 20 | USBF | 1014 |
| UF0CIE21 | UF0 configuration/interface/endpoint descriptor register 21 | USBF | 1014 |
| UF0CIE22 | UF0 configuration/interface/endpoint descriptor register 22 | USBF | 1014 |
| UF0CIE23 | UF0 configuration/interface/endpoint descriptor register 23 | USBF | 1014 |
| UF0CIE24 | UF0 configuration/interface/endpoint descriptor register 24 | USBF | 1014 |
| UF0CIE25 | UF0 configuration/interface/endpoint descriptor register 25 | USBF | 1014 |
| UF0CIE26 | UF0 configuration/interface/endpoint descriptor register 26 | USBF | 1014 |
| UF0CIE27 | UF0 configuration/interface/endpoint descriptor register 27 | USBF | 1014 |
| UF0CIE28 | UF0 configuration/interface/endpoint descriptor register 28 | USBF | 1014 |
| UF0CIE29 | UF0 configuration/interface/endpoint descriptor register 29 | USBF | 1014 |
| UF0CIE30 | UF0 configuration/interface/endpoint descriptor register 30 | USBF | 1014 |
| UF0CIE31 | UF0 configuration/interface/endpoint descriptor register 31 | USBF | 1014 |
| UF0CIE32 | UF0 configuration/interface/endpoint descriptor register 32 | USBF | 1014 |
| UF0CIE33 | UF0 configuration/interface/endpoint descriptor register 33 | USBF | 1014 |
| UF0CIE34 | UF0 configuration/interface/endpoint descriptor register 34 | USBF | 1014 |
| UF0CIE35 | UF0 configuration/interface/endpoint descriptor register 35 | USBF | 1014 |
| UF0CIE36 | UF0 configuration/interface/endpoint descriptor register 36 | USBF | 1014 |
| UF0CIE37 | UF0 configuration/interface/endpoint descriptor register 37 | USBF | 1014 |
| UF0CIE38 | UF0 configuration/interface/endpoint descriptor register 38 | USBF | 1014 |
| UF0CIE39 | UF0 configuration/interface/endpoint descriptor register 39 | USBF | 1014 |
| UF0CIE40 | UF0 configuration/interface/endpoint descriptor register 40 | USBF | 1014 |
| UF0CIE41 | UF0 configuration/interface/endpoint descriptor register 41 | USBF | 1014 |
| UF0CIE42 | UF0 configuration/interface/endpoint descriptor register 42 | USBF | 1014 |
| UF0CIE43 | UF0 configuration/interface/endpoint descriptor register 43 | USBF | 1014 |
| UF0CIE44 | UF0 configuration/interface/endpoint descriptor register 44 | USBF | 1014 |
| UF0CIE45 | UF0 configuration/interface/endpoint descriptor register 45 | USBF | 1014 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| UF0CIE46 | UF0 configuration/interface/endpoint descriptor register 46 | USBF | 1014 |
| UF0CIE47 | UF0 configuration/interface/endpoint descriptor register 47 | USBF | 1014 |
| UF0CIE48 | UF0 configuration/interface/endpoint descriptor register 48 | USBF | 1014 |
| UF0CIE49 | UF0 configuration/interface/endpoint descriptor register 49 | USBF | 1014 |
| UF0CIE50 | UF0 configuration/interface/endpoint descriptor register 50 | USBF | 1014 |
| UF0CIE51 | UF0 configuration/interface/endpoint descriptor register 51 | USBF | 1014 |
| UF0CIE52 | UF0 configuration/interface/endpoint descriptor register 52 | USBF | 1014 |
| UF0CIE53 | UF0 configuration/interface/endpoint descriptor register 53 | USBF | 1014 |
| UF0CIE54 | UF0 configuration/interface/endpoint descriptor register 54 | USBF | 1014 |
| UF0CIE55 | UF0 configuration/interface/endpoint descriptor register 55 | USBF | 1014 |
| UF0CIE56 | UF0 configuration/interface/endpoint descriptor register 56 | USBF | 1014 |
| UF0CIE57 | UF0 configuration/interface/endpoint descriptor register 57 | USBF | 1014 |
| UF0CIE58 | UF0 configuration/interface/endpoint descriptor register 58 | USBF | 1014 |
| UF0CIE59 | UF0 configuration/interface/endpoint descriptor register 59 | USBF | 1014 |
| UF0CIE60 | UF0 configuration/interface/endpoint descriptor register 60 | USBF | 1014 |
| UF0CIE61 | UF0 configuration/interface/endpoint descriptor register 61 | USBF | 1014 |
| UF0CIE62 | UF0 configuration/interface/endpoint descriptor register 62 | USBF | 1014 |
| UF0CIE63 | UF0 configuration/interface/endpoint descriptor register 63 | USBF | 1014 |
| UF0CIE64 | UF0 configuration/interface/endpoint descriptor register 64 | USBF | 1014 |
| UF0CIE65 | UF0 configuration/interface/endpoint descriptor register 65 | USBF | 1014 |
| UF0CIE66 | UF0 configuration/interface/endpoint descriptor register 66 | USBF | 1014 |
| UF0CIE67 | UF0 configuration/interface/endpoint descriptor register 67 | USBF | 1014 |
| UF0CIE68 | UF0 configuration/interface/endpoint descriptor register 68 | USBF | 1014 |
| UF0CIE69 | UF0 configuration/interface/endpoint descriptor register 69 | USBF | 1014 |
| UF0CIE70 | UF0 configuration/interface/endpoint descriptor register 70 | USBF | 1014 |
| UF0CIE71 | UF0 configuration/interface/endpoint descriptor register 71 | USBF | 1014 |
| UF0CIE72 | UF0 configuration/interface/endpoint descriptor register 72 | USBF | 1014 |
| UF0CIE73 | UF0 configuration/interface/endpoint descriptor register 73 | USBF | 1014 |
| UF0CIE74 | UF0 configuration/interface/endpoint descriptor register 74 | USBF | 1014 |
| UF0CIE75 | UF0 configuration/interface/endpoint descriptor register 75 | USBF | 1014 |
| UF0CIE76 | UF0 configuration/interface/endpoint descriptor register 76 | USBF | 1014 |
| UF0CIE77 | UF0 configuration/interface/endpoint descriptor register 77 | USBF | 1014 |
| UF0CIE78 | UF0 configuration/interface/endpoint descriptor register 78 | USBF | 1014 |
| UF0CIE79 | UF0 configuration/interface/endpoint descriptor register 79 | USBF | 1014 |
| UF0CIE80 | UF0 configuration/interface/endpoint descriptor register 80 | USBF | 1014 |
| UF0CIE81 | UF0 configuration/interface/endpoint descriptor register 81 | USBF | 1014 |
| UF0CIE82 | UF0 configuration/interface/endpoint descriptor register 82 | USBF | 1014 |
| UF0CIE83 | UF0 configuration/interface/endpoint descriptor register 83 | USBF | 1014 |
| UF0CIE84 | UF0 configuration/interface/endpoint descriptor register 84 | USBF | 1014 |
| UF0CIE85 | UF0 configuration/interface/endpoint descriptor register 85 | USBF | 1014 |

(19/25)

