# FM0+ Family

## 32-BIT MICROCONTROLLER

***PERIPHERAL MANUAL***

For the information for microcontroller supports, see the following web site.

**http://www.spansion.com/support/microcontrollers/**

ARM™

**Publication Number S6E1A1_MN710-00001 Revision 1.0 Issue Date January 17, 2014**

# Preface

Thank you for your continued use of Spansion products.
Read this manual and "Data Sheet" thoroughly before using products in this family.

### Purpose of this manual and intended readers

This manual explains the functions and operations of this family and describes how it is used. The manual is intended for engineers engaged in the actual development of products using this family.
For the descriptions on Analog macro, Timer, and Communication Macro, see the respective separate peripheral manual.

***Note:***

- *This manual explains the configuration and operation of the peripheral functions, but does not cover the specifics of each device in the series.*
  *Users should refer to the respective data sheets of devices for device-specific details.*

### Trademark

ARM and Cortex are the trademarks of ARM Limited in the EU and other countries.

The company names and brand names herein are the trademarks or registered trademarks of their respective owners.

### Sample programs and development environment

Spansion offers sample programs free of charge for using the peripheral functions of the FM0+ family. Spansion also makes available descriptions of the development environment required for this series. Feel free to use them to verify the operational specifications and usage of this Spansion microcontroller.

### Microcontroller support information:

**http://www.spansion.com/support/microcontrollers/**

***Note:***

- *Note that the sample programs are subject to change without notice. Since they are offered as a way to demonstrate standard operations and usage, evaluate them sufficiently before running them on your system.*
  *Spansion assumes no responsibility for any damage that may occur as a result of using a sample program.*

### Overall Organization of This Manual

Peripheral Manual Timer Part has 14 chapters and APPENDIXES as shown below.

CHAPTER 1: System Overview
CHAPTER 2-1: Clock
CHAPTER 2-2: Peripheral Clock Gating
CHAPTER 2-3: High-Speed CR Trimming
CHAPTER 2-4: Low-Speed CR Prescaler
CHAPTER 3: Clock Supervisor
CHAPTER 4: Resets
CHAPTER 5: Low-voltage Detection
CHAPTER 6: Low Power Consumption Mode
CHAPTER 7-1: Interrupts
CHAPTER 7-2: Interrupts (A)
CHAPTER 7-3: Interrupts (B)
CHAPTER 8: External Interrupt and NMI Control Sections

# Related Manuals

The manuals related to this family are listed below. See the manual appropriate to the applicable conditions.

The contents of these manuals are subject to change without notice. Contact us to check the latest versions available.

### Peripheral Manual

- FM0+ Family PERIPHERAL MANUAL (this manual)
(Called "PERIPHERAL MANUAL" hereafter)
- FM0+ Family PERIPHERAL MANUAL Timer Part
(Called "Timer Part" hereafter)
- FM0+ Family PERIPHERAL MANUAL Analog Macro Part
(Called "Analog Macro Part" hereafter)
- FM0+ Family PERIPHERAL MANUAL Communication Macro Part
(Called "Communication Macro Part" hereafter)

### Data sheet

For details about device-specific, electrical characteristics, package dimensions, ordering information etc., see the following document.

- 32-bit Microcontroller FM0+ Family DATA SHEET

***Note:***

- *The data sheets for each series are provided.*
  *See the appropriate data sheet for the series that you are using.*

### CPU Programming manual

For details about ARM Cortex-M0+ core, see the following documents that can be obtained from http://www.arm.com/.

- Cortex-M0+ Technical Reference Manual
- ARMv6-M Architecture Application Level Reference Manual

### Flash Programming manual

For details about the functions and operations of the built-in flash memory, see the following document.

- FM0+ Family FLASH PROGRAMMING MANUAL

***Note:***

- *The Flash Programming manuals for each series are provided.*
  *See the appropriate Flash Programming manual for the series that you are using.*

# How to Use This Manual

### Finding a function

The following methods can be used to search for the explanation of a desired function in this manual:

■ Search from the table of the contents

The table of the contents lists the manual contents in the order of description.

■ Search from the register

The address where each register is located is not described in the text. To verify the address of a register, see "A. Register Map" in "APPENDIXES".

### About the chapters

Basically, this manual explains 1 peripheral function per chapter.

### Terminology

This manual uses the following terminology.

| Term | Explanation |
|---|---|
| Word | Indicates access in units of 32 bits. |
| Half word | Indicates access in units of 16 bits. |
| Byte | Indicates access in units of 8 bits. |

### Notations

■ The notations in bit configuration of the register explanation of this manual are written as follows.

- bit : bit number
- Field : bit field name
- Attribute : Attributes for read and write of each bit
  - R : Read only
  - W : Write only
  - R/W : Readable/Writable
  - - : Undefined
- Initial value : Initial value of the register after reset
  - 0 : Initial value is "0"
  - 1 : Initial value is "1"
  - X : Initial value is undefined

■ The multiple bits are written as follows in this manual.

Example : bit7:0 indicates the bits from bit7 to bit0

■ The values such as for addresses are written as follows in this manual.

- Hexadecimal number : "0x" is attached in the beginning of a value as a prefix (example : 0xFFFF)
- Binary number : "0b" is attached in the beginning of a value as a prefix (example: 0b1111)
- Decimal number : Written using numbers only (example : 1000)

### The target products in this manual

- In this manual, the products are classified into the following groups and are described as follows.
  For the descriptions such as "TYPE1", see the relevant items of the target FM0+ product in the list below.

**Table 1 FM0+ family TYPE1 Product list**

| TYPE | Flash memory size | |
|---|---|---|
| | 88 Kbytes | 56 Kbytes |
| TYPE1 | S6E1A12B0A<br>S6E1A12C0A | S6E1A11B0A<br>S6E1A11C0A |

# Table of Contents

# Figures

# Tables

# CHAPTER1: System Overview

**This chapter explains the system overview.**



CODE: 9AFSYSTEM-E01.0

# 1. Bus Architecture

This section explains the bus architecture.

For this series bus, AHB Bus Matrix circuit actualizes a multi-layer bus. Master and slave architectures are shown below:

- Master
  - Cortex-M0+ CPU (AHB-Lite)
  - DMAC

- Slave
  - On-chip Flash Memory
  - On-chip SRAM (MTB sharable)
  - AHB-AHB Bus Bridge
  - AHB-APB Bus Bridge (APB0, APB1)

See Figure 1-1 for the bus block diagram.

### Features

- RAM Architecture

The user SRAM area can be shared with the MTB SRAM area. The two areas are divided according to user configuration.

- APB Extension Bus

APB1 Peripheral Bus is an APB extension bus to which the following functions are originally added based on AMBA3.0. (APB0 is not included.)

- Supporting Halfword (16 bits) and Byte(8 bits) Accesses

  For supported registers, halfword access and byte access are enabled.
  See "A. Register Map" in "APPENDIXES" for the supported registers.

- Adding Read-Modify-Write (RMW) Signal

  HMASTLOCK signal in bit-band operations is used to generate.
  RMW signal is a signal added to prevent that an unrelated flag is cleared mistakenly in read-modify-write process of bit-band operations.
  The corresponding flag reads "1" in read during the read-modify-write process and is designed to ignore "1" write.
  This prevents any unrelated flag from being mistakenly cleared in the next write when the flag is set immediately after the read in the sequence from read to modify to write.
  For the corresponding flags and registers, it is described that "regardless of bit values, "1" can be read in "Read-Modify-Write".

***Notes:***

- *No bit-band operation must be performed on a register for which RMW is prohibited.*
- *When Read-Modify-Write process is performed over the software without bit-band operation, RMW signal is not output.*
  *Therefore, in this case, the flag value can be read in read operation although a register supports RMW process, and it is necessary not to be cleared an unrelated flag mistakenly in write operation.*
- *It is compatible with Cortex M3 bit band operations, see the "Cortex M3 Technical Reference Manual" for detail information.*

- Priority Level

  A priority of the bus right is determined in round-robin scheme.

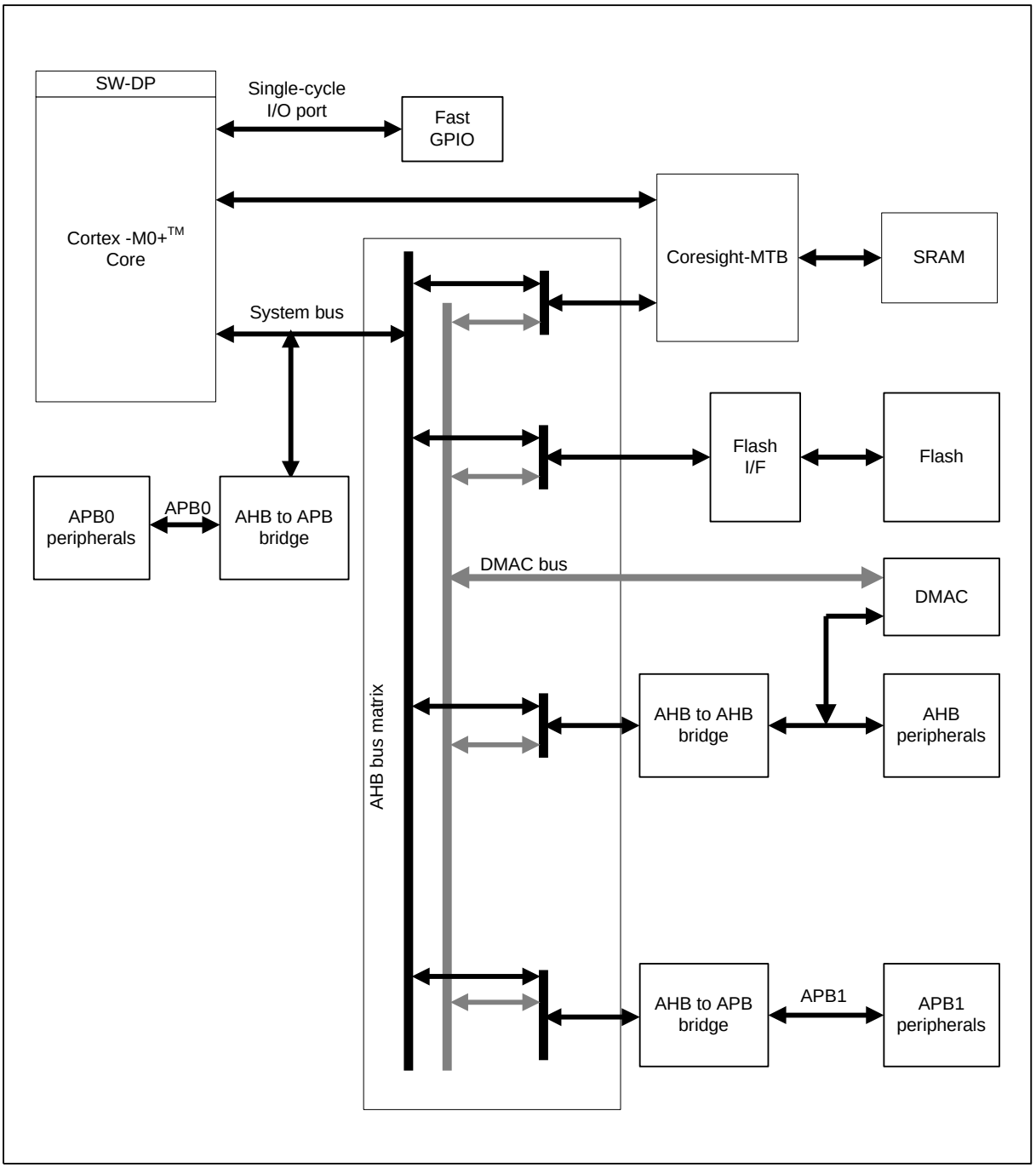
- Endian

  This family uses little endian byte order.

## 1.1 Bus Block Diagram

Figure 1-1 illustrates the bus block diagram.

**Figure 1-1 Bus Block Diagram**




***Note:***

- *There are some areas which no DMAC transfer can be performed. For details, see the notes in "1.3 Memory Map" and DMAC Transfer column in Table 1-1.*

## 1.2 Memory Architecture

This section shows the memory architecture.

For this family, 4 GB address space is available.

A Flash memory area up to 1 MB in size and an SRAM area up to 512 KB in size are defined.

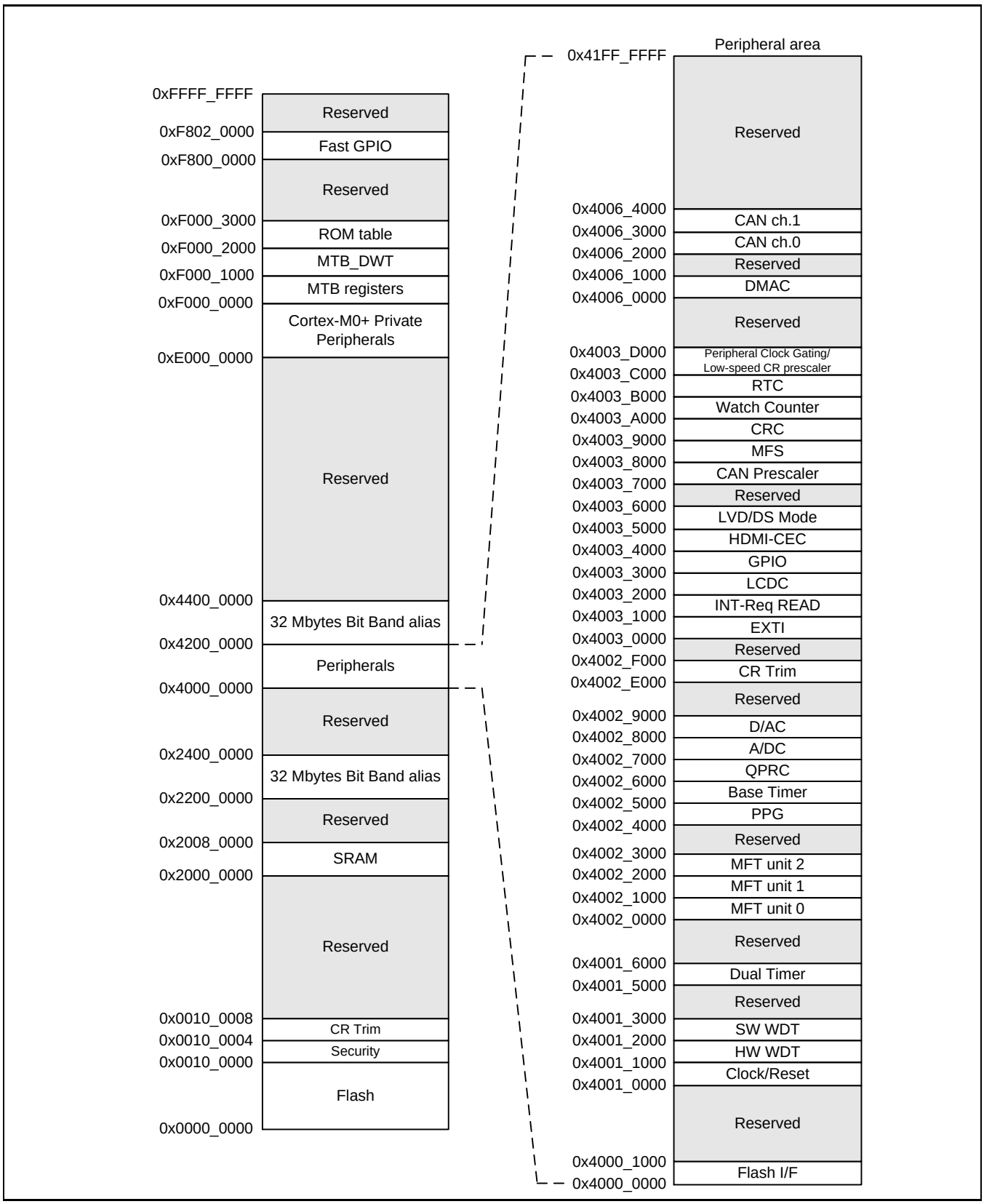
Section "1.3 Memory Map" illustrates the memory map, and Section "1.4 Peripheral Address Map" illustrates the peripheral address map.

For the details of Cortex-M0+ private peripheral shown in Figure 1-2, see "Cortex-M0+ Technical Reference Manual".

## 1.3 Memory Map

Figure 1-2 illustrates the memory map.

**Figure 1-2 Memory Map**



***Notes:***

- *Do not access to a reserved area.*
- *For details of the flash memory, see "FLASH PROGRAMMING MANUAL" of the product used.*
- *Do not perform DMAC transfer on a following area.*
  - *Bit Band Alias area*
  - *Fast GPIO*
  - *ROM table*
  - *MTB_DWT*
  - *MTB registers(SFR)*
  - *Coretex-M0+ Private Peripherals*

## 1.4 Peripheral Address Map

Table 1-1 shows the peripheral address map.

**Table 1-1 Peripheral Address Map**

| Start Address | End Address | Bus | DMAC Transfer | Peripheral | Register Map | CHAPTER |
|---|---|---|---|---|---|---|
| 0x4000_0000 | 0x4000_0FFF | AHB | Disabled | FLASH IF Register / Unique ID Register | FLASH_IF/Unique ID | * Chapter15 |
| 0x4000_1000 | 0x4000_FFFF | | | Reserved | - | - |
| 0x4001_0000 | 0x4001_0FFF | APB0 | Disabled | Clock and Reset Control | Clock / Reset | Chapter 2-1 Chapter 3 Chapter 4 Chapter 6 |
| 0x4001_1000 | 0x4001_1FFF | | | Hardware Watchdog Timer | HWWDT | Chapter 1 in Timer Part |
| 0x4001_2000 | 0x4001_2FFF | | | Software Watchdog Timer | SWWDT | |
| 0x4001_3000 | 0x4001_4FFF | | | Reserved | - | - |
| 0x4001_5000 | 0x4001_5FFF | | | Dual Timer | Dual_ Timer | Chapter 2 in Timer Part |
| 0x4001_6000 | 0x4001_FFFF | | | Reserved | - | - |
| 0x4002_0000 | 0x4002_0FFF | APB1 | Enabled | Multi-function Timer unit 0 | MFT | Chapter 6 in Timer Part |
| 0x4002_1000 | 0x4002_1FFF | | | Multi-function Timer unit 1 | MFT | |
| 0x4002_2000 | 0x4002_2FFF | | | Multi-function Timer unit 2 | MFT | |
| 0x4002_3000 | 0x4002_3FFF | | | Reserved | - | - |
| 0x4002_4000 | 0x4002_4FFF | | | PPG | PPG | Chapter 7-2 in Timer Part |
| 0x4002_5000 | 0x4002_5FFF | | | Base Timer | Base Timer/ Base Timer Selector | Chapter 5-1 Chapter 5-2 in Timer Part |
| 0x4002_6000 | 0x4002_6FFF | | | QPRC | QPRC | Chapter 8-1 Chapter 8-2 in Timer Part |
| 0x4002_7000 | 0x4002_7FFF | | | A/D Converter | A/DC | Chapter 1-2 Chapter 1-3 in Analog Macro Part |
| 0x4002_8000 | 0x4002_8FFF | | | D/A Converter | D/AC | Chapter 2 in Analog Macro Part |
| 0x4002_9000 | 0x4002_DFFF | | | Reserved | - | - |
| 0x4002_E000 | 0x4002_EFFF | | | High speed CR trimming | CR Trim | Chapter 2-3 |
| 0x4002_F000 | 0x4002_FFFF | | | Reserved | - | - |

| Start Address | End Address | Bus | DMAC Transfer | Peripheral | Register Map | CHAPTER |
|---|---|---|---|---|---|---|
| 0x4003_0000 | 0x4003_0FFF | APB1 | Enabled | External Interrupt | EXTI | Chapter 8 |
| 0x4003_1000 | 0x4003_1FFF | | | Interrupt Source Check Register | INT-Req READ | Chapter 7 |
| 0x4003_2000 | 0x4003_2FFF | | | LCDC | LCDC | Chapter 3 in Analog Macro Part |
| 0x4003_3000 | 0x4003_3FFF | | | GPIO | GPIO | Chapter 10 |
| 0x4003_4000 | 0x4003_4FFF | | | HDMI-CEC/Remote Control Reception | RCEC | Chapter 3 in Communication Macro Part |
| 0x4003_5000 | 0x4003_50FF | | | Low Voltage Detection | LVD | Chapter 5 |
| 0x4003_5100 | 0x4003_5FFF | | | Deep standby control block | DS_Mode | Chapter 6 |
| 0x4003_6000 | 0x4003_6FFF | | | Reserved | - | - |
| 0x4003_7000 | 0x4003_7FFF | | | CAN Prescaler | CAN_ Prescaler | Chapter 2-1 in Communication Macro Part |
| 0x4003_8000 | 0x4003_8FFF | | | Multi-function serial | MFS | Chapter 1-2<br>Chapter 1-3<br>Chapter 1-4<br>Chapter 1-5 in Communication Macro Part |
| 0x4003_9000 | 0x4003_9FFF | | | CRC | CRC | Chapter 11 |
| 0x4003_A000 | 0x4003_AFFF | | | Watch counter | Watch Counter | Chapter 3-1<br>Chapter 3-2 in Timer Part |
| 0x4003_B000 | 0x4003_BFFF | | | Real time clock | RTC | Chapter 4 in Timer Part |
| 0x4003_C000 | 0x4003_CFFF | | | Peripheral Clock Gating /Low Speed CR Prescaler | Peripheral Clock Gating | Chapter 2-2<br>Chapter 2-4 |
| 0x4003_D000 | 0x4003_FFFF | | | Reserved | - | - |
| 0x4004_0000 | 0x4005_FFFF | AHB | Enabled | Reserved | - | - |
| 0x4006_0000 | 0x4006_0FFF | | | DMAC | DMAC | Chapter 9 |
| 0x4006_1000 | 0x4006_1FFF | | | Reserved | - | - |
| 0x4006_2000 | 0x4006_2FFF | | | CAN ch.0 | CAN | Chapter 2-2 in Communication Macro Part |
| 0x4006_3000 | 0x4006_3FFF | | | CAN ch.1 | CAN | |
| 0x4006_4000 | 0x41FF_FFFF | | | Reserved | - | - |

*: For the details of "FLASH IF Register", see "FLASH PROGRAMMING MANUAL" of the product used.

# 2. Cortex-M0+ Architecture

This section explains the core architecture used in this family.

The Cortex-M0+ core block architecture* used in this family is as follows:

- Cortex-M0+ Core
- NVIC

- DWT
- MTB
- SW-DP
- ROM Table
- Single-cycle I/O port

*: The architecture varies depending on the products. For details, see "Data Sheet" of the product used.

### Cortex-M0+ Core

This family is equipped with a highly energy-efficient 32-bit processor core (ARM Cortex-M0+ core).
This peripheral manual does not describe the details of Cortex-M0+ core.
For the details, see "Cortex-M0+ Technical Reference Manual".

- Cortex-M0+ core version

For the version of Cortex-M0+ core, see "Data Sheet" of the product used.

### NVIC (Nested Vectored Interrupt Controller)

For this family, one NMI (non-maskable interrupt) and maximum 32 peripheral interrupts (IRQ0 to IRQ31)*[1] can be used.
In addition, the interrupt priority register (from 0xE000E400) has 2 bits, and 4 interrupt priority levels can be configured.

For the details of peripheral interrupts, see the chapter of the target "Interrupts" after check the product currently used with "Configuration of interrupts", and for NMI operations, see also another chapter "External Interrupt and NMI Control Block".

NMI pin is assigned for a combined use with a general-purpose port. Its initial value after a reset release is set to the general-purpose port, and NMI input is masked.
When NMI is used, enable NMI in the port setting.
For the details, see another chapter "I/O Port".

*1: "Cortex-M0+ Technical Reference Manual" defines an exception type: IRQ as an external interrupt. In this peripheral manual, to distinguish from an interrupt by an external pin "External Interrupt and NMI Control Block", the exception type: IRQ is indicated as a peripheral interrupt.

■ SysTick Timer

SysTick Timer is a system timer for OS task management integrated into NVIC.

This family generates STCLK through dividing HCLK by eight and sets the values of SysTick Calibration Value Register (Address: 0xE000E01C) as shown below:

| | | |
|---|---|---|
| bit31 | : | NOREF = 0 |
| bit30 | : | SKEW = 1 |
| bit23:0 | : | TENMS = 0x00C350 (50000)*1 |

*1: TENMS value is set to a value which becomes 10 ms when 1/8 clock of HCLK is input to STCLK and that HCLK is in 40 MHz (5 MHz in 1/8 case).
The value of TENMS is not always 10ms because HCLK can be changed to another frequency in the clock control block. Therefore, it is required to calculate an appropriate interrupt timing according to HCLK frequency.

**DWT (Data Watchpoint & Trace Unit)**

This family is equipped with DWT to use as the debug function.
DWT contains four comparators, and each comparator can be set as a hardware watchpoint.

**BPU (Breakpoint Unit)**

BPU provides support for breakpoint functionality on instruction fetches.

**MTB (Micro Trace Buffer)**

This family is equipped with a Cortex-M0+ optional component MTB to support instruction trace.

**SW-DP**

This family is equipped with SW-DP to support the serial wire protocol.

**ROM Table**

ROM table provides the address information of a debug component to an external debug tool.

**Single-cycle I/O port**

This family is equipped with Single-cycle I/O port to use as the high speed access to tightly-coupled peripherals.

# 3. Mode

This section explains the function of operating modes.

In this family, the following operating modes can be used:

■ User Mode
Internal ROM (Flash) Startup: CPU obtains a reset vector from Flash memory and starts operations.

■ Serial Writer Mode
Flash serial write is enabled.
*: For the details of this mode, see "FLASH PROGRAMMING MANUAL" of the product used.

Operating modes are determined after a release of respective power-on reset, low voltage detection reset, and INITX pin input reset.

*: For the details of power consumption control and clock selection modes, see other chapters "Low Power Consumption Mode" and "Clock".

### How to Set Operating Mode

Operating modes are configured by MD pin (MD0) input.

| MD Pins | Operating Mode |
|---|---|
| MD0 | |
| 0 | User Mode Internal ROM(Flash) Startup |
| 1 | Serial Writer Mode |

### Startup Sequence

Processes to determine operating modes in the startup sequence are shown below:

1. MD Pin Sampling
2. Determining Operating Mode and Retaining Mode Data

The descriptions of these processes are as follows:

1. MD Pin Sampling

Operating mode is configured by MD pin input (MD0). This input is sampled by power-on reset, low-voltage detection reset, and INITX pin input reset.
Determine the MD0 pin input before a reset that is the sampling factor is released.

2. Determining Operating Mode and Retaining Mode Data

MD0, which is sampled by a reset, is retained until another reset is input again.
Operating modes are determined by the retained MD0. Therefore, even MD0 is changed after a reset is released, it does not affect an operating mode.

# CHAPTER2-1: Clock

**This chapter explains the operating clock.**



CODE: 9AFCLOCK-E01.0

# 1. Overview

This section provides an overview of the clock generation unit.

The clock generation unit generates various types of clocks used to operate the MCU.

Source clock is the generic name for external and internal oscillation clocks of this MCU.

The following five types of clocks are source clocks:

- Main clock (CLKMO)
- Sub clock (CLKSO)
- High-speed CR clock (CLKHC)
- Low-speed CR clock (CLKLC)
- Main PLL clock (CLKPLL)

Select one from the source clocks. In this chapter, the selected clock is referred to as the master clock. The master clock is a source of internal bus clocks used to operate this MCU.

Dividing the master clock frequency can generate a base clock. In addition, dividing the base clock can generate each bus clock.

In this chapter, the base clock and bus clocks are referred to as internal bus clocks. The following three types of clocks are internal bus clocks:

- Base clock (HCLK/FCLK)
- APB0 bus clock (PCLK0)
- APB1 bus clock (PCLK1)

In addition to source clocks, the master clock, and internal bus clocks, the following clocks are provided:

- CAN prescaler clock
- Software watchdog timer count clock

The following shows the features of the clock generation unit.

- It can set the oscillation stabilization wait time of the main clock (CLKMO).
- It can set the interrupt which generates at completing the oscillation stabilization wait time of the main clock (CLKMO).
- It can set the oscillation stabilization wait time of the sub clock (CLKSO).
- It can set the interrupt which generates at completing the oscillation stabilization wait time of the sub clock (CLKSO).
- It can set the oscillation stabilization wait time of the main PLL clock (CLKPLL).
- It can set the interrupt which generates at completing the oscillation stabilization wait time of the main PLL clock (CLKPLL).
- It can set the PLL multiplication ratio.
- It can select the master clock.
- It can set the frequency division ratio of each internal bus clock frequency.
- It can select run or stop of the APB1 bus clock.
- It can set the frequency division ratio of the software watchdog timer count clock frequency.
- It can set run/stop of the software watchdog timer count clock.
- It can set the watchdog timer count operation in debug mode.
- It includes registers for enabling clock-related interrupts, checking interrupt status, and clearing interrupt factors.

# 2. Configuration

This section explains configuration of the clock generation unit.

### Source clock

Source clock is the generic name for external and internal oscillation clocks of this MCU. The following five types of clocks are source clocks:

- Main clock (CLKMO)

CLKMO is generated by connecting a crystal oscillator etc. to the main clock oscillation pins (X0, X1), or input using an external clock.

- Sub clock (CLKSO)

CLKSO is generated by connecting a crystal oscillator etc. to the sub clock oscillator pins (X0A, X1A), or input using an external clock.

- High-speed CR clock (CLKHC)

CLKHC is an output clock for the high-speed CR oscillator.

- Low-speed CR clock (CLKLC)

CLKLC is an output clock for the low-speed CR oscillator.

(Notes) The low-speed CR clock is a clock after a prerscaler.
For details on the low-speed CR clock prescaler, see Chapter "Low-speed CR Clock Prescaler".

- Main PLL clock (CLKPLL)

CLKPLL is generated by multiplying the main oscillation clock or high-speed CR clock using the PLL Clock Multiplication Circuit (PLL Oscillation Circuit).

### Master clock

The signal selected from source clocks are referred to as the master clock.
The master clock is a source for all bus clocks.

Note: See "1. Notes when high-speed CR is used for the master clock" in "B. List of Notes" when you use the following clock for the master clock.

- High-speed CR clock
- Main PLL clock (When selecting high-speed CR clock for the input clock of PLL)
- The master clock value should not be larger than the maximum value in "Internal operating clock frequency: Fcc (Base clock HCLK/FCLK)"of "Data Sheet".

### Internal bus clocks

The following signals are bus clocks generated internally.

- Base clock (HCLK/FCLK)

HCLK and FCLK are collectively called the base clock. Both HCLK and FCLK are supplied to the CPU.
HCLK is a clock for macro connected to the AHB bus.
The clock frequency can be set to between 1/1 and 1/16 frequency of the master clock.
This clock stops in timer mode, RTC mode, stop mode, deep standby RTC mode, and deep standby stop mode.
In sleep mode, the CPU stops the supply of HCLK while continuing the supply of FCLK.

■ APB0 bus clock (PCLK0)

PCLK0 is a clock for peripheral macro connected to the APB0 bus.
The clock frequency can be set to between 1/1 and 1/8 frequency of the base clock.
This clock stops in timer mode, RTC mode, stop mode, deep standby RTC mode, and deep standby stop mode.

■ APB1 bus clock (PCLK1)

PCLK1 is a clock for peripheral macro connected to the APB1 bus.
The clock frequency can be set to between 1/1 and 1/8 frequency of the base clock.
This clock stops in timer mode, RTC mode, stop mode, deep standby RTC mode, and deep standby stop mode.
The supply of the clock can be also stopped by setting a register.

### Clocks other than source clocks and internal bus clocks

■ CAN prescaler clock

This clock is the same clock as CLKPLL, used for CAN prescaler.
The frequency division used for the clock must be configured on the prescaler side.
This clock stops in RTC mode, stop mode, deep standby RTC mode, , and deep standby stop mode.

For operation settings of CAN prescaler, see Chapter "CAN Prescaler" in "Communication Macro Part".

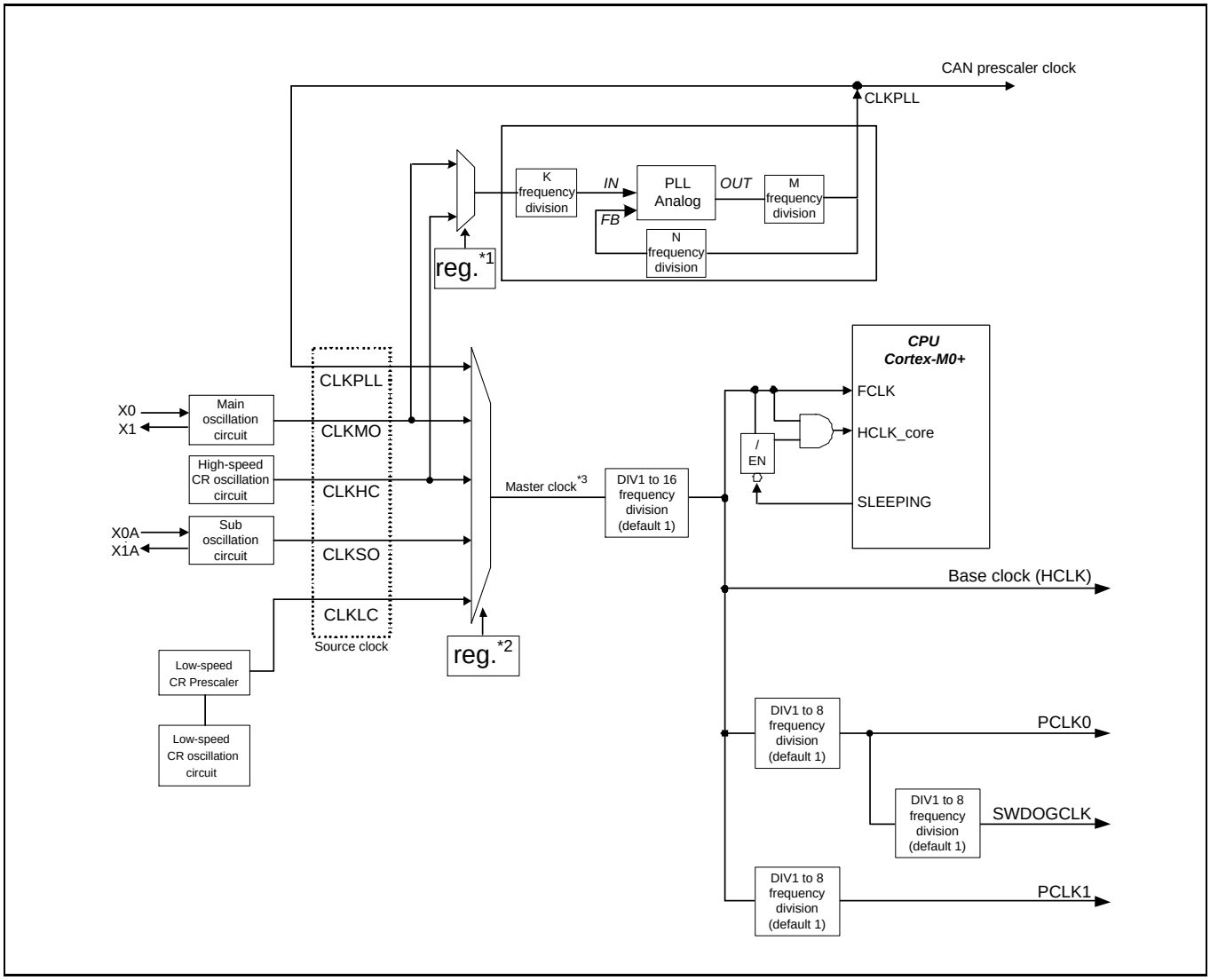
■ Software watchdog timer count clock (SWDOGCLK)

SWDOGCLK is a clock for the software watchdog timer connected to the APB0 bus.
The clock frequency can be set to between 1/1 and 1/8 frequency of the APB0 bus clock.
This clock stops in timer mode, RTC mode, stop mode, deep standby RTC mode, and deep standby stop mode.
For operation settings of the software watchdog timer, see Chapter "Watchdog Timer" in "Timer Part".

### Block diagram

Figure 2-1 shows the block diagram of the clock generation unit.

**Figure 2-1 Block Diagram of Clock Generation Unit**




*1: PSW_TMR: PINC (PLL input clock select bit)

*2: SCM_CTL: RCS[2:0] (Master clock switch control bits)

*3: The master clock frequency should not be larger than the maximum frequency of base clock (HCLK/FCLK). For the maximum frequency of base clock (HCLK/FCLK), see "Data Sheet" of the product used.

# 3. Operations

This section explains the clock generation unit.

## 3.1 Selecting Clock Mode

**Definition of clock mode (selecting the master clock)**

The MCU clock mode is defined by the source clock selected by the system clock mode control register. Five types of clock modes are provided: Main clock mode, sub clock mode, high-speed CR clock mode, low-speed CR clock mode, and main PLL clock mode.

■ Main clock mode

In main clock mode, the main clock (CLKMO) is used as a master clock. The clock runs a bus clock used to operate the CPU, and most peripheral functions.

The status of the PLL clock (CLKPLL) depends on the setting of the PLLE bit in the System Clock Mode Control Register (SCM_CTL), the status of the sub clock (CLKSO) on the setting of the SOSCE bit in the System Clock Mode Control Register (SCM_CTL), and the status of the high-speed CR clock (CLKHC) on the setting of the HCRE bit in the System Clock Mode Control Register (SCM_CTL). Besides, the MCSVE/FCSDE in CSV_CTL also activate the high-speed CR clock(CLKHC).The low-speed CR clock (CLKLC) cannot be stopped by user program.

■ Sub-clock mode

In sub clock mode, the sub clock (CLKSO) is used as a master clock. The clock runs a bus clock used to operate the CPU, and most peripheral functions.

The main clock (CLKMO), high-speed CR clock (CLKHC), and main PLL clock (CLKPLL) are stopped by hardware. The low-speed CR clock (CLKLC) cannot be stopped by user program.

■ High-speed CR clock mode

In high-speed CR clock mode, the high-speed CR clock (CLKHC) is used as a master clock. The clock runs a bus clock used to operate the CPU, and most peripheral functions.

Statuses of the main clock (CLKMO), main PLL clock (CLKPLL), and sub clock (CLKSO) differ depending on the settings of MOSCE, PLLE, and SOSCE bits in the System Clock Mode Control Register (SCM_CTL). The high-speed CR clock (CLKHC) and low-speed CR clock (CLKLC) cannot be stopped by user program.

■ Low-speed CR clock mode

In low-speed CR clock mode, the low-speed CR clock (CLKLC) is used as a master clock. The clock runs a bus clock used to operate the CPU, and most peripheral functions.

In low-speed CR clock mode, the main clock (CLKMO), high-speed CR clock (CLKHC), and main PLL clock (CLKPLL) are stopped by hardware. Status of the sub clock (CLKSO) differs depending on the setting of the SOSCE bit in the System Clock Mode Control Register (SCM_CTL).

■ Main PLL clock mode

In main PLL clock mode, the main PLL clock (CLKPLL) is used as a master clock. The clock runs a bus clock used to operate the CPU, and most peripheral functions.

Status of the sub clock (CLKSO) differs depending on the setting of the SOSCE bit in the System Clock Mode Control Register (SCM_CTL). The status of the high-speed CR clock (CLKHC) depends on the setting of some registers.For details , see Chapter "Low Power Consumption Mode". The status of main clock (CLKMO) depends on the setting of PINC in PSW_TMR or MOSCE in SCM_CTL. The low-speed CR clock (CLKLC) cannot be stopped by user program.

## 3.2 Internal Bus Clock Frequency Division Control

This section explains the internal bus clock frequency division.

Frequency division ratio from the base clock can be set independently for each internal bus clock.
This function can set the operating frequency optimized for each circuit.

The maximum frequency of the internal bus clock differs by product. For details, see "Data Sheet" of the product used.

To set the frequency division ratio of internal bus clocks, use the Base Clock Prescaler Register (BSC_PSR), APB0 Prescaler Register (APBC0_PSR) and APB1 Prescaler Register (APBC1_PSR). For details of each register, see "5. Registers".

**Setting the bus clock frequency division**

- The set frequency division ratio is not cleared by a software reset. The latest value is retained before the software reset.
- The value is initialized by a reset other than software resets.
  Before changing the initially set master clock to a faster source clock, be sure to set the frequency division ratio.
- If a combined value of master clock, PLL multiplication, and frequency division ratio settings exceeds the maximum operating frequency of each internal bus, the operation corresponding to the setting is not guaranteed.

## 3.3 PLL Clock Control

This section explains the PLL clock control.

The PLL Clock Control Circuit is used to generate the main PLL clock from the main clock or high-speed CR clock. The PLL Oscillation Circuit can enable/disable operation (oscillation), select the input clock, set the stabilization wait time, and set the multiplication.

### PLL operation

The following explains operation of the main PLL clock.

- Configure the following settings using the PLL Clock Oscillation Stabilization Wait Time Setup Register (PSW_TMR).
  - Selecting the PLL input clock
  - Setting the main PLL clock stabilization wait time
- The PLL oscillation enable bit (PLLE) of the System Clock Mode Control Register (SCM_CTL) must be enabled to let the PLL Circuit start oscillating.
- When the PLL clock stabilization wait time has elapsed, and the "PLL oscillation stable bit" of the System Clock Mode Status Register (SCM_STR) indicates a stable state, the preparation for transition to main PLL clock mode completes.
- Master clock switch control bit (RCS[2:0]) of the System Clock Mode Control Register (SCM_CTL) must be set to main PLL clock mode (RCS[2:0]=010) to change to main PLL clock mode.

### Setting the main PLL clock oscillation stabilization wait time

The details are given in "5.8 PLL Clock Stabilization Wait Time Setup Register (PSW_TMR)".

***Notes:***

- *For block diagram of the PLL Clock Control Circuit, see "2.Configuration".*
- *For the order of frequency division settings for each internal bus clock, see "4 Clock Setup Procedure Examples".*
- *For the oscillation stabilization wait time, see "3.4 Oscillation Stabilization Wait".*
- *When selecting high-speed CR in the input clock of PLL, see "1. Notes when high-speed CR is used for the master clock" in "APPENDIXES B. List of Notes".*

### Setting the multiplication ratio to generate the main PLL clock

Each frequency division clock in the PLL Multiplication Circuit must be set using PLL Control Register 1 (PLL_CTL1) and PLL Control Register 2 (PLL_CTL2). Table 3-1 provides an example of frequency division settings.

**Table 3-1 Example of PLL multiplication ratio settings**

| Input clock | K | PLLin | N | PLLout | M | CLKPLL |
|---|---|---|---|---|---|---|
| 4 MHz | 1 | 4 MHz | 2 | 80 MHz | 10 | 8 MHz |
| 4 MHz | 1 | 4 MHz | 4 | 80 MHz | 5 | 16 MHz |
| 4 MHz | 1 | 4 MHz | 5 | 80 MHz | 4 | 20 MHz |
| 4 MHz | 1 | 4 MHz | 6 | 120 MHz | 5 | 24 MHz |
| 4 MHz | 1 | 4 MHz | 9 | 108 MHz | 3 | 36 MHz |
| 4 MHz | 1 | 4 MHz | 10 | 80 MHz | 2 | 40 MHz |
| 8 MHz | 1 | 8 MHz | 5 | 80 MHz | 2 | 40 MHz |
| 8 MHz | 2 | 4 MHz | 10 | 80 MHz | 2 | 40 MHz |
| 12 MHz | 3 | 4 MHz | 10 | 80 MHz | 2 | 40 MHz |
| 16 MHz | 2 | 8 MHz | 5 | 80 MHz | 2 | 40 MHz |
| 16 MHz | 4 | 4 MHz | 10 | 80 MHz | 2 | 40 MHz |
| 24 MHz | 3 | 8 MHz | 5 | 80 MHz | 2 | 40 MHz |

***Notes:***

- *For PLL characteristics, see "Data Sheet" of the product used.*
- *Set the PLLin within the value "PLL input clock frequency: $f_{PLLI}$" shown in the "Data Sheet".*
- *The value "M×N" is a multiplication ratio for the PLLin. Set this value within the range shown in the "PLL multiple rate" of the "Data Sheet".*
- *The frequency of the PLLin multiplied by "M×N" becomes PLLout. Set this value within the range shown in the "PLL macro oscillation clock frequency: $f_{PLLO}$" of the "Data Sheet".*
- *The value of the PLLout divided by "M" becomes CLKPLL.*
- *See Figure 2-1 for the configurations of PLL and divider.*
- *The master clock / CLKPLL value should not be larger than the maximum value in "Internal operating clock frequency: Fcc (Base clock HCLK/FCLK)" of "Data Sheet".*

## 3.4 Oscillation Stabilization Wait Time

This section explains the oscillation stabilization wait time.

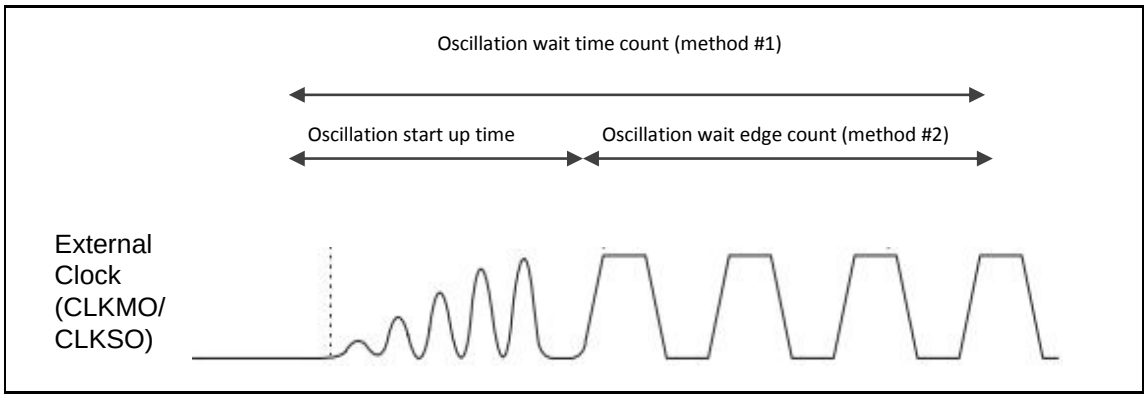
An oscillation stabilization wait time is required if the source clock is not in a stable operating state. During the oscillation stabilization wait time, internal and external clocks stop the supply, There are two method to wait until the stabilization wait time passes, a time value set in the Clock Stabilization Wait Time Register (CSW_TMR) or PLL Clock Oscillation Stabilization Wait Time Setup Register (PSW_TMR). When the wait time has been passed, the corresponding oscillator is ready to operate, and the clock can be used as a master clock.

**Method for clock stabilization wait count.**

- There are two kind methods to count external clock (CLKMO/CLKSO) stabilization time.
  1. The one of method is that a time of external clock (CLKMO/CLKSO) stabilization is counted by internal CR clock (CLKHC/CLKLC). In that way, after a particular time exceeded, external clock (CLKMO/CLKSO) becomes stable.
  2. The other method is that a number of the external clock (CLKMO/CLKSO) positive edges are counted for themselves wait count.
- In each those methods there are selection how long time / how many clock edge must be counted to it assumes the external clock becomes stable.

- Below is the example of clock stabilization time / number of edges selection.

**Figure 3-1 Oscillation wait count method.**




- The clock stabilization method is selected by whether clock supervisor is enabled or disabled.

  CSV_CTL.FCSDE=0 and CSV_CTL.MCSVE=0 : Number of CLKMO clock edge count method 2 is selected.

  CSV_CTL.FCSDE=1 or CSV_CTL.MCSVE=1 : Time counted by CLKHC method 1 is selected.

  CSV_CTL.SCSVE=0 : Number of CLKSO clock edge count method 2 is selected.

  CSV_CTL.SCSVE=1 : Time counted by CLKLC method 1 is selected.

### Setting the oscillation stabilization wait time

- Main clock (CLKMO)

Set the stabilization wait time of the main clock using the Clock Stabilization Wait Time Register (CSW_TMR). The set time value is counted by CLKHC (if Clock supervisor for CLKMO is enabled)/CLKMO (if Clock supervisor for CLKMO is disabled).

- Sub clock (CLKSO)

Set the stabilization wait time of the sub clock using the Clock Stabilization Wait Time Register (CSW_TMR). The set time value is counted by CLKLC (if CSV for CLKSO is enabled)/CLKSO (if CSV for CLKSO is disabled).

- Main PLL clock

Configure the following settings using the PLL Clock Oscillation Stabilization Wait Time Setup Register (PSW_TMR). The set time value is counted by CLKPLL.

- Selecting the PLL input clock
- Setting the main PLL clock stabilization wait time

### Cause of waiting for oscillation stability

- After the oscillation is enabled via software

If the PLLE, SOSCE, MOSCE and HCRE bits in the System Clock Mode Control Register (SCM_CTL) are set to "1", each relevant oscillator waits during the oscillation stabilization wait time.

- When returning to watch counter interrupt, RTC interrupt, and external interrupt from RTC mode

It returns to the clock mode before RTC mode by watch counter interrupt, RTC interrupt, and external interrupt. Hardware of a source clock waits for the oscillation stabilization wait time automatically.

- When returning from stop mode using an external interrupt

The status returns to clock mode, a state before stop mode, using an external clock. During stop mode, all source clocks stop and, therefore, the hardware automatically waits during the oscillation stabilization wait time.

- After PLL operation is enabled

After PLL operation is enabled, the PLL oscillation stabilization wait time is waited.

***Notes:***

- *Each set value of the oscillation stabilization wait time must be changed before the clock is enabled.*
- *After software reset, the oscillation stabilization wait time is not applied.*
- *The oscillation stabilization wait completion flag will be activated when the counting is complete. The oscillation stabilization wait time may be completed before oscillator stabilization if the setting of the oscillation stabilization wait time is too short.*
- *As the stabilization wait times for main clock and sub clock oscillators depend on the type of the oscillator (crystal, ceramics, etc.), proper oscillation stabilization wait time must be chosen for the oscillator to be used.*
- *Set the PLL oscillation stabilization wait time by referring to PLL Clock LOCKUP Time of the electric characteristics described in "Data Sheet" of the product used.*

## 3.5 Interrupt Factors

This section explains interrupt factors relevant to clocks.

The clock generation unit has the following interrupt factors.

**Interrupt factors**

The clock generation unit has the following four types of interrupt factors:

- FCS (anomalous frequency detection) interrupt

When the FCS (anomalous frequency detection) is enabled, and an anomalous frequency of the main clock is detected, an interrupt occurs.

- Main PLL clock oscillation stabilization wait completion interrupt

When the main PLL clock oscillation stabilization wait time ends, an interrupt occurs.

- Sub clock oscillation stabilization wait completion interrupt

When the sub clock oscillation stabilization wait time ends, an interrupt occurs.

- Main clock oscillation stabilization wait completion interrupt

When the main clock oscillation stabilization wait time ends, an interrupt occurs.

**Registers**

The following three types of registers are provided for each interrupt factor:

- Interrupt Enable Register (INT_ENR)

This register enables each interrupt.

- Interrupt Status Register (INT_STR)

This register indicates each interrupt status. This register is read-only.

- Interrupt Clear Register (INT_CLR)

This register clears each interrupt factor. This register is write-only.

# 4. Clock Setup Procedure Examples

This section explains procedure examples of setting up clocks.

### Setup procedure examples

**Figure 4-1 Example of clock setup procedure (Power-on -> High-speed CR run mode -> Desired clock mode)**

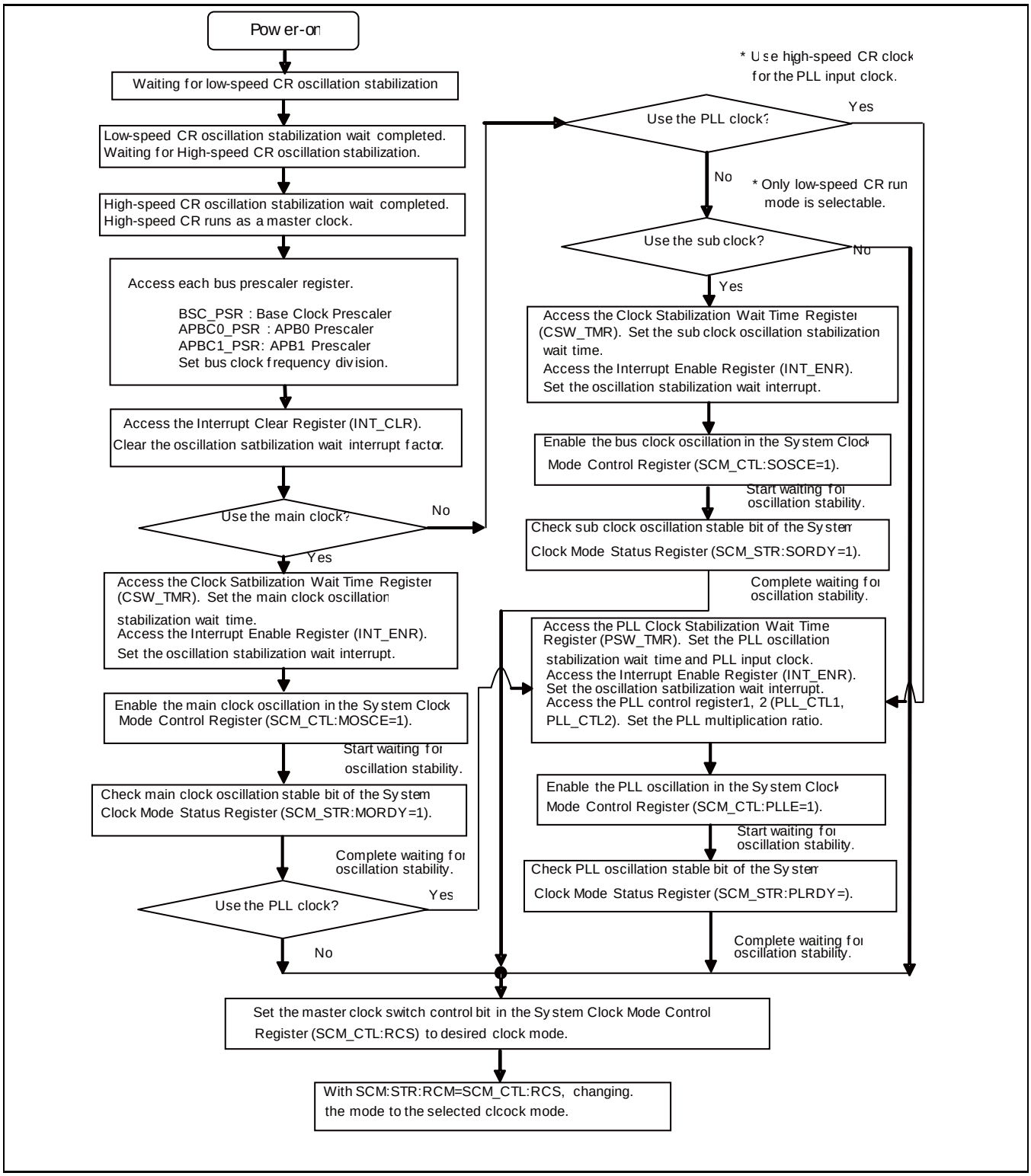
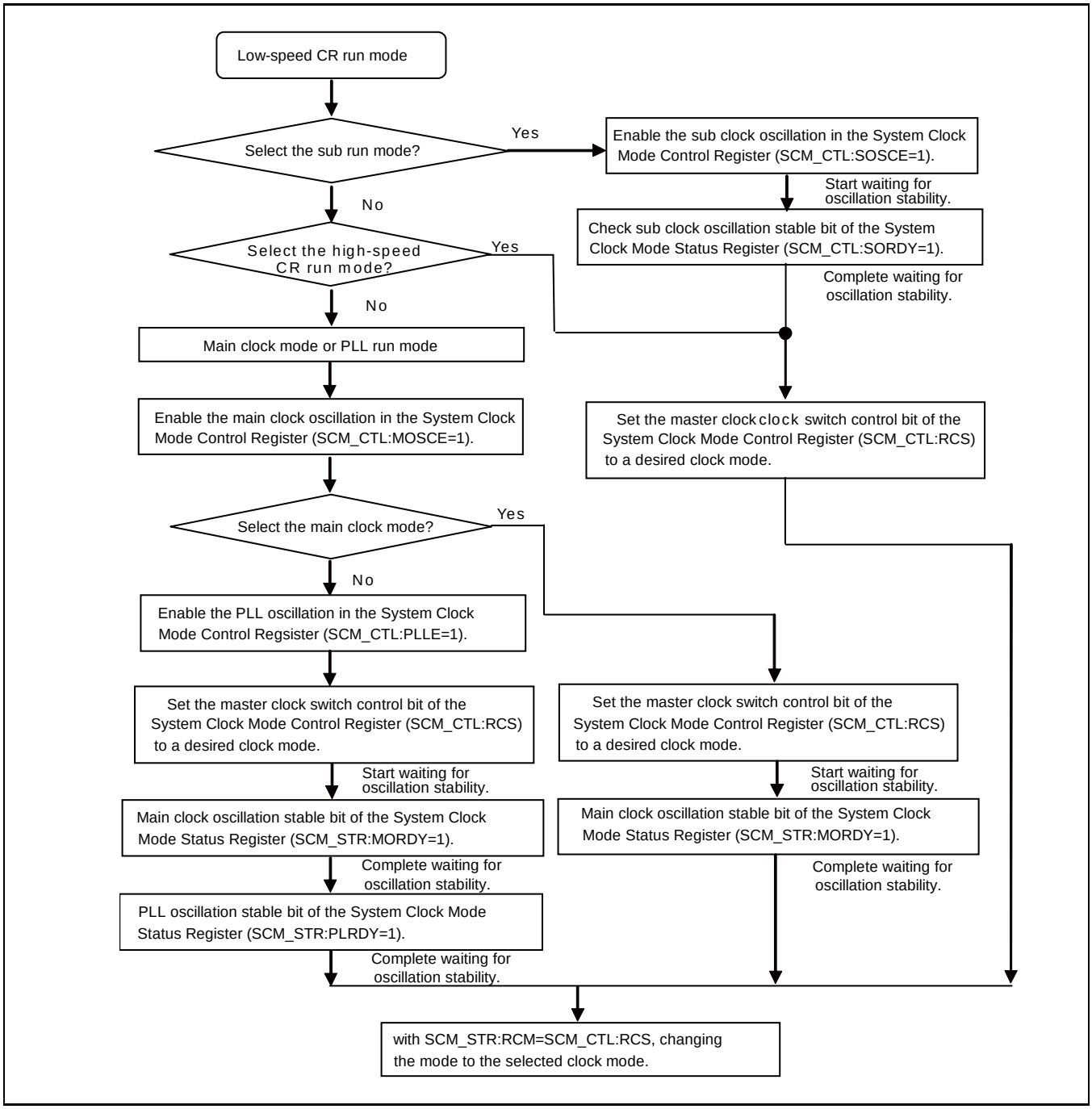


**Figure 4-2 Example of clock setup procedure (Low-speed CR run mode -> Desired clock run mode)**




***Notes:***

- *Figure 4-2 assumes that settings of the oscillation stabilization wait time, interrupts, PLL multiplication ratio and bus clock frequency division for each clock have been configured previously, and they are omitted from the flowchart.*
- *In the sub clock mode/low-speed CR clock mode, the main clock (CLKMO), high-speed CR (CLKHC), main PLL clock (CLKPLL) is stopped by hardware. So CLKMO/CLKHC/CLKPLL does not start oscillation only setting oscillation enable bit=1. These oscillations will start by changing the SCM_CTL:RCS bit with setting oscillation enable bit=1.*
- *If the main clock/sub clock oscillation stabilization wait times are short and the oscillation stabilization wait times run out before oscillators stabilize, reset may be applied by the clock supervisor function.*

# 5. Registers

This section describes the registers of the clock generation unit.

### List of clock generation unit registers

**Table 5-1 List of clock generation unit registers**

| Abbreviation | Register name | Reference |
|---|---|---|
| SCM_CTL | System Clock Mode Control Register | 5.1 |
| SCM_STR | System Clock Mode Status Register | 5.2 |
| BSC_PSR | Base Clock Prescaler Register | 5.3 |
| APBC0_PSR | APB0 Prescaler Register | 5.4 |
| APBC1_PSR | APB1 Prescaler Register | 5.5 |
| SWC_PSR | Software Watchdog Clock Prescaler Register | 5.6 |
| CSW_TMR | Clock Stabilization Wait Time Register | 5.7 |
| PSW_TMR | PLL Clock Stabilization Wait Time Setup Register | 5.8 |
| PLL_CTL1 | PLL Control Register 1 | 5.9 |
| PLL_CTL2 | PLL Control Register 2 | 5.10 |
| DBWDT_CTL | Debug Break Watchdog Timer Control Register | 5.11 |
| INT_ENR | Interrupt Enable Register | 5.12 |
| INT_STR | Interrupt Status Register | 5.13 |
| INT_CLR | Interrupt Clear Register | 5.14 |

## 5.1 System Clock Mode Control Register (SCM_CTL)

The System Clock Mode Control Register (SCM_CTL) selects the master clock and enables/disables the clock oscillation.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | RCS[2:0] | | | PLLE | SOSCE | Reserved | MOSCE | HCRE |
| Attribute | R/W | | | R/W | R/W | - | R/W | R/W |
| Initial value | 000 | | | 0 | 0 | - | 0 | 1 |

### Register functions

#### [bit7:5] RCS[2:0]: Master clock switch control bits

| bit7 | bit6 | bit5 | Description |
|---|---|---|---|
| 0 | 0 | 0 | High-speed CR clock [Initial value] |
| 0 | 0 | 1 | Main clock |
| 0 | 1 | 0 | Main PLL clock |
| 0 | 1 | 1 | Setting is prohibited |
| 1 | 0 | 0 | Low-speed CR clock |
| 1 | 0 | 1 | Sub clock |
| 1 | 1 | 0 | Setting is prohibited |
| 1 | 1 | 1 | Setting is prohibited |

#### [bit4] PLLE: PLL oscillation enable bit

| bit | Description |
|---|---|
| 0 | Disables PLL oscillation [Initial value] |
| 1 | Enables PLL oscillation |

#### [bit3] SOSCE: Sub clock oscillation enable bit

| bit | Description |
|---|---|
| 0 | Disables sub clock oscillation [Initial value] |
| 1 | Enables sub clock oscillation |

#### [bit2] Reserved: Reserved bit

"0" is read from this bit. Set this bit to "0" when writing.

#### [bit1] MOSCE: Main clock oscillation enable bit

| bit | Description |
|---|---|
| 0 | Disables main clock oscillation [Initial value] |
| 1 | Enables main clock oscillation |

#### [bit0] HCRE: High-speed CR clock oscillation enable bit

| bit | Description |
|---|---|
| 0 | Disables high-speed CR oscillation. |
| 1 | Enables the high-speed CR oscillation. [Initial value] |

***Notes:***

- *This register is not initialized by software reset.*
- *When you change the clock mode, you should set the enable bit to transition for desired clock oscillation. Then, you can change the clock switch control bits (SCM_CTL:RCS[2:0]).*
- *When RTCE bit (PMD_CTL:RTCE) is "1", it becomes a sub clock oscillation enable state regardless of the SOSCE bit and SORDY bit values.*
- *Writing "1" to RTCE bit (PMD_CTL:RTCE) is enabled only when SORDY bit is "1".*
- *RTCE bit (PMD_CTL:RTCE) does not exist in the products that do not have RTC mode and deep standby RTC mode. See Table 1-1 in the Chapter "Low Power Consumption Mode".*
- *When FCSDE bit (CSV_CTL:FCSDE) or MCSVE bit(CSV_CTL:MCSVE) is "1", it becomes a High-speed CR clock oscillation enable state regardless of the HCRE bit value.*
- *When any clock is in the stabilization waiting state, don't enable other clocks oscillation.*

## 5.2 System Clock Mode Status Register (SCM_STR)

The System Clock Mode Status Register (SCM_STR) indicates a clock selected for the master clock and a waiting state for clock oscillation stability.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | RCM[2:0] | | | PLRDY | SORDY | Reserved | MORDY | HCRDY |
| Attribute | R | | | R | R | - | R | R |
| Initial value | 000 | | | 0 | 0 | - | 0 | 1 |

### Register functions

#### [bit7:5] RCM[2:0]: Master clock selection bits

| bit7 | bit6 | bit5 | Description |
|---|---|---|---|
| 0 | 0 | 0 | High-speed CR clock [Initial value] |
| 0 | 0 | 1 | Main clock |
| 0 | 1 | 0 | Main PLL clock |
| 0 | 1 | 1 | Setting is prohibited |
| 1 | 0 | 0 | Low-speed CR clock |
| 1 | 0 | 1 | Sub clock |
| 1 | 1 | 0 | Setting is prohibited |
| 1 | 1 | 1 | Setting is prohibited |

#### [bit4] PLRDY: PLL oscillation stable bit

| bit | Description |
|---|---|
| 0 | In the stabilization wait or the oscillation stop state [Initial value] |
| 1 | In the stable state |

#### [bit3] SORDY: Sub clock oscillation stable bit

| bit | Description |
|---|---|
| 0 | In the stabilization wait or the oscillation stop state [Initial value] |
| 1 | In the stable state |

#### [bit2] Reserved: Reserved bit

"0" is read from this bit.

#### [bit1] MORDY: Main clock oscillation stable bit

| bit | Description |
|---|---|
| 0 | In the stabilization wait or the oscillation stop state [Initial value] |
| 1 | In the stable state |

#### [bit0] HCRDY: High-speed CR clock oscillation stable bit

| bit | Description |
|---|---|
| 0 | In the stabilization wait or the oscillation stop state |
| 1 | In the stable state [Initial value] |

***Notes:***

- *This register is not initialized by software reset.*
- *When RTCE bit (PMD_CTL:RTCE) is "1", it becomes a sub clock oscillation enable state regardless of the SOSCE bit and SORDY bit values.*
- *Writing "1" to RTCE bit (PMD_CTL:RTCE) is enabled only when SORDY bit is "1".*

## 5.3 Base Clock Prescaler Register (BSC_PSR)

The Base Clock Prescaler Register (BSC_PSR) sets the frequency division ratio of the base clock.

**Register configuration**

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | BSR | | |
| Attribute | - | | | | | R/W | | |
| Initial value | - | | | | | 000 | | |

**Register functions**

**[bit7:3] Reserved: Reserved bits**

"0b00000" is read from these bits. Set these bits to "0b00000" when writing.

**[bit2:0] BSR: Base clock frequency division ratio setting bits**

| bit2 | bit1 | bit0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | 1/1 [Initial value] |
| 0 | 0 | 1 | 1/2 |
| 0 | 1 | 0 | 1/3 |
| 0 | 1 | 1 | 1/4 |
| 1 | 0 | 0 | 1/6 |
| 1 | 0 | 1 | 1/8 |
| 1 | 1 | 0 | 1/16 |
| 1 | 1 | 1 | Setting is prohibited |

***Note:***

- *This register is not initialized by software reset.*

## 5.4 APB0 Prescaler Register (APBC0_PSR)

The APB0 Prescaler Register (APBC0_PSR) sets the APB0 bus clock frequency division.

**Register configuration**

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | APBC0 | |
| Attribute | - | | | | | | R/W | |
| Initial value | - | | | | | | 00 | |

**Register functions**

**[bit7:2] Reserved: Reserved bits**

"0b000000" is read from these bits. Set these bits to "0b000000" when writing.

**[bit1:0] APBC0: APB0 bus clock frequency division setting bits**

| bit1 | bit0 | Description |
|---|---|---|
| 0 | 0 | 1/1 [Initial value] |
| 0 | 1 | 1/2 |
| 1 | 0 | 1/4 |
| 1 | 1 | 1/8 |

***Note:***

- *This register is not initialized by software reset.*

## 5.5 APB1 Prescaler Register (APBC1_PSR)

The APB1 Prescaler Register (APBC1_PSR) sets the APB1 bus clock frequency division.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | APBC1EN | Reserved | | APBC1RST | Reserved | | APBC1 | |
| Attribute | R/W | - | | R/W | - | | R/W | |
| Initial value | 1 | - | | 0 | - | | 00 | |

### Register functions

#### [bit7] APBC1EN: APB1 clock enable bit

| bit | Description |
|---|---|
| 0 | Disables PCLK1 output |
| 1 | Enables PCLK1 output [Initial value] |

#### [bit6:5] Reserved: Reserved bits

"0b00" is read from these bits. Set these bits to "0b00" when writing.

#### [bit4] APBC1RST: APB1 bus reset control bit

| bit | Description |
|---|---|
| 0 | APB1 bus reset, inactive [Initial value] |
| 1 | APB1 bus reset, active |

#### [bit3:2] Reserved: Reserved bits

"0b00" is read from these bits. Set these bits to "0b00" when writing.

#### [bit1:0] APBC1: APB1 bus clock frequency division setting bits

| bit1 | bit0 | Description |
|---|---|---|
| 0 | 0 | 1/1 [Initial value] |
| 0 | 1 | 1/2 |
| 1 | 0 | 1/4 |
| 1 | 1 | 1/8 |

***Note:***

- *This register is not initialized by software reset.*

## 5.6 Software Watchdog Clock Prescaler Register (SWC_PSR)

The Software Watchdog Clock Prescaler Register (SWC_PSR) sets the frequency division and enables the output of the software watchdog clock.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | TESTB | Reserved | | | | | SWDS | |
| Attribute | R/W | - | | | | | R/W | |
| Initial value | X | - | | | | | 00 | |

### Register functions

#### [bit7] TESTB: TEST bit

| bit | Description |
|---|---|
| 0 | Setting is prohibited |
| 1 | Always written by "1" |

Note: The read value of this bit is undefined.

#### [bit6:2] Reserved: Reserved bits

"0b00000" is read from these bits. Set these bits to "0b00000" when writing.

#### [bit1:0] SWDS: Software watchdog clock frequency division ratio setting bits

| bit1 | bit0 | Description |
|---|---|---|
| 0 | 0 | Sets 1/1 frequency of PCLK0. [Initial value] |
| 0 | 1 | Sets 1/2 frequency of PCLK0. |
| 1 | 0 | Sets 1/4 frequency of PCLK0. |
| 1 | 1 | Sets 1/8 frequency of PCLK0. |

***Notes:***

- *This register is not initialized by software reset.*
- *Be sure to set the TESTB bit to "1" when writing a value to this register.*

## 5.7 Clock Stabilization Wait Time Register (CSW_TMR)

The Clock Stabilization Wait Time Register (CSW_TMR) sets the stabilization wait time of the main/sub clock.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | SOWT | | | | MOWT | | | |
| Attribute | R/W | | | | R/W | | | |
| Initial value | 0000 | | | | 0000 | | | |

### Register functions

#### [bit7:4] SOWT: Sub clock stabilization wait time setup bits

The stabilization wait time is counted by the CLKLC or CLKSO.

| bit7 | bit6 | bit5 | bit4 | Description | Calculation example1 CLKLC=100kHz | Calculation example2 CLKSO=32.768kHz |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | $2^{10}$ cycles [Initial value] | Approx. 10.3 ms | Approx. 31.3 ms |
| 0 | 0 | 0 | 1 | $2^{11}$ cycles | Approx. 20.5 ms | Approx. 62.5 ms |
| 0 | 0 | 1 | 0 | $2^{12}$ cycles | Approx. 41 ms | Approx. 125 ms |
| 0 | 0 | 1 | 1 | $2^{13}$ cycles | Approx. 82 ms | Approx. 250 ms |
| 0 | 1 | 0 | 0 | $2^{14}$ cycles | Approx. 164 ms | Approx. 500 ms |
| 0 | 1 | 0 | 1 | $2^{15}$ cycles | Approx. 327 ms | Approx. 1.00 s |
| 0 | 1 | 1 | 0 | $2^{16}$ cycles | Approx. 655 ms | Approx. 2.00 s |
| 0 | 1 | 1 | 1 | $2^{17}$ cycles | Approx. 1.31 s | Approx. 4.00 s |
| 1 | 0 | 0 | 0 | $2^{18}$ cycles | Approx. 2.62 s | Approx. 8.00 s |
| 1 | 0 | 0 | 1 | $2^{19}$ cycles | Approx. 5.24 s | Approx. 16.0 s |
| 1 | 0 | 1 | 0 | $2^{20}$ cycles | Approx. 10.49 s | Approx. 32.0 s |
| 1 | 0 | 1 | 1 | $2^{21}$ cycles | Approx. 20.97 s | Approx. 64.0 s |
| 1 | 1 | 0 | 0 | $2^{1}$ cycles | Approx. 0.2 ms | Approx. 0.06 ms |
| 1 | 1 | 0 | 1 | $2^{2}$ cycles | Approx. 0.4 ms | Approx. 0.12 ms |
| 1 | 1 | 1 | 0 | $2^{3}$ cycles | Approx. 0.8 ms | Approx. 0.24 ms |
| 1 | 1 | 1 | 1 | $2^{4}$ cycles | Approx. 1.6 ms | Approx. 0.49 ms |

**[bit3:0] MOWT: Main clock stabilization wait time setup bits**

The stabilization wait time is counted by the CLKHC or CLKMO.

| bit3 | bit2 | bit1 | bit0 | Description | Calculation example1 CLKHC=4MHz or CLKMO=4MHz | Calculation example2 CLKMO=40MHz |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | $2^1$ cycles [Initial value] | Approx. 500 ns | Approx. 50 ns |
| 0 | 0 | 0 | 1 | $2^5$ cycles | Approx. 8 µs | Approx. 0.8 µs |
| 0 | 0 | 1 | 0 | $2^6$ cycles | Approx. 16 µs | Approx. 1.6 µs |
| 0 | 0 | 1 | 1 | $2^7$ cycles | Approx. 32 µs | Approx. 3.2 µs |
| 0 | 1 | 0 | 0 | $2^8$ cycles | Approx. 64 µs | Approx. 6.4 µs |
| 0 | 1 | 0 | 1 | $2^9$ cycles | Approx. 128 µs | Approx. 12.8 µs |
| 0 | 1 | 1 | 0 | $2^{10}$ cycles | Approx. 256 µs | Approx. 25.6 µs |
| 0 | 1 | 1 | 1 | $2^{11}$ cycles | Approx. 512 µs | Approx. 51.2 µs |
| 1 | 0 | 0 | 0 | $2^{12}$ cycles | Approx. 1.0 ms | Approx. 0.1 ms |
| 1 | 0 | 0 | 1 | $2^{13}$ cycles | Approx. 2.0 ms | Approx. 0.2 ms |
| 1 | 0 | 1 | 0 | $2^{14}$ cycles | Approx. 4.0 ms | Approx. 0.4 ms |
| 1 | 0 | 1 | 1 | $2^{15}$ cycles | Approx. 8.0 ms | Approx. 0.8 ms |
| 1 | 1 | 0 | 0 | $2^{17}$ cycles | Approx. 33.0 ms | Approx. 3.3 ms |
| 1 | 1 | 0 | 1 | $2^{19}$ cycles | Approx. 131 ms | Approx. 13.1 ms |
| 1 | 1 | 1 | 0 | $2^{21}$ cycles | Approx. 524 ms | Approx. 52.4 ms |
| 1 | 1 | 1 | 1 | $2^{23}$ cycles | Approx. 2.0 s | Approx. 0.2 s |

***Notes:***

- *Set each oscillation stabilization wait time before enabling each oscillation enable bit (SOSCE, MOSCE) of the SCM_CTL register.*
  *If you change MOWT or SOWT bit while waiting for oscillation stability of each oscillator, each oscillation stabilization wait time is not guaranteed.*
- *This register is not initialized by software reset.*

## 5.8 PLL Clock Stabilization Wait Time Setup Register (PSW_TMR)

The PLL Clock Stabilization Wait Time Setup Register (PSW_TMR) sets the main PLL clock stabilization wait time.

**Register configuration**

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | PINC | Reserved | POWT | | |
| Attribute | - | | | R/W | - | R/W | | |
| Initial value | - | | | 0 | - | 000 | | |

**Register functions**

**[bit7:5] Reserved: Reserved bits**

"0b000" is read from these bits. Set these bits to "0b000" when writing.

**[bit4] PINC: PLL input clock select bit**

| bit | Description |
|---|---|
| 0 | Selects CLKMO (main clock oscillation) [Initial value] |
| 1 | Selects CLKHC (high-speed CR clock) |

Note: Setting this bit to "1" has some restrictions.
See "1. Notes when high-speed CR is used for the master clock" in "B. List of Notes".

**[bit3] Reserved: Reserved bit**

"0" is read from this bit. Set this bit to "0" when writing.

**[bit2:0] POWT: Main PLL clock stabilization wait time setup bits**

| bit2 | bit1 | bit0 | Description | Calculation example1 CLKPLL=20MHz | Calculation example2 CLKPLL=40MHz |
|---|---|---|---|---|---|
| 0 | 0 | 0 | $2^9$ cycles [Initial value] | Approx. 25.6 µs | Approx. 12.8 µs |
| 0 | 0 | 1 | $2^{10}$ cycles | Approx. 51.2 µs | Approx. 25.6 µs |
| 0 | 1 | 0 | $2^{11}$ cycles | Approx. 102.4 µs | Approx. 51.2 µs |
| 0 | 1 | 1 | $2^{12}$ cycles | Approx. 204.8 µs | Approx. 102.4 µs |
| 1 | 0 | 0 | $2^{13}$ cycles | Approx. 409.6 µs | Approx. 204.8 µs |
| 1 | 0 | 1 | $2^{14}$ cycles | Approx. 819.2 µs | Approx. 409.6 µs |
| 1 | 1 | 0 | $2^{15}$ cycles | Approx. 1638.4 µs | Approx. 819.2 µs |
| 1 | 1 | 1 | $2^{16}$ cycles | Approx. 3276.8 µs | Approx. 1638.4 µs |

***Notes:***

- *Set each oscillation stabilization wait time before enabling the PLL oscillation enable bit (PLLE) of the SCM_CTL.*
  *If you change POWT bit while waiting for oscillation stability of the PLL oscillator, the oscillation stabilization wait time is not guaranteed.*
- *This register is not initialized by software reset.*
- *In Main PLL clock mode, When PINC bit is "1", it becomes a High-speed CR clock oscillation enable state regardless of the HCRE bit(SCM_CTL:HCRE) value. When PINC bit is "0", The status of the high-speed CR clock (CLKHC) depends on the setting of HCRE bit, FCSDE bit(CSV_CTL:FCSDE) and MCSVE bit(CSV_CTL:MCSVE).*

- *In Main PLL clock mode, When PINC bit is "1", The status of the Main clock oscillation depends on the setting of MOSCE(SCM_CTL).When PINC bit is "0", it becomes a Main clock oscillation enable state regardless of the MOSCE bit value.*

## 5.9 PLL Control Register 1 (PLL_CTL1)

The PLL Control Register 1 (PLL_CTL1) sets the PLL frequency division ratio.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | PLLK | | | | PLLM | | | |
| Attribute | R/W | | | | R/W | | | |
| Initial value | 0000 | | | | 0000 | | | |

### Register functions

#### [bit7:4] PLLK: PLL input clock frequency division ratio setting bits

| bit 7:4 | Description |
|---|---|
| 0000 | The frequency division is (PLLK value +1). (Frequency division : 1 to 16)<br>Example: PLLK value (0000) +1 => 1/1 frequency [Initial value] |
| 0001 | |
| • | |
| • | |
| 1111 | |

#### [bit3:0] PLLM: PLL VCO clock frequency division ratio setting bits

| bit3:0 | Description |
|---|---|
| 0000 | The frequency division is (PLLM value +1). (Frequency division : 1 to 16)<br>Example: PLLM value (0000) +1 => 1/1 frequency [Initial value] |
| 0001 | |
| • | |
| • | |
| 1111 | |

***Notes:***

- *Set each frequency division ratio before enabling the PLL oscillation enable bit (PLLE) of the SCM_CTL register.*
- *This register is not initialized by software reset.*

## 5.10 PLL Control Register 2 (PLL_CTL2)

The PLL Control Register 2 (PLL_CTL2) sets the PLL frequency division ratio.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | PLLN | | | | | |
| Attribute | - | | R/W | | | | | |
| Initial value | - | | 000000 | | | | | |

### Register functions

#### [bit7:6] Reserved: Reserved bits

"0b000" is read from these bits. Set these bits to "0b000" when writing.

#### [bit5:0] PLLN: PLL feedback frequency division ratio setting bits

| bit5:0 | Description |
|---|---|
| 000000 | The frequency division is (PLLN value +1). (Frequency division : 1 to 50)<br>Example: PLLN value (000000) +1 => 1/1 division [Initial value] |
| 000001 | |
| . | |
| . | |
| 110001 | |
| 110010 | Setting is prohibited |
| . | |
| 111111 | |

***Notes:***

- *Set the frequency division ratio before enabling the PLL oscillation enable bit (PLLE) of the SCM_CTL register.*
- *This register is not initialized by software reset.*

## 5.11 Debug Break Watchdog Timer Control Register (DBWDT_CTL)

The Debug Break Watchdog Timer Control Register (DBWDT_CTL) sets the watchdog timer count operation for debug mode tool break.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | DPHWBE | Reserved | DPSWBE | Reserved | | | | |
| Attribute | R/W | - | R/W | - | | | | |
| Initial value | 0 | - | 0 | - | | | | |

### Register functions

#### [bit7] DPHWBE: HW-WDG debug mode break bit

| bit | Description |
|---|---|
| 0 | HW-WDG stops counting at the tool break [Initial value] |
| 1 | HW-WDG continues counting at the tool break |

#### [bit6] Reserved: Reserved bit

"0" is read from this bit. Set this bit to "0" when writing.

#### [bit5] DPSWBE: SW-WDG debug mode break bit

| bit | Description |
|---|---|
| 0 | SW-WDG stops counting at the tool break [Initial value] |
| 1 | SW-WDG continues counting at the tool break |

#### [bit4:0] Reserved: Reserved bits

"0b00000" is read from these bits. Set these bits to "0b00000" when writing.

***Note:***

- *This register is not initialized by software reset.*

## 5.12 Interrupt Enable Register (INT_ENR)

The Interrupt Enable Register (INT_ENR) enables/disables interrupts.

**Register configuration**

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | FCSE | Reserved | | PCSE | SCSE | MCSE |
| Attribute | - | | R/W | - | | R/W | R/W | R/W |
| Initial value | - | | 0 | - | | 0 | 0 | 0 |

**Register functions**

**[bit7:6] Reserved: Reserved bits**

"0b00" is read from these bits. Set these bits to "0b00" when writing.

**[bit5] FCSE: Anomalous frequency detection interrupt enable bit**

| bit | Description |
|---|---|
| 0 | Disables FCS interrupts |
| 1 | Enables FCS interrupts |

**[bit4:3] Reserved: Reserved bits**

"0b00" is read from these bits. Set these bits to "0b00" when writing.

**[bit2] PCSE: PLL oscillation stabilization wait completion interrupt enable bit**

| bit | Description |
|---|---|
| 0 | Disables PLL oscillation stabilization wait completion interrupts |
| 1 | Enables PLL oscillation stabilization wait completion interrupts |

**[bit1] SCSE: Sub clock oscillation stabilization wait completion interrupt enable bit**

| bit | Description |
|---|---|
| 0 | Disables sub clock oscillation stabilization wait completion interrupts |
| 1 | Enables sub clock oscillation stabilization wait completion interrupts |

**[bit0] MCSE: Main clock oscillation stabilization wait completion interrupt enable bit**

| bit | Description |
|---|---|
| 0 | Disables main clock oscillation stabilization wait completion interrupts |
| 1 | Enables main clock oscillation stabilization wait completion interrupts |

***Note:***

- *For "Anomalous frequency detection", see Chapter "Clock supervisor".*

## 5.13 Interrupt Status Register (INT_STR)

The Interrupt Status Register (INT_STR) indicates the status of interrupts.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | FCSI | Reserved | | PCSI | SCSI | MCSI |
| Attribute | - | | R | - | | R | R | R |
| Initial value | - | | 0 | - | | 0 | 0 | 0 |

### Register functions

**[bit7:6] Reserved: Reserved bits**

"0b00" is read from these bits. Set these bits to "0b00" when writing.

**[bit5] FCSI: Anomalous frequency detection interrupt status bit**

| bit | Description |
|---|---|
| 0 | No FCS interrupt has been asserted. |
| 1 | An FCS interrupt has been asserted. |

**[bit4:3] Reserved: Reserved bits**

"0b00" is read from these bits. Set these bits to "0b00" when writing.

**[bit2] PCSI: PLL oscillation stabilization wait completion interrupt status bit**

| bit | Description |
|---|---|
| 0 | No PLL oscillation stabilization wait completion interrupt has been asserted. |
| 1 | A PLL oscillation stabilization wait completion interrupt has been asserted. |

**[bit1] SCSI: Sub clock oscillation stabilization wait completion interrupt status bit**

| bit | Description |
|---|---|
| 0 | No sub clock oscillation stabilization wait completion interrupt has been asserted. |
| 1 | A sub clock oscillation stabilization wait completion interrupt has been asserted. |

**[bit0] MCSI: Main clock oscillation stabilization wait completion interrupt status bit**

| bit | Description |
|---|---|
| 0 | No main clock oscillation stabilization wait completion interrupt has been asserted. |
| 1 | A main clock oscillation stabilization wait completion interrupt has been asserted. |

## 5.14 Interrupt Clear Register (INT_CLR)

The Interrupt Clear Register (INT_CLR) clears interrupt factors.

**Register configuration**

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | FCSC | Reserved | | PCSC | SCSC | MCSC |
| Attribute | - | | W | - | | W | W | W |
| Initial value | - | | 0 | - | | 0 | 0 | 0 |

**Register functions**

**[bit7:6] Reserved: Reserved bits**

"0b00" is read from these bits. Set these bits to "0b00" when writing.

**[bit5] FCSC: Anomalous frequency detection interrupt factor clear bit**

| bit | Description |
|---|---|
| When 0 is written | The FCS interrupt factor is not affected by the written value. |
| When 1 is written | Clears the FCS interrupt factor. |
| When read | The fixed value "0" is read. |

**[bit4:3] Reserved: Reserved bits**

"0b00" is read from these bits. Set these bits to "0b00" when writing.

**[bit2] PCSC: PLL oscillation stabilization wait completion interrupt factor clear bit**

| bit | Description |
|---|---|
| When 0 is written | The PLL oscillation stabilization wait completion interrupt factor is not affected by the written value. |
| When 1 is written | Clears the PLL oscillation stabilization wait completion interrupt factor. |
| When read | The fixed value "0" is read. |

**[bit1] SCSC: Sub clock oscillation stabilization wait completion interrupt factor clear bit**

| bit | Description |
|---|---|
| When 0 is written | The sub clock oscillation stabilization wait completion interrupt factor is not affected by the written value. |
| When 1 is written | Clears the sub clock oscillation stabilization wait completion interrupt factor. |
| When read | The fixed value "0" is read. |

**[bit0] MCSC: Main clock oscillation stabilization wait completion interrupt factor clear bit**

| bit | Description |
|---|---|
| When 0 is written | The main clock oscillation stabilization wait completion interrupt factor is not affected by the written value. |
| When 1 is written | Clears the main clock oscillation stabilization wait completion interrupt factor. |
| When read | The fixed value "0" is read. |

***Note:***

- *When this register is cleared, each interrupt status bit (FCSI, PCSI, SCSI, MCSI) in the INT_STR register is also cleared.*

# 6. Usage Precautions

This section explains the precautions for using the clock generation unit.

■ The oscillation stabilization wait time of main clock and sub clock oscillators

Because the stabilization wait time of main clock/sub clock oscillator depends on the oscillator type (crystal, ceramic, etc.), the oscillation stabilization wait time suitable for the oscillator type must be selected.

■ Changing the frequency division under stabilized PLL oscillation

When the PLL frequency division ratio is changed after stabilization of PLL oscillation, stop the PLL oscillation once, change the frequency division ratio, and then re-enable the PLL oscillation.

■ Peripherals independent of clock control by the clock generation unit

The following peripherals run independently of clock control by the clock generation unit.
For information about how to handle each operating clock, see the following chapter.

- Clock supervisor: See Chapter "Clock supervisor".
- Watchdog timer : See Chapter "Watchdog Timer" in "Timer Part".
- Watch counter : See Chapter "Watch Counter" in "Timer Part".
- Real-time clock : See Chapter "REAL-TIME CLOCK" in "Timer Part".
- CAN prescaler : See Chapter "CAN Prescaler" in "Communication Macro Part".

■ Setting the oscillation stabilization wait time

Set the oscillation stabilization wait time of the main clock, sub clock, and PLL oscillators with relevant oscillation stabilization wait time setup registers, and then enable each oscillator.
Do not change the oscillation stabilization wait time while waiting for oscillation to stabilize.