| Symbol | Name | Unit | Page |
|---|---|---|---|
| UF0CIE86 | UF0 configuration/interface/endpoint descriptor register 86 | USBF | 1014 |
| UF0CIE87 | UF0 configuration/interface/endpoint descriptor register 87 | USBF | 1014 |
| UF0CIE88 | UF0 configuration/interface/endpoint descriptor register 88 | USBF | 1014 |
| UF0CIE89 | UF0 configuration/interface/endpoint descriptor register 89 | USBF | 1014 |
| UF0CIE90 | UF0 configuration/interface/endpoint descriptor register 90 | USBF | 1014 |
| UF0CIE91 | UF0 configuration/interface/endpoint descriptor register 91 | USBF | 1014 |
| UF0CIE92 | UF0 configuration/interface/endpoint descriptor register 92 | USBF | 1014 |
| UF0CIE93 | UF0 configuration/interface/endpoint descriptor register 93 | USBF | 1014 |
| UF0CIE94 | UF0 configuration/interface/endpoint descriptor register 94 | USBF | 1014 |
| UF0CIE95 | UF0 configuration/interface/endpoint descriptor register 95 | USBF | 1014 |
| UF0CIE96 | UF0 configuration/interface/endpoint descriptor register 96 | USBF | 1014 |
| UF0CIE97 | UF0 configuration/interface/endpoint descriptor register 97 | USBF | 1014 |
| UF0CIE98 | UF0 configuration/interface/endpoint descriptor register 98 | USBF | 1014 |
| UF0CIE99 | UF0 configuration/interface/endpoint descriptor register 99 | USBF | 1014 |
| UF0CIE100 | UF0 configuration/interface/endpoint descriptor register 100 | USBF | 1014 |
| UF0CIE101 | UF0 configuration/interface/endpoint descriptor register 101 | USBF | 1014 |
| UF0CIE102 | UF0 configuration/interface/endpoint descriptor register 102 | USBF | 1014 |
| UF0CIE103 | UF0 configuration/interface/endpoint descriptor register 103 | USBF | 1014 |
| UF0CIE104 | UF0 configuration/interface/endpoint descriptor register 104 | USBF | 1014 |
| UF0CIE105 | UF0 configuration/interface/endpoint descriptor register 105 | USBF | 1014 |
| UF0CIE106 | UF0 configuration/interface/endpoint descriptor register 106 | USBF | 1014 |
| UF0CIE107 | UF0 configuration/interface/endpoint descriptor register 107 | USBF | 1014 |
| UF0CIE108 | UF0 configuration/interface/endpoint descriptor register 108 | USBF | 1014 |
| UF0CIE109 | UF0 configuration/interface/endpoint descriptor register 109 | USBF | 1014 |
| UF0CIE110 | UF0 configuration/interface/endpoint descriptor register 110 | USBF | 1014 |
| UF0CIE111 | UF0 configuration/interface/endpoint descriptor register 111 | USBF | 1014 |
| UF0CIE112 | UF0 configuration/interface/endpoint descriptor register 112 | USBF | 1014 |
| UF0CIE113 | UF0 configuration/interface/endpoint descriptor register 113 | USBF | 1014 |
| UF0CIE114 | UF0 configuration/interface/endpoint descriptor register 114 | USBF | 1014 |
| UF0CIE115 | UF0 configuration/interface/endpoint descriptor register 115 | USBF | 1014 |
| UF0CIE116 | UF0 configuration/interface/endpoint descriptor register 116 | USBF | 1014 |
| UF0CIE117 | UF0 configuration/interface/endpoint descriptor register 117 | USBF | 1014 |
| UF0CIE118 | UF0 configuration/interface/endpoint descriptor register 118 | USBF | 1014 |
| UF0CIE119 | UF0 configuration/interface/endpoint descriptor register 119 | USBF | 1014 |
| UF0CIE120 | UF0 configuration/interface/endpoint descriptor register 120 | USBF | 1014 |
| UF0CIE121 | UF0 configuration/interface/endpoint descriptor register 121 | USBF | 1014 |
| UF0CIE122 | UF0 configuration/interface/endpoint descriptor register 122 | USBF | 1014 |
| UF0CIE123 | UF0 configuration/interface/endpoint descriptor register 123 | USBF | 1014 |
| UF0CIE124 | UF0 configuration/interface/endpoint descriptor register 124 | USBF | 1014 |
| UF0CIE125 | UF0 configuration/interface/endpoint descriptor register 125 | USBF | 1014 |

(20/25)

| Symbol | Name | Unit | Page |
|---|---|---|---|
| UF0CIE126 | UF0 configuration/interface/endpoint descriptor register 126 | USBF | 1014 |
| UF0CIE127 | UF0 configuration/interface/endpoint descriptor register 127 | USBF | 1014 |
| UF0CIE128 | UF0 configuration/interface/endpoint descriptor register 128 | USBF | 1014 |
| UF0CIE129 | UF0 configuration/interface/endpoint descriptor register 129 | USBF | 1014 |
| UF0CIE130 | UF0 configuration/interface/endpoint descriptor register 130 | USBF | 1014 |
| UF0CIE131 | UF0 configuration/interface/endpoint descriptor register 131 | USBF | 1014 |
| UF0CIE132 | UF0 configuration/interface/endpoint descriptor register 132 | USBF | 1014 |
| UF0CIE133 | UF0 configuration/interface/endpoint descriptor register 133 | USBF | 1014 |
| UF0CIE134 | UF0 configuration/interface/endpoint descriptor register 134 | USBF | 1014 |
| UF0CIE135 | UF0 configuration/interface/endpoint descriptor register 135 | USBF | 1014 |
| UF0CIE136 | UF0 configuration/interface/endpoint descriptor register 136 | USBF | 1014 |
| UF0CIE137 | UF0 configuration/interface/endpoint descriptor register 137 | USBF | 1014 |
| UF0CIE138 | UF0 configuration/interface/endpoint descriptor register 138 | USBF | 1014 |
| UF0CIE139 | UF0 configuration/interface/endpoint descriptor register 139 | USBF | 1014 |
| UF0CIE140 | UF0 configuration/interface/endpoint descriptor register 140 | USBF | 1014 |
| UF0CIE141 | UF0 configuration/interface/endpoint descriptor register 141 | USBF | 1014 |
| UF0CIE142 | UF0 configuration/interface/endpoint descriptor register 142 | USBF | 1014 |
| UF0CIE143 | UF0 configuration/interface/endpoint descriptor register 143 | USBF | 1014 |
| UF0CIE144 | UF0 configuration/interface/endpoint descriptor register 144 | USBF | 1014 |
| UF0CIE145 | UF0 configuration/interface/endpoint descriptor register 145 | USBF | 1014 |
| UF0CIE146 | UF0 configuration/interface/endpoint descriptor register 146 | USBF | 1014 |
| UF0CIE147 | UF0 configuration/interface/endpoint descriptor register 147 | USBF | 1014 |
| UF0CIE148 | UF0 configuration/interface/endpoint descriptor register 148 | USBF | 1014 |
| UF0CIE149 | UF0 configuration/interface/endpoint descriptor register 149 | USBF | 1014 |
| UF0CIE150 | UF0 configuration/interface/endpoint descriptor register 150 | USBF | 1014 |
| UF0CIE151 | UF0 configuration/interface/endpoint descriptor register 151 | USBF | 1014 |
| UF0CIE152 | UF0 configuration/interface/endpoint descriptor register 152 | USBF | 1014 |
| UF0CIE153 | UF0 configuration/interface/endpoint descriptor register 153 | USBF | 1014 |
| UF0CIE154 | UF0 configuration/interface/endpoint descriptor register 154 | USBF | 1014 |
| UF0CIE155 | UF0 configuration/interface/endpoint descriptor register 155 | USBF | 1014 |
| UF0CIE156 | UF0 configuration/interface/endpoint descriptor register 156 | USBF | 1014 |
| UF0CIE157 | UF0 configuration/interface/endpoint descriptor register 157 | USBF | 1014 |
| UF0CIE158 | UF0 configuration/interface/endpoint descriptor register 158 | USBF | 1014 |
| UF0CIE159 | UF0 configuration/interface/endpoint descriptor register 159 | USBF | 1014 |
| UF0CIE160 | UF0 configuration/interface/endpoint descriptor register 160 | USBF | 1014 |
| UF0CIE161 | UF0 configuration/interface/endpoint descriptor register 161 | USBF | 1014 |
| UF0CIE162 | UF0 configuration/interface/endpoint descriptor register 162 | USBF | 1014 |
| UF0CIE163 | UF0 configuration/interface/endpoint descriptor register 163 | USBF | 1014 |
| UF0CIE164 | UF0 configuration/interface/endpoint descriptor register 164 | USBF | 1014 |
| UF0CIE165 | UF0 configuration/interface/endpoint descriptor register 165 | USBF | 1014 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| UF0CIE166 | UF0 configuration/interface/endpoint descriptor register 166 | USBF | 1014 |
| UF0CIE167 | UF0 configuration/interface/endpoint descriptor register 167 | USBF | 1014 |
| UF0CIE168 | UF0 configuration/interface/endpoint descriptor register 168 | USBF | 1014 |
| UF0CIE169 | UF0 configuration/interface/endpoint descriptor register 169 | USBF | 1014 |
| UF0CIE170 | UF0 configuration/interface/endpoint descriptor register 170 | USBF | 1014 |
| UF0CIE171 | UF0 configuration/interface/endpoint descriptor register 171 | USBF | 1014 |
| UF0CIE172 | UF0 configuration/interface/endpoint descriptor register 172 | USBF | 1014 |
| UF0CIE173 | UF0 configuration/interface/endpoint descriptor register 173 | USBF | 1014 |
| UF0CIE174 | UF0 configuration/interface/endpoint descriptor register 174 | USBF | 1014 |
| UF0CIE175 | UF0 configuration/interface/endpoint descriptor register 175 | USBF | 1014 |
| UF0CIE176 | UF0 configuration/interface/endpoint descriptor register 176 | USBF | 1014 |
| UF0CIE177 | UF0 configuration/interface/endpoint descriptor register 177 | USBF | 1014 |
| UF0CIE178 | UF0 configuration/interface/endpoint descriptor register 178 | USBF | 1014 |
| UF0CIE179 | UF0 configuration/interface/endpoint descriptor register 179 | USBF | 1014 |
| UF0CIE180 | UF0 configuration/interface/endpoint descriptor register 180 | USBF | 1014 |
| UF0CIE181 | UF0 configuration/interface/endpoint descriptor register 181 | USBF | 1014 |
| UF0CIE182 | UF0 configuration/interface/endpoint descriptor register 182 | USBF | 1014 |
| UF0CIE183 | UF0 configuration/interface/endpoint descriptor register 183 | USBF | 1014 |
| UF0CIE184 | UF0 configuration/interface/endpoint descriptor register 184 | USBF | 1014 |
| UF0CIE185 | UF0 configuration/interface/endpoint descriptor register 185 | USBF | 1014 |
| UF0CIE186 | UF0 configuration/interface/endpoint descriptor register 186 | USBF | 1014 |
| UF0CIE187 | UF0 configuration/interface/endpoint descriptor register 187 | USBF | 1014 |
| UF0CIE188 | UF0 configuration/interface/endpoint descriptor register 188 | USBF | 1014 |
| UF0CIE189 | UF0 configuration/interface/endpoint descriptor register 189 | USBF | 1014 |
| UF0CIE190 | UF0 configuration/interface/endpoint descriptor register 190 | USBF | 1014 |
| UF0CIE191 | UF0 configuration/interface/endpoint descriptor register 191 | USBF | 1014 |
| UF0CIE192 | UF0 configuration/interface/endpoint descriptor register 192 | USBF | 1014 |
| UF0CIE193 | UF0 configuration/interface/endpoint descriptor register 193 | USBF | 1014 |
| UF0CIE194 | UF0 configuration/interface/endpoint descriptor register 194 | USBF | 1014 |
| UF0CIE195 | UF0 configuration/interface/endpoint descriptor register 195 | USBF | 1014 |
| UF0CIE196 | UF0 configuration/interface/endpoint descriptor register 196 | USBF | 1014 |
| UF0CIE197 | UF0 configuration/interface/endpoint descriptor register 197 | USBF | 1014 |
| UF0CIE198 | UF0 configuration/interface/endpoint descriptor register 198 | USBF | 1014 |
| UF0CIE199 | UF0 configuration/interface/endpoint descriptor register 199 | USBF | 1014 |
| UF0CIE200 | UF0 configuration/interface/endpoint descriptor register 200 | USBF | 1014 |
| UF0CIE201 | UF0 configuration/interface/endpoint descriptor register 201 | USBF | 1014 |
| UF0CIE202 | UF0 configuration/interface/endpoint descriptor register 202 | USBF | 1014 |
| UF0CIE203 | UF0 configuration/interface/endpoint descriptor register 203 | USBF | 1014 |
| UF0CIE204 | UF0 configuration/interface/endpoint descriptor register 204 | USBF | 1014 |
| UF0CIE205 | UF0 configuration/interface/endpoint descriptor register 205 | USBF | 1014 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| UF0CIE206 | UF0 configuration/interface/endpoint descriptor register 206 | USBF | 1014 |
| UF0CIE207 | UF0 configuration/interface/endpoint descriptor register 207 | USBF | 1014 |
| UF0CIE208 | UF0 configuration/interface/endpoint descriptor register 208 | USBF | 1014 |
| UF0CIE209 | UF0 configuration/interface/endpoint descriptor register 209 | USBF | 1014 |
| UF0CIE210 | UF0 configuration/interface/endpoint descriptor register 210 | USBF | 1014 |
| UF0CIE211 | UF0 configuration/interface/endpoint descriptor register 211 | USBF | 1014 |
| UF0CIE212 | UF0 configuration/interface/endpoint descriptor register 212 | USBF | 1014 |
| UF0CIE213 | UF0 configuration/interface/endpoint descriptor register 213 | USBF | 1014 |
| UF0CIE214 | UF0 configuration/interface/endpoint descriptor register 214 | USBF | 1014 |
| UF0CIE215 | UF0 configuration/interface/endpoint descriptor register 215 | USBF | 1014 |
| UF0CIE216 | UF0 configuration/interface/endpoint descriptor register 216 | USBF | 1014 |
| UF0CIE217 | UF0 configuration/interface/endpoint descriptor register 217 | USBF | 1014 |
| UF0CIE218 | UF0 configuration/interface/endpoint descriptor register 218 | USBF | 1014 |
| UF0CIE219 | UF0 configuration/interface/endpoint descriptor register 219 | USBF | 1014 |
| UF0CIE220 | UF0 configuration/interface/endpoint descriptor register 220 | USBF | 1014 |
| UF0CIE221 | UF0 configuration/interface/endpoint descriptor register 221 | USBF | 1014 |
| UF0CIE222 | UF0 configuration/interface/endpoint descriptor register 222 | USBF | 1014 |
| UF0CIE223 | UF0 configuration/interface/endpoint descriptor register 223 | USBF | 1014 |
| UF0CIE224 | UF0 configuration/interface/endpoint descriptor register 224 | USBF | 1014 |
| UF0CIE225 | UF0 configuration/interface/endpoint descriptor register 225 | USBF | 1014 |
| UF0CIE226 | UF0 configuration/interface/endpoint descriptor register 226 | USBF | 1014 |
| UF0CIE227 | UF0 configuration/interface/endpoint descriptor register 227 | USBF | 1014 |
| UF0CIE228 | UF0 configuration/interface/endpoint descriptor register 228 | USBF | 1014 |
| UF0CIE229 | UF0 configuration/interface/endpoint descriptor register 229 | USBF | 1014 |
| UF0CIE230 | UF0 configuration/interface/endpoint descriptor register 230 | USBF | 1014 |
| UF0CIE231 | UF0 configuration/interface/endpoint descriptor register 231 | USBF | 1014 |
| UF0CIE232 | UF0 configuration/interface/endpoint descriptor register 232 | USBF | 1014 |
| UF0CIE233 | UF0 configuration/interface/endpoint descriptor register 233 | USBF | 1014 |
| UF0CIE234 | UF0 configuration/interface/endpoint descriptor register 234 | USBF | 1014 |
| UF0CIE235 | UF0 configuration/interface/endpoint descriptor register 235 | USBF | 1014 |
| UF0CIE236 | UF0 configuration/interface/endpoint descriptor register 236 | USBF | 1014 |
| UF0CIE237 | UF0 configuration/interface/endpoint descriptor register 237 | USBF | 1014 |
| UF0CIE238 | UF0 configuration/interface/endpoint descriptor register 238 | USBF | 1014 |
| UF0CIE239 | UF0 configuration/interface/endpoint descriptor register 239 | USBF | 1014 |
| UF0CIE240 | UF0 configuration/interface/endpoint descriptor register 240 | USBF | 1014 |
| UF0CIE241 | UF0 configuration/interface/endpoint descriptor register 241 | USBF | 1014 |
| UF0CIE242 | UF0 configuration/interface/endpoint descriptor register 242 | USBF | 1014 |
| UF0CIE243 | UF0 configuration/interface/endpoint descriptor register 243 | USBF | 1014 |
| UF0CIE244 | UF0 configuration/interface/endpoint descriptor register 244 | USBF | 1014 |
| UF0CIE245 | UF0 configuration/interface/endpoint descriptor register 245 | USBF | 1014 |

| Symbol | Name | Unit | Page |
|--------|------|------|------|
| UF0CIE246 | UF0 configuration/interface/endpoint descriptor register 246 | USBF | 1014 |
| UF0CIE247 | UF0 configuration/interface/endpoint descriptor register 247 | USBF | 1014 |
| UF0CIE248 | UF0 configuration/interface/endpoint descriptor register 248 | USBF | 1014 |
| UF0CIE249 | UF0 configuration/interface/endpoint descriptor register 249 | USBF | 1014 |
| UF0CIE250 | UF0 configuration/interface/endpoint descriptor register 250 | USBF | 1014 |
| UF0CIE251 | UF0 configuration/interface/endpoint descriptor register 251 | USBF | 1014 |
| UF0CIE252 | UF0 configuration/interface/endpoint descriptor register 252 | USBF | 1014 |
| UF0CIE253 | UF0 configuration/interface/endpoint descriptor register 253 | USBF | 1014 |
| UF0CIE254 | UF0 configuration/interface/endpoint descriptor register 254 | USBF | 1014 |
| UF0CIE255 | UF0 configuration/interface/endpoint descriptor register 255 | USBF | 1014 |
| UF0CLR | UF0 CLR request register | USBF | 935 |
| UF0CNF | UF0 configuration register | USBF | 1009 |
| UF0DD0 | UF0 device descriptor register 0 | USBF | 1013 |
| UF0DD1 | UF0 device descriptor register 1 | USBF | 1013 |
| UF0DD2 | UF0 device descriptor register 2 | USBF | 1013 |
| UF0DD3 | UF0 device descriptor register 3 | USBF | 1013 |
| UF0DD4 | UF0 device descriptor register 4 | USBF | 1013 |
| UF0DD5 | UF0 device descriptor register 5 | USBF | 1013 |
| UF0DD6 | UF0 device descriptor register 6 | USBF | 1013 |
| UF0DD7 | UF0 device descriptor register 7 | USBF | 1013 |
| UF0DD8 | UF0 device descriptor register 8 | USBF | 1013 |
| UF0DD9 | UF0 device descriptor register 9 | USBF | 1013 |
| UF0DD10 | UF0 device descriptor register 10 | USBF | 1013 |
| UF0DD11 | UF0 device descriptor register 11 | USBF | 1013 |
| UF0DD12 | UF0 device descriptor register 12 | USBF | 1013 |
| UF0DD13 | UF0 device descriptor register 13 | USBF | 1013 |
| UF0DD14 | UF0 device descriptor register 14 | USBF | 1013 |
| UF0DD15 | UF0 device descriptor register 15 | USBF | 1013 |
| UF0DD16 | UF0 device descriptor register 16 | USBF | 1013 |
| UF0DD17 | UF0 device descriptor register 17 | USBF | 1013 |
| UF0DEND | UF0 data end register | USBF | 965 |
| UF0DMS0 | UF0 DMA status 0 register | USBF | 961 |
| UF0DMS1 | UF0 DMA status 1 register | USBF | 962 |
| UF0DSCL | UF0 descriptor length register | USBF | 1012 |
| UF0DSTL | UF0 device status register L | USBF | 1001 |
| UF0E0L | UF0 EP0 length register | USBF | 979 |
| UF0E0N | UF0 EP0NAK register | USBF | 929 |
| UF0E0NA | UF0 EP0NAKALL register | USBF | 929 |
| UF0E0R | UF0 EP0 read register | USBF | 977 |
| UF0E0SL | UF0 EP0 status register L | USBF | 1002 |