■ Checking main clock oscillation while using the main PLL clock

It is prohibited to stop main clock oscillation while using PLL oscillation.

■ Switching clock modes

Clock modes can be switched by changing the RCS[2:0] bits of the SCM_CTL register.
To switch clock modes, take the following steps:

1. Set the oscillation stabilization wait time of each oscillator.
2. Set the oscillation enable bit of the desired clock (SCM_CTL:xxxE) to "1".
3. Check the oscillation stable bit of the desired clock (SCM_CTL:xxxRDY) to "1".
4. Switch SCM_CTL:RCS[2:0].
5. Wait until SCM_STR:RCM[2:0] = SCM_CTL:RCS[2:0].

■ Correlation between the clock mode switching and the oscillation stable bit

The timings when the oscillation stable bit (SCM_STR:xxxRDY) turns to "1" vary for the following clock mode switching.

- When switching from the high-speed CR run, main run, or PLL run to another clock mode:

  Setting SCM_CTL:xxxE to "1" can start the oscillation stabilization wait time. You can check that SCM_STR:xxxRDY is "1" after the oscillation stabilization wait time has elapsed.

- When switching from the low-speed CR run or sub run to the high-speed CR run, main run, or PLL run:

  Setting SCM_CTL:MOSCE (or PLLE) to 1 does not start the oscillation stabilization wait time. To start the main clock (or high-speed CR or PLL) oscillation stabilization wait time, SCM_CTL:RCS [2:0] must be switched. After the oscillation stabilization wait time has elapsed, you can check that SCM_STR:xxxRDY is "1".

- If the standby mode is released by an interrupt, the device restarts in the clock mode that indicated by the RCS[2:0] bits in the SCM_CTL register.
- If any reset occurs other than software resets, the high-speed CR clock (CLKHC) is set as a master clock. In addition, high-speed CR clock mode is set as clock mode.

- If any reset other than software resets is executed, the main clock and sub clock oscillators, and PLL oscillation stop. If you want to use those oscillators again after the reset, enable them using the SCM_CTL register.
- For the correlation between each clock mode and start/stop of the oscillator, see Chapter "Low Power Consumption Mode".

# CHAPTER2-2: Peripheral Clock Gating

**This chapter explains the functions of Peripheral Clock Gating.**



CODE: 9BFPCG-J1.0

# 1. Peripheral Clock Gating Overview

This section shows an overview of the Peripheral Clock Gating which stops the operation clocks of peripheral functions individually. By using these functions, the system can reduce the current consumption of the total system with gating the operation clocks of peripheral functions not used.

**Overview of Peripheral Clock Gating**

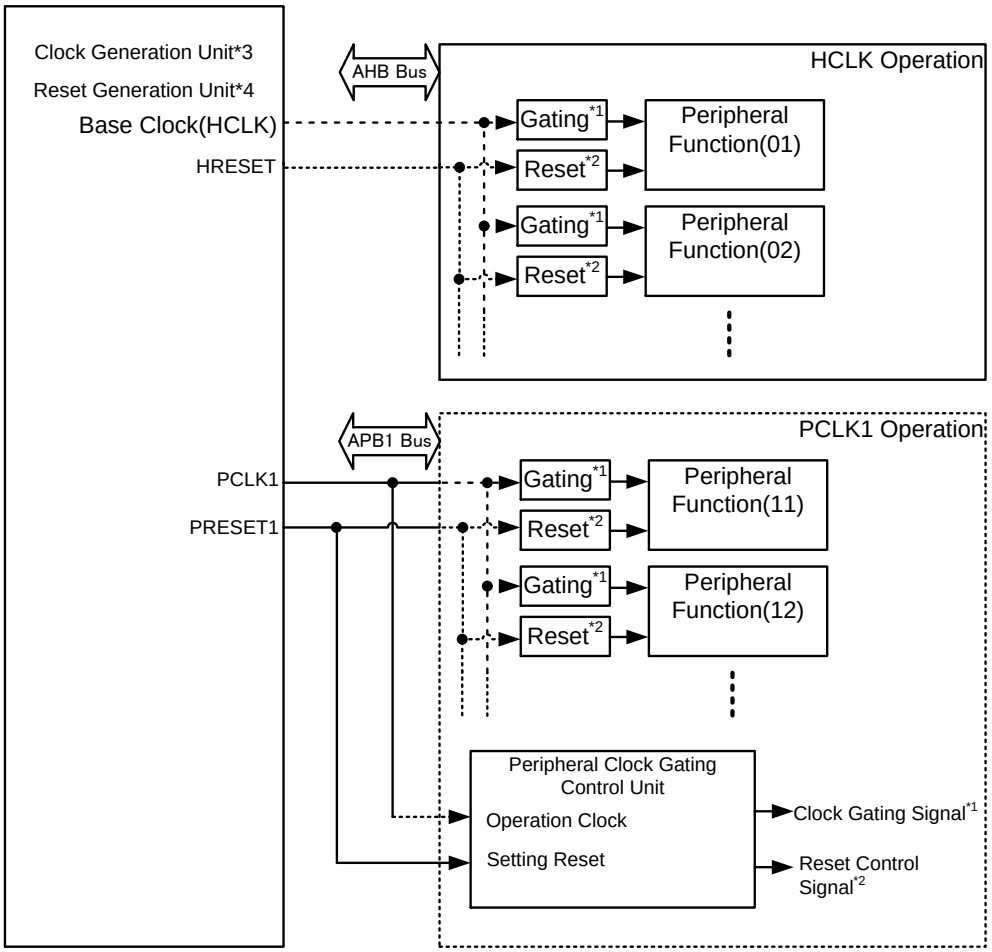
- The operation clocks of peripheral functions not used in the system operation are gated individually.
- For target clocks and units of the Peripheral Clock Gating, see "■ Gating units and their initial states of Peripheral Clock Gating"
- When a clock is gated or before a clock is supplied, the internal states of peripheral functions can be reset.

The above peripheral clock gating and reset control are implemented by the setting of a register connecting to APB1 bus.

### Overview of Connection with Clock and Reset Generation Units

Figure 1-1 shows the connection between peripheral clock gating and clock generation unit or reset generation unit. The peripheral clock gating exists between peripheral function and clock generation unit or reset generation unit and gates clocks and controls resets in the unit of peripheral function. When the internal bus clock supply from the reset control units are stopped, the priority is given to the settings of the clock control unit and the operation clock supplies to peripheral functions are gated To use the peripheral clock gating, be sure to make the settings which enable the output of APB1 bus clock (PCLK1) in the clock generation unit to control the rest.

**Figure 1-1 Clock/Reset Connection related to Peripheral Clock Gating**




*1 : Clock gating (internal bus clock gating) in the unit of peripheral function

*2: Forcible reset or bus reset (see Chapter "Reset") by peripheral function unit

*3: For details, see Chapter "Clock".

*4: For details, see Chapter "Reset".

### Gating units and their initial states of Peripheral Clock

For gating units and their initial states of Peripheral Clock Gating, see Table 1-1.

**Table 1-1Control units and their initial states of Peripheral Clock Gating**

| Peripheral Functions | Clock Stop Units | Initial States | Remarks |
|---|---|---|---|
| Multi-function Serial Interface | One channel | Clock supply | |
| Base timer | Four channels | Clock supply | The clock gating can be controlled with every four channels "Ch.0 to Ch.3", "Ch.4 to Ch.7", "Ch.8 to Ch.11", and "Ch.12 to Ch.15". |
| Multi-function timer | Unit | Clock supply | |
| PPG | Four channels | Clock supply | The clock gating can be controlled with every four channels "Chs.0, 2, 4, and 6", "Chs.8, 10, 12, and 14", "Chs.16, 18, 20, and 22", and "Chs.24, 26, 28, and 30". |
| Quad counter | Unit | Clock supply | |
| DMAC | Unit | Clock supply | |
| CAN controller | One channel | Clock supply | |
| A/D converter | Unit | Clock supply | |
| IO Port (GPIO/Fast GPIO) | All ports | Clock supply | For constrains at clock gating, "Usage Precautions" in 5. Peripheral Clock Gating. |

***Notes:***

- *For types and the number of mounted peripheral functions, see 'Data sheet' of the product used*
- *The clock control of PPG shares the setting bits with the multi-function timer. For details, see "4.3Peripheral Clock Control Register 1 (CKEN1)".*

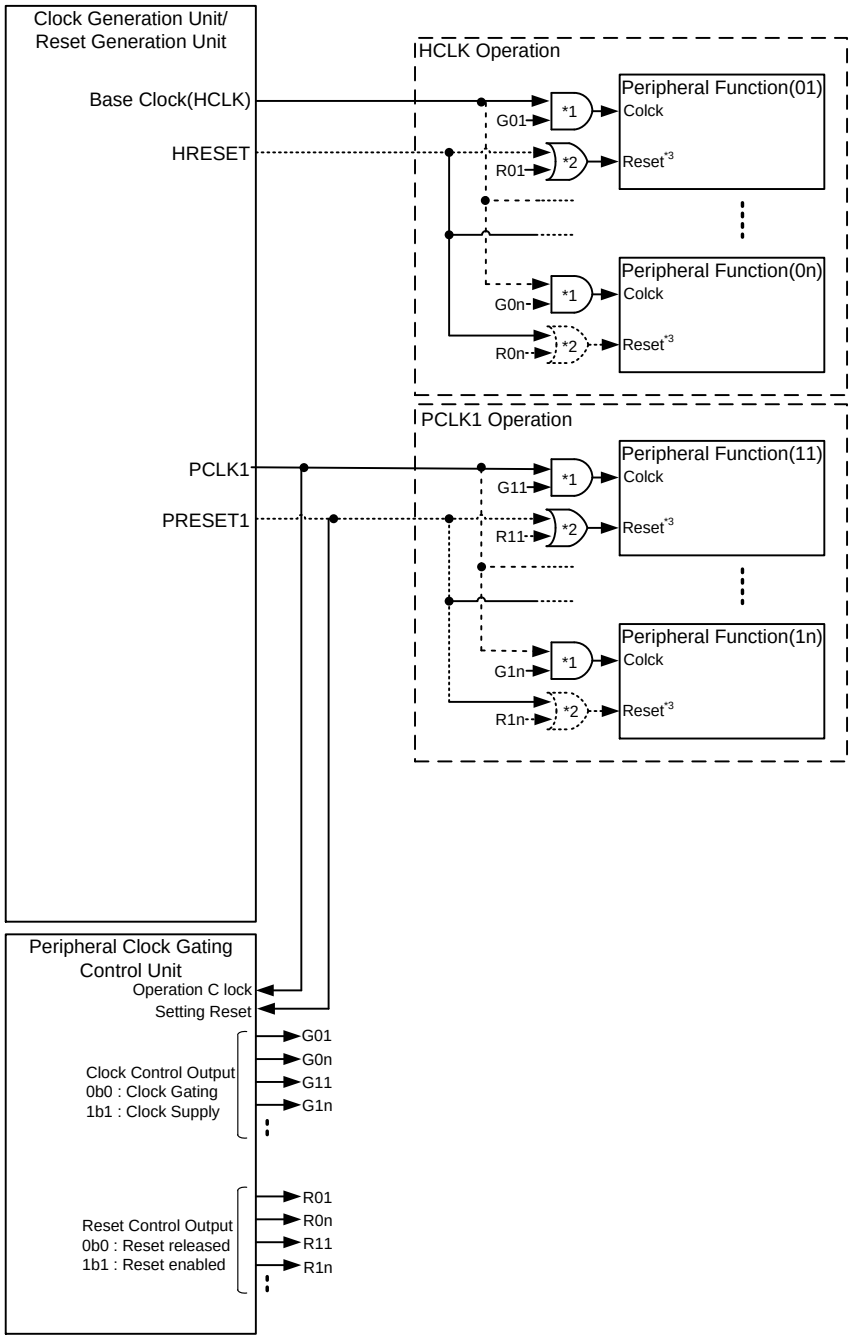
# 2. Peripheral Clock Gating Configuration

This section explains the configuration of the Peripheral Clock Gating.

### Block diagram

Figure 2-1 shows the system configuration of Peripheral Clock Gating.

**Figure 2-1 Block Diagram of Peripheral Clock Gating**




*1 : Gates the operation clocks of peripheral functions
*2 : Resets peripheral functions
*3 : At High-level signal input: Reset enabled, At Low-level signal input: Reset released

### Explanation on Block Diagram

■ Peripheral Clock Gating Control Unit

The clock control or the reset control of each peripheral function is executed by changing the register setting value via the APB1 bus. Be sure to rewrite this register with setting APB1 clock enable bit (APBC1EN) in APB1 prescaler register (APBC1_PSR) of the clock control unit to the output enable and permitting PCLK1 output.

The clock of each peripheral function stops when the bit field of the target function is set to "0". When the bit field is set to "1", the clock is supplied. The initial value of a register is different by peripheral function. For details, see Table 1-1.

The reset of each peripheral function is issued when the bit field of the target function is set to"1". When the bit field is set to "0", the reset is released. The initial value of each register is always "0" to release the reset.

■ Peripheral Clock Gating Logic

Internal bus clock (HCLK, PCLK1) is supplied or gated by each specific peripheral function according to clock gating signal from the peripheral clock gating control unit.

■ Peripheral Reset Control Logic

The reset is individually controlled by each peripheral function according to the reset control signal from the peripheral clock gating control unit. The reset control unit is the same with the peripheral clock control unit. However, it does not exceptionally have the bit field of this reset control for I/O Port alone

# 3. Peripheral Clock Gating Control

This section explains the control of the peripheral clock gating.

The register of the peripheral clock gating becomes an initial state by bus reset (PRESET1)*. Be sure to execute the clock control for necessary peripheral functions immediately after reset of the bus because the bus reset (PRESET1) is generated by all reset factors.

*: For the generating condition of bus reset (PRESET1), see Chapter 'Reset'.

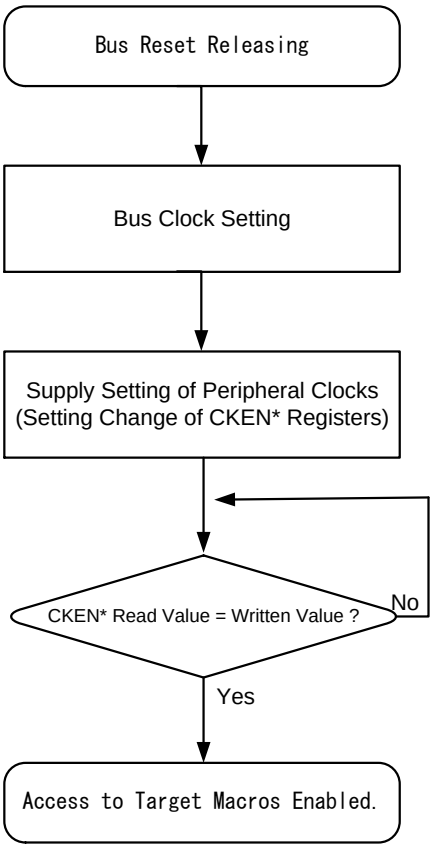
## 3.1 Peripheral Clock Control Procedures

This section explains the control procedures of supplying and stopping peripheral clocks.

**Clock supply procedures**

The settings of the bus clocks and the peripheral clocks are reset to the initial values immediately after the bus reset release. So, for the clocks of peripheral functions which have been stopped in the initial state, set the clock supplies conforming to the procedures in Figure 3-1.

**Figure 3-1 Clock Supply Procedures**




*:Indicates CKEN0, CKEN1, and CKEN2.

1. Bus clock setting

   Execute the setting of each bus clock by using the register of the clock generation part.

   For the setting details, see Chapter 'Clock'.

2. Supply setting of peripheral clocks

   Change the setting of the bit corresponding to the peripheral function to which the clock is to be supplied for peripheral clock control registers (CKEN0, CKEN1, and CKEN2) of the clock control in the clock gating state of the initial state.

3. Set value confirmation of peripheral clock control register

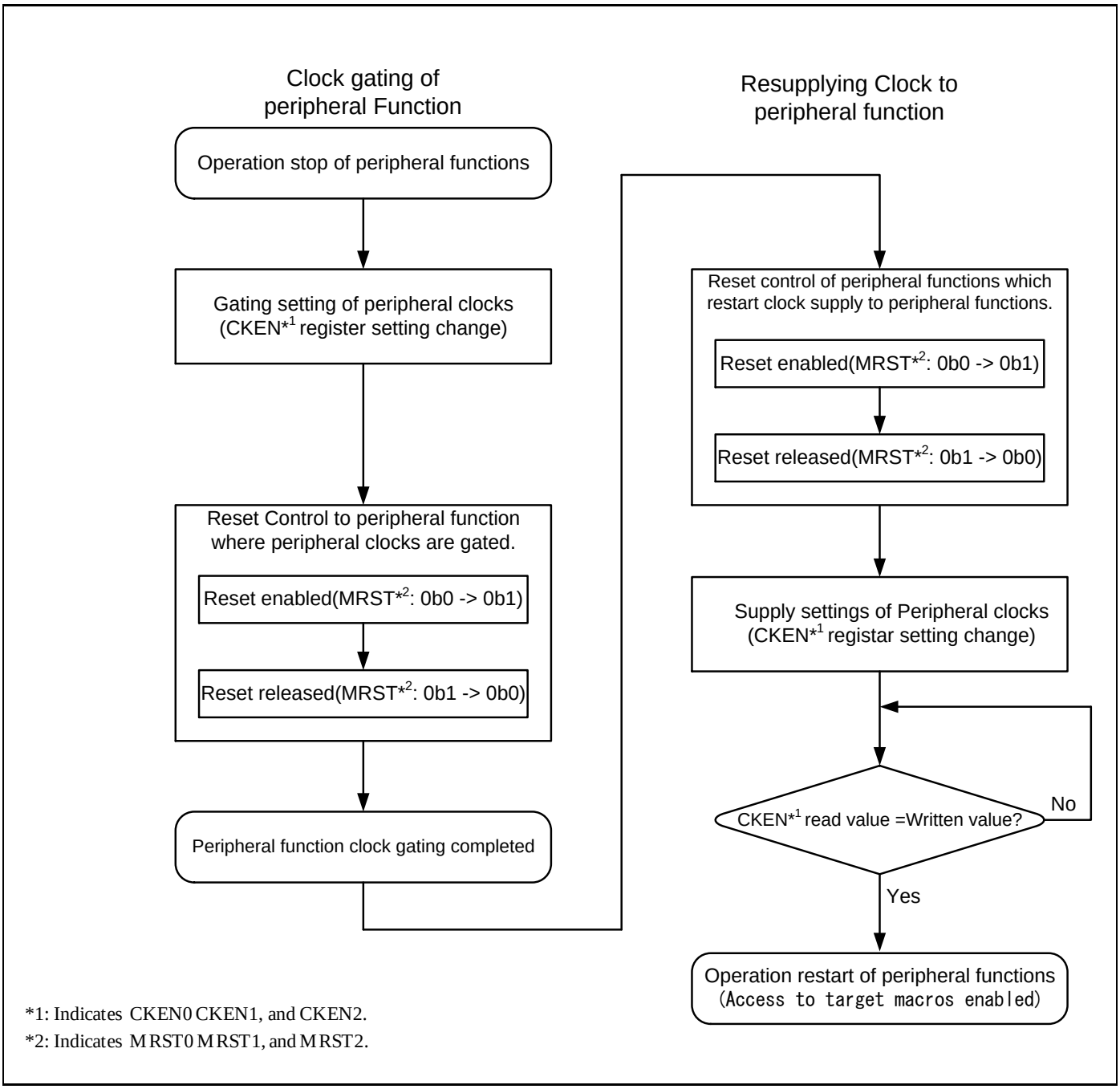
   The peripheral clock registers (CKEN0, CKEN1, and CKEN2) updates the register value to the written value at the step of starting the clock supply to the peripheral function to which the setting is changed.

   Be sure to start the access to the peripheral function after setting a change in the above-mentioned Item 2, reading this register, and then confirming the agreement with the written value because an access to peripheral function is invalid at clock gating.

### Procedures of gating and resupplying clocks

Figure 3-2 explains the procedures of gating the clocks of peripheral functions and resupplying clocks to peripheral functions.

**Figure 3-2 Procedures of Gating Clocks of Peripheral Functions and Resupplying Clocks to Peripheral Functions**




*1: Indicates CKEN0 CKEN1, and CKEN2.
*2: Indicates MRST0 MRST1, and MRST2.

■ Clock gating of peripheral functions

1. Gating setting of peripheral clocks

For the peripheral clock control registers (CKEN0 CKEN1, and CKEN2), change the bit corresponding to the peripheral function for which the clock supply is to be stopped to "0".

After gating the clock to the peripheral function to which the clock gating is instructed, the peripheral clock control registers (CKEN0 CKEN1, and CKEN2) updates the register value to the written value.

2. Reset control to peripheral functions whose peripheral clocks are gated

For the peripheral functions whose clocks are gated, to reset their internal state, execute the reset control of each peripheral function according to the following procedures.

Reset enabled :

Write "1" to the target bits of peripheral function reset control registers (MRST0 MRST1, and MRST2).

Reset released:

Write "0" to the target bits of peripheral function reset control registers (MRST0MRST1, and MRST2).

■ Resupplying clocks to peripheral functions

1. Reset control of peripheral functions which restart clock supply to peripheral functions

For the peripheral functions which gate the peripheral clocks, execute the reset control to each peripheral function by using peripheral function reset control registers (MRST0, MRST1, and MRST2) before restarting their operation. The procedures are the same as the above-mentioned procedures of reset control immediately after peripheral clocks gated.

2. Supply settings of peripheral clocks

For the peripheral clock control registers (CKEN0 CKEN1, and CKEN2), change the settings of bit corresponding to the peripheral function for which the clock is to be resupplied.

At this time, do not set the bit where the peripheral function is not provided and the bit whose bus clock has been gated to the values other than the initial value The reason is that the read value cannot coincide with the written value not to get out of the processing loop at the register set value confirmation in the following Item 3.

3. Confirmation of set values of peripheral clock control registers

At the step where the clock setting change is reflected to the peripheral function whose settings are changed, the peripheral clock control registers (CKEN0 CKEN1, and CKEN2) updates the register value to the written values.

Be sure to start the access to the peripheral function after executing the setting change of the above-mentioned Item 2 , reading the register, and then confirming the agreement with the written value because the access to the peripheral functions is invalid at clock gating.

# 4. Peripheral Clock Gating Function Registers

This section explains each register function of the peripheral clock gating functions.

Table 4-1 shows the list of registers of peripheral clock gating functions.

**Table 4-1 Registers of the Peripheral Clock Gating Functions**

| Abbreviated register name | Register name | Reference |
|---|---|---|
| CKEN0 | Peripheral Function Clock Control Register 0 | 4.1 |
| MRST0 | Peripheral Function Reset Control Register 0 | 4.2 |
| CKEN1 | Peripheral Function Clock Control Register 1 | 4.3 |
| MRST1 | Peripheral Function Reset Control Register 1 | 4.4 |
| CKEN2 | Peripheral Function Clock Control Register 2 | 4.5 |
| MRST2 | Peripheral Function Reset Control Register 2 | 4.6 |

## 4.1 Peripheral Function Clock Control Register 0 (CKEN0)

This section explains Peripheral Function Reset Clock Register 0 (CKEN0).

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | GIOCK | Reserved | | | DMACK |
| Attribute | - | | | R/W | - | | | R/W |
| Initial value | - | | | 1 | - | | | 1 |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | ADCCK[3:0] | | | |
| Attribute | - | | | | R/W | | | |
| Initial value | - | | | | 1111 | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | MFSCK[15:8] | | | | | | | |
| Attribute | R/W | | | | | | | |
| Initial value | 0xFF | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | MFSCK[7:0] | | | | | | | |
| Attribute | R/W | | | | | | | |
| Initial value | 0xFF | | | | | | | |

**[bit31:29] Reserved: Reserved bits**

Write ”0”.to these bits.

**[bit28] GIOCK: Software clock control of GPIO/Fast GPIO function**

This bit controls the operation clock supplying and the gating to the I/O Port function. This bit controls all the operation clocks to the I/O Port functions collectively.

When this bit is set to "1", the bus clock is supplied to the I/O Port function block and the I/O Port function can be used.

When this bit is set to "0", the bus clock input to the I/O Port function block is gated. Note that the reading of the input level and the setting change of the output power level are disabled while the bus clock is gated. For details, see "5. Peripheral Clock Gating Functions Usage Precautions".

| bit | Description |
|---|---|
| 0 | The bus clock input to the I/O Port function block is gated. |
| 1 | The bus clock is supplied to the I/O Port function block. (Initial value)<br>Be sure to set "1" in order to use I/O Port function. |

**[bit27:25] Reserved: Reserved bit**

Write”0” to this bit.

**[bit24] DMACK: Supplying and gating settings of DMAC operation clock**

This bit controls the operation clock supplying and the gating to the DMAC function. When this bit is set to"1", the bus clock is supplied to the DMAC block and the DMAC function can be used.
When this bit is set to "0", the bus clock input to the DMAC block is gated. While the bus clock input is gated, the DMAC function cannot be used.

| bit | Description |
|---|---|
| 0 | The bus clock input to DMAC is gated. |
| 1 | The bus clock is supplied to DMAC. (Initial value) |

**[bit23:20] Reserved: Reserved bits**

Write"0" to these bits.

**[bit19:16] ADCCK[3:0]: Settings for operation clock supplying and gating to AD converter**

These bits control the operation clock supplying and gating to the AD converter. The following show the correspondence between each bit and the AD converter unit:

bit16 - ADCCK0 : AD converter unit 0
bit17 - ADCCK1 : AD converter unit 1
bit18 - ADCCK2 : AD converter unit 2
bit19 - ADCCK3 : AD converter unit 3

When the relevant bit is set to "1", the bus clock is supplied to the unit of the corresponding AD converter to enable the AD converter function. For products to which the corresponding AD converter is not mounted, do not change the relevant bit from the initial value.
When the relevant bit is set to "0", the bus clock input to the corresponding AD converter is gated. While the bus clock input is gated, the relevant AD converter cannot be used.

| bit | Description |
|---|---|
| 0 | The bus clock input to the A/D converter unit corresponding to the relevant bit is gated. |
| 1 | The bus clock is supplied to the A/D converter unit corresponding to the relevant bit.<br>(Initial value) |

**[bit15:0] MFSCK[15:0]: Settings for operation clock supply and gating to multi-function serial interface**

These bits control the operation clock supply and gating to the multi-function serial interface. The correspondence between each bit and the channel is shown below:

bit0 - MFSCK0 : Multi-function serial interface channel 0
bit1 - MFSCK1 : Multi-function serial interface channel 1
bit2 - MFSCK2 : Multi-function serial interface channel 2
bit3 - MFSCK3 : Multi-function serial interface channel 3
bit4 - MFSCK4 : Multi-function serial interface channel 4
bit5 - MFSCK5 : Multi-function serial interface channel 5
bit6 - MFSCK6 : Multi-function serial interface channel 6
bit7 - MFSCK7 : Multi-function serial interface channel 7
bit8 - MFSCK8 : Multi-function serial interface channel 8
bit9 - MFSCK9 : Multi-function serial interface channel 9
bit10 - MFSCK10 : Multi-function serial interface channel 10
bit11 - MFSCK11 : Multi-function serial interface channel 11
bit12 - MFSCK12 : Multi-function serial interface channel 12
bit13 - MFSCK13 : Multi-function serial interface channel 13
bit14 - MFSCK14 : Multi-function serial interface channel 14
bit15 - MFSCK15 : Multi-function serial interface channel 15

When the relevant bit is set to "1", the bus clock is supplied to the channel of the corresponding multi-function serial interface to enable the function of the multi-function serial interface. For products to which the relevant multi-function serial interface channel is not mounted, do not change the relevant bit from the initial value.
When the relevant bi is set to "0", the bus clock input to the channel of the corresponding multi-function serial interface is gated. While the bus clock input is gated, the multi-function serial interface function of the corresponding channel cannot be used.

| bit | Description |
|---|---|
| 0 | The bus clock input to the multi-function serial interface channel corresponding to the relevant bit is gated. |
| 1 | The bus clock is supplied to the multi-function serial interface channel corresponding to the relevant bit. (Initial value) |

## 4.2 Peripheral Reset Control Register 0 (MRST0)

This section explains the peripheral reset control register 0 (MRST0).

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | DMARST |
| Attribute | - | | | | | | | R/W |
| Initial value | - | | | | | | | 0 |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | ADCRST[3:0] | | | |
| Attribute | - | | | | R/W | | | |
| Initial value | - | | | | 0000 | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | MFSRST [15:8] | | | | | | | |
| Attribute | R/W | | | | | | | |
| Initial value | 0x00 | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | MFSRST [7:0] | | | | | | | |
| Attribute | R/W | | | | | | | |
| Initial value | 0x00 | | | | | | | |

**[bit31:25] Reserved: Reserved bits**

Write"0" to these bits.

**[bit24] DMARST: Reset control of DMAC**

This bit controls reset of the DMAC unit. If this bit is set to "1", DMAC becomes a reset state, the DMA transfer operation stops, and all the register settings are initialized. To release the reset state, be sure to set this bit to "0" again.

| bit | Description |
|---|---|
| 0 | Releases the DMAC reset. (Initial value) |
| 1 | Issues reset signal to DMAC. |

**[bit23:20] Reserved: Reserved bits**

Write"0" to these bits.

**[bit19:16] ADCRST[3:0]: Reset control of AD converter**

These bits control the reset of each unit of the AD converter. The correspondence between each bit and AD converter unit is shown below:

bit16 - ADCRST 0 : AD converter unit 0
bit17 - ADCRST 1 : AD converter unit 1
bit18 - ADCRST 2 : AD converter unit 2
bit19 - ADCRST 3 : AD converter unit 3

If the relevant bit is set to "1", the corresponding AD converter unit becomes a reset state, the analog to digital conversion operation stops, and the register settings are initialized. For products to which the relevant AD converter unit is not mounted, do not change the relevant bit from the initial state. To release the reset state, be sure to set this bit to "0" again.

| bit | Description |
|---|---|
| 0 | Releases the reset of the AD converter unit corresponding to the relevant bit. (Initial value) |
| 1 | Issues the reset to the AD converter unit corresponding to the relevant bit. |

**[bit15:0] MFSRST[15:0]: Control of software reset of multi-function serial interface**

These bits control the reset of each channel of the multi-function serial interface. The correspondence between each bit and the channel is shown below.

bit0 - MFSRST0: Multi-function serial interface channel 0
bit1 - MFSRST1: Multi-function serial interface channel 1
bit2 - MFSRST2 : Multi-function serial interface channel 2
bit3 - MFSRST3 : Multi-function serial interface channel 3
bit4 - MFSRST4 : Multi-function serial interface channel 4
bit5 - MFSRST5 : Multi-function serial interface channel 5
bit6 - MFSRST6 : Multi-function serial interface channel 6
bit7 - MFSRST7 : Multi-function serial interface channel 7
bit8 - MFSRST8 : Multi-function serial interface channel 8
bit9 - MFSRST9 : Multi-function serial interface channel 9
bit10 - MFSRST10 : Multi-function serial interface channel 10
bit11 - MFSRST11 : Multi-function serial interface channel 11
bit12 - MFSRST12 : Multi-function serial interface channel 12
bit13 - MFSRST13 : Multi-function serial interface channel 13
bit14 - MFSRST14 : Multi-function serial interface channel 14
bit15 - MFSRST15 : Multi-function serial interface channel 15

If the relevant bit is set to "1", the channel of the corresponding multi-function serial interface becomes a reset state, its serial communications stop, and the register setting is initialized. For products to which the relevant multi-function serial interface channel is not mounted, it is prohibited to set the relevant bit to "1". To release the above-mentioned reset, be sure to set this bit to "0" again.

| bit | Description |
|---|---|
| 0 | Releases the reset of the multi-function serial interface channel corresponding to the relevant bit. (Initial value) |
| 1 | Issues the reset the multi-function serial interface channel corresponding to the relevant bit. |

## 4.3 Peripheral Clock Control Register 1 (CKEN1)

This section explains the peripheral clock control register 1 (CKEN1).

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | QDUCK[3:0] | | | |
| Attribute | - | | | | R/W | | | |
| Initial value | - | | | | 1111 | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | MFTCK[3:0] | | | |
| Attribute | - | | | | R/W | | | |
| Initial value | - | | | | 1111 | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | BTMCK[3:0] | | | |
| Attribute | - | | | | R/W | | | |
| Initial value | - | | | | 1111 | | | |

**[bit31:20] Reserved: Reserved bits**
Write"0" to these bits.

**[bit19:16] QDUCK[3:0]: Settings for operation clock supply and gating of quad counter**
These bits control the operation clock supply and gating of quad counter. The correspondence between each bit and quad counter is shown below.

bit16 - QDUCK0: Quad counter unit 0
bit17 - QDUCK1: Quad counter unit 1
bit18 - QDUCK2: Quad counter unit 2
bit19 - QDUCK3: Quad counter unit 3

When the relevant bit is set to "1", the bus clock is supplied to the unit of the corresponding quad counter to use the quad counter function. For products to which the relevant quad counter unit is not mounted, do not change the relevant bit from the initial value.
When the relevant bit is set to "0", the bus clock input to corresponding quad counter is stopped. While the bus clock input is gated, the quad counter of the relevant unit cannot be used.

| bit | Description |
|---|---|
| 0 | Gates the bus clock input to the corresponding quad counter. |
| 1 | Supplies the bus clock to the quad counter corresponding to the relevant bit. (Initial value) |

**[bit15:12]Reserved: Reserved bits**
Write"0" to these bits.

**[bit11:8] MFTCK[3:0]: Settings for operation clock supply and gating of multi-function timer and PPG**

These bits control the operation clock supply and gating to the multi-function timer and PPG. The correspondence among each bit, the multi-function timer unit, and the PPG channel is shown below.

bit8 - MFTCK0 : Multi-function timer unit 0 - PPG channels 0, 2, 4, 6
bit9 - MFTCK1 : Multi-function timer unit 1 - PPG channels 8, 10, 12, 14
bit10 - MFTCK2 : Multi-function timer unit 2 - PPG channels 16, 18, 20, 22
bit11 - MFTCK3 : Multi-function timer unit 3 - PPG channels 24, 26, 28, 30

When the relevant bit is set to "1", the bus clock is supplied to corresponding multi-function timer unit and PPG channels to use the multi-function timer and PPG function. For products to which the relevant multi-function timer unit and PPG channels are not mounted, it is prohibited to change the relevant bit from the initial value.
When the relevant bit is set to "0", the bus clock input to the corresponding multi-function timer unit and PPG channels is gated. While the bus clock is gated, the relevant multi-function timer and PPG function cannot be used.

| bit | Description |
|---|---|
| 0 | The bus clock input to the multi-function timer unit and the PPG channel corresponding to the relevant bit is gated. |
| 1 | The bus clock is supplied to the multi-function timer unit and the PPG channel corresponding to the relevant bit. (Initial value) |

**[bit7:4] Reserved: Reserved bits**

Write"0" to these bits.

**[bit3:0] BTMCK[3:0]: Settings operation clock supply and gating to base timer**

These bits control the operation clock supply and gating to the base timer. The correspondence between each bit and the base timer channels is shown below.

bit0 - BTMCK0 : Base timer channel 0, 1, 2, 3
bit1 - BTMCK1 : Base timer channel 4, 5, 6, 7
bit2 - BTMCK2 : Base timer channel 8, 9, 10, 11
bit3 - BTMCK3 : Base timer channel 12, 13, 14, 15

When the relevant bit is set to "1", the bus clock is supplied to the corresponding base timer channel to use the base timer, do not change the relevant bit from the initial value.
When "0" is set to the relevant bit, the bus clock input to the corresponding base timer channel is gated. While the bus clock input is gated, the base timer function of the corresponding channel cannot be used.

| bit | Description |
|---|---|
| 0 | The bus clock input to the base timer channel corresponding to the relevant bit is gated. |
| 1 | The bus clock is supplied to the base timer channel corresponding to the relevant bit. (Initial value) |

## 4.4 Peripheral Function Reset Control Register 1 (MRST1)

This section explains the peripheral function reset control register 1(MRST1).

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | QDURST[3:0] | | | |
| Attribute | - | | | | R/W | | | |
| Initial value | - | | | | 0000 | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | MFTRST[3:0] | | | |
| Attribute | - | | | | R/W | | | |
| Initial value | - | | | | 0000 | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | BTMRST[3:0] | | | |
| Attribute | - | | | | R/W | | | |
| Initial value | - | | | | 0000 | | | |

**[bit31:20] Reserved: Reserved bits**
Write"0" to these bits.

**[bit19:16] QDURST[3:0] : Reset control of quad counter**
These bits control the reset of each unit of the quad counter. The correspondence between each bit and the quad counter unit is shown below.

bit16 - QDURST 0 : Quad counter unit 0
bit17 - QDURST 1 : Quad counter unit 1
bit18 - QDURST 2 : Quad counter unit 2
bit19 - QDURST 3 : Quad counter unit 3

If the relevant bit is set to "1", the unit of the corresponding quad counter becomes a reset state, the quad counter operation stops, and the register settings are initialized. For products to which the relevant quad counter is not mounted, do not change the relevant bit from the initial state. To release the reset state, be sure to set this bit to "0" again.

| bit | Description |
|---|---|
| 0 | Releases the reset of the quad counter corresponding to the relevant bit. (Initial value) |
| 1 | Issues the reset to the quad counter unit corresponding to the relevant bit. |

**[bit15:12] Reserved: Reserved bits**
Write"0" to these bits.

**[bit11:8] MFTRST[3:0] : Control of multi-function timer and PPG reset control**

These bits control multi-function timer reset of each unit and PPG reset of every four channels. The correspondence among each bit, quad counter unit, and the PPG channel is shown below.

bit8 - MFTRST0: Multi-function timer unit 0 - PPG channel 0, 2, 4, 6
bit9 - MFTRST1: Multi-function timer unit 1 - PPG channel 8, 10, 12, 14
bit10 - MFTRST2: Multi-function timer unit 2 - PPG channel 16, 18, 20, 22
bit11 - MFTRST3 Multi-function timer unit 3 - PPG channel 24, 26, 28, 30

If the relevant bit is set to "1", the corresponding multi-function timer unit and PPG channel become the reset states, the multi-function timer operation stops, and the register setting is initialized. For products to which the relevant multi-function timer unit and PPG channels are not mounted, do not change the relevant bit from the initial value. To release the reset state, be sure to set this bit to "0" again.

| bit | Description |
|---|---|
| 0 | Release the resets of the multi-function timer unit and the PPG channel corresponding to the relevant bit. (Initial value) |
| 1 | Issue the resets to the multi-function timer unit and the PPG channels corresponding to the relevant bit. |

**[bit7:4]Reserved: Reserved bits**

Write"0" to these bits.

**[bit3:0] BTMRST[3:0] : Reset control of base timer**

These bits control the reset for four units of the base timer. The correspondence among each bit and the base timer channels is shown below.

bit0 - BTMRST0: Base timer channels 0, 1, 2, 3
bit1 - BTMRST1: Base timer channels 4, 5, 6, 7
bit2 - BTMRST2: Base timer channels 8, 9, 10, 11
bit3 - BTMRST3: Base timer channels 12, 13, 14, 15

If the relevant bit is set to "1", the unit of the corresponding base timer channels becomes a reset state, the base timer operation stops, and the register setting is initialized. For products to which the relevant base timer channels are not mounted, do not change the relevant bit from the initial value. To release the reset state, be sure to set this bit to "0" again.

| bit | Description |
|---|---|
| 0 | Release the reset for the base timer channel corresponding to the relevant bit. (Initial value) |
| 1 | Issue the reset to the base timer channel corresponding to the relevant bit. |

## 4.5 Peripheral Clock Control Register 2 (CKEN2)

This section explains the peripheral clock control register 2(CKEN2).

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | CANCK[1:0] | | Reserved | | | |
| Attribute | - | | R/W* | | - | | | |
| Initial value | - | | 11* | | - | | | |

* : For products not mounting CAN controller, Attribute is "R" and Initial value is "00".

**[bit31:6] Reserved: Reserved bits**
Write"0" to these bits.

**[bit5:4] CANCK[1:0] : Settings for clock supply and gating to CAN controller**
These bits control bus clock (base clock) supply and gating to the CAN controller. The correspondence between each bit and the CAN controller channel is shown below.

bit4 - CANCK0: CAN controller channel 0
bit5 - CANCK1: CAN controller channel 1

When the relevant bit is set to "1", the bus clock is supplied to the corresponding CAN controller channel to use the CAN controller function. For products to which the relevant CAN controller channel is not mounted, do not change the relevant bit from the initial value.
When the relevant bit is set to "0", the bus clock input to the corresponding CAN controller channel is gated. While the bus clock input is gated, the CAN controller function of the corresponding channel cannot be used.

| bit | Description |
|---|---|
| 0 | Gates the bus clock input to the CAN controller channel corresponding to the relevant bit. (Initial value: For products not mounting CAN controller) |
| 1 | Supplies the bus clock input to the CAN controller channel corresponding to the relevant bit. (Initial value: For products mounting CAN controller) |

**[bit3:0] Reserved: Reserved bits**
Write"0" to these bits.

## 4.6 Peripheral Function Reset Control Reset 2 (MRST2)

This section explains the peripheral function reset control register 2 (MRST2).

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | CANRST[1:0] | | Reserved | | | |
| Attribute | - | | R/W | | - | | | |
| Initial value | - | | 00 | | - | | | |

**[bit31:6] Reserved: Reserved bits**
Write"0" to these bits.

**[bit5:4] CANRST[1:0] : Reset control of CAN controller**
These bits control the reset of each CAN controller's channel unit. The correspondence between each bit and the CAN controller channel is shown below.

bit4 - CANRST0: CAN controller channel 0
bit5 - CANRST1 : CAN controller channel 1

If the relevant bit is set to "1", the channel of the corresponding CAN controller becomes a reset state, the CAN controller operation stops, and the register settings are initialized. For products to which the CAN controller channel is not mounted, do not change this bit from the initial value. To release the reset state, be sure to set this bit to "0" again.

| bit | Description |
|---|---|
| 0 | Releases the reset of CAN controller channel corresponding to the relevant bit. (Initial value) |
| 1 | Issues the rest signal to CAN controller channel corresponding to the relevant bit. |

**[bit3:0] Reserved: Reserved bits**
Write"0" to these bits.

# 5. Peripheral Clock Gating Function Usage Precautions

This section explains the precautions for using peripheral clock gating functions by peripheral function.

### Overview

■ Control of a peripheral function to which a clock supply is stopped

The register access to a peripheral function to which a clock supply is stopped, both read and write, is not guaranteed. The read value is undefined, and the write operation is prohibited.
The internal state can be reset by controlling peripheral function reset control register (MRST0 and MRST1 and MRST2) while the peripheral clock is gated.

■ Combination of peripheral clock settings

Be sure to set all the target peripheral functions to the clock supply side by the peripheral clock registers (CKEN0, CKEN1 and CKEN2) for the functions operated by combining two or more peripheral functions. For example, set a relevant unit of the A/D converter used and a relevant channel of the base timer to the clock supply side respectively by the peripheral clock control registers (CKEN0 and CKEN1) when the base timer is selected for use by the timer trigger of the A/D converter.

■ Initialization conditions of peripheral clock settings

The peripheral clock gating function is initialized by the following reset. After issuing the following reset, be sure to reconfigure the peripheral clock gating function.
For details of the following resets, see Chapter "Reset".

- Power-on reset (PONR)
- Low voltage detection reset (LVDH)
- INITX pin input(INITX)
- Software watchdog reset (SWDGR)
- Hardware watchdog reset (HWDGR)
- Clock failure detection reset (CSVR)
- Anomalous frequency detection reset (FCSR)
- Software reset (SRST)
- APB1 bus reset (APBC1_PSR)
- Deep standby transition reset (DSTR)

### Multi-Function Serial Interface

■ LIN Sync field detection: LSYN

Execute the setting of the operation clock supply to the corresponding multi-function timer (input capture) separately with the setting of the peripheral clock of multi-function serial interface when the input capture (ICU) is used in the LIN bus interface mode. For the connection between the multi-function serial interface and the input capture, see "Extended Pin Function Setting Register (EPFR)" in Chapter of "I/O port" of "FM0+ Peripheral Manual".

### Base Timer

■ Clock setting unit of base timer

The peripheral clock control of the base timer is executed in the unit of four channels described in Table 5-1.

**Table 5-1 Correspondence between Peripheral Clock Gating Setting and Base Timer Channels**

| Setting bit of Peripheral Clock Control Register (CKEN1) | Target Channels |
|---|---|
| bit 0 | Base Timer ch3, ch2, ch1, ch0 |
| bit 1 | Base Timer ch7, ch6, ch5, ch4 |
| bit 2 | Base Timer ch11, ch10, ch9, ch8 |
| bit 3 | Base Timer ch15, ch14, ch13, ch12 |

### Multi-function timer

### FRT Selection register

For using the following FRT selection function, set the operation clock of the multi-function timer unit on which source-side FRT is mounted to the supply side。

- OCU Connection FRT selection register (OCFS)
- ICU Connection FRT selection register (ICFS)
- ADC Start-up compare connection FRT selection register (ADCMP)

### PPG

■ Clock Control of PPG

The control of input clock to PPG synchronizes with the settings of input clock to the multi-function timer. For PPG channel numbers and unit numbers of multi-function timer, see Table 5-2.

**Table 5-2 Multi-function Timer and PPG Input Clock Control**

| Unit number of multi-function timer | PPG channel number |
|---|---|
| Unit 0 | Channel 0,1,2,3,4,5,6,7 |
| Unit 1 | Channel 8,9,10,11,12,13,14,15 |
| Unit 2 | Channel 16,17,18,19,20,21,22,23 |
| Unit 3 | Channel 24,25,26,27,28,29,30,31 |

### A/D Converter

■ A/D Timer Trigger Selection

When the base timer is used as a startup trigger of the A/D converter, set the operation clock of the selected base timer channel to the supply side.

### GPIO/Fast GPIO

■ Restrictions when bus clock is gated

While the bus clock of GPIO/Fast GPIO is gated, some functions of I/O port cannot be used as shown in Table 5-3.

Be sure to confirm the using conditions and execute the bus clock control of GPIO/Fast GPIO.

For details on I/O port functions, see Chapter "I/O Port" and "Fast GPIO"

**Table 5-3 Restrictions when GPIO/Fast GPIO clock is gated**

| Restrictions | Bus Clock Status | |
|---|---|---|
| | Supplied* | Gated* |
| I/O port Function-Input level reading<br>(PDIR/FPDIR/M_FPDIR register reading) | Available | Prohibited |
| I/O port Function-Output Level Switching<br>and Status Confirmation<br>(PDOR/FPDOR/M_FPDOR register reading/writing) | Available | Prohibited |
| I/O port Mode Switching<br>(Setting change of PFR, PCR, DDR, ADE, SPSR, EPFR, and PZR, FPOER registers) | Available | Prohibited |
| Peripheral Function Operation (Signal Input and Output) | Available | Available |
| External Interrupt/NMI Control | Available | Available |
| Reset Input (INITX) | Available | Available |
| Return from Deep Standby Mode<br>(WKUP pin input) | Available | Available |

*: Available: can be used, 、Prohibited: cannot be used.

# CHAPTER2-3: High-Speed CR Trimming

**This chapter explains the High-Speed CR Trimming Function.**



CODE: 9BFCRTRIM_FM4-E01.0

# 1. High-Speed CR Trimming Function Overview

This section explains frequency trimming function of the high-speed CR oscillator.

The high-speed CR oscillators used for this device have fluctuation range in frequency accuracy due to process variation. The fluctuation range of frequency accuracy due to process variation and temperature change can be reduced by configuring the trimming function.

The high-speed CR trimming function consists of the frequency trimming setup unit and temperature trimming setup unit.

The frequency trimming setup unit has the following functions:

- It can be configured the high-speed CR frequency trimming by writing a trimming value to the High-speed CR oscillation Frequency Trimming Register (MCR_FTRM).
- By using ch.0 of Base Timer, the setting value to the frequency trimming register can be calculated from count value of the specified period.

The temperature trimming setup unit has the following function:

It can be configured the high-speed CR temperature compensation by writing a trimming value to the High-speed CR oscillation Temperature Trimming Register (MCR_TTRM).

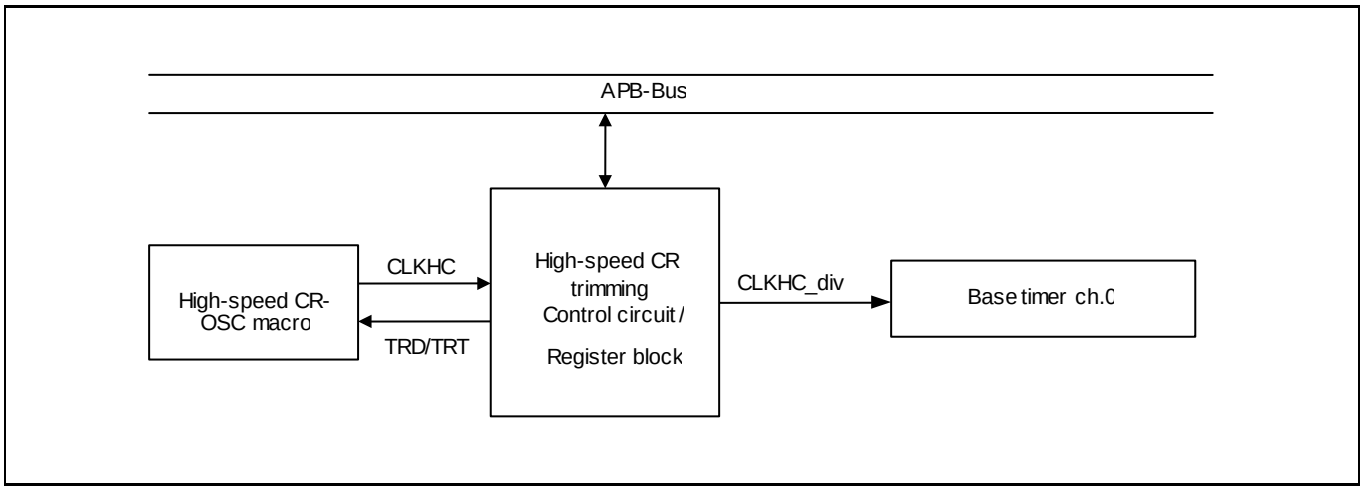
For the high-speed CR frequency accuracy, see electrical characteristics described in "Data Sheet" of the product used.

# 2. High-Speed CR Trimming Function Configuration and Block Diagram

This section explains the configuration and block diagram of high-speed CR oscillator frequency trimming function.

Figure 2-1 shows the block diagram of high-speed CR frequency trimming function.

**Figure 2-1 Block diagram of the High-speed CR Oscillator Timing Circuit**




**Configuration**

- High-speed CR OSC macro

A macro of the high-speed CR clock outputs CLKHC (high-speed CR clock).

In addition, the frequency trimming can be performed with TRD bit of high-speed CR oscillation frequency trimming register (MCR_FTRM) and TRT bit of high-speed CR oscillation temperature trimming register (MCR_TTRM).

- High-speed CR Trimming Control Circuit and register block

A control circuit and registers for trimming high-speed CR.

In addition, the high-speed CR clock (CLKHC_div) divided by the ratio set with CSR bit of high-speed CR oscillation frequency division setup register (MCR_PSR) is output to the base timer ch.0.

- Base timer

This block counts frequency before setting to calculate the frequency trimming data for high-speed CR.

***Note:***

- *For the clock definition, see Chapter "Clock".*

# 3. High-Speed CR Trimming Function Operation

This section explains operation conducted by frequency trimming function of the high-speed CR oscillator.

**Operation of high-speed CR oscillation frequency trimming function**

■ Frequency trimming setup

The setup process writes a trimming data value to the High-speed CR oscillation Frequency Trimming Register (MCR_FTRM) to correct the misalignment of high-speed CR clock accuracy caused by process variation.

■ Temperature trimming setup

The setup process writes a trimming data value to the High-speed CR oscillation Temperature Trimming Register (MCR_TTRM) to correct the misalignment of high-speed CR clock accuracy caused by temperature change.

■ Register lock function

Write protect function is provided for the High-speed CR oscillation Frequency Trimming Register (MCR_FTRM) and the High-speed CR oscillation Temperature Trimming Register (MCR_TTRM), a function that protects the register from being rewritten without authorization when the system runs out of control.

■ Trimming data acquisition

Data written to the High-speed CR oscillation Frequency Trimming Register (MCR_FTRM) can be acquired by one of the following three methods:

- Use the factory preset value stored in the "CR trimming" area inside the flash memory.
- Calculate by yourself the value set to the High-speed CR oscillation Frequency Trimming Register from the count value within a certain period by using base timer.
- Output high-speed CR clock to an external pin, monitor the waveform to trim the frequency and calculate by yourself the value set to the High-speed CR oscillation Frequency Trimming Register.

For data written to the High-speed CR oscillation Temperature Trimming Register (MCR_TTRM), use the factory preset value stored in the "CR trimming" area inside the flash memory.

***Notes:***

- *Erasing the flash memory also erases the "CR trimming" area inside the memory at the same time. If you use a value in the "CR trimming" area, therefore, save the data to other area (such as RAM) before erasing the flash memory, or only erase sectors other than in the "CR trimming" area.*
- *For the address of the "CR trimming" area, see "FLASH PROGRAMMING MANUAL" of the product used.*

# 4. High-Speed CR Trimming Function Setup Procedure Example

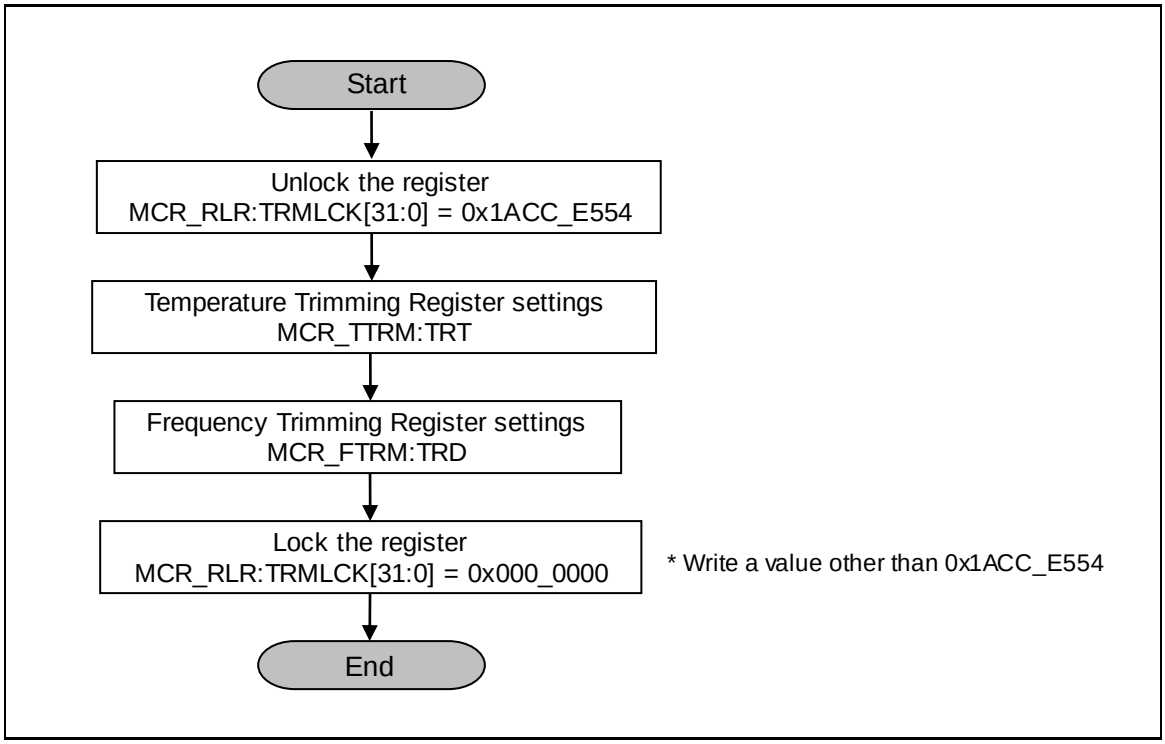
This section provides an example of setting up frequency trimming function of the high-speed CR oscillator.

**Frequency trimming setup**

Take the steps shown in Figure 4-1 to set up frequency trimming.

1. Write "0x1ACCE554" to TRMLCK[31:0] bits of High-speed CR frequency Register Write Protection register (MCR_RLR) to release the lock of Frequency Trimming Setup Register (MCR_FTRM)/ Temperature Trimming Setup Register (MCR_TTRM).
2. Set the trimming data to TRT bit of Temperature Trimming Setup Register (MCR_TTRM).
3. Set TRD bit of Frequency Trimming Setup Register (MCR_FTRM).
4. Write a value other than "0x1ACCE554" to TRMCLK[31:0] bits of High-speed CR Oscillation Register Write Protection Register (MCR_RLR) to lock the Frequency Trimming Setup Register (MCR_FTRM)/ Temperature Trimming Setup Register (MCR_TTRM).

**Figure 4-1 Frequency/Temperature trimming setup**



### Frequency trimming data acquisition example

When acquiring the data from the "CR trimming" area in the flash memory;

Read the "CR trimming" area in the flash memory and get the data.
Write the acquired value to TDR bit of the High-speed CR oscillation Frequency Trimming Setup Register (MCR_FTRM).

### How to calculate the frequency trimming data

The following explains how to calculate the trimming data of high-speed CR oscillation.

1. Let Ftgt, a target oscillation frequency be 4[MHz] and Ttgt, its cycle be 250[ns](Ftgt: 4[MHz]). Let Xtrm_coarse and Xtrm_fine be the TRD[9:5] bit values and TRD[4:0] bit values of the High-speed CR Ocsilation Frequency Trimming Setup register at the time respectively.
2. Set "0b00000" to TRD[4:0] bits.
3. Let Xtrm_coarse be Xtrmmin_coarse when "0b00000" is set to TRD[4:0] bits. Let Tmax_coarse[sec] be the cycle at this time.
4. Let Xtrm_coarse be Xtrmmax_coarse when "0b11111" is set to TRD[9:5] bits. Let Tmin_coarse[sec] be the cycle at this time.
5. By calculating te following expression, obtain TRD[9:5] setting value, Xtrm_coarse giving the value more than target oscilation cycle, Ttgt.

$$Xtrm\_coarse = \frac{Ttgt - \frac{T\max\_coarse - T\min\_corse}{31} - T\max\_coarse}{\frac{T\min\_coarse - T\max\_coarse}{31}}$$

*: Round down decimals.

6. Set the obtained Xtrm_coarse to TRD[9:5] bits.
7. Confirm that the High-speed CR clock, $F_{CRH}$, after setting TRD bits is Ftgt or less. If the $F_{CRH}$ exceed Ftgt, subtract "1" from Xtrm_coarse and then return to Step 6.When the $F_{CRH}$ is Ftgt or less, go to Step 8.
8. Let the value when "0b00000" is set to TRD[4:0] be Xtrimmin_fine. Let Tmax_fine[sec] be the cycle at this time.
9. Let the value when "0b11111" is set to TRD[4:0] be Xtrimmax_fine. Let Tmin_fine[sec] be the cycle at this time.
10. By calculating te following expression, obtain TRD[4:0] setting value, Xtrm_fine giving the target oscilation cycle, Ttgt.

$$Xtrm\_fine = \frac{Ttgt - \frac{T\max\_fine - T\min\_fine}{31} - T\max\_fine}{\frac{T\min\_fine - T\max\_fine}{31}}$$

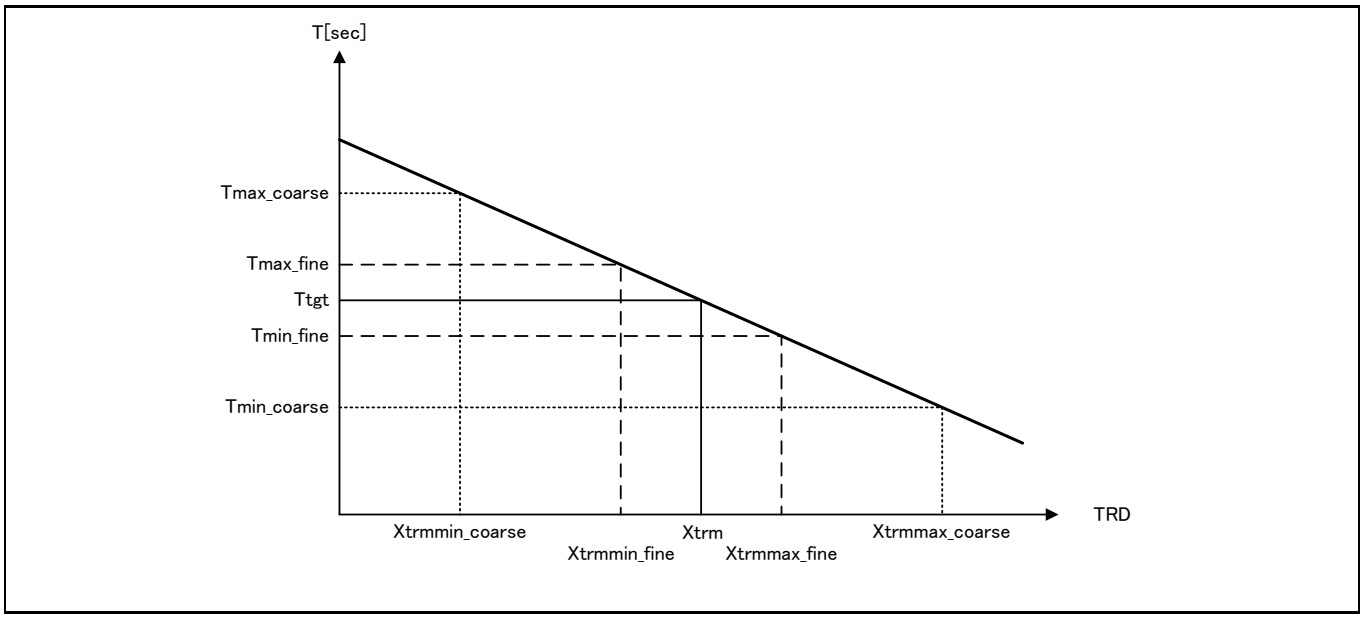
*: Round down decimals.

11. Set the obtained Xtrm_fine to TRD[4:0] bits.
12. Confirm whether the High-speed CR clock, $F_{CRH}$, after setting TRD bits is Ftgt or more and within the specification value of the High-speed CR clock oscillation frequency. If $F_{CRH}$ exceeds the specification value, subtract "1" from Xtrm_fine and return to Step 9. Moreover, if $F_{CRH}$ is less than Ftgt, add "1" to Xtrm_fine and return to Step 11. When the value is within the specification values, the calculation of trimming data is finished.

***Note:***

- *For specifications of High-speed CR Clock Oscillation frequency, see "Data Sheet" of the product used.*

**Figure 4-2 Method to trim high-speed CR clock**



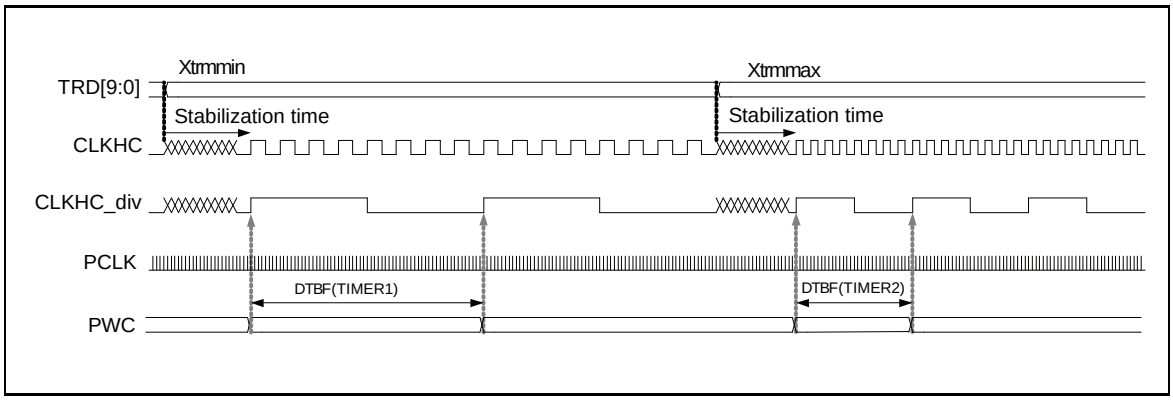

***Note:***

- *For information about how to measure Tmin_coarse/fine and Tmax_coarse/fine, see "■ Example of trimming data acquisition using base timer".*

### Example of trimming data acquisition using base timer

Figure 4-3 shows the time chart of high-speed CR oscillation and the trimming process.

**Figure 4-3 Time chart of high-speed CR oscillation and the trimming process with base timer**




Run the base timer by setting the main oscillation clock (CLKMO) as the master clock (measurement reference clock).

Activate a trigger on the rising of the high-speed CR frequency division clock (CLKHC_div) when setting Xtrmmin or Xtrmmax, read the base timer value at that time, and perform the following calculations.

Tmax = (TIMER1 × PCLK) / DIV

Tmin = (TIMER2 × PCLK) / DIV

- TIMER1, TIMER: Count value of base timer (PWC)
- PCLK: APB1 bus clock
- DIV: Frequency division ratio set by CSR bit of Division Setting Register(MCR_PSR)

Example: When PCLK = 40 MHz (25 ns), frequency division ratio = 1/8, and TIMER1 = 100,

Tmax = (100 × 25 ns) / 8 = 312.5 ns

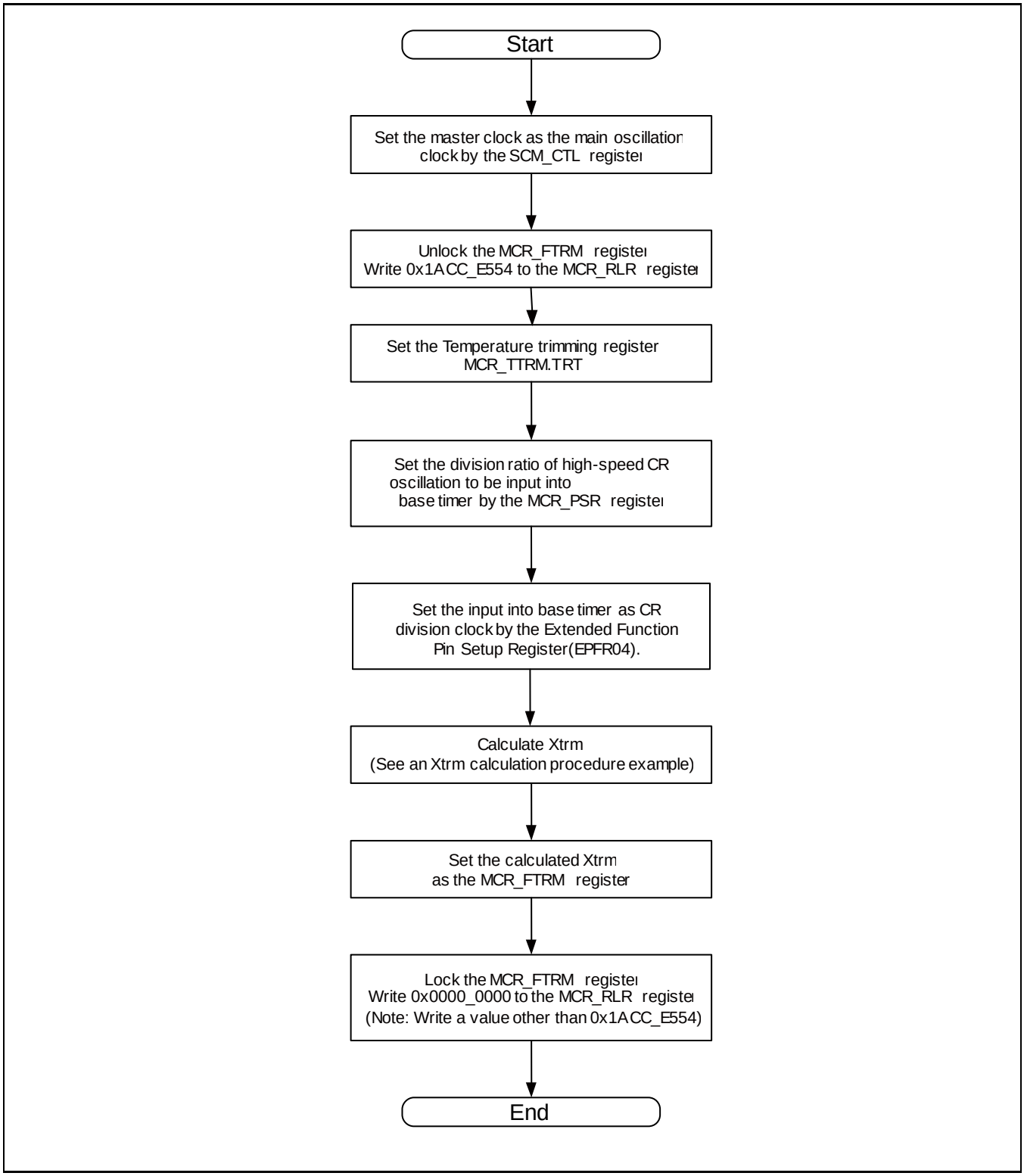
***Note:***

- *The base timer used for trimming is ch.0.*
  *PCLK in Figure 4-3 is an APB1 bus clock.*
  *At this time, select the master clock as the main oscillation for PCLK.*

### Frequency trimming procedure example

Figure 4-4 shows a trimming procedure example of high-speed CR oscillation.

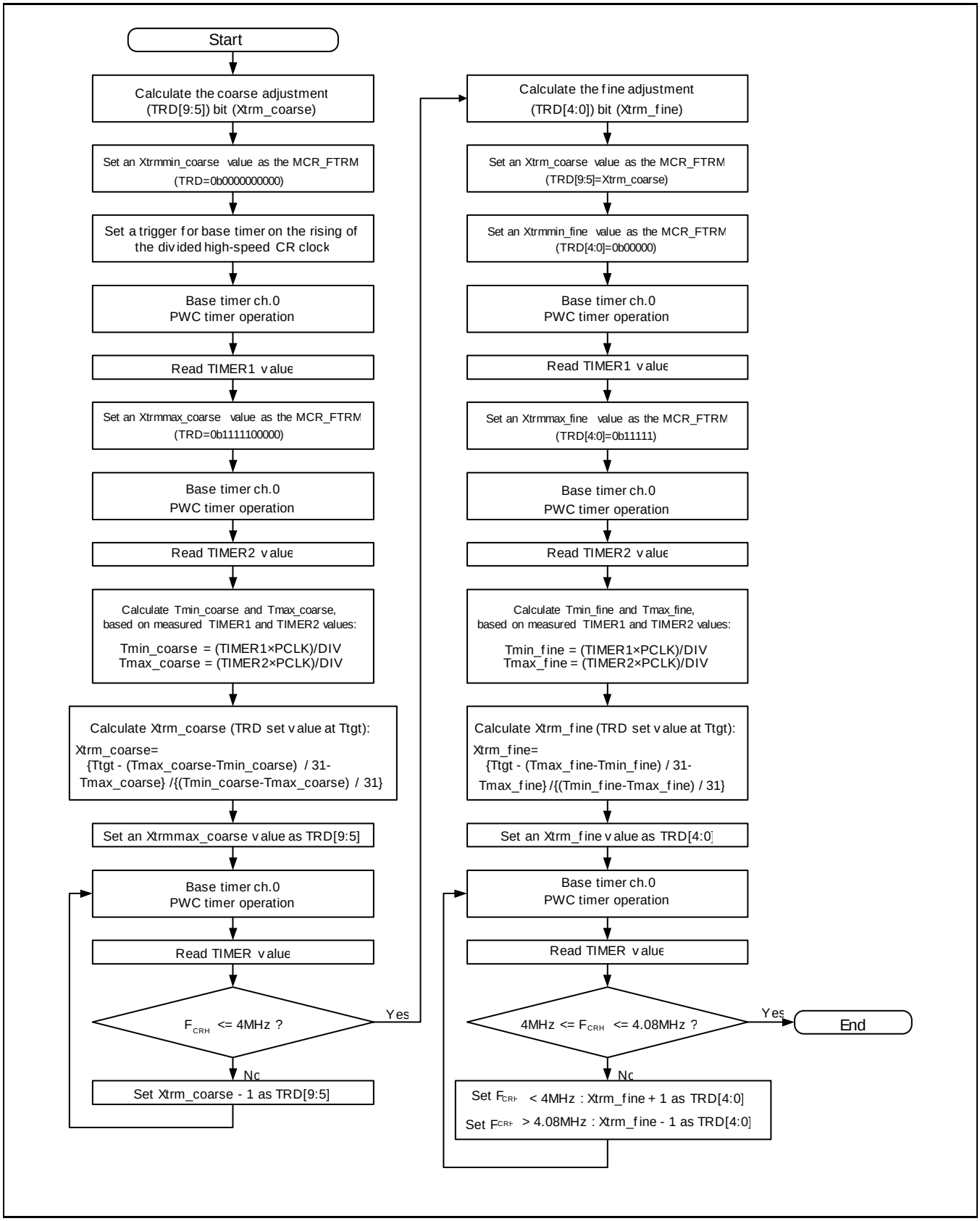
**Figure 4-4 Trimming Procedure Example of High-speed CR Oscillation**



### Xtrm calculation procedure example

Figure 4-5 shows an Xtrm calculation procedure example. Perform frequency trimming in the two stages of coarse adjustment and fine adjustment.

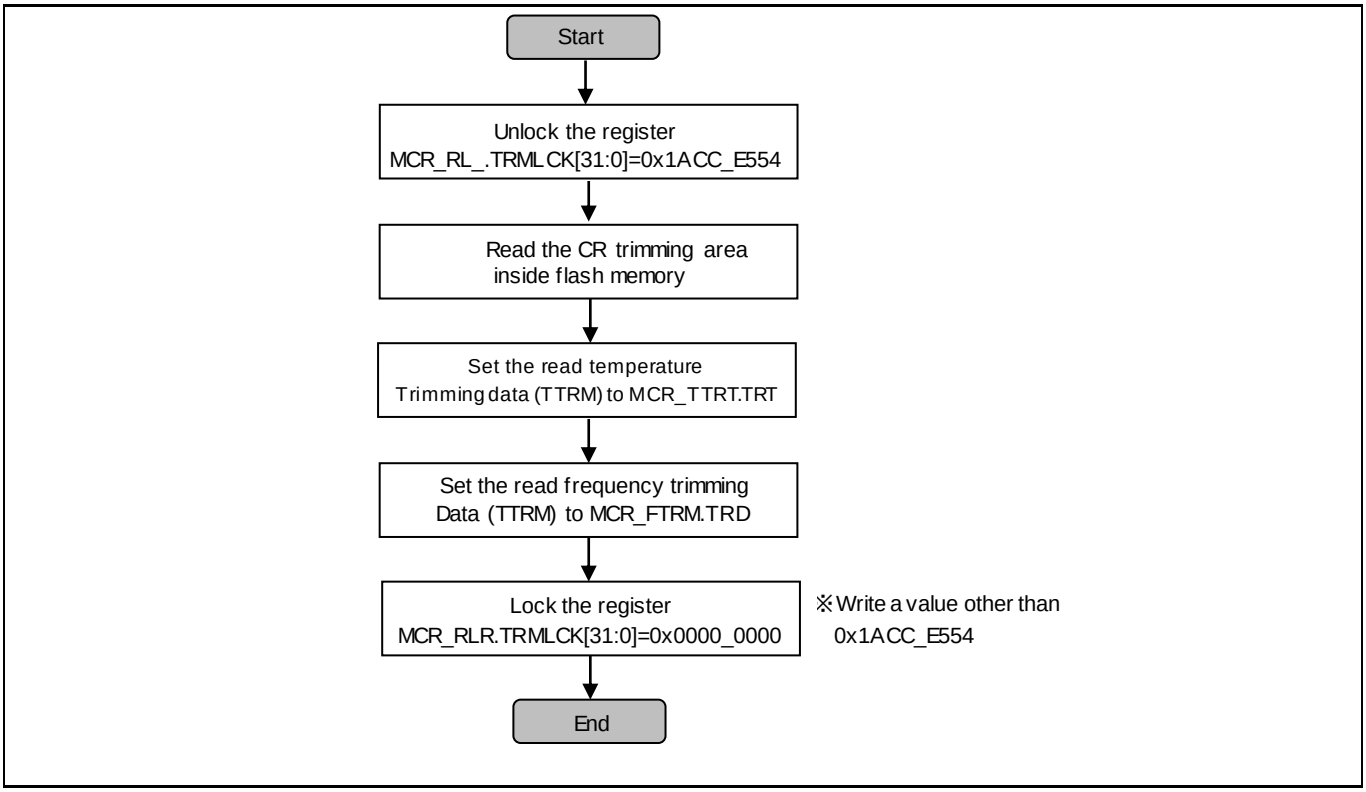
**Figure 4-5 Xtrm Calculation Procedure Example**



**Procedure example of using "CR trimming" area storage data inside flash memory**

Figure 4-6 shows a procedure example of reading trimming data stored in the "CR trimming" area inside the flash memory and setting it in the High-speed CR oscillation Frequency Trimming Register.

**Figure 4-6 Procedure Example of Using "CR Trimming" Area Storage Data**




***Note:***

- *For the address of the CR trimming area, see "FLASH PROGRAMMING MANUAL" for the product used.*

# 5. High-Speed CR Trimming Function Register List

The following lists and explains registers used for frequency trimming function of the high-speed CR oscillator.

Table 5-1 lists the registers.

**Table 5-1 Register list**

| Abbreviation | Register name | Reference |
|---|---|---|
| MCR_PSR | High-speed CR oscillation Frequency Division Setup Register | 5.1 |
| MCR_FTRM | High-speed CR oscillation Frequency Trimming Register | 5.2 |
| MCR_TTRM | High-speed CR oscillation Temperature Trimming Register | 5.3 |
| MCR_RLR | High-speed CR oscillation Register Write-Protect Register | 5.4 |

## 5.1 High-speed CR oscillation Frequency Division Setup Register (MCR_PSR)

The MCR_PSR register sets the frequency division ratio of high-speed CR oscillation.
A divided clock can be input in base timer.

**Register configuration**

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | CSR | | |
| Attribute | - | | | | | R/W | | |
| Initial value | - | | | | | 001 | | |

**Register functions**

**[bit7:3] Reserved : Reserved bits**
"0b00000" is read from these bits.
Set these bits to "0b00000" when writing.

**[bit2:0] CSR: High-speed CR oscillation frequency division ratio setting bits**

| bit2 | bit1 | bit0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | 1/4 |
| 0 | 0 | 1 | 1/8 [Initial value] |
| 0 | 1 | 0 | 1/16 |
| 0 | 1 | 1 | 1/32 |
| 1 | 0 | 0 | 1/64 |
| 1 | 0 | 1 | 1/128 |
| 1 | 1 | 0 | 1/256 |
| 1 | 1 | 1 | 1/512 |

## 5.2 High-speed CR oscillation Frequency Trimming Register (MCR_FTRM)

The MCR_FTRM register sets the frequency trimming value.

### Register configuration

| bit | 31 | | | | | | | | | | | | | | | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | - | | | | | | | | | | | | | | | |
| Initial value | - | | | | | | | | | | | | | | | |

| bit | 15 | | | | | | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | TRD[9:0] | | | | | | | | | |
| Attribute | - | | | | | | | R/W | | | | | | | | | |
| Initial value | - | | | | | | | 0111101111 | | | | | | | | | |

### Register functions

**[bit31:10] Reserved : Reserved bits**

"0" is always read from these bits.
These bits have no effect when written.

**[bit9:0] TRD[9:0] : Frequency trimming setup bits**

| bit9:5 | Description |
|---|---|
| When write | These bits make fine adjustment to the high-speed CR oscillator frequency.<br>For values to be set, see trimming data acquisition in the operation explanation of the frequency trimming function.<br>These bits fluctuate in frequency steps of approximately 1% each time ±1 setting is made. |
| When read | A specified value is read.<br>As an initial value, "0b01111" is read. |

| bit4:0 | Description |
|---|---|
| When write | These bits make fine adjustment to the high-speed CR oscillator frequency.<br>For values to be set, see trimming data acquisition in the operation explanation of the frequency trimming function.<br>These bits fluctuate in frequency steps of approximately 0.4% each time ±1 setting is made. |
| When read | A specified value is read.<br>As an initial value, "0b01111" is read. |

***Notes:***

- *This register is not initialized by software reset.*
- *For values to be set to the TRD bits, see trimming data acquisition in the operation explanation of the frequency trimming function.*

## 5.3 High-speed CR Oscillation Temperature Trimming Setup Register (MCR_TTRM)

The MCR_TTRM register sets the temperature trimming value.

### Register Configuration

| bit | 31 | | | | | | | | | | | | | | | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | - | | | | | | | | | | | | | | | |
| Initial value | - | | | | | | | | | | | | | | | |

| bit | 15 | | | | | | | | | | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | TRT[4:0] | | | | |
| Attribute | - | | | | | | | | | | | R/W | | | | |
| Initial value | - | | | | | | | | | | | 10000 | | | | |

### Register functions

**[bit31:5] Reserved : Reserved bits**

"0" is always read from these bits.

These bits have no effect when written.

**[bit4:0] TRT[4:0] : Temperature trimming setup bits**

| bit4:0 | Description |
|---|---|
| When write | These bits make adjustment to the high-speed CR oscillator frequency.<br>Write the value read from Temperature Trimming bit storage area in Flash Memory.<br>For Temperature Trimming bit storage area, see “FLASH PROGRAMING MANULA” of the product used. |
| When read | A specified value is read.<br>As an initial value, 0b10000 is read. |

***Notes:***

- *This register is not initialized by software reset.*
- *Before obtaining the frequency trimming data, be sure to set this register.*

## 5.4 High-Speed CR Oscillation Register Write-Protect Register (MCR_RLR)

The MCR_RLR register controls the write-protect state of the frequency trimming register (MCR_FTRM)/high-speed CR oscillation temperature trimming register (MCR_TTRM).

### Register configuration

| bit | 31 | 16 |
|---|---|---|
| Field | TRMLCK[31:16] | |
| Attribute | R/W | |
| Initial value | 0x0000 | |

| bit | 15 | 0 |
|---|---|---|
| Field | TRMLCK[15:0] | |
| Attribute | R/W | |
| Initial value | 0x0001 | |

### Register functions

**[bit31:0] TRMLCK[31:0] : Register write-protect bits**

| bit31:0 | Description |
|---|---|
| When read | When 0x00000000 is read, the MCR_FTRM/MCR_TTRM register is currently unlocked.<br>When 0x00000001 is read, the MCR_FTRM/MCR_TTRM register is currently locked. |
| Writing a value other than 0x1ACCE554 | Locks the MCR_FTRM/MCR_TTRM register |
| Writing 0x1ACCE554 | Unlocks the MCR_FTRM/MCR_TTRM register |

***Note:***

- *This register is not initialized by software reset.*

# 6. High-Speed CR Trimming Function Usage Precautions

This section explains the precautions for using the high-speed CR trimming function.

- Low-speed CR oscillator

  This trimming function is only enabled for the high-speed CR oscillator.
  It cannot apply to the low-speed CR oscillator.

- Data stored in the "CR trimming" area

  The CR trimming" area stores the factory preset frequency trimming data. For the address of the "CR trimming" area, see "FLASH PROGRAMMING MANUAL" for the product used.
  When Data in flash memory is erased, the data in “CR trimming” area is also erase at the same time.
  To use the data in the “CR trimming” area, save the data in the “CR trimming” area to other area such as RAM before erasing the data in flash memory.
  Otherwise, erase the sectors other than those in “CR trimming” area.

- For High-seed CR oscillator oscillation frequency accuracy

  Without setting High-speed CR oscillation temperature trimming register (MCR_TTRM) and High-speed CR oscillation temperature trimming register (MCR_FTRM), the accuracy of the High-speed CR oscillator described in “Data Sheet” cannot be guaranteed. So, be sure to set the above registers before use.

- How to use base timer

  For information about how to use base timer, see Chapters "Base Timer" in "Timer Part" and "I/O Port".

- FCS (Anomalous Frequency Detection)

  For FCS function (anomalous frequency detection), see Chapter "Clock supervisor". Do not perform CR trimming after the FCS function is enabled.

# CHAPTER2-4: Low-speed CR Prescaler

**This chapter shows the functions and operation of low-speed CR Prescaler.**



CODE:9BFLCPC-E01.0

# 1. Low-speed CR Prescaler Overview

This section shows the overview of low-speed CR prescaler.

**Low-speed CR Prescaler**

By setting the low-speed CR prescaler load register(LCR_PRSLD), the low-speed CR prescaler divides low-speed CR and generates low-speed CR clock(CLKLC).
This macro can correct the accuracy of low-speed CR. For the correcting method, see the example of correcting low-speed CR.

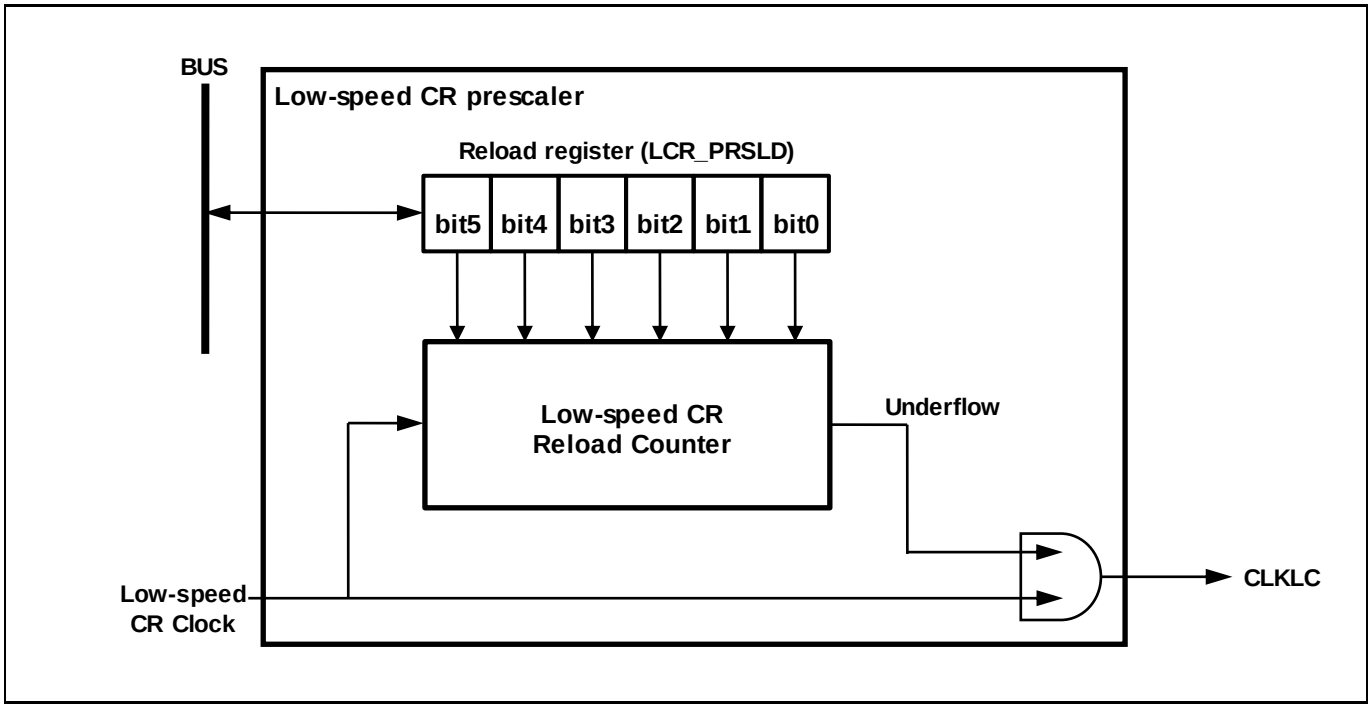
# 2. Low-speed CR Prescaler Configuration

This section shows the block diagram of low-speed CR prescaler.

**Block diagram of low-speed CR prescaler**

For the block diagram of low-speed CR prescaler, see Figure 2-1.

**Figure 2-1 Block Diagram of Low-speed CR Prescaler**




- Low-speed CR Prescaler Load Register (LCR_PRSLD)
  Sets the division ratio (reload value) of Low-speed CR Prescaler.
- Low-speed CR Reload Counter
  This is the down counter which generates the Low-speed CR Division Clock (CLKLC).