| Symbol | Name | Unit | Page |
|--------|------|------|------|
| UF0E0ST | UF0 EP0 setup register | USBF | 979 |
| UF0E0W | UF0 EP0 write register | USBF | 981 |
| UF0E1DC1 | EP1 DMA control register 1 | USBF | 1019 |
| UF0E1DC2 | EP1 DMA control register 2 | USBF | 1021 |
| UF0E1IM | UF0 endpoint 1 interface mapping register | USBF | 972 |
| UF0E1SL | UF0 EP1 status register L | USBF | 1002 |
| UF0E2DC1 | EP2 DMA control register 1 | USBF | 1019 |
| UF0E2DC2 | EP2 DMA control register 2 | USBF | 1021 |
| UF0E2IM | UF0 endpoint 2 interface mapping register | USBF | 973 |
| UF0E2SL | UF0 EP2 status register L | USBF | 1003 |
| UF0E3DC1 | EP3 DMA control register 1 | USBF | 1019 |
| UF0E3DC2 | EP3 DMA control register 2 | USBF | 1021 |
| UF0E3IM | UF0 endpoint 3 interface mapping register | USBF | 974 |
| UF0E3SL | UF0 EP3 status register L | USBF | 1004 |
| UF0E4DC1 | EP4 DMA control register 1 | USBF | 1019 |
| UF0E4DC2 | EP4 DMA control register 2 | USBF | 1021 |
| UF0E4IM | UF0 endpoint 4 interface mapping register | USBF | 975 |
| UF0E4SL | UF0 EP4 status register L | USBF | 1005 |
| UF0E7IM | UF0 endpoint 7 interface mapping register | USBF | 976 |
| UF0E7SL | UF0 EP7 status register L | USBF | 1006 |
| UF0EN | UF0 EPNAK register | USBF | 928 |
| UF0ENM | UF0 EPNAK mask register | USBF | 932 |
| UF0EP1BI | UF0 EP1 bulk-in transfer data register | USBF | 1023 |
| UF0EP2BO | UF0 EP2 bulk-out transfer data register | USBF | 1024 |
| UF0EP3BI | UF0 EP3 bulk-in transfer data register | USBF | 1023 |
| UF0EP4BO | UF0 EP4 bulk-out transfer data register | USBF | 1025 |
| UF0EPS0 | UF0 EP status 0 register | USBF | 936 |
| UF0EPS1 | UF0 EP status 1 register | USBF | 938 |
| UF0EPS2 | UF0 EP status 2 register | USBF | 939 |
| UF0FIC0 | UF0 FIFO clear 0 register | USBF | 962 |
| UF0FIC1 | UF0 FIFO clear 1 register | USBF | 963 |
| UF0GPR | UF0 GPR register | USBF | 966 |
| UF0IC0 | UF0 INT clear 0 register | USBF | 953 |
| UF0IC1 | UF0 INT clear 1 register | USBF | 954 |
| UF0IC2 | UF0 INT clear 2 register | USBF | 955 |
| UF0IC3 | UF0 INT clear 3 register | USBF | 956 |
| UF0IC4 | UF0 INT clear 4 register | USBF | 957 |
| UF0IDR | UF0 INT & DMARQ register | USBF | 958 |
| UF0IF0 | UF0 interface 0 register | USBF | 1009 |
| UF0IF1 | UF0 interface 1 register | USBF | 1010 |

| Symbol | Name | Unit | Page |
|---|---|---|---|
| UF0IF2 | UF0 interface 2 register | USBF | 1010 |
| UF0IF3 | UF0 interface 3 register | USBF | 1010 |
| UF0IF4 | UF0 interface 4 register | USBF | 1010 |
| UF0IM0 | UF0 INT mask 0 register | USBF | 948 |
| UF0IM1 | UF0 INT mask 1 register | USBF | 949 |
| UF0IM2 | UF0 INT mask 2 register | USBF | 950 |
| UF0IM3 | UF0 INT mask 3 register | USBF | 951 |
| UF0IM4 | UF0 INT mask 4 register | USBF | 952 |
| UF0INT1 | UF0 interrupt 1 register | USBF | 998 |
| UF0IS0 | UF0 INT status 0 register | USBF | 940 |
| UF0IS1 | UF0 INT status 1 register | USBF | 942 |
| UF0IS2 | UF0 INT status 2 register | USBF | 944 |
| UF0IS3 | UF0 INT status 3 register | USBF | 945 |
| UF0IS4 | UF0 INT status 4 register | USBF | 947 |
| UF0MODC | UF0 mode control register | USBF | 967 |
| UF0MODS | UF0 mode status register | USBF | 968 |
| UF0SDS | UF0 SNDSIE register | USBF | 934 |
| UF0SET | UF0 SET request register | USBF | 935 |
| UFCKMSK | USB function control register | USBF | 907 |
| UFDRQEN | USBF DMA request enable register | USBF | 1026 |
| UFIC0 | Interrupt control register | INTC | 1995 |
| UFIC1 | Interrupt control register | INTC | 1995 |
| UHCKMSK | USB function select register | USB | 908 |
| UHIC0 | Interrupt register | INTC | 1995 |
| UHIC1 | Interrupt register | INTC | 1995 |
| UHIC2 | Interrupt register | INTC | 1995 |
| Vender ID | Vender ID | USBH | 1104 |
| VSWC | System wait control register | CPU | 93 |
| WDTE | Watchdog timer enable register | WDT | 672 |
| WDTM2 | Watchdog timer mode register 2 | WDT | 670 |

<div align="center">

**APPENDIX  D  INSTRUCTION  SET  LIST**

</div>

## D.1   Conventions

**(1)  Register symbols used to describe operands**

| Register Symbol | Explanation |
|---|---|
| reg1 | General-purpose registers:  Used as source registers. |
| reg2 | General-purpose registers:  Used mainly as destination registers.  Also used as source register in some instructions. |
| reg3 | General-purpose registers:  Used mainly to store the remainders of division results and the higher 32 bits of multiplication results. |
| bit#3 | 3-bit data for specifying the bit number |
| immX | X bit immediate data |
| dispX | X bit displacement data |
| regID | System register number |
| vector | 5-bit data that specifies the trap vector (00H to 1FH) |
| cccc | 4-bit data that shows the conditions code |
| sp | Stack pointer (r3) |
| ep | Element pointer (r30) |
| listX | X item register list |

**(2)  Register symbols used to describe opcodes**

| Register Symbol | Explanation |
|---|---|
| R | 1-bit data of a code that specifies reg1 or regID |
| r | 1-bit data of the code that specifies reg2 |
| w | 1-bit data of the code that specifies reg3 |
| d | 1-bit displacement data |
| I | 1-bit immediate data (indicates the higher bits of immediate data) |
| i | 1-bit immediate data |
| cccc | 4-bit data that shows the condition codes |
| CCCC | 4-bit data that shows the condition codes of Bcond instruction |
| bbb | 3-bit data for specifying the bit number |
| L | 1-bit data that specifies a program register in the register list |

### (3) Register symbols used in operations

| Register Symbol | Explanation |
| --- | --- |
| ← | Input for |
| GR [ ] | General-purpose register |
| SR [ ] | System register |
| zero-extend (n) | Expand n with zeros until word length. |
| sign-extend (n) | Expand n with signs until word length. |
| load-memory (a, b) | Read size b data from address a. |
| store-memory (a, b, c) | Write data b into address a in size c. |
| load-memory-bit (a, b) | Read bit b of address a. |
| store-memory-bit (a, b, c) | Write c to bit b of address a. |
| saturated (n) | Execute saturated processing of n (n is a 2's complement).<br>If, as a result of calculations,<br>   n ≥ 7FFFFFFFH, let it be 7FFFFFFFH.<br>   n ≤ 80000000H, let it be 80000000H. |
| result | Reflects the results in a flag. |
| Byte | Byte (8 bits) |
| Halfword | Half word (16 bits) |
| Word | Word (32 bits) |
| + | Addition |
| − | Subtraction |
| ‖ | Bit concatenation |
| × | Multiplication |
| ÷ | Division |
| % | Remainder from division results |
| AND | Logical product |
| OR | Logical sum |
| XOR | Exclusive OR |
| NOT | Logical negation |
| logically shift left by | Logical shift left |
| logically shift right by | Logical shift right |
| arithmetically shift right by | Arithmetic shift right |

### (4) Register symbols used in execution clock

| Register Symbol | Explanation |
| --- | --- |
| i | If executing another instruction immediately after executing the first instruction (issue). |
| r | If repeating execution of the same instruction immediately after executing the first instruction (repeat). |
| l | If using the results of instruction execution in the instruction immediately after the execution (latency). |

### (5) Register symbols used in flag operations

| Identifier | Explanation |
|---|---|
| (Blank) | No change |
| 0 | Clear to 0 |
| X | Set or cleared in accordance with the results. |
| R | Previously saved values are restored. |

### (6) Condition codes

| Condition Code (cccc) | Condition Formula | Explanation |
|---|---|---|
| 0 0 0 0 | OV = 1 | Overflow |
| 1 0 0 0 | OV = 0 | No overflow |
| 0 0 0 1 | CY = 1 | Carry<br>Lower (Less than) |
| 1 0 0 1 | CY = 0 | No carry<br>Not lower (Greater than or equal) |
| 0 0 1 0 | Z = 1 | Zero |
| 1 0 1 0 | Z = 0 | Not zero |
| 0 0 1 1 | (CY or Z) = 1 | Not higher (Less than or equal) |
| 1 0 1 1 | (CY or Z) = 0 | Higher (Greater than) |
| 0 1 0 0 | S = 1 | Negative |
| 1 1 0 0 | S = 0 | Positive |
| 0 1 0 1 | – | Always (Unconditional) |
| 1 1 0 1 | SAT = 1 | Saturated |
| 0 1 1 0 | (S xor OV) = 1 | Less than signed |
| 1 1 1 0 | (S xor OV) = 0 | Greater than or equal signed |
| 0 1 1 1 | ((S xor OV) or Z) = 1 | Less than or equal signed |
| 1 1 1 1 | ((S xor OV) or Z) = 0 | Greater than signed |

## D.2  Instruction Set (in Alphabetical Order)

(1/6)

| Mnemonic | Operand | Opcode | Operation | | Execution Clock | | | Flags | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | i | r | l | CY | OV | S | Z | SAT |
| ADD | reg1,reg2 | rrrrr001110RRRRR | GR[reg2]←GR[reg2]+GR[reg1] | | 1 | 1 | 1 | × | × | × | × | |
| | imm5,reg2 | rrrrr010010iiiii | GR[reg2]←GR[reg2]+sign-extend(imm5) | | 1 | 1 | 1 | × | × | × | × | |
| ADDI | imm16,reg1,reg2 | rrrrr110000RRRRR iiiiiiiiiiiiiiii | GR[reg2]←GR[reg1]+sign-extend(imm16) | | 1 | 1 | 1 | × | × | × | × | |
| AND | reg1,reg2 | rrrrr001010RRRRR | GR[reg2]←GR[reg2]AND GR[reg1] | | 1 | 1 | 1 | | 0 | × | × | |
| ANDI | imm16,reg1,reg2 | rrrrr110110RRRRR iiiiiiiiiiiiiiii | GR[reg2]←GR[reg1]AND zero-extend(imm16) | | 1 | 1 | 1 | | 0 | × | × | |
| Bcond | disp9 | ddddd1011dddcccc **Note 1** | if conditions are satisfied then PC←PC+sign-extend(disp9) | When conditions are satisfied | 2 Note 2 | 2 Note 2 | 2 Note 2 | | | | | |
| | | | | When conditions are not satisfied | 1 | 1 | 1 | | | | | |
| BSH | reg2,reg3 | rrrrr11111100000 wwwww01101000010 | GR[reg3]←GR[reg2] (23 : 16) ll GR[reg2] (31 : 24) ll GR[reg2] (7 : 0) ll GR[reg2] (15 : 8) | | 1 | 1 | 1 | × | 0 | × | × | |
| BSW | reg2,reg3 | rrrrr11111100000 wwwww01101000000 | GR[reg3]←GR[reg2] (7 : 0) ll GR[reg2] (15 : 8) ll GR[reg2] (23 : 16) ll GR[reg2] (31 : 24) | | 1 | 1 | 1 | × | 0 | × | × | |
| CALLT | imm6 | 0000001000iiiiii | CTPC←PC+2(return PC) CTPSW←PSW adr←CTBP+zero-extend(imm6 logically shift left by 1) PC←CTBP+zero-extend(Load-memory(adr,Halfword)) | | 4 | 4 | 4 | | | | | |
| CLR1 | bit#3,disp16[reg1] | 10bbb111110RRRRR dddddddddddddddd | adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,0) | | 3 Note 3 | 3 Note 3 | 3 Note 3 | | | | × | |
| | reg2,[reg1] | rrrrr111111RRRRR 0000000011100100 | adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,0) | | 3 Note 3 | 3 Note 3 | 3 Note 3 | | | | × | |
| CMOV | cccc,imm5,reg2,reg3 | rrrrr111111iiiii wwwww011000cccc0 | if conditions are satisfied then GR[reg3]←sign-extended(imm5) else GR[reg3]←GR[reg2] | | 1 | 1 | 1 | | | | | |
| | cccc,reg1,reg2,reg3 | rrrrr111111RRRR wwwww011001cccc0 | if conditions are satisfied then GR[reg3]←GR[reg1] else GR[reg3]←GR[reg2] | | 1 | 1 | 1 | | | | | |
| CMP | reg1,reg2 | rrrrr001111RRRRR | result←GR[reg2]–GR[reg1] | | 1 | 1 | 1 | × | × | × | × | |
| | imm5,reg2 | rrrrr010011iiiii | result←GR[reg2]–sign-extend(imm5) | | 1 | 1 | 1 | × | × | × | × | |
| CTRET | | 0000011111100000 0000000101000100 | PC←CTPC PSW←CTPSW | | 3 | 3 | 3 | R | R | R | R | R |
| DBRET | | 0000011111100000 0000000101000110 | PC←DBPC PSW←DBPSW | | 3 | 3 | 3 | R | R | R | R | R |

| Mnemonic | Operand | Opcode | Operation | Execution Clock | | | Flags | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | i | r | l | CY | OV | S | Z | SAT |
| DBTRAP | | 1111100001000000 | DBPC←PC+2 (restored PC)<br>DBPSW←PSW<br>PSW.NP←1<br>PSW.EP←1<br>PSW.ID←1<br>PC←00000060H | 3 | 3 | 3 | | | | | |
| DI | | 0000011111100000<br>0000000101100000 | PSW.ID←1 | 1 | 1 | 1 | | | | | |
| DISPOSE | imm5,list12 | 0000011001iiiiiL<br>LLLLLLLLLLLL00000 | sp←sp+zero-extend(imm5 logically shift left by 2)<br>GR[reg in list12]←Load-memory(sp,Word)<br>sp←sp+4<br>repeat 2 steps above until all regs in list12 is loaded | n+1<br>Note 4 | n+1<br>Note 4 | n+1<br>Note 4 | | | | | |
| | imm5,list12,[reg1] | 0000011001iiiiiL<br>LLLLLLLLLLLLRRRRR<br>Note 5 | sp←sp+zero-extend(imm5 logically shift left by 2)<br>GR[reg in list12]←Load-memory(sp,Word)<br>sp←sp+4<br>repeat 2 steps above until all regs in list12 is loaded<br>PC←GR[reg1] | n+3<br>Note 4 | n+3<br>Note 4 | n+3<br>Note 4 | | | | | |
| DIV | reg1,reg2,reg3 | rrrrr111111RRRRR<br>wwwww01011000000 | GR[reg2]←GR[reg2]÷GR[reg1]<br>GR[reg3]←GR[reg2]%GR[reg1] | 35 | 35 | 35 | | × | × | × | |
| DIVH | reg1,reg2 | rrrrr000010RRRRR | GR[reg2]←GR[reg2]÷GR[reg1]$^{\text{Note 6}}$ | 35 | 35 | 35 | | × | × | × | |
| | reg1,reg2,reg3 | rrrrr111111RRRRR<br>wwwww01010000000 | GR[reg2]←GR[reg2]÷GR[reg1]$^{\text{Note 6}}$<br>GR[reg3]←GR[reg2]%GR[reg1] | 35 | 35 | 35 | | × | × | × | |
| DIVHU | reg1,reg2,reg3 | rrrrr111111RRRRR<br>wwwww01010000010 | GR[reg2]←GR[reg2]÷GR[reg1]$^{\text{Note 6}}$<br>GR[reg3]←GR[reg2]%GR[reg1] | 34 | 34 | 34 | | × | × | × | |
| DIVU | reg1,reg2,reg3 | rrrrr111111RRRRR<br>wwwww01011000010 | GR[reg2]←GR[reg2]÷GR[reg1]<br>GR[reg3]←GR[reg2]%GR[reg1] | 34 | 34 | 34 | | × | × | × | |
| EI | | 1000011111100000<br>0000000101100000 | PSW.ID←0 | 1 | 1 | 1 | | | | | |
| HALT | | 0000011111100000<br>0000000100100000 | Stop | 1 | 1 | 1 | | | | | |
| HSW | reg2,reg3 | rrrrr11111100000<br>wwwww01101000100 | GR[reg3]←GR[reg2](15 : 0) ‖ GR[reg2] (31 : 16) | 1 | 1 | 1 | × | 0 | × | × | |
| JARL | disp22,reg2 | rrrrr11110ddddd<br>ddddddddddddddd0<br>Note 7 | GR[reg2]←PC+4<br>PC←PC+sign-extend(disp22) | 2 | 2 | 2 | | | | | |
| JMP | [reg1] | 00000000011RRRRR | PC←GR[reg1] | 3 | 3 | 3 | | | | | |
| JR | disp22 | 0000011110ddddd<br>ddddddddddddddd0<br>Note 7 | PC←PC+sign-extend(disp22) | 2 | 2 | 2 | | | | | |
| LD.B | disp16[reg1],reg2 | rrrrr111000RRRRR<br>dddddddddddddddd | adr←GR[reg1]+sign-extend(disp16)<br>GR[reg2]←sign-extend(Load-memory(adr,Byte)) | 1 | 1 | Note 11 | | | | | |
| LD.BU | disp16[reg1],reg2 | rrrrr11110bRRRRR<br>ddddddddddddddd1<br>Notes 8, 10 | adr←GR[reg1]+sign-extend(disp16)<br>GR[reg2]←zero-extend(Load-memory(adr,Byte)) | 1 | 1 | Note 11 | | | | | |