# 3. Low-speed CR Prescaler Operation and Setup Procedure Example

This section explains the operation of Low-speed CR Prescaler. This section also shows the example of setup procedures.

## 3.1 Setup procedures of Low-speed CR Prescaler

The Low-speed CR is asynchronous with the peripheral clock (PCLK).

For writing to the Low-speed CR Prescaler Reload Register, the peripheral clock is used. Therefore, if the setting change of the Low-speed CR Prescaler Load Register and the reload of the reload counter occur simultaneously, a value reloaded to the reload counter is not guaranteed.

So, execute the rewriting of the Low-speed CR Prescaler Reload Register conforming to the following procedures.

**For Switching the division clock**

The initial value of the Low-speed CR Prescaler Reload Register(LCR-PRSLD) is “0”.

Thus, for changing the value from the initial value, these procedures are unnecessary.

1. Set "0" to the Low-speed CR Prescaler Reload Register (LCR_PRSLD).
2. Wait until the value of the Low-speed CR Prescaler Reload Register (LCR_PRSLD) is reloaded to the reload counter.
3. The wait time is obtained by calculating the following formula:

   Low-speed CR cycle (50 kHz: 20 µs) × "the set value before changed to "0" in Item 1."
4. Write new setup value to the Low-speed CR Prescaler Reload Register (LCR_PRSLD).

For wait time at setup change, see Table 3-1.

**Table 3-1 Setup Wait Time**

| Reload Value before Setup | Setup Value | Wait Time |
|---|---|---|
| 0 | 0 | Not exists. |
| 1 | 0 | 20 µs (20 µs × 1) |
| 2 | 0 | 40 µs (20 µs × 2) |
| 3 | 0 | 60 µs (20 µs × 3) |
| : | : | : |
| 60 | 0 | 1200 µs (20 µs × 60) |
| 61 | 0 | 1220 µs (20 µs × 61) |
| 62 | 0 | 1240 µs (20 µs × 62) |
| 63 | 0 | 1260 µs (20 µs × 63) |

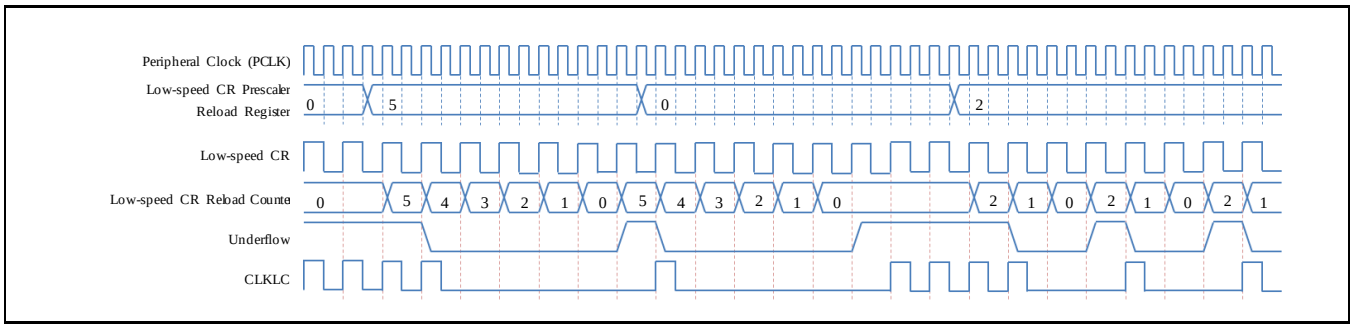
***Notes:***

- *The division clock cannot be stopped.*
- *The setting of the Low-speed CR Prescaler Reload Register (LCR_PRSLD) is executed at the underflow of the Low-speed CR Reload Counter.*

## 3.2 Operation of Low-speed CR Prescaler

For the operation of the Low-speed CR Prescaler, see Figure 3-1.

**Figure 3-1 Low-speed CR Prescaler Operation**

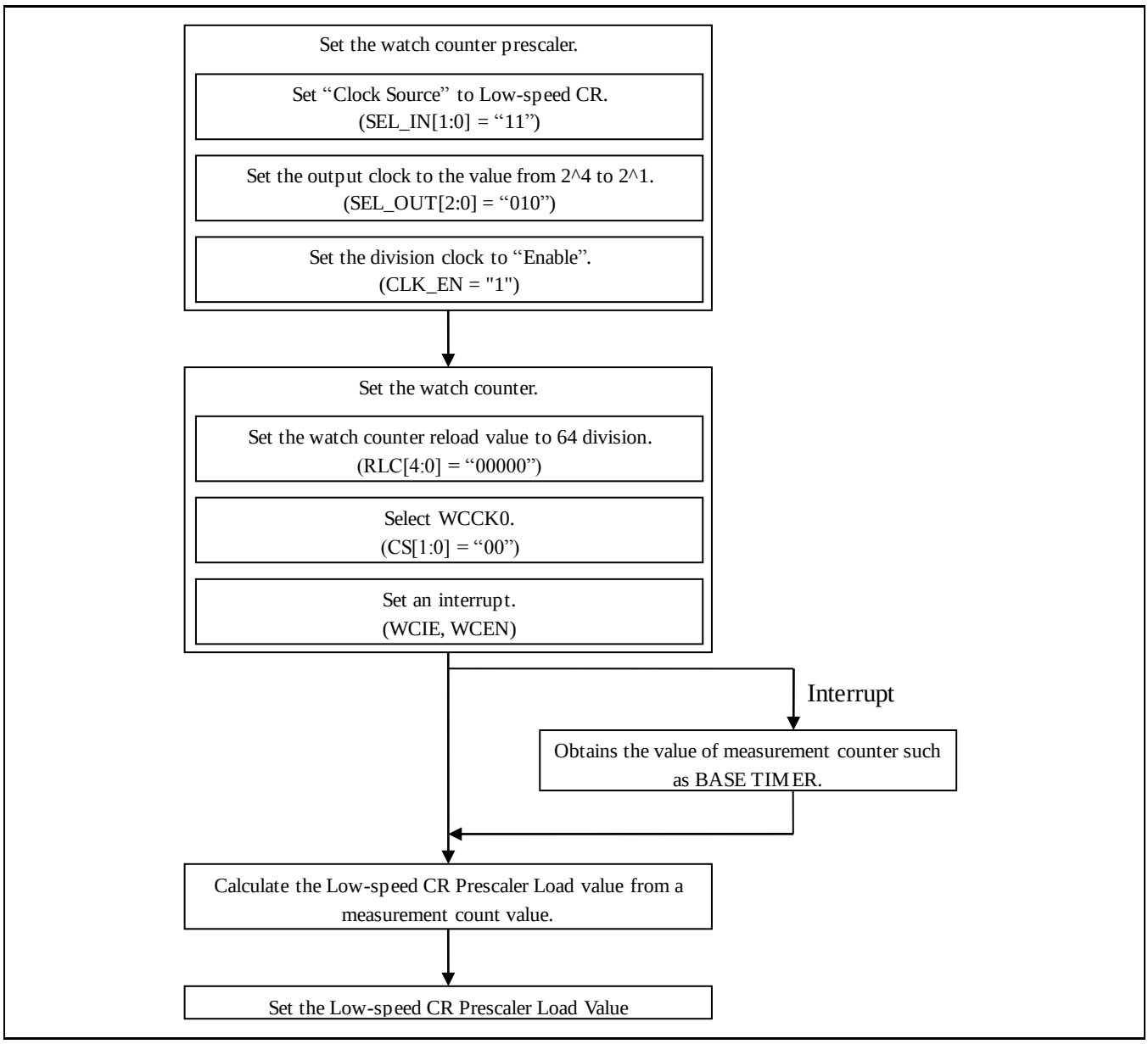



1. Sets the Low-speed CR Prescaler Load Register (LCR_PRSLD in synchronization with the peripheral clock (PCLK)
2. Retrieves the value of the Low-speed CR Prescaler Load Register (LCR_SLD) at the moment the Low-speed CR Reload Counter indicates "0".
3. Outputs the Low-speed CR (CLKLC) at the moment when the Low-speed CR Reload Counter underflow occurs.

## 3.3 Low-speed CR Correction Example

For the correction example of the Low-speed C, see Figure 3-2.

**Figure 3-2 Low-speed CR Correction Example**



# 4. Low-speed CR Prescaler Register

This section shows the list of the Low-speed CR Prescaler Register.

### Low-speed CR Prescaler Register

**Table 4-1 List of Low-speed CR Prescaler Register**

| Abbreviation | Register name | Reference |
|---|---|---|
| LCR_PRSLD | Low -speed CR Prescaler Control Register | 4.1 |

## 4.1 Low-speed CR Prescaler Control Register (LCR_PRSLD)

The Low-speed CR Prescaler Control Register is used to set the division ratio of low-speed CR.

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | LCR_PRSLD[5:0] | | | | | |
| Attribute | - | | R/W | | | | | |
| Initial Value | 00 | | 000000 | | | | | |

**[bit7:6] Reserved: Reserved bits**

Always “0” is read.

They have no effect in write mode.

**[bit5:0] LCR_PRSLD: Low-speed CR Prescaler Load**

At writing, sets the division ratio of the Low-speed CR Prescaler (the reload value of a reload counter).

At reading, the set value is read.

***Note:***

- *This register is not initialized with software reset.*

# CHAPTER3: Clock supervisor

**This chapter explains the clock supervisor functions.**



CODE: 9BFCSV-E02.4

# 1. Overview

This section provides an overview of the clock supervisor functions.

The clock supervisor includes the following two types of functions.

**Clock failure detection (CSV: Clock failure detection by clock Supervisor)**

The clock failure detection monitors the main and sub clocks. If a rising edge of the monitored clock is not detected within the specified period, this function determines that the oscillator has failed, and outputs a system reset request.

**Anomalous frequency detection (FCS: anomalous Frequency detection by Clock Supervisor)**

The anomalous frequency detection monitors frequency of the main clock. Within the specified period between an edge and the next edge of the divided clock of high-speed CR, this function counts up the internal counter value using the main clock. If the count value reaches out of the set window range, the function determines that the main clock frequency is anomalous, and outputs an interrupt request or a system reset request to the CPU.

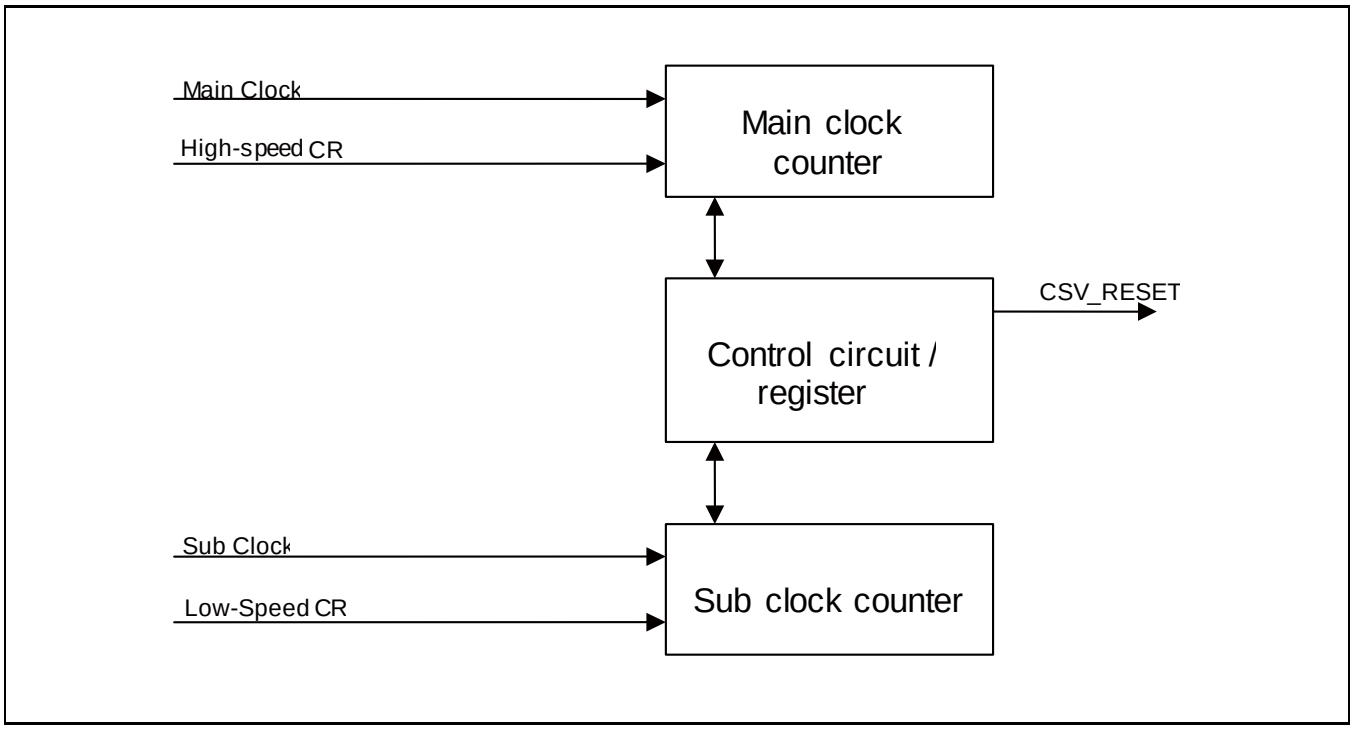
# 2. Configurations and Block Diagrams

This section explains the block diagrams of the clock supervisor functions.

## 2.1 Clock failure detection

Figure 2-1 shows the block diagram of the clock failure detection.

**Figure 2-1 Clock Failure Detection Block Diagram**




The clock failure detection consists of the following three types of blocks.

**Control circuit/register**

- This block includes a circuit controlling the clock failure detection,
- Also includes setup registers enabling/disabling the clock failure detection.

**Main clock counter**

A counter that monitors the main clock with the high-speed CR clock.

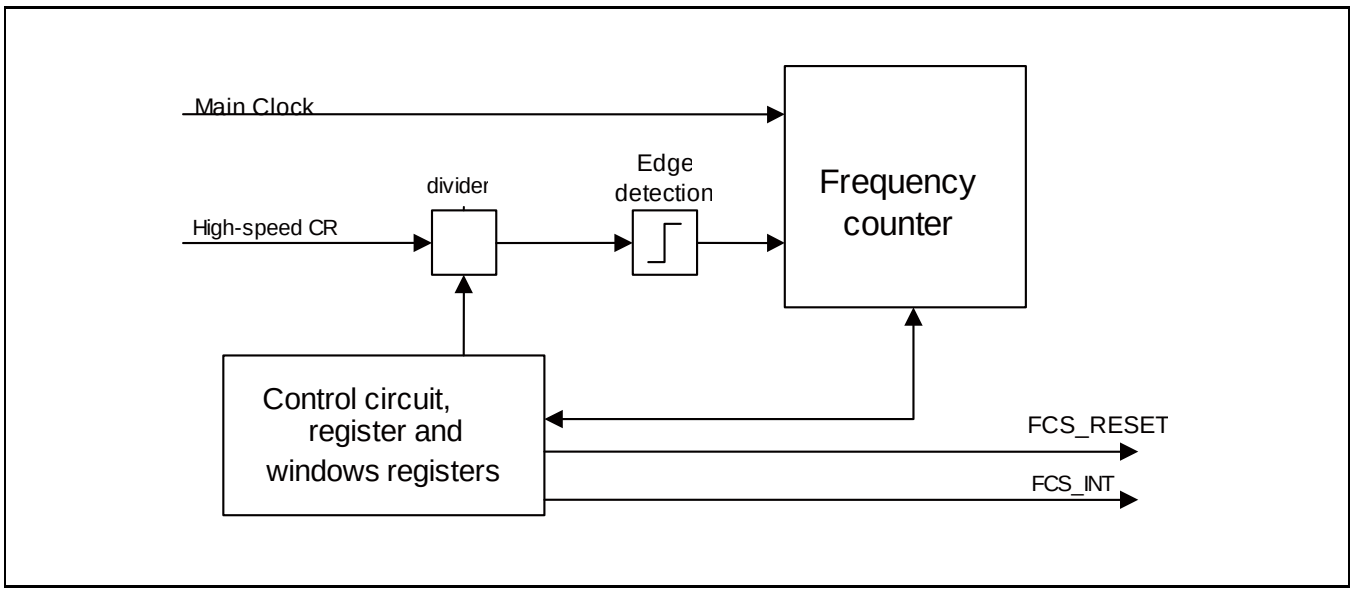
**Sub clock counter**

A counter that monitors the sub clock with the low-speed CR clock.

## 2.2 Anomalous frequency detection

Figure 2-2 shows the block diagram of the anomalous frequency detection.

**Figure 2-2 Anomalous Frequency Detection Block Diagram**




The anomalous frequency detection consists of the following three types of blocks.

**Control circuit/register and window registers**

- This block includes a circuit controlling the anomalous frequency detection.
- Also includes setup registers enabling/disabling the anomalous frequency detection.
- Also includes window registers defining the frequency range for measurements.

**Frequency counter**

A counter based on the main clock.

**Divider/edge detection**

- This block divides the high-speed CR.
- Also detects rising edges of the divided clock of high-speed CR.

# 3. Explanation of Operations

This section explains the operations of the clock supervisor functions.

### Clock failure detection function

The clock failure detection function monitors the main and sub clocks. If a rising edge of the monitored clock is not detected within the specified period, this function determines that the oscillator has failed, and outputs a system reset request.

- This reset request is referred to as the CSV reset request.
- CSV function monitors each of the main and sub clocks independently.
- It stops monitoring when the main and sub oscillators stop oscillating.
- It stops monitoring while waiting for oscillation stabilization wait time.
- When the oscillation stabilization wait time of main and sub oscillators ends, CSV function is automatically enabled.

***Notes:***

- *Each of the main and sub clocks can be enabled/disabled independently using the CSV control register (CSV_CTL).*
- *The main clock is monitored with the high-speed CR clock, and the sub clock is monitored with the low-speed CR clock. When a rising edge is not detected within 32 clocks of high-speed CR for the main clock, or within 32 clocks of low-speed CR for the sub clock, this function determines that the oscillator has failed.*

### Anomalous frequency detection function

The anomalous frequency detection function monitors the main clock.

Within the specified period between a rising edge and the next rising edge of the divided clock of high-speed CR, this function counts up the internal counter using the main clock. If the count value reaches out of the set window range, the function determines that the main clock frequency is anomalous, and outputs an interrupt request or a system reset request to the CPU.

- This interrupt request is referred to as the FCS interrupt request, and reset request as the FCS reset request.
- The FCS function only monitors frequency of the main clock.
- It stops monitoring when the main oscillator stops oscillating.
- It stops monitoring while waiting for oscillation stabilization wait time.
- The FCS function is started with software, a user program.

***Notes:***

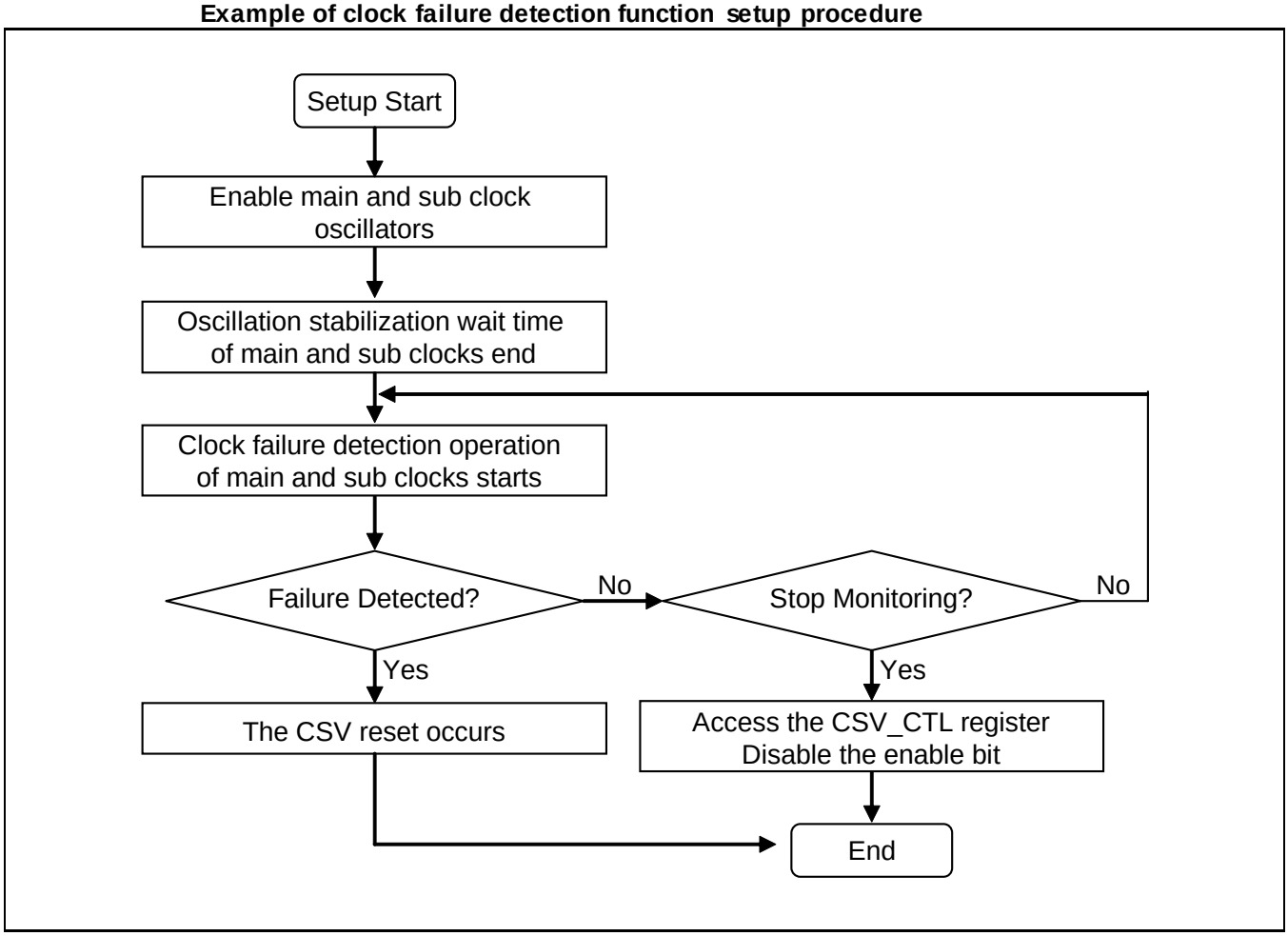
- *If the FCS reset is enabled:*

  An interrupt request occurs the first time a counter value deviates from the set window. If the interrupt request has not been cleared, and the counter value falls out of the specified window, a system reset request is output.
  If the FCS reset is not enabled, the reset request is masked.
- *The counter value, if it goes out of the specified window, is stored in the frequency detection counter register (FCSWD_CTL).*

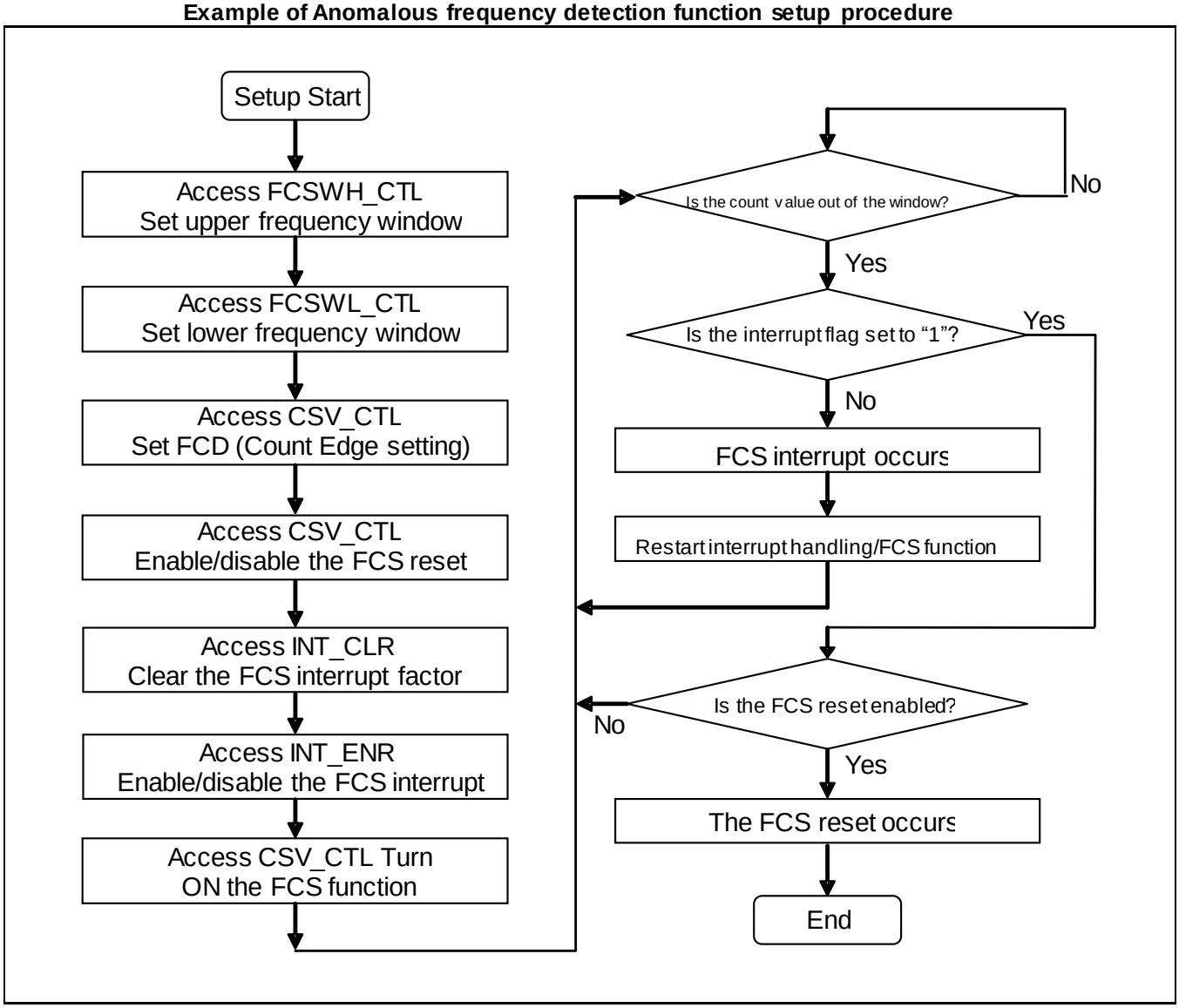
# 4. Setup Procedure Examples

This section explains examples of setting up the clock supervisor functions.

**Example of clock failure detection function setup procedure**



**Example of Anomalous frequency detection function setup procedure**



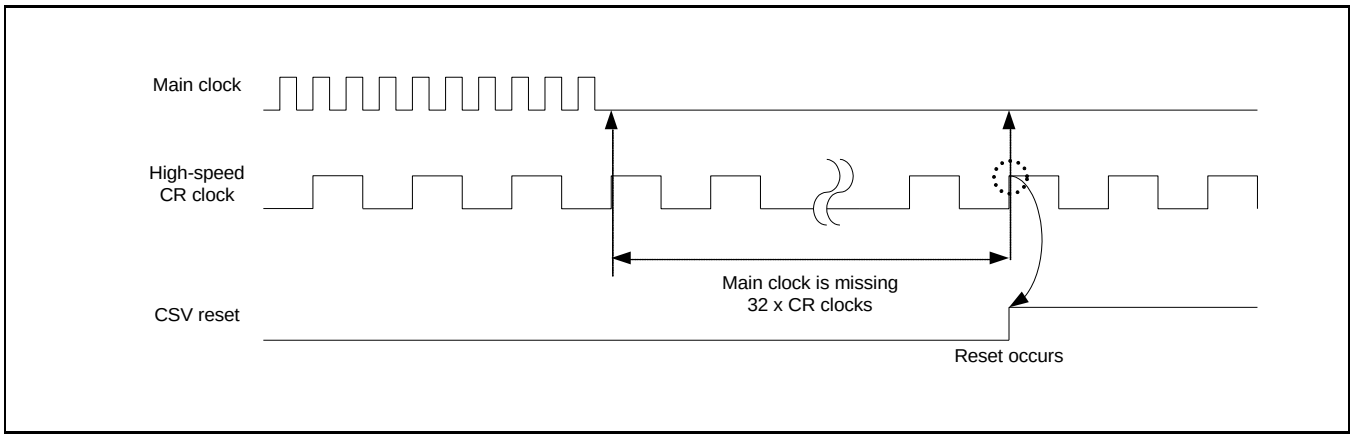
# 5. Operation Examples

This section explains examples of clock supervisor operations.

## 5.1 Clock failure detection

Figure 5-1 provides an example of clock failure detection operation.

**Figure 5-1 Example of clock failure detection operation**



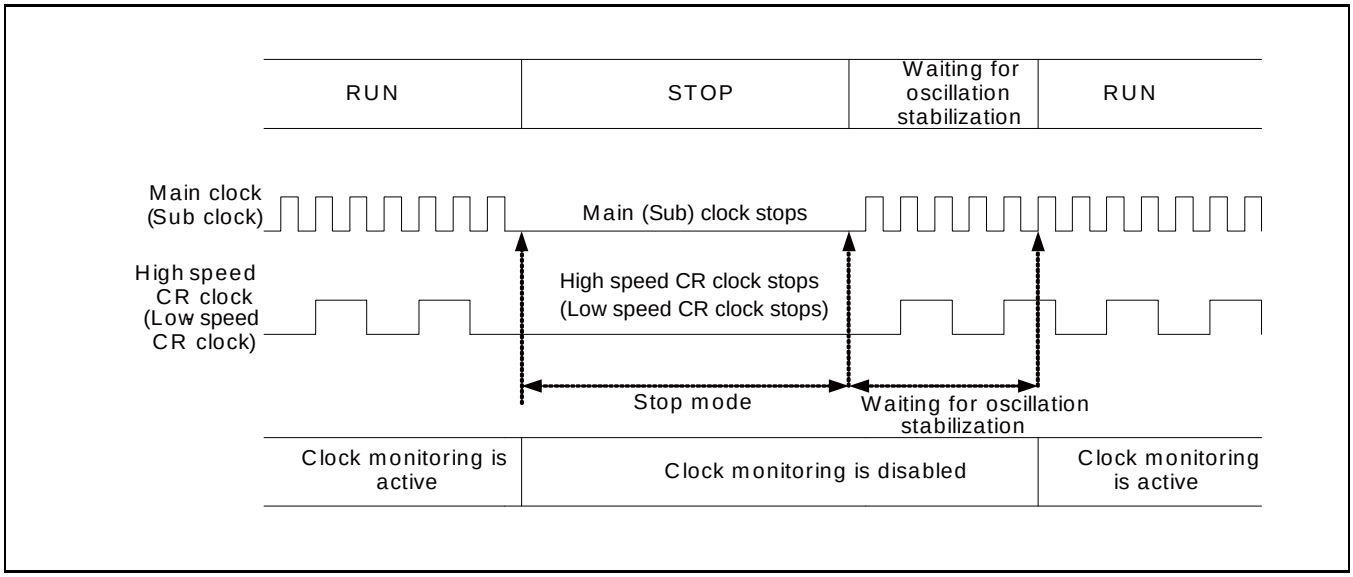

1. The main clock stops due to failure.
2. The function counts up clocks using the high-speed CR clock.
3. If the main clock keeps stopping during 32 clocks of high-speed CR, the function determines that the clock has failed and issues the CSV reset.

***Note:***

- *In case of the sub clock monitoring, the function determines that the sub clock has failed if it keeps stopping during 32 clocks of low-speed CR.*

Figure 5-2 provides an example of the clock failure detection operation in stop mode.

**Figure 5-2 Example of clock failure detection operation in stop mode**

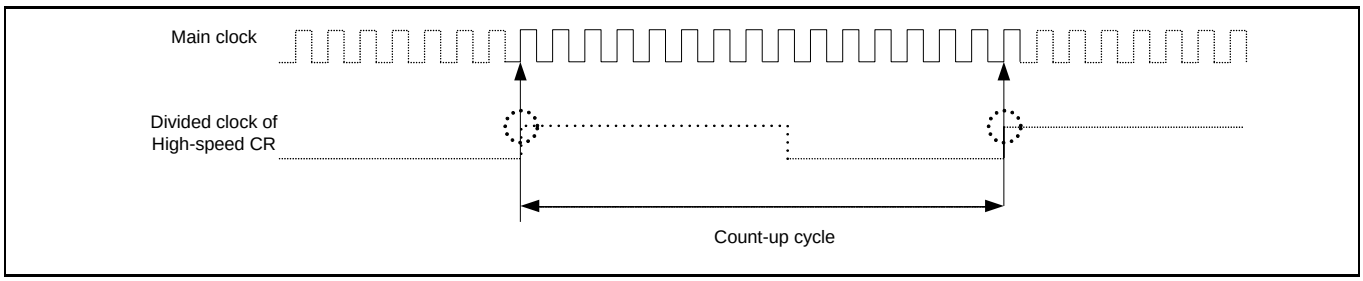



1. In stop mode, the main clock and high-speed CR clock stop.
   Meanwhile, the clock monitoring function also stops.
2. Upon the release of stop mode, oscillation of main clock and high-speed CR clock restart, waiting for oscillation stabilization. Meanwhile, the clock monitoring function keeps stopping.
3. When the oscillation stabilization wait time ends, the clock monitoring restarts.

## 5.2 Anomalous frequency detection

Figure 5-3 provides an example of anomalous frequency detection function operation.

**Figure 5-3 Example of anomalous frequency detection function operation**




1. This function detects rising edges of the divided clock of high-speed CR.
2. After detecting edges, it counts up clocks using the main clock.
3. It keeps counting up until it detects the next rising edge of the divided clock of high-speed CR.
4. Let "$\alpha$" be the count value with the main clock.
   Also let B denote the lower window value, and A the upper window value. Compare the count value $\alpha$ with those window values and if expression

   $B \leq \alpha \leq A$

   holds true, then the frequency is considered to be normal.
   If the count value $\alpha$ is out of the range, i.e., either

   $\alpha < B$, or $A < \alpha$

   is true, then the frequency is considered to be anomalous, and an interrupt occurs.
   If the interrupt flag has not been cleared after the interrupt and an anomalous frequency is detected again, then the function issues a reset depending on the setting.

## 5.3 Example of anomalous frequency detection function window setting

The anomalous frequency detection counts up between edges of the divided clock of high-speed CR. The measurement interval is also affected by the accuracy of CR. When you configure the window register value, therefore, the CR accuracy must be considered for the value.
For frequency accuracy of the CR oscillator, check the relevant "Data Sheet".

### Calculation method

The count value range of anomalous frequency detection must be added the CR accuracy, then, the window register value is set. The count range expression must be used as follows.

$$\text{Count value} = \left( \frac{1}{\text{Frequency of divided clock of CR} \times \left(1 \pm \frac{\text{CR accuracy}}{100}\right)} \right) \times \text{Frequency of main clock}$$

The count value by main clock of frequency L [Hz] can be calculated using the divide-by-Y CR oscillator clock of ±Z% accuracy with frequency K [Hz].

Count value A (positive CR frequency accuracy) = 1/[ (K/Y) × (1 + Z/100)] × L
Count value B (negative CR frequency accuracy) = 1/[ (K/Y) × (1 - Z/100)] × L

Those expressions lead the count value within the range A to B added internal CR accuracy.
Set the value smaller than count value A for the lower limit of the window, and larger than count value B for the upper limit.

The window setting is determined by the value allowed for frequency fluctuation of main oscillation defined by the user.

### Example calculation

The count value by main clock of frequency 4 MHz is calculated using the divide-by-1024 CR oscillator clock of ±5% accuracy with frequency 4 MHz.

Count value A (positive CR frequency accuracy)

$$\text{Count value A} = \left( \frac{1}{\frac{4\times10^6}{1024} \times \left(1 + \frac{5}{100}\right)} \right) \times 4\times10^6 \approx 975$$

Count value B (negative CR frequency accuracy)

$$\text{Count value B} = \left( \frac{1}{\frac{4\times10^6}{1024} \times \left(1 - \frac{5}{100}\right)} \right) \times 4\times10^6 \approx 1078$$

Those expressions yield the count value within the range 975 to 1078 including the high-speed CR error. If the window setting value is 5%, window setting value is as follows.

Window lower limit = 975 × 0.95(-5%) = 926.25 ≈ 3.43 MHz
Window upper limit = 1078 × 1.05(+5%) = 1131.9 ≈ 4.64 MHz

Thus, you can recognize that a main clock frequency out of the 3.4 MHz to 4.6 MHz range is anomalous.
Table 5-1 provides an example of the window settings.

**Table 5-1 Example of window settings**

| Divided clock of High-speed CR | Main clock | High-speed CR error | Count value including high-speed CR error | Lower limit of window set value | Upper limit of window set value |
|---|---|---|---|---|---|
| Divide-by-1024 clocks of CR:4 MHz | 4 MHz | ±5% | 975 (≈3.61 MHz) - 1078 (≈4.42 MHz) | 926 (≈3.43 MHz) | 1131 (≈4.64 MHz) |

# 6. Registers

This section explains the register list of the clock supervisor functions.

### Register list

Table 6-1 shows the register list.

**Table 6-1 Register list**

| Abbreviation | Register name | Reference |
|---|---|---|
| CSV_CTL | CSV control register | 6.1 |
| CSV_STR | CSV status register | 6.2 |
| FCSWH_CTL | Frequency detection window setting register (Upper) | 6.3 |
| FCSWL_CTL | Frequency detection window setting register (Lower) | 6.4 |
| FCSWD_CTL | Frequency detection counter register | 6.5 |

## 6.1 CSV control register (CSV_CTL)

The CSV_CTL register configures the control of CSV function.

### 6.1.1 Register configuration

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | FCD | | | Reserved | | FCSRE | FCSDE |
| Attribute | - | R/W | | | - | | R/W | R/W |
| Initial value | 0 | 111 | | | 00 | | 0 | 0 |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | SCSVE | MCSVE |
| Attribute | - | | | | | | R/W | R/W |
| Initial value | 000000 | | | | | | 1 | 1 |

### 6.1.2 Register functions

**[bit15] Reserved : Reserved bit**

"0" is read from this bit.

Set this bit to "0" when writing.

**[bit14:12] FCD: FCS count cycle setting bits**

| bit14:12 | Description |
|---|---|
| When 000 is written | Setting is prohibited |
| When 001 is written | |
| When 010 is written | |
| When 011 is written | |
| When 100 is written | |
| When 101 is written | 1/256 frequency of high-speed CR oscillation |
| When 110 is written | 1/512 frequency of high-speed CR oscillation |
| When 111 is written | 1/1024 frequency of high-speed CR oscillation [Initial value] |
| When read | The register value is read. |

**[bit11:10] Reserved : Reserved bits**

"0b00" is read from these bits.

Set these bits to "0b00" when writing.

**[bit9] FCSRE: FCS reset output enable bit**

| bit | Description |
|---|---|
| When 0 is written | The FCS reset is disabled [Initial value] |
| When 1 is written | The FCS reset is enabled |
| When read | The register value is read. |

### [bit8] FCSDE: FCS function enable bit

| bit | Description |
|---|---|
| When 0 is written | The FCS function is disabled [Initial value] |
| When 1 is written | The FCS function is enabled. |
| When read | The register value is read. |

### [bit7:2] Reserved : Reserved bits

"0b000000" is read from these bits.
Set these bits to "0b000000" when writing.

### [bit1] SCSVE: Sub CSV function enable bit

| bit | Description |
|---|---|
| When 0 is written | The sub CSV function is disabled |
| When 1 is written | The sub CSV function is enabled. [Initial value] |
| When read | The register value is read. |

### [bit0] MCSVE: Main CSV function enable bit

| bit | Description |
|---|---|
| When 0 is written | The main CSV function is disabled |
| When 1 is written | The main CSV function is enabled. [Initial value] |
| When read | The register value is read. |

***Note:***

- *This register is not initialized by software reset.*

## 6.2 CSV status register (CSV_STR)

The CSV_STR register indicates the status of CSV function.

### 6.2.1 Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | SCMF | MCMF |
| Attribute | - | | | | | | R | R |
| Initial value | 000000 | | | | | | 0 | 0 |

### 6.2.2 Register functions

**[bit7:2] Reserved : Reserved bits**

"0b000000" is read from these bits.
Set these bits to "0b000000" when writing.

**[bit1] SCMF : Sub clock failure detection flag**

| Bit | Description |
|---|---|
| When written | No effect |
| When 0 is read | No sub clock failure has been detected. [Initial value] |
| When 1 is read | A sub clock failure has been detected. |

**[bit0] MCMF : Main clock failure detection flag**

| Bit | Description |
|---|---|
| When written | No effect |
| When 0 is read | No main clock failure has been detected. [Initial value] |
| When 1 is read | A main clock failure has been detected. |

***Note:***

- *This register is cleared when being read.*

## 6.3 Frequency detection window setting register (Upper) (FCSWH_CTL)

The FCSWH_CTL register configures the frequency detection window setting register (Upper).

### 6.3.1 Register configuration

| bit | 15 | 0 |
|---|---|---|
| Field | FWH | |
| Attribute | R/W | |
| Initial value | 0xFFFF | |

### 6.3.2 Register functions

**[bit15:0] FWH: Frequency detection window setting bits (Upper)**

| bit15:0 | Description |
|---|---|
| When written | Any value can be written to these bits. |
| When read | The register value is read. |

***Notes:***

- *Set a value larger than the value set in FCSWL_CTL (Frequency detection window setting register (Lower)).*
- *This register is not initialized by software reset.*

## 6.4 Frequency detection window setting register (Lower) (FCSWL_CTL)

The FCSWL_CTL register configures the frequency detection window setting register (Lower).

### 6.4.1 Register configuration

| bit | 15 | 0 |
|---|---|---|
| Field | FWL | |
| Attribute | R/W | |
| Initial value | 0x0000 | |

### 6.4.2 Register functions

**[bit15:0] FWL: Frequency detection window setting bits (Lower)**

| bit15:0 | Description |
|---|---|
| When written | Any value can be written to these bits. |
| When read | The register value is read. |

***Notes:***

- *Set a value smaller than the value set in FCSWH_CTL (Frequency detection window setting register (Upper)).*
- *This register is not initialized by software reset.*

## 6.5 Frequency detection counter register (FCSWD_CTL)

The FCSWD_CTL register indicates the counter value of frequency detection using the main clock.

### 6.5.1 Register configuration

| bit | 15 | 0 |
|---|---|---|
| Field | FWD | |
| Attribute | R | |
| Initial value | 0x0000 | |

### 6.5.2 Register functions

**[bit15:0] FWD: Frequency detection count data**

| bit15:0 | Description |
|---|---|
| When written | No effect on operation |
| When read | The count value is read. |

***Notes:***

- *This register retains the count value when detecting an error.*
- *This register is not initialized by software reset.*

# 7. Usage Precautions

This section explains the precautions for using the clock supervisor functions.

- For details on enabling and clearing the frequency detection interrupt sources, see Chapter "Clock".

- For details on clock failure detection and anomalous frequency detection reset sources, see Chapter "Resets".

- Operation after the occurrence of a reset

  After the occurrence of a reset triggered by clock failure detection, clock mode returns to high-speed CR.

  Do not select the faulty clock again.

- The high-speed CR clock for use of the frequency detection

  The frequency failure detection is affected by the frequency accuracy of high-speed CR itself. When you configure frequency window, therefore, the accuracy of high-speed CR must be considered for the window value. Do not trim the high-speed CR clock after the anomalous frequency detection has been enabled.

- The order of the anomalous frequency detection settings before using

  Before enabling FCS (FCSDE=1), specify the count cycle (FCD), reset enable (FCSRE), and frequency window (FWH/FWL) settings.

  If you want to change any of FCD/FCSRE/FWH/FWL after FCS has been enabled, stop the FCS function before changing the setting. Do not change the setting while FCS is enabled.

- The enable settings of the anomalous frequency detection before using

  Depending on the setting of the FCSRE bit in the CSV control register (CSV_CTL), operation during anomalous frequency detection varies. Table 7-1 shows the setting list.

**Table 7-1 List of the FCS function and FCSRE bit settings**

| | FCSRE=0 | FCSRE=1 |
|---|---|---|
| FCSDE=0 | Stops FCS function | Stops FCS function |
| FCSDE=1 | Enables FCS function<br>Generates an interrupt upon error detection | Enables FCS function<br>An interrupt occurs upon the first error detection<br>A reset occurs upon the second error detection |

- Interrupt settings for the frequency detection and main timer mode

  The internal bus clock stops while the clock is in main timer mode. In this mode, an interrupt does not occur even if an error is detected while FCSRE is set to "0".

  In main timer mode, therefore, do not set FCSRE bit to "0". If FCSRE bit is set to "1", a reset occurs.

- The settings for CSV OFF and external reset.

  When CSV function is set to OFF, the CSV reset is not generated even if the clock failure occurs, and moreover, the external reset (INITX) is not accepted. So, it is recommended not to turn OFF the CSV function, if you do not have special reason.

# CHAPTER4: Resets

**This chapter explains the function and operation of the resets.**



CODE: 9AFRESET-E01.0

# 1. Overview

This family has the following reset factors and issues a reset to initialize a device upon accepting a reset factor.

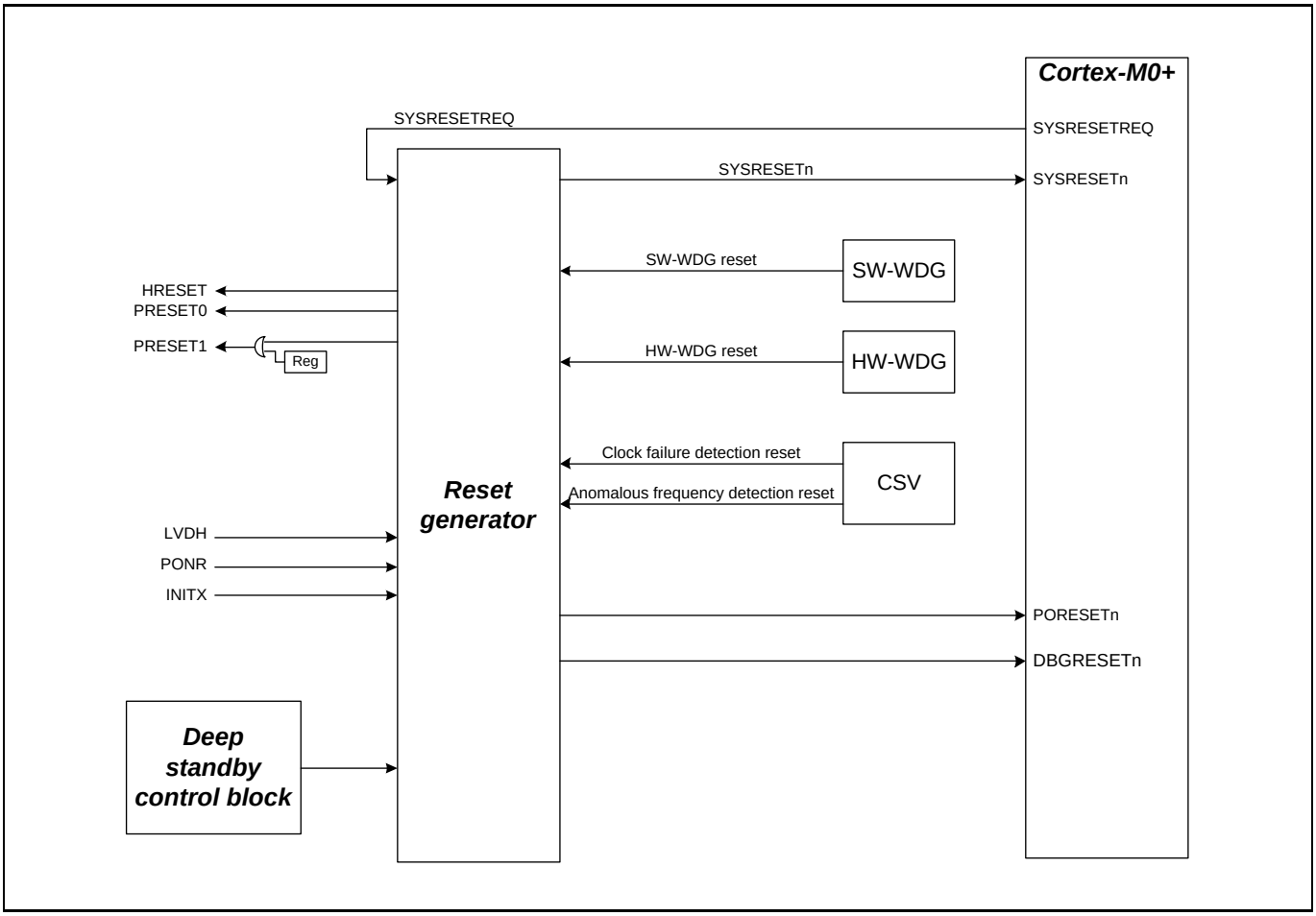
- Power-on reset
- INITX pin input
- External power supply/low-voltage detection reset
- Software watchdog reset
- Hardware watchdog reset
- Clock failure detection reset
- Anomalous frequency detection reset
- Software reset
- Deep standby transition reset

# 2. Configuration

This section explains the configuration of the reset circuit.

### Block Diagram of Resets

**Figure 2-1 Block Diagram of Resets**




| | |
|---|---|
| PONR : | Power-on reset |
| INITX : | INITX pin input reset |
| LVDH : | Low-voltage detection reset |
| HRESET : | AHB bus reset (a bus reset issued by all reset factors) |
| PRESET0, 1 : | APB0, APB1 bus resets (bus resets issued by all reset factors) |
| SW-WDG reset : | Software watchdog reset |
| HW-WDG reset : | Hardware watchdog reset |
| PORESETn : | Power-on reset that is input to Cortex-M0+ |
| SYSRESETn : | System reset that is input to Cortex-M0+ |
| SYSRESETREQ : | "SYSRESETREQ bit" signal of Cortex-M0+ internal reset control register |
| DBGRESETn : | SW-DP reset |
| DSTR : | Deep standby transition reset |

# 3. Operations

This section explains the operations of the resets of this family.

## 3.1 Reset Factors

This section explains reset factors.

### Power-On Reset (PONR)

A reset that is generated at power-up

| | |
|---|---|
| Generated by | This signal is generated by detecting a rising edge of the power supply. |
| Cleared by | This signal is automatically cleared after issuing a reset. |
| Initialization target | Initializes all register settings and hardware. |
| Flag | bit0 (PONR) of reset factor register (RST_STR) = 1 |

### INITX Pin Input Reset (INITX)

A reset that is externally input from a device

| | |
|---|---|
| Generated by | This signal is generated by inputting a low level to INITX pin. |
| Cleared by | This signal is cleared by inputting a high level to INITX pin. |
| Initialization target | Initializes all register settings and hardware except the debug circuit and deep standby control block.<br>Note: The following registers are not initialized.<br>- Reset factor register (RST_STR)<br>- bit15 to bit8 of low-voltage detection voltage setting register<br>- Deep standby return factor register 1 and 2 (WRFSR, WIFSR)<br>- Deep Standby RAM Retention Register (DSRAMR)<br>- Backup registers from 01 to 16 (BUR01 to BUR16) |
| Flag | bit1 (INITX) of reset factor register (RST_STR) = 1 |

* The content of the on-chip SRAM is retained if a reset is asynchronously input from the INITX pin.

### Low-voltage Detection Reset, External Voltage Monitoring (LVDH)

A reset that is input from a low-voltage detection circuit when a decrease in the external voltage is detected

| | |
|---|---|
| Generated by | This signal is generated when an external voltage is lowered than a specified level. |
| Cleared by | This signal is cleared when an external voltage is more than a specified level. |
| Initialization target | Initializes all register settings and hardware. |
| Flag | bit0 (PONR) of reset factor register (RST_STR) = 1 |

### Software Watchdog Reset (SWDGR)

A reset that is input from the software watchdog timer.

| | |
|---|---|
| Generated by | This signal is generated when the software watchdog timer underflows. |
| Cleared by | This signal is automatically cleared after issuing a reset. |
| Initialization target | Initializes all register settings and hardware except the debug circuit and hardware watchdog timer (including control registers) and deep standby control block.<br>Note: The following registers are not initialized.<br>- Reset factor register (RST_STR)<br>- bit15 to bit8 of low-voltage detection voltage setting register (LVD_CTL)<br>- Deep standby return factor register 1 and 2 (WRFSR, WIFSR)<br>- Deep Standby RAM Retention Register (DSRAMR)<br>- Backup registers from 01 to 16 (BUR01 to BUR16) |
| Flag | bit4 (SWDT) of reset factor register (RST_STR)= 1 |

### Hardware Watchdog Reset (HWDGR)

A reset that is input from the hardware watchdog timer.

| | |
|---|---|
| Generated by | This signal is generated when the hardware watchdog timer underflows. |
| Cleared by | This signal is automatically cleared after issuing a reset. |
| Initialization target | Initializes all register settings and hardware except the debug circuit and deep standby control block.<br>Note: The following registers are not initialized.<br>- Reset factor register (RST_STR)<br>- bit15 to bit8 of low-voltage detection voltage setting register (LVD_CTL)<br>- Deep standby return factor register 1 and 2 (WRFSR, WIFSR)<br>- Deep Standby RAM Retention Register (DSRAMR)<br>- Backup registers from 01 to 16 (BUR01 to BUR16) |
| Flag | bit5 (HWDT) of reset factor register (RST_STR) = 1 |

### Clock Failure Detection Reset (CSVR)

A reset that is input when the main or sub crystal oscillator being monitored fails.

| | |
|---|---|
| Generated by | This signal is generated when a clock failure is detected in the main or sub crystal oscillator. |
| Cleared by | This signal is automatically cleared after issuing a reset. |
| Initialization target | Initializes all register settings and hardware except the debug circuit and clock failure detection circuit (some registers) and deep standby control block.<br>Note: The following registers are not initialized.<br>- Reset factor register (RST_STR)<br>- bit15 to bit8 of low-voltage detection voltage setting register (LVD_CTL)<br>- Deep standby return factor register 1 and 2 (WRFSR, WIFSR)<br>- Deep Standby RAM Retention Register (DSRAMR)<br>- Backup registers from 01 to 16 (BUR01 to BUR16) |
| Flag | bit6 (CSVR) of reset factor register (RST_STR) = 1<br>bit1 (SCMF) or bit0 (MCMF) of CSV status register (CSV_STR) = 1<br>Note: For details on the CSV_STR, see Chapter "Clock supervisor". |

### Anomalous Frequency Detection Reset (FCSR)

A reset that is input when an anomalous frequency is detected in the main crystal oscillator.

| | |
|---|---|
| Generated by | This signal is generated when the frequency of the main crystal oscillator is outside of any given setting. |
| Cleared by | This signal is automatically cleared after issuing a reset. |
| Initialization target | Initializes all register settings and hardware except the debug circuit and the anomalous frequency detection (some registers) and deep standby control block.<br>Note: The following registers are not initialized.<br>- Reset factor register (RST_STR)<br>- bit15 to bit8 of low-voltage detection voltage setting register (LVD_CTL)<br>- Deep standby return factor register 1 and 2 (WRFSR, WIFSR)<br>- Deep Standby RAM Retention Register (DSRAMR)<br>- Backup registers from 01 to 16 (BUR01 to BUR16) |
| Flag | bit7 (FCSR) of reset factor register (RST_STR) = 1 |

### Software Reset (SRST)

A reset that is generated when an access to the reset control register occurs.

| | |
|---|---|
| Generated by | This signal is generated by a write to the Cortex-M0+ internal reset control register (SYSRESETREQ bit). |
| Cleared by | This signal is automatically cleared after issuing a reset. |
| Initialization target | Initializes all register settings and hardware except the following:<br>Functions and registers that are not initialized by a software reset<br>- Debug circuit<br>- Deep standby control block<br>- RTC<br>- All registers related to clock control<br>- Part of registers that control software and hardware watchdog timers<br>- Part of registers in the clock failure detection circuit<br>- Part of registers that detect an anomalous frequency<br>- Part of registers for CR trimming<br>- Reset factor register (RST_STR)<br>- bit15 to bit8 of low-voltage detection voltage setting register (LVD_CTL)<br>- RTC mode control register (PMD_CTL)<br>- Deep standby return factor register 1 and 2 (WRFSR, WIFSR)<br>- Deep Standby RAM Retention Register (DSRAMR)<br>- Backup registers from 01 to 16 (BUR01 to BUR16) |
| Flag | bit8 (SRST) of reset factor register (RST_STR) = 1 |

**Deep standby transition reset (DSTR)**

This reset occurs when transiting to deep standby mode.

| | |
|---|---|
| Generated by | This signal is generated by transiting to deep standby mode |
| Cleared by | This signal is cleared by returning from deep standby mode |
| Initialization target | Initializes all register settings and hardware except the following:<br>Functions and registers that are not initialized by a deep standby transition reset.<br>– Deep standby control block<br>– RTC<br>– HDMI-CEC/ Remote Control Reception<br>– Some registers of GPIO<br>– Low-voltage detection circuit register<br>– RTC mode control register (PMD_CTL)<br>– Deep standby return factor register 1 and 2 (WRFSR, WIFSR)<br>– Deep standby return permit register (WIER)<br>– WKUP pin input level register (WILVR)<br>– Deep Standby RAM Retention Register (DSRAMR)<br>– Backup registers from 01 to 16 (BUR01 to BUR16) |
| Flag | The bit of either deep standby return factor register 1 or 2 (WRFSR, WIFSR) is "1".<br>Note: The bit that becomes "1" differs by return factors. |

***Notes:***

- *For Cortex-M0+ internal reset control register (SYSRESETREQ) that controls the software reset, see "Chapter B3, System Address Map", in "ARMv6-M Architecture Reference Manual".*
- *The reset factor register that can determine the occurrence of each reset factor is initialized only by power-on reset.*

## 3.2 Resetting inside Device

This section explains the internal reset signals of this device.

Resets that are internally connected to the device are divided into resets that are input to the Cortex-M0+ core and resets that are input to peripheral circuits.

### 3.2.1 Resets to Cortex-M0+

### 3.2.2 Resets to Peripheral Circuit

### 3.2.1 Resets to Cortex-M0+

The device has three reset inputs to the Cortex-M0+ are PORESETn, SYSRESETn, and DBGRESETn. The following provides reset factors for these three reset inputs.

**Power-on reset PORESETn**

| | |
|---|---|
| Reset factors | - Power-on reset (PONR)<br>- Low-voltage detection reset (LVDH)<br>- Deep standby transition reset (DSTR) |

**System reset SYSRESETn**

| | |
|---|---|
| Reset factors | - Power-on reset (PONR)<br>- Low-voltage detection reset (LVDH)<br>- INITX pin input (INITX)<br>- Software watchdog reset (SWDGR)<br>- Hardware watchdog reset (HWDGR)<br>- Clock Failure Detection reset (CSVR)<br>- Anomalous frequency detection reset (FCSR)<br>- Software reset (SRST)<br>- Deep standby transition reset (DSTR) |

**SW-DP Reset DBGRESETn**

| | |
|---|---|
| Reset factors | - Power-on reset (PONR)<br>- Low-voltage detection reset (LVDH)<br>- Deep standby transition reset (DSTR) |

### 3.2.2 Resets to Peripheral Circuit

The bus resets (HRESET, PRESET0 and PRESET1) that are input to the peripheral circuit are basically generated by all reset factors. Resetting of PRESET1 can be controlled by register settings.

The following provides reset factors for the bus resets.

**Resets to Peripheral Circuit**

■ HRESET and PRESET0

| | |
|---|---|
| Reset factors | – Power-on reset (PONR)<br>– Low-voltage detection reset (LVDH)<br>– INITX pin input (INITX)<br>– Software watchdog reset (SWDGR)<br>– Hardware watchdog reset (HWDGR)<br>– Clock Failure Detection reset (CSVR)<br>– Anomalous frequency detection reset (FCSR)<br>– Software reset (SRST)<br>– Deep standby transition reset (DSTR) |

■ PRESET1

| | |
|---|---|
| Reset factors | – Power-on reset (PONR)<br>– Low-voltage detection reset (LVDH)<br>– INITX pin input (INITX)<br>– Software watchdog reset (SWDGR)<br>– Hardware watchdog reset (HWDGR)<br>– Clock Failure Detection reset (CSVR)<br>– Anomalous frequency detection reset (FCSR)<br>– Software reset (SRST)<br>– APB bus resets (APBC1_PSR)<br>– Deep standby transition reset (DSTR) |

***Notes:***

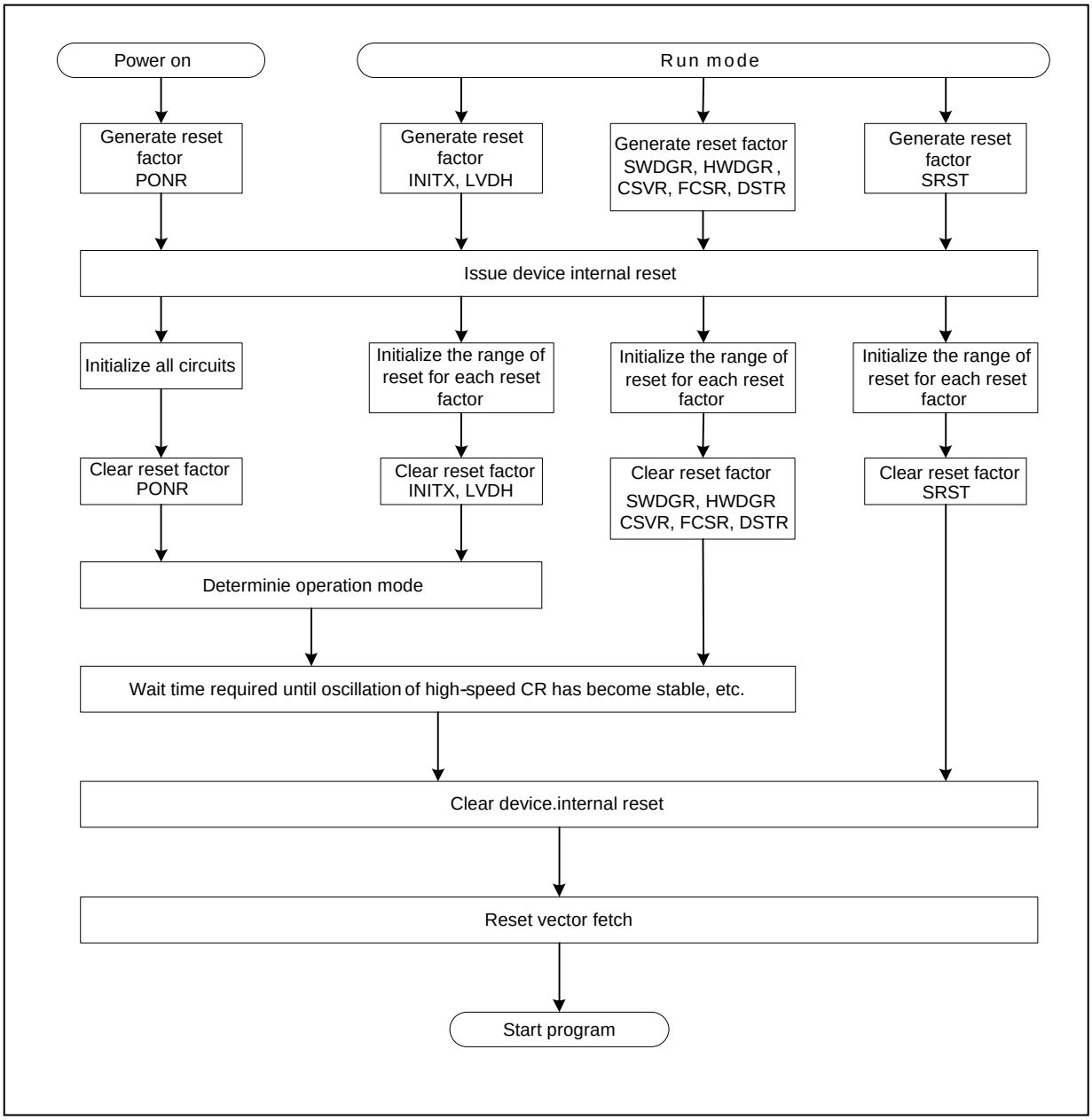
- *The peripheral circuit is essentially initialized with all reset factors. Depending on the specifications of the peripheral circuit, there are registers that are initialized only with specific causes. For the initialization conditions for registers, see the initialization conditions for the registers described in the relevant chapter.*
- *For details on APB bus resets (APBC1_PSR), see Chapter "Clock".*

## 3.3 Reset Sequence

This family initiates the program and hardware operations starting with the initial state when a reset factor is cleared.
This family of operations starting with the reset and ending with the initiation of the operations is called a reset sequence.
The following explains a reset sequence.

### State transition diagram for resets

The following diagram shows a transition of reset states. The detailed operations are given in the following sections "3.4 Operations after Resets are Cleared".



1. Capturing reset factors

   Reset factors are captured and retained until a reset is issued to the device.

2. Issuing resets

   When a reset is ready to be issued, a device internal reset is issued.

3. Clearing resets

   When a reset factor is cleared, a device internal reset is extended for the amount of time required to clear the reset (for example, a wait time required until oscillation of a high-speed CR has become stable). When the extended period of time has expired, the reset is cleared.

4. Determining operation mode

   The operation mode is determined as PONR, LVDH or INITX is cleared and notified to each piece of the hardware. Any other reset factors do not cause the operation mode to change.

5. Reset vector fetch

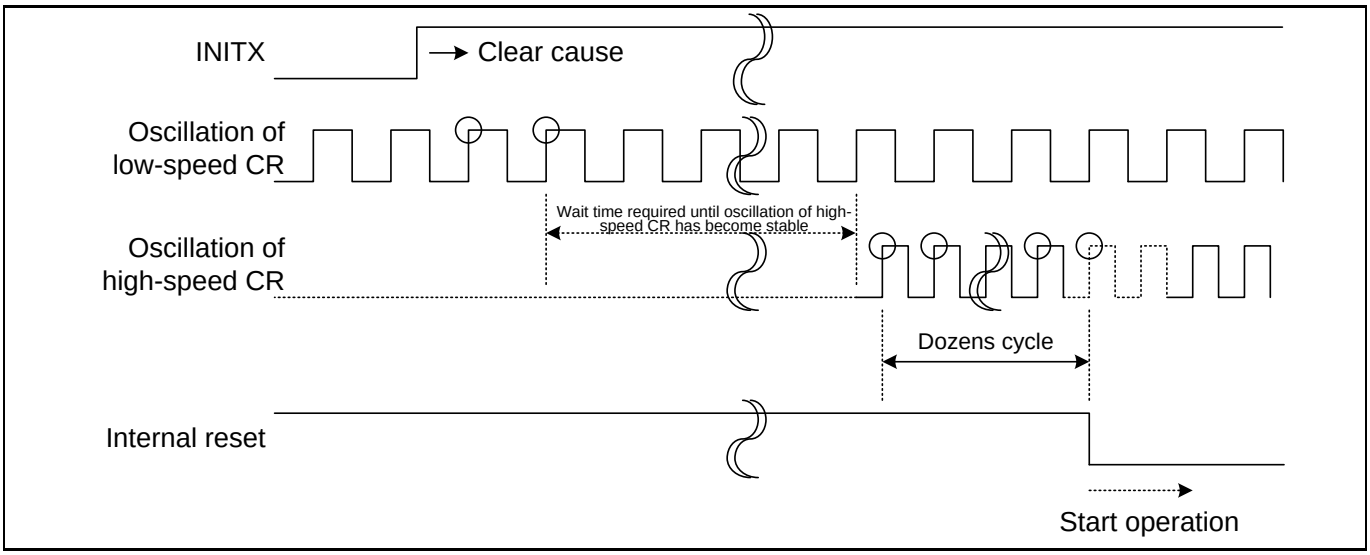
   After a device internal reset is cleared, the CPU starts fetching a reset vector. The CPU fetches the obtained reset vector into the program counter and starts programmed operations.

## 3.4 Operations after Resets are Cleared

**PONR, LVDH, INITX, HWDGR, SWDGR, CSVR, FCSR, DSTR**

Figure 3-1 provides an example of the operation waveform after a cause of INITX pin input reset has been cleared.

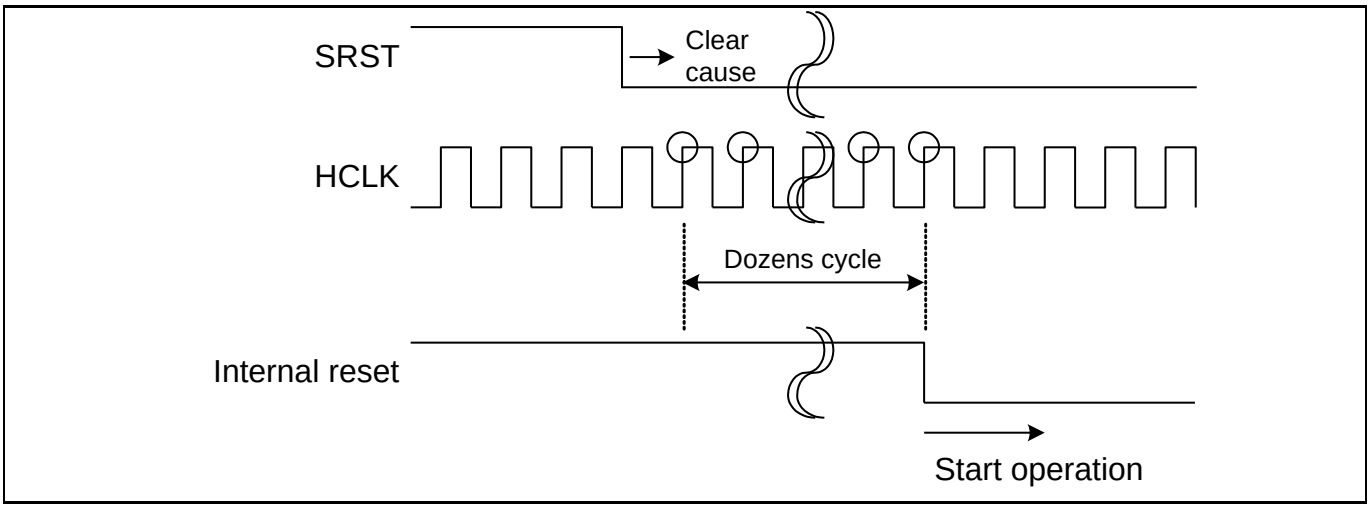
**Figure 3-1 Operation waveform example after INITX pin input reset has been cleared**




**SRST**

Figure 3-2 shows an example of the operation waveform after a software reset has been cleared.

**Figure 3-2 Operation waveform example after a software reset has been cleared**



# 4. Register

This section describes the register for resets.

### List of register for resets

**Table 4-1 List of register for resets**

| Abbreviation | Register name | Reference |
|---|---|---|
| RST_STR | Reset Factor Register | 4.1 |

## 4.1 Reset Factor Register (RST_STR)

The Reset Factor Register (RST_STR) shows causes of resets that have just occurred and initializes values upon power-on.
Reading the register clears all bits.
It stores all reset factors that have been generated until after it has been read upon power-on.

### Register configuration

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | SRST |
| Attribute | - | | | | | | | R |
| Initial value | - | | | | | | | 0 |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | FCSR | CSVR | HWDT | SWDT | Reserved | | INITX | PONR |
| Attribute | R | R | R | R | - | | R | R |
| Initial value | 0 | 0 | 0 | 0 | - | | 0 | 1 |

Note: The initial value is the value upon power-on.

### Register functions

#### [bit15:9] Reserved: Reserved bits

The read value is undefined. These bits have no effect when written.

#### [bit8] SRST: Software reset flag

Indicates a reset that is generated by writing "1" to Cortex-M0+ internal reset control register (SYSRESETREQ bit).

When a software reset is generated, SRST is enabled (SRST = 1).

| bit | Description |
|---|---|
| 0 | A software reset has not been issued. |
| 1 | A software reset has been issued. |

#### [bit7] FCSR: Flag for anomalous frequency detection reset

Indicates a reset when an anomalous frequency is detected in the main oscillation.

When the frequency of the main oscillation is outside of a given setting, a reset is issued and FCSR is enabled (FCSR = 1).

| bit | Description |
|---|---|
| 0 | An anomalous frequency detection reset has not been issued. |
| 1 | An anomalous frequency detection reset has been issued. |

### [bit6] CSVR: Clock failure detection reset flag

Indicates a reset when a failure is detected in the main or sub oscillation.

If a stop is detected, a reset is issued and CSVR is enabled (CSVR = 1).

| bit | Description |
|---|---|
| 0 | A clock failure detection reset has not been issued. |
| 1 | A clock failure detection reset has been issued. |

Note: Please refer to another chapter "Clock supervisor" for the method of judging whether the main oscillation or the sub oscillation broke down.

### [bit5] HWDT: Hardware watchdog reset flag

Indicates a reset from the hardware watchdog timer.

If the timer underflows, a reset is issued and HWDT is enabled (HWDT = 1).

| bit | Description |
|---|---|
| 0 | A hardware watchdog reset has not been issued. |
| 1 | A hardware watchdog reset has been issued. |

### [bit4] SWDT: Software watchdog reset flag

Indicates a reset from the software watchdog timer.

If the timer overflows, a reset is issued and SWDT is enabled (SWDT = 1).

| bit | Description |
|---|---|
| 0 | A software watchdog reset has not been issued. |
| 1 | A software watchdog reset has been issued. |

### [bit3:2] Reserved: Reserved bits

The read value is undefined. These bits have no effect when written.

### [bit1] INITX: INITX pin input reset flag

Indicates a reset that is externally input.

If a reset is externally input, INITX is enabled (INITX = 1).

| bit | Description |
|---|---|
| 0 | An INITX pin input reset has not been issued. |
| 1 | An INITX pin input reset has been issued. |

### [bit0] PONR: Power-on reset/low-voltage detection reset flag

Indicates a reset a reset at power-on and when low voltage is detected.

If a rising edge of power supply or a low-voltage is detected, a reset is issued and the PONR is enabled (PONR = 1).

| bit | Description |
|---|---|
| 0 | A power-on reset or low-voltage detection reset has not been issued. |
| 1 | A power-on reset or low-voltage detection reset has been issued. |

***Notes:***

- *This register is initialized by a power-on reset or a low-voltage detection reset. It is not initialized by any other reset factors. Reading the register clears all bits.*
- *Determine whether it is a return from deep standby mode or not by deep standby return factor registers 1 and 2 (WRFSR and WIFSR). For details, see "8.5 Deep standby return factor register 1(WRFSR)" and "8.6 Deep standby return factor register 2(WIFSR)" in Chapter "Low Power Consumption Mode".*

# CHAPTER5: Low-voltage Detection

**This chapter explains the functions and operations of the Low-voltage Detection Circuit.**



CODE: 9AFLVD-E01.0

# 1. Overview

The Low-voltage Detection Circuit monitors the power supply voltage, and generates reset and interrupt signals when the power supply voltage falls below the detection voltage.

**Overview of Low-voltage Detection Circuit**

■ Operations of Low-voltage Reset Circuit

- This circuit monitors the power supply voltage (VCC) and generates a reset signal when the power supply voltage falls below the specified value.
- This circuit allows selection of whether to enable or stop operations. The initial state is operating.
- This circuit allows specification of the detection voltage. However, when the low-voltage reset is generated, the set value is initialized. So, the release voltage becomes the initial value. If the power supply voltage is higher than the release voltage, the reset is released.
- This circuit monitors the power supply voltage even in standby modes and deep standby modes.
- This circuit generates a reset signal when the reduction of the power supply voltage is detected in standby modes and deep standby modes.

■ Operations of Low-voltage Interrupt Circuit

- This circuit monitors the power supply voltage (VCC) and generates an interrupt signal when the power supply voltage falls below the specified value.
- This circuit allows selection of whether to enable or stop operations. The initial state is set to disable.
- This circuit allows specification of the detection voltage.
- This circuit can monitor the power supply voltage even in standby modes and deep standby modes.
- This circuit returns from standby mode and deep standby modes when the reduction of the power supply voltage is detected in the standby mode and deep standby modes.

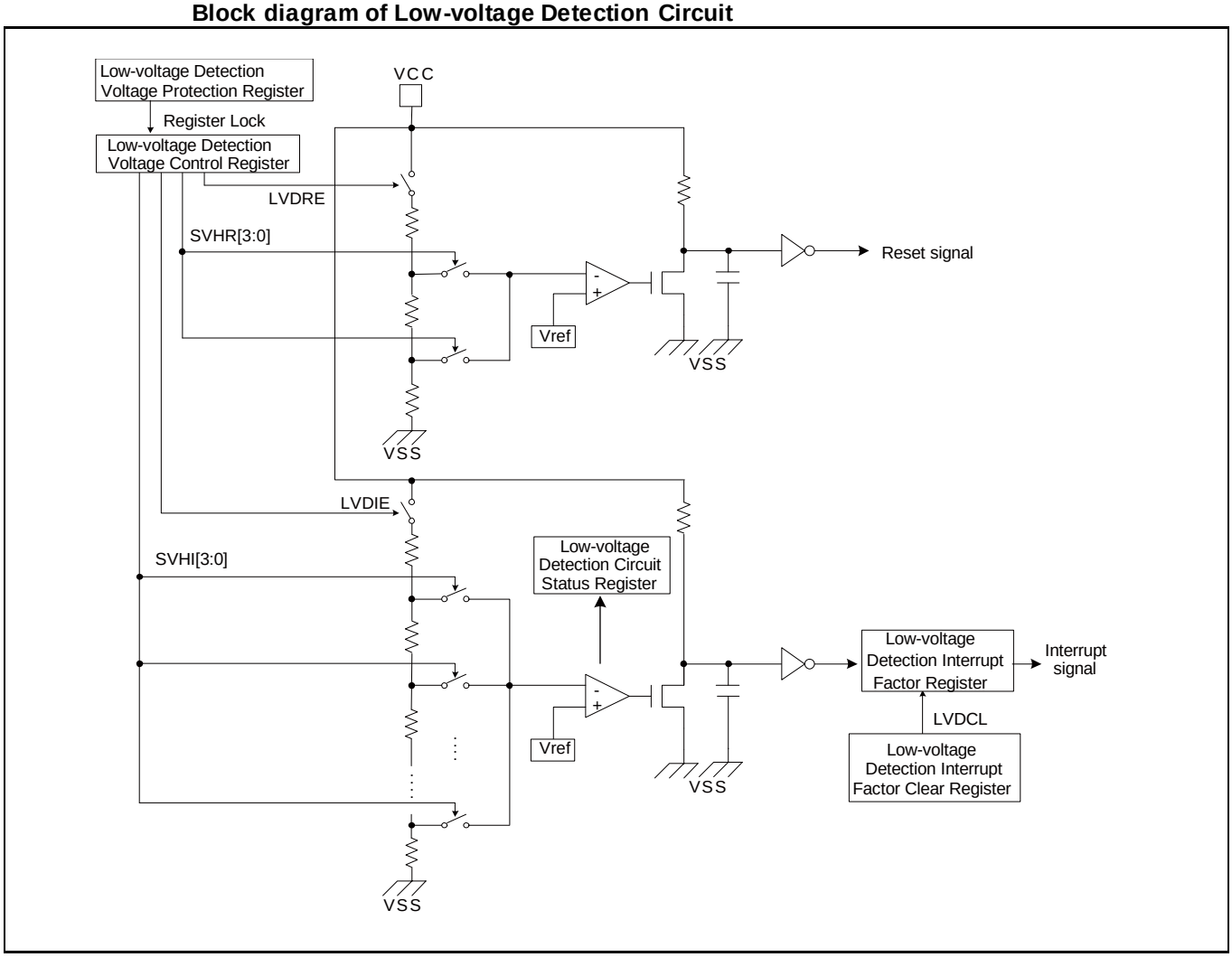
***Notes:***

- *If a low-voltage detection interrupt is enabled or the detection voltage is specified for a low-voltage detection interrupt, this circuit starts VCC voltage monitoring after the stabilization wait time of the Low-voltage Detection Circuit has lapsed.*
  *For the stabilization wait time of the Low-voltage Detection Circuit, refer to "Data Sheet" of the product used.*
- *This circuit does not conduct monitoring the power supply voltage if PCLK1 is stopped by TIMER mode, RTC mode, STOP mode, Deep standby RTC mode, Deep standby STOP mode, or APB1 Prescaler Register (APBC1_PSR) while waiting for the stabilization of the Low-voltage Detection Circuit. After the status flag is read and the stabilization wait time has lapsed, change to the desired mode.*
- *The Low-voltage Detection Voltage Control Register (LVD_CTL) is write-protected to prevent a writing error. To release write protection mode, write 0x1ACCE553 to the Low-voltage Detection Voltage Protection Register (LVD_RLR).*

# 2. Configuration

This section shows the block diagram of the Low-voltage Detection Circuit.

**Block diagram of Low-voltage Detection Circuit**




- Low-voltage Detection Voltage Control Register (LVD_CTL)
  This register controls whether to enable monitoring the power supply voltage for a low-voltage detection reset and low-voltage detection interrupt, and specifies the detection voltage for a low-voltage detection reset and low-voltage detection interrupt.
- Low-voltage Detection Voltage Protection Register (LVD_RLR)
  This register write-protects the Low-voltage Detection Voltage Control Register.
- Low-voltage Detection Interrupt Factor Register (LVD_STR)
  This register holds a low-voltage detection interrupt factor.
- Low-voltage Detection Interrupt Factor Clear Register (LVD_CLR)
  This register clears a low-voltage detection interrupt factor.
- Low-voltage Detection Circuit Status Register (LVD_STR2)
  This register checks the operation status of a low-voltage detection interrupt circuit.

### Pins of Low-voltage Detection Circuit

The following shows the pins used in the Low-voltage Detection Circuit.

- VCC pin

  The Low-voltage Detection Circuit monitors the power supply voltage of this pin.

- VSS pin

  This pin is a GND pin used as a basis to detect the power supply.

# 3. Operations

This section explains the operations of the Low-Voltage Detection Reset Circuit and the Low-voltage Detection Interrupt Circuit.

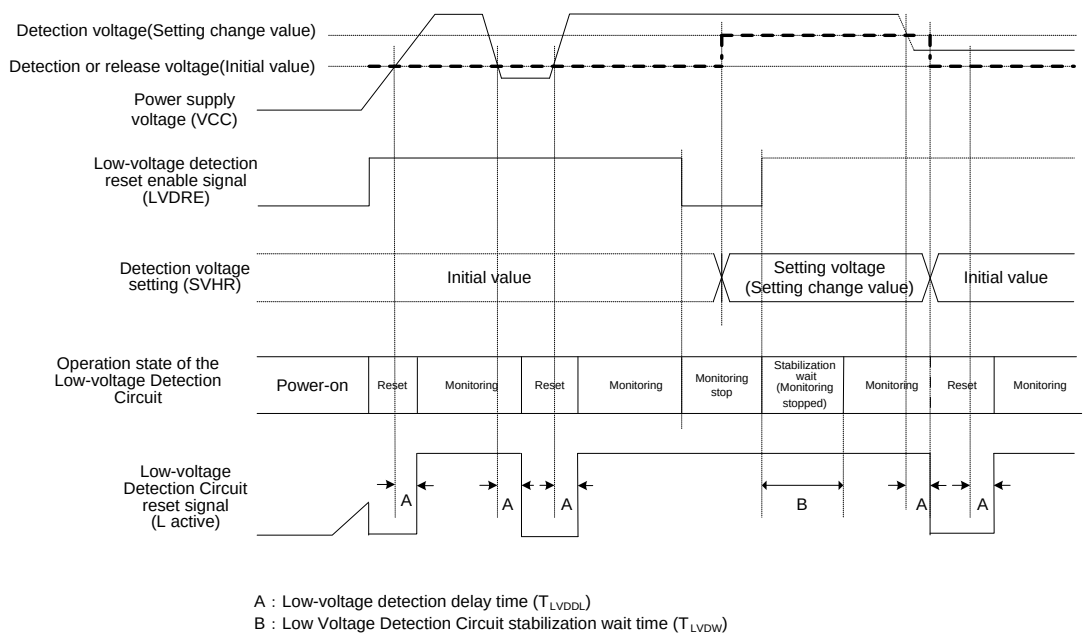
### Operations of Low-Voltage Detection Reset Circuit

■ Explanation of Circuit Operation

The Low-Voltage Detection Reset Circuit enters a monitoring state after power-on. This circuit generates a reset signal when the specified power supply voltage (VCC) falls below the detection voltage. A reset is released when the power supply voltage exceeds the release voltage.

Reset operation is valid only when LVDRE bit of Low-voltage Detection Voltage Control Register (LVD_CTL) is "1". Detection voltage of the reset can be set by SVHR bit of Low-voltage Detection Voltage Control Register (LVD_CTL). However, SVHR bit is initialized by the low-voltage detection reset. So, the release voltage becomes the initial value. If the power supply voltage is higher than the release voltage, the reset is released. When reset enable and reset detection voltage are set, Low-voltage detection reset status flag (LVDRRDY) of Low-voltage Detection Circuit Status Register (LVD_STR2) becomes "1" and starts power supply voltage monitoring after the stabilization wait period for the low-voltage detection circuit lapses.

This circuit is available in standby modes (SLEEP mode, TIMER mode, RTC mode, STOP mode) and deep standby modes (Deep standby RTC mode, Deep standby STOP mode).



### Operations of Low-voltage Detection Interrupt Circuit

■ Explanation of Circuit Operation

The Low-voltage Detection Interrupt Circuit monitors the power supply voltage (VCC) and generates an interrupt signal when the power supply voltage falls below the specified value.

An interrupt request is enabled when "1" is set to the LVDIE bit of the Low-voltage Detection Voltage Control Register (LVD_CTL). The initial value is set to Not Enable ("0"). The interrupt detection voltage can be set by the SVHI bit of the Low-voltage Detection Voltage Control Register (LVD_CTL). When an interrupt request is enabled and the interrupt detection voltage is specified, Low-voltage detection interrupt status flag (LVDIRDY) of Low-voltage Detection Circuit Status Register (LVD_STR2) is set to "1" and this circuit starts monitoring the power supply voltage after the stabilization wait time of the Low-voltage Detection Circuit has lapsed.

This circuit is available in standby modes (SLEEP mode, TIMER mode, RTC mode, STOP mode) and deep standby modes (Deep standby RTC mode, Deep standby STOP mode). It is also applicable when the CPU returns from those modes.

■ Low-voltage detection interrupt request

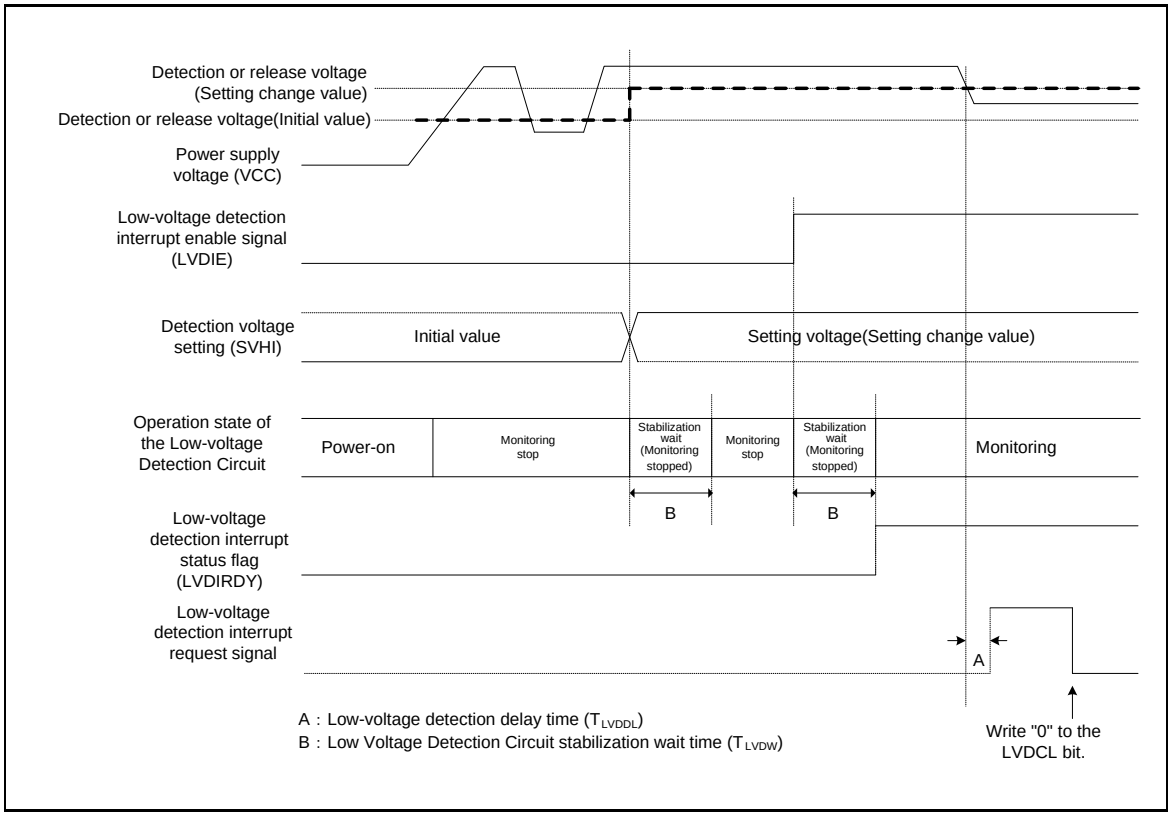
When the power supply voltage (VCC) falls below the specified voltage while a low-voltage detection interrupt is effective, "1" is set to the LVDIR bit of the Low-voltage Detection Interrupt Factor Register (LVD_STR) to generate an interrupt request signal.

An interrupt request can be checked by reading the LVDIR bit.

■ Canceling a low-voltage detection interrupt request

To cancel a low-voltage detection interrupt request, write "0" to the LVDCL bit of the Low-voltage Detection Interrupt Factor Clear Register (LVD_CLR). This clears a low-voltage detection interrupt factor and cancels a low-voltage detection interrupt request.

Even when the power supply voltage is below the specified detection voltage, an interrupt request is canceled when "0" is written to the LVDCL bit.



***Note:***

- *This circuit does not conduct monitoring the power supply voltage if PCLK1 is stopped by TIMER mode, RTC mode, STOP mode, Deep standby RTC mode, Deep standby STOP mode or APB1 Prescaler Register (APBC1_PSR) while waiting for the stabilization of the Low-voltage Detection Circuit. After checking that Low-voltage detection interrupt status flag (LVDIRDY) of Low-voltage Detection Circuit Status Register (LVD_STR2) is set to "1", change to the desired mode.*

# 4. Setup Procedure Examples

This section explains the procedures to set up the Low-voltage Detection Circuit, giving examples.

**Figure 4-1 Setting procedure example for low-voltage detection reset**

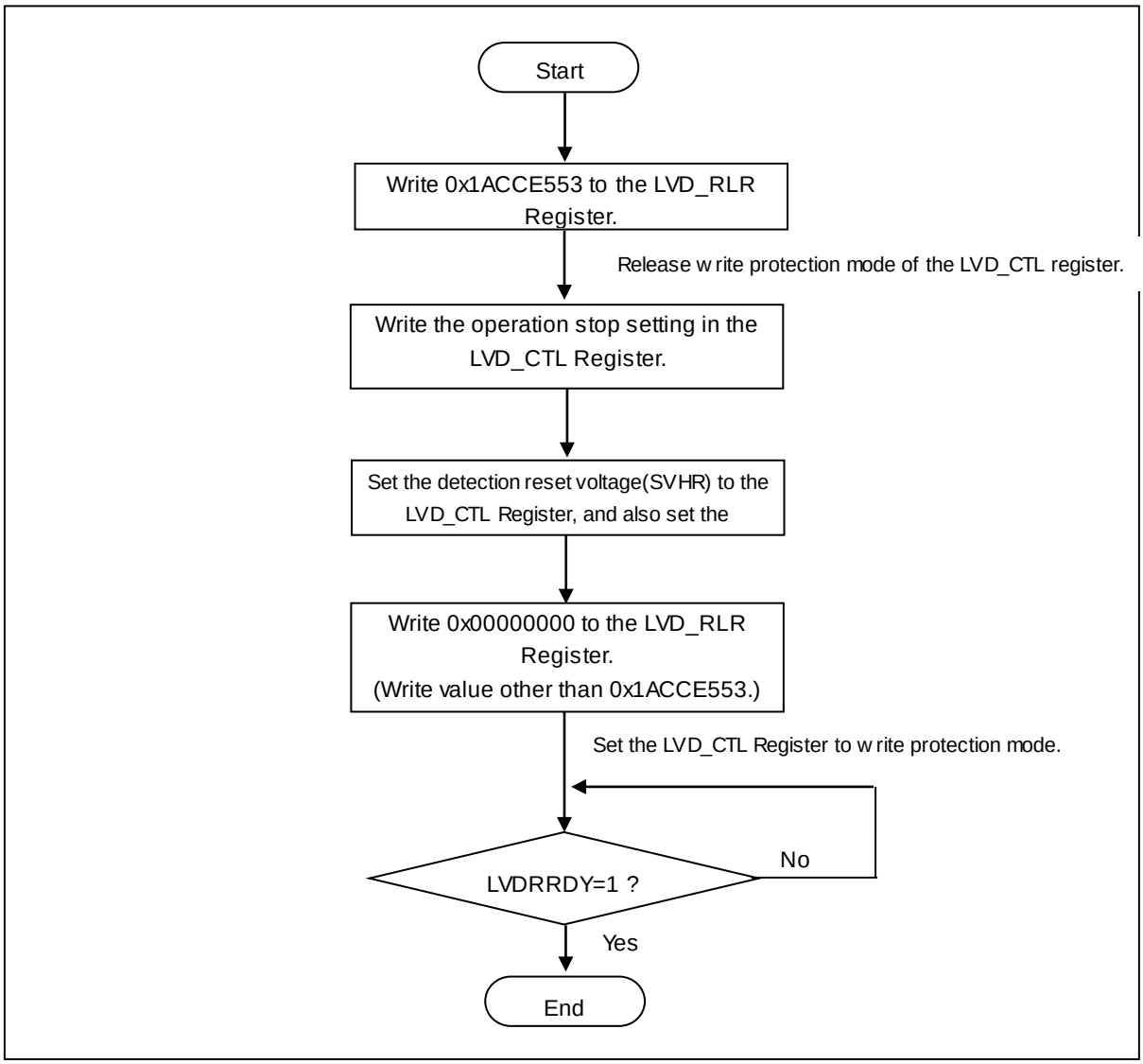
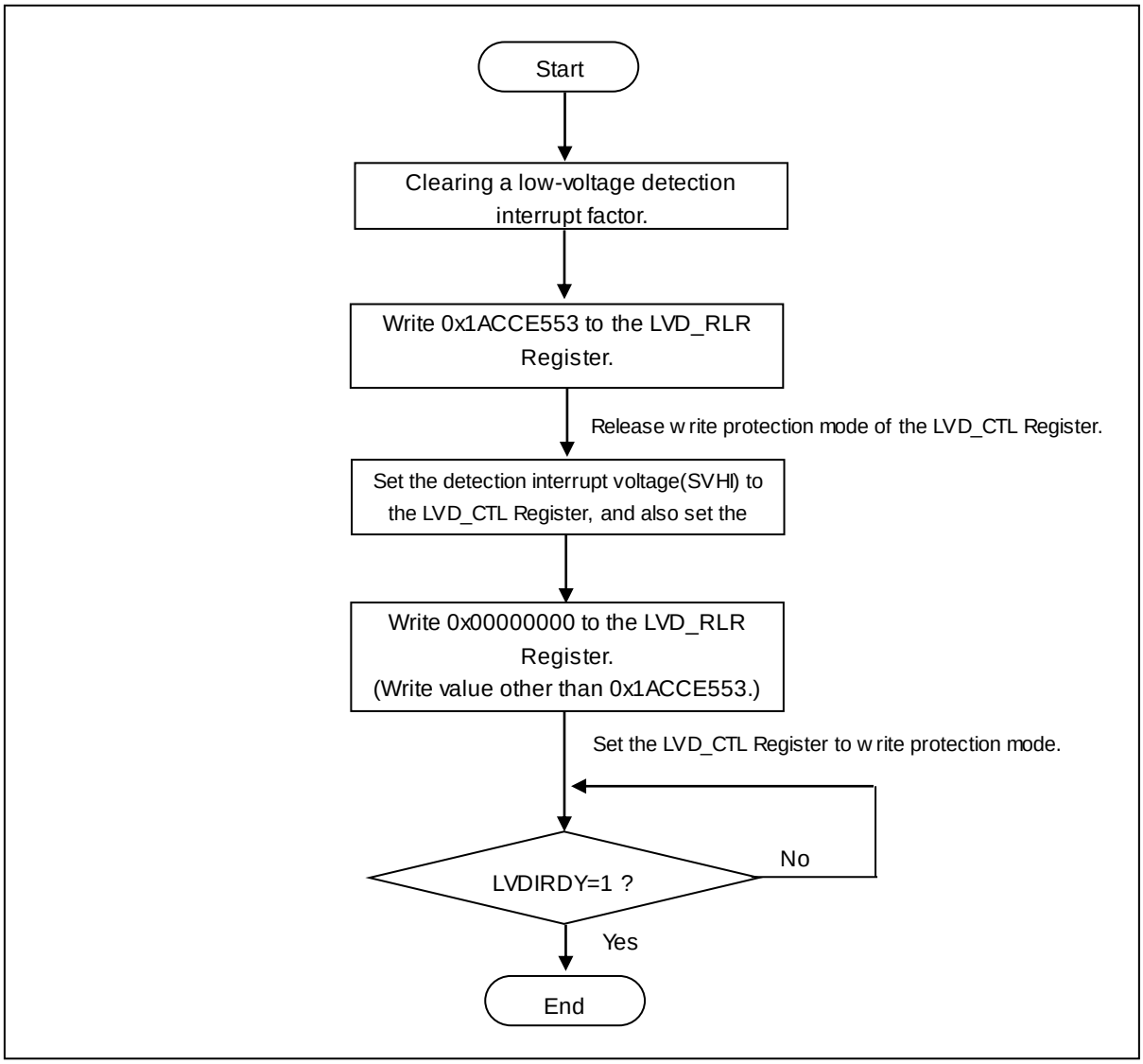


**Figure 4-2 Setting procedure example for low-voltage detection interrupt**



# 5. Registers

This section explains the configuration and functions of the registers used in the Low-voltage Detection Circuit.

### List of low-voltage detection circuit registers

**Table 5-1 List of low-voltage detection circuit registers**

| Abbreviation | Register name | Reference |
|---|---|---|
| LVD_CTL | Low-voltage Detection Voltage Control Register | 5.1 |
| LVD_STR | Low-voltage Detection Interrupt Factor Register | 5.2 |
| LVD_CLR | Low-voltage Detection Interrupt Factor Clear Register | 5.3 |
| LVD_RLR | Low-voltage Detection Voltage Protection Register | 5.4 |
| LVD_STR2 | Low-voltage Detection Circuit Status Register | 5.5 |

## 5.1 Low-voltage Detection Voltage Control Register (LVD_CTL)

The Low-voltage Detection Voltage Control Register (LVD_CTL) controls whether to enable monitoring the power supply voltage for low-voltage detection reset and a low-voltage detection interrupt and specifies the detection voltage for low-voltage detection reset and a low-voltage detection interrupt.

### Register configuration

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | LVDRE | SVHR | | | | | Reserved | |
| Attribute | R/W | R/W | | | | | - | |
| Initial value | 1 | 00000 | | | | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | LVDIE | SVHI | | | | | Reserved | |
| Attribute | R/W | R/W | | | | | - | |
| Initial value | 0 | 00011 | | | | | 00 | |

### Register functions

#### [bit15] LVDRE: Low-voltage detection reset operation enable bit

This bit enables operation of power supply voltage monitoring of low-voltage detection reset. If it is not enabled, the low-voltage detection reset circuit stops operating.

| bit | Description |
|---|---|
| 0 | Generation of low-voltage detection reset is not enabled. |
| 1 | Generation of low-voltage detection reset is enabled. [initial value] |

#### [bit14:10] SVHR: Low-voltage detection reset voltage setting bits

These bits set detection voltage of low-voltage detection reset.

| bit14:10 | Description |
|---|---|
| 00000 | Set the low-voltage detection reset voltage in the vicinity of 2.45 V. [Initial value] |
| 00001 | Set the low-voltage detection reset voltage in the vicinity of 2.60 V. |
| 00010 | Set the low-voltage detection reset voltage in the vicinity of 2.70 V. |
| 00011 | Set the low-voltage detection reset voltage in the vicinity of 2.80 V. |
| 00100 | Set the low-voltage detection reset voltage in the vicinity of 3.00 V. |
| 00101 | Set the low-voltage detection reset voltage in the vicinity of 3.20 V. |
| 00110 | Set the low-voltage detection reset voltage in the vicinity of 3.60 V. |
| 00111 | Set the low-voltage detection reset voltage in the vicinity of 3.70 V. |
| 01000 | Set the low-voltage detection reset voltage in the vicinity of 4.00 V. |
| 01001 | Set the low-voltage detection reset voltage in the vicinity of 4.10 V. |
| 01010 | Set the low-voltage detection reset voltage in the vicinity of 4.20 V. |
| Others | Setting is prohibited. |

#### [bit9:8] Reserved: Reserved bits

The read value is always "0". These bits have no effect on the operation when written.

### [bit7] LVDIE: Low-voltage detection interrupt enable bit

This bit is used to enable monitoring the power supply voltage of a low-voltage detection interrupt. When not enabling monitoring the power supply voltage, the Low-voltage Detection Interrupt Circuit is stopped.

| bit | Description |
|---|---|
| 0 | Does not enable the generation of a low -voltage detection interrupt. [Initial value] |
| 1 | Enables the generation of a low -voltage detection interrupt. |

### [bit6:2] SVHI: Low-voltage detection interrupt voltage setting bits

These bits are used to specify the detection voltage of a low-voltage detection interrupt.

| bit6:2 | Description |
|---|---|
| 00011 | Set the low -voltage detection interrupt voltage in the vicinity of 2.80 V.<br>[Initial value] |
| 00100 | Set the low -voltage detection interrupt voltage in the vicinity of 3.00 V. |
| 00101 | Set the low -voltage detection interrupt voltage in the vicinity of 3.20 V. |
| 00110 | Set the low -voltage detection interrupt voltage in the vicinity of 3.60 V. |
| 00111 | Set the low -voltage detection interrupt voltage in the vicinity of 3.70 V. |
| 01000 | Set the low -voltage detection interrupt voltage in the vicinity of 4.00 V. |
| 01001 | Set the low -voltage detection interrupt voltage in the vicinity of 4.10 V. |
| 01010 | Set the low -voltage detection interrupt voltage in the vicinity of 4.20 V. |
| Others | Setting is prohibited. |

### [bit1:0] Reserved: Reserved bits

The read value is always "0". These bits have no effect on the operation when written.

***Notes:***

- *The low-voltage detection interrupt enable bit (LVDIE) must be enabled after "0" was written to the LVDCL bit of the Low-voltage Detection Interrupt Factor Clear Register (LVD_CLR) to clear the low-voltage detection interrupt factor bit (LVDIR).*
- *When the low-voltage detection interrupt enable bit (LVDIE) is not enabled, the Low-voltage Detection Circuit for detecting a low-voltage interrupt is stopped. Therefore, the low-voltage detection interrupt factor bit (LVDIR) is not set.*
- *The Low-voltage Detection Voltage Control Register (LVD_CTL) is write-protected in the initial state, which makes writing invalid unless write protection mode is released. To write the Low-voltage Detection Voltage Control Register (LVD_CTL), set 0x1ACCE553 to the Low-voltage Detection Voltage Protection Register (LVD_RLR) to release write protection mode.*
- *This register is initialized by power-on reset and low-voltage detection reset. It is not initialized by reset factor other than these resets.*
- *After the setting value of low-voltage detection reset voltage is changed, as the setting value is initialized by the low-voltage detection reset, the released voltage becomes the initial value. At that time, if the power supply voltage is higher than the released voltage, the reset is released. For the initial values of the detection voltage and the released voltage, see the data sheet of the product used.*
- *This register is not initialized by deep standby transition reset.*

## 5.2 Low-voltage Detection Interrupt Factor Register (LVD_STR)

The Low-voltage Detection Interrupt Factor Register (LVD_STR) holds a low-voltage detection interrupt factor.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | LVDIR | Reserved | | | | | | |
| Attribute | R | - | | | | | | |
| Initial value | 0 | 0000000 | | | | | | |

### Register functions

#### [bit7] LVDIR: Low-voltage detection interrupt factor bit

| bit | Description |
|---|---|
| 0 | A low-voltage detection interrupt request is not detected. [Initial value] |
| 1 | A low-voltage detection interrupt request has been detected. |

#### [bit6:0] Reserved: Reserved bits

The read value is always "0". These bits have no effect on the operation when written.

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 5.3 Low-voltage Detection Interrupt Factor Clear Register (LVD_CLR)

The Low-voltage Detection Interrupt Factor Clear Register (LVD_CLR) clears a low-voltage detection interrupt factor.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | LVDCL | Reserved | | | | | | |
| Attribute | R/W | - | | | | | | |
| Initial value | 1 | 0000000 | | | | | | |

### Register functions

#### [bit7] LVDCL: Low-voltage detection interrupt factor clear bit

| bit | Description |
|---|---|
| 0 | Clears the low-voltage detection interrupt factor bit (LVDIR) of the Low-voltage Detection Interrupt Factor Register (LVD_STR) to "0". |
| 1 | Has no effect on the operation in write mode. [Initial value] |

"1" is always set in read mode.

#### [bit6:0] Reserved: Reserved bits

The read value is always "0". These bits have no effect on the operation when written.

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 5.4 Low-voltage Detection Voltage Protection Register (LVD_RLR)

The Low-voltage Detection Voltage Protection Register (LVD_RLR) write-protects the Low-voltage Detection Voltage Control Register (LVD_CTL).

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | LVDLCK[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | LVDLCK[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0001 | | | | | | | | | | | | | | | |

**Register functions**

**[bit31:0] LVDLCK[31:0]: Low-voltage Detection Voltage Control Register protection bits**

- Setting 0x1ACCE553 enables writing the Low-voltage Detection Voltage Control Register (LVD_CTL) (releases write protection mode).
- Setting a value other than 0x1ACCE553 disables writing the Low-voltage Detection Voltage Control Register (LVD_CTL) (enables write protection mode).
- When the Low-voltage Detection Voltage Control Register (LVD_CTL) is not set in write protection mode, 0x00000000 is read.
- When the Low-voltage Detection Voltage Control Register (LVD_CTL) is set in write protection mode, 0x00000001 is read.

***Notes:***

- *The Low-voltage Detection Voltage Control Register (LVD_CTL) is write-protected in the initial state. To write the LVD_CTL Register, set 0x1ACCE553 to the Low-voltage Detection Voltage Protection Register (LVD_RLR) to release write protection mode.*
- *To enable write protection mode of the LVD_CTL register, set a value other than 0x1ACCE553 to the LVD_RLR register.*
- *Once write protection mode is released for the LVD_CTL Register, it remains released until a value other than 0x1ACCE553 is written to the LVD_RLR Register.*
- *This register is not initialized by deep standby transition reset.*

## 5.5 Low-voltage Detection Circuit Status Register (LVD_STR2)

The Low-voltage Detection Circuit Status Register (LVD_STR2) checks the operation status of a low-voltage detection interrupt.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | LVDIRDY | LVDRRDY | Reserved | | | | | |
| Attribute | R | R | - | | | | | |
| Initial value | 0 | 1 | 000000 | | | | | |

### Register functions

#### [bit7] LVDIRDY: Low-voltage detection interrupt status flag

| bit | Description |
|---|---|
| 0 | Stabilization wait state or monitoring stop state [Initial value] |
| 1 | Monitoring state |

This bit has no effect on the operation in write mode.

#### [bit6] LVDRRDY: Low-voltage detection reset status flag

| bit | Description |
|---|---|
| 0 | Stabilization wait state or monitoring stop state |
| 1 | Monitoring state [Initial value] |

This bit has no effect on the operation in write mode.

#### [bit5:0] Reserved: Reserved bits

The read value is always "0". These bits have no effect on the operation when written.

***Note:***

- *This register is not initialized by deep standby transition reset.*

# 6. Usage Precautions

This section explains the precautions for using Low-voltage Detection Circuit.

■ Low-voltage detection interrupt factor bit at STOP mode transition

Even if the power supply voltage is not greater than the detection voltage after clearing the Low-voltage Detection Interrupt factor bit (LVD_STRLVDIR), the interrupt factor will not be enabled again unless the power supply voltage becomes greater than the released voltage once.

But, the voltage comparison is executed by the Low-voltage Detection Circuit without fail when the transition to the STOP mode is executed while the power supply voltage is not greater than the detection voltage after clearing LVDIR.

So, the Low-voltage detection interrupt factor is set again by the transition to STOP mode and the process could go to the interrupt routine.

For example, in case of the transition to STOP mode in the interrupt routine of the Low-voltage detection, after clearing the interrupt factor and finishing the interrupt routine, the interrupt factor bit is set soon again and the interrupt routine might be repeated.

To prevent the occurrence of repeated interrupts when the voltage is not greater than the detection voltage after the detection of the low-voltage interrupt, disable the Low-voltage detection interrupt enable bit (LVDIE) and get out of the routine.

■ Setup of Detection Voltage of Low-voltage Detection Reset

When the Low-voltage detection reset is generated after setting the detection voltage of Low-voltage detection reset voltage, the detection voltage setting value is initialized. When the power supply voltage is higher than the initial setup value as in the case where the power supply voltage lowers slowly, the reset is released. But, as the program is returned to the beginning by the reset, set the detection voltage again. As the power supply voltage is already lower than the detection voltage, the Low-voltage detection reset is set again.

That is to say, the loop of the detection voltage change, reset, initialization (returned to the beginning), the detection voltage change, and reset could be repeated according to the power supply voltage change and program description.

The following measures can be taken to prevent the loop of the Low-voltage detection reset when the power supply voltage is not greater than the detection voltage:

- For the setting value of the Low-voltage detection reset, only the initial value is used.
- Set the Low-voltage detection interrupt before the Low-voltage detection reset. Confirm whether an interrupt flag is set at the beginning of the program and change the setting value of the Low-voltage reset detection voltage, if required.

# CHAPTER6: Low Power Consumption Mode

**This chapter explains the functions and operations of low power consumption mode.**



CODE: 9AFLPMODE-E01.0

# 1. Overview

To reduce the power consumption, the system provides low power consumption mode, which enables the use of the standby mode of SLEEP, TIMER, RTC and STOP modes and the deep standby mode of deep standby RTC and deep standby STOP modes.

### Overview of CPU operation modes

CPU operation modes are classified into the following types.

- Run modes
  - High-speed CR run mode
  - Main run mode
  - PLL run mode
  - Low-speed CR run mode
  - Sub run mode
- Standby modes
  - Sleep modes
    - High-speed CR sleep mode
    - Main sleep mode
    - PLL sleep mode
    - Low-speed CR sleep mode
    - Sub sleep mode
  - Timer modes
    - High-speed CR timer mode
    - Main timer mode
    - PLL timer mode
    - Low-speed CR timer mode
    - Sub timer mode
  - RTC mode
  - STOP mode
- Deep Standby modes
  - Deep Standby RTC mode
  - Deep Standby STOP mode

### Table of low power consumption modes equipped in each TYPE

**Table 1-1 Table of Low Power Consumption Mode**

| Mode | TYPE1 |
|---|---|
| Standby mode | ○ |
| Deep standby mode | - |

### Overview of RUN mode

RUN mode is defined with a clock selected as a master clock. The base clocks, which are obtained by dividing the master clock frequency, are supplied to CPU clock, AHB bus clock, and APB bus clock to run the CPU, buses, and most peripherals.

The source clock frequency can be changed dynamically. When not using the main or sub oscillator, the source clock oscillator can be stopped.

RUN mode is divided into the following modes depending on the clock selected as a master clock.

- High-speed CR run mode

In this mode, the high-speed CR oscillator clock is used as a master clock. When not using the main or sub oscillator, the respective oscillators can be stopped. The status of PLL Multiplier Circuit varies depending on the setting of the PLLE bit. The low-speed CR oscillator is always set to the active state. It changes to this mode after a reset has been released.

- Main run mode

In this mode, the main oscillator clock is used as a master clock. The statuses of the high-speed CR oscillator PLL Multiplier Circuit and sub oscillator vary depending on the settings of the HCRE bit, PLLE bit and SOSCE bit respectively. The low-speed CR oscillator is always set to the active state.

- PLL run mode

In this mode, the PLL clock obtained by multiplying the main oscillator clock or high-speed CR oscillator clock is used as a master clock. The low-speed CR oscillator is always set to the active state. The status of the sub oscillator varies depending on the setting of the SOSCE bit. The statuses of the high-speed CR oscillator and main oscillator vary depending on the setting of the PINC bit in the PSW_TMR Register.

- Low-speed CR run mode

In this mode, the low-speed CR oscillator clock is used as a master clock. The status of the sub oscillator varies depending on the setting of the SOSCE bit. The main oscillator, high-speed CR oscillator, and PLL Multiplier Circuit are not available in this mode.

- Sub run mode

In this mode, the sub oscillator clock is used as a master clock. The low-speed CR oscillator is always set to the active state. The main oscillator, high-speed CR oscillator, and PLL Multiplier Circuit are not available in this mode.

### Overview of SLEEP mode

SLEEP mode is classified as one of standby modes. SLEEP mode is used to stop CPU clocks. This causes the CPU to be stopped, reducing the power consumption. The resources connected to the AHB and APB bus clocks continue operations.

SLEEP mode is divided into the following modes depending on a master clock at the transition to SLEEP mode.

- High-speed CR sleep mode

When the high-speed CR oscillator clock is selected as a master clock, the system changes to high-speed CR sleep mode if the transition to SLEEP mode is requested. In this mode, the status of the PLL Multiplier Circuit, main or sub oscillator varies depending on the setting of the PLLE, MOSCE or SOSCE bit. The low-speed CR oscillator is always set to the active state.

- Main sleep mode

When the main clock is selected as a master clock, the system changes to main sleep mode if the transition to SLEEP mode is requested. In this mode, the statuses of the high-speed CR oscillator PLL Multiplier Circuit and sub oscillator vary depending on the settings of the HCRE bit, PLLE bit and SOSCE bit respectively. The low-speed CR oscillator is always set to the active state.

- PLL sleep mode

When the main PLL clock is selected as a master clock, the system changes to PLL sleep mode if the transition to SLEEP mode is requested. In this mode, the low-speed CR oscillator is always set to the active state. The status of the sub oscillator varies depending on the setting of the SOSCE bit. The statuses of the high-speed CR oscillator and main oscillator vary depending on the setting of the PINC bit in the PSW_TMR Register.

- Low-speed CR sleep mode

When the low-speed CR clock is selected as a master clock, the system changes to low-speed CR sleep mode if the transition to SLEEP mode is requested. In this mode, the status of the sub oscillator varies depending on the setting of the SOSCE bit. The main oscillator, high-speed CR oscillator, and PLL Multiplier Circuit are not available in this mode.

- Sub sleep mode

When the sub clock is selected as a master clock, the system changes to sub sleep mode if the transition to SLEEP mode is requested. In this mode, the low-speed CR oscillator is always set to the active state. The main oscillator, high-speed CR oscillator, and PLL Multiplier Circuit are not available in this mode.

### Overview of TIMER mode

TIMER mode is classified as one of standby modes. TIMER mode is used to stop supplying a base clock. This causes the CPU clock, AHB bus clock, and all APB bus clocks to be stopped, reducing the power consumption. In this case, all functions are stopped, excluding the oscillators, PLL, hardware watchdog timer, watch counter, RTC, clock failure detector, and Low Voltage Detection Circuit.

TIMER mode is divided into the following modes depending on a master clock at the transition to TIMER mode.

- High-speed CR timer mode

When the high-speed CR oscillator clock is selected as a master clock, the system changes to high-speed CR timer mode if the transition to TIMER mode is requested. In this mode, the status of the PLL Multiplier Circuit, main or sub oscillator varies depending on the setting of the PLLE, MOSCE or SOSCE bit. The low-speed CR oscillator is always set to the active state.

- Main timer mode

When the main clock is selected as a master clock, the system changes to main timer mode if the transition to TIMER mode is requested. In this mode, the statuses of the high-speed CR oscillator PLL Multiplier Circuit and sub oscillator vary depending on the settings of the HCRE bit, PLLE bit and SOSCE bit respectively. The low-speed CR oscillator is always set to the active state.

- PLL timer mode

When the main PLL clock is selected as a master clock, the system changes to PLL timer mode if the transition to TIMER mode is requested. In this mode, the low-speed CR oscillator is always set to the active state. The status of the sub oscillator varies depending on the setting of the SOSCE bit. The statuses of the high-speed CR oscillator and main oscillator vary depending on the setting of the PINC bit in the PSW_TMR Register.

- Low-speed CR timer mode

When the low-speed CR clock is selected as a master clock, the system changes to low-speed CR timer mode if the transition to TIMER mode is requested. In this mode, the status of the sub oscillator varies depending on the setting of the SOSCE bit. The main oscillator, high-speed CR oscillator, and PLL Multiplier Circuit are not available in this mode.

- Sub timer mode

When the sub clock is selected as a master clock, the system changes to sub timer mode if the transition to TIMER mode is requested. In this mode, the sub oscillator and low-speed CR oscillator are always set to the active state. The main oscillator, high-speed CR oscillator, and PLL Multiplier Circuit are not available in this mode.

### Overview of RTC mode

RTC mode is classified as one of the standby modes. RTC mode stops oscillation other than that of the sub oscillator. All the functions except for watch counter, RTC, and low voltage detection circuit will be stopped.

### Overview of STOP mode

STOP mode is classified as one of standby modes. STOP mode is used to stop all oscillating operations. Enabling this mode stops all functions, excluding the Low Voltage Detection Circuit.

### Overview of deep standby RTC mode

Deep standby RTC mode is classified as one of the deep standby modes. Deep standby RTC mode stops oscillation other than that of the sub oscillator. All the functions except for the RTC and low voltage detection circuit will be stopped. It turns off RTC, low voltage detection circuit, CPUs other than GPIO, on-chip Flash memory, on-chip SRAM*, and peripheral functions inside the chip.

### Overview of deep standby stop mode

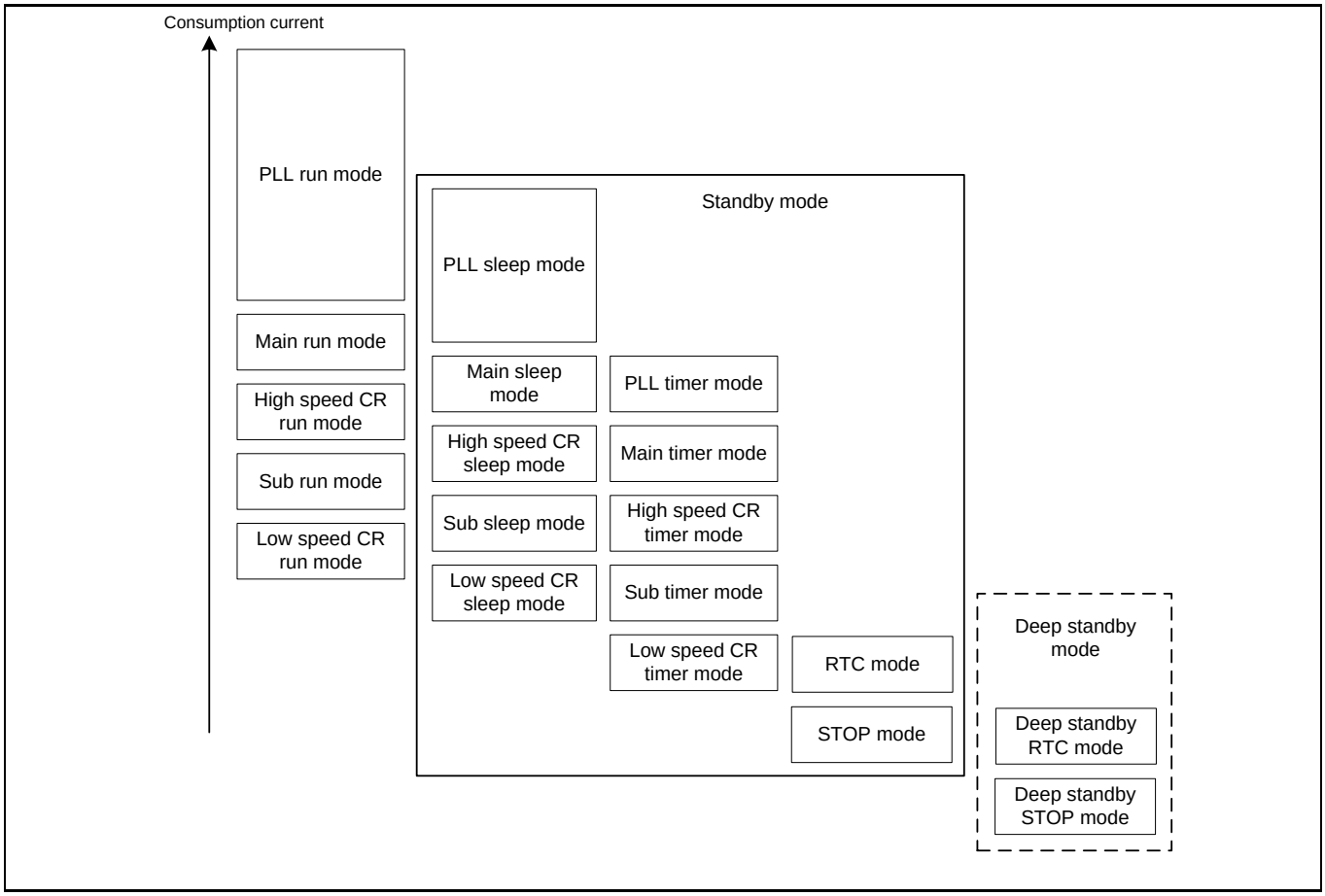
Deep standby stop mode is classified as one of the deep standby modes. Deep standby stop mode stops all oscillations. All the functions except for the low voltage detection circuit will be stopped. It turns off RTC, low voltage detection circuit, CPUs other than GPIO, on-chip Flash memory, on-chip SRAM*, and peripheral functions inside the chip.

*: Data in on-chip SRAM can be retained even in the deep standby modes.
In the setting to retain on-chip SRAM data, the power of on-chip SRAM is ON.

### Relationships between CPU operation modes and consumption current values

Figure 1-1 shows the relationships between CPU operation modes and consumption current values.

**Figure 1-1 Relationships between CPU operation modes and consumption current values**




***Note:***

- *Figure 1-1 shows only an overview of the magnitude relationship among consumption currents of each mode. The actual consumption current values vary depending on the oscillator and PLL starting conditions in each mode or the clock configuration of the selected frequency and other elements.*

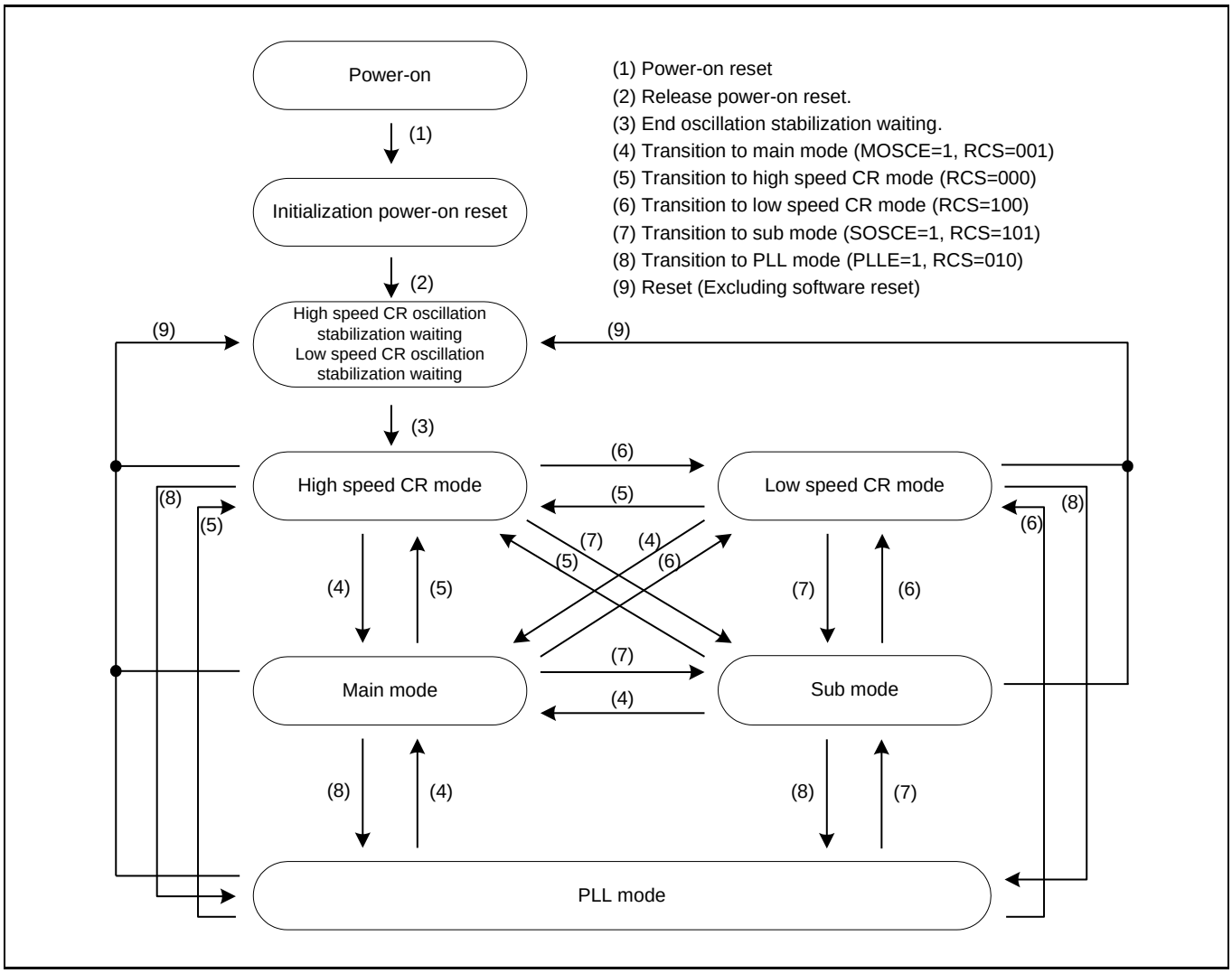
# 2. Configuration of CPU Operation Modes

This section explains the configuration of CPU operation modes.

### CPU operation mode transition diagram

Figure 2-1 shows the CPU operation mode transition diagram.

**Figure 2-1 CPU operation mode transition diagram**

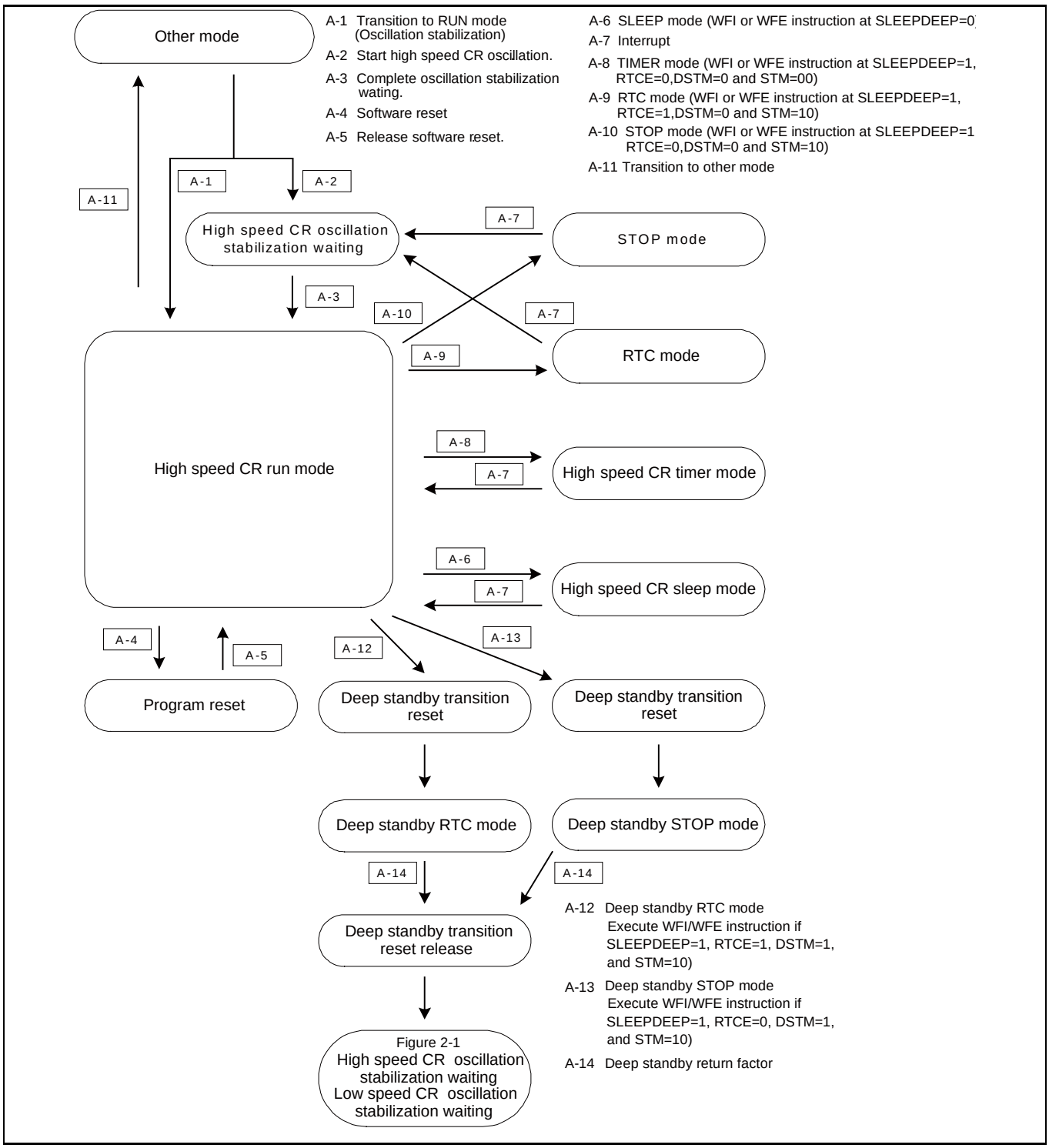



- High-speed CR mode
  In this mode, the high-speed CR oscillator clock is used as a master clock.
- Main mode
  In this mode, the main oscillator clock is used as a master clock.
- Low-speed CR mode
  In this mode, the low-speed CR oscillator clock is used as a master clock.
- Sub mode
  In this mode, the sub oscillator clock is used as a master clock.
- PLL mode
  In this mode, the PLL oscillator clock is used as a master clock.

■ High-speed CR mode transition diagram

In high-speed CR mode, the high-speed CR oscillator clock is used as a master clock.

**Figure 2-2 High-speed CR mode transition diagram**



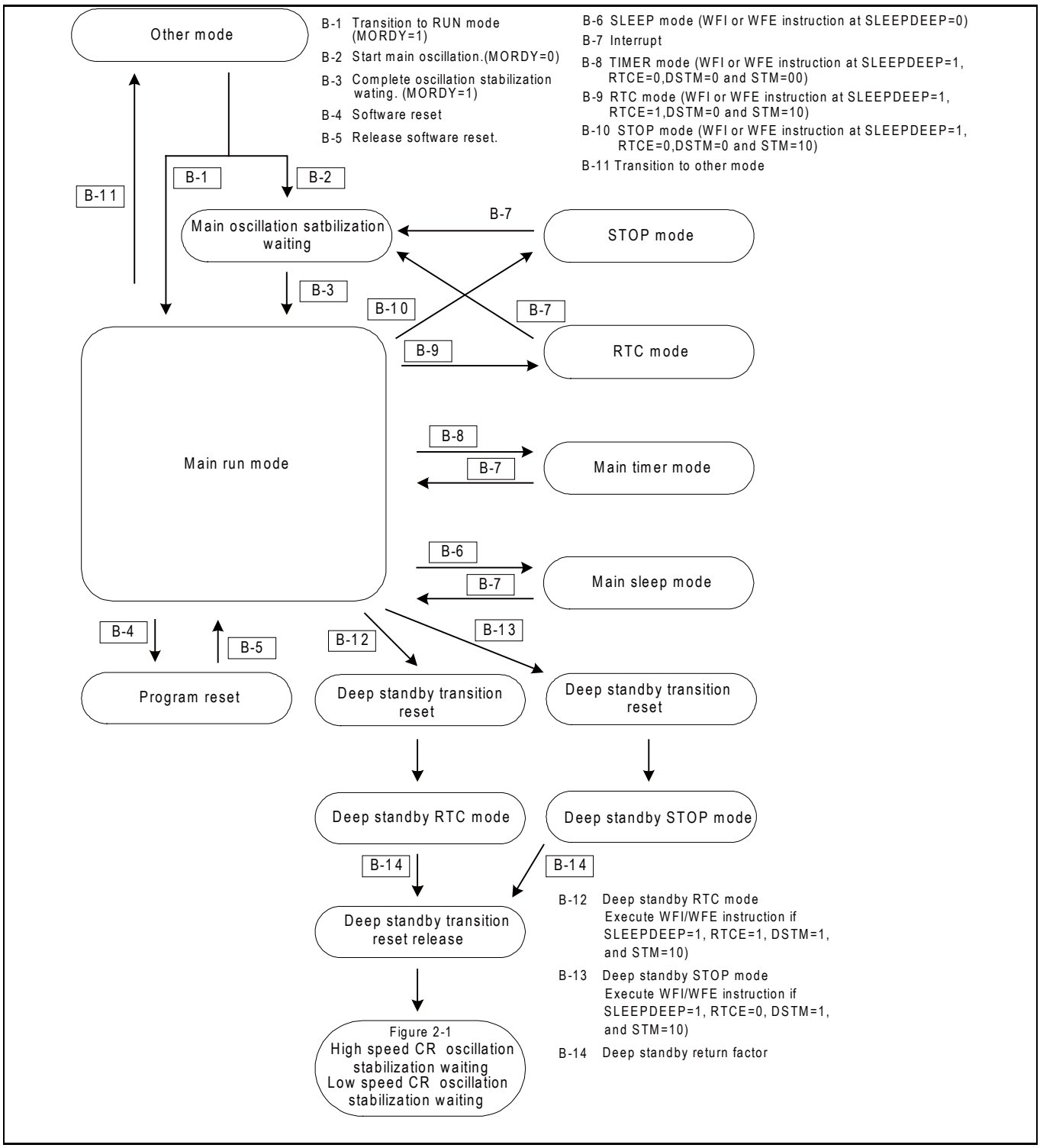

***Note:***

- *There are products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1 (table of low power consumption modes).*

### Main mode transition diagram

In main mode, the main oscillator clock is used as a master clock.

**Figure 2-3 Main mode transition diagram**



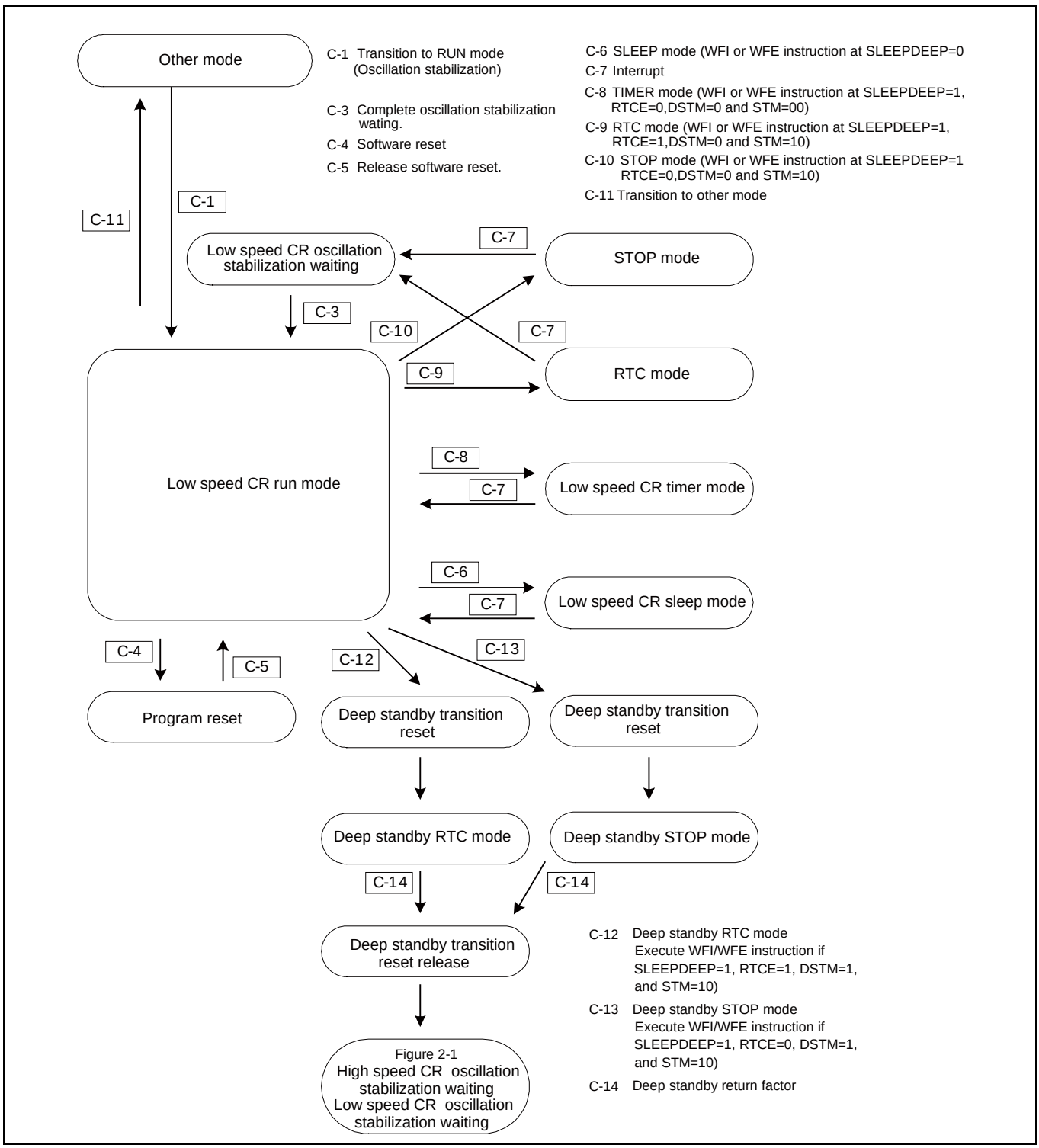

***Note:***

- *There are products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1 (table of low power consumption modes).*

### Low-speed CR mode transition diagram

In low-speed CR mode, the low-speed CR oscillator clock is used as a master clock.

**Figure 2-4 Low-speed CR mode transition diagram**



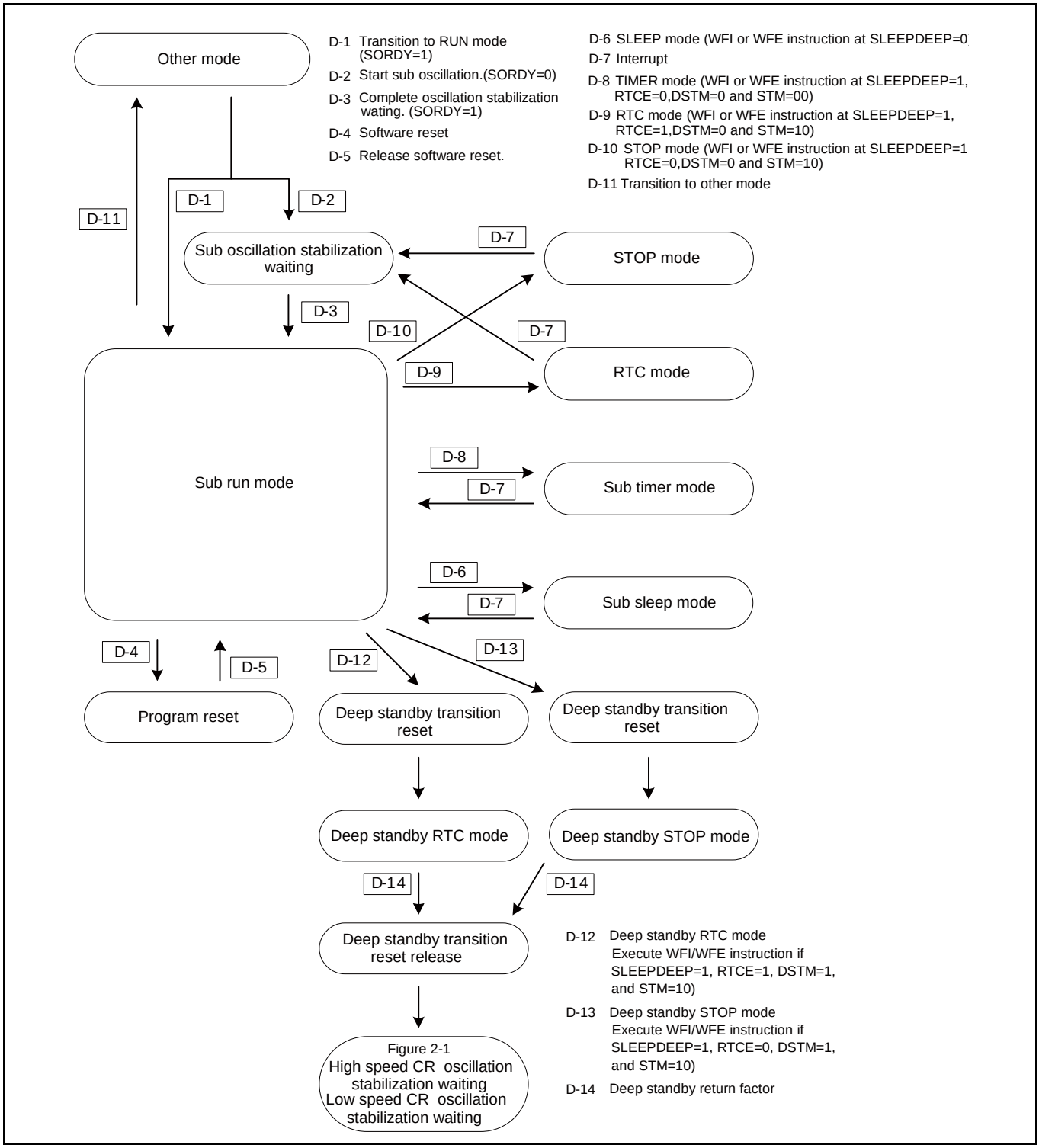

***Note:***

- *There are products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1 (table of low power consumption modes).*

### Sub mode transition diagram

In sub mode, the sub oscillator clock is used as a master clock.

**Figure 2-5 Sub mode transition diagram**



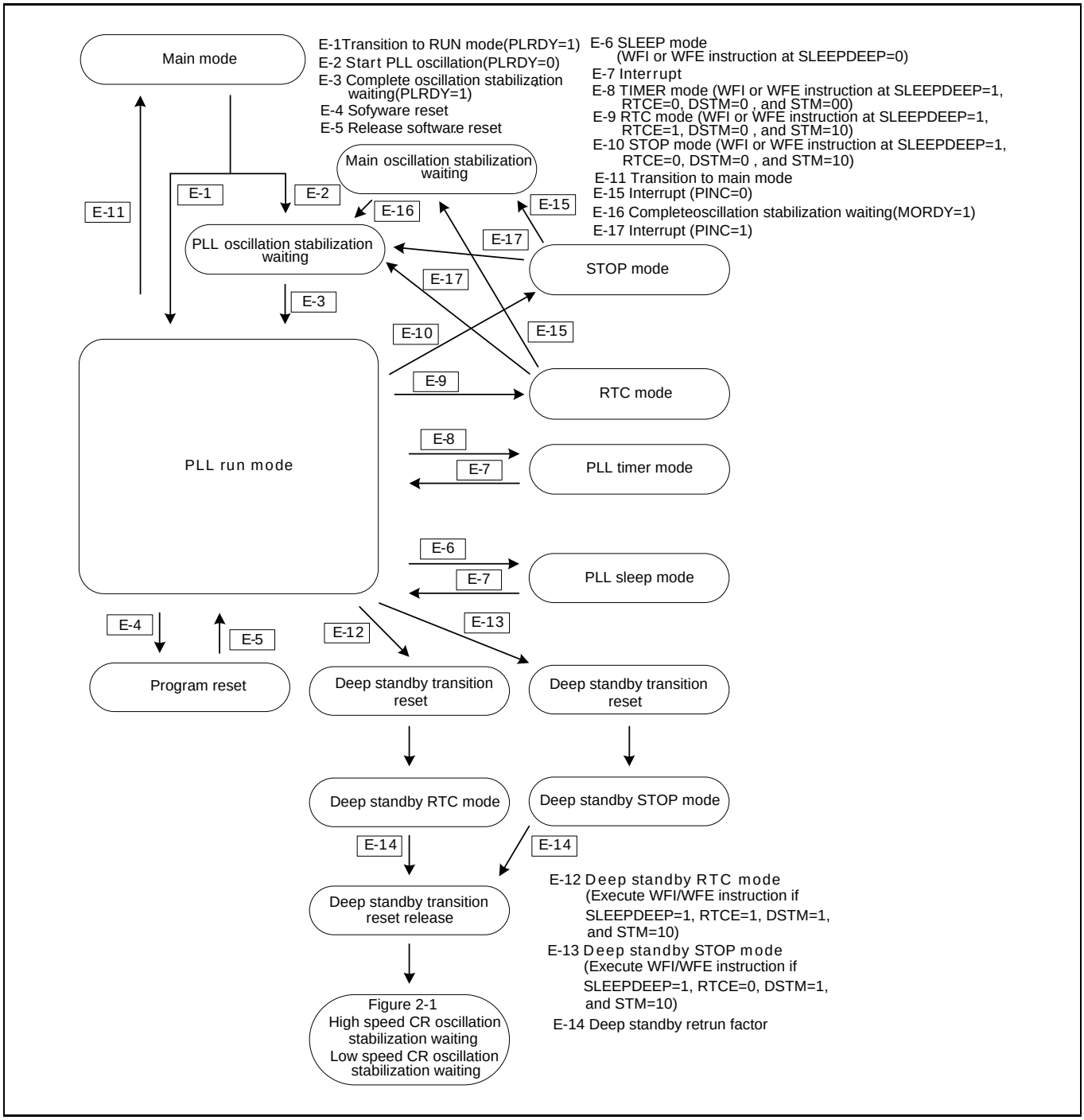

***Note:***

- *There are products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1 (table of low power consumption modes).*

### PLL mode transition diagram

In PLL mode, the main PLL clock is used as a master clock.

**Figure 2-6 PLL mode transition diagram**




***Note:***

- *There are products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1 (table of low power consumption modes).*

MOSCE : MOSCE bit in System Clock Mode Control Register (SCM_CTL)
SOSCE : SOSCE bit in System Clock Mode Control Register (SCM_CTL)
PLLE : PLLE bit in System Clock Mode Control Register (SCM_CTL)
RCS : RCS bit in System Clock Mode Control Register (SCM_CTL)
MORDY : MORDY bit in System Clock Mode Status Register (SCM_STR)
SORDY : SORDY bit in System Clock Mode Status Register (SCM_STR)
PLRDY : PLRDY bit in System Clock Mode Status Register (SCM_STR)
PINC : PINC bit in PLL Clock Stabilization Wait Time Setup Register (PSW_TMR)

*: For the SCM_CTL, SCM_STR and PSW_TMR Registers, refer to Chapter "Clock".

***Note:***

- *To return from low-speed CR timer mode, sub timer mode, RTC mode, STOP mode, Deep standby RTC mode, or Deep standby STOP mode, the voltage stabilization wait time (a few hundred μs) of the built-in regulator is ensured. After the wait time has lapsed, the system performs operations to return to each RUN mode.*

# 3. Operations of Standby Modes

This section explains operations of standby modes.
Standby modes are classified into four types: SLEEP modes (high-speed CR sleep, main sleep, PLL sleep, low-speed CR sleep, and sub sleep), TIMER modes (high-speed CR timer, main timer, PLL timer, low-speed CR timer, and sub timer), RTC mode and STOP mode.

### Clock operation states in standby modes

The table below shows the states of the oscillator clock, CPU clock, AHB bus clock, and APB bus clock in SLEEP, TIMER, RTC and STOP modes.

**Table 3-1 Clock operation states in SLEEP modes**

| | SLEEP modes | | | | |
|---|---|---|---|---|---|
| | **High-speed CR sleep mode** | **Main sleep mode** | **PLL sleep mode** | **Low-speed CR sleep mode** | **Sub sleep mode** |
| High-speed CR clock | Operating | Varies depending on the setting of the HCRE bit,MCSVE bit and FCSDE bit. | Varies depending on the setting of the PINC bit, HCRE bit, MCSVE bit and FCSDE bit. | Stopped | Stopped |
| Main clock | Varies depending on the setting of the MOSCE bit. | Operating | Varies depending on the setting of the PINC bit and MOSCE bit. | Stopped | Stopped |
| Main PLL clock | Varies depending on the setting of the PLLE bit. | Varies depending on the setting of the PLLE bit. | Operating | Stopped | Stopped |
| Low-speed CR clock | Operating | Operating | Operating | Operating | Operating |
| Sub clock | Varies depending on the setting of the SOSCE bit. | Varies depending on the setting of the SOSCE bit. | Varies depending on the setting of the SOSCE bit. | Varies depending on the setting of the SOSCE bit. | Operating |
| CPU clock | Stopped | | | | |
| AHB bus clock | High-speed CR clock | Main clock | PLL clock | Low-speed CR clock | Sub clock |
| APB0 bus clock | High-speed CR clock | Main clock | PLL clock | Low-speed CR clock | Sub clock |
| APB1 bus clock | High-speed CR clock | Main clock | PLL clock | Low-speed CR clock | Sub clock |
| | * Whether or not operation is enabled is determined depending on the setting of the APBC1EN bit. | | | | |

**Table 3-2 Clock operation states in TIMER modes**

| | TIMER modes | | | | |
|---|---|---|---|---|---|
| | **High-speed CR timer mode** | **Main timer mode** | **PLL timer mode** | **Low-speed CR timer mode** | **Sub timer mode** |
| High-speed CR clock | Operating | Varies depending on the setting of the HCRE bit, MCSVE bit and FCSDE bit. | Varies depending on the setting of the PINC bit, HCRE bit, MCSVE bit and FCSDE bit. | Stopped | Stopped |
| Main clock | Varies depending on the setting of the MOSCE bit. | Operating | Varies depending on the setting of the PINC bit and MOSCE bit. | Stopped | Stopped |
| Main PLL clock | Varies depending on the setting of the PLLE bit. | Varies depending on the setting of the PLLE bit. | Operating | Stopped | Stopped |
| Low-speed CR clock | Operating | Operating | Operating | Operating | Operating |
| Sub clock | Varies depending on the setting of the SOSCE bit. | Varies depending on the setting of the SOSCE bit. | Varies depending on the setting of the SOSCE bit. | Varies depending on the setting of the SOSCE bit. | Operating |
| CPU clock | Stopped | | | | |
| AHB bus clock | Stopped | | | | |
| APB0 bus clock | Stopped | | | | |
| APB1 bus clock | Stopped | | | | |

**Table 3-3 Clock operation state in RTC mode and STOP mode**

| | RTC mode | STOP mode |
|---|---|---|
| High-speed CR clock | Stopped | Stopped |
| Main clock | | |
| Main PLL clock | | |
| Low-speed CR clock | | |
| Sub clock | Operating | |
| CPU clock | Stopped | |
| AHB bus clock | | |
| APB0 bus clock | | |
| APB1 bus clock | | |

MOSCE : MOSCE bit of System Clock Mode Control Register (SCM_CTL)
SOSCE : SOSCE bit of System Clock Mode Control Register (SCM_CTL)
PLLE : PLLE bit of System Clock Mode Control Register (SCM_CTL)
HCRE : HCRE bit of System Clock Mode Control Register (SCM_CTL)
MCSVE : MCSVE bit of CSV Control Register (CSV_CTL)
FCSDE : FCSDE bit of CSV Control Register (CSV_CTL)
PINC : PINC bit of PLL Clock Stabilization Wait Time Setup Register (PSW_TMR)
APBC1EN : APBC1EN bit of Peripheral Bus Clock Frequency Division Register (APBC1_PSR)

*: For the SCM_CTL and APBC1_PSR Registers, refer to Chapter "Clock".

### Return factors from standby modes

The table below shows the factors by which the system returns from the SLEEP, TIMER, RTC and STOP modes.

**Table 3-4 Return factors from standby modes**

| | SLEEP mode | TIMER mode | RTC mode | STOP mode |
|---|---|---|---|---|
| Return factors by reset | - INITX pin input reset<br>- Low-voltage detection reset<br>- Software watchdog reset<br>- Hardware watchdog reset<br>- Clock failure detection reset<br>- Anomalous frequency detection reset | - INITX pin input reset<br>- Low-voltage detection reset<br>- Hardware watchdog reset<br>- Clock failure detection reset<br>- Anomalous frequency detection reset(Main Timer Mode, PLL Timer Mode) | - INITX pin input reset<br>- Low-voltage detection reset | - INITX pin input reset<br>- Low-voltage detection reset |
| Return factors by interrupt | - Effective interrupt from each peripheral | - NMI interrupt<br>- External interrupt<br>- Hardware watchdog timer interrupt<br>- Watch counter interrupt<br>- RTC interrupt<br>- HDMI-CEC/ Remote Control Reception interrupt<br>- Low voltage detection interrupt | - NMI interrupt<br>- External interruptt<br>- RTC interrupt<br>- HDMI-CEC/ Remote Control Reception interrupt<br>- Low voltage detection interrupt | - NMI interrupt<br>- External interrupt<br>- Low voltage detection interrupt |

## 3.1 Operations of SLEEP modes (high-speed CR sleep, main sleep, PLL sleep, low-speed CR sleep, and sub sleep modes)

SLEEP mode is classified as one of standby modes. Enabling SLEEP mode stops CPU clocks, reducing the power consumption.

**Functions of SLEEP mode**

- CPU and on-chip memory

In SLEEP mode, the clock supplied to the CPU is stopped. AHB bus clock continues to operate. On-chip memory keeps operating and retains the data.

- Peripherals

The APB0 bus clock is still active in SLEEP mode. The state of the APB1 bus clock varies depending on the setting of the APBC1EN bit. Peripherals are operated in the state that is set at transition.

- Watch counter and RTC

Watch counter and RTC remain unaffected by SLEEP mode. They continue to operate according to the setting before transiting to SLEEP mode.

- Oscillator clocks

Table 3-1 shows the status of each oscillator clock.

- Reset and interrupt

Reset and interrupt are available to return from SLEEP mode.

**Status of pin**

All pin settings are held in SLEEP mode.

**SLEEP mode setting procedure**

Execute the following steps to transit to SLEEP mode.

1. Set "0" to the SLEEPDEEP bit of the Cortex-M0+ System Control Register.
2. Execute the WFI or WFE instruction.
   The system transits to the appropriate SLEEP mode according to the current clock mode indicated in the RCM[2:0] bits of the System Clock Mode Status Register (SCM_STR).

For the System Clock Mode Status Register (SCM_STR), refer to Chapter "Clock".

### Return from SLEEP mode

The CPU returns from SLEEP mode in one of the following cases.

■ Return by reset

If a reset (INITX pin input reset, low-voltage detection reset, software watchdog reset, hardware watchdog reset, clock supervisor reset, or anomalous frequency detection reset) occurs, the CPU changes to high-speed CR run mode regardless of clock mode.

■ Return by interrupt

If an effective interrupt is received from a peripheral in SLEEP mode, the CPU returns from SLEEP mode and changes to RUN mode to fit clock mode indicated in the RCM[2:0] bits of the System Clock Mode Status Register (SCM_STR).

**Table 3-5 Operation modes after the CPU returned from SLEEP mode by interrupt**

| | Status of master clock before transition to SLEEP mode | | | | |
|---|---|---|---|---|---|
| | RCM=000 (High-speed CR oscillator) | RCM=001 (Main oscillator) | RCM=010 (PLL oscillator) | RCM=100 (Low-speed CR oscillator) | RCM=101 (Sub oscillator) |
| Operation modes after return by interrupt | High-speed CR run mode | Main run mode | PLL run mode | Low-speed CR run mode | Sub run mode |

RCM: RCM[2:0] bits of System Clock Mode Status Register (SCM_STR)

* For the SCM_CTL and SCM_STR Registers, refer to Chapter "Clock".

■ Waiting for oscillation stabilization at return

When the CPU returns by reset, it waits for the stabilization of high and low-speed CR clock oscillations. If the CPU returns by interrupt, it does not need to wait for oscillation to stabilize.

## 3.2 Operations of TIMER Modes (high-speed CR Timer, Main Timer, PLL Timer, low-speed CR Timer, and Sub Timer Modes)

TIMER mode is used to stop supplying a base clock. This causes the CPU clock, AHB bus clock, and all APB bus clocks to be stopped, leading to the further reduction of power consumption. In this case, all functions are stopped, excluding the oscillators, PLL, hardware watchdog timer, watch counter, RTC clock failure detector, and Low Voltage Detection Circuit.

**Functions of TIMER mode**

- CPU and on-chip memory

In TIMER mode, the CPU clock supplied to the CPU and AHB bus clock supplied to the on-chip memory and DMA controller are stopped. However, the contents of on-chip memory are retained. Also, the debug function is stopped.

- Peripherals

In TIMER mode, all APB clocks are stopped, and all resources, excluding the hardware watchdog timer, watch counter, RTC, clock supervisor, and Low Voltage Detection Circuit, are stopped in the last state.

- Watch counter and RTC

Watch counter and RTC remain unaffected by TIMER mode. They continue to operate according to the setting before transiting to TIMER mode.

- Oscillator clocks

Table 3-2 shows the status of each oscillator clock.

- Reset and interrupt

Reset and interrupt are available to return from TIMER mode.

- External bus

The external bus is stopped in TIMER mode.

- Status of pin

The system can control whether to retain the state just before the external pin changes to TIMER mode or change to high impedance status depending on the setting of the SPL bit in the Standby Mode Control Register (STB_CTL).

**TIMER mode setting procedure**

Execute the following steps to transit to TIMER mode.

1. Set "0" in RTCE bit of RTC Mode Control Register (PMD_CTL).
2. Write "0x1ACC" to the KEY bit and "0" to DSTM bit and "0b00" to the STM bit of the Standby Mode Control Register (STB_CTL). Use the SPL bit to set the status of each pin in TIMER mode.
3. Set "1" to the SLEEPDEEP bit of the Cortex-M0+ System Control Register.
4. Execute the WFI or WFE instruction.
   The system transits to the appropriate TIMER mode according to the current clock mode indicated in the RCM[2:0] bits of the System Clock Mode Status Register (SCM_STR).

***Note:***

- *DSTM bit is not in the products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1.*

### Return from TIMER mode

The CPU returns from TIMER mode in one of the following cases.

■ Return by reset

If a reset (INITX pin input reset, low-voltage detection reset, hardware watchdog reset, clock supervisor reset, or anomalous frequency detection reset (main timer mode, PLL, TIMER mode) occurs, the CPU changes to high-speed CR run mode regardless of clock mode.

Software watchdog reset is not available in this mode; therefore, the CPU cannot return by this reset.

■ Return by interrupt

If an effective NMI interrupt, external interrupt, hardware watchdog timer interrupt, watch counter interrupt, RTC interrupt, or low voltage detection interrupt request is received in TIMER mode, the CPU returns from TIMER mode and changes to RUN mode to fit clock mode indicated in the RCM[2:0] bits of SCM_STR register.

**Table 3-6 Operation modes after the CPU returned from TIMER mode by interrupt**

| | Status of master clock before transition to TIMER mode | | | | |
|---|---|---|---|---|---|
| | RCM=000 (High-speed CR oscillator) | RCM=001 (Main oscillator) | RCM=010 (PLL oscillator) | RCM=100 (Low-speed CR oscillator) | RCM=101 (Sub oscillator) |
| Operation modes after return by interrupt | High-speed CR run mode | Main run mode | PLL run mode | Low-speed CR run mode | Sub run mode |

■ Waiting for oscillation stabilization at return

When the CPU returns by reset, it waits for the stabilization of high and low-speed CR clock oscillations. If the CPU returns by interrupt, it does not need to wait for oscillation to stabilize.

■ Waiting for the stabilization of the built-in regulator voltage at return

To return from low-speed CR timer mode or sub timer mode by reset or interrupt, the voltage stabilization wait time (a few hundred μs) of the built-in regulator is ensured automatically. After the wait time has lapsed, return operations are performed.

***Notes:***

- *When an interrupt priority used for return is not set at a level to return the CPU, clock will be returned with the interrupt but the CPU remains stop state without returning. In order to do this, be sure to set the interrupt priority at a level which CPU is able to return.*
- *Before transiting to the timer mode, ensure that causes returning from timer modes in Table 3-4 are not set. If these interrupt causes are set, clear them.*
- *If the transition to TIMER mode is made during debugging, as the clock to the CPU stops, a return to RUN mode cannot be performed by the ICE. Use a return by reset or interrupt.*
- *In the case of transiting to the Low-speed CR timer mode or Sub timer mode, ensure that the flash memory automatic algorithm is terminated before executing transition.*

## 3.3 Operation of RTC Mode

RTC mode stops oscillation other than that of the sub oscillator. All the functions except for the watch counter, RTC, and low voltage detection circuit will be stopped.

**Functions of RTC mode**

- CPU and on-chip memory

In RTC mode, the CPU clock supplied to the CPU and AHB bus clock supplied to the on-chip memory and DMA controller are stopped. However, the contents of on-chip memory are retained. Also, the debug function is stopped.

- Peripheral functions

In RTC mode, all APB bus clocks are stopped, and all resources, excluding the watch counter, RTC, and Low-voltage Detection Circuit, are stopped keeping the last state.

- Watch counter and RTC

Watch counter remains unaffected by RTC mode. It continues to operate according to the setting before transiting to RTC mode and it cannot be returned from RTC mode by the Watch counter interrupt.
RTC remains unaffected by RTC mode. It continues to operate according to the setting before transiting to RTC mode

- Oscillation clocks

Table 3-3 shows the status of each oscillation clock.

- Reset and interrupt

Reset and interrupt can be used to return from RTC mode.

- External bus

The external bus is stopped in RTC mode.

- Status of pin

The system can control whether to retain the status just before the external pin changes to RTC mode or changes to the high impedance status depending on the setting of the SPL bit in the Standby Mode Control Register (STB_CTL).

**RTC mode setting procedure**

Execute the following steps to transit to RTC mode.

1. Set "1" in RTCE bit of RTC mode control register (PMD_CTL) while SORDY bit of System Clock Mode Status Register (SCM_STR) is "1".
2. Write "0x1ACC" to the KEY bit, "0" to the DSTM bit, and "0b10" to the STM bit of the Standby Mode Control Register (STB_CTL). Use the SPL bit to set the status of each pin in RTC mode.
3. Set "1" to the SLEEPDEEP bit of the Cortex-M0+ System Control Register.
4. Execute the WFI or WFE instruction.

***Note:***

- *DSTM bit is not in the products that do not equip deep standby RTC mode, and deep standby stop mode. See Table 1-1.*

### Return from RTC mode

The CPU returns from RTC mode in any one of the following cases.

■ Return by reset

If a reset (INITX pin input reset, low-voltage detection reset) occurs, the CPU changes to high-speed CR run mode regardless of clock mode.
Software watchdog reset, hardware watchdog reset, clock supervisor reset, and anomalous frequency detection reset do not operate; therefore, the CPU cannot return by those resets.

■ Return by interrupt

If an effective NMI interrupt, external interrupt, RTC interrupt, or low voltage detection interrupt request is received in RTC mode, the CPU returns from RTC mode and transits to RUN mode to fit clock mode indicated in the RCM[2:0] bits of SCM_STR register.

**Table 3-7 Operation modes after the CPU has returned from RTC mode by interrupt.**

| | Status of master clock before transition to RTC mode | | | | |
|---|---|---|---|---|---|
| | **RCM = 000 (High-speed CR oscillator)** | **RCM = 001 (Main oscillator)** | **RCM = 010 (PLL oscillator)** | **RCM = 100 (Low-speed CR oscillator)** | **RCM = 101 (Sub oscillator)** |
| Operation modes after return by interrupt | High-speed CR run mode | Main run mode | PLL run mode | Low-speed CR run mode | Sub run mode |

■ Waiting for oscillation stabilization at return

When the CPU returns by reset, it waits for the stabilization of high and low-speed CR clock oscillations. If the CPU returns by interrupt, the oscillation stabilization wait changes by the master clock before transition to the RTC mode as shown in Table 3-8.

**Table 3-8 Oscillation stabilization wait when returning by interrupt from RTC mode**

| | | Status of master clock before transition to RTC mode | | | | |
|---|---|---|---|---|---|---|
| | | **RCM = 000 (High-speed CR oscillator)** | **RCM = 001 (Main oscillator)** | **RCM = 010 (PLL oscillator)** | **RCM = 100 (Low-speed CR oscillator)** | **RCM = 101 (Sub oscillator)** |
| Oscillation stabilization wait after returning by interrupt | High-speed CR clock | ON | HCR="enable": ON*1<br>Others: OFF | PINC="1" or HCR="enable": ON*1<br>Others: OFF | OFF | OFF |
| | Main clock | MOSCE="0": OFF<br>MOSCE="1": ON | ON | PINC="0" or MOSCE="1": ON<br>Others: OFF | OFF | OFF |
| | Main PLL clock | PLLE="0": OFF<br>PLLE="1": ON | PLLE="0": OFF<br>PLLE="1": ON | ON | OFF | OFF |
| | Low-speed CR clock | ON | ON | ON | ON | ON |
| | Sub clock | OFF*2 | OFF*2 | OFF*2 | OFF*2 | OFF*2 |

*1 : HCR is defined as enabled when HCRE ="1" or MCSVE="1" or FCSDE="1".

*2 : TYPE1 product has the oscillation stabilization wait time, but the actual oscillation continues. CSW_TMR:SOWT bit is possible to set minimum wait time "1100".

■ Waiting for the stabilization of the built-in regulator voltage at return

To return from RTC mode, the voltage stabilization wait time (a few hundred μs) of the built-in regulator is ensured automatically. After the wait time has lapsed, return operations are performed.

***Notes:***

- *When an interrupt priority used for return is not set at a level to return the CPU, the clock will be returned by the interrupt but the CPU remains in stop state without returning. In order to do this, be sure to set the interrupt priority at a level which the CPU is able to return.*
- *Before transiting to the RTC mode, ensure that the causes of return from timer mode in Table 3-4 are not set. If these return causes are set, clear them.*
- *If the transition to RTC mode is made during debugging, as the clock to the CPU stops, a return to RUN mode cannot be performed by the ICE. Use a return by reset or interrupt.*
- *In case of transiting to the RTC mode, ensure that the flash memory automatic algorithm is terminated before executing the transition.*

## 3.4 Operations of STOP mode

STOP mode is used to stop all oscillating operations. Enabling this mode stops all functions, excluding the Low Voltage Detection Circuit.

### Functions of STOP mode

- CPU and on-chip memory

In STOP mode, the CPU clock supplied to the CPU and AHB bus clock supplied to the on-chip memory and DMA controller are stopped. However, the contents of on-chip memory are retained. The debug function is stopped.

- Peripheral functions

All APB bus clocks are stopped, and all resources, excluding the Low Voltage Detection Circuit, are stopped in the last state.

- Oscillator clocks

All oscillator clocks are stopped.

- Reset and interrupt

Reset and interrupt are available to return from STOP mode.

- External bus

The external bus is stopped in STOP mode.

- Status of pin

The system can control whether to retain the state just before the external pin changes to STOP mode or change to high impedance status depending on the setting of the SPL bit in the Standby Mode Control Register (STB_CTL).

### STOP mode setting procedure

Execute the following steps to transit to STOP mode.

1. Set "0" in RTCE bit of RTC Mode Control Register (PMD_CTL).
2. Write "0x1ACC" to the KEY bit and "0" to DSTM bit and "0b10" to the STM bit of the Standby Mode Control Register (STB_CTL). Use the SPL bit to set the status of each pin in STOP mode.
3. Set "1" to the SLEEPDEEP bit of the Cortex-M0+ System Control Register.
4. Execute the WFI or WFE instruction.

***Note:***

- *DSTM bit is not in the products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1.*

### Return from STOP mode

The CPU returns from STOP mode in one of the following cases.

■ Return by reset

If a reset (INITX pin input reset or low-voltage detection reset) occurs, the CPU changes to the high-speed CR run mode regardless of clock mode.

Software watchdog reset, hardware watch dog reset, clock supervisor reset, and anomalous frequency detection reset are not available in this mode; therefore, the CPU cannot return by those resets.

■ Return by interrupt

If an effective NMI interrupt, external interrupt, or low voltage detection interrupt request is received in STOP mode, the CPU returns from STOP mode and changes to RUN mode to fit clock mode indicated in the RCM[2:0] bits of SCM_STR register.

**Table 3-9 Operation modes after the CPU returned from the STOP mode by interrupt**

| | Status of master clock before changing to STOP mode | | | | |
|---|---|---|---|---|---|
| | RCM=000 (High-speed CR oscillator) | RCM=001 (Main oscillator) | RCM=010 (PLL oscillator) | RCM=100 (Low-speed CR oscillator) | RCM=101 (Sub oscillator) |
| Operation modes after return by interrupt | High-speed CR run mode | Main run mode | PLL run mode | Low-speed CR run mode | Sub run mode |

■ Waiting for oscillation stabilization at return

When the CPU returns by reset, it waits for the stabilization of high and low-speed CR clock oscillations. If the CPU returns by interrupt, the oscillation stabilization wait state varies depending on the master clock that is output before the CPU changes to STOP mode as shown in Table 3-10.

**Table 3-10 Waiting for oscillation to stabilize at return from STOP mode by interrupt**

| | | Status of master clock before changing to STOP mode | | | | |
|---|---|---|---|---|---|---|
| | | RCM=000 (High-speed CR oscillator) | RCM=001 (Main oscillator) | RCM=010 (PLL oscillator) | RCM=100 (Low-speed CR oscillator) | RCM=101 (Sub oscillator) |
| Oscillation stabilization waiting after return by interrupt | High-speed CR clock | ON | HCR="enable": ON*<br>Others: OFF | PINC="1" or HCR="enable": ON*<br>Others: OFF | OFF | OFF |
| | Main clock | MOSCE="0": OFF<br>MOSCE="1": ON | ON | PINC="0" or MOSCE="1": ON<br>Others: OFF | OFF | OFF |
| | Main PLL clock | PLLE="0": OFF<br>PLLE="1": ON | PLLE="0": OFF<br>PLLE="1": ON | ON | OFF | OFF |
| | Low-speed CR clock | ON | ON | ON | ON | ON |
| | Sub clock | SOSCE="0": OFF<br>SOSCE="1": ON | SOSCE="0": OFF<br>SOSCE="1": ON | SOSCE="0": OFF<br>SOSCE="1": ON | SOSCE="0": OFF<br>SOSCE="1": ON | ON |

* : HCR is defined as enabled when HCRE ="1" or MCSVE="1" or FCSDE="1".

■ Waiting for the stabilization of the built-in regulator voltage at return

When the CPU returns from STOP mode, the voltage stabilization wait time (a few hundred μs) of the built-in regulator is ensured automatically. After the wait time has lapsed, return operations are performed.

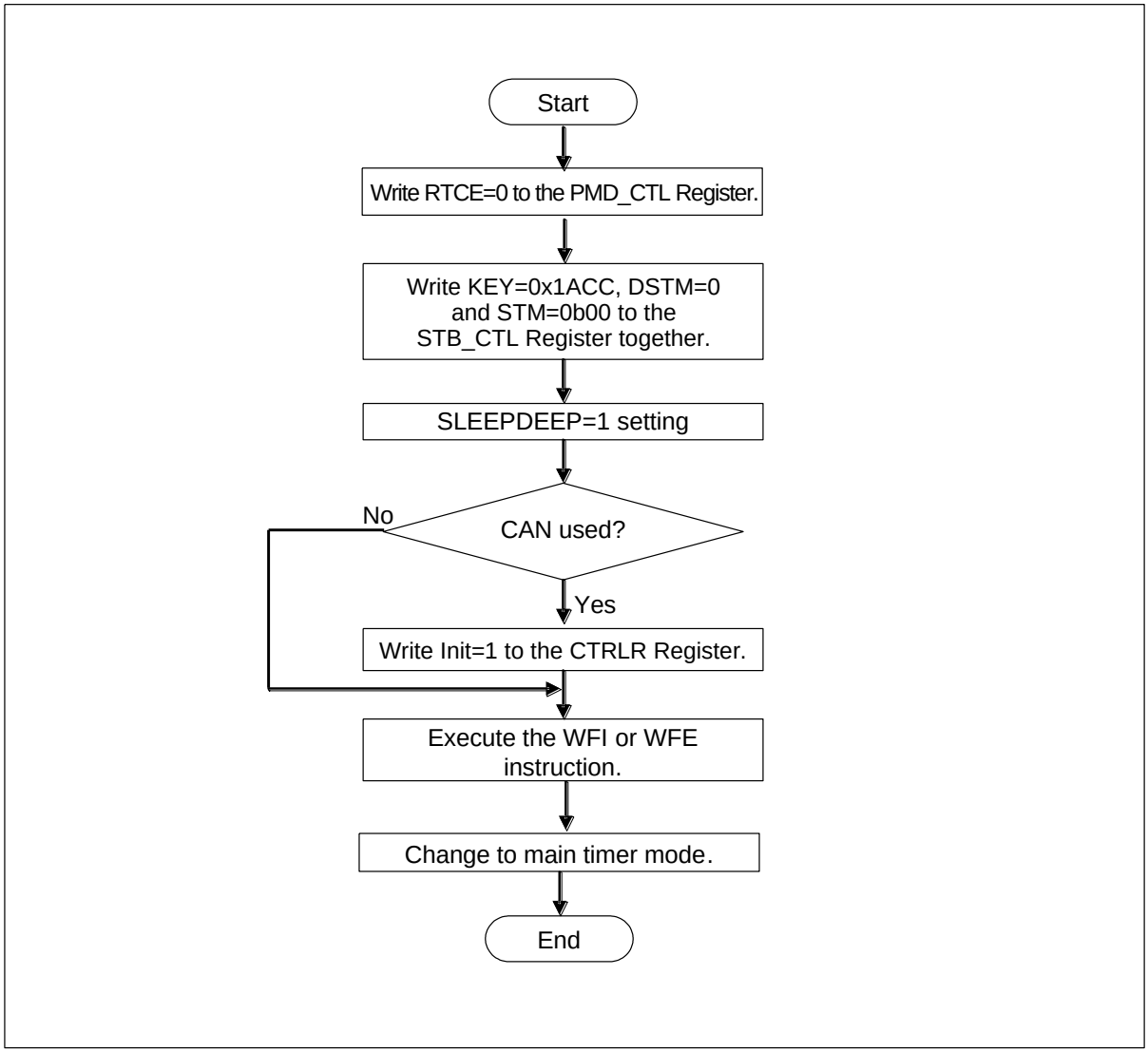
***Notes:***

- *When an interrupt priority used for return is not set at a level to return the CPU, clock will be returned with the interrupt but the CPU remains stop state without returning. In order to do this, be sure to set the interrupt priority at a level which CPU is able to return.*
- *Before transiting to the stop mode, ensure causes of return timer modes in Table 3-4 are not set. If these interrupt causes are set, clear them.*
- *If the transition to STOP mode is made during debugging, as the clock to the CPU stops, a return to RUN mode cannot be performed by the ICE. Use a return by reset or interrupt.*
- *In case of transiting to the STOP mode, ensure that the flash memory automatic algorithm is terminated before executing transition.*

# 4. Standby Mode Setting Procedure Examples

This section provides standby mode setting procedure examples.