| Mnemonic | Operand | Opcode | Operation | | Execution Clock | | | Flags | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | i | r | l | CY | OV | S | Z | SAT |
| LD.H | disp16[reg1],reg2 | rrrrr111001RRRRR<br>ddddddddddddddd0<br>**Note 8** | adr←GR[reg1]+sign-extend(disp16)<br>GR[reg2]←sign-extend(Load-memory(adr,Halfword)) | | 1 | 1 | **Note 11** | | | | | |
| LDSR | reg2,regID | rrrrr111111RRRRR<br>0000000000100000<br>**Note 12** | SR[regID]←GR[reg2] | Other than regID = PSW | 1 | 1 | 1 | | | | | |
| | | | | regID = PSW | 1 | 1 | 1 | × | × | × | × | × |
| LD.HU | disp16[reg1],reg2 | rrrrr111111RRRRR<br>ddddddddddddddd1<br>**Note 8** | adr←GR[reg1]+sign-extend(disp16)<br>GR[reg2]←zero-extend(Load-memory(adr,Halfword) | | 1 | 1 | **Note 11** | | | | | |
| LD.W | disp16[reg1],reg2 | rrrrr111001RRRRR<br>ddddddddddddddd1<br>**Note 8** | adr←GR[reg1]+sign-extend(disp16)<br>GR[reg2]←Load-memory(adr,Word) | | 1 | 1 | **Note 11** | | | | | |
| MOV | reg1,reg2 | rrrrr000000RRRRR | GR[reg2]←GR[reg1] | | 1 | 1 | 1 | | | | | |
| | imm5,reg2 | rrrrr010000iiiii | GR[reg2]←sign-extend(imm5) | | 1 | 1 | 1 | | | | | |
| | imm32,reg1 | 00000110001RRRRR<br>iiiiiiiiiiiiiiii<br>IIIIIIIIIIIIIIII | GR[reg1]←imm32 | | 2 | 2 | 2 | | | | | |
| MOVEA | imm16,reg1,reg2 | rrrrr110001RRRRR<br>iiiiiiiiiiiiiiii | GR[reg2]←GR[reg1]+sign-extend(imm16) | | 1 | 1 | 1 | | | | | |
| MOVHI | imm16,reg1,reg2 | rrrrr110010RRRRR<br>iiiiiiiiiiiiiiii | GR[reg2]←GR[reg1]+(imm16 ll 0$^{16}$) | | 1 | 1 | 1 | | | | | |
| MUL | reg1,reg2,reg3 | rrrrr111111RRRRR<br>wwwww01000100000<br>**Note 14** | GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1] | | 1 | 4 | 5 | | | | | |
| | imm9,reg2,reg3 | rrrrr111111iiiii<br>wwwww01001IIII00<br>**Note 13** | GR[reg3] ll GR[reg2]←GR[reg2]xsign-extend(imm9) | | 1 | 4 | 5 | | | | | |
| MULH | reg1,reg2 | rrrrr000111RRRRR | GR[reg2]←GR[reg2]$^{Note 6}$xGR[reg1]$^{Note 6}$ | | 1 | 1 | 2 | | | | | |
| | imm5,reg2 | rrrrr010111iiiii | GR[reg2]←GR[reg2]$^{Note 6}$xsign-extend(imm5) | | 1 | 1 | 2 | | | | | |
| MULHI | imm16,reg1,reg2 | rrrrr110111RRRRR<br>iiiiiiiiiiiiiiii | GR[reg2]←GR[reg1]$^{Note 6}$ximm16 | | 1 | 1 | 2 | | | | | |
| MULU | reg1,reg2,reg3 | rrrrr111111RRRRR<br>wwwww01000100010<br>**Note 14** | GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1] | | 1 | 4 | 5 | | | | | |
| | imm9,reg2,reg3 | rrrrr111111iiiii<br>wwwww01001IIII10<br>**Note 13** | GR[reg3] ll GR[reg2]←GR[reg2]xzero-extend(imm9) | | 1 | 4 | 5 | | | | | |
| NOP | | 0000000000000000 | Pass at least one clock cycle doing nothing. | | 1 | 1 | 1 | | | | | |
| NOT | reg1,reg2 | rrrrr000001RRRRR | GR[reg2]←NOT(GR[reg1]) | | 1 | 1 | 1 | | 0 | × | × | |
| NOT1 | bit#3,disp16[reg1] | 01bbb111110RRRRR<br>dddddddddddddddd | adr←GR[reg1]+sign-extend(disp16)<br>Z flag←Not(Load-memory-bit(adr,bit#3))<br>Store-memory-bit(adr,bit#3,Z flag) | | 3<br>Note 3 | 3<br>Note 3 | 3<br>Note 3 | | | | × | |
| | reg2,[reg1] | rrrrr111111RRRRR<br>0000000011100010 | adr←GR[reg1]<br>Z flag←Not(Load-memory-bit(adr,reg2))<br>Store-memory-bit(adr,reg2,Z flag) | | 3<br>Note 3 | 3<br>Note 3 | 3<br>Note 3 | | | | × | |

(4/6)