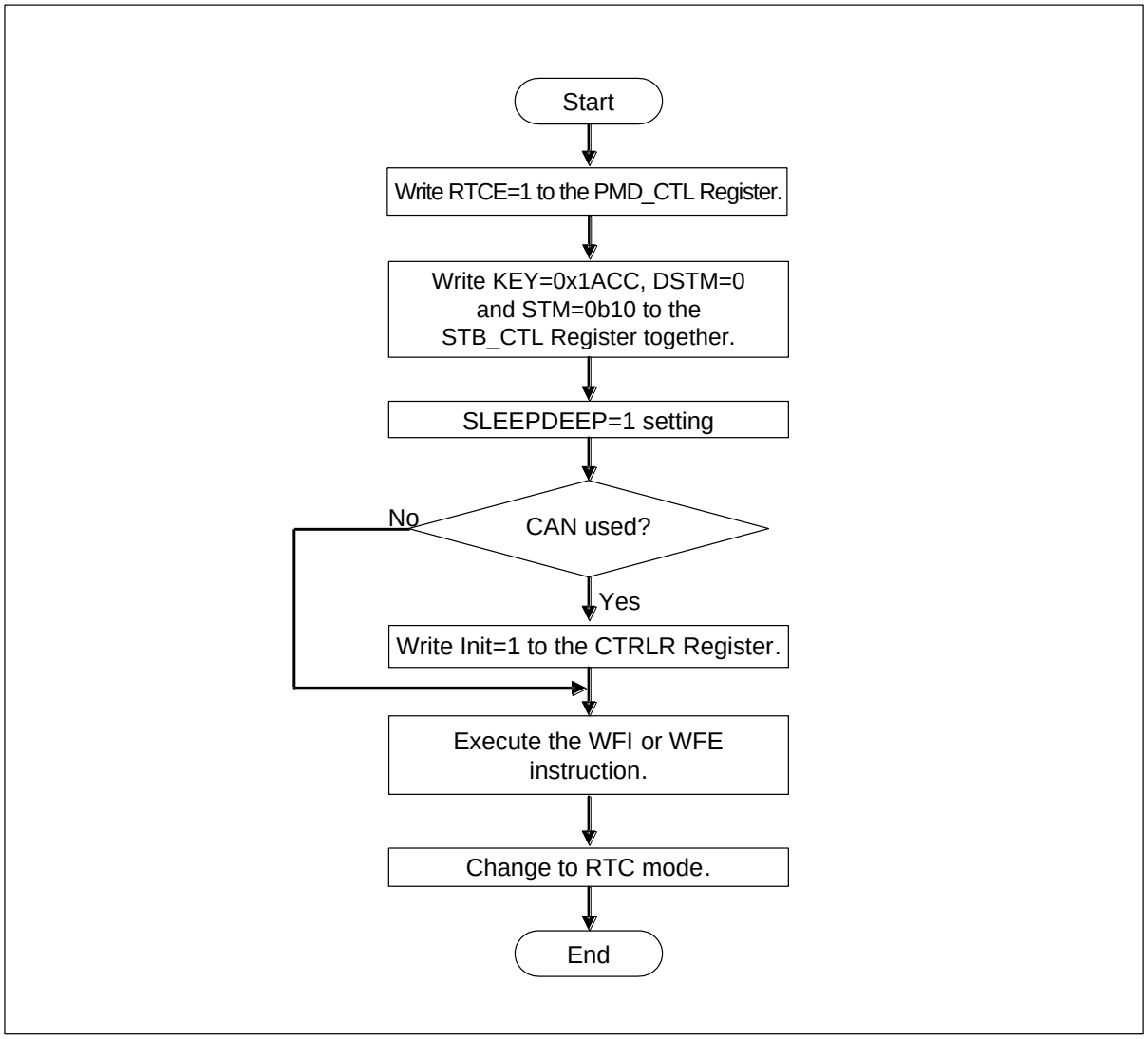
**Figure 4-1 Main timer mode setting procedure example**




***Note:***

- *DSTM bit is not in the products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1.*

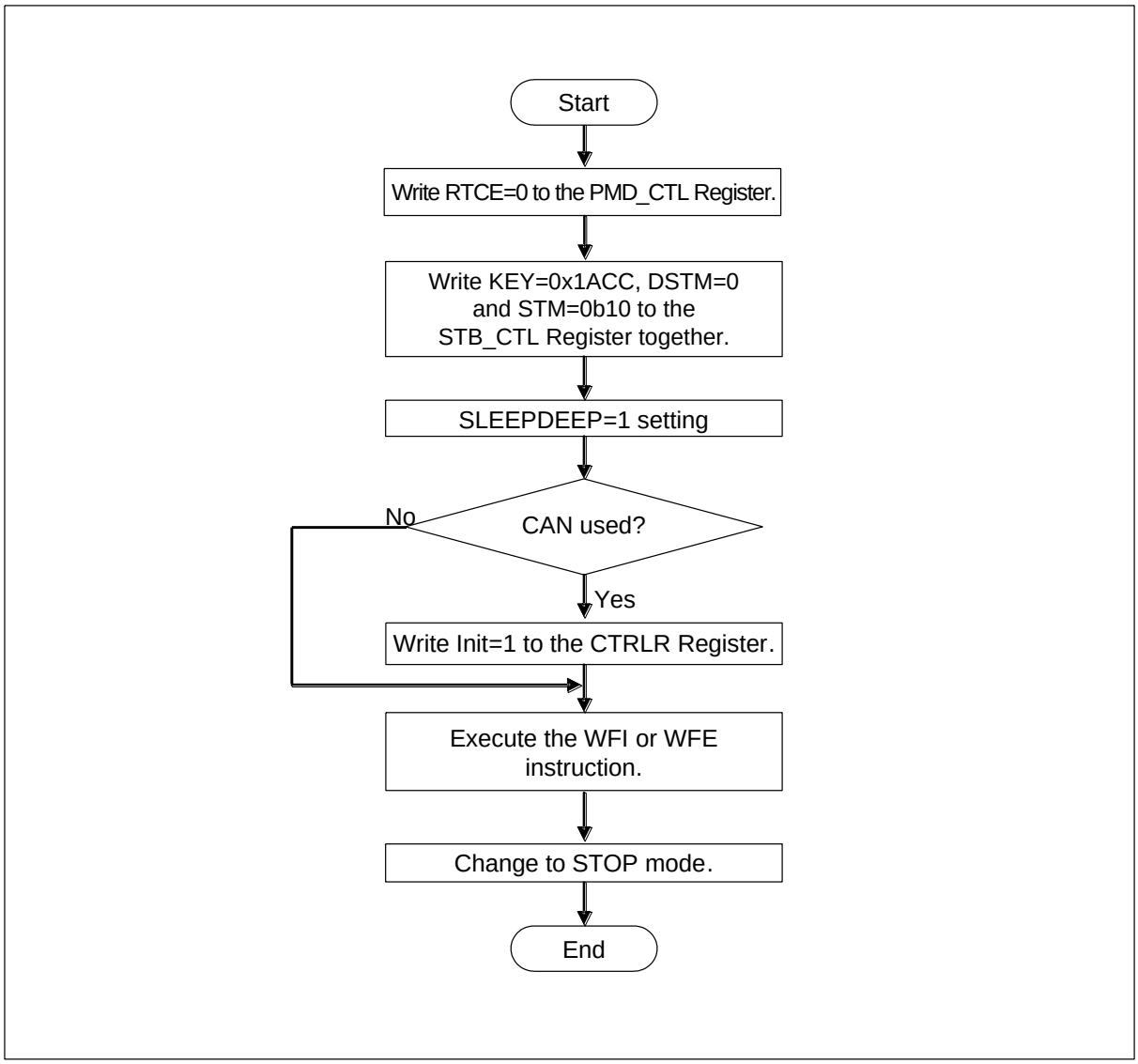
**Figure 4-2 RTC mode setting procedure example (Main clock is selected as a master clock)**




***Notes:***

- *In case of transiting to the RTC mode, ensure that the flash memory automatic algorithm is terminated before executing the transition.*
- *Writing "1" to RTCE bit of the RTC Mode Control Register (PMD_CTL) is valid only when the SORDY bit of the System Clock Mode Status Register (SCM_STR) is "1".*
- *DSTM bit is not in the products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1.*

**Figure 4-3 STOP mode setting procedure example (Main clock is selected as a master clock)**




***Notes:***

- *In case of transiting to the STOP mode, ensure that the flash memory automatic algorithm is terminated before executing the transition.*
- *DSTM bit is not in the products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1.*

# 5. Description of Deep Standby Mode Operation

This section describes the operation of deep standby mode.
Deep standby mode includes deep standby RTC mode and deep standby STOP mode.

### Clock operation status in deep standby mode

The following shows the status of the oscillation clock, CPU clock, AHB bus clock, and APB bus clock while in deep standby RTC mode and deep standby STOP mode.

**Table 5-1 Clock operation state in deep standby mode**

| | Deep standby RTC mode | Deep standby STOP mode |
|---|---|---|
| High-speed CR clock | Stopped | Stopped |
| Main clock | | |
| Main PLL clock | | |
| Low-speed CR clock | | |
| Sub clock | Operating | |
| CPU clock | Stopped | |
| AHB bus clock | | |
| APB0 bus clock | | |
| APB1 bus clock | | |

### Return factors from deep standby mode

The following shows the return factors from deep standby RTC mode and deep standby STOP mode.

**Table 5-2 Return factors from deep standby mode**

| | Deep standby RTC mode | Deep standby STOP mode |
|---|---|---|
| Deep standby return factor | – INITX pin input reset<br>– Low-voltage detection reset | – INITX pin input reset<br>– Low-voltage detection reset |
| | – Low-voltage detection interrupt<br>– RTC interrupt<br>– HDMI-CEC/ Remote Control Reception interrupt | – Low-voltage detection interrupt |
| | – WKUP pin input | – WKUP pin input |

***Note:***

- *Although each interrupt factor is retained after returning from deep standby mode, interrupt processing will not be executed since NVIC is initialized by deep standby transition reset.*

### Internal power supply status and reset status in deep standby mode

The following shows the power supply status of each function in deep standby mode and initialization status in deep standbytransition reset.

**Table 5-3 Internal power supply status and initialization status in deep standby mode**

| | Power supply status | Reset status |
|---|---|---|
| CPU | Off | Initialize |
| On-chip Flash | Off | *1 |
| On-chip SRAM | Off *2 | *3 |
| RTC | On | Do not initialize |
| HDMI-CEC/ Remote Control Reception | On | Do not initialize |
| Low-voltage detection circuit | On | Do not initialize |
| GPIO | On | Partly initialize *4 |
| Deep standby control block | On | Do not initialize |
| Peripheral functions other than the above | Off | Initialize |

*1: The contents of on-chip Flash memoryare retained.

*2: The contents of on-chip SRAM can be retained.
In the setting to retain on-chip SRAM data, the power of on-chip SRAM is ON.

*3: The contents of on-chip SRAM are not retained when the power is OFF.
In the setting to retain on-chip SRAM data, on-chip SRAM data is retained.

*4: PFRx registers excluding bit4:0 of PFR0 are initialized and others are not initialized.

## 5.1 Operation of Deep Standby RTC Mode

Deep standby RTC mode stops oscillation other than that of the sub oscillator. All the functions except for RTC, HDMI-CEC/Remote Control Reception and low voltage detection circuit will be stopped. It turns off RTC, HDMI-CEC/Remote Control Reception, the low-voltage detection circuit, CPUs excluding GPIO, on-chip Flash memory, on-chip SRAM*, and peripheral functions, inside the chip.

### Functions of deep standby RTC mode

■ CPU and on-chip memory

In deep standby RTC mode, the CPU clock supplied to CPU and AHB bus clock supplied to on-chip memory and DMA controller are stopped. It turns off the CPU, on-chip Flash memory, and on-chip SRAM*. The contents of the CPU register and on-chip SRAM are not retained*. The contents of on-chip flash memory are retained. Also, the debug function is stopped and turned off.

*: Data in on-chip SRAM can be retained.
In the setting to retain on-chip SRAM data, the power of on-chip SRAM is ON.

■ Peripheral functions

All APB bus clocks are stopped, and RTC, HDMI-CEC/ Remote Control Reception, Low-voltage Detection Circuit, and all resources excluding GPIO, are turned off.

■ RTC, HDMI-CEC/ Remote Control Reception

RTC and HDMI-CEC/ Remote Control Reception remains unaffected by deep standby RTC mode. It continues to operate according to the setting before transiting to deep standby RTC mode.

■ Oscillation clock

The status of each oscillation clock is shown in Table 5-1.

■ Reset, interrupt, and WKUP pin input

Reset, interrupt, and WKUP pin input can be used for returning from deep standby RTC mode.

■ Status of pin

In deep standby RTC mode, the system can control whether the external pin switches to GPIO or to high impedance status by the SPL bit in the Standby Mode Control Register (STB_CTL).

### Setting procedure of deep standby RTC mode

Execute the following steps to transit to deep standby RTC mode.

1. Set "1" in RTCE bit of the RTC Mode Control Register (PMD_CTL) while the SORDY bit of the System Clock Mode Status Register (SCM_STR) is "1".
2. Write "0x1ACC" to the KEY bit, "1" to the DSTM bit, and "0b10" to the STM bit of the Standby Mode Control Register (STB_CTL). Use the SPL bit to set the status of each pin in deep standby RTC mode.
3. Set "1" to the SLEEPDEEP bit of the Cortex-M0+ System Control Register.
4. Execute the WFI or WFE instruction.

***Note:***

- *DSTM bit is not in the products that do not equip deep standby RTC mode, and deep standby stop mode. See Table 1-1.*

### Return from deep standby RTC mode

CPU returns from deep standby RTC mode in any one of the following cases.

- Return by reset, interrupt, and WKUP pin input

If a reset (INITX pin input reset and low-voltage detection reset) occurs or if effective RTC interrupt, HDMI-CEC/ Remote Control Reception interrupt, low-voltage detection interrupt, and WKUP pin input request are received while in deep standby RTC mode, the CPU returns from deep standby RTC mode and changes to high-speed CR run mode regardless of clock mode by deep standby transition reset occurrence.

Software watchdog reset, hardware watchdog reset, clock supervisor reset, and anomalous frequency detection reset do not operate; therefore, the CPU cannot return by those resets.

- Waiting for oscillation stabilization at return

The oscillation stabilization wait for the high-speed CR clock and low-speed CR clock is executed regardless of return factor.

- Waiting for the stabilization of the built-in regulator voltage at return

To return from deep standby RTC mode, the voltage stabilization wait time (a few hundred μs) of the built-in regulator is ensured automatically. After the wait time has lapsed, return operations are performed.

***Notes:***

- *Before transiting to the deep standby RTC mode, ensure that the return factor from deep standby RTC mode in Table 5-2 is not set. If the factor is set, clear it.*
- *If the transition to deep standby RTC mode is made during debugging, a return to RUN mode cannot be performed by the ICE as the debug function is turned off. Use a return by reset, interrupt, or WKUP pin input.*
- *In the case of transiting to the deep standby RTC mode, ensure that the flash memory automatic algorithm is terminated before executing the transition.*

## 5.2 Operation of Deep Standby Stop Mode

Deep standby stop mode stops all oscillations. All the functions except for the low voltage detection circuit will be stopped. It turns off, RTC, HDMI-CEC/Remote Control Reception, the low-voltage detection circuit, CPUs excluding GPIO, on-chip Flash memory, on-chip SRAM*, and peripheral functions, inside the chip.

**Functions of deep standby STOP mode**

■ CPU and on-chip memory

In deep standby STOP mode, the CPU clock supplied to CPU and AHB bus clock supplied to on-chip memory and DMA controller are stopped. It turns off the CPU and on-chip Flash, on-chip SRAM*. The contents of the CPU register and on-chip SRAM are not retained*. The contents of on-chip flash memory are retained. Also, the debug function is stopped and turned off.

*: Data in on-chip SRAM can be retained.
In the setting to retain on-chip SRAM data, the power of on-chip SRAM is ON.

■ Peripherals

All APB bus clocks are stopped, and RTC, HDMI-CEC/ Remote Control Reception, Low-voltage Detection Circuit, and all resources excluding GPIO, are turned off.

■ Oscillation clock

All oscillations are stopped.

■ Reset and WKUP pin input

Reset and WKUP pin input can be used for returning from deep standby STOP mode.

■ Status of pin

The system can control whether the external pin switches to GPIO in deep standby STOP mode or to high impedance status by the SPL bit in the Standby Mode Control Register (STB_CTL).

**Setting procedure of deep standby STOP mode**

Execute the following steps to transit to deep standby STOP mode.

1. Set "0" in RTCE bit of the RTC Mode Control Register (PMD_CTL).
2. Write "0x1ACC" to the KEY bit, "1" to the DSTM bit, and "0b10" to the STM bit of the Standby Mode Control Register (STB_CTL). Use the SPL bit to set the status of each pin in deep standby STOP mode.
3. Set "1" to the SLEEPDEEP bit of the Cortex-M0+ System Control Register.
4. Execute the WFI or WFE instruction.

***Note:***

- *DSTM bit is not in the products that do not equip deep standby RTC mode, and deep standby STOP mode. See Table 1-1.*

### Return from deep standby STOP mode

CPU returns from deep standby STOP mode in any one of the following cases.

- Return by reset, interrupt, and WKUP pin input

If a reset (INITX pin input reset and low-voltage detection reset) occurs, or if effective low-voltage detection interrupt, or WKUP pin input request is received while in deep standby STOP mode, the CPU returns from deep standby STOP mode and changes to high-speed CR run mode regardless of clock mode by deep standby transition reset.

Software watchdog reset, hardware watchdog reset, clock supervisor reset, and anomalous frequency detection reset do not operate; therefore, the CPU cannot return by those resets.

- Waiting for oscillation stabilization at return

The oscillation stabilization wait for the high-speed CR clock and low-speed CR clock is executed regardless of return factor.

- Waiting for the stabilization of the built-in regulator voltage at return

To return from deep standby STOP mode, the voltage stabilization wait time (a few hundred μs) of the built-in regulator is ensured automatically. After the wait time has lapsed, return operations are performed.

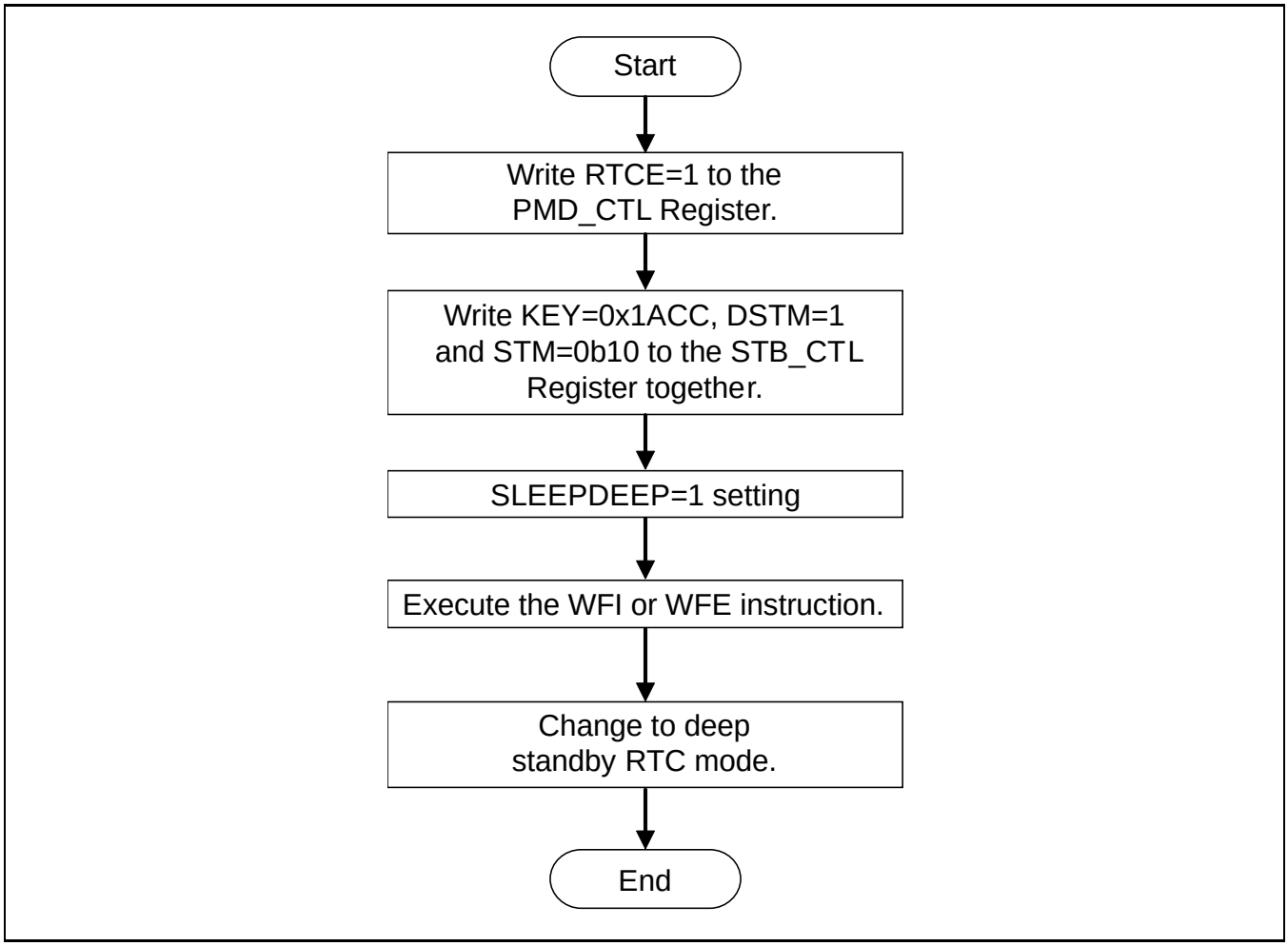
***Notes:***

- *Before transiting to the deep standby RTC mode, ensure that the return factor from deep standby RTC mode in Table 5-2 is not set. If the factor is set, clear it.*
- *If the transition to deep standby stop mode is made during debugging, a return to RUN mode cannot be performed by the ICE as the debug function turns off. Use a return by reset, interrupt, or WKUP pin input.*
- *In the case of transiting to the deep standby STOP mode, ensure that the flash memory automatic algorithm is terminated before executing the transition.*

# 6. Deep Standby Mode Setting Procedure Examples

This section explains the deep standby mode setting procedure examples.

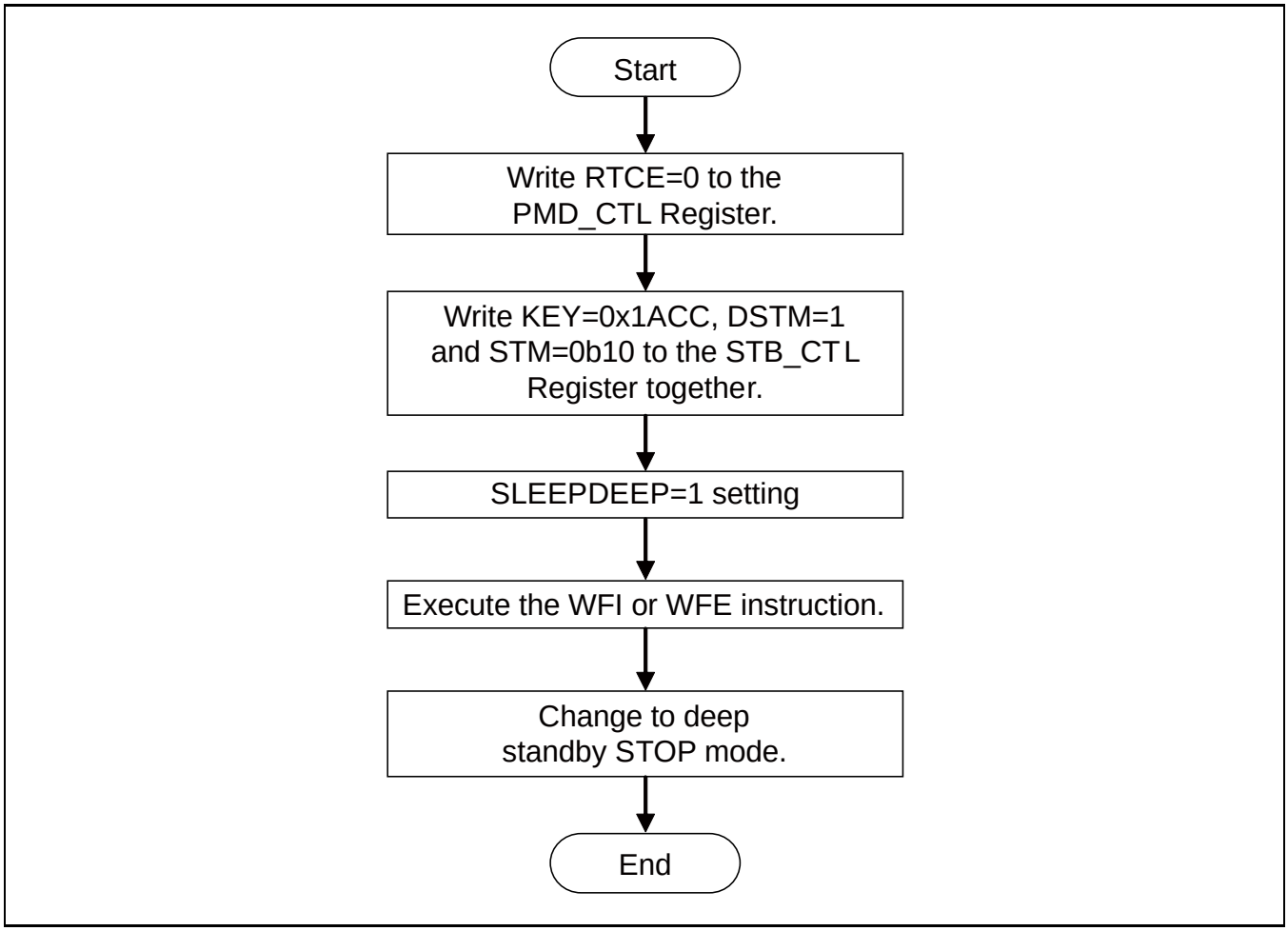
**Figure 6-1 Setting procedure example for deep standby RTC mode**




***Notes:***

- *In case of transiting to the deep standby RTC mode, ensure that the flash memory automatic algorithm is terminated before executing the transition.*
- *Writing "1" to RTCE bit of the RTC Mode Control Register (PMD_CTL) is valid only when the SORDY bit of the System Clock Mode Status Register (SCM_STR) is "1".*

**Figure 6-2 Setting procedure example for deep standby STOP mode**



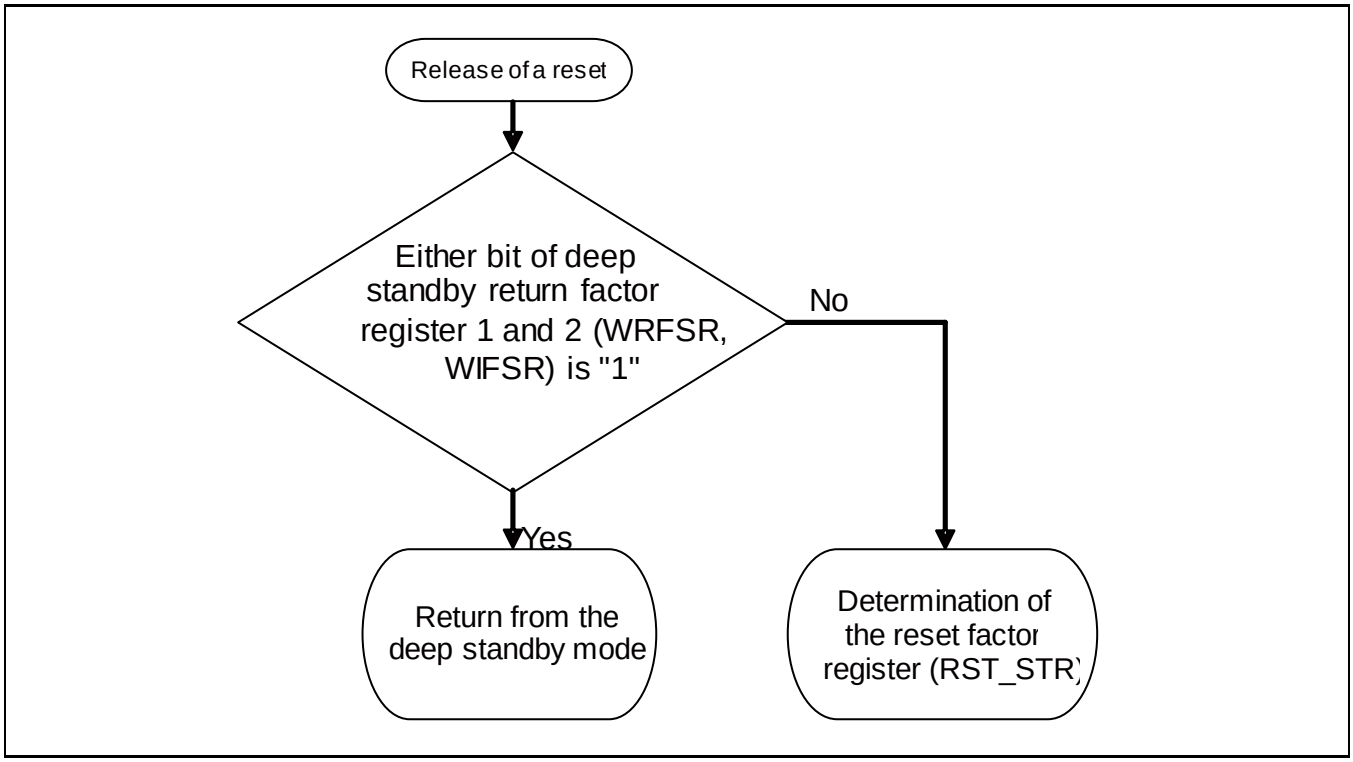

***Note:***

- *In case of transiting to the deep standby STOP mode, ensure that the flash memory automatic algorithm is terminated before executing the transition.*

# 7. Deep Standby Return Factor Determination Procedure

Figure 7-1 shows a procedure example to determine a return from deep standby mode.

**Figure 7-1 Procedure example for deep standby return factor determination**




***Note:***

- *At the transition to deep standby mode, the power supply of the CPU is turned off after deep standby transition reset. Therefore, the value of the reset factor register (RST_STR) is invalid when returning from deep standby mode.*

# 8. Registers

This section describes the registers used in low power consumption mode.

### List of low power consumption mode register

**Table 8-1 List of low power consumption mode register**

| Abbreviation | Register name | Reference |
|---|---|---|
| STB_CTL | Standby Mode Control Register | 8.1 |

■ List of registers of deep standby control block

**Table 8-2 List of registers of deep standby control block**

| Abbreviation | Register name | Reference |
|---|---|---|
| REG_CTL | Sub Oscillation Circuit Power Supply Control Register | 8.2 |
| RCK_CTL | Sub Clock Control Register | 8.3 |
| PMD_CTL | RTC Mode Control Register | 8.4 |
| WRFSR | Deep Standby Return Factor Register 1 | 8.5 |
| WIFSR | Deep Standby Return Factor Register 2 | 8.6 |
| WIER | Deep Standby Return Enable Register | 8.7 |
| WILVR | WKUP Pin Input Level Register | 8.8 |
| DSRAMR | Deep Standby RAM Retention Register | 8.9 |
| BUR01 to 16 | Backup Registers 01 to 16 | 8.10 |

***Note:***

- *For the System Clock Mode Control Register (SCM_CTL), refer to Chapter "Clock".*
  *Registers of the deep standby control block are not turned off in deep standby mode.*

## 8.1 Standby Mode Control Register (STB_CTL)

The Standby Mode Control Register (STB_CTL) controls standby mode and deep standby mode. The value written to the SPL, DSTM or STM bit is effective only when 0x1ACC is simultaneously written to the KEY bit.

### Register configuration

| bit | 31 | 16 | 15 | 8 |
|---|---|---|---|---|
| Field | KEY | | Reserved | |
| Attribute | R/W | | - | |
| Initial value | 0x0000 | | 0x00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | SPL | Reserved | DSTM | STM | |
| Attribute | - | | | R/W | - | R/W | R/W | |
| Initial value | 000 | | | 0 | 0 | 0 | 00 | |

### Register functions

**[bit31:16] KEY: Standby mode control write control bits**

These bits release the SPL bit, DSTM bit or STM bit writing control.

- The value written to the SPL bit, DSTM bit or STM bit is effective only when 0x1ACC is written to the KEY bit.
- If a value other than 0x1ACC is written to the KEY bit, the value written to the SPL bit, DSTM bit or STM bit is not effective.
- 0x0000 is always read in read mode.

**[bit15:5] Reserved: Reserved bits**

The read value is always "0". Writing has no effect on the operation.

**[bit4] SPL: Standby pin level setting bit**

This bit sets the status of pin in TIMER mode, RTC mode, STOP mode, Deep standby RTC mode, or Deep standby STOP mode.

| bit | Description |
|---|---|
| 0 | Retains status of each pin in TIMER mode, RTC mode, and STOP mode and switches to GPIO in deep standby RTC mode and deep standby stop mode. [Initial value] |
| 1 | Sets the status of each pin to high impedance in TIMER mode, RTC mode, STOP mode, Deep standby RTC mode, or Deep standby STOP mode. |

**[bit3] Reserved: Reserved bit**

The read value is always "0". Writing has no effect on the operation.

**[bit2] DSTM: Deep standby mode select bit**

This bit selects transiting to either standby mode or deep standby mode.

**[bit1:0] STM: Standby mode select bits**

These bits are a combination of DSTM bit and RTCE bit of the RTC Mode Control Register (PMD_CTL) and select transiting to any one of the following: TIMER mode, RTC mode, STOP mode, deep standby RTC mode, and deep standby STOP mode.

| DSTM | STM | | PMD_CTL: RTCE | Description |
|---|---|---|---|---|
| | bit1 | bit0 | | |
| 0 | 0 | 0 | 0 | TIMER mode [initial value] |
| 0 | 0 | 0 | 1 | Setting is prohibited. |
| 0 | 0 | 1 | 0 | Setting is prohibited |
| 0 | 0 | 1 | 1 | Setting is prohibited |
| 0 | 1 | 0 | 0 | STOP mode |
| 0 | 1 | 0 | 1 | RTC mode |
| 0 | 1 | 1 | 0 | Setting is prohibited |
| 0 | 1 | 1 | 1 | Setting is prohibited |
| 1 | 0 | 0 | 0 | Setting is prohibited |
| 1 | 0 | 0 | 1 | Setting is prohibited |
| 1 | 0 | 1 | 0 | Setting is prohibited |
| 1 | 0 | 1 | 1 | Setting is prohibited |
| 1 | 1 | 0 | 0 | Deep standby STOP mode |
| 1 | 1 | 0 | 1 | Deep standby RTC mode |
| 1 | 1 | 1 | 0 | Setting is prohibited |
| 1 | 1 | 1 | 1 | Setting is prohibited |

***Notes:***

- *The written value to SPL bit, DSTM bit, STM bit in the Standby Mode Control Register (STB_CTL) is valid only when "0x1ACC" is written to KEY bit at the same time. If a value other than "0x1ACC" is written to KEY bit, writing to SPL bit, DSTM bit, and STM bit becomes invalid.*

## 8.2 Sub Oscillation Circuit Power Supply Control Register (REG_CTL)

The Sub Oscillation Circuit Power Supply Control Register (REG_CTL) controls the power supply for sub oscillation circuit.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | ISUBSEL | | Reserved |
| Attribute | - | | | | | R/W | | - |
| Initial value | 00000 | | | | | 10 | | 0 |

### Register functions

**[bit7:3] Reserved: Reserved bits**

The read value is always "0". Writing has no effect on the operation.

**[bit2:1] ISUBSEL: Sub oscillation circuit current setting bits**

These bits set the current to sub oscillation circuit.

| bit1 | bit0 | Description |
|---|---|---|
| 0 | 0 | Setting is prohibited |
| 0 | 1 | Setting is prohibited |
| 1 | 0 | 360nA [initial value] |
| 1 | 1 | Setting is prohibited |

**[bit0] Reserved: Reserved bit**

The read value is always "0". Writing has no effect on the operation.

***Notes:***

- *This register is not initialized by software reset or deep standby transition reset.*

## 8.3 Sub Clock Control Register (RCK_CTL)

The Sub Clock Control Register (RCK_CTL) controls the clock to RTC, HDMI-CEC/remote control reception. Power consumption can be reduced by stopping the clock supply to unused resource.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | CECCKE | RTCCKE |
| Attribute | - | | | | | | R/W | R/W |
| Initial value | 000000 | | | | | | 0 | 1 |

### Register functions

**[bit7:2] Reserved: Reserved bits**

The read value is always "0". Writing has no effect on the operation.

**[bit1] CECCKE : CEC clock control bit**

This bit controls sub clock for HDMI-CEC/remote control reception macro.

| bit | Description |
|---|---|
| 0 | Sub clock is not supplied to HDMI-CEC/remote control reception macro. [Initial value] |
| 1 | Sub clock is supplied to HDMI-CEC/remote control reception macro. |

**[bit0] RTCCKE : RTC clock control bit**

This bit controls sub clock for RTC macro.

| bit | Description |
|---|---|
| 0 | Sub clock is not supplied to RTC macro. |
| 1 | Sub clock is supplied to RTC macro. [Initial value] |

## 8.4 RTC Mode Control Register (PMD_CTL)

The RTC Mode Control Register (PMD_CTL) controls either RTC mode or STOP mode and either deep standby RTC mode or deep standby STOP mode.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | RTCE |
| Attribute | - | | | | | | | R/W |
| Initial value | 0000000 | | | | | | | 0 |

### Register functions

#### [bit7:1] Reserved: Reserved bits

The read value is always "0". Writing has no effect on the operation.

#### [bit0] RTCE: RTC mode control bit

This bit selects transiting to either RTC mode or STOP mode and either deep standby RTC mode or deep standby stop mode.

| bit | Description |
|---|---|
| 0 | STOP mode and deep standby stop mode [initial value] |
| 1 | RTC mode and deep standby RTC mode |

Standby mode is selected when DSTM bit is "0" and deep standby mode is selected when DSTM bit is "1".

***Notes:***

- *This register is not initialized by software reset and deep standby transition reset.*
- *Writing "1" to RTCE bit is valid only when the SORDY bit of the System Clock Mode Status Register (SCM_STR) is "1".*
- *Sub oscillation is enabled when RTCE bit is "1" regardless of the SOSCE bit value of System Clock Mode Control Register (SCM_CTL) and the SORDY bit value of the System Clock Mode Status Register (SCM_STR).*

## 8.5 Deep Standby Return Factor Register 1 (WRFSR)

The Deep Standby Return Factor Register 1 (WRFSR) indicates return factors by low-voltage detection reset and the INITX pin input reset that occur in deep standby mode.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | WLVDH | WINITX |
| Attribute | - | | | | | | R | R |
| Initial value | 000000 | | | | | | 0 | 0 |

### Register functions

#### [bit7:2] Reserved: Reserved bits

The read value is always "0". Writing has no effect on the operation.

#### [bit1] WLVHD: Low-voltage detection reset return bit

This bit indicates returning from deep standby mode by low-voltage detection reset.

| bit | Description |
|---|---|
| 0 | Not returned by low-voltage detection reset [initial value] |
| 1 | Returned by low-voltage detection reset |

#### [bit0] WINITX: INITX pin input reset return bit

This bit indicates returning from deep standby mode by INITX pin input reset.

| bit | Description |
|---|---|
| 0 | Not returned by INITX pin input reset [initial value] |
| 1 | Returned by INITX pin input reset |

***Notes:***

- *This register is initialized by power-on reset and low-voltage detection reset. It is not initialized by other reset factors. In addition, all bits are cleared by reading.*
- *Before transiting to the deep standby mode, ensure that the return factor from the deep standby mode is not set. If the factor is set, clear it.*
- *This register can be set only in the deep standby mode.*

## 8.6 Deep Standby Return Factor Register 2 (WIFSR)

The Deep Standby Return Factor Register 2 (WIFSR) indicates return factors by WKUPx pin input, low-voltage detection interrupt, and RTC interrupt, and HDMI-CEC/Remote Control Reception interrupt that occur in deep standby mode.

### Register configuration

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | WCEC1I | WCEC0I |
| Attribute | - | | | | | | R | R |
| Initial value | 000000 | | | | | | 0 | 0 |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | WUI5 | WUI4 | WUI3 | WUI2 | WUI1 | WUI0 | WLVDI | WRTCI |
| Attribute | R | R | R | R | R | R | R | R |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### Register functions

**[bit15:10] Reserved: Reserved bits**

The read value is always "0". Writing has no effect on the operation.

**[bit9] WCEC1I: CEC ch.1 interrupt return bit**

This bit indicates returning from deep standby mode by HDMI-CEC/ Remote Control Reception ch.1 interrupt.

| bit | Description |
|---|---|
| 0 | Not returned by HDMI-CEC/ Remote Control Reception ch.1 interrupt<br>[initial value] |
| 1 | Returned by HDMI-CEC/ Remote Control Reception ch.1 interrupt |

**[bit8] WCEC0I: CEC ch.0 interrupt return bit**

This bit indicates returning from deep standby mode by HDMI-CEC/ Remote Control Reception ch.0 interrupt.

| bit | Description |
|---|---|
| 0 | Not returned by HDMI-CEC/ Remote Control Reception ch.0 interrupt<br>[initial value] |
| 1 | Returned by HDMI-CEC/ Remote Control Reception ch.0 interrupt |

### [bit7:2] WUI5 to WUI0: WKUPx pin input return bits

These bits indicate returning from deep standby mode by WKUPx pin input.

| bit | Description |
|---|---|
| 0 | Not returned by WKUPx pin input [initial value] |
| 1 | Returned by WKUPx pin input |

### [bit1] WLVDI: LVD interrupt return bit

This bit indicates returning from deep standby mode by LVD interrupt.

| bit | Description |
|---|---|
| 0 | Not returned by LVD interrupt [initial value] |
| 1 | Returned by LVD interrupt |

### [bit0] WRTCI: RTC interrupt return bit

This bit indicates returning from deep standby mode by RTC interrupt.

| bit | Description |
|---|---|
| 0 | Not returned by RTC interrupt [initial value] |
| 1 | Returned by RTC interrupt |

***Notes:***

- *This register is initialized by power-on reset and low-voltage detection reset. It is not initialized by other reset factor. In addition, all bits are cleared by reading.*
- *Before transiting to the deep standby mode, ensure that the return factor from deep standby mode is not set. If the factor is set, clear it.*
- *This register can be set only in the deep standby mode.*

## 8.7 Deep Standby Return Enable Register (WIER)

The Deep Standby Return Enable Register (WIER) enables a return by WKUPx pin input, low-voltage detection interrupt, RTC interrupt, and HDMI-CEC/Remote Control Reception interrupt that occur in deep standby mode.

### Register configuration

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | WCEC1E | WCEC0E |
| Attribute | - | | | | | | R/W | R/W |
| Initial value | 000000 | | | | | | 0 | 0 |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | WUE5 | WUE4 | WUE3 | WUE2 | WUE1 | Reserved | WLVDE | WRTCE |
| Attribute | R/W | R/W | R/W | R/W | R/W | - | R/W | R/W |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### Register functions

**[bit15:10] Reserved: Reserved bits**

The read value is always "0". Writing has no effect on the operation.

**[bit9] WCEC1E: HDMI-CEC/ Remote Control Reception ch.1 interrupt return enable bit**

A return from deep standby mode by HDMI-CEC/ Remote Control Reception ch.1 interrupt is disabled or enabled.

| bit | Description |
|---|---|
| 0 | Disable a return by HDMI-CEC/ Remote Control Reception ch.1 interrupt<br>[initial value] |
| 1 | Enable a return by HDMI-CEC/ Remote Control Reception ch.1 interrupt |

**[bit8] WCEC0E: HDMI-CEC/ Remote Control Reception ch.0 interrupt return enable bit**

A return from deep standby mode by HDMI-CEC/ Remote Control Reception ch.0 interrupt is disabled or enabled.

| bit | Description |
|---|---|
| 0 | Disable a return by HDMI-CEC/ Remote Control Reception ch.0 interrupt<br>[initial value] |
| 1 | Enable a return by HDMI-CEC/ Remote Control Reception ch.0 interrupt |

### [bit7:3] WUI5E to WUI1E: WKUPx pin input return enable bits

A return from deep standby mode by WKUPx pin input is disabled or enabled.

| bit | Description |
|---|---|
| 0 | Disable a return by WKUPx pin input [initial value] |
| 1 | Enable a return by WKUPx pin input |

### [bit2] Reserved: Reserved bit

The read value is always "0". Writing has no effect on the operation.

### [bit1] WLVDE: LVD interrupt return enable bit

A return from deep standby mode by LVD interrupt is disabled or enabled.

| bit | Description |
|---|---|
| 0 | Disable a return by LVD interrupt [initial value] |
| 1 | Enable a return by LVD interrupt |

### [bit0] WRTCE: RTC interrupt return enable bit

A return from deep standby mode by RTC interrupt is disabled or enabled.

| bit | Description |
|---|---|
| 0 | Disable a return by RTC interrupt [initial value] |
| 1 | Enable a return by RTC interrupt |

***Notes:***

- *A return from deep standby mode by WKUP0 pin input is always enabled.*
- *This register is not initialized by deep standby transition reset.*

## 8.8 WKUP Pin Input Level Register (WILVR)

The WKUP Pin Input Level Register (WILVR) selects a valid level of WKUP1 to WKUP5 pin inputs that occur in deep standby mode.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | WUI5LV | WUI4LV | WUI3LV | WUI2LV | WUI1LV |
| Attribute | | - | | R/W | R/W | R/W | R/W | R/W |
| Initial value | | 000 | | 0 | 0 | 0 | 0 | 0 |

### Register functions

**[bit7:5] Reserved: Reserved bits**

The read value is always "0". Writing has no effect on the operation.

**[bit4:0] WUI5LV to WUI1LV: WKUPx pin input level select bits**

A valid level of WKUPx pin input is selected.

| bit | Description |
|---|---|
| 0 | Request a return when WKUPx pin input is Low level [initial value] |
| 1 | Request a return when WKUPx pin input is High level |

***Notes:***

- *WKUP0 pin input always requests a return in Low level*
  *For example, it returns as soon as it transits to deep standby mode when WKUP1 inputs in Low level (WUI1LV = 0).*
- *This register is not initialized by deep standby transition reset.*

## 8.9 Deep Standby RAM Retention Register (DSRAMR)

The Deep Standby RAM Retention Register (DSRAMR) controls the retention of the on-chip SRAM contents in deep standby modes.

### Register configuration

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | SRAMR | |
| Attribute | - | | | | | | R/W | |
| Initial value | 000000 | | | | | | 00 | |

### Register functions

**[bit7:2] Reserved: Reserved bits**

The read value is always "0". Writing has no effect on the operation.

**[bit1:0] SRAMR: On-chip SRAM retention control bits**

These bits control the retention of the on-chip SRAM contents in deep standby modes.

| bit1 | bit0 | Description |
|---|---|---|
| 0 | 0 | Not retain the on-chip SRAM contents in the deep standby mode. [initial value] |
| 0 | 1 | Setting is prohibited |
| 1 | 0 | Setting is prohibited |
| 1 | 1 | Retain the on-chip SRAM contents in the deep standby mode. |

***Note:***

- *This register is initialized by power-on reset and low-voltage detection reset. It is not initialized by other reset factor.*

## 8.10 Backup Registers 01 to 16 (BUR01 to 16)

The Backup Registers 01 to 16 (BUR01 to 16) are general registers that retain values in deep standby mode.

**Register configuration**

| bit | 31 24 | 23 16 | 15 8 | 7 0 |
|---|---|---|---|---|
| Field | BUR04 | BUR03 | BUR02 | BUR01 |
| Attribute | R/W | R/W | R/W | R/W |
| Initial value | 0x00 | 0x00 | 0x00 | 0x00 |

| bit | 31 24 | 23 16 | 15 8 | 7 0 |
|---|---|---|---|---|
| Field | BUR08 | BUR07 | BUR06 | BUR05 |
| Attribute | R/W | R/W | R/W | R/W |
| Initial value | 0x00 | 0x00 | 0x00 | 0x00 |

| bit | 31 24 | 23 16 | 15 8 | 7 0 |
|---|---|---|---|---|
| Field | BUR12 | BUR11 | BUR10 | BUR09 |
| Attribute | R/W | R/W | R/W | R/W |
| Initial value | 0x00 | 0x00 | 0x00 | 0x00 |

| bit | 31 24 | 23 16 | 15 8 | 7 0 |
|---|---|---|---|---|
| Field | BUR16 | BUR15 | BUR14 | BUR13 |
| Attribute | R/W | R/W | R/W | R/W |
| Initial value | 0x00 | 0x00 | 0x00 | 0x00 |

***Notes:***

- *This register is initialized by power-on reset and low-voltage detection reset. It is not initialized by other reset factor.*

# 9. Usage Precautions

Pay attention to the following points when using low power consumption mode.

For the pin shared for analog input and WKUP, WKUPx pin input is blocked when ADE bit of corresponding analog input setting register (ADE) is set to "1" even if the recovery by WKUPx pin input is allowed. To use the recovery by WKUPx pin input, set ADE bit of corresponding analog input setting register (ADE) to "0" before shifting to deep standby mode.

# CHAPTER7-1: Interrupts

**This chapter explains the interrupt controller and peripheral interrupt requests.**



CODE: 9AFIRQC_A-E01.0

# 1. Overview

The interrupt controller determines the priority of interrupt requests and sends the requests to the CPU. The Cortex-M0+ CPU core is equipped with the nested vectored interrupt controller (NVIC) internally within the core. Interrupt signals from several peripherals are aggregated and input to a single interrupt factor vector. The interrupt requests that have occurred can be checked using the interrupt request batch read register. Furthermore, for some of the interrupt factors, the interrupt requests can be configured to be converted into DMA request signals.

### Features of the Nested Vectored Interrupt Controller (NVIC)

- 32 maskable peripheral interrupt channels (not including the 16 exception interrupts of Cortex-M0+)
- 4 programmable interrupt priority levels (using 2-bit prioritized interrupts)
- Facilitates low-latency exception and interrupt handling
- Implements System Control Registers
- Supports non-maskable interrupt (NMI) input

The NVIC and the processor core interface are closely coupled, providing mechanisms that enable low-latency interrupt processing and efficient processing of late arriving interrupts. The NVIC maintains the nested interrupt information to enable tail chaining of interrupts.

All interrupts are managed by the NVIC, including core exceptions. See "Chapter 5: Exceptions" and "Chapter 8: Nested Vectored Interrupt Controller" in the "Cortex-M0+ Technical Reference Manual" published by ARM for details on exceptions and NVIC.

***Note:***

- *In the "Cortex-M0+ Technical Reference Manual", all exception type:IRQ are defined as external interrupt inputs. In this manual, exception type:IRQ are expressed as peripheral interrupts. Peripheral interrupts include "External Interrupt and NMI Control Unit" interrupts from external pins and interrupts from peripheral resources within the LSI.*

### Interrupt Factor Aggregation Function

The interrupt request signals from each peripheral resource are aggregated into 32 sources and input to the NVIC. Furthermore, the interrupt request signal from the external NMIX pin is logically OR'ed with the hardware watchdog interrupt signal and input to the NVIC.

### Peripheral Interrupt Request Batch Read Function

The interrupt request batch read register allows the interrupt request signals from the peripheral resources aggregated into a single interrupt request signal to be read out at once. Reading this register makes it possible to check which interrupt request has occurred. However, the interrupt request flags cannot be cleared by using this function. Clear the interrupt request flags using the registers of each peripheral function.

### DMA transfer request output select function

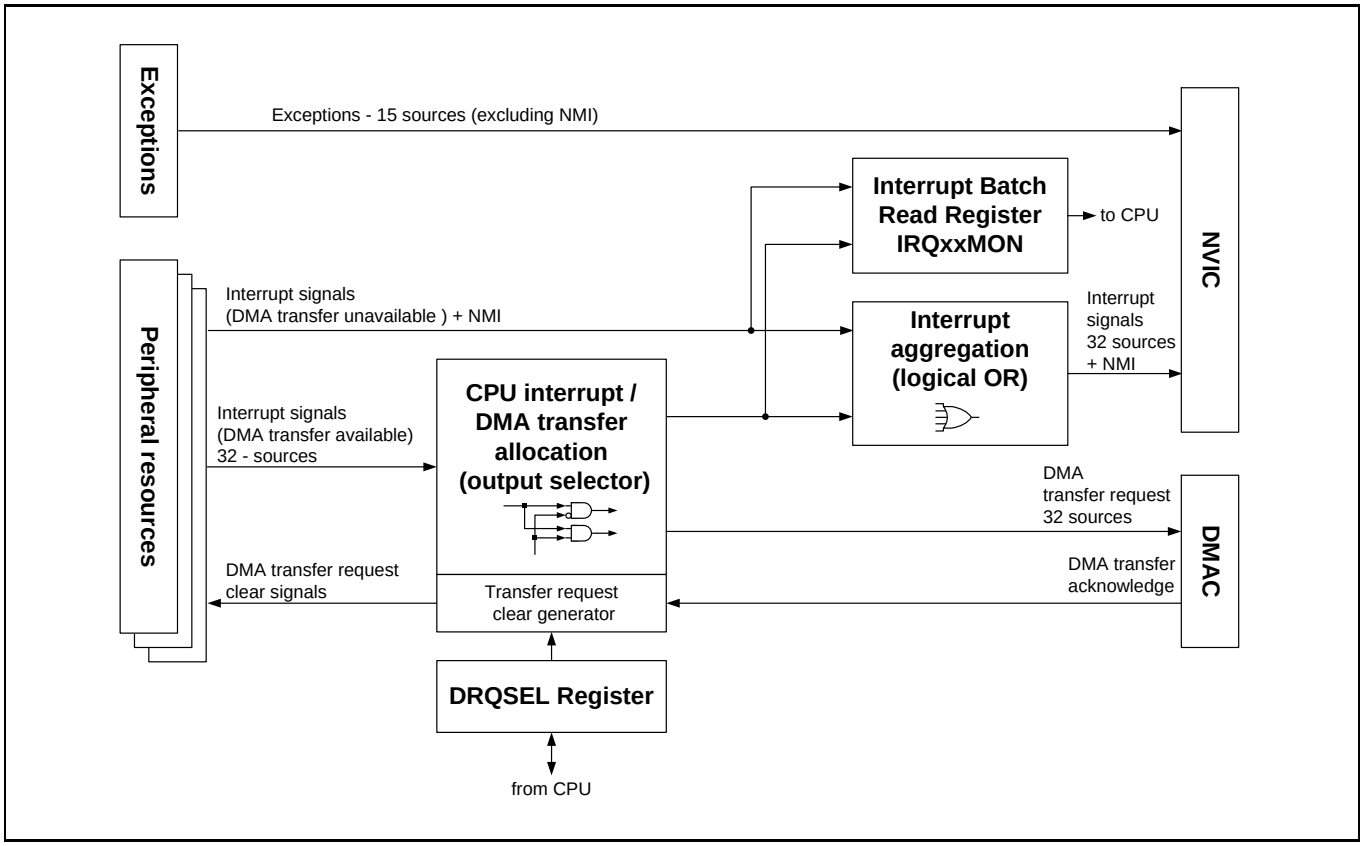
DMA transfer can be activated using interrupt request from some peripheral functions. By the DRQSEL register, select to output interrupt request signal from each peripheral resource to the CPU as the interrupt request signal or to output it to DMAC as a transfer request signal. For the DMA transfer request signal, see the chapter "DMAC".

# 2. Configuration

This section shows the configuration of the relationship between the interrupt controller and DMA transfer requests.

### Block diagram of interrupt controller and DMA transfer request

**Figure 2-1 Block diagram of interrupt controller and DMA transfer request**




- Interrupt factor aggregation block
  Aggregate (logical OR) interrupt request signals from each peripheral resource to 32 factors and output them to NVIC.
- Peripheral interrupt request batch read register block
  For interrupt request signals from a peripheral resource aggregated to one interrupt request signal, this register can check what interrupt request of each peripheral resource signal generates such interrupt.
- CPU interrupt request/DMA transfer request allocation block
  This is the output selector that selects whether to output interrupt request signal from a peripheral resource to the CPU as an interrupt request signal or to DMAC as a transfer request signal by using the DRQSEL register setting.

■ Interrupt factor vector relocate function

Two types of the interrupt factor vector shown in 2 can be selected by the IRQCMODE register setting. For IRQCMODE bit, refer to "Interrupts (B)".For the details of each setting, refer to the each chapter as following Table 1-1.

Moreover, the arbitrary interrupt factor can be selected with the RCINTSEL0 and RCINTSEL1 registers. For details on RCINTSEL0 and RCINTSEL1 registers, see "Interrupts (B)".

**Table 1-1 Correspondence table for Interrupt chapter**

| IRQCMODE setting | Reference |
|---|---|
| IRQCMODE=0<br>Relocate not selected | Chapter "Interrupts (A)" |
| IRQCMODE=1<br>Relocate selected | Chapter "Interrupts (B)" |

**Table 1-2 Exceptions and Interrupt factor vectors list**

| Vector No. | IRQ No. | Exceptions and Interrupt factor vectors | |
|---|---|---|---|
| | | IRQCMODE=0 | IRQCMODE=1 |
| 0 | - | Stack pointer initial value | |
| 1 | - | Reset | |
| 2 | - | Non-Maskable Interrupt (NMI) / Hardware Watchdog Timer | |
| 3 | - | Hard Fault | |
| 4 | - | Memory Management | |
| 5 | - | Bus Fault | |
| 6 | - | Usage Fault | |
| 7 to 10 | - | Reserved | |
| 11 | - | SVCall (Supervisor Call) | |
| 12 | - | Debug Monitor | |
| 13 | - | Reserved | |
| 14 | - | PendSV | |
| 15 | - | SysTick | |
| 16 | 0 | Anomalous Frequency Detection by Clock Supervisor (FCS) | |
| 17 | 1 | Software Watchdog Timer | |
| 18 | 2 | Low Voltage Detector (LVD) | |
| 19 | 3 | MFT unit 0, unit 1, unit 2 Wave Form Generator / DTIF(Motor Emergency Stop) | Selecting the interrupt factor with RCINTSEL0 register |
| 20 | 4 | External Pin Interrupt ch.0 to ch.7 | Selecting the interrupt factor with RCINTSEL0 register |
| 21 | 5 | External Pin Interrupt ch.8 to ch.31 | Selecting the interrupt factor with RCINTSEL0 register |
| 22 | 6 | Dual Timer / Quad Counter (QPRC) ch.0, ch.1, ch.2 | Selecting the interrupt factor with RCINTSEL0 register |
| 23 | 7 | Reception Interrupt of MFS ch.0 / Reception Interrupt of MFS ch.8 | Selecting the interrupt factor with RCINTSEL1 register |
| 24 | 8 | Transmission Interrupt and Status Interrupt of MFS ch.0 / Transmission Interrupt and Status Interrupt of MFS ch.8 | Selecting the interrupt factor with RCINTSEL1 register |
| 25 | 9 | Reception Interrupt of MFS ch.1 / Reception Interrupt of MFS ch.9 | Selecting the interrupt factor with RCINTSEL1 register |
| 26 | 10 | Transmission Interrupt and Status Interrupt of MFS ch.1 / Transmission Interrupt and Status Interrupt of MFS ch.9 | Selecting the interrupt factor with RCINTSEL1 register |

| Vector No. | IRQ No. | Exceptions and Interrupt factor vectors | |
|---|---|---|---|
| | | IRQCMODE=0 | IRQCMODE=1 |
| 27 | 11 | Reception Interrupt of MFS ch.2 / Reception Interrupt of MFS ch.10 | MFT unit 0 Wave Form Generator / DTIF(Motor Emergency Stop) / Reception Interrupt, Transmission Interrupt, and Status Interrupt of MFS ch.8 |
| 28 | 12 | Transmission Interrupt and Status Interrupt of MFS ch.2 / Transmission Interrupt and Status Interrupt of MFS ch.10 | External pin interrupt ch.0 to ch.7 |
| 29 | 13 | Reception Interrupt of MFS ch.3 / Reception Interrupt of MFS ch.11 | External pin interrupt ch.8 to ch.31 |
| 30 | 14 | Transmission Interrupt and Status Interrupt of MFS ch.3 / Transmission Interrupt and Status Interrupt of MFS ch.11 | Dual Timer / Quad Counter (QPRC) ch.0 |
| 31 | 15 | Reception Interrupt of MFS ch.4 / Reception Interrupt of MFS ch.12 | Reception Interrupt, Transmission Interrupt and Status Interrupt of MFS ch.0 |
| 32 | 16 | Transmission Interrupt and Status Interrupt of MFS ch.4 / Transmission Interrupt and Status Interrupt of MFS ch.12 | Reception Interrupt, Transmission Interrupt and Status Interrupt of MFS ch.1 |
| 33 | 17 | Reception Interrupt of MFS ch.5 / Reception Interrupt of MFS ch.13 | Reception Interrupt, Transmission Interrupt and Status Interrupt of MFS ch.2 |
| 34 | 18 | Transmission Interrupt and Status Interrupt of MFS ch.5 / Transmission Interrupt and Status Interrupt of MFS ch.13 | Reception Interrupt, Transmission Interrupt and Status Interrupt of MFS ch.3 |
| 35 | 19 | Reception Interrupt of MFS ch.6 / Reception Interrupt of MFS ch.14 / DMA Controller (DMAC) ch.0 | Reception Interrupt of MFS ch.4 |
| 36 | 20 | Transmission Interrupt and Status Interrupt of MFS ch.6 / Transmission Interrupt and Status Interrupt of MFS ch.14 / DMA Controller (DMAC) ch.1 | Transmission Interrupt and Status Interrupt of MFS ch.4 |
| 37 | 21 | Reception Interrupt of MFS ch.7 / Reception Interrupt of MFS ch.15 / DMA Controller (DMAC) ch.2 | Reception Interrupt of MFS ch.5 |
| 38 | 22 | Transmission Interrupt and Status Interrupt of MFS ch.7 / Transmission Interrupt and Status Interrupt of MFS ch.15 / DMA Controller (DMAC) ch.3 | Transmission Interrupt and Status Interrupt of MFS ch.5 |
| 39 | 23 | PPG ch.0/2/4/8/10/12/16/18/20 | |
| 40 | 24 | External Main OSC / External Sub OSC / Main PLL / Watch Counter/Real Time Counter | |
| 41 | 25 | A/D Converter unit 0 | A/D Converter unit 0 /Reception Interrupt, Transmission Interrupt, and Status Interrupt of MFS ch.9 |
| 42 | 26 | A/D Converter unit 1 | A/D Converter unit 1 /Reception Interrupt, Transmission Interrupt, and Status Interrupt of MFS ch.10 |
| 43 | 27 | A/D Converter unit 2 / LCD Controller | A/D Converter unit 2/ LCD Controller / Reception Interrupt, Transmission Interrupt, and Status Interrupt of MFS ch.11 |
| 44 | 28 | MFT unit 0, unit 1, unit 2 Free-run Timer | MFT unit 0 Free-run Timer, Input Capture, Output Compare |

| Vector No. | IRQ No. | Exceptions and Interrupt factor vectors | |
|---|---|---|---|
| | | IRQCMODE=0 | IRQCMODE=1 |
| 45 | 29 | MFT unit 0, unit 1, unit 2 Input Capture | MFT unit 1 Free-run Timer, Input Capture, Output Compare |
| 46 | 30 | MFT unit 0, unit 1, unit 2 Output Compare | DMA Controller (DMAC) ch.0 to ch.7 |
| 47 | 31 | Base Timer ch.0 to ch.7 / Flash RDY interrupt /Flash HANG interrupt | |

# CHAPTER7-2: Interrupts (A)

**This chapter explains Exception and Interrupt Factor Vectors and Registers at IRQCMODE=0.**



CODE: 9AFIRQC_A-E01.0

# 1. Exception and Interrupt Factor Vectors

This section shows a vector table of the exceptions and interrupts input to the NVIC.

**Table 1-1 Exception and interrupt factor vectors**

| Vector No. | IRQ No. | Exception and Interrupt Factor | Vector Offset |
|---|---|---|---|
| 0 | - | Stack pointer initial value | 0x00 |
| 1 | - | Reset | 0x04 |
| 2 | - | Non-Maskable Interrupt (NMI) / Hardware Watchdog Timer | 0x08 |
| 3 | - | Hard Fault | 0x0C |
| 4 | - | Memory Management | 0x10 |
| 5 | - | Bus Fault | 0x14 |
| 6 | - | Usage Fault | 0x18 |
| 7-10 | - | Reserved | 0x1C - 0x2B |
| 11 | - | SVCall (Supervisor Call) | 0x2C |
| 12 | - | Debug Monitor | 0x30 |
| 13 | - | Reserved | 0x34 |
| 14 | - | PendSV | 0x38 |
| 15 | - | SysTick | 0x3C |
| 16 | 0 | Anomalous Frequency Detection by Clock Supervisor (FCS) | 0x40 |
| 17 | 1 | Software Watchdog Timer | 0x44 |
| 18 | 2 | Low Voltage Detector (LVD) | 0x48 |
| 19 | 3 | MFT unit 0, unit 1, unit 2 Wave Form Generator / DTIF(Motor Emergency Stop) | 0x4C |
| 20 | 4 | External Pin Interrupt Request ch.0 to ch.7 | 0x50 |
| 21 | 5 | External Pin Interrupt Request ch.8 to ch.31 | 0x54 |
| 22 | 6 | Dual Timer / Quad Counter (QPRC) ch.0, ch.1, ch.2 | 0x58 |
| 23 | 7 | Reception Interrupt Request of MFS ch.0 / Reception Interrupt Request of MFS ch.8 | 0x5C |
| 24 | 8 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.0 / Transmission Interrupt Request and Status Interrupt Request of MFS ch.8 | 0x60 |
| 25 | 9 | Reception Interrupt Request of MFS ch.1 / Reception Interrupt Request of MFS ch.9 | 0x64 |
| 26 | 10 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.1 / Transmission Interrupt Request and Status Interrupt Request of MFS ch.9 | 0x68 |
| 27 | 11 | Reception Interrupt Request of MFS ch.2 / Reception Interrupt Request of MFS ch.10 | 0x6C |
| 28 | 12 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.2 / Transmission Interrupt Request and Status Interrupt Request of MFS ch.10 | 0x70 |
| 29 | 13 | Reception Interrupt Request of MFS ch.3 / Reception Interrupt Request of MFS ch.11 | 0x74 |
| 30 | 14 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.3 / Transmission Interrupt Request and Status Interrupt Request of MFS ch.11 | 0x78 |
| 31 | 15 | Reception Interrupt Request of MFS ch.4 / Reception Interrupt Request of MFS ch.12 | 0x7C |
| 32 | 16 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.4 / Transmission Interrupt Request and Status Interrupt Request of MFS ch.12 | 0x80 |
| 33 | 17 | Reception Interrupt Request of MFS ch.5 / Reception Interrupt Request of MFS ch.13 | 0x84 |
| 34 | 18 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.5 / Transmission Interrupt Request and Status Interrupt Request of MFS ch.13 | 0x88 |
| 35 | 19 | Reception Interrupt Request of MFS ch.6 / Reception Interrupt Request of MFS ch.14 / DMAC ch.0 | 0x8C |
| 36 | 20 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.6 / Transmission Interrupt Request and Status Interrupt Request of MFS ch.14 / DMAC ch.1 | 0x90 |

| Vector No. | IRQ No. | Exception and Interrupt Factor | Vector Offset |
|---|---|---|---|
| 37 | 21 | Reception Interrupt Request of MFS ch.7 / Reception Interrupt Request of MFS ch.15 / DMAC ch.2 | 0x94 |
| 38 | 22 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.7 / Transmission Interrupt Request and Status Interrupt Request of MFS ch.15 / DMAC ch.3 | 0x98 |
| 39 | 23 | PPG ch.0/2/4/8/10/12/16/18/20 | 0x9C |
| 40 | 24 | External Main OSC / External Sub OSC / Main PLL / Watch Counter/ Real Time Counter | 0xA0 |
| 41 | 25 | A/D Converter unit 0 | 0xA4 |
| 42 | 26 | A/D Converter unit 1 | 0xA8 |
| 43 | 27 | A/D Converter unit 2 / LCD Controller | 0xAC |
| 44 | 28 | MFT unit 0, unit 1, unit 2 Free-run Timer | 0xB0 |
| 45 | 29 | MFT unit 0, unit 1, unit 2 Input Capture | 0xB4 |
| 46 | 30 | MFT unit 0, unit 1, unit 2 Output Compare | 0xB8 |
| 47 | 31 | Base Timer ch.0 to ch.7/ Flash RDY interrupt / Flash HANG interrupt | 0xBC |

The priorities of the exceptions for vectors No. 4 to No. 15 can be configured using the System Handler Priority Registers (address 0xE000ED18, 0xE000ED1C, 0xE000ED20) built into the NVIC. The priorities of the peripheral interrupts for vectors No. 16 and after can be configured using the IRQ Priority Registers (address 0xE000E400 to 0xE000E42C) built into the NVIC.

The sources of the interrupts for vectors No. 2 and No. 16 to No. 47 can be checked using the batch read register. See "Chapter 8: Nested Vectored Interrupt Controller" in the "Cortex-M0+ Technical Reference Manual" for details on the other exceptions and interrupts.

Furthermore, for the interrupts of vectors No. 2 and No. 16 to No. 47, the sources that are batch read may be a signal that multiple interrupt factors are logical OR'ed within each of the peripheral macros. See the descriptions of each of the peripheral resource interrupts for details.

# 2. Registers

This section describes the DMA transfer request selection registers and the interrupt request batch read registers.

**List of DMA transfer request selection registers and interrupt request batch read registers**

**Table 2-1 List of DMA transfer request selection registers and interrupt request batch read registers**

| Abbreviation | Register Name | Reference |
|---|---|---|
| DRQSEL | DMA Request Selection Register | 2.1 |
| EXC02MON | EXC02 Batch Read Register | 2.2 |
| IRQ00MON | IRQ00 Batch Read Register | 2.3 |
| IRQ01MON | IRQ01 Batch Read Register | 2.4 |
| IRQ02MON | IRQ02 Batch Read Register | 2.5 |
| IRQ03MON | IRQ03 Batch Read Register | 2.6 |
| IRQ04MON | IRQ04 Batch Read Register | 2.7 |
| IRQ05MON | IRQ05 Batch Read Register | 2.8 |
| IRQ06MON | IRQ06 Batch Read Register | 2.9 |
| IRQ07MON | IRQ07 Batch Read Register | 2.10 |
| IRQ08MON | IRQ08 Batch Read Register | 2.12 |
| IRQ09MON | IRQ09 Batch Read Register | 2.10 |
| IRQ10MON | IRQ10 Batch Read Register | 2.12 |
| IRQ11MON | IRQ11 Batch Read Register | 2.10 |
| IRQ12MON | IRQ12 Batch Read Register | 2.12 |
| IRQ13MON | IRQ13 Batch Read Register | 2.10 |
| IRQ14MON | IRQ14 Batch Read Register | 2.12 |
| IRQ15MON | IRQ15 Batch Read Register | 2.10 |
| IRQ16MON | IRQ16 Batch Read Register | 2.12 |
| IRQ17MON | IRQ17 Batch Read Register | 2.10 |
| IRQ18MON | IRQ18 Batch Read Register | 2.12 |
| IRQ19MON | IRQ19 Batch Read Register | 2.11 |
| IRQ20MON | IRQ20 Batch Read Register | 2.13 |
| IRQ21MON | IRQ21 Batch Read Register | 2.11 |
| IRQ22MON | IRQ22 Batch Read Register | 2.13 |
| IRQ23MON | IRQ23 Batch Read Register | 2.14 |
| IRQ24MON | IRQ24 Batch Read Register | 2.15 |
| IRQ25MON | IRQ25 Batch Read Register | 2.16 |
| IRQ26MON | IRQ26 Batch Read Register | 2.16 |
| IRQ27MON | IRQ27 Batch Read Register | 2.17 |
| IRQ28MON | IRQ28 Batch Read Register | 2.18 |
| IRQ29MON | IRQ29 Batch Read Register | 2.19 |
| IRQ30MON | IRQ30 Batch Read Register | 2.20 |
| IRQ31MON | IRQ31 Batch Read Register | 2.21 |

See "Chapter 5: Nested Vectored Interrupt Controller" in the "Cortex-M0+ Technical Reference Manual" for details on the registers in the NVIC.

## 2.1 DMA Request Selection Register (DRQSEL)

The DMA Request Selection Register (DRQSEL) selects whether interrupt signals that can start DMA transfers are output as interrupt requests to the CPU or output as transfer requests to the DMAC.

### Register configuration

| bit | 31 | 5 4 3 2 1 0 |
|---|---|---|
| Field | DRQSEL[31:5] | Reserved |
| Attribute | R/W | R |
| Initial value | 00000000000000000000000000 | 00000 |

### Register functions

**[bit31:5] DRQSEL :**

| bit No. | bit | Description |
|---|---|---|
| 31 | 0 | The interrupt signal of the external interrupt ch.3 is output as an interrupt request to the CPU. |
| | 1 | The interrupt signal of the external interrupt ch.3 is output as a transfer request to the DMAC. |
| 30 | 0 | The interrupt signal of the external interrupt ch.2 is output as an interrupt request to the CPU. |
| | 1 | The interrupt signal of the external interrupt ch.2 is output as a transfer request to the DMAC. |
| 29 | 0 | The interrupt signal of the external interrupt ch.1 is output as an interrupt request to the CPU. |
| | 1 | The interrupt signal of the external interrupt ch.1 is output as a transfer request to the DMAC. |
| 28 | 0 | The interrupt signal of the external interrupt ch.0 is output as an interrupt request to the CPU. |
| | 1 | The interrupt signal of the external interrupt ch.0 is output as a transfer request to the DMAC. |
| 27 | 0 | The transmission interrupt signal of the MFS ch.7 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.7 is output as a transfer request to the DMAC. |
| 26 | 0 | The reception interrupt signal of the MFS ch.7 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.7 is output as a transfer request to the DMAC. |
| 25 | 0 | The transmission interrupt signal of the MFS ch.6 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.6 is output as a transfer request to the DMAC. |
| 24 | 0 | The reception interrupt signal of the MFS ch.6 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.6 is output as a transfer request to the DMAC. |
| 23 | 0 | The transmission interrupt signal of the MFS ch.5 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.5 is output as a transfer request to the DMAC. |
| 22 | 0 | The reception interrupt signal of the MFS ch.5 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.5 is output as a transfer request to the DMAC. |
| 21 | 0 | The transmission interrupt signal of the MFS ch.4 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.4 is output as a transfer request to the DMAC. |
| 20 | 0 | The reception interrupt signal of the MFS ch.4 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.4 is output as a transfer request to the DMAC. |
| 19 | 0 | The transmission interrupt signal of the MFS ch.3 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.3 is output as a transfer request to the DMAC. |
| 18 | 0 | The reception interrupt signal of the MFS ch.3 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.3 is output as a transfer request to the DMAC. |
| 17 | 0 | The transmission interrupt signal of the MFS ch.2 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.2 is output as a transfer request to the DMAC. |
| 16 | 0 | The reception interrupt signal of the MFS ch.2 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.2 is output as a transfer request to the DMAC. |
| 15 | 0 | The transmission interrupt signal of the MFS ch.1 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.1 is output as a transfer request to the DMAC. |
| 14 | 0 | The reception interrupt signal of the MFS ch.1 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.1 is output as a transfer request to the DMAC. |

| bit No. | bit | Description |
|---|---|---|
| 13 | 0 | The transmission interrupt signal of the MFS ch.0 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.0 is output as a transfer request to the DMAC. |
| 12 | 0 | The reception interrupt signal of the MFS ch.0 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.0 is output as a transfer request to the DMAC. |
| 11 | 0 | The IRQ0 interrupt signal of the base timer ch.6 is output as an interrupt request to the CPU. |
| | 1 | The IRQ0 interrupt signal of the base timer ch.6 is output as a transfer request to the DMAC (including extension). |
| 10 | 0 | Setting is prohibited.The IRQ0 interrupt signal of the base timer ch.4 is output as an interrupt request to the CPU. |
| | 1 | The IRQ0 interrupt signal of the base timer ch.4 is output as a transfer request to the DMAC (including extension). |
| 9 | 0 | The IRQ0 interrupt signal of the base timer ch.2 is output as an interrupt request to the CPU. |
| | 1 | The IRQ0 interrupt signal of the base timer ch.2 is output as a transfer request to the DMAC. |
| 8 | 0 | The IRQ0 interrupt signal of the base timer ch.0 is output as an interrupt request to the CPU. |
| | 1 | The IRQ0 interrupt signal of the base timer ch.0 is output as a transfer request to the DMAC. |
| 7 | 0 | The scan conversion interrupt signal of the A/D converter unit 2 is output as an interrupt request to the CPU. |
| | 1 | The scan conversion interrupt signal of the A/D converter unit 2 is output as a transfer request to the DMAC. |
| 6 | 0 | The scan conversion interrupt signal of the A/D converter unit 1 is output as an interrupt request to the CPU. |
| | 1 | The scan conversion interrupt signal of the A/D converter unit 1 is output as a transfer request to the DMAC. |
| 5 | 0 | The scan conversion interrupt signal of the A/D converter unit 0 is output as an interrupt request to the CPU. |
| | 1 | The scan conversion interrupt signal of the A/D converter unit 0 is output as a transfer request to the DMAC. |

MFS: Multifunction serial interface

### [bit4:0] Reserved: Reserved bits

A reserved bit reads "0".

***Notes:***

- *If interrupt signal is selected as a transfer request to DMAC, the read bit value of the appropriate interrupt request batch read register (IRQxxMON, xx = 00 to 31) becomes "0" regardless of the interrupt occurrence.*
- *When changing the DRQSEL settings, clear the interrupt request signals from appropriate peripherals before making the change.*
- *DMA transfers cannot be started from hardware for interrupt signals not specified in the DRQSEL settings.*

## 2.2 EXC02 Batch Read Register (EXC02MON)

The EXC02 Batch Read Register (EXC02MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 2.

EXC02MON indicates the status of the interrupt requests of the hardware watchdog timer and NMIX external pin.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | HWINT | NMI |
| Attribute | R | | | | | | | | | | | | | | R | R |
| Initial value | 00000000000000 | | | | | | | | | | | | | | 0 | 0 |

### Register functions

**[bit31:2] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit1] HWINT:**

| bit | Description |
|---|---|
| 0 | No hardware watchdog timer interrupt request |
| 1 | Hardware watchdog timer interrupt request |

**[bit0] NMI:**

| bit | Description |
|---|---|
| 0 | No NMIX external pin interrupt request |
| 1 | NMIX external pin interrupt request |

## 2.3 IRQ00 Batch Read Register (IRQ00MON)

The IRQ00 Batch Read Register (IRQ00MON) can batch-read the interrupt request allocated to interrupt factor vector No. 16.

IRQ00MON indicates the status of the interrupt request of anomalous frequency detection by the CSV.

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | FCSINT |
| Attribute | R | | | | | | | | | | | | | | | R |
| Initial value | 000000000000000 | | | | | | | | | | | | | | | 0 |

**Register functions**

**[bit31:1] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit0] FCSINT:**

| bit | Description |
|---|---|
| 0 | No anomalous frequency detection by CSV interrupt request |
| 1 | Anomalous frequency detection by CSV interrupt request |

## 2.4 IRQ01 Batch Read Register (IRQ01MON)

The IRQ01 Batch Read Register (IRQ01MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 17.

IRQ01MON indicates the status of the interrupt request of the software watchdog timer.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | SWWDTINT |
| Attribute | R | | | | | | | | | | | | | | | R |
| Initial value | 000000000000000 | | | | | | | | | | | | | | | 0 |

### Register functions

**[bit31:1] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit0] SWWDTINT:**

| bit | Description |
|---|---|
| 0 | No software watchdog timer interrupt request |
| 1 | Software watchdog timer interrupt request |

## 2.5 IRQ02 Batch Read Register (IRQ02MON)

The IRQ02 Batch Read Register (IRQ02MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 18.

IRQ02MON indicates the status of the interrupt request of low voltage detection (LVD).

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | LVDINT |
| Attribute | R | | | | | | | | | | | | | | | R |
| Initial value | 000000000000000 | | | | | | | | | | | | | | | 0 |

### Register functions

**[bit31:1] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit0] LVDINT:**

| bit | Description |
|---|---|
| 0 | No low voltage detection (LVD) interrupt request |
| 1 | Low voltage detection (LVD) interrupt request |

## 2.6 IRQ03 Batch Read Register (IRQ03MON)

The IRQ03 Batch Read Register (IRQ03MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 19.

IRQ02MON indicates the status of the interrupt requests of MFT unit 0, MFT unit 1 and MFT unit 2.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | WAVE2INT | | | | WAVE1INT | | | | WAVE0INT | | | |
| Attribute | R | | | | R | | | | R | | | | R | | | |
| Initial value | 0000 | | | | 0000 | | | | 0000 | | | | 0000 | | | |

### Register functions

**[bit31:12] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit11:8] WAVE2INT:**

| bit No. | bit | Description |
|---|---|---|
| 11 | 0 | No WFG timer 54 interrupt request in MFT unit 2 |
| | 1 | WFG timer 54 interrupt request in MFT unit 2 |
| 10 | 0 | No WFG timer 32 interrupt request in MFT unit 2 |
| | 1 | WFG timer 32 interrupt request in MFT unit 2 |
| 9 | 0 | No WFG timer 10 interrupt request in MFT unit 2 |
| | 1 | WFG timer 10 interrupt request in MFT unit 2 |
| 8 | 0 | No DTIF (motor emergency stop) interrupt request in MFT unit 2 |
| | 1 | DTIF (motor emergency stop) interrupt request in MFT unit 2 |

**[bit7:4] WAVE1INT:**

| bit No. | bit | Description |
|---|---|---|
| 7 | 0 | No WFG timer 54 interrupt request in MFT unit 1 |
| | 1 | WFG timer 54 interrupt request in MFT unit 1 |
| 6 | 0 | No WFG timer 32 interrupt request in MFT unit 1 |
| | 1 | WFG timer 32 interrupt request in MFT unit 1 |
| 5 | 0 | No WFG timer 10 interrupt request in MFT unit 1 |
| | 1 | WFG timer 10 interrupt request in MFT unit 1 |
| 4 | 0 | No DTIF (motor emergency stop) interrupt request in MFT unit 1 |
| | 1 | DTIF (motor emergency stop) interrupt request in MFT unit 1 |

**[bit3:0] WAVE0INT:**

| bit No. | bit | Description |
|---|---|---|
| 3 | 0 | No WFG timer 54 interrupt request in MFT unit 0 |
| | 1 | WFG timer 54 interrupt request in MFT unit 0 |
| 2 | 0 | No WFG timer 32 interrupt request in MFT unit 0 |
| | 1 | WFG timer 32 interrupt request in MFT unit 0 |
| 1 | 0 | No WFG timer 10 interrupt request in MFT unit 0 |
| | 1 | WFG timer 10 interrupt request in MFT unit 0 |
| 0 | 0 | No DTIF (motor emergency stop) interrupt request in MFT unit 0 |
| | 1 | DTIF (motor emergency stop) interrupt request in MFT unit 0 |

## 2.7 IRQ04 Batch Read Register (IRQ04MON)

The IRQ04 Batch Read Register (IRQ04MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 20.

IRQ04MON indicates the status of the interrupt requests of the external interrupt ch.0 to ch.7.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | EXTINT | | | | | | | |
| Attribute | R | | | | | | | | R | | | | | | | |
| Initial value | 0x00 | | | | | | | | 0x00 | | | | | | | |

### Register functions

**[bit31:8] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit7:0] EXTINT:**

| bit No. | bit | Description |
|---|---|---|
| 7 | 0 | No interrupt request of external interrupt ch.7 |
| | 1 | Interrupt request of external interrupt ch.7 |
| 6 | 0 | No interrupt request of external interrupt ch.6 |
| | 1 | Interrupt request of external interrupt ch.6 |
| 5 | 0 | No interrupt request of external interrupt ch.5 |
| | 1 | Interrupt request of external interrupt ch.5 |
| 4 | 0 | No interrupt request of external interrupt ch.4 |
| | 1 | Interrupt request of external interrupt ch.4 |
| 3 | 0 | No interrupt request of external interrupt ch.3 |
| | 1 | Interrupt request of external interrupt ch.3 |
| 2 | 0 | No interrupt request of external interrupt ch.2 |
| | 1 | Interrupt request of external interrupt ch.2 |
| 1 | 0 | No interrupt request of external interrupt ch.1 |
| | 1 | Interrupt request of external interrupt ch.1 |
| 0 | 0 | No interrupt request of external interrupt ch.0 |
| | 1 | Interrupt request of external interrupt ch.0 |

If DMA transfer requests are selected by the DRQSEL register, the corresponding EXTINT bit is "0".

## 2.8 IRQ05 Batch Read Register (IRQ05MON)

The IRQ05 Batch Read Register (IRQ05MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 21.

IRQ05MON indicates the status of the interrupt requests of the external interrupt ch.8 to ch.31.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | EXTINT | | | | | | | |
| Attribute | R | | | | | | | | R | | | | | | | |
| Initial value | 0x00 | | | | | | | | 0x00 | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | EXTINT | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

### Register functions

**[bit31:24] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit23:0] EXTINT:**

| bit No. | bit | Description |
|---|---|---|
| 23 | 0 | No interrupt request of external interrupt ch.31 |
| | 1 | Interrupt request of external interrupt ch.31 |
| 22 | 0 | No interrupt request of external interrupt ch.30 |
| | 1 | Interrupt request of external interrupt ch.30 |
| 21 | 0 | No interrupt request of external interrupt ch.29 |
| | 1 | Interrupt request of external interrupt ch.29 |
| 20 | 0 | No interrupt request of external interrupt ch.28 |
| | 1 | Interrupt request of external interrupt ch.28 |
| 19 | 0 | No interrupt request of external interrupt ch.27 |
| | 1 | Interrupt request of external interrupt ch.27 |
| 18 | 0 | No interrupt request of external interrupt ch.26 |
| | 1 | Interrupt request of external interrupt ch.26 |
| 17 | 0 | No interrupt request of external interrupt ch.25 |
| | 1 | Interrupt request of external interrupt ch.25 |
| 16 | 0 | No interrupt request of external interrupt ch.24 |
| | 1 | Interrupt request of external interrupt ch.24 |
| 15 | 0 | No interrupt request of external interrupt ch.23 |
| | 1 | Interrupt request of external interrupt ch.23 |
| 14 | 0 | No interrupt request of external interrupt ch.22 |
| | 1 | Interrupt request of external interrupt ch.22 |
| 13 | 0 | No interrupt request of external interrupt ch.21 |
| | 1 | Interrupt request of external interrupt ch.21 |
| 12 | 0 | No interrupt request of external interrupt ch.20 |
| | 1 | Interrupt request of external interrupt ch.20 |
| 11 | 0 | No interrupt request of external interrupt ch.19 |
| | 1 | Interrupt request of external interrupt ch.19 |

| bit No. | bit | Description |
|---|---|---|
| 10 | 0 | No interrupt request of external interrupt ch.18 |
| | 1 | Interrupt request of external interrupt ch.18 |
| 9 | 0 | No interrupt request of external interrupt ch.17 |
| | 1 | Interrupt request of external interrupt ch.17 |
| 8 | 0 | No interrupt request of external interrupt ch.16 |
| | 1 | Interrupt request of external interrupt ch.16 |
| 7 | 0 | No interrupt request of external interrupt ch.15 |
| | 1 | Interrupt request of external interrupt ch.15 |
| 6 | 0 | No interrupt request of external interrupt ch.14 |
| | 1 | Interrupt request of external interrupt ch.14 |
| 5 | 0 | No interrupt request of external interrupt ch.13 |
| | 1 | Interrupt request of external interrupt ch.13 |
| 4 | 0 | No interrupt request of external interrupt ch.12 |
| | 1 | Interrupt request of external interrupt ch.12 |
| 3 | 0 | No interrupt request of external interrupt ch.11 |
| | 1 | Interrupt request of external interrupt ch.11 |
| 2 | 0 | No interrupt request of external interrupt ch.10 |
| | 1 | Interrupt request of external interrupt ch.10 |
| 1 | 0 | No interrupt request of external interrupt ch.9 |
| | 1 | Interrupt request of external interrupt ch.9 |
| 0 | 0 | No interrupt request of external interrupt ch.8 |
| | 1 | Interrupt request of external interrupt ch.8 |

## 2.9 IRQ06 Batch Read Register (IRQ06MON)

The IRQ06 Batch Read Register (IRQ06MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 22.