| Mnemonic | Operand | Opcode | Operation | Execution Clock | | | Flags | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | i | r | l | CY | OV | S | Z | SAT |
| OR | reg1,reg2 | rrrrr001000RRRRR | GR[reg2]←GR[reg2]OR GR[reg1] | 1 | 1 | 1 | | 0 | × | × | |
| ORI | imm16,reg1,reg2 | rrrrr110100RRRRR<br>iiiiiiiiiiiiiiii | GR[reg2]←GR[reg1]OR zero-extend(imm16) | 1 | 1 | 1 | | 0 | × | × | |
| PREPARE | list12,imm5 | 0000011110iiiiiL<br>LLLLLLLLLLLL00001 | Store-memory(sp–4,GR[reg in list12],Word)<br>sp←sp–4<br>repeat 1 step above until all regs in list12 is stored<br>sp←sp-zero-extend(imm5) | n+1<br>Note 4 | n+1<br>Note 4 | n+1<br>Note 4 | | | | | |
| | list12,imm5,<br>sp/imm<sup>Note 15</sup> | 0000011110iiiiiL<br>LLLLLLLLLLLLff011<br>imm16/imm32<br>**Note 16** | Store-memory(sp–4,GR[reg in list12],Word)<br>sp←sp+4<br>repeat 1 step above until all regs in list12 is stored<br>sp←sp-zero-extend (imm5)<br>ep←sp/imm | n+2<br>Note 4<br>Note17 | n+2<br>Note 4<br>Note17 | n+2<br>Note 4<br>Note17 | | | | | |
| RETI | | 0000011111100000<br>0000000101000000 | if PSW.EP=1<br>then  PC   ←EIPC<br>       PSW  ←EIPSW<br>else if PSW.NP=1<br>     then   PC   ←FEPC<br>          PSW  ←FEPSW<br>     else   PC   ←EIPC<br>          PSW  ←EIPSW | 3 | 3 | 3 | R | R | R | R | R |
| SAR | reg1,reg2 | rrrrr111111RRRRR<br>0000000010100000 | GR[reg2]←GR[reg2]arithmetically shift right<br>by GR[reg1] | 1 | 1 | 1 | × | 0 | × | × | |
| | imm5,reg2 | rrrrr010101iiiii | GR[reg2]←GR[reg2]arithmetically shift right<br>by zero-extend (imm5) | 1 | 1 | 1 | × | 0 | × | × | |
| SASF | cccc,reg2 | rrrrr1111110cccc<br>0000001000000000 | if conditions are satisfied<br>then GR[reg2]←(GR[reg2]Logically shift left by 1)<br>OR 00000001H<br>else GR[reg2]←(GR[reg2]Logically shift left by 1)<br>OR 00000000H | 1 | 1 | 1 | | | | | |
| SATADD | reg1,reg2 | rrrrr000110RRRRR | GR[reg2]←saturated(GR[reg2]+GR[reg1]) | 1 | 1 | 1 | × | × | × | × | × |
| | imm5,reg2 | rrrrr010001iiiii | GR[reg2]←saturated(GR[reg2]+sign-extend(imm5) | 1 | 1 | 1 | × | × | × | × | × |
| SATSUB | reg1,reg2 | rrrrr000101RRRRR | GR[reg2]←saturated(GR[reg2]–GR[reg1]) | 1 | 1 | 1 | × | × | × | × | × |
| SATSUBI | imm16,reg1,reg2 | rrrrr110011RRRRR<br>iiiiiiiiiiiiiiii | GR[reg2]←saturated(GR[reg1]–sign-extend(imm16) | 1 | 1 | 1 | × | × | × | × | × |
| SATSUBR | reg1,reg2 | rrrrr000100RRRRR | GR[reg2]←saturated(GR[reg1]–GR[reg2]) | 1 | 1 | 1 | × | × | × | × | × |
| SETF | cccc,reg2 | rrrrr1111110cccc<br>0000000000000000 | If conditions are satisfied<br>then GR[reg2]←00000001H<br>else GR[reg2]←00000000H | 1 | 1 | 1 | | | | | |

(5/6)

| Mnemonic | Operand | Opcode | Operation | Execution Clock | | | Flags | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | i | r | l | CY | OV | S | Z | SAT |
| SET1 | bit#3,disp16[reg1] | 00bbb111110RRRRR<br>dddddddddddddddd | adr←GR[reg1]+sign-extend(disp16)<br>Z flag←Not (Load-memory-bit(adr,bit#3))<br>Store-memory-bit(adr,bit#3,1) | 3<br>Note 3 | 3<br>Note 3 | 3<br>Note 3 | | | | × | |
| | reg2,[reg1] | rrrrr111111RRRRR<br>0000000011100000 | adr←GR[reg1]<br>Z flag←Not(Load-memory-bit(adr,reg2))<br>Store-memory-bit(adr,reg2,1) | 3<br>Note 3 | 3<br>Note 3 | 3<br>Note 3 | | | | × | |
| SHL | reg1,reg2 | rrrrr111111RRRRR<br>0000000011000000 | GR[reg2]←GR[reg2] logically shift left by GR[reg1] | 1 | 1 | 1 | × | 0 | × | × | |
| | imm5,reg2 | rrrrr010110iiiii | GR[reg2]←GR[reg2] logically shift left<br>by zero-extend(imm5) | 1 | 1 | 1 | × | 0 | × | × | |
| SHR | reg1,reg2 | rrrrr111111RRRRR<br>0000000010000000 | GR[reg2]←GR[reg2] logically shift right by GR[reg1] | 1 | 1 | 1 | × | 0 | × | × | |
| | imm5,reg2 | rrrrr010100iiiii | GR[reg2]←GR[reg2] logically shift right<br>by zero-extend(imm5) | 1 | 1 | 1 | × | 0 | × | × | |
| SLD.B | disp7[ep],reg2 | rrrrr0110ddddddd | adr←ep+zero-extend(disp7)<br>GR[reg2]←sign-extend(Load-memory(adr,Byte)) | 1 | 1 | Note 9 | | | | | |
| SLD.BU | disp4[ep],reg2 | rrrrr0000110dddd<br>Note 18 | adr←ep+zero-extend(disp4)<br>GR[reg2]←zero-extend(Load-memory(adr,Byte)) | 1 | 1 | Note 9 | | | | | |
| SLD.H | disp8[ep],reg2 | rrrrr1000ddddddd<br>Note 19 | adr←ep+zero-extend(disp8)<br>GR[reg2]←sign-extend(Load-memory(adr,Halfword)) | 1 | 1 | Note 9 | | | | | |
| SLD.HU | disp5[ep],reg2 | rrrrr0000111dddd<br>Notes 18, 20 | adr←ep+zero-extend(disp5)<br>GR[reg2]←zero-extend(Load-memory(adr,Halfword)) | 1 | 1 | Note 9 | | | | | |
| SLD.W | disp8[ep],reg2 | rrrrr1010dddddd0<br>Note 21 | adr←ep+zero-extend(disp8)<br>GR[reg2]←Load-memory(adr,Word) | 1 | 1 | Note 9 | | | | | |
| SST.B | reg2,disp7[ep] | rrrrr0111ddddddd | adr←ep+zero-extend(disp7)<br>Store-memory(adr,GR[reg2],Byte) | 1 | 1 | 1 | | | | | |
| SST.H | reg2,disp8[ep] | rrrrr1001ddddddd<br>Note 19 | adr←ep+zero-extend(disp8)<br>Store-memory(adr,GR[reg2],Halfword) | 1 | 1 | 1 | | | | | |
| SST.W | reg2,disp8[ep] | rrrrr1010dddddd1<br>Note 21 | adr←ep+zero-extend(disp8)<br>Store-memory(adr,GR[reg2],Word) | 1 | 1 | 1 | | | | | |
| ST.B | reg2,disp16[reg1] | rrrrr111010RRRRR<br>dddddddddddddddd | adr←GR[reg1]+sign-extend(disp16)<br>Store-memory(adr,GR[reg2],Byte) | 1 | 1 | 1 | | | | | |
| ST.H | reg2,disp16[reg1] | rrrrr111011RRRRR<br>ddddddddddddddd0<br>Note 8 | adr←GR[reg1]+sign-extend(disp16)<br>Store-memory (adr,GR[reg2], Halfword) | 1 | 1 | 1 | | | | | |
| ST.W | reg2,disp16[reg1] | rrrrr111011RRRRR<br>ddddddddddddddd1<br>Note 8 | adr←GR[reg1]+sign-extend(disp16)<br>Store-memory (adr,GR[reg2], Word) | 1 | 1 | 1 | | | | | |
| STSR | regID,reg2 | rrrrr111111RRRRR<br>0000000001000000 | GR[reg2]←SR[regID] | 1 | 1 | 1 | | | | | |

(6/6)

| Mnemonic | Operand | Opcode | Operation | Execution Clock | | | Flags | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | i | r | l | CY | OV | S | Z | SAT |
| SUB | reg1,reg2 | rrrrr001101RRRRR | GR[reg2]←GR[reg2]–GR[reg1] | 1 | 1 | 1 | × | × | × | × | |
| SUBR | reg1,reg2 | rrrrr001100RRRRR | GR[reg2]←GR[reg1]–GR[reg2] | 1 | 1 | 1 | × | × | × | × | |
| SWITCH | reg1 | 00000000010RRRRR | adr←(PC+2) + (GR [reg1] logically shift left by 1) PC←(PC+2) + (sign-extend (Load-memory (adr,Halfword)) logically shift left by 1 | 5 | 5 | 5 | | | | | |
| SXB | reg1 | 00000000101RRRRR | GR[reg1]←sign-extend (GR[reg1] (7 : 0)) | 1 | 1 | 1 | | | | | |
| SXH | reg1 | 00000000111RRRRR | GR[reg1]←sign-extend (GR[reg1] (15 : 0)) | 1 | 1 | 1 | | | | | |
| TRAP | vector | 00000111111iiiii 0000000100000000 | EIPC ←PC+4 (Restored PC) EIPSW ←PSW ECR.EICC ←Interrupt code PSW.EP ←1 PSW.ID ←1 PC ←00000040H (when vector is 00H to 0FH) 00000050H (when vector is 10H to 1FH) | 3 | 3 | 3 | | | | | |
| TST | reg1,reg2 | rrrrr001011RRRRR | result←GR[reg2] AND GR[reg1] | 1 | 1 | 1 | | 0 | × | × | |
| TST1 | bit#3,disp16[reg1] | 11bbb111110RRRRR dddddddddddddddd | adr←GR[reg1]+sign-extend(disp16) Z flag←Not (Load-memory-bit (adr,bit#3)) | 3 Note 3 | 3 Note 3 | 3 Note 3 | | | | × | |
| | reg2, [reg1] | rrrrr111111RRRRR 0000000011100110 | adr←GR[reg1] Z flag←Not (Load-memory-bit (adr,reg2)) | 3 Note 3 | 3 Note 3 | 3 Note 3 | | | | × | |
| XOR | reg1,reg2 | rrrrr001001RRRRR | GR[reg2]←GR[reg2] XOR GR[reg1] | 1 | 1 | 1 | | 0 | × | × | |
| XORI | imm16,reg1,reg2 | rrrrr110101RRRRR iiiiiiiiiiiiiiii | GR[reg2]←GR[reg1] XOR zero-extend (imm16) | 1 | 1 | 1 | | 0 | × | × | |
| ZXB | reg1 | 00000000100RRRRR | GR[reg1]←zero-extend (GR[reg1] (7 : 0)) | 1 | 1 | 1 | | | | | |
| ZXH | reg1 | 00000000110RRRRR | GR[reg1]←zero-extend (GR[reg1] (15 : 0)) | 1 | 1 | 1 | | | | | |