IRQ06MON indicates the status of the interrupt requests of the QPRC and dual timer.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | QUD2INT | | | |
| Attribute | R | | | | | | | | | | | | R | | | |
| Initial value | 0x000 | | | | | | | | | | | | 0000 | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | QUD2INT | | QUD1INT | | | | | | QUD0INT | | | | | | TIMINT | |
| Attribute | R | | R | | | | | | R | | | | | | R | |
| Initial value | 0x0000 | | 000000 | | | | | | 000000 | | | | | | 00 | |

### Register functions

**[bit31:20] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit19:14] QUD2INT:**

| bit No. | bit | Description |
|---|---|---|
| 19 | 0 | No PC match & RC match interrupt request of QPRC ch.2 |
| | 1 | PC match & RC match interrupt request of QPRC ch.2 |
| 18 | 0 | No interrupt request detected RC out of range on QPRC ch.2 |
| | 1 | Interrupt request detected RC out of range on QPRC ch.2 |
| 17 | 0 | No PC count invert interrupt request of QPRC ch.2 |
| | 1 | PC count invert interrupt request of QPRC ch.2 |
| 16 | 0 | No overflow/underflow/zero index interrupt request of QPRC ch.2 |
| | 1 | Overflow/underflow/zero index interrupt request of QPRC ch.2 |
| 15 | 0 | No PC&RC match interrupt request of QPRC ch.2 |
| | 1 | PC&RC match interrupt request of QPRC ch.2 |
| 14 | 0 | No PC match interrupt request of QPRC ch.2 |
| | 1 | PC match interrupt request of QPRC ch.2 |

**[bit13:8] QUD1INT:**

| bit No. | bit | Description |
|---|---|---|
| 13 | 0 | No PC match & RC match interrupt request of QPRC ch.1 |
| | 1 | PC match & RC match interrupt request of QPRC ch.1 |
| 12 | 0 | No interrupt request detected RC out of range on QPRC ch.1 |
| | 1 | Interrupt request detected RC out of range on QPRC ch.1 |
| 11 | 0 | No PC count invert interrupt request of QPRC ch.1 |
| | 1 | PC count invert interrupt request of QPRC ch.1 |
| 10 | 0 | No overflow/underflow/zero index interrupt request of QPRC ch.1 |
| | 1 | Overflow/underflow/zero index interrupt request of QPRC ch.1 |
| 9 | 0 | No PC&RC match interrupt request of QPRC ch.1 |
| | 1 | PC&RC match interrupt request of QPRC ch.1 |
| 8 | 0 | No PC match interrupt request of QPRC ch.1 |
| | 1 | PC match interrupt request of QPRC ch.1 |

**[bit7:2] QUD0INT:**

| bit No. | bit | Description |
|---|---|---|
| 7 | 0 | No PC match & RC match interrupt request of QPRC ch.0 |
| | 1 | PC match & RC match interrupt request of QPRC ch.0 |
| 6 | 0 | No interrupt request detected RC out of range on QPRC ch.0 |
| | 1 | Interrupt request detected RC out of range on QPRC ch.0 |
| 5 | 0 | No PC count invert interrupt request of QPRC ch.0 |
| | 1 | PC count invert interrupt request of QPRC ch.0 |
| 4 | 0 | No overflow/underflow/zero index interrupt request of QPRC ch.0 |
| | 1 | Overflow/underflow/zero index interrupt request of QPRC ch.0 |
| 3 | 0 | No PC&RC match interrupt request of QPRC ch.0 |
| | 1 | PC&RC match interrupt request of QPRC ch.0 |
| 2 | 0 | No PC match interrupt request of QPRC ch.0 |
| | 1 | PC match interrupt request of QPRC ch.0 |

**[bit1:0] TIMINT:**

| bit No. | bit | Description |
|---|---|---|
| 1 | 0 | No dual timer TIMINT2 interrupt request |
| | 1 | Dual timer TIMINT2 interrupt request |
| 0 | 0 | No dual timer TIMINT1 interrupt request |
| | 1 | Dual timer TIMINT1 interrupt request |

## 2.10 IRQ07/09/11/13/15/17 Batch Read Register (IRQxxMON)

The IRQ07 Batch Read Register (IRQ07MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 23.
The IRQ09 Batch Read Register (IRQ09MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 25.
The IRQ11 Batch Read Register (IRQ11MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 27.
The IRQ13 Batch Read Register (IRQ13MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 29.
The IRQ15 Batch Read Register (IRQ15MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 31.
The IRQ17 Batch Read Register (IRQ17MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 33.

IRQ07MON indicates the status of the reception interrupt request of MFS ch.0 / ch.8.
IRQ09MON indicates the status of the reception interrupt request of MFS ch.1 / ch.9.
IRQ11MON indicates the status of the reception interrupt request of MFS ch.2 / ch.10.
IRQ13MON indicates the status of the reception interrupt request of MFS ch.3 / ch.11.
IRQ15MON indicates the status of the reception interrupt request of MFS ch.4 / ch.12.
IRQ17MON indicates the status of the reception interrupt request of MFS ch.5 / ch.13.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | MFSINT | |
| Attribute | R | | | | | | | | | | | | | | R | |
| Initial value | 00000000000000 | | | | | | | | | | | | | | 00 | |

### Register functions

**[bit31:2] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit1:0] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 1 | 0 | No reception interrupt request of the corresponding MFS channel (ch.8 to ch.13) |
| | 1 | Reception interrupt request of the corresponding MFS channel (ch.8 to ch.13) |
| 0 | 0 | No reception interrupt request of the corresponding MFS channel (ch.0 to ch.5) |
| | 1 | Reception interrupt request of the corresponding MFS channel (ch.0 to ch.5) |

If the DMA transfer request is selected by the DRQSEL register, the MFSINT bit of the corresponding MFS channel (ch.0 to ch.5) is "0".

## 2.11 IRQ19/21 Batch Read Register (IRQxxMON)

The IRQ19 Batch Read Register (IRQ19MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 35.
The IRQ21 Batch Read Register (IRQ21MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 37.

IRQ19MON indicates the status of the reception interrupt request of MFS ch.6 / ch.14, and the status of the interrupt request of DMAC ch.0.
IRQ21MON indicates the status of the reception interrupt request of MFS ch.7 / ch.15, and the status of the interrupt request of DMAC ch.2.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | DMAINT | Reserved | | MFSINT | |
| Attribute | R | | | | | | | | | | | R | R | | R | |
| Initial value | 00000000000 | | | | | | | | | | | 0 | 00 | | 00 | |

### Register functions

**[bit31:5] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit4] DMAINT:**

| bit | Description |
|---|---|
| 1 | No interrupt request of DMAC ch.0 / ch.2 |
| | Interrupt request of DMAC ch.0 / ch.2 |

**[bit3:2] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit1:0] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 1 | 0 | No reception interrupt request of the corresponding MFS channel (ch.14, ch.15) |
| | 1 | Reception interrupt request of the corresponding MFS channel (ch.14, ch.15) |
| 0 | 0 | No reception interrupt request of the corresponding MFS channel (ch.6, ch.7) |
| | 1 | Reception interrupt request of the corresponding MFS channel (ch.6, ch.7) |

If the DMA transfer request is selected by the DRQSEL register, the MFSINT bit of the corresponding MFS channel (ch.6, ch.7) is "0".

## 2.12 IRQ08/10/12/14/16/18 Batch Read Register (IRQxxMON)

The IRQ08 Batch Read Register (IRQ08MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 24.
The IRQ10 Batch Read Register (IRQ10MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 26.
The IRQ12 Batch Read Register (IRQ12MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 28.
The IRQ14 Batch Read Register (IRQ14MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 30.
The IRQ16 Batch Read Register (IRQ16MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 32.
The IRQ18 Batch Read Register (IRQ18MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 34.

IRQ08MON indicates the status of the transmission interrupt request and the status interrupt request of MFS ch.0 and ch.8.
IRQ10MON indicates the status of the transmission interrupt request and the status interrupt request of MFS ch.1 and ch.9.
IRQ12MON indicates the status of the transmission interrupt request and the status interrupt request of MFS ch.2 and ch.10.
IRQ14MON indicates the status of the transmission interrupt request and the status interrupt request of MFS ch.3 and ch.11.
IRQ16MON indicates the status of the transmission interrupt request and the status interrupt request of MFS ch.4 and ch.12.
IRQ18MON indicates the status of the transmission interrupt request and the status interrupt request of MFS ch.5 and ch.13.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | MFSINT | | | |
| Attribute | R | | | | | | | | | | | | R | | | |
| Initial value | 00000000000000 | | | | | | | | | | | | 0000 | | | |

### Register functions

**[bit31:4] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit3:0] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 3 | 0 | No status interrupt request of the corresponding MFS channel (ch.8 to ch.13) |
| | 1 | Status interrupt request of the corresponding MFS channel (ch.8 to ch.13) |
| 2 | 0 | No transmission interrupt request of the corresponding MFS channel (ch.8 to ch.13) |
| | 1 | Transmission interrupt request of the corresponding MFS channel (ch.8 to ch.13) |
| 1 | 0 | No status interrupt request of the corresponding MFS channel (ch.0 to ch.5) |
| | 1 | Status interrupt request of the corresponding MFS channel (ch.0 to ch.5) |
| 0 | 0 | No transmission interrupt request of the corresponding MFS channel (ch.0 to ch.5) |
| | 1 | Transmission interrupt request of the corresponding MFS channel (ch.0 to ch.5) |

If the DMA transfer request is selected by the DRQSEL register, the MFSINT bit of the corresponding MFS channel (ch.0 to ch.5) is "0".

## 2.13 IRQ20/22 Batch Read Register (IRQxxMON)

The IRQ20 Batch Read Register (IRQ20MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 36.
The IRQ22 Batch Read Register (IRQ22MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 38.

IRQ20MON indicates the status of the transmission interrupt request, the status of the status interrupt request of MFS ch.6 / ch.14, and the status of the interrupt request of DMAC ch.1.
IRQ22MON indicates the status of the transmission interrupt request, the status of the status interrupt request of MFS ch.7 / ch.15, and the status of the interrupt request of DMAC ch.3.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | DMAINT | MFSINT | | | |
| Attribute | R | | | | | | | | | | | R | R | | | |
| Initial value | 00000000000 | | | | | | | | | | | 0 | 0000 | | | |

### Register functions

**[bit31:5] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit4] DMAINT:**

| bit | Description |
|---|---|
| 1 | No interrupt request of DMAC ch.1 / ch.3 |
| | Interrupt request of DMAC ch.1 / ch.3 |

**[bit3:0] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 3 | 0 | No status interrupt request of the corresponding MFS channel (ch.14, ch.15) |
| | 1 | Status interrupt request of the corresponding MFS channel (ch.14, ch.15) |
| 2 | 0 | No transmission interrupt request of the corresponding MFS channel (ch.14, ch.15) |
| | 1 | Transmission interrupt request of the corresponding MFS channel (ch.14, ch.15) |
| 1 | 0 | No status interrupt request of the corresponding MFS channel (ch.6, ch.7) |
| | 1 | Status interrupt request of the corresponding MFS channel (ch.6, ch.7) |
| 0 | 0 | No transmission interrupt request of the corresponding MFS channel (ch.6, ch.7) |
| | 1 | Transmission interrupt request of the corresponding MFS channel (ch.6, ch.7) |

If the DMA transfer request is selected by the DRQSEL register, the MFSINT bit of the corresponding MFS channel (ch.6, ch.7) is "0".

## 2.14 IRQ23 Batch Read Register (IRQ23MON)

The IRQ23 Batch Read Register (IRQ23MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 39.

IRQ23MON indicates the status of the interrupt request of the PPG.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | PPGINT | | | | | | | | |
| Attribute | R | | | | | | | R | | | | | | | | |
| Initial value | 0000000 | | | | | | | 000000000 | | | | | | | | |

### Register functions

**[bit31:9] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit8:0] PPGINT:**

| bit No. | bit | Description |
|---|---|---|
| 8 | 0 | No interrupt request of PPG ch.20 |
| | 1 | Interrupt request of PPG ch.20 |
| 7 | 0 | No interrupt request of PPG ch.18 |
| | 1 | Interrupt request of PPG ch.18 |
| 6 | 0 | No interrupt request of PPG ch.16 |
| | 1 | Interrupt request of PPG ch.16 |
| 5 | 0 | No interrupt request of PPG ch.12 |
| | 1 | Interrupt request of PPG ch.12 |
| 4 | 0 | No interrupt request of PPG ch.10 |
| | 1 | Interrupt request of PPG ch.10 |
| 3 | 0 | No interrupt request of PPG ch.8 |
| | 1 | Interrupt request of PPG ch.8 |
| 2 | 0 | No interrupt request of PPG ch.4 |
| | 1 | Interrupt request of PPG ch.4 |
| 1 | 0 | No interrupt request of PPG ch.2 |
| | 1 | Interrupt request of PPG ch.2 |
| 0 | 0 | No interrupt request of PPG ch.0 |
| | 1 | Interrupt request of PPG ch.0 |

## 2.15 IRQ24 Batch Read Register (IRQ24MON)

The IRQ24 Batch Read Register (IRQ24MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 40.

IRQ24MON indicates the status of the interrupt requests of the RTC, watch counter, main PLL oscillation, sub oscillation and main clock oscillation.

### Register configuration

| bit | 31 | | | | | | | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | R | | | | | | | |
| Initial value | 0x000000 | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | RTCINT | WCINT | Reserved | MPLLINT | SOSCINT | MOSCINT |
| Attribute | R | | R | R | R | R | R | R |
| Initial value | 00 | | 0 | 0 | 0 | 0 | 0 | 0 |

### Register functions

**[bit31:6] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit5] RTCINT:**

| bit | Description |
|---|---|
| 0 | No RTC interrupt request |
| 1 | RTC interrupt request |

**[bit4] WCINT:**

| bit | Description |
|---|---|
| 0 | No watch counter interrupt request |
| 1 | Watch counter interrupt request |

**[bit3] Reserved: Reserved bit**
A reserved bit reads "0".

**[bit2] MPLLINT:**

| bit | Description |
|---|---|
| 0 | No stabilization wait completion interrupt request for main PLL oscillation |
| 1 | Stabilization wait completion interrupt request for main PLL oscillation |

**[bit1] SOSCINT:**

| bit | Description |
|---|---|
| 0 | No stabilization wait completion interrupt request for sub-clock oscillation |
| 1 | Stabilization wait completion interrupt request for sub-clock oscillation |

**[bit0] MOSCINT:**

| bit | Description |
|---|---|
| 0 | No stabilization wait completion interrupt request for main clock oscillation |
| 1 | Stabilization wait completion interrupt request for main clock oscillation |

## 2.16 IRQ25/26 Batch Read Register (IRQxxMON)

The IRQ25 Batch Read Register (IRQ25MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 41.
The IRQ26 Batch Read Register (IRQ26MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 42.

IRQ25MON indicates the status of the interrupt request of A/D converter unit 0.
IRQ26MON indicates the status of the interrupt request of A/D converter unit 1.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | ADCINT | | | | |
| Attribute | R | | | | | | | | | | | R | | | | |
| Initial value | 00000000000 | | | | | | | | | | | 00000 | | | | |

### Register functions

**[bit31:5] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit4:0] ADCINT:**

| bit No. | bit | Description |
|---|---|---|
| 4 | 0 | No range comparison result interrupt request in the corresponding A/D converter unit |
| | 1 | Range comparison result interrupt request in the corresponding A/D converter unit |
| 3 | 0 | No conversion result comparison interrupt request in the corresponding A/D converter unit |
| | 1 | Conversion result comparison interrupt request in the corresponding A/D converter unit |
| 2 | 0 | No FIFO overrun interrupt request in the corresponding A/D converter unit |
| | 1 | FIFO overrun interrupt request in the corresponding A/D converter unit |
| 1 | 0 | No scan conversion interrupt request in the corresponding A/D converter unit |
| | 1 | Scan conversion interrupt request in the corresponding A/D converter unit |
| 0 | 0 | No priority conversion interrupt request in the corresponding A/D converter unit |
| | 1 | Priority conversion interrupt request in the corresponding A/D converter unit |

If the DMA transfer request is selected by the DRQSEL register, the corresponding ADCINT bit is "0".

## 2.17 IRQ27 Batch Read Register (IRQ27MON)

The IRQ27 Batch Read Register (IRQ27MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 43.

IRQ27MON indicates the status of the interrupt requests of A/D converter unit 2 and LCD controller.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | LCDCINT | ADCINT | | | | |
| Attribute | R | | | | | | | | | | R | R | | | | |
| Initial value | 0000000000 | | | | | | | | | | 0 | 00000 | | | | |

### Register functions

**[bit31:6] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit5] LCDCINT:**

| bit | Description |
|---|---|
| 0 | No interrupt request for LCD controller |
| 1 | Interrupt request for LCD controller |

**[bit4:0] ADCINT:**

| bit No. | bit | Description |
|---|---|---|
| 4 | 0 | No range comparison result interrupt request in the corresponding A/D converter unit 2 |
| | 1 | Range comparison result interrupt request in the corresponding A/D converter unit 2 |
| 3 | 0 | No conversion result comparison interrupt request in the A/D converter unit 2 |
| | 1 | Conversion result comparison interrupt request in the A/D converter unit 2 |
| 2 | 0 | No FIFO overrun interrupt request in the A/D converter unit 2 |
| | 1 | FIFO overrun interrupt request in the A/D converter unit 2 |
| 1 | 0 | No scan conversion interrupt request in the A/D converter unit 2 |
| | 1 | Scan conversion interrupt request in the A/D converter unit 2 |
| 0 | 0 | No priority conversion interrupt request in the A/D converter unit 2 |
| | 1 | Priority conversion interrupt request in the A/D converter unit 2 |

If the DMA transfer request is selected by the DRQSEL register, the corresponding ADCINT bit is "0".

## 2.18 IRQ28 Batch Read Register (IRQ28MON)

The IRQ28 Batch Read Register (IRQ28MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 44.

IRQ28MON indicates the status of the interrupt requests of MFT unit 0, MFT unit 1 and MFT unit 2.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | FRT2INT | |
| Attribute | R | | | | | | | | | | | | | | R | |
| Initial value | 00000000000000 | | | | | | | | | | | | | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | FRT2INT | | | | FRT1INT | | | | | | FRT0INT | | | | | |
| Attribute | R | | | | R | | | | | | R | | | | | |
| Initial value | 0000 | | | | 000000 | | | | | | 000000 | | | | | |

### Register functions

**[bit31:18] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit17:12] FRT2INT:**

| bit No. | bit | Description |
|---|---|---|
| 17 | 0 | No zero detection interrupt request of the free run timer ch.2 in the MFT unit 2 |
| | 1 | Zero detection interrupt request of the free run timer ch.2 in the MFT unit 2 |
| 16 | 0 | No zero detection interrupt request of the free run timer ch.1 in the MFT unit 2 |
| | 1 | Zero detection interrupt request of the free run timer ch.1 in the MFT unit 2 |
| 15 | 0 | No zero detection interrupt request of the free run timer ch.0 in the MFT unit 2 |
| | 1 | Zero detection interrupt request of the free run timer ch.0 in the MFT unit 2 |
| 14 | 0 | No peak value detection interrupt request of the free run timer ch.2 in the MFT unit 2 |
| | 1 | Peak value detection interrupt request of the free run timer ch.2 in the MFT unit 2 |
| 13 | 0 | No peak value detection interrupt request of the free run timer ch.1 in the MFT unit 2 |
| | 1 | Peak value detection interrupt request of the free run timer ch.1 in the MFT unit 2 |
| 12 | 0 | No peak value detection interrupt request of the free run timer ch.0 in the MFT unit 2 |
| | 1 | Peak value detection interrupt request of the free run timer ch.0 in the MFT unit 2 |

**[bit11:6] FRT1INT:**

| bit No. | bit | Description |
|---|---|---|
| 11 | 0 | No zero detection interrupt request of the free run timer ch.2 in the MFT unit 1 |
| | 1 | Zero detection interrupt request of the free run timer ch.2 in the MFT unit 1 |
| 10 | 0 | No zero detection interrupt request of the free run timer ch.1 in the MFT unit 1 |
| | 1 | Zero detection interrupt request of the free run timer ch.1 in the MFT unit 1 |
| 9 | 0 | No zero detection interrupt request of the free run timer ch.0 in the MFT unit 1 |
| | 1 | Zero detection interrupt request of the free run timer ch.0 in the MFT unit 1 |
| 8 | 0 | No peak value detection interrupt request of the free run timer ch.2 in the MFT unit 1 |
| | 1 | Peak value detection interrupt request of the free run timer ch.2 in the MFT unit 1 |
| 7 | 0 | No peak value detection interrupt request of the free run timer ch.1 in the MFT unit 1 |
| | 1 | Peak value detection interrupt request of the free run timer ch.1 in the MFT unit 1 |
| 6 | 0 | No peak value detection interrupt request of the free run timer ch.0 in the MFT unit 1 |
| | 1 | Peak value detection interrupt request of the free run timer ch.0 in the MFT unit 1 |

**[bit5:0] FRT0INT:**

| bit No. | bit | Description |
|---|---|---|
| 5 | 0 | No zero detection interrupt request of the free run timer ch.2 in the MFT unit 0 |
| | 1 | Zero detection interrupt request of the free run timer ch.2 in the MFT unit 0 |
| 4 | 0 | No zero detection interrupt request of the free run timer ch.1 in the MFT unit 0 |
| | 1 | Zero detection interrupt request of the free run timer ch.1 in the MFT unit 0 |
| 3 | 0 | No zero detection interrupt request of the free run timer ch.0 in the MFT unit 0 |
| | 1 | Zero detection interrupt request of the free run timer ch.0 in the MFT unit 0 |
| 2 | 0 | No peak value detection interrupt request of the free run timer ch.2 in the MFT unit 0 |
| | 1 | Peak value detection interrupt request of the free run timer ch.2 in the MFT unit 0 |
| 1 | 0 | No peak value detection interrupt request of the free run timer ch.1 in the MFT unit 0 |
| | 1 | Peak value detection interrupt request of the free run timer ch.1 in the MFT unit 0 |
| 0 | 0 | No peak value detection interrupt request of the free run timer ch.0 in the MFT unit 0 |
| | 1 | Peak value detection interrupt request of the free run timer ch.0 in the MFT unit 0 |

## 2.19 IRQ29 Batch Read Register (IRQ29MON)

The IRQ29 Batch Read Register (IRQ29MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 45.

IRQ29MON indicates the status of the interrupt requests of MFT unit 0, MFT unit 1 and MFT unit 2.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | ICU2INT | | | | ICU1INT | | | | ICU0INT | | | |
| Attribute | R | | | | R | | | | R | | | | R | | | |
| Initial value | 0000 | | | | 0000 | | | | 0000 | | | | 0000 | | | |

### Register functions

**[bit31:12] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit11:8] ICU2INT:**

| bit No. | bit | Description |
|---|---|---|
| 11 | 0 | No interrupt request of the input capture ch.3 in the MFT unit 2 |
| | 1 | Interrupt request of the input capture ch.3 in the MFT unit 2 |
| 10 | 0 | No interrupt request of the input capture ch.2 in the MFT unit 2 |
| | 1 | Interrupt request of the input capture ch.2 in the MFT unit 2 |
| 9 | 0 | No interrupt request of the input capture ch.1 in the MFT unit 2 |
| | 1 | Interrupt request of the input capture ch.1 in the MFT unit 2 |
| 8 | 0 | No interrupt request of the input capture ch.0 in the MFT unit 2 |
| | 1 | Interrupt request of the input capture ch.0 in the MFT unit 2 |

**[bit7:4] ICU1INT:**

| bit No. | bit | Description |
|---|---|---|
| 7 | 0 | No interrupt request of the input capture ch.3 in the MFT unit 1 |
| | 1 | Interrupt request of the input capture ch.3 in the MFT unit 1 |
| 6 | 0 | No interrupt request of the input capture ch.2 in the MFT unit 1 |
| | 1 | Interrupt request of the input capture ch.2 in the MFT unit 1 |
| 5 | 0 | No interrupt request of the input capture ch.1 in the MFT unit 1 |
| | 1 | Interrupt request of the input capture ch.1 in the MFT unit 1 |
| 4 | 0 | No interrupt request of the input capture ch.0 in the MFT unit 1 |
| | 1 | Interrupt request of the input capture ch.0 in the MFT unit 1 |

**[bit3:0] ICU0INT:**

| bit No. | bit | Description |
|---|---|---|
| 3 | 0 | No interrupt request of the input capture ch.3 in the MFT unit 0 |
| | 1 | Interrupt request of the input capture ch.3 in the MFT unit 0 |
| 2 | 0 | No interrupt request of the input capture ch.2 in the MFT unit 0 |
| | 1 | Interrupt request of the input capture ch.2 in the MFT unit 0 |
| 1 | 0 | No interrupt request of the input capture ch.1 in the MFT unit 0 |
| | 1 | Interrupt request of the input capture ch.1 in the MFT unit 0 |
| 0 | 0 | No interrupt request of the input capture ch.0 in the MFT unit 0 |
| | 1 | Interrupt request of the input capture ch.0 in the MFT unit 0 |

## 2.20 IRQ30 Batch Read Register (IRQ30MON)

The IRQ30 Batch Read Register (IRQ30MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 46.

IRQ30MON indicates the status of the interrupt requests of MFT unit 0, MFT unit 1 and MFT unit 2.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | OCU2INT | |
| Attribute | R | | | | | | | | | | | | | | R | |
| Initial value | 00000000000000 | | | | | | | | | | | | | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | OCU2INT | | | | OCU1INT | | | | | | OCU0INT | | | | | |
| Attribute | R | | | | R | | | | | | R | | | | | |
| Initial value | 0000 | | | | 000000 | | | | | | 000000 | | | | | |

### Register functions

**[bit31:18] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit17:12] OCU2INT:**

| bit No. | bit | Description |
|---|---|---|
| 17 | 0 | No interrupt request of the output compare ch.5 in the MFT unit 2 |
| | 1 | Interrupt request of the output compare ch.5 in the MFT unit 2 |
| 16 | 0 | No interrupt request of the output compare ch.4 in the MFT unit 2 |
| | 1 | Interrupt request of the output compare ch.4 in the MFT unit 2 |
| 15 | 0 | No interrupt request of the output compare ch.3 in the MFT unit 2 |
| | 1 | Interrupt request of the output compare ch.3 in the MFT unit 2 |
| 14 | 0 | No interrupt request of the output compare ch.2 in the MFT unit 2 |
| | 1 | Interrupt request of the output compare ch.2 in the MFT unit 2 |
| 13 | 0 | No interrupt request of the output compare ch.1 in the MFT unit 2 |
| | 1 | Interrupt request of the output compare ch.1 in the MFT unit 2 |
| 12 | 0 | No interrupt request of the output compare ch.0 in the MFT unit 2 |
| | 1 | Interrupt request of the output compare ch.0 in the MFT unit 2 |

**[bit11:6] OCU1INT:**

| bit No. | bit | Description |
|---|---|---|
| 11 | 0 | No interrupt request of the output compare ch.5 in the MFT unit 1 |
| | 1 | Interrupt request of the output compare ch.5 in the MFT unit 1 |
| 10 | 0 | No interrupt request of the output compare ch.4 in the MFT unit 1 |
| | 1 | Interrupt request of the output compare ch.4 in the MFT unit 1 |
| 9 | 0 | No interrupt request of the output compare ch.3 in the MFT unit 1 |
| | 1 | Interrupt request of the output compare ch.3 in the MFT unit 1 |
| 8 | 0 | No interrupt request of the output compare ch.2 in the MFT unit 1 |
| | 1 | Interrupt request of the output compare ch.2 in the MFT unit 1 |
| 7 | 0 | No interrupt request of the output compare ch.1 in the MFT unit 1 |
| | 1 | Interrupt request of the output compare ch.1 in the MFT unit 1 |
| 6 | 0 | No interrupt request of the output compare ch.0 in the MFT unit 1 |
| | 1 | Interrupt request of the output compare ch.0 in the MFT unit 1 |

**[bit5:0] OCU0INT:**

| bit No. | bit | Description |
|---|---|---|
| 5 | 0 | No interrupt request of the output compare ch.5 in the MFT unit 0 |
| | 1 | Interrupt request of the output compare ch.5 in the MFT unit 0 |
| 4 | 0 | No interrupt request of the output compare ch.4 in the MFT unit 0 |
| | 1 | Interrupt request of the output compare ch.4 in the MFT unit 0 |
| 3 | 0 | No interrupt request of the output compare ch.3 in the MFT unit 0 |
| | 1 | Interrupt request of the output compare ch.3 in the MFT unit 0 |
| 2 | 0 | No interrupt request of the output compare ch.2 in the MFT unit 0 |
| | 1 | Interrupt request of the output compare ch.2 in the MFT unit 0 |
| 1 | 0 | No interrupt request of the output compare ch.1 in the MFT unit 0 |
| | 1 | Interrupt request of the output compare ch.1 in the MFT unit 0 |
| 0 | 0 | No interrupt request of the output compare ch.0 in the MFT unit 0 |
| | 1 | Interrupt request of the output compare ch.0 in the MFT unit 0 |

## 2.21 IRQ31 Batch Read Register (IRQ31MON)

The IRQ31 Batch Read Register (IRQ31MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 47.

IRQ31MON indicates the status of the interrupt requests of the Flash memory and base timer.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | FLASHINT | Reserved | | | | | | | | | | |
| Attribute | R | | | | R | R | | | | | | | | | | |
| Initial value | 0000 | | | | 0 | 00000000000 | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | BTINT | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

### Register functions

**[bit31:28] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit27] FLASHINT:**

| bit | Description |
|---|---|
| 0 | No RDY, HANG interrupt request for flash memory |
| 1 | RDY, HANG interrupt request for flash memory |

**[bit26:16] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit15:0] BTINT:**

| bit No. | bit | Description |
|---|---|---|
| 15 | 0 | No IRQ1 interrupt request on the base timer ch.7 |
| | 1 | IRQ1 interrupt request on the base timer ch.7 |
| 14 | 0 | No IRQ0 interrupt request on the base timer ch.7 |
| | 1 | IRQ0 interrupt request on the base timer ch.7 |
| 13 | 0 | No IRQ1 interrupt request on the base timer ch.6 |
| | 1 | IRQ1 interrupt request on the base timer ch.6 |
| 12 | 0 | No IRQ0 interrupt request on the base timer ch.6 |
| | 1 | IRQ0 interrupt request on the base timer ch.6 |
| 11 | 0 | No IRQ1 interrupt request on the base timer ch.5 |
| | 1 | IRQ1 interrupt request on the base timer ch.5 |
| 10 | 0 | No IRQ0 interrupt request on the base timer ch.5 |
| | 1 | IRQ0 interrupt request on the base timer ch.5 |
| 9 | 0 | No IRQ1 interrupt request on the base timer ch.4 |
| | 1 | IRQ1 interrupt request on the base timer ch.4 |
| 8 | 0 | No IRQ0 interrupt request on the base timer ch.4 |
| | 1 | IRQ0 interrupt request on the base timer ch.4 |
| 7 | 0 | No IRQ1 interrupt request on the base timer ch.3 |
| | 1 | IRQ1 interrupt request on the base timer ch.3 |
| 6 | 0 | No IRQ0 interrupt request on the base timer ch.3 |
| | 1 | IRQ0 interrupt request on the base timer ch.3 |
| 5 | 0 | No IRQ1 interrupt request on the base timer ch.2 |
| | 1 | IRQ1 interrupt request on the base timer ch.2 |
| 4 | 0 | No IRQ0 interrupt request on the base timer ch.2 |
| | 1 | IRQ0 interrupt request on the base timer ch.2 |
| 3 | 0 | No IRQ1 interrupt request on the base timer ch.1 |
| | 1 | IRQ1 interrupt request on the base timer ch.1 |
| 2 | 0 | No IRQ0 interrupt request on the base timer ch.1 |
| | 1 | IRQ0 interrupt request on the base timer ch.1 |
| 1 | 0 | No IRQ1 interrupt request on the base timer ch.0 |
| | 1 | IRQ1 interrupt request on the base timer ch.0 |
| 0 | 0 | No IRQ0 interrupt request on the base timer ch.0 |
| | 1 | IRQ0 interrupt request on the base timer ch.0 |

If the DMA transfer request is selected by the DRQSEL register, the corresponding BTINT bit is "0".

As shown in the Table 2-1, base timer interrupt factors IRQ0 and IRQ1 differ depending on the base timer function to be used.

**Table 2-1 Interrupt factors for each function of the base timer**

| Function | Interrupt Factor IRQ0 | Interrupt Factor IRQ1 |
|---|---|---|
| 16-bit PWM timer | Underflow detection/<br>duty match detection | Timer start trigger detection |
| 16-bit PPG timer | Underflow detection | Timer start trigger detection |
| 16/32-bit reload timer | Underflow detection | Timer start trigger detection |
| 16/32-bit PWC timer | Overflow detection | Measurement finished detection |

# 3. Usage Precautions

Be careful with the following points when using the interrupt controller.

- The interrupt request signals from each of the peripheral resources are notified by level. When exiting from interrupt processing, always clear the interrupt request.
- The NMIX pin is allocated shared with a general-purpose port. The initial value after reset is released is configured as a general-purpose port, and the NMI input is masked. In order to use the NMI, enable NMI by configuring the port setting. See the chapter of "External Interrupt and NMI Control Unit" for details.
- See the chapters for each of the macros for the correlation between the specific event detection registers and interrupt enable registers in each peripheral resource.

# CHAPTER7-3: Interrupts (B)

**This chapter explains Exception and Interrupt Factor Vectors and Registers at IRQCMODE=1.**



CODE: 9AFIRQC_B-E01.0

# 1. Exception and Interrupt Factor Vectors

This section shows a vector table of the exceptions and interrupts input to the NVIC.

**Table 1-1 Exception and interrupt factor vectors**

| Vector No. | IRQ No. | Exception and Interrupt Factor | Vector Offset |
|---|---|---|---|
| 0 | - | Stack pointer initial value | 0x00 |
| 1 | - | Reset | 0x04 |
| 2 | - | Non-Maskable Interrupt (NMI) / Hardware Watchdog Timer | 0x08 |
| 3 | - | Hard Fault | 0x0C |
| 4 | - | Memory Management | 0x10 |
| 5 | - | Bus Fault | 0x14 |
| 6 | - | Usage Fault | 0x18 |
| 7-10 | - | Reserved | 0x1C - 0x2B |
| 11 | - | SVCall (Supervisor Call) | 0x2C |
| 12 | - | Debug Monitor | 0x30 |
| 13 | - | Reserved | 0x34 |
| 14 | - | PendSV | 0x38 |
| 15 | - | SysTick | 0x3C |
| 16 | 0 | Anomalous Frequency Detection by Clock Supervisor (FCS) | 0x40 |
| 17 | 1 | Software Watchdog Timer | 0x44 |
| 18 | 2 | Low Voltage Detector (LVD) | 0x48 |
| 19 | 3 | Interrupt source selected in the INTSEL0 bits in the RCINTSEL0 Register | 0x4C |
| 20 | 4 | Interrupt source selected in the INTSEL1 bits in the RCINTSEL0 Register | 0x50 |
| 21 | 5 | Interrupt source selected in the INTSEL2 bits in the RCINTSEL0 Register | 0x54 |
| 22 | 6 | Interrupt source selected in the INTSEL3 bits in the RCINTSEL0 Register | 0x58 |
| 23 | 7 | Interrupt source selected in the INTSEL0 bits in the RCINTSEL1 Register | 0x5C |
| 24 | 8 | Interrupt source selected in the INTSEL1 bits in the RCINTSEL1 Register | 0x60 |
| 25 | 9 | Interrupt source selected in the INTSEL2 bits in the RCINTSEL1 Register | 0x64 |
| 26 | 10 | Interrupt source selected in the INTSEL3 bits in the RCINTSEL1 Register | 0x68 |
| 27 | 11 | MFT unit 0 Wave Form Generator / DTIF(Motor Emergency Stop) / Reception Interrupt Request, Transmission Interrupt Request and Status Interrupt Request of MFS ch.8 | 0x6C |
| 28 | 12 | External Pin Interrupt Request ch.0 to ch.7 | 0x70 |
| 29 | 13 | External Pin Interrupt Request ch.8 to ch.31 | 0x74 |
| 30 | 14 | Dual Timer / Quad Counter (QPRC) ch.0 | 0x78 |
| 31 | 15 | Reception Interrupt Request, Transmission Interrupt Request and Status Interrupt Request of MFS ch.0 | 0x7C |
| 32 | 16 | Reception Interrupt Request, Transmission Interrupt Request and Status Interrupt Request of MFS ch.1 | 0x80 |
| 33 | 17 | Reception Interrupt Request, Transmission Interrupt Request and Status Interrupt Request of MFS ch.2 | 0x84 |
| 34 | 18 | Reception Interrupt Request, Transmission Interrupt Request and Status Interrupt Request of MFS ch.3 | 0x88 |
| 35 | 19 | Reception Interrupt Request of MFS ch.4 | 0x8C |
| 36 | 20 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.4 | 0x90 |
| 37 | 21 | Reception Interrupt Request of MFS ch.5 | 0x94 |
| 38 | 22 | Transmission Interrupt Request and Status Interrupt Request of MFS ch.5 | 0x98 |
| 39 | 23 | PPG ch.0/2/4/8/10/12/16/18/20 | 0x9C |

| Vector No. | IRQ No. | Exception and Interrupt Factor | Vector Offset |
|---|---|---|---|
| 40 | 24 | External Main OSC / External Sub OSC / Main PLL / Watch Counter/Real Time Counter | 0xA0 |
| 41 | 25 | A/D Converter unit 0 / Reception Interrupt Request, Transmission Interrupt Request and Status Interrupt Request of MFS ch.9 | 0xA4 |
| 42 | 26 | A/D Converter unit 1 / Reception Interrupt Request, Transmission Interrupt Request and Status Interrupt Request of MFS ch.10 | 0xA8 |
| 43 | 27 | A/D Converter unit 2 / LCD Controller / Reception Interrupt Request, Transmission Interrupt Request and Status Interrupt Request of MFS ch.11 | 0xAC |
| 44 | 28 | MFT unit 0, unit 1, unit 2 Free-run Timer | 0xB0 |
| 45 | 29 | MFT unit 0, unit 1, unit 2 Input Capture | 0xB4 |
| 46 | 30 | MFT unit 0, unit 1, unit 2 Output Compare/ DMAC ch.0 to ch.7 | 0xB8 |
| 47 | 31 | Base Timer ch.0 to ch.7 / Flash RDY interrupt / Flash HANG interrupt | 0xBC |

The priorities of the exceptions for vectors No. 4 to No. 15 can be configured using the System Handler Priority Registers (address 0xE000ED18, 0xE000ED1C, 0xE000ED20) built into the NVIC. The priorities of the peripheral interrupts for vectors No. 16 and after can be configured using the IRQ Priority Registers (address 0xE000E400 to 0xE000E42C) built into the NVIC.

The sources of the interrupts for vectors No. 2 and No. 16 to No. 47 can be checked using the batch read register. See "Chapter 5: Nested Vectored Interrupt Controller" in the "Cortex-M0+ Technical Reference Manual" for details on the other exceptions and interrupts.

Furthermore, for the interrupts of vectors No. 2 and No. 16 to No. 47, the sources that are batch read may be a signal that multiple interrupt factors are logical OR'ed within each of the peripheral macros. See the descriptions of each of the peripheral resource interrupts for details.

# 2. Registers

This section describes the DMA transfer request selection register and the interrupt request batch read registers.

### List of DMA transfer request selection register and interrupt request batch read registers

**Table 2-1 List of DMA transfer request selection register and interrupt request batch read registers**

| Abbreviation | Register Name | Reference |
|---|---|---|
| DRQSEL | DMA Request Selection Register | 2.1 |
| EXC02MON | EXC02 Batch Read Register | 2.2 |
| IRQ00MON | IRQ00 Batch Read Register | 2.3 |
| IRQ01MON | IRQ01 Batch Read Register | 2.4 |
| IRQ02MON | IRQ02 Batch Read Register | 2.5 |
| IRQ03MON | IRQ03 Batch Read Register | 2.6 |
| IRQ04MON | IRQ04 Batch Read Register | |
| IRQ05MON | IRQ05 Batch Read Register | |
| IRQ06MON | IRQ06 Batch Read Register | |
| IRQ07MON | IRQ07 Batch Read Register | |
| IRQ08MON | IRQ08 Batch Read Register | |
| IRQ09MON | IRQ09 Batch Read Register | |
| IRQ10MON | IRQ10 Batch Read Register | |
| IRQ11MON | IRQ11 Batch Read Register | 2.7 |
| IRQ12MON | IRQ12 Batch Read Register | 2.8 |
| IRQ13MON | IRQ13 Batch Read Register | 2.9 |
| IRQ14MON | IRQ14 Batch Read Register | 2.10 |
| IRQ15MON | IRQ15 Batch Read Register | 2.11 |
| IRQ16MON | IRQ16 Batch Read Register | |
| IRQ17MON | IRQ17 Batch Read Register | |
| IRQ18MON | IRQ18 Batch Read Register | |
| IRQ19MON | IRQ19 Batch Read Register | 2.12 |
| IRQ20MON | IRQ20 Batch Read Register | 2.13 |
| IRQ21MON | IRQ21 Batch Read Register | 2.12 |
| IRQ22MON | IRQ22 Batch Read Register | 2.13 |
| IRQ23MON | IRQ23 Batch Read Register | 2.14 |
| IRQ24MON | IRQ24 Batch Read Register | 2.15 |
| IRQ25MON | IRQ25 Batch Read Register | 2.16 |
| IRQ26MON | IRQ26 Batch Read Register | |
| IRQ27MON | IRQ27 Batch Read Register | 2.17 |
| IRQ28MON | IRQ28 Batch Read Register | 2.18 |
| IRQ29MON | IRQ29 Batch Read Register | |
| IRQ30MON | IRQ30 Batch Read Register | 2.19 |
| IRQ31MON | IRQ31 Batch Read Register | 2.20 |
| IRQCMODE | Interrupt Factor Vector Relocate Setting Register | 2.21 |
| RCINTSEL0 | Interrupt Factor Selection Register 0 | 2.22 |
| RCINTSEL1 | Interrupt Factor Selection Register 1 | 2.23 |

See "Chapter 5: Nested Vectored Interrupt Controller" in the "Cortex-M0+ Technical Reference Manual" for details on the registers in the NVIC.

## 2.1 DMA Request Selection Register (DRQSEL)

The DMA Request Selection Register (DRQSEL) selects whether interrupt signals that can start DMA transfers are output as interrupt requests to the CPU or output as transfer requests to the DMAC.

### Register configuration

| bit | 31 | 5 4 3 2 1 0 |
|---|---|---|
| Field | DRQSEL[31:5] | Reserved |
| Attribute | R/W | R |
| Initial value | 00000000000000000000000000 | 00000 |

### Register functions

**[bit31:0] DRQSEL :**

| bit No. | bit | Description |
|---|---|---|
| 31 | 0 | The interrupt signal of the external interrupt ch.3 is output as an interrupt request to the CPU. |
| | 1 | The interrupt signal of the external interrupt ch.3 is output as a transfer request to the DMAC. |
| 30 | 0 | The interrupt signal of the external interrupt ch.2 is output as an interrupt request to the CPU. |
| | 1 | The interrupt signal of the external interrupt ch.2 is output as a transfer request to the DMAC. |
| 29 | 0 | The interrupt signal of the external interrupt ch.1 is output as an interrupt request to the CPU. |
| | 1 | The interrupt signal of the external interrupt ch.1 is output as a transfer request to the DMAC. |
| 28 | 0 | The interrupt signal of the external interrupt ch.0 is output as an interrupt request to the CPU. |
| | 1 | The interrupt signal of the external interrupt ch.0 is output as a transfer request to the DMAC. |
| 27 | 0 | The transmission interrupt signal of the MFS ch.7 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.7 is output as a transfer request to the DMAC. |
| 26 | 0 | The reception interrupt signal of the MFS ch.7 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.7 is output as a transfer request to the DMAC. |
| 25 | 0 | The transmission interrupt signal of the MFS ch.6 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.6 is output as a transfer request to the DMAC. |
| 24 | 0 | The reception interrupt signal of the MFS ch.6 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.6 is output as a transfer request to the DMAC. |
| 23 | 0 | The transmission interrupt signal of the MFS ch.5 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.5 is output as a transfer request to the DMAC. |
| 22 | 0 | The reception interrupt signal of the MFS ch.5 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.5 is output as a transfer request to the DMAC. |
| 21 | 0 | The transmission interrupt signal of the MFS ch.4 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.4 is output as a transfer request to the DMAC. |
| 20 | 0 | The reception interrupt signal of the MFS ch.4 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.4 is output as a transfer request to the DMAC. |
| 19 | 0 | The transmission interrupt signal of the MFS ch.3 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.3 is output as a transfer request to the DMAC. |
| 18 | 0 | The reception interrupt signal of the MFS ch.3 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.3 is output as a transfer request to the DMAC. |
| 17 | 0 | The transmission interrupt signal of the MFS ch.2 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.2 is output as a transfer request to the DMAC. |
| 16 | 0 | The reception interrupt signal of the MFS ch.2 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.2 is output as a transfer request to the DMAC. |
| 15 | 0 | The transmission interrupt signal of the MFS ch.1 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.1 is output as a transfer request to the DMAC. |
| 14 | 0 | The reception interrupt signal of the MFS ch.1 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.1 is output as a transfer request to the DMAC. |

| bit No. | bit | Description |
|---|---|---|
| 13 | 0 | The transmission interrupt signal of the MFS ch.0 is output as an interrupt request to the CPU. |
| | 1 | The transmission interrupt signal of the MFS ch.0 is output as a transfer request to the DMAC. |
| 12 | 0 | The reception interrupt signal of the MFS ch.0 is output as an interrupt request to the CPU. |
| | 1 | The reception interrupt signal of the MFS ch.0 is output as a transfer request to the DMAC. |
| 11 | 0 | The IRQ0 interrupt signal of the base timer ch.6 is output as an interrupt request to the CPU. |
| | 1 | The IRQ0 interrupt signal of the base timer ch.6 is output as a transfer request to the DMAC (including extension). |
| 10 | 0 | The IRQ0 interrupt signal of the base timer ch.4 is output as an interrupt request to the CPU. |
| | 1 | The IRQ0 interrupt signal of the base timer ch.4 is output as a transfer request to the DMAC (including extension). |
| 9 | 0 | The IRQ0 interrupt signal of the base timer ch.2 is output as an interrupt request to the CPU. |
| | 1 | The IRQ0 interrupt signal of the base timer ch.2 is output as a transfer request to the DMAC. |
| 8 | 0 | The IRQ0 interrupt signal of the base timer ch.0 is output as an interrupt request to the CPU. |
| | 1 | The IRQ0 interrupt signal of the base timer ch.0 is output as a transfer request to the DMAC. |
| 7 | 0 | The scan conversion interrupt signal of the A/D converter unit 2 is output as an interrupt request to the CPU. |
| | 1 | The scan conversion interrupt signal of the A/D converter unit 2 is output as a transfer request to the DMAC. |
| 6 | 0 | The scan conversion interrupt signal of the A/D converter unit 1 is output as an interrupt request to the CPU. |
| | 1 | The scan conversion interrupt signal of the A/D converter unit 1 is output as a transfer request to the DMAC. |
| 5 | 0 | The scan conversion interrupt signal of the A/D converter unit 0 is output as an interrupt request to the CPU. |
| | 1 | The scan conversion interrupt signal of the A/D converter unit 0 is output as a transfer request to the DMAC. |

MFS: Multifunction serial interface

**[bit4:0] Reserved: Reserved bits**

A reserved bit reads "0".

***Notes:***

- *If interrupt signal is selected as a transfer request to DMAC, the read bit value of the appropriate interrupt request batch read register (IRQxxMON, xx = 00 to 31) becomes "0" regardless of the interrupt occurrence.*
- *When changing the DRQSEL settings, clear the interrupt request signals from appropriate peripherals before making the change.*
- *DMA transfers cannot be started from hardware for interrupt signals not specified in the DRQSEL settings.*

## 2.2 EXC02 Batch Read Register (EXC02MON)

The EXC02 Batch Read Register (EXC02MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 2.

EXC02MON indicates the status of the interrupt requests of the hardware watchdog timer and NMIX external pin.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | HWINT | NMI |
| Attribute | R | | | | | | | | | | | | | | R | R |
| Initial value | 00000000000000 | | | | | | | | | | | | | | 0 | 0 |

### Register functions

**[bit31:2] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit1] HWINT:**

| bit | Description |
|---|---|
| 0 | No hardware watchdog timer interrupt request |
| 1 | Hardware watchdog timer interrupt request |

**[bit0] NMI:**

| bit | Description |
|---|---|
| 0 | No NMIX external pin interrupt request |
| 1 | NMIX external pin interrupt request |

## 2.3 IRQ00 Batch Read Register (IRQ00MON)

The IRQ00 Batch Read Register (IRQ00MON) can batch-read the interrupt request allocated to interrupt factor vector No. 16.

IRQ00MON indicates the status of the interrupt request of anomalous frequency detection by the CSV.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | FCSINT |
| Attribute | R | | | | | | | | | | | | | | | R |
| Initial value | 000000000000000 | | | | | | | | | | | | | | | 0 |

### Register functions

**[bit31:1] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit0] FCSINT:**

| bit | Description |
|---|---|
| 0 | No anomalous frequency detection by CSV interrupt request |
| 1 | Anomalous frequency detection by CSV interrupt request |

## 2.4 IRQ01 Batch Read Register (IRQ01MON)

The IRQ01 Batch Read Register (IRQ01MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 17.

IRQ01MON indicates the status of the interrupt request of the software watchdog timer.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | SWWDTINT |
| Attribute | R | | | | | | | | | | | | | | | R |
| Initial value | 000000000000000 | | | | | | | | | | | | | | | 0 |

### Register functions

**[bit31:1] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit0] SWWDTINT:**

| bit | Description |
|---|---|
| 0 | No software watchdog timer interrupt request |
| 1 | Software watchdog timer interrupt request |

## 2.5 IRQ02 Batch Read Register (IRQ02MON)

The IRQ02 Batch Read Register (IRQ02MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 18.

IRQ02MON indicates the status of the interrupt request of low voltage detection (LVD).

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | LVDINT |
| Attribute | R | | | | | | | | | | | | | | | R |
| Initial value | 000000000000000 | | | | | | | | | | | | | | | 0 |

### Register functions

**[bit31:1] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit0] LVDINT:**

| bit | Description |
|---|---|
| 0 | No low voltage detection (LVD) interrupt request |
| 1 | Low voltage detection (LVD) interrupt request |

## 2.6 IRQ03 to IRQ10 Batch Read Register (IRQ03MON to IRQ10MON)

The IRQ03 Batch Read Register (IRQ03MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 19.
The IRQ04 Batch Read Register (IRQ04MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 20.
The IRQ05 Batch Read Register (IRQ05MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 21.
The IRQ06 Batch Read Register (IRQ06MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 22.
The IRQ07 Batch Read Register (IRQ07MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 23.
The IRQ08 Batch Read Register (IRQ08MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 24.
The IRQ09 Batch Read Register (IRQ09MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 25.
The IRQ10 Batch Read Register (IRQ10MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 26.

IRQ03MON indicates the status of the interrupt request selected in the INTSEL0 bits in the RCINTSEL0 Register.
IRQ04MON indicates the status of the interrupt request selected in the INTSEL1 bits in the RCINTSEL0 Register.
IRQ05MON indicates the status of the interrupt request selected in the INTSEL2 bits in the RCINTSEL0 Register.
IRQ06MON indicates the status of the interrupt request selected in the INTSEL3 bits in the RCINTSEL0 Register.
IRQ07MON indicates the status of the interrupt request selected in the INTSEL0 bits in the RCINTSEL1 Register.
IRQ08MON indicates the status of the interrupt request selected in the INTSEL1 bits in the RCINTSEL1 Register.
IRQ09MON indicates the status of the interrupt request selected in the INTSEL2 bits in the RCINTSEL1 Register.
IRQ10MON indicates the status of the interrupt request selected in the INTSEL3 bits in the RCINTSEL1 Register.

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | RCINT |
| Attribute | R | | | | | | | | | | | | | | | R |
| Initial value | 00000000000000 | | | | | | | | | | | | | | | 0 |

**Register functions**

**[bit31:1] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit0] RCINT:**

| bit | Description |
|---|---|
| 0 | No interrupt request selected with relevant RCINTSEL0:INTSELx/RCINTSEL1:INTSELx |
| 1 | Interrupt request selected with relevant RCINTSEL0:NTSELx/RCINTSEL1:INTSELx* |

*: If the base timer is selected as the interrupt factor, this bit is set to "1" by either one of the interrupt factors IRQ0 and IRQ1.

## 2.7 IRQ11 Batch Read Register (IRQxxMON)

The IRQ11 Batch Read Register (IRQ11MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 27.

IRQ11MON indicates the status of the interrupt requests of MFT unit 0 and MFS ch.8.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | MFSINT | | | WAVEINT | | | |
| Attribute | R | | | | | | | | | R | | | R | | | |
| Initial value | 000000000 | | | | | | | | | 000 | | | 0000 | | | |

### Register functions

**[bit31:7] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit6:4] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 6 | 0 | No status interrupt request of the corresponding MFS channel |
| | 1 | Status interrupt request of the corresponding MFS channel |
| 5 | 0 | No transmission interrupt request of the corresponding MFS channel |
| | 1 | Transmission interrupt request of the corresponding MFS channel |
| 4 | 0 | No reception interrupt request of the corresponding MFS channel |
| | 1 | Reception interrupt request of the corresponding MFS channel |

**[bit3:0] WAVEINT:**

| bit No. | bit | Description |
|---|---|---|
| 3 | 0 | No interrupt request of WFG timer 54 in the corresponding MFT unit |
| | 1 | Interrupt request of WFG timer 54 in the corresponding MFT unit |
| 2 | 0 | No interrupt request of WFG timer 32 in the corresponding MFT unit |
| | 1 | Interrupt request of WFG timer 32 in the corresponding MFT unit |
| 1 | 0 | No interrupt request of WFG timer 10 in the corresponding MFT unit |
| | 1 | Interrupt request of WFG timer 10 in the corresponding MFT unit |
| 0 | 0 | No interrupt request of DTIF (Motor emergency stop) in the corresponding MFT unit |
| | 1 | Interrupt request of DTIF (Motor emergency stop) in the corresponding MFT unit |

## 2.8 IRQ12 Batch Read Register (IRQ12MON)

The IRQ12 Batch Read Register (IRQ12MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 28.

IRQ12MON indicates the status of the interrupt requests of external interrupt ch.0 to ch.7.

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | EXTINT | | | | | | | |
| Attribute | R | | | | | | | | R | | | | | | | |
| Initial value | 0x00 | | | | | | | | 0x00 | | | | | | | |

**Register functions**

**[bit31:8] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit7:0] EXTINT:**

| bit No. | bit | Description |
|---|---|---|
| 7 | 0 | No interrupt request of external interrupt ch.7 |
| | 1 | Interrupt request of external interrupt ch.7 |
| 6 | 0 | No interrupt request of external interrupt ch.6 |
| | 1 | Interrupt request of external interrupt ch.6 |
| 5 | 0 | No interrupt request of external interrupt ch.5 |
| | 1 | Interrupt request of external interrupt ch.5 |
| 4 | 0 | No interrupt request of external interrupt ch.4 |
| | 1 | Interrupt request of external interrupt ch.4 |
| 3 | 0 | No interrupt request of external interrupt ch.3 |
| | 1 | Interrupt request of external interrupt ch.3 |
| 2 | 0 | No interrupt request of external interrupt ch.2 |
| | 1 | Interrupt request of external interrupt ch.2 |
| 1 | 0 | No interrupt request of external interrupt ch.1 |
| | 1 | Interrupt request of external interrupt ch.1 |
| 0 | 0 | No interrupt request of external interrupt ch.0 |
| | 1 | Interrupt request of external interrupt ch.0 |

If the DMA transfer request is selected by the DRQSEL register, a corresponding bit in the EXTINT bits becomes "0".

## 2.9 IRQ13 Batch Read Register (IRQ13MON)

The IRQ13 Batch Read Register (IRQ13MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 29.

IRQ13MON indicates the status of the interrupt requests of external interrupt ch.8 to ch.31.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | EXTINT | | | | | | | |
| Attribute | R | | | | | | | | R | | | | | | | |
| Initial value | 0x00 | | | | | | | | 0x00 | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | EXTINT | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

### Register functions

**[bit31:24] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit23:0] EXTINT:**

| bit No. | bit | Description |
|---|---|---|
| 23 | 0 | No interrupt request of external interrupt ch.31 |
| | 1 | Interrupt request of external interrupt ch.31 |
| 22 | 0 | No interrupt request of external interrupt ch.30 |
| | 1 | Interrupt request of external interrupt ch.30 |
| 21 | 0 | No interrupt request of external interrupt ch.29 |
| | 1 | Interrupt request of external interrupt ch.29 |
| 20 | 0 | No interrupt request of external interrupt ch.28 |
| | 1 | Interrupt request of external interrupt ch.28 |
| 19 | 0 | No interrupt request of external interrupt ch.27 |
| | 1 | Interrupt request of external interrupt ch.27 |
| 18 | 0 | No interrupt request of external interrupt ch.26 |
| | 1 | Interrupt request of external interrupt ch.26 |
| 17 | 0 | No interrupt request of external interrupt ch.25 |
| | 1 | Interrupt request of external interrupt ch.25 |
| 16 | 0 | No interrupt request of external interrupt ch.24 |
| | 1 | Interrupt request of external interrupt ch.24 |
| 15 | 0 | No interrupt request of external interrupt ch.23 |
| | 1 | Interrupt request of external interrupt ch.23 |
| 14 | 0 | No interrupt request of external interrupt ch.22 |
| | 1 | Interrupt request of external interrupt ch.22 |
| 13 | 0 | No interrupt request of external interrupt ch.21 |
| | 1 | Interrupt request of external interrupt ch.21 |
| 12 | 0 | No interrupt request of external interrupt ch.20 |
| | 1 | Interrupt request of external interrupt ch.20 |
| 11 | 0 | No interrupt request of external interrupt ch.19 |
| | 1 | Interrupt request of external interrupt ch.19 |
| 10 | 0 | No interrupt request of external interrupt ch.18 |
| | 1 | Interrupt request of external interrupt ch.18 |

| bit No. | bit | Description |
|---|---|---|
| 9 | 0 | No interrupt request of external interrupt ch.17 |
| | 1 | Interrupt request of external interrupt ch.17 |
| 8 | 0 | No interrupt request of external interrupt ch.16 |
| | 1 | Interrupt request of external interrupt ch.16 |
| 7 | 0 | No interrupt request of external interrupt ch.15 |
| | 1 | Interrupt request of external interrupt ch.15 |
| 6 | 0 | No interrupt request of external interrupt ch.14 |
| | 1 | Interrupt request of external interrupt ch.14 |
| 5 | 0 | No interrupt request of external interrupt ch.13 |
| | 1 | Interrupt request of external interrupt ch.13 |
| 4 | 0 | No interrupt request of external interrupt ch.12 |
| | 1 | Interrupt request of external interrupt ch.12 |
| 3 | 0 | No interrupt request of external interrupt ch.11 |
| | 1 | Interrupt request of external interrupt ch.11 |
| 2 | 0 | No interrupt request of external interrupt ch.10 |
| | 1 | Interrupt request of external interrupt ch.10 |
| 1 | 0 | No interrupt request of external interrupt ch.9 |
| | 1 | Interrupt request of external interrupt ch.9 |
| 0 | 0 | No interrupt request of external interrupt ch.8 |
| | 1 | Interrupt request of external interrupt ch.8 |

## 2.10 IRQ14 Batch Read Register (IRQ14MON)

The IRQ14 Batch Read Register (IRQ14MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 30.

IRQ14MON indicates the status of the interrupt requests of dual timer and QPRC.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | QUDINT | | | | | | TIMINT | |
| Attribute | R | | | | | | | | R | | | | | | R | |
| Initial value | 0x00 | | | | | | | | 000000 | | | | | | 00 | |

### Register functions

**[bit31:8] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit7:2] QUDINT:**

| bit No. | bit | Description |
|---|---|---|
| 7 | 0 | No PC match & RC match interrupt request of QPRC ch.0 |
| | 1 | PC match & RC match interrupt request of QPRC ch.0 |
| 6 | 0 | No interrupt request detected RC out of range on QPRC ch.0 |
| | 1 | Interrupt request detected RC out of range on QPRC ch.0 |
| 5 | 0 | No PC count invert interrupt request of QPRC ch.0 |
| | 1 | PC count invert interrupt request of QPRC ch.0 |
| 4 | 0 | No overflow/underflow/zero index interrupt request of QPRC ch.0 |
| | 1 | Overflow/underflow/zero index interrupt request of QPRC ch.0 |
| 3 | 0 | No PC&RC match interrupt request of QPRC ch.0 |
| | 1 | PC&RC match interrupt request of QPRC ch.0 |
| 2 | 0 | No PC match interrupt request of QPRC ch.0 |
| | 1 | PC match interrupt request of QPRC ch.0 |

**[bit1:0] TIMINT:**

| bit No. | bit | Description |
|---|---|---|
| 1 | 0 | No dual timer TIMINT2 interrupt request |
| | 1 | Dual timer TIMINT2 interrupt request |
| 0 | 0 | No dual timer TIMINT1 interrupt request |
| | 1 | Dual timer TIMINT1 interrupt request |

## 2.11 IRQ15 to IRQ18 Batch Read Register (IRQ15MON to IRQ18MON)

The IRQ15 Batch Read Register (IRQ15MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 31
The IRQ16 Batch Read Register (IRQ16MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 32.
The IRQ17 Batch Read Register (IRQ17MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 33.
The IRQ18 Batch Read Register (IRQ18MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 34.

IRQ15MON indicates the status of the reception interrupt request, transmission interrupt request and status request of MFS ch.0.
IRQ16MON indicates the status of the reception interrupt request, transmission interrupt request and status request of MFS ch.1.
IRQ17MON indicates the status of the reception interrupt request, transmission interrupt request and status request of MFS ch.2.
IRQ18MON indicates the status of the reception interrupt request, transmission interrupt request and status request of MFS ch.3.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | MFSINT | | |
| Attribute | R | | | | | | | | | | | | | R | | |
| Initial value | 0000000000000 | | | | | | | | | | | | | 000 | | |

### Register functions

**[bit31:3] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit2:0] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 2 | 0 | No status interrupt request of the corresponding MFS channel |
| | 1 | Status interrupt request of the corresponding MFS channel |
| 1 | 0 | No transmission interrupt request of the corresponding MFS channel |
| | 1 | Transmission interrupt request of the corresponding MFS channel |
| 0 | 0 | No reception interrupt request of the corresponding MFS channel |
| | 1 | Reception interrupt request of the corresponding MFS channel |

If the DMA transfer request is selected by the DRQSEL register, the MFSINT bit of the corresponding MFS channel is "0".

## 2.12 IRQ19/21 Batch Read Register (IRQxxMON)

The IRQ19 Batch Read Register (IRQ19MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 35.
The IRQ21 Batch Read Register (IRQ21MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 37.

IRQ19MON indicates the status of the reception interrupt request of MFS ch.4.
IRQ21MON indicates the status of the reception interrupt request of MFS ch.5.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | MFSINT |
| Attribute | R | | | | | | | | | | | | | | | R |
| Initial value | 000000000000000 | | | | | | | | | | | | | | | 0 |

### Register functions

**[bit31:1] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit0] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 0 | 0 | No reception interrupt request of the corresponding MFS channel |
| | 1 | Reception interrupt request of the corresponding MFS channel |

If the DMA transfer request is selected by the DRQSEL register, the MFSINT bit of the corresponding MFS channel is "0".

## 2.13 IRQ20/22 Batch Read Register (IRQxxMON)

The IRQ20 Batch Read Register (IRQ20MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 36.
The IRQ22 Batch Read Register (IRQ22MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 38.

IRQ20MON indicates the status of the transmission interrupt request and the status of the status interrupt request of MFS ch.4.
IRQ22MON indicates the status of the transmission interrupt request and the status of the status interrupt request of MFS ch.5.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | MFSINT | |
| Attribute | R | | | | | | | | | | | | | | R | |
| Initial value | 00000000000000 | | | | | | | | | | | | | | 00 | |

### Register functions

**[bit31:2] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit1:0] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 1 | 0 | No status interrupt request of the corresponding MFS channel |
| | 1 | Status interrupt request of the corresponding MFS channel |
| 0 | 0 | No transmission interrupt request of the corresponding MFS channel |
| | 1 | Transmission interrupt request of the corresponding MFS channel |

If the DMA transfer request is selected by the DRQSEL register, the MFSINT bit of the corresponding MFS channel is "0".

## 2.14 IRQ23 Batch Read Register (IRQ23MON)

The IRQ23 Batch Read Register (IRQ23MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 39.

IRQ23MON indicates the status of the interrupt request of the PPG.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | PPGINT | | | | | | | | |
| Attribute | R | | | | | | | R | | | | | | | | |
| Initial value | 0000000 | | | | | | | 000000000 | | | | | | | | |

### Register functions

**[bit31:9] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit8:0] PPGINT:**

| bit No. | bit | Description |
|---|---|---|
| 8 | 0 | No interrupt request of PPG ch.20 |
| | 1 | Interrupt request of PPG ch.20 |
| 7 | 0 | No interrupt request of PPG ch.18 |
| | 1 | Interrupt request of PPG ch.18 |
| 6 | 0 | No interrupt request of PPG ch.16 |
| | 1 | Interrupt request of PPG ch.16 |
| 5 | 0 | No interrupt request of PPG ch.12 |
| | 1 | Interrupt request of PPG ch.12 |
| 4 | 0 | No interrupt request of PPG ch.10 |
| | 1 | Interrupt request of PPG ch.10 |
| 3 | 0 | No interrupt request of PPG ch.8 |
| | 1 | Interrupt request of PPG ch.8 |
| 2 | 0 | No interrupt request of PPG ch.4 |
| | 1 | Interrupt request of PPG ch.4 |
| 1 | 0 | No interrupt request of PPG ch.2 |
| | 1 | Interrupt request of PPG ch.2 |
| 0 | 0 | No interrupt request of PPG ch.0 |
| | 1 | Interrupt request of PPG ch.0 |

## 2.15 IRQ24 Batch Read Register (IRQ24MON)

The IRQ24 Batch Read Register (IRQ24MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 40.

5. IRQ24MON indicates the status of the interrupt requests of the RTC, watch counter, main PLL oscillation, sub oscillation and main clock oscillation.

### Register configuration

| bit | 31 | | | | | | | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | R | | | | | | | |
| Initial value | 0x000000 | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | RTCINT | WCINT | Reserved | MPLLINT | SOSCINT | MOSCINT |
| Attribute | R | | R | R | R | R | R | R |
| Initial value | 00 | | 0 | 0 | 0 | 0 | 0 | 0 |

### Register functions

**[bit31:6] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit5] RTCINT:**

| bit | Description |
|---|---|
| 0 | No RTC interrupt request |
| 1 | RTC interrupt request |

**[bit4] WCINT:**

| bit | Description |
|---|---|
| 0 | No watch counter interrupt request |
| 1 | Watch counter interrupt request |

**[bit3] Reserved: Reserved bit**

A reserved bit reads "0".

**[bit2] MPLLINT:**

| bit | Description |
|---|---|
| 0 | No stabilization wait completion interrupt request for main PLL oscillation |
| 1 | Stabilization wait completion interrupt request for main PLL oscillation |

**[bit1] SOSCINT:**

| bit | Description |
|---|---|
| 0 | No stabilization wait completion interrupt request for sub-clock oscillation |
| 1 | Stabilization wait completion interrupt request for sub-clock oscillation |

**[bit0] MOSCINT:**

| bit | Description |
|---|---|
| 0 | No stabilization wait completion interrupt request for main clock oscillation |
| 1 | Stabilization wait completion interrupt request for main clock oscillation |

## 2.16 IRQ25/26 Batch Read Register (IRQxxMON)

The IRQ25 Batch Read Register (IRQ25MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 41.
The IRQ26 Batch Read Register (IRQ26MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 42.

IRQ25MON indicates the status of the interrupt requests of A/D converter unit 0 and MFS ch.9.
IRQ26MON indicates the status of the interrupt request of A/D converter unit 1 and MFS ch.10.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | MFSINT | | | ADCINT | | | | |
| Attribute | R | | | | | | | | R | | | R | | | | |
| Initial value | 0x00 | | | | | | | | 000 | | | 00000 | | | | |

### Register functions

**[bit31:8] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit7:5] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 7 | 0 | No status interrupt request of the corresponding MFS channel |
| | 1 | Status interrupt request of the corresponding MFS channel |
| 6 | 0 | No transmission interrupt request of the corresponding MFS channel |
| | 1 | Transmission interrupt request of the corresponding MFS channel |
| 5 | 0 | No reception interrupt request of the corresponding MFS channel |
| | 1 | Reception interrupt request of the corresponding MFS channel |

**[bit4:0] ADCINT:**

| bit No. | bit | Description |
|---|---|---|
| 4 | 0 | No range comparison result interrupt request in the corresponding A/D converter unit |
| | 1 | Range comparison result interrupt request in the corresponding A/D converter unit |
| 3 | 0 | No conversion result comparison interrupt request in the corresponding A/D converter unit |
| | 1 | Conversion result comparison interrupt request in the corresponding A/D converter unit |
| 2 | 0 | No FIFO overrun interrupt request in the corresponding A/D converter unit |
| | 1 | FIFO overrun interrupt request in the corresponding A/D converter unit |
| 1 | 0 | No scan conversion interrupt request in the corresponding A/D converter unit |
| | 1 | Scan conversion interrupt request in the corresponding A/D converter unit |
| 0 | 0 | No priority conversion interrupt request in the corresponding A/D converter unit |
| | 1 | Priority conversion interrupt request in the corresponding A/D converter unit |

If the DMA transfer request is selected by the DRQSEL register, the corresponding ADCINT bit is "0".

## 2.17 IRQ27 Batch Read Register (IRQ27MON)

The IRQ27 Batch Read Register (IRQ27MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 43.

IRQ27MON indicates the status of the interrupt requests of A/D converter unit 2, LCD controller and MFS ch.11.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | MFSINT | | | LCDCINT | ADCINT | | | | |
| Attribute | R | | | | | | | R | | | R | R | | | | |
| Initial value | 0000000 | | | | | | | 000 | | | 0 | 00000 | | | | |

### Register functions

**[bit31:9] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit8:6] MFSINT:**

| bit No. | bit | Description |
|---|---|---|
| 8 | 0 | No status interrupt request of MFS ch.11 |
| | 1 | Status interrupt request of the corresponding MFS ch.11 |
| 7 | 0 | No transmission interrupt request of the corresponding MFS ch.11 |
| | 1 | Transmission interrupt request of the corresponding MFS ch.11 |
| 6 | 0 | No reception interrupt request of the corresponding MFS ch.11 |
| | 1 | Reception interrupt request of the corresponding MFS ch.11 |

**[bit5] LCDCINT:**

| bit | Description |
|---|---|
| 0 | No interrupt request for LCD controller |
| 1 | Interrupt request for LCD controller |

**[bit4:0] ADCINT:**

| bit No. | bit | Description |
|---|---|---|
| 4 | 0 | No range comparison result interrupt request in the corresponding A/D converter unit 2 |
| | 1 | Range comparison result interrupt request in the corresponding A/D converter unit 2 |
| 3 | 0 | No conversion result comparison interrupt request in the A/D converter unit 2 |
| | 1 | Conversion result comparison interrupt request in the A/D converter unit 2 |
| 2 | 0 | No FIFO overrun interrupt request in the A/D converter unit 2 |
| | 1 | FIFO overrun interrupt request in the A/D converter unit 2 |
| 1 | 0 | No scan conversion interrupt request in the A/D converter unit 2 |
| | 1 | Scan conversion interrupt request in the A/D converter unit 2 |
| 0 | 0 | No priority conversion interrupt request in the A/D converter unit 2 |
| | 1 | Priority conversion interrupt request in the A/D converter unit 2 |

If the DMA transfer request is selected by the DRQSEL register, the corresponding ADCINT bit is "0".

## 2.18 IRQ28/29 Batch Read Register (IRQxxMON)

The IRQ28 Batch Read Register (IRQ28MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 44.
The IRQ29 Batch Read Register (IRQ29MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 45.

IRQ28MON indicates the status of the interrupt request of MFT unit 0.
IRQ29MON indicates the status of the interrupt request of MFT unit 1.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | OCUINT | | | | | | ICUINT | | | | FRTINT | | | | | |
| Attribute | R | | | | | | R | | | | R | | | | | |
| Initial value | 000000 | | | | | | 0000 | | | | 000000 | | | | | |

### Register functions

**[bit31:16] Reserved: Reserved bits**
A reserved bit reads "0".

**[bit15:10] OCUINT:**

| bit No. | bit | Description |
|---|---|---|
| 15 | 0 | No interrupt request of the output compare ch.5 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.5 in the corresponding MFT unit |
| 14 | 0 | No interrupt request of the output compare ch.4 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.4 in the corresponding MFT unit |
| 13 | 0 | No interrupt request of the output compare ch.3 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.3 in the corresponding MFT unit |
| 12 | 0 | No interrupt request of the output compare ch.2 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.2 in the corresponding MFT unit |
| 11 | 0 | No interrupt request of the output compare ch.1 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.1 in the corresponding MFT unit |
| 10 | 0 | No interrupt request of the output compare ch.0 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.0 in the corresponding MFT unit |

**[bit9:6] ICUINT:**

| bit No. | bit | Description |
|---|---|---|
| 9 | 0 | No interrupt request of the input capture ch.3 in the corresponding MFT unit |
| | 1 | Interrupt request of the input capture ch.3 in the corresponding MFT unit |
| 8 | 0 | No interrupt request of the input capture ch.2 in the corresponding MFT unit |
| | 1 | Interrupt request of the input capture ch.2 in the corresponding MFT unit |
| 7 | 0 | No interrupt request of the input capture ch.1 in the corresponding MFT unit |
| | 1 | Interrupt request of the input capture ch.1 in the corresponding MFT unit |
| 6 | 0 | No interrupt request of the input capture ch.0 in the corresponding MFT unit |
| | 1 | Interrupt request of the input capture ch.0 in the corresponding MFT unit |

**[bit5:0] FRTINT:**

| bit No. | bit | Description |
|---|---|---|
| 5 | 0 | No zero detection interrupt request of the free run timer ch.2 in the corresponding MFT unit |
| | 1 | Zero detection interrupt request of the free run timer ch.2 in the corresponding MFT unit |
| 4 | 0 | No zero detection interrupt request of the free run timer ch.1 in the corresponding MFT unit |
| | 1 | Zero detection interrupt request of the free run timer ch.1 in the corresponding MFT unit |
| 3 | 0 | No zero detection interrupt request of the free run timer ch.0 in the corresponding MFT unit |
| | 1 | Zero detection interrupt request of the free run timer ch.0 in the corresponding MFT unit |
| 2 | 0 | No peak value detection interrupt request of the free run timer ch.2 in the corresponding MFT unit |
| | 1 | Peak value detection interrupt request of the free run timer ch.2 in the corresponding MFT unit |
| 1 | 0 | No peak value detection interrupt request of the free run timer ch.1 in the corresponding MFT unit |
| | 1 | Peak value detection interrupt request of the free run timer ch.1 in the corresponding MFT unit |
| 0 | 0 | No peak value detection interrupt request of the free run timer ch.0 in the corresponding MFT unit |
| | 1 | Peak value detection interrupt request of the free run timer ch.0 in the corresponding MFT unit |

## 2.19 IRQ30 Batch Read Register (IRQ30MON)

The IRQ30 Batch Read Register (IRQ30MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 46.

IRQ30MON indicates the status of the interrupt requests of MFT unit 2 and DMAC ch.0 to ch.7.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | DMAINT | | | | | | | |
| Attribute | R | | | | | | | | R | | | | | | | |
| Initial value | 0x00 | | | | | | | | 0x00 | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | OCUINT | | | | | | ICUINT | | | | FRTINT | | | | | |
| Attribute | R | | | | | | R | | | | R | | | | | |
| Initial value | 000000 | | | | | | 0000 | | | | 000000 | | | | | |

### Register functions

**[bit31:24] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit23:16] DMAINT:**

| bit No. | bit | Description |
|---|---|---|
| 23 | 0 | No interrupt request of DMA controller ch.7 |
| | 1 | Interrupt request of DMA controller ch.7 |
| 22 | 0 | No interrupt request of DMA controller ch.6 |
| | 1 | Interrupt request of DMA controller ch.6 |
| 21 | 0 | No interrupt request of DMA controller ch.5 |
| | 1 | Interrupt request of DMA controller ch.5 |
| 20 | 0 | No interrupt request of DMA controller ch.4 |
| | 1 | Interrupt request of DMA controller ch.4 |
| 19 | 0 | No interrupt request of DMA controller ch.3 |
| | 1 | Interrupt request of DMA controller ch.3 |
| 18 | 0 | No interrupt request of DMA controller ch.2 |
| | 1 | Interrupt request of DMA controller ch.2 |
| 17 | 0 | No interrupt request of DMA controller ch.1 |
| | 1 | Interrupt request of DMA controller ch.1 |
| 16 | 0 | No interrupt request of DMA controller ch.0 |
| | 1 | Interrupt request of DMA controller ch.0 |

**[bit15:10] OCUINT:**

| bit No. | bit | Description |
|---|---|---|
| 15 | 0 | No interrupt request of the output compare ch.5 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.5 in the corresponding MFT unit |
| 14 | 0 | No interrupt request of the output compare ch.4 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.4 in the corresponding MFT unit |
| 13 | 0 | No interrupt request of the output compare ch.3 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.3 in the corresponding MFT unit |
| 12 | 0 | No interrupt request of the output compare ch.2 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.2 in the corresponding MFT unit |
| 11 | 0 | No interrupt request of the output compare ch.1 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.1 in the corresponding MFT unit |
| 10 | 0 | No interrupt request of the output compare ch.0 in the corresponding MFT unit |
| | 1 | Interrupt request of the output compare ch.0 in the corresponding MFT unit |

**[bit9:6] ICUINT:**

| bit No. | bit | Description |
|---|---|---|
| 9 | 0 | No interrupt request of the input capture ch.3 in the corresponding MFT unit |
| | 1 | Interrupt request of the input capture ch.3 in the corresponding MFT unit |
| 8 | 0 | No interrupt request of the input capture ch.2 in the corresponding MFT unit |
| | 1 | Interrupt request of the input capture ch.2 in the corresponding MFT unit |
| 7 | 0 | No interrupt request of the input capture ch.1 in the corresponding MFT unit |
| | 1 | Interrupt request of the input capture ch.1 in the corresponding MFT unit |
| 6 | 0 | No interrupt request of the input capture ch.0 in the corresponding MFT unit |
| | 1 | Interrupt request of the input capture ch.0 in the corresponding MFT unit |

**[bit5:0] FRTINT:**

| bit No. | bit | Description |
|---|---|---|
| 5 | 0 | No zero detection interrupt request of the free run timer ch.2 in the corresponding MFT unit |
| | 1 | Zero detection interrupt request of the free run timer ch.2 in the corresponding MFT unit |
| 4 | 0 | No zero detection interrupt request of the free run timer ch.1 in the corresponding MFT unit |
| | 1 | Zero detection interrupt request of the free run timer ch.1 in the corresponding MFT unit |
| 3 | 0 | No zero detection interrupt request of the free run timer ch.0 in the corresponding MFT unit |
| | 1 | Zero detection interrupt request of the free run timer ch.0 in the corresponding MFT unit |
| 2 | 0 | No peak value detection interrupt request of the free run timer ch.2 in the corresponding MFT unit |
| | 1 | Peak value detection interrupt request of the free run timer ch.2 in the corresponding MFT unit |
| 1 | 0 | No peak value detection interrupt request of the free run timer ch.1 in the corresponding MFT unit |
| | 1 | Peak value detection interrupt request of the free run timer ch.1 in the corresponding MFT unit |
| 0 | 0 | No peak value detection interrupt request of the free run timer ch.0 in the corresponding MFT unit |
| | 1 | Peak value detection interrupt request of the free run timer ch.0 in the corresponding MFT unit |

## 2.20 IRQ31 Batch Read Register (IRQ31MON)

The IRQ31 Batch Read Register (IRQ31MON) can batch-read the interrupt requests allocated to interrupt factor vector No. 47.

IRQ31MON indicates the status of the interrupt requests of the Flash memory and base timer.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | FLASHINT | Reserved | | | | | | | | | | |
| Attribute | R | | | | R | R | | | | | | | | | | |
| Initial value | 0000 | | | | 0 | 00000000000 | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | BTINT | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

### Register functions

**[bit31:28] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit27] FLASHINT:**

| bit | Description |
|---|---|
| 0 | No RDY, HANG interrupt request for flash memory |
| 1 | RDY, HANG interrupt request for flash memory |

**[bit26:16] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit15:0] BTINT:**

| bit No. | bit | Description |
|---|---|---|
| 15 | 0 | No IRQ1 interrupt request on the base timer ch.7 |
| | 1 | IRQ1 interrupt request on the base timer ch.7 |
| 14 | 0 | No IRQ0 interrupt request on the base timer ch.7 |
| | 1 | IRQ0 interrupt request on the base timer ch.7 |
| 13 | 0 | No IRQ1 interrupt request on the base timer ch.6 |
| | 1 | IRQ1 interrupt request on the base timer ch.6 |
| 12 | 0 | No IRQ0 interrupt request on the base timer ch.6 |
| | 1 | IRQ0 interrupt request on the base timer ch.6 |
| 11 | 0 | No IRQ1 interrupt request on the base timer ch.5 |
| | 1 | IRQ1 interrupt request on the base timer ch.5 |
| 10 | 0 | No IRQ0 interrupt request on the base timer ch.5 |
| | 1 | IRQ0 interrupt request on the base timer ch.5 |
| 9 | 0 | No IRQ1 interrupt request on the base timer ch.4 |
| | 1 | IRQ1 interrupt request on the base timer ch.4 |
| 8 | 0 | No IRQ0 interrupt request on the base timer ch.4 |
| | 1 | IRQ0 interrupt request on the base timer ch.4 |
| 7 | 0 | No IRQ1 interrupt request on the base timer ch.3 |
| | 1 | IRQ1 interrupt request on the base timer ch.3 |
| 6 | 0 | No IRQ0 interrupt request on the base timer ch.3 |
| | 1 | IRQ0 interrupt request on the base timer ch.3 |
| 5 | 0 | No IRQ1 interrupt request on the base timer ch.2 |
| | 1 | IRQ1 interrupt request on the base timer ch.2 |
| 4 | 0 | No IRQ0 interrupt request on the base timer ch.2 |
| | 1 | IRQ0 interrupt request on the base timer ch.2 |
| 3 | 0 | No IRQ1 interrupt request on the base timer ch.1 |
| | 1 | IRQ1 interrupt request on the base timer ch.1 |
| 2 | 0 | No IRQ0 interrupt request on the base timer ch.1 |
| | 1 | IRQ0 interrupt request on the base timer ch.1 |
| 1 | 0 | No IRQ1 interrupt request on the base timer ch.0 |
| | 1 | IRQ1 interrupt request on the base timer ch.0 |
| 0 | 0 | No IRQ0 interrupt request on the base timer ch.0 |
| | 1 | IRQ0 interrupt request on the base timer ch.0 |

If the DMA transfer request is selected by the DRQSEL register, the corresponding BTINT bit is "0".

As shown in the Table 2-1, base timer interrupt factors IRQ0 and IRQ1 differ depending on the base timer function to be used.

**Table 2-1 Interrupt factors for each function of the base timer**

| Function | Interrupt Factor IRQ0 | Interrupt Factor IRQ1 |
|---|---|---|
| 16-bit PWM timer | Underflow detection/<br>duty match detection | Timer start trigger detection |
| 16-bit PPG timer | Underflow detection | Timer start trigger detection |
| 16/32-bit reload timer | Underflow detection | Timer start trigger detection |
| 16/32-bit PWC timer | Overflow detection | Measurement finished detection |

## 2.21 Interrupt Factor Vector Relocate Setting Register (IRQCMODE)

The Interrupt Factor Vector Relocate Setting Register (IRQCMODE) selects whether the interrupt factor vectors are assigned according to Table 1-1 in chapter "Interrupts (A)" or to Table 1-1 in this chapter.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | IRQCMODE |
| Attribute | R/W | | | | | | | | | | | | | | | R/W |
| Initial value | 000000000000000 | | | | | | | | | | | | | | | 0 |

### Register functions

**[bit31:1] Reserved: Reserved bits**

A reserved bit reads "0".

**[bit0] IRQCMODE:**

| bit | Description |
|---|---|
| 0 | Assigns the interrupt factor vector according to Table 1-1 in chapter "Interrupts (A)". |
| 1 | Assigns the interrupt factor vector according to Table 1-1 in this chapter. |

## 2.22 Interrupt Factor Selection Register 0 (RCINTSEL0)

The Interrupt Factor Selection Register 0 (RCINTSEL0) selects the interrupt factors for the interrupt vectors No. 19 to No. 22. This register is valid when the IRQQCMODE:IRQCMODE bit is "1".

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | INTSEL3 | | | | | | | | INTSEL2 | | | | | | | |
| Attribute | R/W | | | | | | | | R/W | | | | | | | |
| Initial value | 0x00 | | | | | | | | 0x00 | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | INTSEL1 | | | | | | | | INTSEL0 | | | | | | | |
| Attribute | R/W | | | | | | | | R/W | | | | | | | |
| Initial value | 0x00 | | | | | | | | 0x00 | | | | | | | |

**Register functions**

**[bit31:24] INTSEL3:**
These bits select* the interrupt factor for the interrupt vector No. 22.

**[bit23:16] INTSEL2:**
These bits select* the interrupt factor for the interrupt vector No. 21.

**[bit15:8] INTSEL1:**
These bits select* the interrupt factor for the interrupt vector No. 20.

**[bit7:0] INTSEL0:**
These bits select* the interrupt factor for the interrupt vector No. 19.

*: See Table 2-1 for details of selection interrupt factors

***Notes:***

- *The interrupt factors selected by RCINTSEL0 are masked with IRQ11 to IRQ31. (The usable bits in IRQ11MON to IRQ31MON Registers are also masked.)*
- *Ensure that all interrupt factors selected in the INTSEL0 to INTSEL7 bits are different when selecting interrupt factors.*

## 2.23 Interrupt Factor Selection Register 1 (RCINTSEL1)

The Interrupt Factor Selection Register 1 (RCINTSEL1) selects the interrupt factors for the interrupt vectors No. 23 to No. 26. This register is valid when the IRQQCMODE:IRQCMODE bit is "1".

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | INTSEL7 | | | | | | | | INTSEL6 | | | | | | | |
| Attribute | R/W | | | | | | | | R/W | | | | | | | |
| Initial value | 0x00 | | | | | | | | 0x00 | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | INTSEL5 | | | | | | | | INTSEL4 | | | | | | | |
| Attribute | R/W | | | | | | | | R/W | | | | | | | |
| Initial value | 0x00 | | | | | | | | 0x00 | | | | | | | |

**Register functions**

**[bit31:24] INTSEL7:**
These bits select* the interrupt factor for the interrupt vector No. 26.

**[bit23:16] INTSEL6:**
These bits select* the interrupt factor for the interrupt vector No. 25.

**[bit15:8] INTSEL5:**
These bits select* the interrupt factor for the interrupt vector No. 24.

**[bit7:0] INTSEL4:**
These bits select* the interrupt factor for the interrupt vector No. 23.

*: See Table 2-1 for details of selection interrupt factors

***Notes:***

- *The interrupt factors selected by RCINTSEL0 are masked with IRQ11 to IRQ31. (The usable bits in IRQ11MON to IRQ31MON Registers are also masked.)*
- *Ensure that all interrupt factors selected in the INTSEL0 to INTSEL7 bits are different when selecting interrupt factors.*

**Table 2-1 Interrupt factor selection**

| Setting of RCINTSELx:INTSELx | Interrupt factor |
|---|---|
| 0x00 | No interrupt factor is selected. |
| 0x01 | External interrupt ch.0 |
| 0x02 | External interrupt ch.1 |
| 0x03 | External interrupt ch.2 |
| 0x04 | External interrupt ch.3 |
| 0x05 | External interrupt ch.4 |
| 0x06 | External interrupt ch.5 |
| 0x07 | External interrupt ch.6 |
| 0x08 | External interrupt ch.7 |
| 0x09 | External interrupt ch.8 |
| 0x0A | External interrupt ch.9 |
| 0x0B | External interrupt ch.10 |
| 0x0C | External interrupt ch.11 |
| 0x0D | IRQ0/IRQ1 of base timer ch.0 |
| 0x0E | IRQ0/IRQ1 of base timer ch.1 |
| 0x0F | IRQ0/IRQ1 of base timer ch.2 |
| 0x10 | IRQ0/IRQ1 of base timer ch.3 |
| 0x11 | IRQ0/IRQ1 of base timer ch.4 |
| 0x12 | IRQ0/IRQ1 of base timer ch.5 |
| 0x13 | IRQ0/IRQ1 of base timer ch.6 |
| 0x14 | IRQ0/IRQ1 of base timer ch.7 |
| 0x15 | Reception interrupt of MFS ch.0 |
| 0x16 | Reception interrupt of MFS ch.1 |
| 0x17 | Reception interrupt of MFS ch.2 |
| 0x18 | Reception interrupt of MFS ch.3 |
| 0x19 | Zero detection interrupt of MFT unit 0 free-run timer ch.0 |
| 0x1A | Zero detection interrupt of MFT unit 1 free-run timer ch.0 |
| 0x1B | Zero detection interrupt of MFT unit 2 free-run timer ch.0 |
| 0x1C | DMAC ch.0 |
| 0x1D | DMAC ch.1 |
| 0x1E | DMAC ch.2 |
| 0x1F | DMAC ch.3 |
| 0x20 | Reception interrupt of MFS ch.8 |
| 0x21 | Reception interrupt of MFS ch.9 |
| 0x22 | Reception interrupt of MFS ch.10 |
| 0x23 | Reception interrupt of MFS ch.11 |
| 0x24 | Reception interrupt of MFS ch.12 |
| 0x25 | Reception interrupt of MFS ch.13 |
| 0x26 | Reception interrupt of MFS ch.14 |
| 0x27 | Reception interrupt of MFS ch.15 |
| 0x28 | Transmission/Status Interrupt of MFS ch.8 |
| 0x29 | Transmission/Status Interrupt of MFS ch.9 |
| 0x2A | Transmission/Status Interrupt of MFS ch.10 |
| 0x2B | Transmission/Status Interrupt of MFS ch.11 |
| 0x2C | Transmission/Status Interrupt of MFS ch.12 |
| 0x2D | Transmission/Status Interrupt of MFS ch.13 |
| 0x2E | Transmission/Status Interrupt of MFS ch.14 |
| 0x2F | Transmission/Status Interrupt of MFS ch.15 |
| 0x30 to 0xFF | Reserved |

# 3. Usage Precautions

Be careful with the following points when using the interrupt controller.

- The interrupt request signals from each of the peripheral resources are notified by level. When exiting from interrupt processing, always clear the interrupt request.
- The NMIX pin is allocated shared with a general-purpose port. The initial value after reset is released is configured as a general-purpose port, and the NMI input is masked. In order to use the NMI, enable NMI by configuring the port setting. See the chapter of "External Interrupt and NMI Control Unit" for details.
- See the chapters for each of the macros for the correlation between the specific event detection registers and interrupt enable registers in each peripheral resource.

# CHAPTER8: External Interrupt and NMI Control Sections

**This chapter explains the functions and operations of the external interrupt and NMI control sections.**



CODE: 9BFEXTINT-E03.0_FW12-E1.04

# 1. Overview

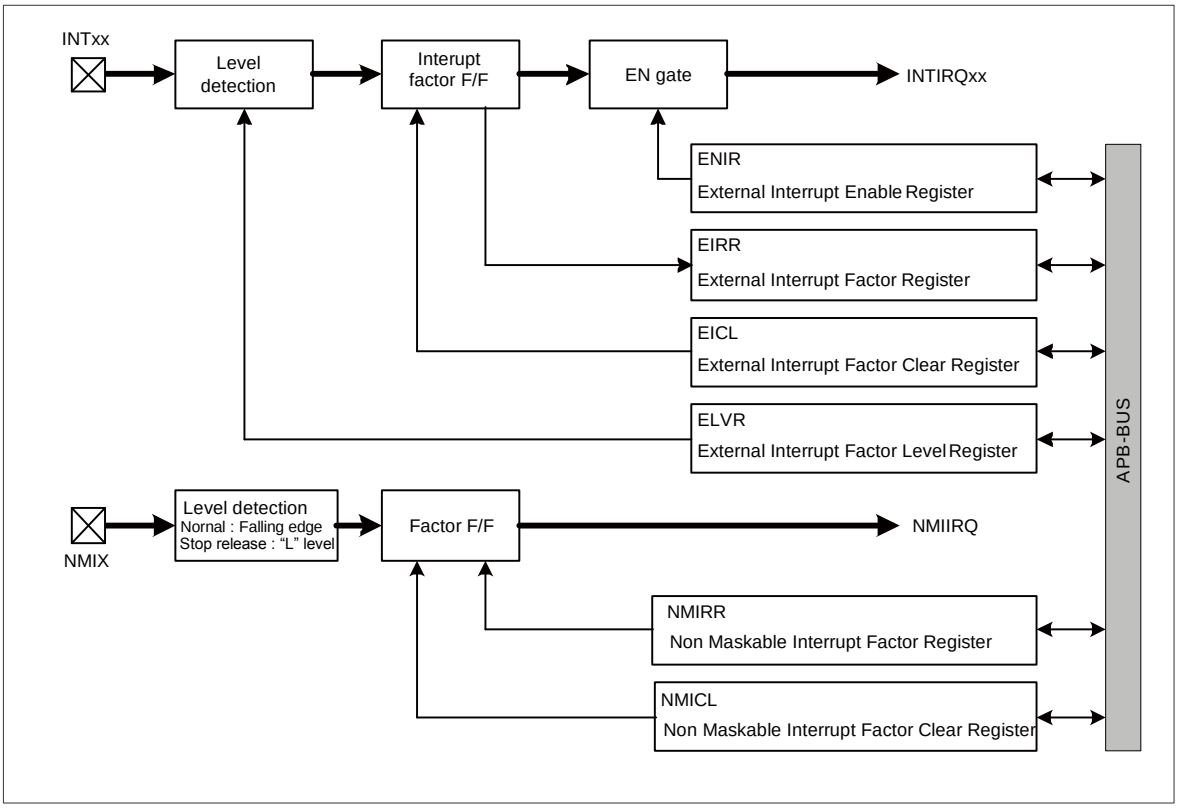
The external interrupt and NMI control sections have the following features.

- Has up to 32 external interrupt input pins and one NMI input pin mounted.
- Possible to select the "H" level, "L" level, rising edge, or falling edge to detect an external interrupt.
- Possible to use an external interrupt input or NMI input to return from standby mode.

# 2. Block Diagram

The following shows the block diagram of the external interrupt and NMI control sections.

**Figure 2-1 Block diagram of external interrupt and NMI control sections**



# 3. Operations and Setting Procedure Examples

This section explains operations and setting procedure examples.

## 3.1 Operations of external interrupt control section

This section shows the operations of the external interrupt control section.

### Overview of operations in external interrupt control section

The external interrupt control section outputs an external interrupt request to the interrupt controller in the following procedure.

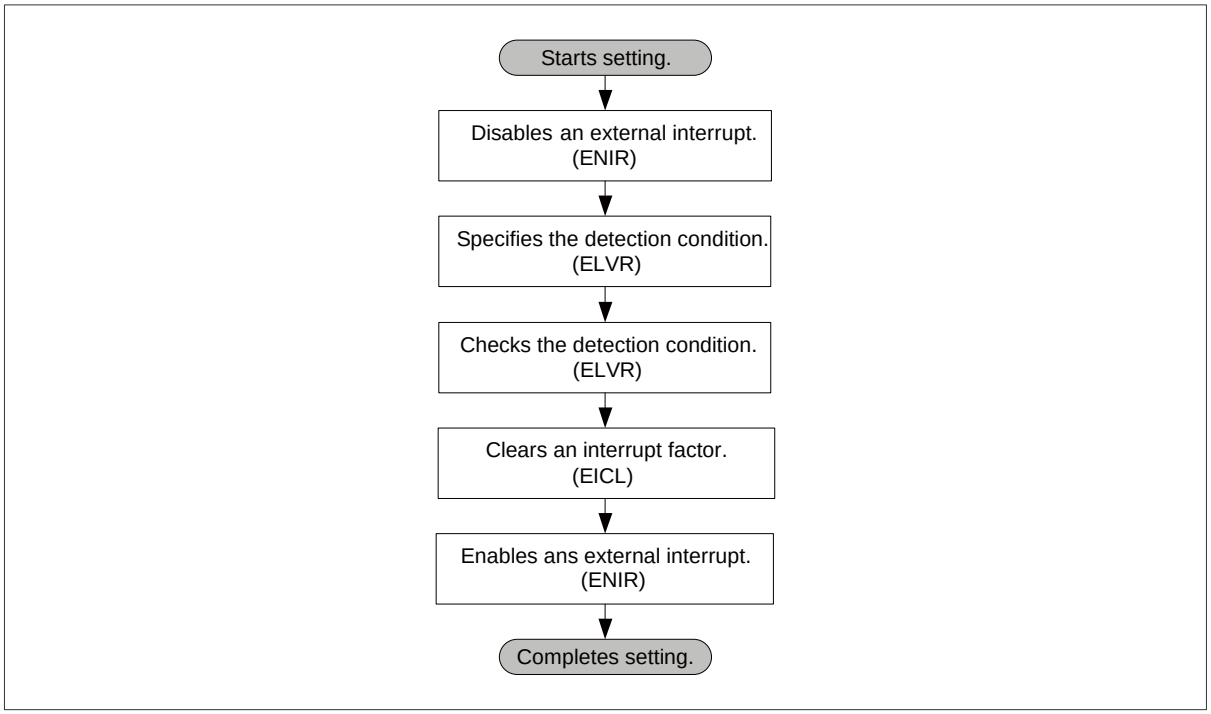
1. The signal input to pin INTxx detects the edge or level specified in the External Interrupt Level Register (ELVR). The edge or level to be detected can be selected from the following four types:
   "H" level, "L" level, rising edge, falling edge
2. The detected interrupt input is held in the interrupt factor F/F.
   It is read with the External Interrupt Factor Register (EIRR).
   The held interrupt factor is cleared with the External Interrupt Factor Clear Register (EICL).
3. If an external interrupt is enabled with the External Interrupt Enable Register (ENIR), an external interrupt request (INTIRQxx) is output to the interrupt controller.

### Setting procedure

Execute the following steps to configure external interrupt setting.

1. Disable an external interrupt with the External Interrupt Enable Register (ENIR).
2. Specify the detection condition (effective edge or level) with the External Interrupt Factor Level Register (ELVR).
3. Read the External Interrupt Factor Level Register (ELVR).
4. Clear the external interrupt factor with the External Interrupt Factor Clear Register (EICL).
5. Enable the external interrupt with the External Interrupt Enable Register (ENIR).

**Figure 3-1 External interrupt setting procedure**



### Canceling an external interrupt request

When the external interrupt detection condition is set to the "H" or "L" level, an interrupt factor is held in the External Interrupt Factor Register (EIRR) even if an external interrupt request input (INTxx) is canceled. Therefore, an external interrupt request (INTIRQxx) remains output to the interrupt controller.

Execute the following steps to cancel an external interrupt request.

1. Read the External Interrupt Factor Register (EIRR), and check the interrupt factor.
2. Write "0" to the corresponding bit in the External Interrupt Factor Clear Register (EICL) to clear it.
3. Read the External Interrupt Factor Register (EIRR), and check that the interrupt factor is cleared.

**Figure 3-2 Clearing an interrupt factor**

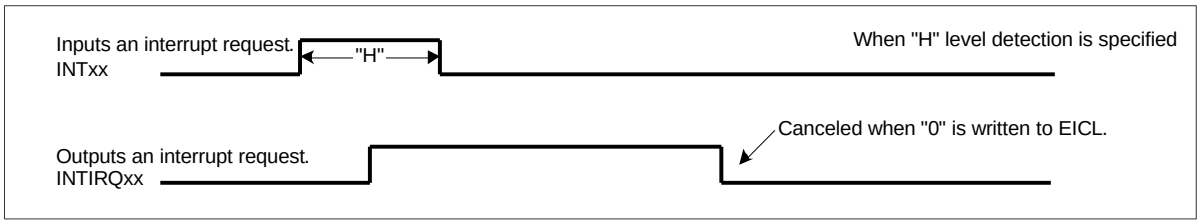
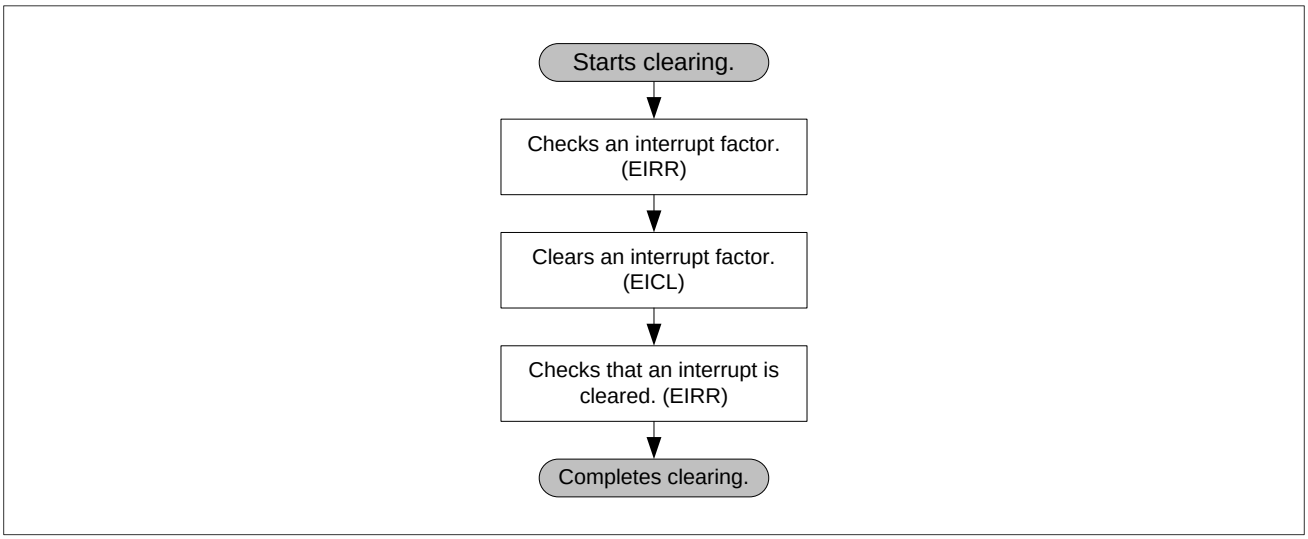



**Figure 3-3 Canceling an external interrupt request**



## 3.2 Operations of NMI control section

This section shows the operations of the NMI control section.

### Overview of NMI control section

The NMI control section outputs an NMI interrupt request (NMIIRQ) to the CPU if the edge or level is detected from the signal input to the NMI input pin (NMIX).
The following edge or level is detected.

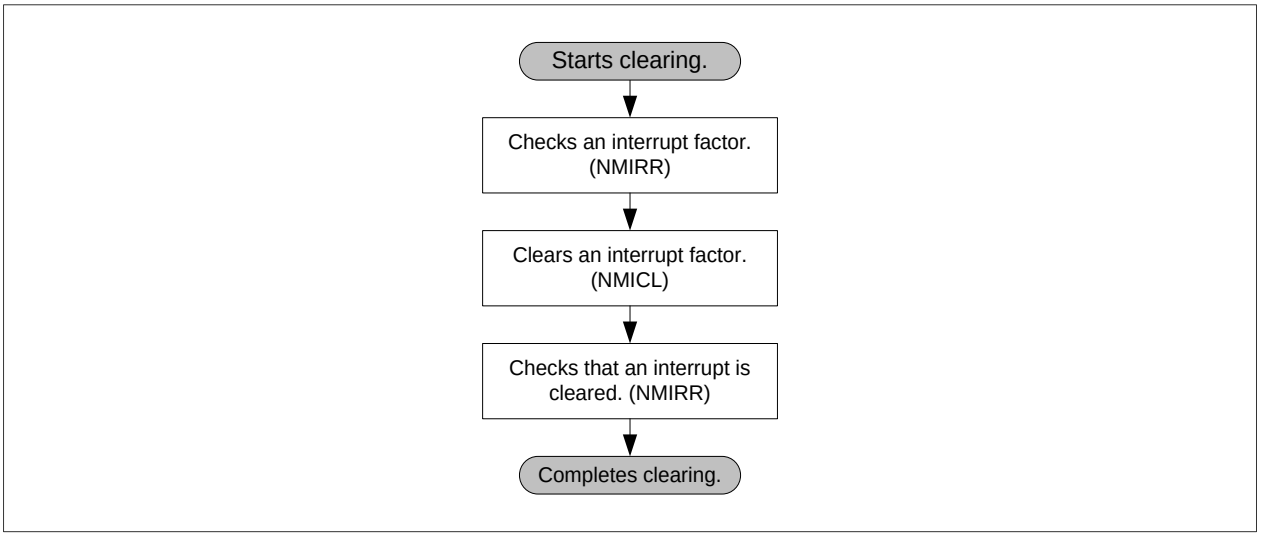
- Normal mode : Falling edge
- Stop mode : "L" level

### Canceling an NMI request

To cancel an NMI request, clear the request register in the same way as for an external interrupt request.
Execute the following steps to cancel an NMI interrupt request.

1. Read the NMI Factor Register (NMIRR), and check the interrupt factor.
2. Write "0" to the corresponding bit in the NMI Factor Clear Register (NMICL) to clear it.
3. Read the NMI Factor Register (NMIRR), and check that the interrupt factor is cleared.

**Figure 3-4 Canceling an NMI request**



## 3.3 Returning from timer or stop mode

This section shows a return from the timer or stop mode.

**Overview**

An external interrupt and NMI requests can be used to return from timer or stop mode.
In timer or stop mode, the signal first input to pin INTxx or NMIX is input asynchronously, and the device can return from timer or stop mode.

**Setting before changing to stop mode**

To use an external interrupt request, in the External Interrupt Enable Register (ENIR), specify the pin used to return from stop mode and also specify the effective detection level before changing to stop mode.

- Pin used to return from stop mode. : Interrupt request output enable (ENIR = 1)
- Pin not used to return from stop mode. : Interrupt request output disable (ENIR = 0)

To use an NMI request, only the "L" level is detected, and no register setting is required.

**Returning from stop mode**

For external interrupt request, if the pre-specified effective level is detected in the pin used to return from stop mode, the device returns from stop mode.
For NMI request, if the "L" level is detected in stop mode, the device returns from stop mode.

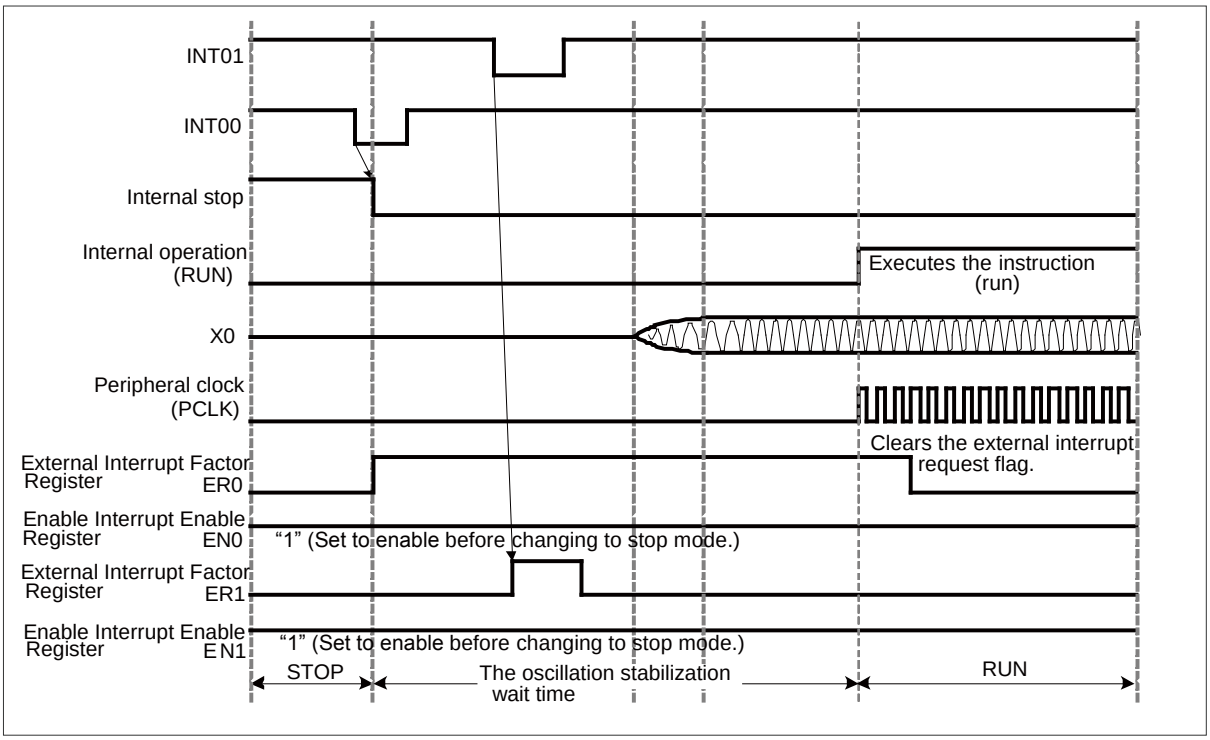
**Notes on returning from stop mode**

Any other external interrupt requests cannot be recognized until the oscillation stabilization wait time lapses after stop mode was released.

(For INT01 in Figure 3-5, any external interrupt requests cannot be recognized.)

Therefore, to input an external interrupt after stop mode was released, input an external interrupt signal after the oscillation stabilization wait time lapsed.

**Figure 3-5 Returning from stop mode**



# 4. Registers

This section provides a list of registers.

### Register list

The following shows a list of registers in the external interrupt and NMI control sections.

**Table 4-1 Registers in external interrupt and NMI control sections**

| Abbreviation | Register name | Reference |
|---|---|---|
| ENIR | External Interrupt Enable Register | 4.1 |
| EIRR | External Interrupt Factor Register | 4.2 |
| EICL | External Interrupt Factor Clear Register | 4.3 |
| ELVR | External Interrupt Factor Level Register | 4.4 |
| ELVR1 | External Interrupt Factor Level Register 1 | 4.5 |
| NMIRR | Non Maskable Interrupt Factor Register | 4.6 |
| NMICL | Non Maskable Interrupt Factor Clear Register | 4.7 |

## 4.1 External Interrupt Enable Register (ENIR)

The ENIR register is used to control masking an external interrupt request output.

### 4.1.1 Register configuration

| bit | 31 | | | | | | | | | | | | | | | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | EN[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | EN[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

### 4.1.2 Register functions

**[bit31:0] EN31 to EN0: External interrupt enable bits**

EN31 to EN0 bits correspond to pins INT31 to INT00.

It is not possible to set the bit corresponding to a pin that is not defined in the product specifications.

| bit | Description |
|---|---|
| 0 | Disables the output of an external interrupt request of INTx pin corresponding to the relevant bit. |
| 1 | Enables the output of an external interrupt request of INTx pin corresponding to the relevant bit. |

This function enables the interrupt request output corresponding to the bit that is set to "1" in this register, and outputs a request to the interrupt controller. The pin corresponding to the bit that is set to "0" holds an interrupt factor, but outputs no request to the interrupt controller.

## 4.2 External Interrupt Factor Register (EIRR)

The EIRR register indicates that an external interrupt request is detected.

### 4.2.1 Register configuration

| bit | 31 | | | | | | | | | | | | | | | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | ER[31:16] | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0xXXXX | | | | | | | | | | | | | | | |

| bit | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | ER[15:0] | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0xXXXX | | | | | | | | | | | | | | | |

### 4.2.2 Register functions

**[bit31:0] ER31 to ER0: External interrupt request detection bits**

ER31 to ER0 bits correspond to pins INT31 to INT00.

The bit corresponding to a pin that is not defined in the product specifications is indefinite.

| bit | Function |
|---|---|
| 0 | Detects no external interrupt request of INTx pin corresponding to the relevant bit. |
| 1 | Detects an external interrupt request of INTx pin corresponding to the relevant bit. |
| Writing | No effect on operation |

***Notes:***

- *When level detection is set with ELVR and while valid level is input from INTxx pin, clearing applicable bit (write "0") with the External Interrupt Factor Clear register (EICL) will reset "1" to applicable bit in the External Interrupt Factor Register (EIRR).*
- *As the initial values of GPIO are set to general purpose ports, applicable bit in the External Interrupt Factor Register (EIRR) may be set to "1". After set the GPIO to external interrupt pin, clear the External Interrupt Factor Register (EIRR).*

## 4.3 External Interrupt Factor Clear Register (EICL)

The EICL register is used to clear the held interrupt factor.

### 4.3.1 Register configuration

| bit | 31 | | | | | | | | | | | | | | | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | ECL[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0xFFFF | | | | | | | | | | | | | | | |

| bit | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | ECL[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0xFFFF | | | | | | | | | | | | | | | |

### 4.3.2 Register functions

**[bit31:0] ECL[31:0]: External interrupt factor clear bits**

ECL[31:0] bits correspond to pins INT31 to INT00.

It is not possible to write "0" to the bit corresponding to a pin that is not defined in the product specifications.

| bit | Function |
|---|---|
| When "0" is written | Clears an external interrupt factor of INTx pin corresponding to the relevant bit. |
| When "1" is written | No effect on operation |
| Reading | Always reads "1". |

## 4.4 External Interrupt Factor Level Register (ELVR)

The ELVR is used to select the level or edge of the signal detected as an external interrupt request.

### 4.4.1 Register configuration

| bit | 31 | | | | | | | | | | | | | | | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | LB15 | LA15 | LB14 | LA14 | LB13 | LA13 | LB12 | LA12 | LB11 | LA11 | LB10 | LA10 | LB9 | LA9 | LB8 | LA8 |
| Attribute | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| bit | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | LB7 | LA7 | LB6 | LA6 | LB5 | LA5 | LB4 | LA4 | LB3 | LA3 | LB2 | LA2 | LB1 | LA1 | LB0 | LA0 |
| Attribute | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 4.4.2 Register functions

**[bit31:0] LA15 to LA0 or LB15 to LB0: External interrupt request detection level selection bits**

LA15 to LA0 or LB15 to LB0 bits correspond to pins INT15 to INT00 on a 2-bit (LA and LB) basis.
It is not possible to set the bit corresponding to a pin that is not defined in the product specifications.
If the edge or level selected with this bit is detected, it is recognized as an external interrupt request.

| LBx | LAx | Description |
|---|---|---|
| 0 | 0 | Detects the "L" level. |
| 0 | 1 | Detects the "H" level. |
| 1 | 0 | Detects the rising edge. |
| 1 | 1 | Detects the falling edge. |

## 4.5 External Interrupt Factor Level Register 1 (ELVR1)

The ELVR is used to select the level or edge of the signal detected as an external interrupt request.

### 4.5.1 Register configuration

| bit | 31 | | | | | | | | | | | | | | | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | LB31 | LA31 | LB30 | LA30 | LB29 | LA29 | LB28 | LA28 | LB27 | LA27 | LB26 | LA26 | LB25 | LA25 | LB24 | LA24 |
| Attribute | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| bit | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | LB23 | LA23 | LB22 | LA22 | LB21 | LA21 | LB20 | LA20 | LB19 | LA19 | LB18 | LA18 | LB17 | LA17 | LB16 | LA16 |
| Attribute | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 4.5.2 Register functions

**[bit31:0] LA31 to LA16 or LB31 to LB16: External interrupt request detection level selection bits**

LA31 to LA16 or LB31 to LB16 bits correspond to pins INT31 to INT16 on a 2-bit (LA and LB) basis.
It is prohibited to set the bit corresponding to a pin that is not defined in the product specifications.
If the edge or level selected with this bit is detected, it is recognized as an external interrupt request.

| LBx | LAx | Description |
|---|---|---|
| 0 | 0 | Detects the "L" level. |
| 0 | 1 | Detects the "H" level. |
| 1 | 0 | Detects the rising edge. |
| 1 | 1 | Detects the falling edge. |

## 4.6 Non Maskable Interrupt Factor Register (NMIRR)

The NMIRR Register indicates that a non maskable interrupt (NMI) request is detected.

### 4.6.1 Register configuration

| bit | 15 | 1 | 0 |
|---|---|---|---|
| Field | Reserved | | NR |
| Attribute | - | | R |
| Initial value | - | | 0 |

### 4.6.2 Register functions

**[bit15:1] Reserved: Reserved bits**

The read value is undefined.

They have no effect in write mode.

**[bit0] NR: NMI interrupt request detection bit**

The NR bit corresponds to NMIX pin.

| bit | Function |
|---|---|
| 0 | Detects no NMI interrupt request. |
| 1 | Detects an NMI interrupt request. |
| Writing | No effect on operation |

## 4.7 Non Maskable Interrupt Factor Clear Register (NMICL)

The NMICL register is used to clear the held interrupt factor.

### 4.7.1 Register configuration

| bit | 15 | 1 | 0 |
|---|---|---|---|
| Field | Reserved | | NCL |
| Attribute | - | | R/W |
| Initial value | - | | 1 |

### 4.7.2 Register functions

**[bit15:1] Reserved: Reserved bits**

The read value is undefined.
They have no effect in write mode.

**[bit0] NCL: NMI interrupt factor clear bit**

The NCL bit corresponds to NMIX pin.

| bit | Function |
|---|---|
| When "0" is written | Clears an NMI interrupt factor. |
| When "1" is written | No effect on operation |
| Reading | Always reads "1". |

***Notes:***

- *If ELVR is rewritten to change the detection condition, an invalid interrupt factor may occur.*
  *To avoid an invalid interrupt factor from occurring, keep the procedure shown in Figure 3-1 when changing the detection condition.*
- *To detect the edge or level specified in ELVR, at least 3T (T: PCLK cycle) is required as the pulse width.*
  *If a signal that does not satisfy the pulse width is input, it is not guaranteed that correct operations will be carried out.*
- *When level detection is specified in ELVR, the corresponding bit in the External Interrupt Factor Register (EIRR) is set to "1" again while the effective level is input from pin INTxx even if the corresponding bit is cleared (set to "0") with the External Interrupt Factor Clear Register (EICL).*
- *The NMI detection level setting register is not provided. In normal mode, the falling edge is detected. This register is used to return from stop mode when the "L" level is detected.*
- *NMI is targeted for non maskable interrupt, so an NMI Enable Interrupt Request Register is not provided.*

# CHAPTER9: DMAC

**This chapter explains DMAC.**



CODE: 9BFDMAC-E02.0_MHDMAC-E01.1

# 1. Overview of DMAC

DMAC (Direct Memory Access Controller) is a function block that transfers data at high speed without CPU. Using DMAC improves the system performance.

### Overview of DMAC

- DMAC has its own bus which is independent from the CPU bus; therefore, it allows for transfer operation even when the CPU bus is accessed.
- It consists of maximum 8 channels enabled to execute 8 types of different DMA transfers independently from one another.
- It can set the address of the transfer destination, the address of the transfer source, the size of transfer data, the source of transfer request and the transfer mode, and control the start of transfer operation, the forced termination of transfer and the pause of transfer for each channel.
- It can control the batch start of transfers, the forced batch termination of transfers and the batch pause of transfers for all of the channels.
- When multiple channels are operating simultaneously, it can select the priority of such channel operations from the fixed method or the rotated method.
- It supports hardware DMA transfer using an interrupt signal from Peripherals.
- It complies with the system bus (AHB), supporting 32-bit address space (4Gbytes).

### Overview of Functions of Each Channel

- The addresses of the transfer source and transfer destination can be incremented or fixed.
- Reload function for the addresses of the transfer source and transfer destination (i.e. function to return the values to the original settings upon completion of the transfer) is available.
- The size of data to be transferred can be selected from the following three specifications:

  Transfer data width : (Select from byte/half-word/word)
  Setting the number of blocks : (Select from 1 to 16)
  Setting the number of transfers : (Select from 1 to 65536)
  (For information about the difference between the number of blocks and the number of transfers, see "3 Functions and Operations of DMAC".)

- Whether or not to give notification of the successful completion of transfer and unsuccessful completion of transfer can be specified.
- Transfer mode can be selected from the following five types:

  Software-Block transfer
  Software-Burst transfer
  Hardware-Demand transfer
  Hardware-Block transfer
  Hardware-Burst transfer

### Transfer Modes

Software transfer is a method used to start DMAC by direct instruction from CPU.

Hardware transfer is a method using an interrupt signal from a Peripheral as the DMAC transfer request signal to start DMAC directly when the Peripheral issues a transfer request.

Multifunction serial unit and ADC unit directly instruct DMAC to start data transfer, when sending/receiving data or A/D conversion data needs to be transferred. External interrupt unit and Base timer unit directly instruct DMAC to start data transfer at a transfer timing. In either of the cases, data can be transferred without CPU by making such setting beforehand.

### Abbreviations

This chapter contains the following terms: DE, DS, DH, PR, EB, PB, ST, IS, BC, TC, MS, TW, FS, FD, RC, RS, RD, EI, CI, SS, EM. All of these terms refer to each bit of DMAC control registers (DMACR, DMACSA, DMACDA, DMACA, DMACB). See "5 Registers of DMAC".

# 2. Configuration of DMAC

This section explains the system configuration of DMAC and the I/O pins of DMAC.

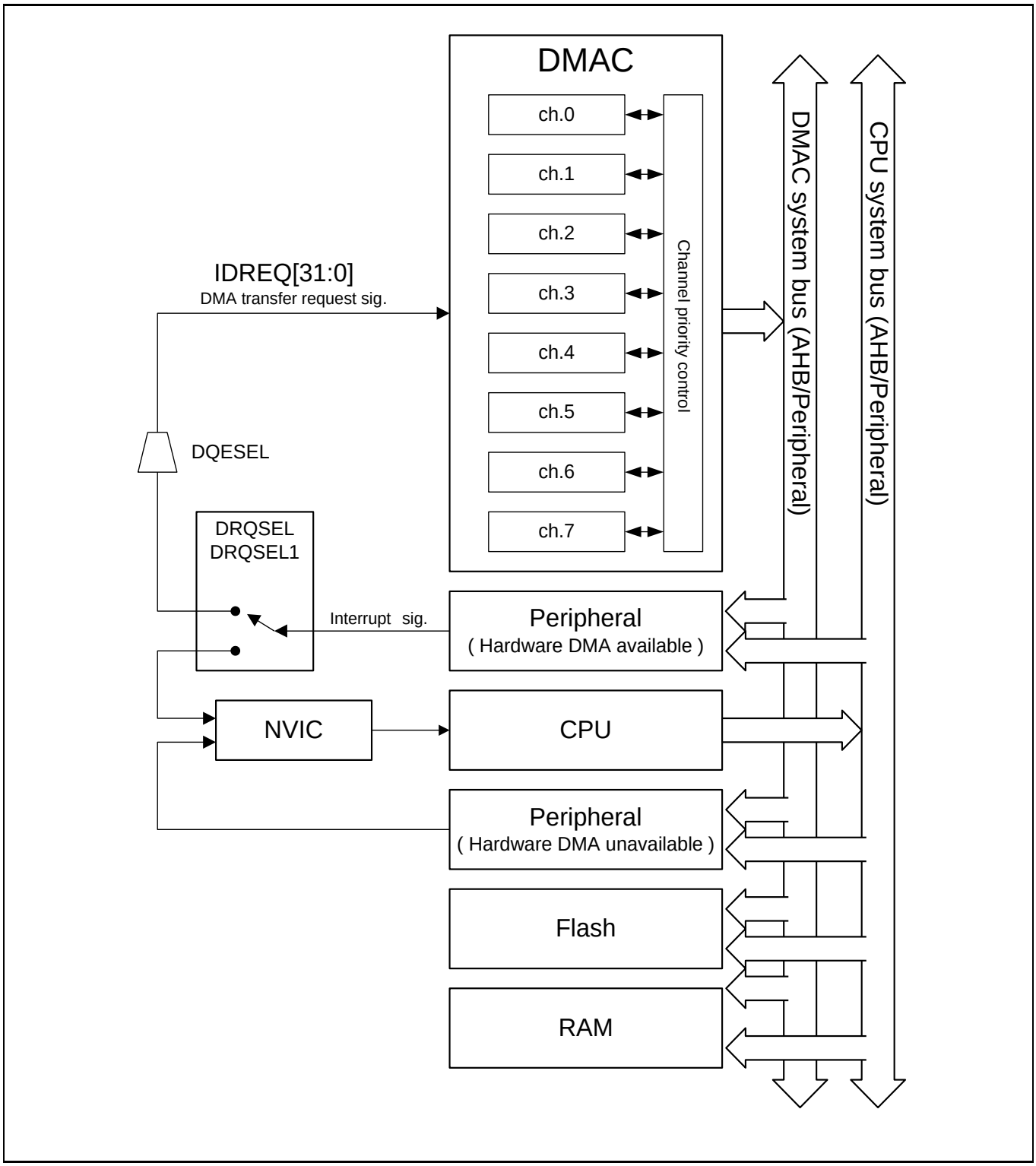
## 2.1 DMAC and System Configuration

This section explains DMAC and its system configuration.

**Block Diagram**

Figure 2-1 shows a diagram of DMAC and its system configuration.

**Figure 2-1 Block Diagram of DMAC and System Configuration**



### Explanation of Block Diagram

■ DMAC

DMAC is in maximum 8-ch configuration. Each channel performs independent transfer. The priority controller controls the transfer operations of these channels, when there is a conflict among them.

■ Connection to the system

The diagram of the system configuration in the figure has been simplified for explanation purposes. For more details, see the chapter " System Overview ". DMAC is connected to CPU, Flash, RAM and Peripherals via the system bus. It has its own bus that is independent from the CPU bus, allowing for transfer operation at CPU bus access. It accesses any address area in the system by specifying the address of transfer destination and transfer source for each channel in order to transfer data between the memory and Peripheral. Since some areas cannot be accessed from DMAC, check the memory map.

■ Connection of the hardware transfer request signal

The interrupt signal from the Peripheral supporting hardware transfer is selected in the interrupt controller block (indicated as DRQSEL/DRQSEL1 in Figure 2-1) either to be used as the interrupt signal to CPU or the DMA transfer request signal to DMAC. When performing DMA transfer by hardware request, connect the interrupt signal from each Peripheral as the transfer request signal to DMAC in advance by setting DRQSEL/DREQSEL1. The interrupt signal from the Peripheral that does not support hardware transfer cannot be used as the DMA transfer request signal. When the interrupt signal is used as the transfer request signal to DMAC, it cannot be used as the interrupt signal to CPU. See the chapter "Interrupts".

There are 32 DMA transfer request signals to be input to DMAC. For the correspondence between each signal and Peripheral, see Table 2-1 in the next section.

Interrupt signals from the peripheral that is not integrated cannot be selected. It should be noted that for a Peripheral with multiple channels and multiple interrupt factors, some interrupts support DMA transfer, while others don't.

In the case of hardware transfer, each channel of DMAC selects one transfer request signal out of the above 32 transfer request signals in its operation. The IS register is used for the selection.

■ Connection of the hardware transfer request clear signal

Some of the Peripherals that support hardware transfer are required to clear the transfer request signal (interrupt signal) after the completion of the transfer. Although it is not illustrated in Figure 2-1, the transfer request signal is cleared for such Peripherals via DMAC by selecting it by DRQSEL/DRQSEL1.

■ Connection of the hardware transfer stop request signal

The multifunction serial unit (hereinafter abbreviated as "MFS") outputs the DMA transfer stop request signal. Although it is not illustrated in Figure 2-1, MFS's transfer stop request signal is connected to DMAC, when MFS is selected by DRQSEL/DRQSEL1. When the transfer stop request signal is asserted, DMAC stops the transfer operation. It is configured to mask the succeeding transfer request signals.

■ Interrupt signal from DMAC

Although it is not illustrated in Figure 2-1, an interrupt signal used to give notification of transfer completion is connected to NVIC. Each channel has 8 interrupt outputs.

## 2.2 I/O Signals of DMAC

This section explains the I/O signals of DMAC.

### Transfer Request Signals to be Input to DMAC

Table 2-1 shows a list of the transfer request signals to be input to DMAC and the interrupt signals from the corresponding Peripherals.

**Table 2-1 List of Transfer Request Signals and Interrupt Signals from Corresponding Peripherals**

| IDREQ No. | Interrupt Signal of Corresponding Peripheral |
|---|---|
| 0 | Reserverd |
| 1 | Reserverd |
| 2 | Reserverd |
| 3 | Reserverd |
| 4 | Reserverd |
| 5 | Scan conversion interrupt signal from A/D converter unit0 |
| 6 | Scan conversion interrupt signal from A/D converter unit1 |
| 7 | Scan conversion interrupt signal from A/D converter unit2 |
| 8 | Interrupt signal from IRQ0 of base timer ch.0 |
| 9 | Interrupt signal from IRQ0 of base timer ch.2 |
| 10 | Interrupt signal from IRQ0 of base timer ch.4 |
| 11 | Interrupt signal from IRQ0 of base timer ch.6 |
| 12 | Receiving interrupt signal from MFS ch.0 |
| 13 | Sending interrupt signal from MFS ch.0 |
| 14 | Receiving interrupt signal from MFS ch.1 |
| 15 | Sending interrupt signal from MFS ch.1 |
| 16 | Receiving interrupt signal from MFS ch.2 |
| 17 | Sending interrupt signal from MFS ch.2 |
| 18 | Receiving interrupt signal from MFS ch.3 |
| 19 | Sending interrupt signal from MFS ch.3 |
| 20 | Receiving interrupt signal from MFS ch.4 |
| 21 | Sending interrupt signal from MFS ch.4 |
| 22 | Receiving interrupt signal from MFS ch.5 |
| 23 | Sending interrupt signal from MFS ch.5 |
| 24 | Receiving interrupt signal from MFS ch.6 |
| 25 | Sending interrupt signal from MFS ch.6 |
| 26 | Receiving interrupt signal from MFS ch.7 |
| 27 | Sending interrupt signal from MFS ch.7 |
| 28 | Interrupt signal from external interrupt unit ch.0 |
| 29 | Interrupt signal from external interrupt unit ch.1 |
| 30 | Interrupt signal from external interrupt unit ch.2 |
| 31 | Interrupt signal from external interrupt unit ch.3 |

### Interrupt Signals Output from DMAC

Table 2-2 shows a list of the interrupt signals output from DMAC.

**Table 2-2 List of Interrupt Signals from DMAC**

| Name of Interrupt Signal | Interrupt Factor Register | Interrupt Enable Register | Interrupt Type |
|---|---|---|---|
| DIRQ0 | DMACB0:SS[2:0] | DMACB0:CI | ch.0 successful transfer completion interrupt |
| | | DMACB0:EI | ch.0 unsuccessful transfer completion interrupt |
| DIRQ1 | DMACB1:SS[2:0] | DMACB1:CI | ch.1 successful transfer completion interrupt |
| | | DMACB1:EI | ch.1 unsuccessful transfer completion interrupt |
| DIRQ2 | DMACB2:SS[2:0] | DMACB2:CI | ch.2 successful transfer completion interrupt |
| | | DMACB2:EI | ch.2 unsuccessful transfer completion interrupt |
| DIRQ3 | DMACB3:SS[2:0] | DMACB3:CI | ch.3 successful transfer completion interrupt |
| | | DMACB3:EI | ch.3 unsuccessful transfer completion interrupt |
| DIRQ4 | DMACB4:SS[2:0] | DMACB4:CI | ch.4 successful transfer completion interrupt |
| | | DMACB4:EI | ch.4 unsuccessful transfer completion interrupt |
| DIRQ5 | DMACB5:SS[2:0] | DMACB5:CI | ch.5 successful transfer completion interrupt |
| | | DMACB5:EI | ch.5 unsuccessful transfer completion interrupt |
| DIRQ6 | DMACB6:SS[2:0] | DMACB6:CI | ch.6 successful transfer completion interrupt |
| | | DMACB6:EI | ch.6 unsuccessful transfer completion interrupt |
| DIRQ7 | DMACB7:SS[2:0] | DMACB7:CI | ch.7 successful transfer completion interrupt |
| | | DMACB7:EI | ch.7 unsuccessful transfer completion interrupt |

Reference: Interrupt Generation Factors and Clearing (For details, see "4 DMAC Control".)

Interrupt from each channel is generated by the following factors:

- Upon the successful completion of channel transfer, "101" is set to SS[2:0] of each channel. If the above value is set to SS[2:0] with CI=1 (successful transfer completion interrupt enabled), a successful transfer completion interrupt occurs.
- Upon the unsuccessful completion of channel transfer, "001", "010", "011" and "100" are set to SS[2:0] of each channel. If the above value is set to SS[2:0] with EI=1 (unsuccessful transfer completion interrupt enabled), an unsuccessful transfer completion interrupt occurs.
- The successful transfer completion interrupt and the unsuccessful transfer completion interrupt undergo logic OR; therefore, if either of the interrupts occurs, an interrupt occurs from the channel.

Interrupt from each channel can be cleared by writing "000" to SS[2:0].

# 3. Functions and Operations of DMAC

This section explains the operations of DMAC in each transfer mode.

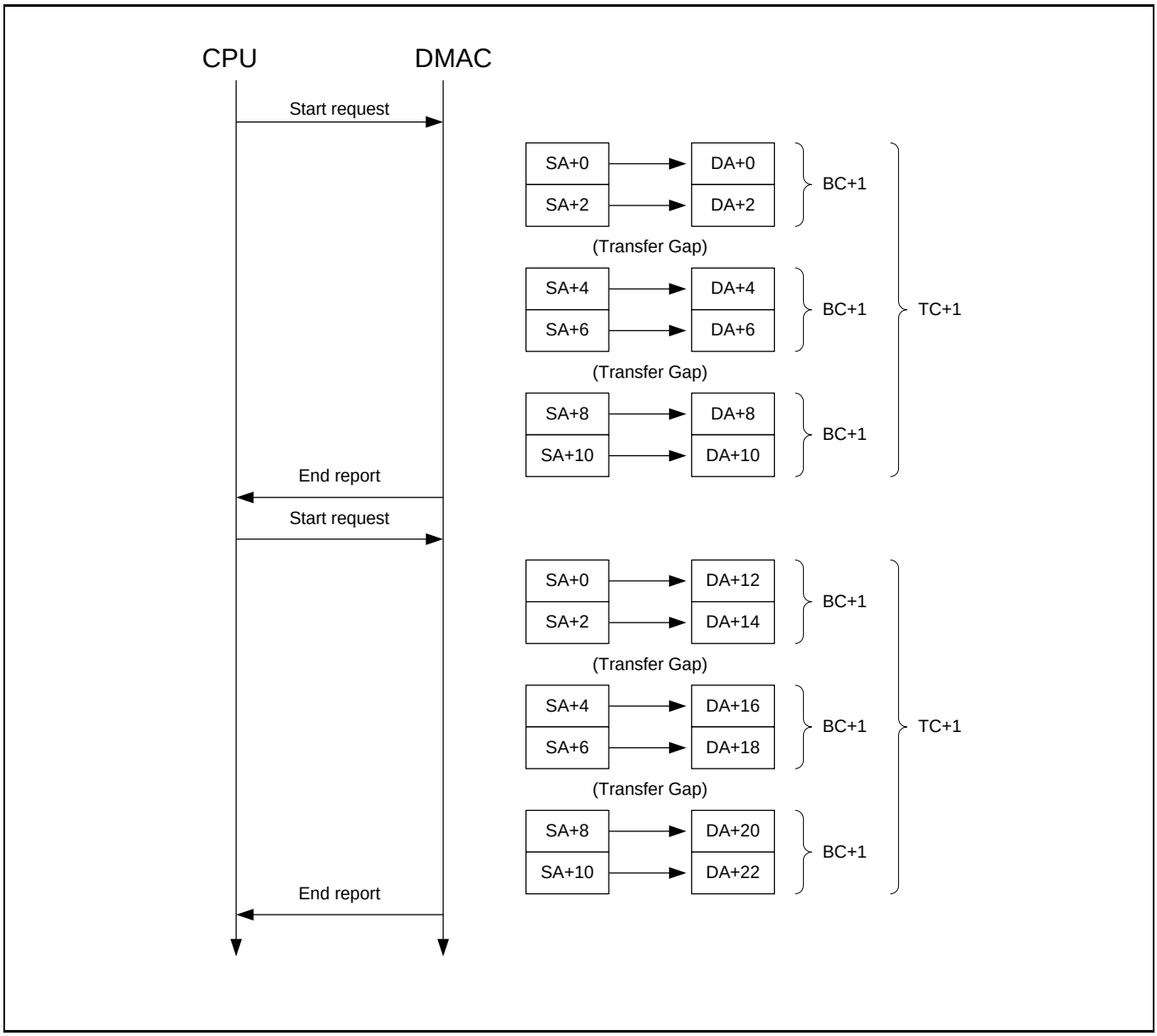
## 3.1 Software-Block Transfer

This section explains Software-Block transfer.

Figure 3-1 shows an example of the operation of Software-Block transfer. In Figure 3-1, the following settings apply.

- Transfer mode : Software request Block transfer (ST=1, IS[5:0]=000000, MS=00)
- Transfer source start address : SA(DMACSA=SA)
- Transfer source address control : Increment and reload available (FS=0, RS=1)
- Transfer destination start address : DA(DMACDA=DA)
- Transfer destination address control : Increment and reload not available (FD=0, RD=0)
- Transfer data size : Half-word (16 bits), the number of blocks = 2, the number of transfers = 3 (TW=01, BC=1, TC=2)
- BC/TC reload : Reload available (RC=1)

**Figure 3-1 Example of Operation of Software-Block Transfer**



DMAC performs the following operation, when the transfer content is set from CPU and then the start of the transfer is instructed.

- Due to the specification of the transfer data width, each transfer is performed by half-word (16 bits).
- According to the start addresses of the transfer source and transfer destination, the data width and the incremented/fixed specification, the transfer is performed in the area from the address SA to address DA, for the number of blocks (=BC+1).
- In the case of Block transfer, a Transfer Gap occurs every time transfer of one block is completed.
- DMAC performs data transfer for the number of blocks (=BC+1) by the number of transfers (=TC+1). The size of data to be transferred by each transfer request from CPU is "Data width (TW) × Number of blocks (BC+1) × Number of transfers (TC+1)".
- Once the transfer is completed, DMAC notifies CPU of the completion.
- If the start of transfer is instructed again after the completion of the transfer, the transfer is restarted from the previous transfer start address (SA+0), because the transfer source address has been set to be reloaded (RS=1). As the transfer destination address has not been specified to be reloaded (RD=0), the transfer is started from the next address (DA+12) after the previous transfer end address. Also, as the reload of BC/TC has been specified, the same values as for the previous transfer are reloaded for the number of blocks and the number of transfers for the next transfer.

Transfer Gap is a time period during which no transfer is performed, and it is inserted to prevent one of the DMAC channels from taking the possession of the system bus access right. If multiple channels have transfer requests, DMAC switches the channels that will perform the transfer operation at the timing of the Transfer Gap. The frequency of Transfer Gap generation can be controlled by adjusting the settings of BC and TC.

Moreover, the bus access right is also passed on to CPU at the Transfer Gap timing. System buses in this product are in Multi-layered configuration with a special system bus dedicated to DMA. For this reason, if there is no conflict between CPU and the destination of access, transfer can be performed at the same time as the CPU operation. Even if there is a conflict between CPU and the destination of access, the CPU operation is little affected, as long as the DMAC transfer is in a different address area group (RAM and Peripheral, or Flash memory and RAM, etc). However, if the transfer is in the same address area group (RAM and RAM, etc.), the CPU operation and/or system performance may be affected, depending on the number of blocks used; therefore, attention must be paid.

("Address area group" mentioned above refers to a group of address areas that are connected on the AHB system bus with the same bus bridge.)

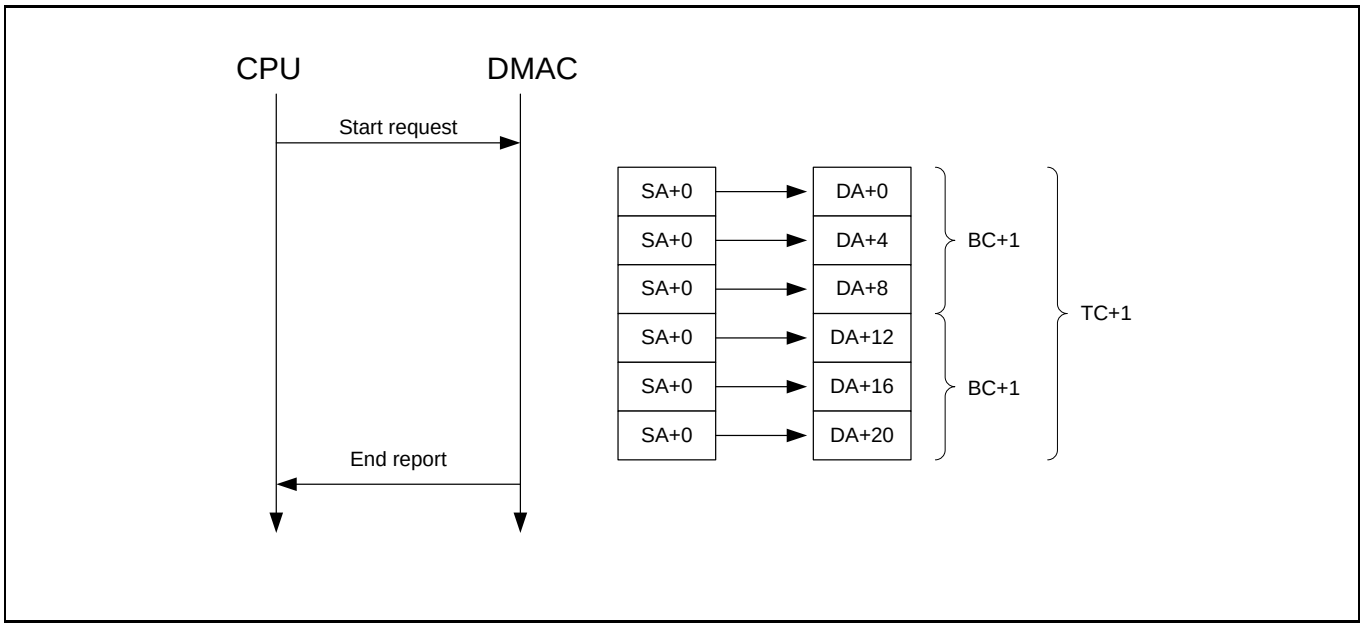
## 3.2 Software-Burst Transfer

This section explains Software-Burst transfer.

Figure 3-2 shows an example of the operation of Software-Burst transfer. In Figure 3-2, the following settings apply.

- Transfer mode : Software request Burst transfer (ST=1, IS[5:0]=000000, MS=01)
- Transfer source start address : SA(DMACSA=SA)
- Transfer source address : Fixed, reload available (FS=1, RS=1)
- Transfer destination start address : DA(DMACDA=DA)
- Transfer destination address : Increment and reload not available (FD=0, RD=0)
- Transfer data size : Word (32 bits), the number of blocks =3, the number of transfers =2 (TW=10, BC=2, TC=1)
- Reload of the number of transfers : Number of transfers to be reloaded (RC=1)

**Figure 3-2 Example of Operation of Software-Burst Transfer**




DMAC performs the following operation, when the transfer content is set from CPU and then the start of the transfer is instructed.

- Due to the specification of the transfer data width, each transfer is performed by word (32 bits).
- According to the start addresses of the transfer source and transfer destination, the data width and the incremented/fixed specification, the transfer is performed in the area from the address SA to address DA, for the number of blocks (=BC+1). As the transfer source address is specified to be fixed, it is the same as the transfer source start address (SA+0).
- In the case of Burst transfer, the transfer is executed continuously without generating Transfer Gaps.
- DMAC performs data transfer for the number of blocks (=BC+1) by the number of transfers (=TC+1). The size of data to be transferred by each transfer request from CPU is "Data width (TW) × Number of blocks (BC+1) × Number of transfers (TC+1)".
- When the transfer is completed, DMAC notifies CPU of the completion.

In the case of Burst transfer, no Transfer Gap is generated, unlike the Block transfer. As the channel to be controlled takes the possession of the system bus access right, it can be used to put the priority on that particular channel.

## 3.3 Hardware-Demand Transfer

This section explains Hardware-Demand transfer.

Hardware-Demand transfer is used when performing DMA transfer by the transfer request signal from the Peripherals of MFS and ADC.

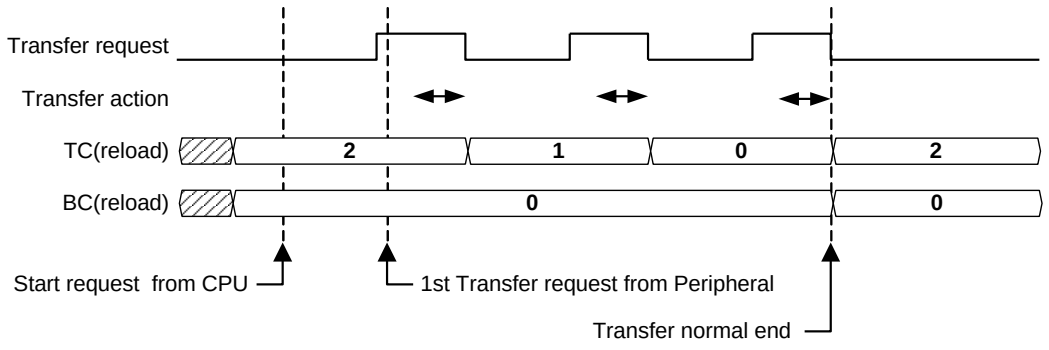
Hardware-Demand transfer is a method used to receive the transfer request signal from Peripherals on a signal level. If the transfer request signal is on High level, transfer is executed. If the transfer request signal is on Low level, no transfer is executed. Transfer is executed by setting the output of the interrupt signal from each Peripheral to High level (with interrupt request) when transfer data exists, or to Low level (without transfer request) when no transfer data exists.

In the case of Hardware-Demand transfer, always specify "1" (BC=0) as the number of blocks.

Figure 3-3 shows an example of the operation of Hardware-Demand transfer. In this example, the following settings apply. The settings of the addresses of the transfer source and transfer destination as well as the transfer data width are omitted.

- Transfer mode : Hardware-Demand transfer
  (ST=0, IS= Peripheral at the transfer request source, MS=10)
- Transfer data size : Number of blocks = 1, Number of transfers = 3 (BC=0, TC=2)

**Figure 3-3 Example of Operation of Hardware-Demand Transfer**




The operation of Hardware-Demand transfer is as follows:

The start of the operation is instructed by specifying the transfer content from CPU. DMAC waits for a transfer request from the Peripheral. After receiving the transfer request, it performs one transfer and then waits for the next transfer request. During the wait period, a Transfer Gap is generated. Every time a transfer request is received, it performs the same operation for the number of transfers (TC+1). The total number of transfers to be performed is (TC+1). Match the number of transfer requests from the Peripheral and the number of DMAC transfers (TC+1). Once all of the transfers are completed, DMAC notifies CPU of the completion.

## 3.4 Hardware-Block Transfer & Burst Transfer

This section explains Hardware-Block transfer and Burst transfer.

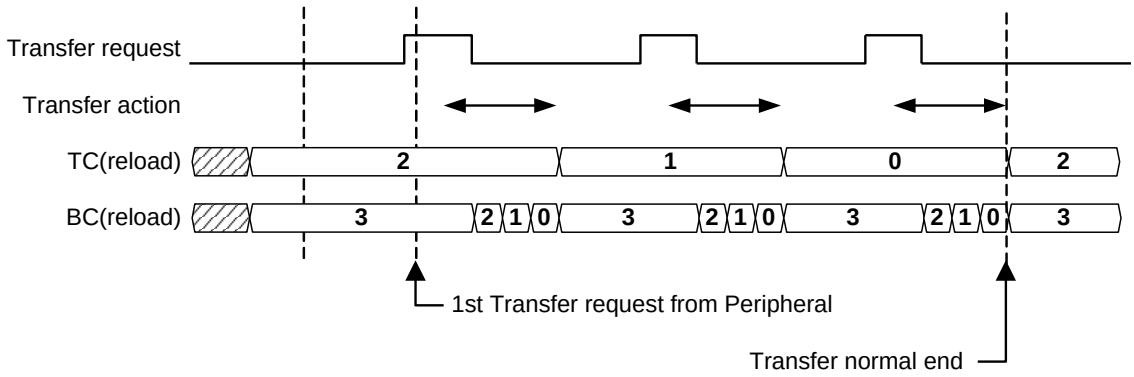
Hardware-Block transfer or Hardware-Burst transfer is used when performing DMA transfer by the transfer request signal from the Peripheral of the base timer or external interrupt.

Hardware-Block transfer and Hardware-Burst transfer are methods used to receive the transfer request signal at the rising edge of the signal. Transfer is executed, when the rising edge of the transfer request signal is detected. DMAC's transfer start timing can be specified by the output of the interrupt signal from each Peripheral.

Figure 3-4 shows an example of the operation of Hardware-Block transfer. In Figure 3-4, the following settings apply. The settings of the addresses of the transfer source and transfer destination as well as the transfer data width are omitted.

- Transfer mode : Hardware-Block transfer
  (ST=0, IS= Peripheral at the transfer request source, MS=00)
- Transfer data size : Number of blocks = 4, Number of transfers = 3 (BC=3, TC=2)

**Figure 3-4 Example of Operation of Hardware-Block Transfer**



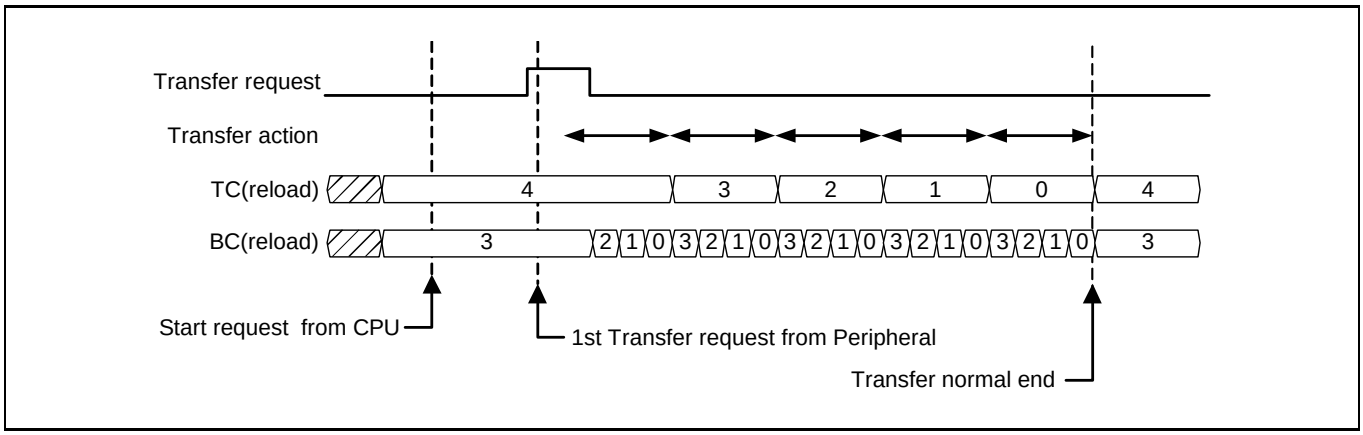

The operation of Hardware-Block transfer is as follows:

The start of the operation is instructed by specifying the transfer content from CPU. DMAC waits for a transfer request from the Peripheral. After receiving the transfer request, it performs transfers for the number of blocks (=BC+1) and then waits for the next transfer request. During the wait period, a Transfer Gap is generated. Every time a transfer request is received, it performs the same operation for the number of transfers (TC+1). The total number of transfers to be performed is (BC+1) × (TC+1). Match the number of transfer requests from the Peripheral and the number of DMAC transfers (TC+1). Once all of the transfers are completed, DMAC notifies CPU of the completion.

Figure 3-5 shows an example of the operation of Hardware-Burst transfer. In Figure 3-5, the following settings apply. The settings of the addresses of the transfer source and transfer destination as well as the transfer data width are omitted.

- Transfer mode : Hardware-Burst transfer
  (ST=0, IS= Peripheral at the transfer request source, MS=01)
- Transfer data size : Number of blocks =4, Number of transfers = 5 (BC=3, TC=4)

**Figure 3-5 Example of Operation of Hardware-Burst Transfer**




The operation of Hardware-Burst transfer is as follows:

The start of the operation is instructed by specifying the transfer content from CPU. DMAC waits for a transfer request from the Peripheral. After receiving the first transfer request, it performs all of the transfers for the number of times calculated by (BC+1) × (TC+1). During the Hardware-Burst transfer, no Transfer Gap is generated. Once all of the transfers are completed, DMAC notifies CPU of the completion.

## 3.5 Channel Priority Control

This section explains the channel priority control.

### Channel Priority Control

If multiple channels have transfer requests, DMAC switches the channel subject to the transfer among them at the timing of the Transfer Gap of each channel. At this point, the next channel to which the transfer will be performed is determined according to the priority control. The priority control can be selected from either fixed priority or rotated priority by the PR. Figure 3-6 shows an explanatory diagram. In this figure, the right axis indicates the time axis. The arrows indicate transfer timings of each channel to perform its transfer operation when all of the channels issue transfer requests simultaneously.

### Operation in Fixed Priority Mode (PR=0)

In fixed priority mode, the channel with the smallest channel number among all the channels with a transfer request has the priority to perform transfer operation.

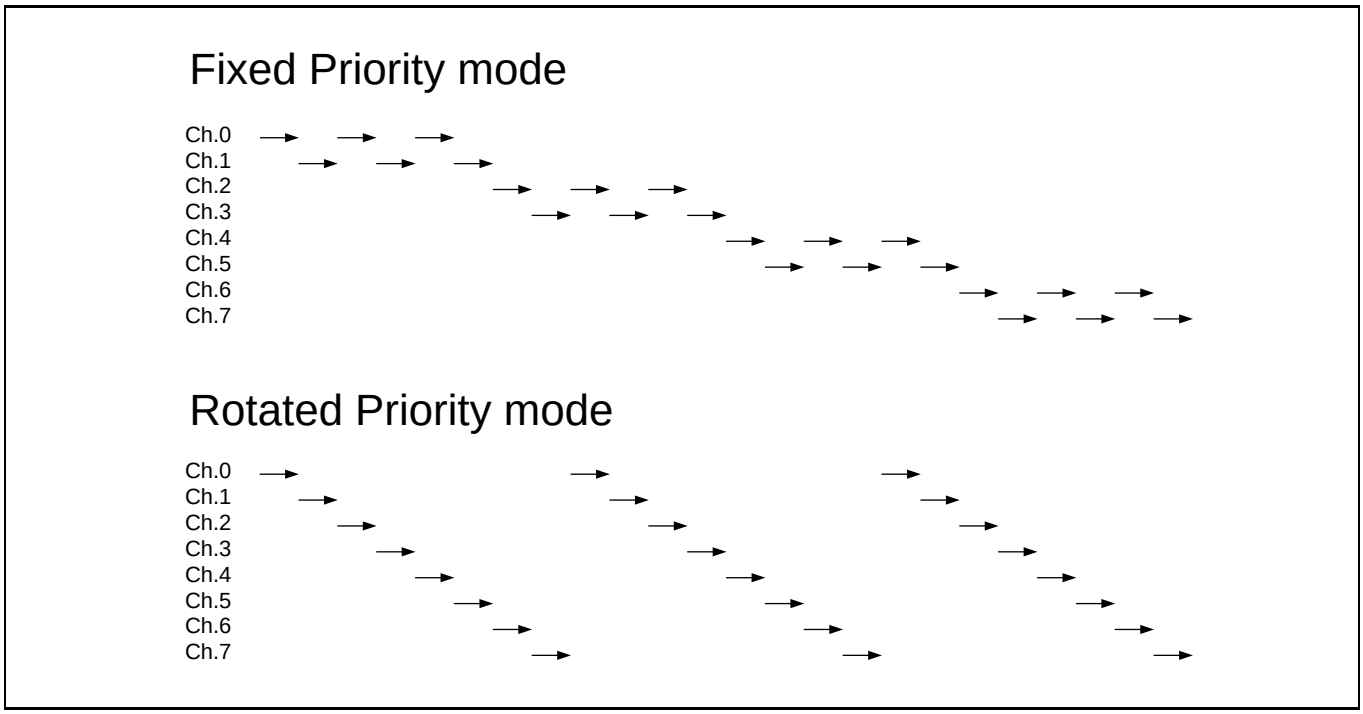
(Priority order: ch.0 > ch.1 > ch.2 > ch.3 > ch.4 > ch.5 > ch.6 > ch.7)

First, the channel with the highest priority performs its transfer (ch.0 in Figure 3-6). As the channel with the highest priority halts the transfer operation at the timing of a Transfer Gap, then, the channel with the second highest priority performs its transfer operation (ch.1 in Figure 3-6). For this reason, the channels with the highest and the second highest priority perform the transfer operations alternately. After that, when the channel with higher priority completes its transfer, the channel with lower priority starts its transfer operation (ch.3 in Figure 3-6).

### Operation in Rotated Priority Mode (PR=1)

In rotate priority mode, all channels perform their transfer operations equally.

**Figure 3-6 Explanatory Diagram of Channel Priority Control**



# 4. DMAC Control

This section explains DMAC control methods in details.

## 4.1 Overview of DMAC Control

This section provides an overview of DMAC control.

The control register of each channel of DMAC has EB (individual-channel operation enable bit) and PB (individual-channel pause bit). By manipulating these bits, the start of DMA transfer operation (operation enabled), the forced termination of transfer operation (operation disabled) and the pause of transfer operation can be controlled by channel. The control register also has DE (all-channel operation enable bit) and DH (all-channel pause bit), which allow the transfer operations of all channels to be controlled at once.

Each channel is originally in the operation-prohibited state (Disable state) in which the transfer content (the address of the transfer source, the address of the transfer destination, the transfer data width, the number of transfers, the transfer mode, etc.) are specified for each channel to its configuration register. Then, the transfer operations are controlled by writing to EB, PB, DE and DH to instruct the transfer operations to be started or paused.

Once each channel completes its transfer, it sets the end code to SS (Stop Status) to give the notification of its stop state. An interrupt can be generated upon the completion of transfer. After the transfer ends, each channel clears EB and PB and returns to the operation-prohibited state (Disable state).

The following sections describe the operations of and control procedures for DMA transfer by software request and hardware DMA transfer by transfer request from Peripherals.
The following terms are used in the explanations as instructions from CPU, which refer to writing the following values to the EB, PB, DE and DH bits.

- Instruction to enable individual-channel operation (write EB=1, PB=0)
- Instruction to disable individual-channel operation (write EB=0)
- Instruction to pause individual-channel operation (write EB=1, PB=1)
- Instruction to enable all-channel operation (write DE=1, DH=0000)
- Instruction to disable all-channel operation (write DE=0)
- Instruction to pause all-channel operation (write DE=1, DH=0000)

## 4.2 DMAC Operation and Control Procedure for Software Transfer

This section explains DMAC operation and control procedure for software transfer.

**Figure 4-1 Transitional Diagram of Software DMA Transfer State**

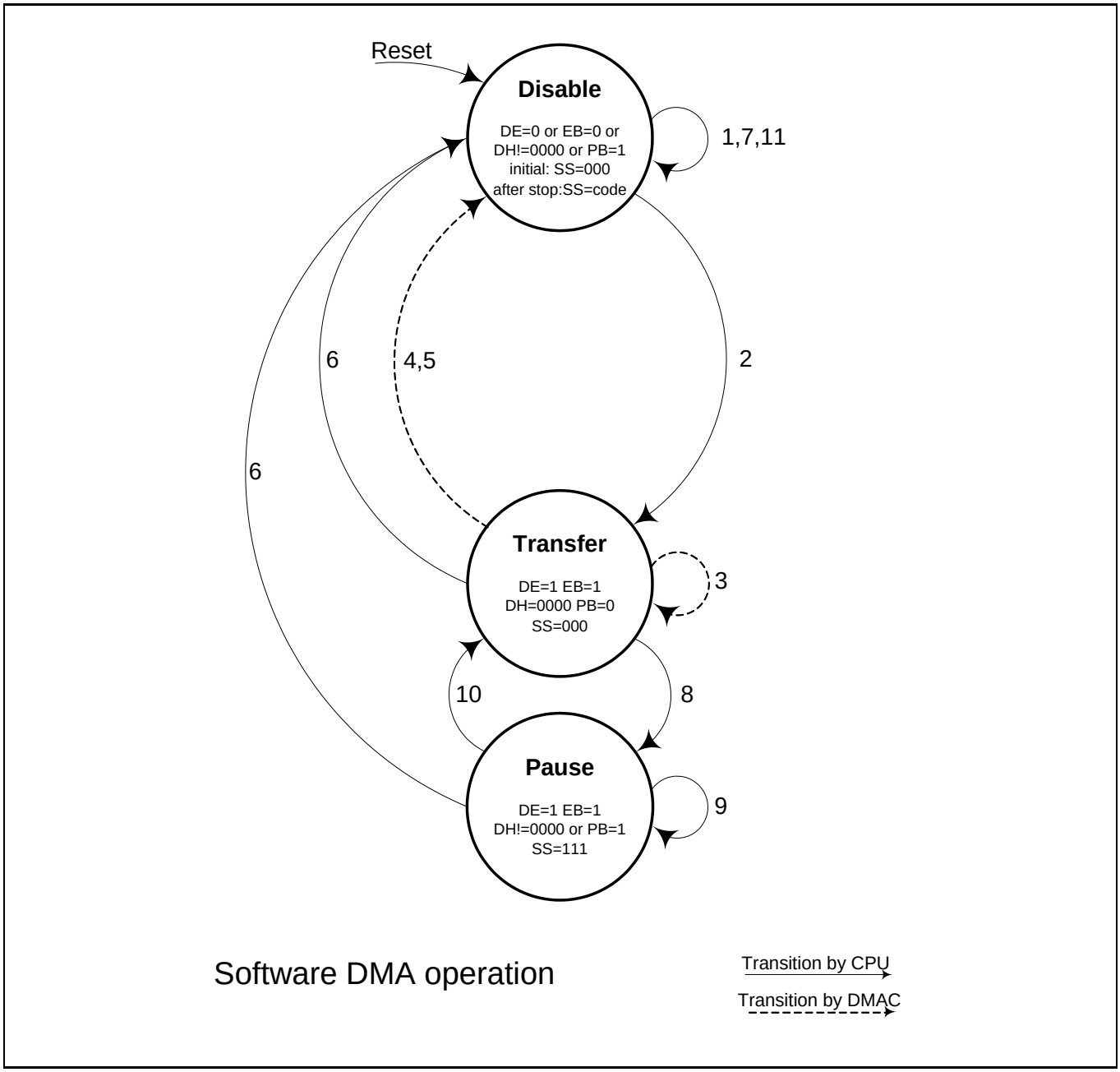



Figure 4-1 shows a transitional diagram of the states of the channel to be controlled for software transfer. The numbers next to the transitional lines in the figure correspond to the numbers which appear in the following control procedures. The solid transitional lines indicate transitions of state instructed by CPU, while the broken transitional lines indicate transitions of state due to DMAC operation.

### Description of Each State

■ Disable state

In this state, the transfer of the channel to be controlled is prohibited. Channels in this state do nothing and wait for instruction from CPU. At the system reset, DE=0, EB=0, DH=0000 and PB=0 apply to this Disable state.

■ Transfer state

In this state, the transfer of the channel to be controlled is enabled. Channels in this state perform transfer operation as specified. Once all of the transfer operations are completed, they return to the Disable state. The state is also changed as instructed by CPU.

■ Pause state

In this state, the channel to be controlled has its transfer operation on pause due to an instruction to pause, issued by CPU, and is waiting for another instruction from CPU.

### Explanation of Control Procedure

1. Disable state / Preparation for transfer

   Specify via CPU the transfer content for the channel to be controlled (writing to DMACSA, DMACDA, DMACA and DMACB). For details of transfer content to be specified, see the section describing register functions. When generating an interrupt from DMAC upon the completion of transfer, set EI and CI.

   The following restrictions apply to software transfer. Specify ST=1 and IS[5:0]=000000. Demand transfer mode cannot be specified to MS. Always set "0" to EM.

   Give an instruction to enable all-channel operation and set PR. Data can also be written to DMACA at the same time in Step 2.

2. Disable state => Transfer state / Start of transfer

   Give an instruction to enable individual-channel operation from CPU. When DE=1, EB=1, DH=0000 and PB=0 are set, the channel to be controlled moves to Transfer state.

3. Transfer state

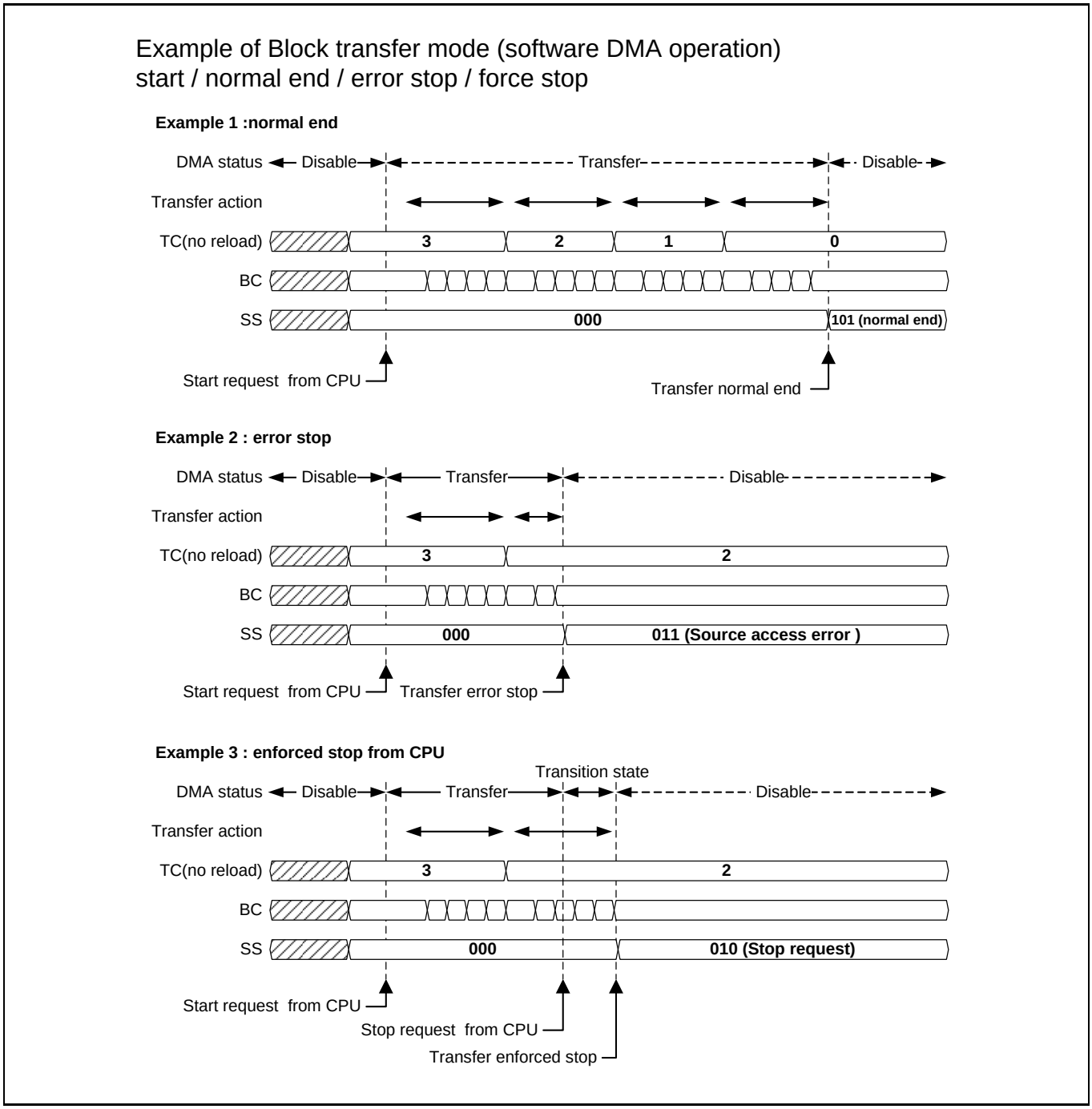
   When the channel in Transfer state becomes enabled to access the system bus, it performs a transfer according to the transfer content (it may take time to start the transfer, depending on the status of other channels). In the case of Block transfer, a Transfer Gap is generated every time TC is updated. In the case of Burst transfer, no Transfer Gap is generated. During the transfer operation, BC, TC, DMACSA and DMACDA indicate the remaining number of transfers and the transfer address at that time point. The transfer status can be checked by reading from CPU.

   The specified transfer content cannot be changed via CPU to the channel in Transfer state (rewriting to DMACSA, DMACDA, DMACA[29:0], DMACB[31:1]). (However, EB, PB and EM can be rewritten.)

4. Transfer state => Disable state / Successful completion of transfer

   When transfers are successfully completed for the number of times calculated by (BC+1) × (TC+1), the channel in Transfer state clears EB, PB and ST and moves to Disable state. It sets SS=101 to provide the notification of the successful completion. See Example 1 in Figure 4-2. If successful transfer completion interrupt has been enabled by CI, an interrupt occurs. If reload has been specified to BC, TC, DMACSA and DMACDA, such reload is executed according to the specified transfer content.

**Figure 4-2 Example of Operation of Software-Block Transfer**




5. Transfer state => Disable state / Transfer error stop

The channel in Transfer state suspends the transfer process, if an address overflow, transfer source access error or transfer destination access error occurs. It clears EB, PB and ST and moves to Disable state. It sets the value that indicates the error content to SS[2:0] to give the notification of the error stop. See Example 2 in Figure 4-2. If unsuccessful transfer completion interrupt has been enabled by EI, an interrupt occurs. BC, TC, DMACSA and DMACDA to which reload has not been specified hold the values set at the time of the transfer suspension.

Normally, a transfer error occurs, when an attempt is made to access an address area that does not exist in the system bus or an address area that prohibits access from DMAC. No such error occurs in general applications.

6. Transfer state, Pause state => Disable state / Forced transfer stop

   If an instruction to disable individual-channel operation or an instruction to disable all-channel operation is issued from CPU to a channel in Transfer state or Pause state, the transfer operation of that channel can be forced to stop (for the operation when an instruction to disable operation is issued to a channel in Disable state, see Step 11 in the software procedure).

   If an instruction is given from CPU, the relevant channel suspends its transfer process. It clears EB, PB and ST and moves to Disable state. It sets SS[2:0]=010 and gives the notification that the transfer of that channel has been forced to stop. If unsuccessful transfer completion interrupt has been enabled by EI, an interrupt occurs. BC, TC, DMACSA and DMACDA to which reload has not been specified hold the values set at the time of the transfer suspension.

   After instructed from CPU, the transfer stops at the timing when the relevant channel is not performing transfer (in Transfer Gap before the transfer starts), as shown in the Example 3 in Figure 4-2. In the case of a channel in Pause state, the transfer stops immediately. There is a time difference (Transition state) between the instruction and the stop. It may take some time, depending on the BC setting. As a new transfer cannot be set or started during this period, always make sure that the operation has stopped before setting the next transfer.

   In the case of an instruction to disable all-channel operation, the timing to stop varies depending on the channel. As DS is set when all of the channels are stopped, it can confirm that all of the channels have stopped.

   Even if instructed from CPU, the transfer may not be forced to stop, and instead, it may be successfully completed due to factors such as transfer mode (Burst/Block/Demand) and transfer status (the number of transfers performed, the timing of instruction to disable the operation). Also, if a transfer error occurs before the transfer stops, error stop applies to the transfer.

7. Disable state / Post-transfer process

   SS is read from CPU to check the state of completion of the transfer. CPU clears SS to prepare for the next transfer. If interrupts have been enabled, the interrupt signal from DMAC is deasserted by clearing SS.

   In the case of successful completion, CPU resets the transfer content, as required. If each reload has been specified, the values set before the start of the transfer are reloaded to BC, TC, DMACSA and DMACDA. If each reload has not been specified, BC and TC are initialized to "0". DMACSA and DMACDA show the address for the next transfer.

   In the cases of error stop and forced stop, BC, TC, DMACSA and DMACDA must always be reset, because they may have the values set at the time of the suspension.

   If the transfer is stopped due to an instruction to disable all-channel operation, DE is set to "0"; therefore, the next transfer will require an instruction to enable all-channel operation and an instruction to enable individual-channel operation.

8. Transfer state / Transfer pause

   If an instruction to put individual-channel operation on pause or an instruction to put all-channel operation on pause is issued from CPU to a channel in Transfer state, the transfer operation of the relevant channel(s) can be put on pause (for the operation when an instruction to put the operation on pause is issued to a channel in Disable state, see Step 11 in the software procedure).

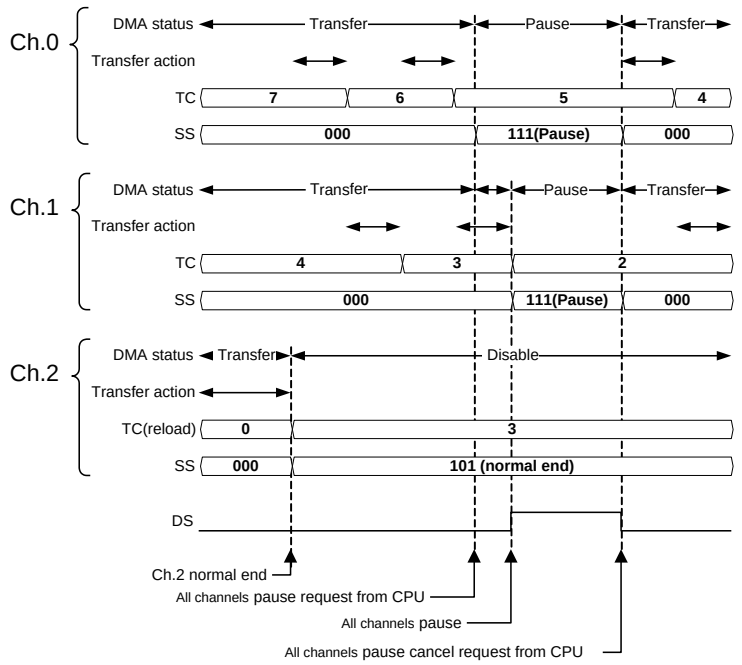
   If an instruction is given from CPU, the relevant channel(s) temporarily suspends the transfer process. It sets SS=111 and gives the notification that it is in Pause state. In this case, no interrupt can be generated.

   After instructed from CPU, the transfer stops at the timing when the relevant channel is not performing transfer (in Transfer Gap before the start of the transfer). There is a time difference (Transition state) between the instruction and the stop. It may take some time, depending on the BC setting. See Figure 4-3.

   In the case of an instruction to put all-channel operation on pause, the timing to stop varies depending on the channel. As DS is set when all of the channels are stopped, it can confirm that all of the channels have stopped. See Figure 4-3.

Even if instructed from CPU, the transfer may not be put on pause, and instead, it may be successfully completed due to factors such as transfer mode (Burst/Block/Demand) and transfer status (the number of transfers performed, the timing of instruction to put the operation on pause). Also, if a transfer error occurs before the transfer stops, error stop applies to the transfer.

**Figure 4-3 Operation when All-channel Pause is Instructed**




9. Pause state

SS is read from CPU to confirm the pause of the transfer. The SS of a channel in Pause state is "111". While in this state, it cannot be cleared from CPU.

Even during the pause, the transfer content cannot be specified or changed (writing DMACSA, DMACDA, DMACA[29:0] or DMACB[31:1]). Also, when a channel in Pause state is instructed to pause, it continues to remain in the Pause state.

10. Pause state / Cancellation of transfer pause

If an instruction to enable individual-channel operation is issued to a channel that has been in Pause state due to an instruction to put individual-channel operation on pause, that channel returns to Transfer state. If an instruction to enable all-channel operation is issued to channels that have been in Pause state due to an instruction to put all-channel operation on pause, those channels return to Transfer state. If both of the pause instructions have been given, issue an instruction to cancel both of them.

After the instruction, SS[2:0] is cleared to "000" via DMAC.

If an instruction to enable individual-channel operation and an instruction to enable all-channel operation are issued in Pause state, they instruct the pause to be cancelled. If they are issued in Disable state, attention must be paid, as they may instruct a new transfer to be started. See Step 11 in the software procedure.

Figure 4-3 shows an example of the case where an instruction to put all-channel operation on pause. The explanation of the figure is as follows.