**Notes 1.** dddddddd: Higher 8 bits of disp9.

**2.** 3 if there is an instruction that rewrites the contents of the PSW immediately before.

**3.** If there is no wait state (3 + the number of read access wait states).

**4.** n is the total number of list12 load registers. (According to the number of wait states. Also, if there are no wait states, n is the total number of list12 registers. If n = 0, same operation as when n = 1)

**5.** RRRRR: other than 00000.

**6.** The lower halfword data only are valid.

**7.** dddddddddddddddddddd: The higher 21 bits of disp22.

**8.** ddddddddddddddd: The higher 15 bits of disp16.

**9.** According to the number of wait states (1 if there are no wait states).

**10.** b: bit 0 of disp16.

**11.** According to the number of wait states (2 if there are no wait states).

**Notes 12.** In this instruction, for convenience of mnemonic description, the source register is made reg2, but the reg1 field is used in the opcode. Therefore, the meaning of register specification in the mnemonic description and in the opcode differs from other instructions.

        rrrrr      = regID specification

        RRRRR    = reg2 specification

**13.** iiiii:    Lower 5 bits of imm9.

     IIII:      Higher 4 bits of imm9.

**14.** Do not specify the same register for general-purpose registers reg1 and reg3.

**15.** sp/imm: Specified by bits 19 and 20 of the sub-opcode.

**16.** ff =    00: Load sp in ep.

            01: Load sign-expanded 16-bit immediate data (bits 47 to 32) in ep.

            10: Load 16-bit logically-left-shifted 16-bit immediate data (bits 47 to 32) in ep.

            11: Load 32-bit immediate data (bits 63 to 32) in ep.

**17.** If imm = imm32, n + 3 clocks.

**18.** rrrrr: Other than 00000.

**19.** ddddddd: Higher 7 bits of disp8.

**20.** dddd: Higher 4 bits of disp5.

**21.** dddddd: Higher 6 bits of disp8.

## APPENDIX E REVISION HISTORY

### E.1 Major Revisions in This Edition

| Page | Description |
|---|---|
| p.34 | Modification of **2.1 List of Pin Functions (1) Port pins** |
| pp.38 to 46 | Modification of **2.1 List of Pin Functions (2) Non-port Pins** |
| p.98 | Addition of Caution to **Figure 4-1. Port Configuration Diagram (V850ES/JG3-U)** |
| p.98 | Addition of Caution to **Figure 4-2. Port Configuration Diagram (V850ES/JH3-U)** |
| p.817 | Modification of **Figure 19-3. UARTC1 and I$^2$C02 Mode Switch Settings** |

## E.2   Revision History of Preceding Editions

Here is the revision history of the preceding editions.  Chapter indicates the chapter of each edition.

(1/2)

| Edition | Description | Chapter |
|---|---|---|
| 2nd | Addition of **Note** of **5.5.1 (1) Data wait control register 0 (DWC0)** | **CHAPTER 5  BUS CONTROL  FUNCTION** |
| | Addition of **Note** of **5.5.4 (1) Address wait control register (AWC)** | |
| | Addition of **Note** of **5.6 (1) Bus cycle control register (BCC)** | |
| | Addition of **Caution** of **20.1 Overview** | **CHAPTER 20  USB FUNCTION CONTROLLER  (USBF)** |
| | Change of **Figure 20-3. Example of USB Function Controller Connection** | |
| | Change of **20.4 (2) Stopping the USB clock** | |
| | Change of **20.6.1 (2) USB function control register (UFCKMSK)** | |
| | **20.6.3 (26) UF0 INT & DMARQ register (UF0IDR)** Change of description of MODE1 bit and MODE0 bit | |
| | Change of **20.6.8 (1) UF0 EP1 bulk-in transfer data register (UF0EP1BI)** | |
| | Change of **20.6.8 (2) UF0 EP3 bulk-in transfer data register (UF0EP3BI)** | |
| | Change of **20.6.9 (1) UF0 EP2 bulk-out transfer data register (UF0EP2BO)** | |
| | Change of **20.6.9 (2) UF0 EP4 bulk-out transfer data register (UF0EP4BO)** | |
| | Addition of **20.9.6 (1) Initial settings for a bulk transfer (OUT: EP2, EP4)** | |
| | Addition of **20.9.7 (1) Initial settings for a bulk transfer (IN: EP1, EP3)** | |
| | Change of **Table 31-6. Wiring of V850ES/JG3-U Flash Writing Adapters** | **CHAPTER 31  FLASH MEMORY** |
| | Change of **Table 31-7. Wiring of V850ES/JH3-U Flash Writing Adapters** | |
| | Addition of **31.6 Creating ROM code to place order for previously written product** | |
| | Change of **32.2.3 (5) Securement of communication serial interface • Port registers when CSIF0 is used (a)** | **CHAPTER 32  ON-CHIP  DEBUG FUNCTION** |
| | Change of **32.2.3 (5) Securement of communication serial interface • Port registers when CSIF3 is used (a)** | |
| | Change of **33.1 Absolute Maximum Ratings** | **CHAPTER 33 ELECTRICAL SPECIFICATIONS** |
| | Addition of **33.4.1 (1) KYOCERA KINSEKI CORPORATION: Crystal resonator** | |
| | Addition of **33.4.1 (2) KYOCERA CORPORATION: Ceramic resonator** | |
| | Addition of **33.4.1 (3) Toyama Murata Mfg. Co. Ltd.: Ceramic resonator** | |
| | Addition of **33.4.2 (1) Seiko Instruments Inc.: Crystal resonator** | |
| | Change of **33.5.1 I/O level** | |
| | Change of **33.5.2 Supply current** | |
| | Change of Data retention current (MAX. value) of **33.6 (1) In STOP mode** | |
| | Change of Number of rewrites of **33.9 (1) Basic characteristics** | |
| | Addition of **CHAPTER 35 RECOMMENDED SOLDERING CONDITIONS** | **CHAPTER 35 RECOMMENDED SOLDERING CONDITIONS** |
| | Addition of **APPENDIX E REVISION HISTORY** | **APPENDIX  E REVISION HISTORY** |

(2/2)

| Edition | Description | Chapter |
|---|---|---|
| 3rd | Modification of **4.3.3 (6)Port 2 alternate function specifications** | **CHAPTER 4 PORT FUNCTION** |
| | Modification of **4.3.9 Port 9 function control register (PFC9)** | |
| | Modification of **4.3.9 Port 9 function control expansion register (PFCE9)** | |
| | Modification of **Table 4-20. Using Port Pin as Alternate-Function Pin** | **CHAPTER 12 REAL-TIME COUNTER** |
| | Modification of **Figure 12-1. Block Diagram of Real-Time Counter** | |
| | Modification of **12.2.2 (3)** | |
| | Modification of **Figure 12-10. Watch Error Correction Example** | |
| | Modification of **18.4 (2) CSIFn control register 1 (CFnCTL1) Note** | **CHAPTER 18 3-WIRE VARIABLE-LENGTH SERIAL I/O (CSIF)** |
| | Modification of **Figure 21-30. Example of Suspend/Resume Processing** | **CHAPTER 21 USB HOST CONTROLLER (USBH)** |
| | Addition of Caution to **(6) DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)** | **CHAPTER 22 DMA FUNCTION (DMA CONTROLLER)** |
| | Modification of **Table 31-2. Basic Functions** | **CHAPTER 31 FLASH MEMORY** |
| | Modification of **Table 31-8. Flash Memory Control Commands** | |
| | Modification of **33.9 (1) Basic characteristics** | **CHAPTER 33 ELECTRICAL SPECIFICATIONS** |

# RENESAS

**SALES OFFICES**　　　　　Renesas Electronics Corporation　　　http://www.renesas.com

# V850ES/JG3-U，V850ES/JH3-U

RENESAS

Renesas Electronics Corporation

# Mouser Electronics

Authorized Distributor


Click to View Pricing, Inventory, Delivery & Lifecycle Information:


[Renesas Electronics](#):

[UPD70F3764GC-UEU-AX](#)  [UPD70F3763GC-UEU-AX](#)  [UPD70F3769GF-GAT-AX](#)  [UPD70F3768GF-GAT-AX](#)