At the beginning, three channels, namely ch.0, ch.1 and ch.2, perform their transfer operations in Block transfer mode. ch.2 successfully completes its transfer, moves to Disable state and sets SS=101. Then, ch.0 and ch.1 perform transfers alternately.

If an instruction to put all-channel operation on pause is issued from CPU at this point, the following operation applies. As ch.0 is subject to the Transfer Gap timing, it immediately moves to Pause state and sets SS=111. As ch.1 is in the middle of transfer operation, it performs the transfer until the timing of the next Transfer Gap, and then moves to Pause state and sets SS=111. As ch.2 is in Disable state, it remains in the Disable state without changing SS. DS is set, when all of the channels stop their operations.

Next, if an instruction to enable all-channel operation (instruction to cancel the pause) is issued from CPU, the following operation applies. ch.0 and ch.1 return to Transfer state and clear SS[2:0] to "000". As ch.2 is in Disable state (DE=1, EB=0), it remains in that state without starting the operation. Because the pause of all of the channels has been cancelled now, DS is reset.

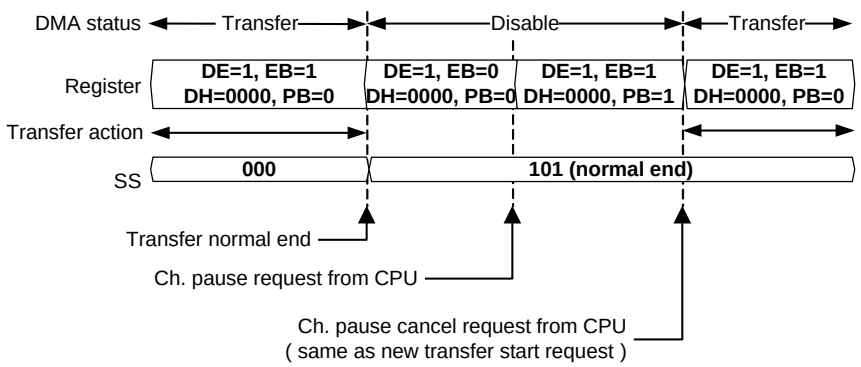
11. Operation in Disable state

A channel in Disable state remains in the Disable state, unless the conditions such as DE=1, DH=0000, EB=1, and PB=0 are established. Although in 1-2 of the software procedure, DE is set from the conditions of DE=0 and EB=0, and then, EB is set, there is no problem to set EB before DE. DE can be set last after all of the transfer settings of multiple channels subject to transfer are completed. In this case, an instruction can be issued to allow the multiple channels subject to transfer to start their transfer operations simultaneously. If such instruction for simultaneous start of transfers is issued, DMAC selects the channels to which transfers are to be started, according to the PR setting (PR can be set or changed, only when all-channel operation is disabled).

If an instruction to disable individual-channel operation, an instruction to put individual-channel operation on pause, an instruction to disable all-channel operation or an instruction to put all-channel operation on pause is issued to a channel in Disable state, only the settings of DE, DH, EB and PB are changed, but the conditions of DE=1, DH=0000, EB=1 and PB=0 are not established. Therefore, the relevant channels do nothing and do not change SS. If an instruction to put all-channel operation on pause is issued from CPU to a channel in Disable state, as shown in the example of ch.2 operation in Figure 4-3, that channel does not change its state with SS[2:0] indicating the completion of the previous transfer.

If an instruction to put individual- or all-channel operation on pause is issued to a channel in Disable state, it may be put in Disable state with DE=1, EB=1, (DH=0000 or PB=1). Although the bit values in this state are the same as DE, EB, DH and PB, they can be distinguished because SS[2:0] has a different value. Figure 4-4 shows such an example.

**Figure 4-4 Example of Operation when Instruction to Put Individual-channel Operation on Pause is Issued in Disabled State**




A certain channel is performing transfer operation. CPU issues an instruction to put individual-channel operation on pause to that channel. The instruction is issued after the transfer is completed and it moves to Disable state (DE=1, DH=0000, EB=0, PB=0). This phenomenon can occur, because the channel currently performing transfer operation changes its state outside CPU's intention. In this case, the bit values of the relevant channel change to (DE=1, DH=0000, EB=1, PB=1) due to instruction from CPU, but SS[2:0]

remains "101", the value set upon the completion. If the operation is stopped by a pause instruction, SS[2:0] will be "111"; therefore, it will be possible to distinguish between the pause state and the state in which the transfer has been completed. It should be noted that if an instruction to cancel the pause is issued without checking the state of the channel by SS[2:0], a new transfer will accidentally start, as shown in Figure 4-4.

■ Additional Matter 1

As ST is cleared upon the completion of a transfer, the read value of ST is "0" after the completion of the transfer. In the case of software transfer, it should be noted that "1" must always be written to ST, regardless of its read value.

■ Additional Matter 2

An instruction to enable individual-channel operation cannot be issued during the period after the previous instruction to enable individual-channel operation instructs the start of transfer and before the completion of the transfer is confirmed. This is because the channel to be controlled may change its state outside CPU's intention and an instruction to start a new transfer may be issued when DMAC has moved to Disable state (EB=0). Even if the SS[2:0] value confirms that the channel to be controlled is in Transfer state, the channel to be controlled may move to Disable state during the period between that point and the write operation.

■ Additional Matter 3

The DE and DH values can only be rewritten from CPU and these registers are never cleared from DMAC. Therefore, there is no problem to write DE=1 and DH=0000 during the transfer operation.

DH is not cleared, if an instruction to disable individual-channel operation is issued to a channel in all-channel Pause state (DE=1, DH!=0000, EB=1, PB=0). After the instruction, the relevant channel moves to Disable state (DE=1, DH!=0000, EB=0, PB=0). To start a new transfer of the relevant channel, write DE=1 and DH=0000. This indicates that the cancellation of the pause of all-channel operation is required in order to start a new transfer of the individual channel.

■ Additional Matter 4

The SS[2:0] value is set from DMAC upon the completion of a transfer and it is never rewritten from DMAC as long as it is in Disable state. Even if the SS[2:0] value is not cleared, the next transfer can be started. However, if it moves to Transfer state, the SS[2:0] value may be cleared from DMAC (or may not be cleared). When an interrupt from DMAC is used, it should be noted that the interrupt signal is deasserted at a timing which is not intended by CPU, if it moves to Transfer state without clearing SS[2:0].

## 4.3 DMAC Operation and Control Procedure for Hardware (EM=0) Transfer

This section explains DMAC operation and control procedure for hardware (EM=0) transfer.

**Figure 4-5 Transitional Diagram of Hardware (EM=0) Transfer State**

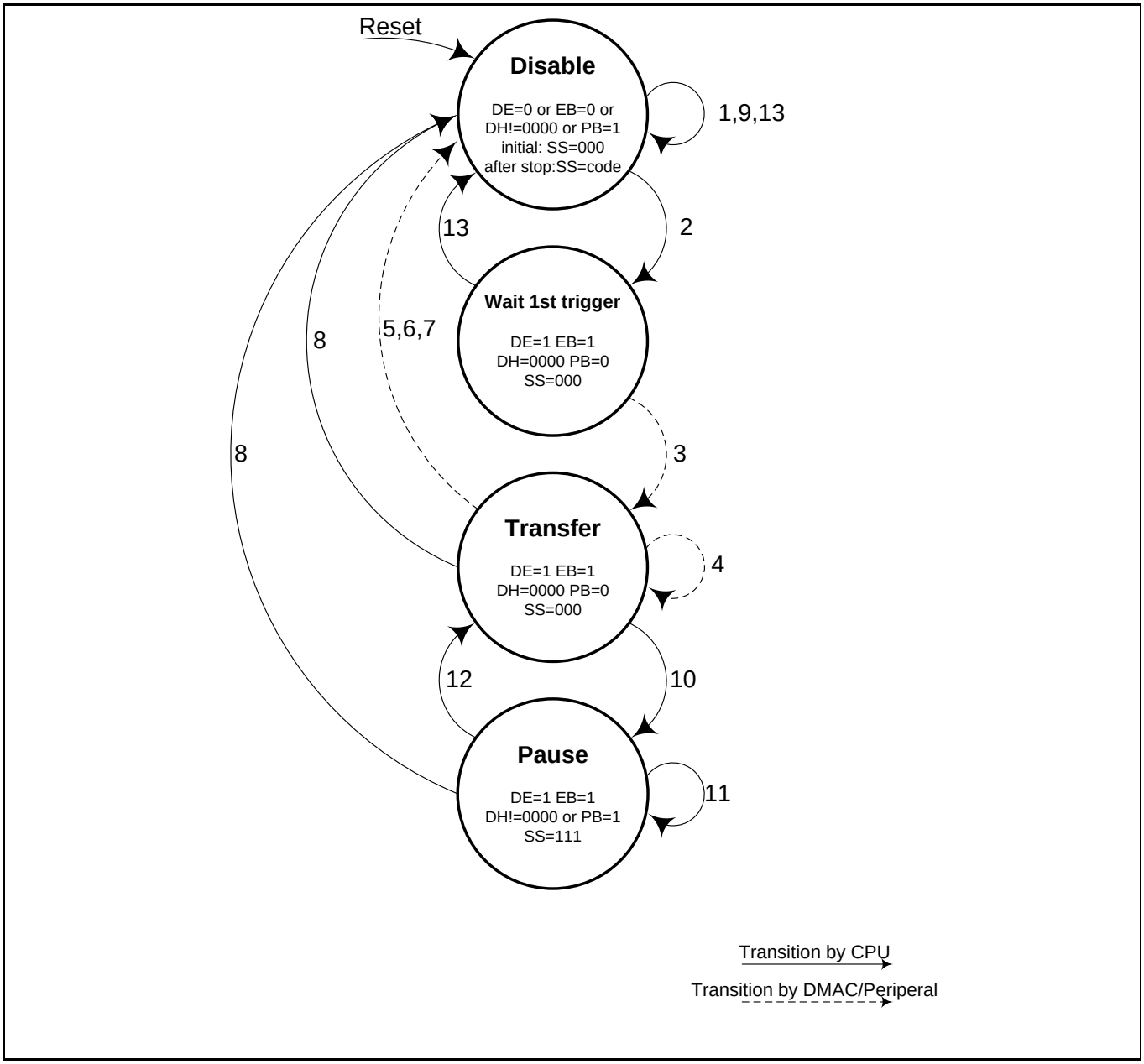



Figure 4-5 shows a transitional diagram of the states of the channel to be controlled for hardware (EM=0) transfer. The numbers next to the transitional lines in the figure correspond to the numbers which appear in the following control procedures. The solid transitional lines indicate transitions of state instructed by CPU, while the broken transitional lines indicate transitions of state due to DMAC/Peripheral operation.

Some parts of the explanation below state "See the software transfer procedure". This means that where the same control as in the software transfer procedure applies, no special mentioning is required; therefore, such redundant explanation has been omitted. In this example, the explanation assumes that EM=0 is set.

### Description of Each State

- Disable state

See Explanation of Control Procedure in "4.2 DMAC Operation and Control Procedure for Software Transfer".

- Wait-1st-trigger state

In this state, the channel to be controlled is enabled to perform transfer. A channel in this state waits for the first transfer request from a Peripheral to be asserted. It also changes its state upon instruction from CPU.

- Transfer state

In this state, the channel to be controlled has received the first transfer request from the Peripheral. A channel in this state performs transfer operation as specified. Once all the transfer operation is completed, it returns to Disable state. It also changes its state upon instruction from CPU.

- Pause state

See Explanation of Control Procedure in "4.2 DMAC Operation and Control Procedure for Software Transfer".

### Explanation of Control Procedure

1. Disable state / Preparation for transfer

See Step 1 in the software transfer procedure.

The following restrictions apply to hardware transfer. Decide in advance on which Peripheral's interrupt signal to be used as the transfer request signal to DMAC using the interrupt controller block (see the section on the functional explanation). Set ST=0 and specify which Peripheral's transfer request to be processed at the channel that will perform the transfer, by IS at the same time. Multiple channels cannot process transfer request of the same Peripheral. In the case of Demand transfer mode, set BC=0. This section explains the operation when EM=0 is set.

2. Disable state => Wait-1st-trigger state / Transfer enabled

An instruction to enable individual-channel operation is issued from CPU. When DE=1, EB=1, DH=0000 and PB=0 are set, the channel to be controlled moves to Wait-1st-trigger state.

3. Wait-1st-trigger state / Start of transfer

The channel in Wait-1st-trigger state is waiting for the transfer request signal to be asserted from the Peripheral or for an instruction from CPU. When the first transfer request signal is asserted, it moves to Transfer state.

4. Transfer state

See Step 3 in the software transfer procedure.

In the case of hardware transfer, a channel in Transfer state performs transfer operation by the transfer request signal from a Peripheral, as described in Sections 3.3 and 3.4. In each mode, match the number of transfer requests from the Peripheral with the number of transfer requests required by DMAC. Below is the explanation for the operation when the number of transfer requests goes over or below the requirement in each operation mode.

Figure 4-6 shows a case of Demand transfer. In the case of Demand transfer, the number of transfer requests required to complete the transfer is TC+1. Unless the number of transfer requests goes over or below the requirement, CPU does not have to intervene (Example 1 in Figure 4-6).

If the number of transfer requests generated from the Peripheral exceeds the DMAC's setting for the number of transfers, DMAC moves to Disable state after the completion of the specified number of transfers. In the Disable state, no further transfers are executed. Also, as the excessive transfer request signals are not cleared from DMAC, the asserted state continues (Example 2 in Figure 4-6).

If the number of transfer requests generated from the Peripheral is smaller than DMAC's setting for the number of transfers, DMAC waits for the remaining number of transfer requests in Transfer state (Example 3 in Figure 4-6).

It is supposed that DMAC's transfer processing maybe too slow to catch up with the generation interval of transfer requests from Peripheral. In the case of Demand transfer, the transfer request signal remains asserted; therefore, as many as TC+1 of transfers can be performed (Example 4 in Figure 4-6).

**Figure 4-6 Operation of Hardware-Demand Transfer**

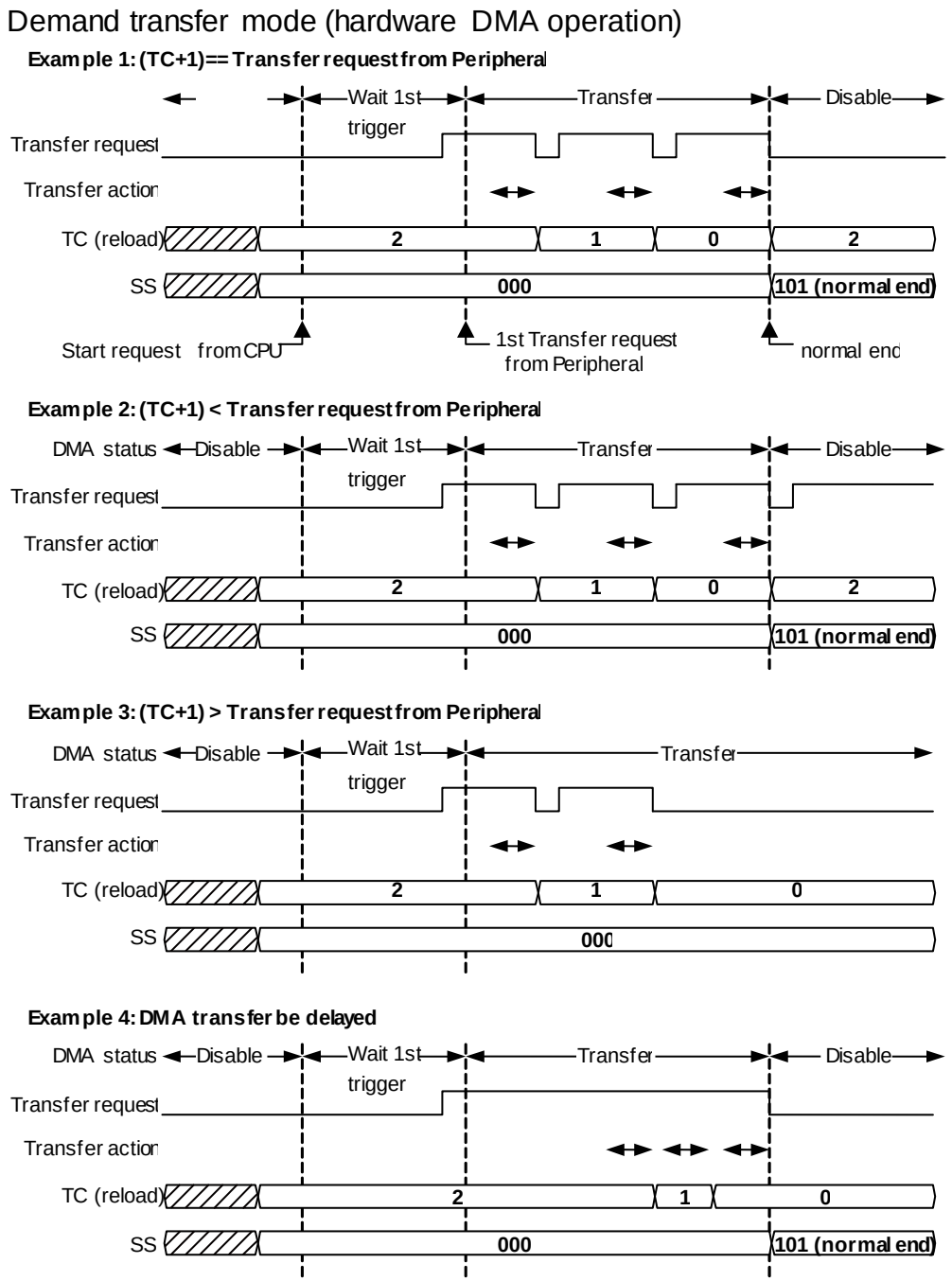
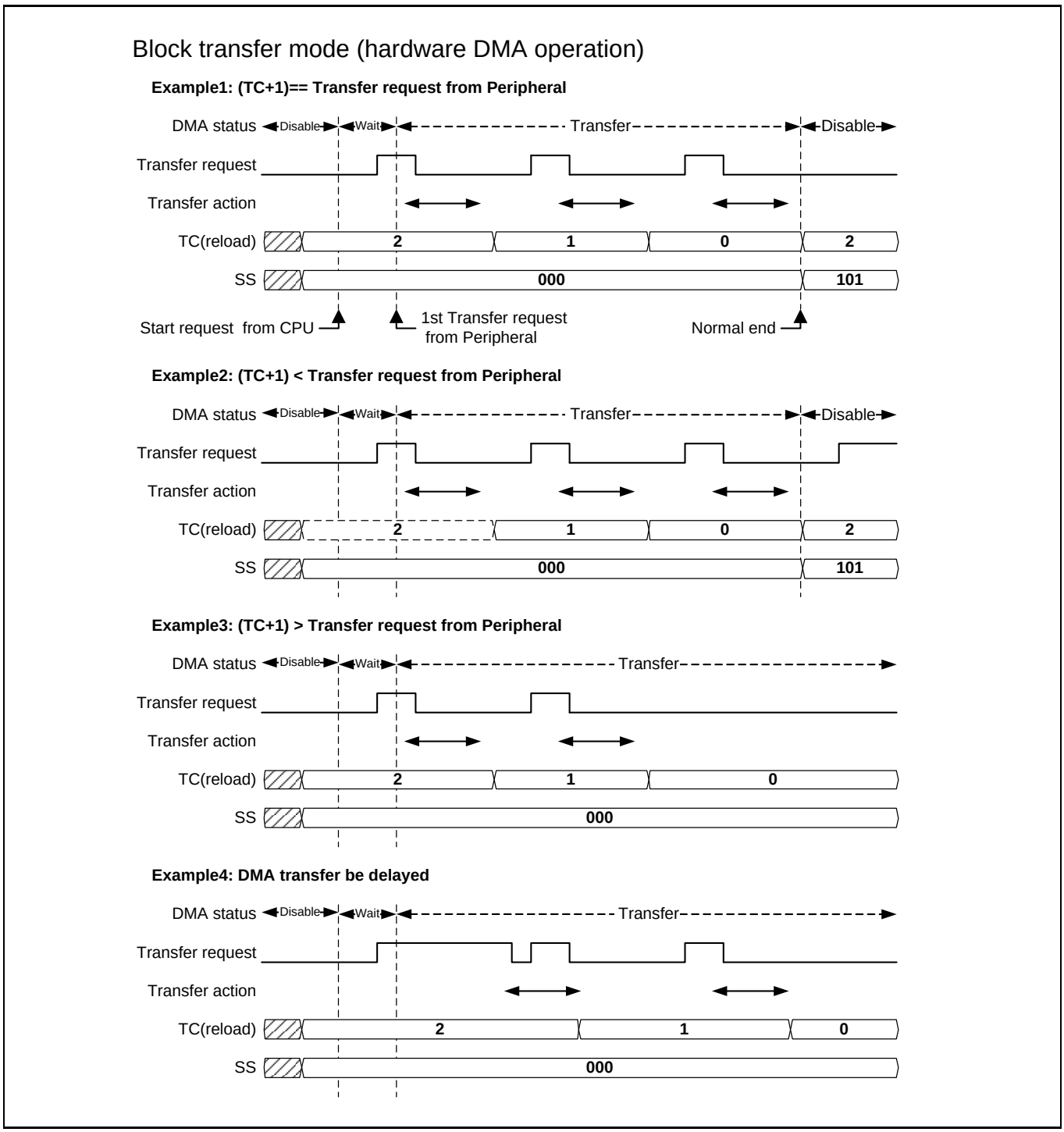


Figure 4-7 shows a case of Block transfer. In the case of Block transfer, the number of transfer requests required to complete the transfer is TC+1. Unless the number of transfer requests goes over or below the requirement, CPU does not have to intervene (Example 1 in Figure 4-7).

**Figure 4-7 Operation of Hardware-Block Transfer**




If the number of transfer requests generated from the Peripheral exceeds the DMAC's setting for the number of transfers, DMAC moves to Disable state after the completion of the specified number of transfers. In the Disable state, no further transfers are executed. Also, as the excessive transfer request signals are not cleared from DMAC, the asserted state continues, In this case, deassert the transfer request signal from CPU (Example 2 in Figure 4-7).

If the number of transfer requests generated from the Peripheral is smaller than DMAC's setting for the number of transfers, DMAC waits for the remaining number of transfer requests in Transfer state (Example 3 in Figure 4-7).

It is supposed that DMAC's transfer processing maybe too slow to catch up with the generation interval of transfer requests from Peripheral. In the case of Block transfer, if DMAC's transfer processing is delayed from the transfer request from the Peripheral, the rising edge of the next transfer request signal during the transfer operation is ignored. Also, the transfer request signal asserted during the transfer operation is cleared from DMAC. Then, DMAC waits for the remaining transfer requests in Transfer state (Example 4 in Figure 4-7).

In the case of Burst transfer, all of the (BC+1) × (TC+1) of transfers are performed when it becomes accessible to the system bus after the first transfer request is received. The required number of transfer requests from the Peripheral is only the first one. If the number of transfer request signals generated exceeds the requirement, it is ignored in Disable state, just like Block transfer.

5. Transfer state => Disable state / Successful completion of transfer

   See Step 4 in the software transfer procedure.

6. Transfer state => Disable state / Transfer error stop

   See Step 5 in the software transfer procedure.

7. Transfer state => Disable state / End of Peripheral stop request

   The channel in Transfer state suspends its transfer processing, if the transfer stop request signal is asserted from the Peripheral. It clears EB, PB and ST and moves to Disable state. It sets "010" to SS[2:0] and gives the notification of the error stop. If interrupts have been enabled by EI, an unsuccessful transfer completion interrupt occurs. BC, TC, DMACSA and DMACDA to which reload has not been specified hold the values set during the suspension of the transfer. Attention must be paid to the SS[2:0] value, which is the same as the stop request from software.

8. Transfer state, Pause state => Disable state / Forced termination of transfer

   See Step 6 in the software transfer procedure.

9. Disable state / Post-transfer processing

   See Step 7 in the software transfer procedure.

   Normally, in the cases of stop request from Peripherals, forced termination from software and transfer error stop, the transfer request signal remains asserted, because the number of transfers processed is smaller than the number of transfer requests from the Peripheral. Instruct from CPU the Peripheral to deassert the transfer request signal. In the case of stop request from Peripherals, the transfer request signal is masked as long as the stop request signal is asserted. Also deassert the transfer stop request signal.

   Even if DMAC has successfully completed the specified number of transfers, the transfer request signal may remain asserted or maybe reasserted, depending on Peripheral's settings. Attention must be paid to the possibility that this may affect the next transfer.

10. Transfer state, Pause state / Transfer pause

    See Step 8 in the software transfer procedure.

11. Pause state

    See Step 9 in the software transfer procedure.

    The channel in Pause state does not execute transfer, even if the transfer request signal from the Peripheral is asserted. It does not clear the transfer request signal either.

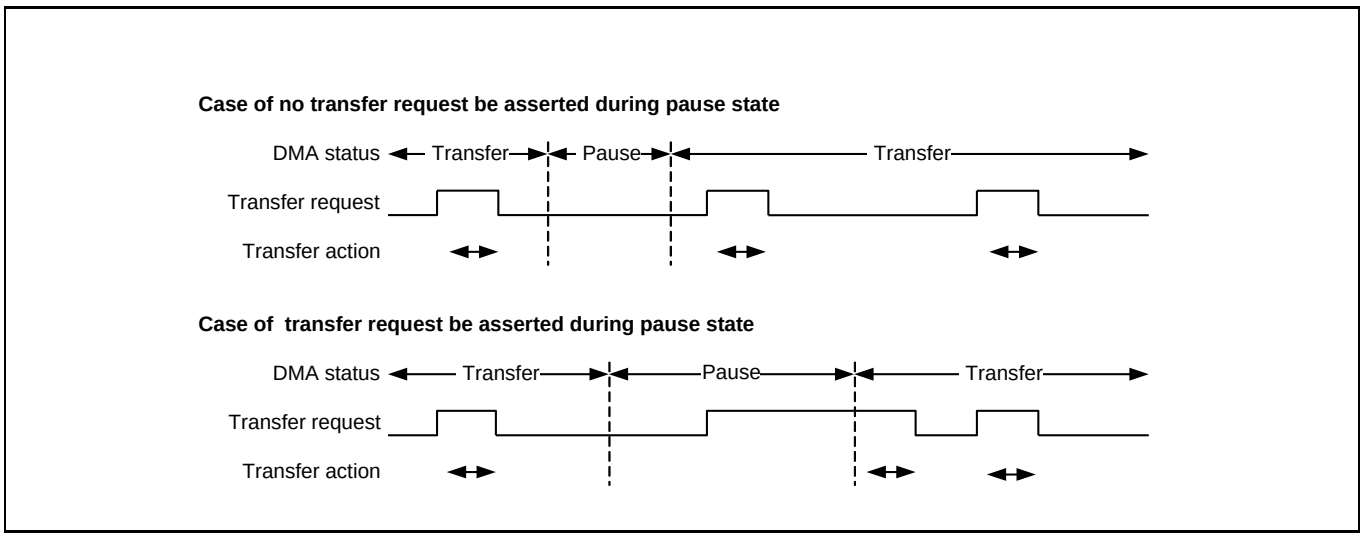
12. Pause state / Cancellation of transfer pause

    See Step 10 in the software transfer procedure.

    When an instruction to cancel the pause is issued while it is in Pause state, it returns to Transfer state. If the transfer request signal was asserted in the previous Pause state, the operation to follow varies as shown below, depending on the transfer mode.

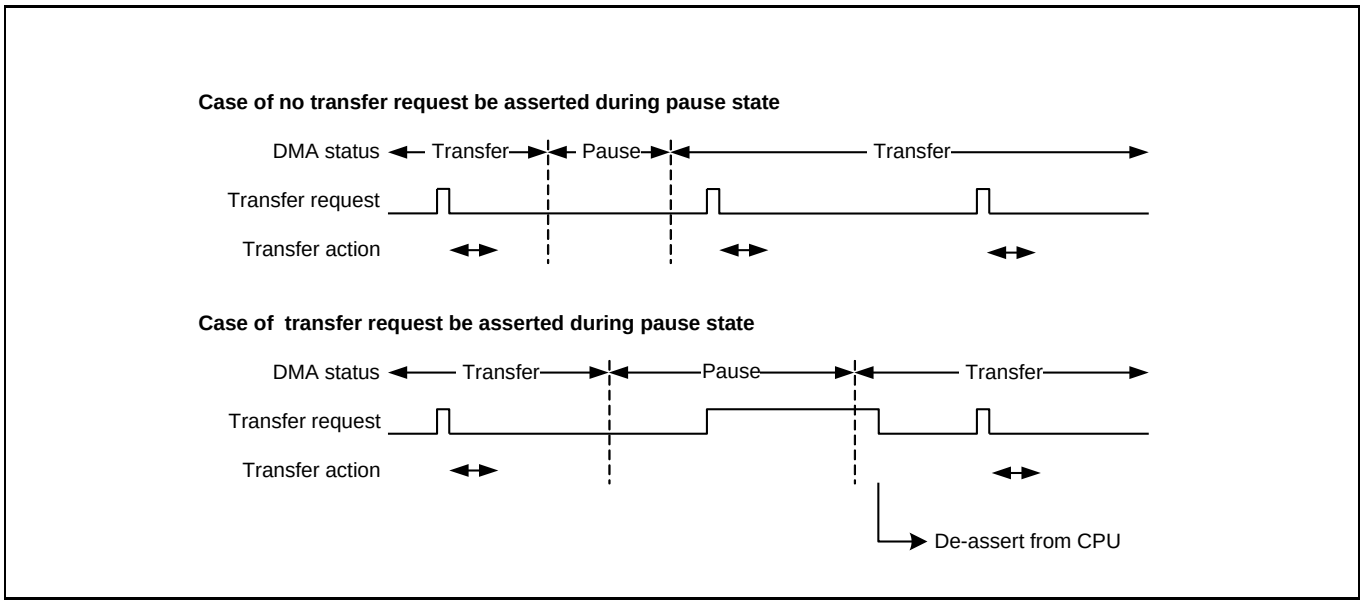
In the case of Demand transfer mode, the transfer request signal remains asserted from the Pause state. Therefore, the transfer is resumed when DMAC returns to Transfer state, and the transfer request signal is cleared as normal. See Figure 4-8.

**Figure 4-8 Operation of Demand Transfer in Pause State**




In the case of Block transfer mode, the transfer request signal remains asserted. Even when it returns to Transfer state, the rising edge of the transfer request signal is not detected, and the transfer is not resumed. Therefore, the transfer request is ignored during Pause state. Also, the transfer request signal is not cleared from DMAC. To resume the transfer which has been put on pause, instruct from CPU the Peripheral to deassert the transfer request signal after an instruction to cancel the pause is issued to DMAC. After that, the transfer will be resumed when the next transfer request is generated from the Peripheral. In this case, attention must be paid to the difference between the number of transfer requests output from the Peripheral and the number of transfer requests received by DMAC. See Figure 4-9.

**Figure 4-9 Operation of Block Transfer in Pause State**



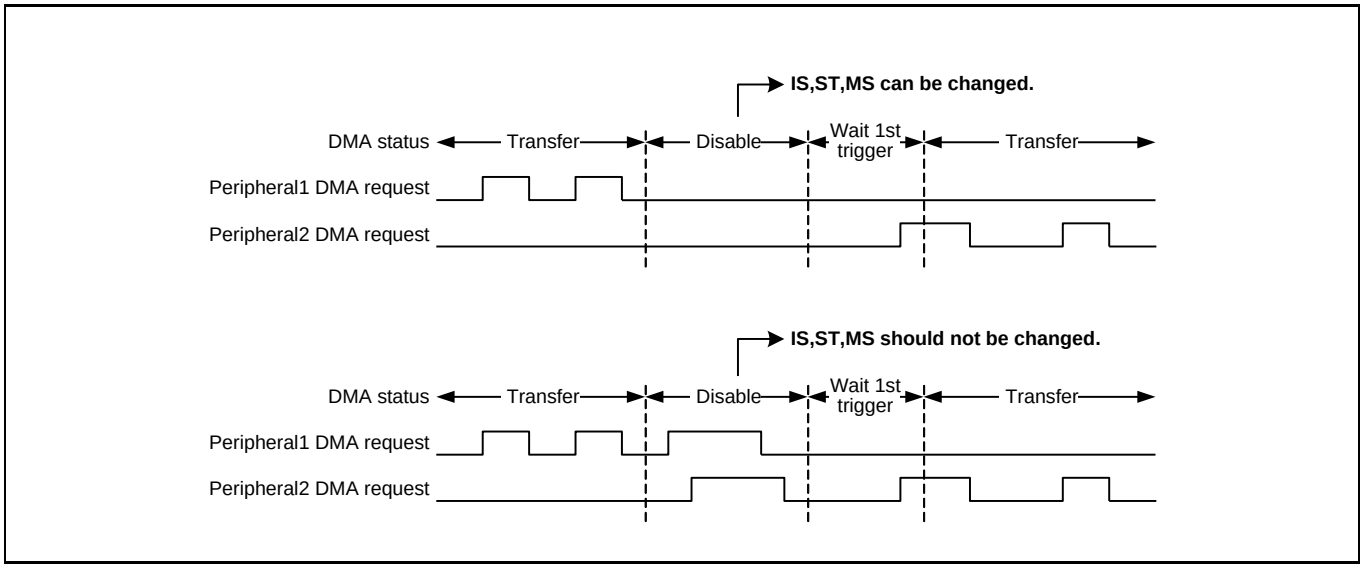
13. Operation in Disable state and Wait-1st-trigger state

See Step 11 in the software transfer procedure.

If the transfer request signal is not asserted to the channel in Disable state, the specifications of the transfer content can be changed freely (rewriting to registers DMACSA, DMACDA, DMACA[29:0], and DMACB).

If the transfer request signal is asserted or maybe asserted to the channel in Disable state, the specifications of IS, ST and MS in the transfer content cannot be changed. If an attempt is made to change these settings, DMAC may perform unexpected behaviors. To change the settings of IS, ST and MS, first clear the transfer request signal to both of the Peripherals (used before and after the change) from CPU, and then always change the settings while the transfer request signal is deasserted. See Figure 4-10.

**Figure 4-10 Changing IS, ST and MS Settings**




The specifications of the transfer content cannot be changed to the channel in Wait-1st-trigger state from CPU

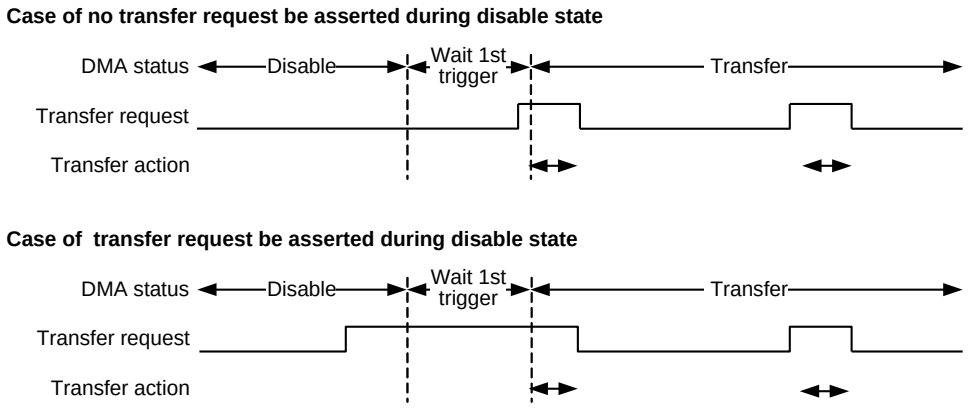
If the transfer request signal is not asserted to the channel in Wait-1st-trigger state, it moves to Disable state when CPU issues an instruction to disable individual- or all-channel operation or an instruction to put individual- or all-channel operation on pause. In this case, it is considered that the enabled transfer has been cancelled. In any case, SS does not change.

If the transfer request signal maypossiblybe asserted to the channel in Wait-1st-trigger state, it should be noted that DMAC has already started or completed the transfer before the attempted cancellation of th e enabled transfer from CPU.

In Disable state, DMAC does not start the transfer or clear the transfer request, even if the transfer request signal is asserted. If it moves to Wait-1st-trigger state by instruction from CPU while the transfer request signal is asserted, the following operation applies (only when the settings of IS, ST and MS are not intended to be changed, as explained earlier).

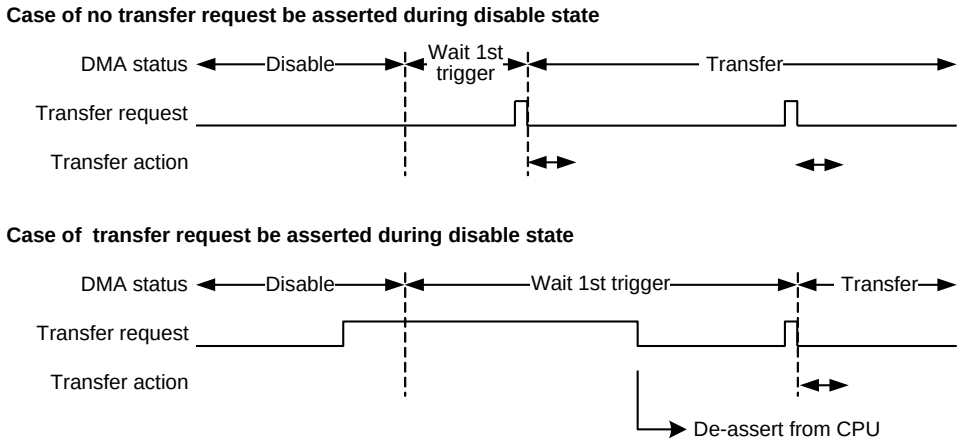
In the case of Demand transfer mode, DMAC immediately moves to Transfer state and starts the transfer, because the transfer request signal remains asserted. The transfer request signal is cleared from DMAC as normal. See Figure 4-11.

**Figure 4-11 Operation of Demand Transfer in Disable State**




In the case of Block transfer mode, the transfer request signal remains asserted. Even when it moves to Wait-1st-trigger state, the rising edge of the transfer request signal is not detected, and the transfer is not resumed. Therefore, the transfer request is ignored during Disable state. Also, the transfer request signal is not cleared from DMAC. To resume the transfer, instruct DMAC to move to Wait-1st-trigger state, and then instruct from CPU the Peripheral to deassert the transfer request signal. After that, it will move to Transfer state and the transfer will be resumed when the next transfer request is generated from the Peripheral. In this case, attention must be paid to the difference between the number of transfer requests output from the Peripheral and the number of transfer requests received by DMAC. See Figure 4-12.

**Figure 4-12 Operation of Block Transfer in Disable State**



■ Additional Matter 1

See Additional Matter 1 in “ 4.2. DMAC Operation and Control Procedure for Software Transfer”.

In the case of hardware transfer, always write "0" to ST.

■ Additional Matter 2

See Additional Matter 2 in “ 4.2. DMAC Operation and Control Procedure for Software Transfer”.

■ Additional Matter 3

See Additional Matter 3 in “4.2. DMAC Operation and Control Procedure for Software Transfer”.

■ Additional Matter 4

See Additional Matter 4 in “4.2.DMAC Operation and Control Procedure for Software Transfer”.

■ Additional Matter 5

If the transfer request signal (interrupt signal) from the Peripheral needs to be deasserted, the following method is available. Normally, the interrupt signal from the Peripheral is the interrupt factor flag masked (logic AND) by the interrupt enable flag. The interrupt signal can be deasserted by resetting either of the flags. When the interrupt enable flag is reset and then set, the rising edge occurs to the interrupt signal. Following this procedure can notify DMAC of the transfer request for Block transfer again. For details, check the manual for each Peripheral.

## 4.4 DMAC Operation and Control Procedure for Hardware (EM=1) Transfer

This section explains DMAC operation and control procedure for hardware (EM=1) transfer.

**Figure 4-13 Transitional Diagram of Hardware (EM=1) Transfer State**

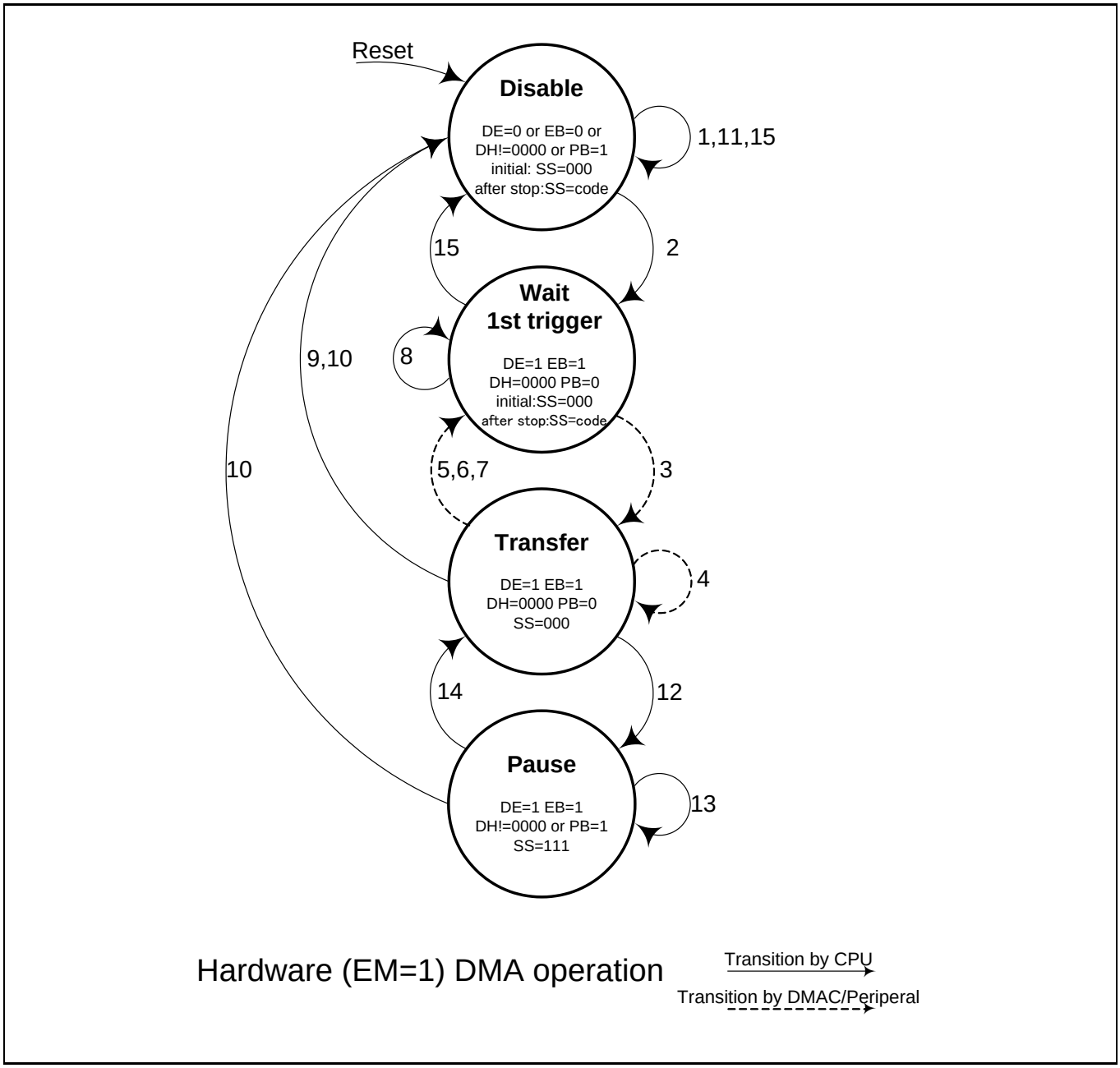



Figure 4-13 shows a transitional diagram of the states of the channel to be controlled for hardware (EM=1) transfer. The numbers next to the transitional lines in Figure 4-13 correspond to the numbers which appear in the following control procedures. The solid transitional lines indicate transitions of state instructed by CPU, while the broken transitional lines indicate transitions of state due to DMAC/Peripheral operation.

EM (Enable bit clear mask) is a bit that masks EB clear upon the completion of transfer of the channel to be controlled. EM=1 enables the same transfer process to be repeated without giving instructions from CPU.

### Description of Each State

- Disable state

See the hardware transfer (EM=0) procedure.

- Wait-1st-trigger state

See the hardware transfer (EM=0) procedure.

- Transfer state

In this state, the channel to be controlled has received the first transfer request from the Peripheral. A channel in this state performs transfer operation as specified. In the case of EM=1, it moves to Wait-1st-trigger state, once all the transfer operation is completed. It also changes its state upon instruction from CPU.

- Pause state

See the hardware transfer (EM=0) procedure.

### Explanation of Control Procedure

1. Disable state / Preparation for transfer

   See Step 1 in the hardware transfer (EM=0) procedure.

   To set EM=1, set all of the reload specifications for the transfer content (RC, RS, RD) in order to prevent data transfer in an unintended address area. Also, CI is not set, because it is meaningless to generate a successful transfer completion interrupt from DMAC. EI is set to generate an unsuccessful transfer completion interrupt from DMAC.

2. Disable state => Wait-1st-trigger state / Enabling transfer

   See Step 2 in the hardware transfer (EM=0) procedure.

3. Wait-1st-trigger state / Start of transfer

   See Step 3 in the hardware transfer (EM=0) procedure.

4. Transfer state

   See Step 4 in the hardware transfer (EM=0) procedure.

5. Transfer state => Wait-1st-trigger state / Successful completion of transfer

   When transfers are successfully completed for the number of times calculated by (BC+1) × (TC+1), the channel in Transfer state does not clear EB but does clear PB and ST and moves to Wait-1st-trigger. It sets SS=101 to provide the notification of the successful completion. As CI is not set, no successful transfer completion interrupt is generated. Since RC, RS and RD are set, the specifications of the transfer content of BC, TC, DMACSA and DMACDA are reloaded.

6. Transfer state => Wait-1st-trigger state / Transfer error end

   See Step 6 in the hardware transfer (EM=0) procedure.

   In the case of EM=1, EB is not cleared even if the transfer ends due to an error. It clears PB and ST, moves to Wait-1st-trigger state and waits for the next transfer request. Therefore, it is recommended not to use DMA transfer with EM=1 in an address area where a transfer error may occur.

7. Transfer state =>Wait-1st-trigger state /End of Peripheral stop request

   See Step 7 in the hardware transfer (EM=0) procedure.

   In the case of EM=1, EB is not cleared even if a stop request is issued from the Peripheral. It clears PB and ST and moves to Wait-1st-trigger state. Since RC, RS and RD are set, the specifications of the transfer content of BC, TC, DMACSA and DMACDA are reloaded. As EI is set, an unsuccessful transfer completion interrupt is generated.

8. Wait-1st-trigger state / Post-transfer process

In the case of EM=1, EB is not cleared upon the completion of the transfer. (DE=1, EB=1, DH=0000, PB=0) is set and it moves to Wait-1st-trigger state. When the next transfer request is generated from the Peripheral, therefore, the next transfer starts without an instruction from CPU.

If it moves to Wait-1st-trigger state due to a stop request from the Peripheral, an unsuccessful completion interrupt occurs and that state can be confirmed. Also, the transfer request signal is masked while the stop request signal is asserted. Even if the next transfer request signal is asserted from the Peripheral, it will not be recognized and the channel to be controlled will remain in Wait-1st-trigger state, waiting for an instruction from CPU.

In the above case, SS is read from CPU to check the state of the transfer completion. The interrupt signal is deasserted by clearing SS from CPU. CPU clears EB and it returns to Disable state (this operation is the operation shown in Step 15 of the hardware transfer (EM=1) procedure). The transfer request signal and the stop request signal from the Peripheral are deasserted, as shown in Step 7 of the hardware transfer (EM=0) procedure

9. Transfer state => Disable state / Completion of transfer by EM=0

The operation can exit from the loop of Wait-1st-trigger state and Transfer state by writing EM=0 from CPU. At the timing when the transfer stops after the instruction, EB, ST and PB are cleared and the Transfer state changes to Disable state (DE=1, EB=0, DH=0000, PB=0) to successfully complete the transfer. In this case, no successful transfer completion interrupt is generated, as CI is not set.

10. Transfer state, Pause state => Disable state / Forced termination of transfer

See Step 8 in the hardware transfer (EM=0) procedure.

The operation can exit from the loop of Wait-1st-trigger state and Transfer state by an operation disable instruction. When an instruction to disable individual-channel operation is issued, the relevant channel moves to Disable state (DE=1, EB=0, DH=0000, PB=0) and stops the operation. When an instruction to enable all-channel operation is issued, it moves to Disable state (DE=0, EB=1, DH=0000, PB=0) and stops the operation. In the case of an instruction to disable all-channel operation, EB is not cleared either; therefore, attention must be paid.

When the operation exits from Transfer state, an unsuccessful transfer completion interrupt occurs because it is unsuccessful completion due to the forced stop. When it exits from Wait-1st-trigger state, the enabled transfer is cancelled (this operation is the operation shown in Step 15 of the hardware transfer (EM=1) procedure).

11. Disable state / Post-transfer processing

See Step 9 in the hardware transfer (EM=0) procedure.

12. Transfer state, Pause state / Transfer pause

See Step 10 in the hardware transfer (EM=0) procedure.

13. Pause state

See Step 11 in the hardware transfer (EM=0) procedure.

14. Pause state / Cancellation of transfer pause

See Step 12 in the hardware transfer (EM=0) procedure.

15. Operation in Disable state and Wait-1st-trigger state

See Step 13 in the hardware transfer (EM=0) procedure.

In the case of EM=1, the Transfer state changes directly to Wait-1st-trigger state. Therefore, the specifications of the transfer content cannot be rewritten during the repeated transfer operation (rewriting the registers DMACSA, DMACDA, DMACB[31:1] and DMACA[28:0]).

- Additional Matter 1

  See Additional Matter 1 in “4.3. DMAC Operation and Control Procedure for Hardware (EM=0) Transfer”.

- Additional Matter 2

  See Additional Matter 2 in “4.3. DMAC Operation and Control Procedure for Hardware (EM=0) Transfer.

  In the case of EM=1, Additional Matter 2 does not apply, because EB is not cleared during the transfer operation”.

- Additional Matter 3

  See Additional Matter 3 in “4.3. DMAC Operation and Control Procedure for Hardware (EM=0) Transfer”.

- Additional Matter 4

  See Additional Matter 4 in “4.3. DMAC Operation and Control Procedure for Hardware (EM=0) Transfer”.

  The following explains what must be noted when setting interrupts from DMAC with EM=1. As the target channel does not change from Wait-1st-trigger state due to an unsuccessful completion interrupt by a stop request from the Peripheral, the interrupt signal is not deasserted until it is cleared from CPU. Similarly, as the target channel moves to Disable state due to an unsuccessful transfer completion interrupt by a stop request from software, the interrupt signal is not deasserted until it is cleared from CPU. Other successful transfer completion interrupts and unsuccessful transfer completion interrupts may be deasserted at a timing that is not intended by CPU, if the relevant channel moves to Transfer state. Therefore, attention must be paid.

- Additional Matter 5

  See Additional Matter 5 in “4.3. DMAC Operation and Control Procedure for Hardware (EM=0) Transfer”.

# 5. Registers of DMAC

This section explains each register function of DMAC.

## 5.1 List of Registers

Table 5-1 shows a list of DMAC control registers.

**Table 5-1 List of DMAC Control Registers**

| Abbreviation | Ch. Controlled | Register name | Reference |
|---|---|---|---|
| DMACR | All | Entire DMAC configuration register | 5.2 |
| DMACA0 | ch.0 | Configuration A register | 5.3 |
| DMACB0 | | Configuration B register | 5.4 |
| DMACSA0 | | Transfer source address register | 5.5 |
| DMACDA0 | | Transfer destination address register | 5.6 |
| DMACA1 | ch.1 | Configuration A register | 5.3 |
| DMACB1 | | Configuration B register | 5.4 |
| DMACSA1 | | Transfer source address register | 5.5 |
| DMACDA1 | | Transfer destination address register | 5.6 |
| DMACA2 | ch.2 | Configuration A register | 5.3 |
| DMACB2 | | Configuration B register | 5.4 |
| DMACSA2 | | Transfer source address register | 5.5 |
| DMACDA2 | | Transfer destination address register | 5.6 |
| DMACA3 | ch.3 | Configuration A register | 5.3 |
| DMACB3 | | Configuration B register | 5.4 |
| DMACSA3 | | Transfer source address register | 5.5 |
| DMACDA3 | | Transfer destination address register | 5.6 |
| DMACA4 | ch.4 | Configuration A register | 5.3 |
| DMACB4 | | Configuration B register | 5.4 |
| DMACSA4 | | Transfer source address register | 5.5 |
| DMACDA4 | | Transfer destination address register | 5.6 |
| DMACA5 | ch.5 | Configuration A register | 5.3 |
| DMACB5 | | Configuration B register | 5.4 |
| DMACSA5 | | Transfer source address register | 5.5 |
| DMACDA5 | | Transfer destination address register | 5.6 |
| DMACA6 | ch.6 | Configuration A register | 5.3 |
| DMACB6 | | Configuration B register | 5.4 |
| DMACSA6 | | Transfer source address register | 5.5 |
| DMACDA6 | | Transfer destination address register | 5.6 |
| DMACA7 | ch.7 | Configuration A register | 5.3 |
| DMACB7 | | Configuration B register | 5.4 |
| DMACSA7 | | Transfer source address register | 5.5 |
| DMACDA7 | | Transfer destination address register | 5.6 |

## 5.2 Entire DMAC Configuration Register (DMACR)

This section explains entire DMAC configuration register (DMACR).

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | DE | DS | Reserved | PR | DH[3:0] | | | | Reserved | | | | | | | |
| Attribute | R/W | R/W | R/W | R/W | R/W | | | | R/W | | | | | | | |
| Initial Value | 0 | 0 | 0 | 0 | 0000 | | | | 00000000 | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial Value | 0x0000 | | | | | | | | | | | | | | | |

**Register functions**

**[bit31] DE : DMA Enable (all-channel operation enable bit)**

This bit controls the enabling and disabling of transfer operations for all of the channels.
When "1" is set to this bit, the operations of all of the channels are enabled and each channel operates according to its settings.
When "0" is set to this bit, the operations of all of the channels are disabled, and no transfer is performed until "1" is set to the bit. Also, a channel in the middle of its transfer operation is forced to stop the transfer.

This bit can be used to force all of the channels that are currently performing a transfer to stop it and reset the configuration register.

| bit | Function |
|---|---|
| 0 | Disables the operations of all of the channels. (Initial value) |
| 1 | Enables the operations of all of the channels. |

**[bit30] DS : DMA Stop**

This bit indicates the transfer state of all of the channels.
If either of the following conditions is established during transfer operation, the bit is set to "1" by DMAC.

- When "0" is written to the DMACR:DE bit and then the transfers of all of the channels are completed.
- When a value other than "0000" is written to the DMACR:DH bit and then the transfers of all of the channels pause.

When DMACR:DE=1 and DMACR:DH=0000 are set and all of the channels become enabled to operate, this bit is set to "0" by DMAC.

Although the attribute of this bit is R/W, writing to it by CPU does not affect DMAC's operation. If, however, the DMACR register needs to be updated without affecting the state of this bit, first read from this bit and then rewrite the same value.

| bit | Function |
|---|---|
| 0 | Clears the disabling of all-channel operation or the setting of all-channel pause. (Initial value) |
| 1 | The transfers of all of the channels have stopped due to the disabling of all-channel operation or the setting of all-channel pause. |

**[bit29] Reserved: Reserved bit**
When writing, always write "0". "0" is always read.

**[bit28] PR : Priority Rotation**
This bit controls the order of transfer priority among channels.
When this bit is set to "0", the priority order is fixed for all of the channels.
When this bit is set to "1", the priority order is determined in a rotation method for all of the channels.

| bit | Function |
|---|---|
| 0 | Fixes the priority order.<br>(ch.0>ch.1>ch.2>ch.3>ch.4>ch.5>ch.6>ch.7) (Initial value) |
| 1 | Applies the rotation method to the priority order. |

For selection of the transfer priority order, see Section "3.5 Channel Priority Control".

**[bit27:24] DH : DMA Halt (All-channel pause bit)**
This bit controls the pause/cancellation of transfer operations for all of the channels.
When this bit is set to a value other than "0000", all of the channels that are currently performing a transfer are put on pause. When it is set to "0000", the transfers are resumed.

Even if a transfer request from an external/peripheral device is asserted, the channels in Pause state ignore the transfer request. In the cases of Block transfer and Burst transfer, the relevant channel does not start a transfer, even if the pause is cleared. In order to complete a transfer when a pause is set during the transfer, an additional transfer request is required after the pause is cancelled.

This bit can be used to put a transfer on pause without resetting the configuration registers of all of the channels.

| bit27:24 | Function |
|---|---|
| 0000 | Cancels the pause of transfers for all of the channels. (Initial value) |
| Other than 0000 | Puts the transfers of all of the channels on pause. |

**[bit23:0] Reserved: Reserved bits**
When writing, always write "0". "0" is always read.

## 5.3 Configuration A Register (DMACA)

This section explains configuration Aregister (DMACA).

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | EB | PB | ST | IS[5:0] | | | | | | Reserved | | | BC[3:0] | | | |
| Attribute | R/W | R/W | R/W | R/W | | | | | | R/W | | | R/W | | | |
| Initial Value | 0 | 0 | 0 | 000000 | | | | | | 000 | | | 0000 | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | TC[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial Value | 0x0000 | | | | | | | | | | | | | | | |

**Register functions**

**[bit31] EB : Enable bit (individual-channel operation enable bit)**

This bit controls the enabling and disabling of the transfer operation of an individual channel.
When this bit is set to "1", the relevant channel is enabled to operate and waits for a trigger to start its transfer operation (the DMACR:DE must be set to "1").

If the EM bit (DMACB[0]) is not set to "1", DMAC clears this bit to "0" upon the completion of the transfer.

When this bit is set to "0", the relevant channel is disabled to operate and does not perform transfer operation until it is set to "1". Also, if it is in the middle of transfer operation, it is forced to stop the transfer. This bit can be used to force the relevant channel that is currently in transfer operation to stop it and reset the configuration register.

| bit | Function |
|---|---|
| 0 | The operation of the relevant channel is disabled. (Initial value) |
| 1 | The operation of the relevant channel is enabled. |

**[bit30] PB : Pause bit (individual-channel pause bit)**

This bit controls the pause/cancellation of the transfer operation of an individual channel.
When this bit is set to "1" and the relevant channel is currently in transfer operation, it puts the transfer on pause. When this bit is set to "0", it resumes the transfer.

This bit is cleared to "0", when the transfer operation of the channel is completed.

Even if a transfer request from an external/peripheral device is asserted, the channels in Pause state ignore the transfer request. In the cases of Block transfer and Burst transfer, the relevant channel does not start a transfer, even if the pause is cleared. In order to complete a transfer when a pause is set during the transfer, an additional transfer request is required after the pause is cancelled.

This bit can be used to put a transfer on pause without resetting the configuration register of the relevant channel.

| bit | Function |
|---|---|
| 0 | Cancels the pause of the transfer of the relevant channel. |
| 1 | Puts the transfer of the relevant channel on pause. |

### [bit29] ST : Software Trigger

This bit is used to generate a software transfer request for an individual channel.

When this bit is set to "1", a trigger is generated by the software transfer request and the relevant channel starts its transfer. After the completion of the transfer, DMAC clears this bit to "0".
When this bit is set to "0" during the transfer, the transfer stops.

| bit | Function |
|---|---|
| 0 | No software transfer request (Initial value) |
| 1 | Software transfer request available |

### [bit28:23] IS[5:0] : Input Select

These bits select the trigger for transfer requests.

When the transfer trigger is set to software request (ST=1), set the IS[5:0] bits to "000000".
When the transfer trigger is set to hardware request, specify which Peripheral's interrupt signal to be used to start transfer. Any Peripheral can be selected for all of the channels.

The hardware transfer request signal to be connected to DMAC varies depending on the product. Check the transfer request signal to be connected in "2.2 I/O Signals of DMAC" before setting the selection.

| bit28:23 | Function |
|---|---|
| 000000 | Software (Initial value) |
| 100000 | IDREQ[0] |
| 100001 | IDREQ[1] |
| 100010 | IDREQ[2] |
| 100011 | IDREQ[3] |
| 100100 | IDREQ[4] |
| 100101 | IDREQ[5] |
| 100110 | IDREQ[6] |
| 100111 | IDREQ[7] |
| 101000 | IDREQ[8] |
| 101001 | IDREQ[9] |
| 101010 | IDREQ[10] |
| 101011 | IDREQ[11] |
| 101100 | IDREQ[12] |
| 101101 | IDREQ[13] |
| 101110 | IDREQ[14] |
| 101111 | IDREQ[15] |
| 110000 | IDREQ[16] |
| 110001 | IDREQ[17] |
| 110010 | IDREQ[18] |
| 110011 | IDREQ[19] |
| 110100 | IDREQ[20] |
| 110101 | IDREQ[21] |
| 110110 | IDREQ[22] |
| 110111 | IDREQ[23] |
| 111000 | IDREQ[24] |
| 111001 | IDREQ[25] |
| 111010 | IDREQ[26] |
| 111011 | IDREQ[27] |
| 111100 | IDREQ[28] |
| 111101 | IDREQ[29] |
| 111110 | IDREQ[30] |

| bit28:23 | Function |
|---|---|
| 111111 | IDREQ[31] |
| Setting other than above | Setting is prohibited. |

### [bit22:20] Reserved: Reserved bits

When writing, always write "0". "0" is always read.

### [bit19:16] BC[3:0] : Block Count

These bits specify the number of blocks for Block/Burst transfer.

When the transfer mode is set to Demand transfer, set BC[3:0] to "0000".
Set the value "BC[3:0]=Number of blocks - 1". The maximum allowed number of blocks is 16.
The value of these bits can be read during a transfer. Normally, as one transfer source access or one transfer destination access is completed successfully, BC[3:0] is decreased by 1.

- In the case of DMACB:RC=1:

  The value set when the transfer started is reloaded upon the completion of the transfer.

- In the case of DMACB:RC=0:

  The value is set to "0" upon successful completion of the transfer, while the value remains the same value as set during the transfer suspension upon unsuccessful completion of the transfer.

| bit19:16 | Function |
|---|---|
| | Number of transfer blocks (Initial value : 0x0) |

### [bit15:0] TC[15:0] : Transfer Count

These bits specify the number of transfers for Block/Burst/Demand transfer.

Set the value "TC = Number of transfers - 1". The maximum allowed number of transfers is 65536.

The value of these bits can be read during a transfer. Normally, as the transfer of one block is completed, TC is decreased by 1.

- In the case of DMACB:RC=1

  The value set when the transfer started is reloaded upon the completion of the transfer.

- In the case of DMACB:RC=0

  The value is set to "0" upon successful completion of the transfer, while the value remains the same value as set during the transfer suspension upon unsuccessful completion of the transfer.

| bit15:0 | Function |
|---|---|
| | Number of transfers (Initial value : 0x0000) |

## 5.4 Configuration B Register (DMACB)

This section explains configuration B register (DMACB).

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | MS[1:0] | | TW[1:0] | | FS | FD | RC | RS | RD | EI | CI | SS[2:0] | | |
| Attribute | R/W | | R/W | | R/W | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W0 | | |
| Initial Value | 00 | | 00 | | 00 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 000 | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | EM |
| Attribute | R/W | | | | | | | | | | | | | | | R/W |
| Initial Value | 000000000000000 | | | | | | | | | | | | | | | 0 |

**Register functions**

**[bit31:30] Reserved: Reserved bits**

When writing, always write "0". "0" is always read.

**[bit29:28] MS[1:0] : Mode Select**

These bits select the transfer mode.

| bit29:28 | Function |
|---|---|
| 00 | Block transfer mode (Initial value) |
| 01 | Burst transfer mode |
| 10 | Demand transfer mode |
| 11 | Reserved |

**[bit27:26] TW[1:0] : Transfer Width**

These bits specify the bit width of transfer data.

| bit27:26 | Function |
|---|---|
| 00 | Byte (8 bits) (Initial value) |
| 01 | Half-word (16 bits) |
| 10 | Word (32 bits) |
| 11 | Reserved |

**[bit25] FS : Fixed Source**

This bit specifies whether to increment or fix the transfer source address.

| bit | Function |
|---|---|
| 0 | Increments the transfer source address according to TW[1:0]. (Initial value) |
| 1 | Fixes the transfer source address. |

### [bit24] FD : Fixed Destination

This bit specifies whether to increment or fix the transfer destination address.

| bit | Function |
|---|---|
| 0 | Increments the transfer destination address according to TW[1:0]. (Initial value) |
| 1 | Fixes the transfer destination address. |

### [bit23] RC : Reload Count (BC/TC reload)

This bit controls the reload function of BC[3:0] and TC[15:0].

When this bit is set to "1", the value set when the transfer started is reloaded to BC[3:0] and TC[15:0] upon completion of the transfer.

| bit | Function |
|---|---|
| 0 | Disables the reload function of BC/TC. (Initial value) |
| 1 | Enables the reload function of BC/TC. |

### [bit22] RS : Reload Source

This bit controls the reload function of the transfer source address.

When this bit is set to "1", the value set when the transfer started is reloaded to DMACSA upon completion of the transfer.

| bit | Function |
|---|---|
| 0 | Disables the reload function of the transfer source address. (Initial value) |
| 1 | Enables the reload function of the transfer source address. |

### [bit21] RD : Reload Destination

This bit controls the reload function of the transfer destination address (DMACDA).

When this bit is set to "1", the value set when the transfer started is reloaded to DMACDA upon completion of the transfer.

| bit | Function |
|---|---|
| 0 | Disables the reload function of the transfer destination address. (Initial value) |
| 1 | Enables the reload function of the transfer destination address. |

### [bit20] EI :Error Interrupt (unsuccessful transfer completion interrupt enable)

This bit enables or disables the notification of an interrupt when a transfer has been unsuccessfully completed.

When this bit is set to "1", an interrupt is issued if SS is in the following status upon completion of the transfer.

- Address overflow SS[2:0]=001
- Stop by transfer stop request from a Peripheral, or the disabling of transfer by the EB/DE bit SS[2:0]=010
- Transfer source access error SS[2:0]=011
- Transfer destination access error SS[2:0]=100

| bit | Function |
|---|---|
| 0 | Disables an interrupt to be issued upon unsuccessful completion of transfer. (Initial value) |
| 1 | Enables an interrupt to be issued upon unsuccessful completion of transfer. |

**[bit19] CI :Completion Interrupt : (successful transfer completion interrupt enable)**

This bit enables or disables the notification of an interrupt when a transfer has been successfully completed.

When this bit is set to "1", an interrupt is generated, if SS is set to successful completion upon completion of the transfer.

| bit | Function |
|---|---|
| 0 | Disables an interrupt to be issued upon successful completion of transfer.<br>(Initial value) |
| 1 | Enables an interrupt to be issued upon successful completion of transfer. |

**[bit18:16] SS[2:0] : Stop Status (stop status notification)**

These bits represent a code that indicates the stop status or completion status of a transfer.

The following table shows the available codes.

If a successful transfer completion interrupt or unsuccessful transfer completion interrupt is issued, the interrupt signal is deasserted by writing "000" to these bits.

| bit18:16 | Description |
|---|---|
| 000 | Initial value |
| 001 | Termination by transfer error (address overflow) |
| 010 | Termination by transfer stop request (stop by transfer stop request for Peripheral or the disabling of transfer by the EB/DE bit) |
| 011 | Termination by transfer error (transfer source access error) |
| 100 | Termination by transfer error (transfer destination access error) |
| 101 | Successful transfer completion |
| 110 | Reserved |
| 111 | Transfer on pause |

If various errors occur simultaneously, the termination code is indicated according to the following priority.

Highest priority

Reset
Clearing by writing "000"
Address overflow
Stop request
Transfer source access error
Transfer destination access error

Lowest priority

**[bit15:1] Reserved: Reserved bits**

When writing, always write "0". "0" is always read.

**[bit0] EM : Enable bit Mask (EB bit clear mask)**

This bit is used to mask the clear of the EB bit (DMACA[31]) from DMAC upon completion of the transfer.

- In the case of EM=0

  DMAC clears the EB bit (DMACA[31]) to "0" upon completion of the transfer.

- In the case of EM=1

  It does not clear the EB bit upon completion of the transfer. This function allows transfers to be repeated without instruction from CPU.

This function can only be used for hardware transfer. To use the function, enable the reload function of RC, RS and RD bits.

| bit | Function |
|---|---|
| 0 | Clear DMACA:EB bit (bit31) upon completion of the transfer. (Initial value) |
| 1 | Dose not clear DMACA:EB bit (bit31) upon completion of the transfer. |

## 5.5 Transfer Source Address Register (DMACSA)

This section explains transfer source address register (DMACSA).

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | DMACSA[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial Value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | DMACSA[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial Value | 0x0000 | | | | | | | | | | | | | | | |

**Register functions**

**[bit31:0] DMACSA : DMAC Source Address**

These bits specify the transfer start address of the transfer source.
It is not possible to specify an address causing an analingend transfer to the setting of W[1:0]. The value of these bits can be read during the transfer.

- In the case of DMACB:FS=1

  The transfer source address is set to a fixed value and no change occurs.

- In the cases of DMACB:FS=0 and DMACB:RS=0

  The value is incremented according to TW[1:0]. Upon successful transfer completion, it is the next address after the transfer completion address. Upon unsuccessful transfer completion, it is the value set during the suspension.

In the cases of DMACB:FS=0 and DMACB:RS=1
It is incremented according to TW[1:0] during the transfer. Upon completion of the transfer, the value set when the transfer started is reloaded.

| | Function |
|---|---|
| **bit31:0** | Specifies the transfer source address from which the transfer starts.<br>(Initial value: 0x00000000) |

## 5.6 Transfer Destination Address Register (DMACDA)

This section explains transfer destination address register (DMACDA).

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | DMACDA[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial Value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | DMACDA[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial Value | 0x0000 | | | | | | | | | | | | | | | |

**Register functions**

**[bit31:0] DMACDA : DMAC Destination Address**

These bits specify the transfer start address of the transfer destination.
It is not possible to specify an address causing an unalingend transfer to the setting of W[1:0].. The value of these bits can be read during the transfer.

In the case of DMACB:FD=1, the transfer destination address is set to a fixed value and no change occurs.

In the cases of DMACB:FD=0 and DMACB:RD=0, the value is incremented according to TW[1:0]. Upon successful transfer completion, it is the next address after the transfer completion address. Upon unsuccessful transfer completion, it is the value set during the suspension.

In the cases of DMACB:FD=0 and DMACB:RD=1, it is incremented according to TW[1:0] during the transfer. Upon completion of the transfer, the value set when the transfer started is reloaded.

| | Function |
|---|---|
| **bit31:0** | Transfer destination address from which DMA transfer starts<br>(Initial value: 0x00000000) |

# 6. Usage Precautions

This section explains the precautions on using DMAC.

### Precautions on register setting

When setting DMAC register, please note the following.

- The DMACR, DMACA, DMACB, DMACSA and DMACDA registers can be accessed by byte, half-word and word.
- The register address in DMAC cannot be set to the DMACSA or DMACDA register.
- Channel setting registers cannot be changed during DMA transfer, except the DE/DH bits of DMACR, the EB/PB bits of DMACA and the EM bit of DMACB.

### Precautions on STOP and TIMER mode transition

When transiting to STOP mode and TIMER mode, make sure to stop the operation of all channels of the DMAC and confirm the stop of the DMAC by DS flag. If the transition is made to STOP mode and TIMER mode while DMAC is operating, an unexpected operation can be executed when returning to RUN mode.

### Note on Transfer Memory Space

Do not execute the transfer to the bit band area.

# CHAPTER10-1: I/O Port

**This chapter explains the I/O port.**



CODE: 9BFGPIO-E06.0

# 1. Overview

This section provides an overview of the I/O port.

The I/O port of this series provides the following features.

- The I/O port of this series shares the following functions.
  - GPIO

    General-purpose I/O ports, which can read an input level and set an output level from the CPU.
  - Fast GPIO

    Fast GPIO, which can read an input level and set an output level from the CPU by 1 cycle. For the details, refer to the "CHAPTER: Fast GPIO"
  - Peripheral input/output

    Digital input/output signal ports of peripheral functions.
  - Special I/O ports
  - Analog input port
    An analog input port of an A/D converter and LCD controller.
  - Analog output port
    An analog output port of a D/A converter and LCD controller.
  - Oscillation port
- The followings settings can be made for each pin.
  - You can set whether the I/O port will be used as a GPIO, a digital pin of peripheral functions, or a special pin.
  - You can set whether the I/O port will be used as an input port or an output port.
  - You can enable or disable pull-up.
  - Peripheral functions are assigned to two or more I/O ports with input/output of the same function. You can set to which I/O port the function can be allocated (relocation function).
  - By setting registers, you can set the I/O port to Hi-Z status while the CPU is in standby mode.

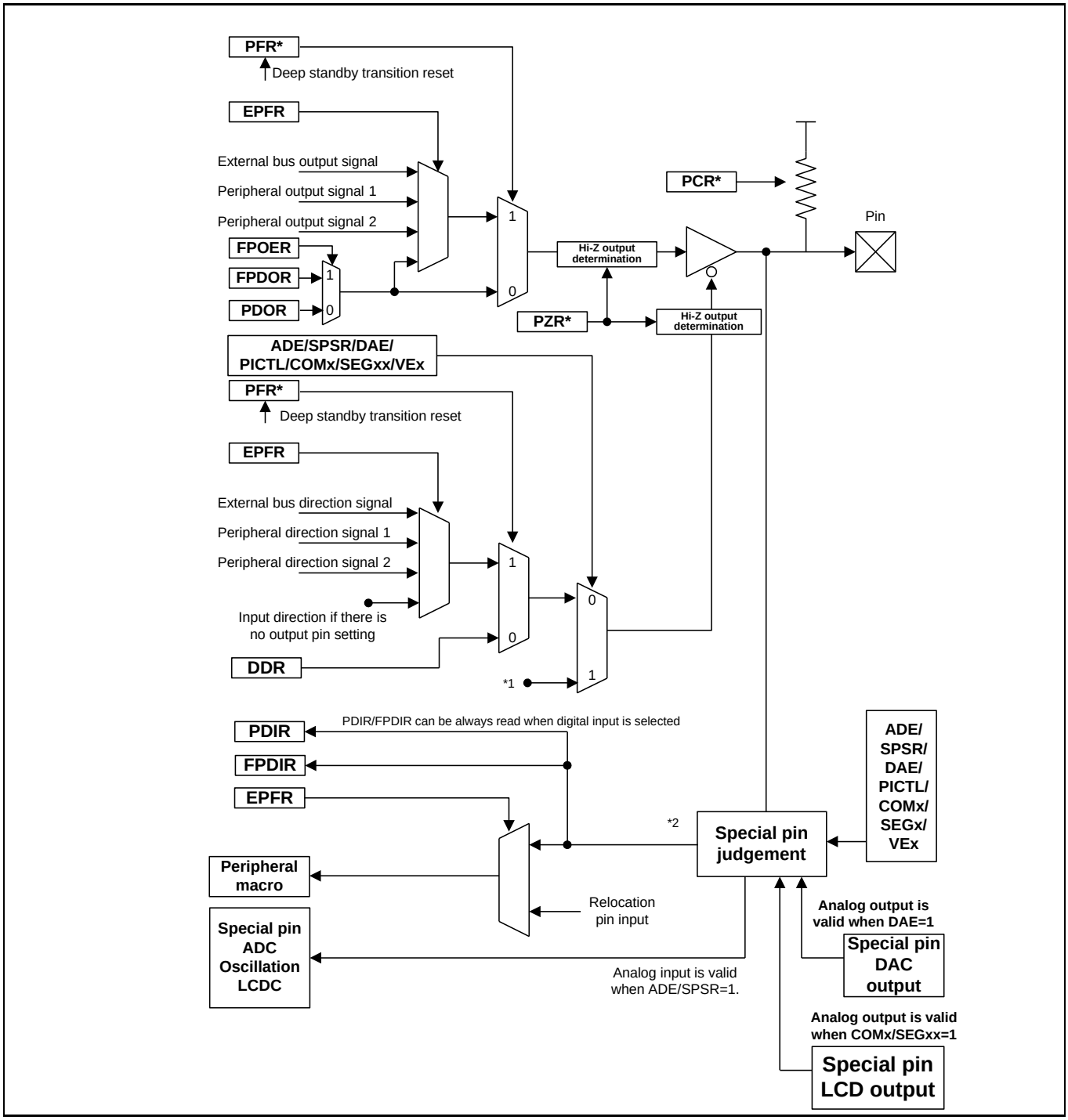
# 2. Configuration, Block Diagram, and Operation

This section explains the configuration, block diagram, and operation of the I/O port.

### Configuration of the I/O Port

By setting registers of the I/O port, select Input/Output direction and select GPIO/peripheral.
Figure 2-1 shows the details of the I/O port.

**Figure 2-1 Block Diagram of the I/O Port**



*1: When one of the followings is set, I/O port is set to input direction.

- ADE/SPSR=1
- DAE=1
- PICTL=0
- PICTL=1 and COMx/SEGxx=1
- VEx=1

*2: When one of the followings is set, the input value is fixed to "0".

Otherwise, the pin is set as the digital input pin.

- ADE/SPSR=1
- DAE=1
- PICTL=0
- PICTL=1 and COMx/SEGxx=1
- VEx=1

***Notes:***

- *For some products, 5V tolerant I/O does not have a pull-up resistor.*
- *If it does not have a pull-up resistor, the PCR register setting is null.*
- *PZR register function is implemented only in some specific pins.*
- *Only pins described as "PZR register control is enabled" in the remarks column of "I/O CIRCUIT TYPE" of the Data Sheet can control this feature.*
- *PFR0 register is not initialized by deep standby transition reset.*
- *For details of DAE bit, "5.1. D/A Control Register (DACR)" in "10-bit D/A CONVERTER" in "Analog Macro Part".*
- *For details of PICTL/COMx/SEGxx/VEx bit, "5.3 LCDC Control Register 3 (LCDCC3)", "5.5 LCDC COM Output Enable Register (LCDC_COMEN)" and "5.6 LCDC SEG Output Enable Register 1/2 (LCDC_SEG1/2)" of "LCD CONTROLLER" in "Analog Macro Part".*
- *FPDIR/FPDOR/FPOER are registers for Fast GPIO. For the details, refer to the "CHAPTER: Fast GPIO".*

Table 2-1 describes register function.

- The PFR, DDR, PDIR, PDOR, and PCR register have 1-bit control register for each I/O port and select a function for the I/O port.
- The ADE register has 1-bit control register for each I/O port which doubles as an analog input pin and selects a function for the I/O port.
- The SPSR register selects a function for the I/O port as an oscillation pin.
- The EPFR register has control register for each I/O pin of peripheral functions and selects to which I/O port an I/O pin of peripheral functions will be relocated.
- PZR register sets open drain control in pseudo mode by the Hi-Zing I/O port when outputting the High level of a particular pin.

**Table 2-1 Register Function Descriptions**

| Register name | Function description |
|---|---|
| ADE | A register to set whether the I/O port will be used as a special pin (an analog input pin) or a digital input/output pin. |
| SPSR | A register to set whether the I/O port will be used as a special pin (oscillation) or a digital input/output pin. |
| PFR | A register to set whether the I/O port will be used as an input/output pin of GPIO function or an input/output pin of peripheral functions. |
| PCR | A register to set whether a pull-up resistor of the I/O port will be connected or disconnected if the I/O port is used as a digital input pin or a digital bidirectional pin. |
| DDR | A register to set whether the I/O port will be used as an input pin or an output pin if the I/O port is used as a GPIO function pin.<br>Note: If a pin is selected as an I/O pin of peripheral functions, a setting value is invalid. |
| PDIR | A register to read the level status of the I/O port.<br>– If the I/O port is used as a digital input pin, it reads input level.<br>– If the I/O port is used as a digital output pin, it reads output level.<br>– If the I/O port is used as an analog input pin, it always reads "0". |
| PDOR | A register to set output level if the I/O port is used as an output pin of GPIO function.<br>– When "0" is set, it outputs Low level.<br>– When "1" is set, it outputs High level.<br>Note: If a pin is selected as GPIO input or input/output of peripheral functions, a setting value is invalid. |
| EPFR | A register to select a function for an input/output of peripheral functions and set relocation function.<br>– Setting a peripheral output pin<br>It sets whether to produce output for the I/O port or not. In addition, it can also set to which I/O port a pin of peripheral functions will be relocated for each pin.<br>– Setting a peripheral input pin<br>It can set to which I/O port a pin of peripheral functions will be relocated for each pin.<br>– Setting a peripheral bidirectional pin<br>It can set to which I/O port a pin of peripheral functions will be relocated for each pin |
| PZR | This register sets open the drain control of the I/O port.<br>– Set the I/O port to Low output when the I/O port is outputting Low level (pull-up disconnection regardless of PCR setting value)<br>– Set open drain control in pseudo mode by setting the I/O port on Hi-Z status when the I/O port outputs High level (pull-up disconnection regardless of PCR setting value)<br>– Set the I/O port on Hi-Z status when the I/O port is used for input (pull-up disconnection regardless of PCR setting value)<br>Note:<br>This function is implemented only in some specific pins.<br>Only pins described as "PZR register control is enabled" in the remarks column of "I/O CIRCUIT TYPE" of the Data Sheet can control the open drain. |

Table 2-2 lists pin functions which availability depends on selected I/O port functions and register setting values.

**Table 2-2 I/O Port Functions and Register Setting Values**

| I/O Port Function | | ADE/ SPSR/ DAE/ COMx/ SEGx/ VEx | PFR | DDR | PZR | PCR | EPFR |
|---|---|---|---|---|---|---|---|
| **Available main function** | **Available sub function** | | | | | | |
| Special pin (Analog input, Analog output, Oscillation) | N/A | 1 | - | - | - | Disconnect | *0 |
| GPIO function input pin | Peripheral function input pin*5 | 0 | 0 | 0 | 0 | Valid | *1 |
| | | | | 0 | 1 | Disconnect | |
| GPIO function output pin | GPIO function input pin (FB)<br>Peripheral function input pin (FB) | | | 1 | 0 | Disconnect | |
| | | | | 1 | 1 | Disconnect | |
| Peripheral function output pin | GPIO function input pin (FB)<br>Peripheral function input pin (FB) | | 1 | - | 0 | Disconnect | *2 |
| | | | | | 1 | Disconnect | |
| Peripheral function bidirectional pin | GPIO function input pin (FB)<br>Peripheral function input pin (FB) | | | | 0 | Valid | *3 |
| | | | | | 1 | Disconnect | |
| Peripheral function input pin | GPIO function input pin | | | | 0 | Valid | *4 |
| | | | | | 1 | Disconnect | |

Legends

-: Indicates that a register setting value does not affect pin functions.

Valid: Indicates that a pull-up resistor is disconnected if PCR register value is 0.
Indicates that a pull-up resistor is connected if PCR register value is 1.

Disconnect: Indicates that a pull-up resistor is disconnected regardless of PCR register value.

(FB): Indicates that an output signal of the I/O port provides feedback and the level of the I/O port can be read from PDIR. The signal can be also used as input for peripheral functions.

*0: If the input pin of peripheral functions is selected for the I/O port, the setting is invalid.
If the output pin of peripheral functions is selected for the I/O port, the setting is invalid.
If the bidirectional pin of peripheral functions is selected for the I/O port, the setting is invalid.

*1: If the input pin of peripheral functions is selected for the I/O port, the setting is valid.
If the output pin of peripheral functions is selected for the I/O port, the setting is invalid.
If the bidirectional pin of peripheral functions is selected for the I/O port, the setting is invalid.

*2: Indicates that the output pin of peripheral functions is selected for the I/O port.

*3: Indicates that the bidirectional pin of peripheral functions is selected for the I/O port.

*4: Indicates that neither the output pin nor the bidirectional pin of peripheral functions is selected for the I/O port.

*5: When NMIX pin is used, set NMIS="1" and PFR="1".

### Initially Selected Functions for the I/O Port

Table 2-3 describes initially selected functions for each I/O port after reset is released.

**Table 2-3 Initially Selected Functions for Each I/O Port after Reset Is Released**

| No | Pin | Initially selected function |
|---|---|---|

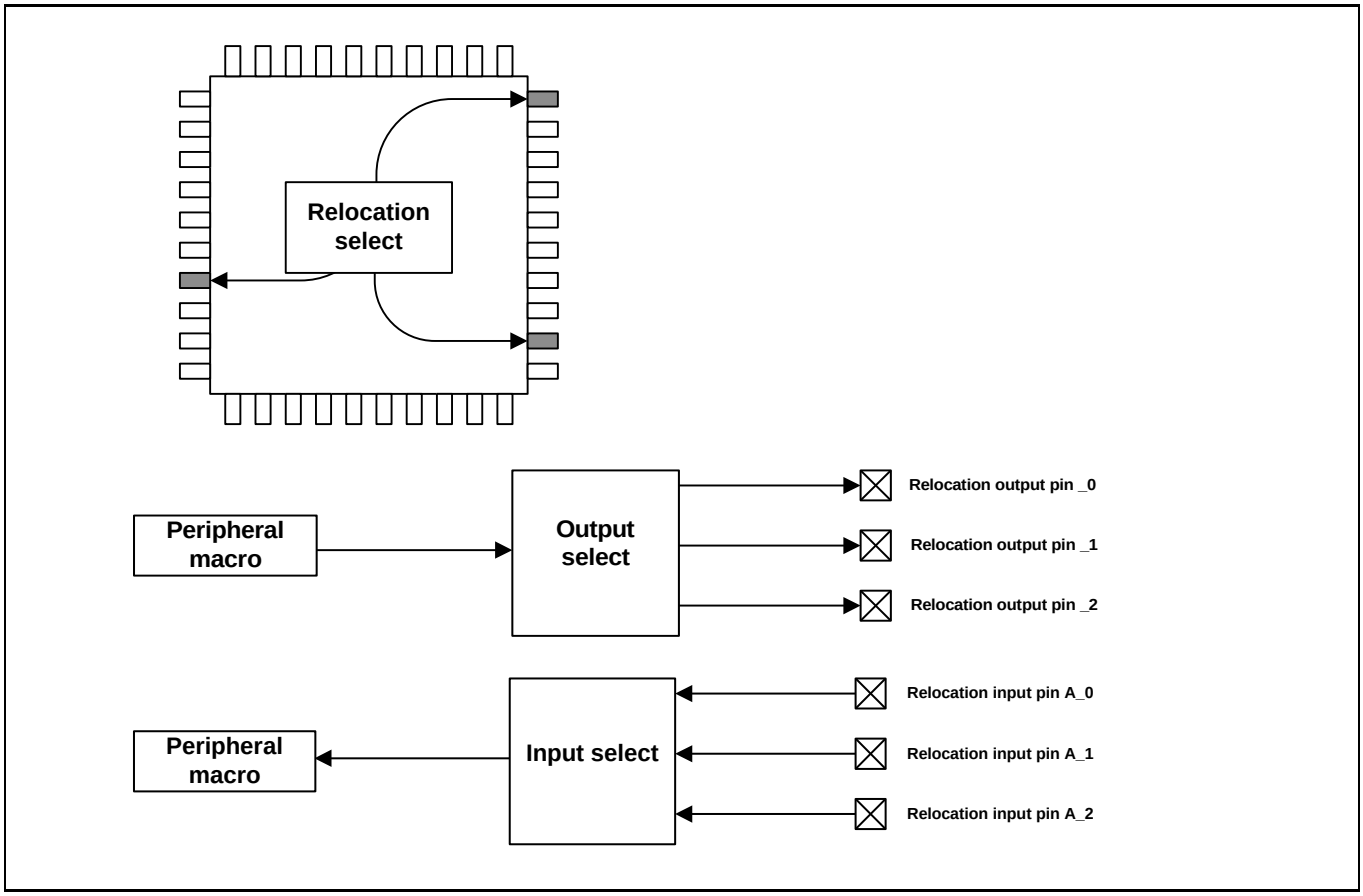
| | | |
|---|---|---|
| 1 | SWCLK, SWDIO | Serial Wire Debug (SWD) pin is selected. Pull-up is enabled. |
| 2 | ANxx | Can be used as an analog input pin. Digital input is cut off and "0" is input. |
| 3 | X0,X1,X0A, X1A | Can be used as an oscillation pin. Digital input is cut off and "0" is input. |
| 4 | All GPIO pins other than the above pins | Digital input. Output is Hi-Z. |

Note: For the status of pins other than GPIO (MD pins, a reset pin), see "Data Sheet" of the product used.
All the output selection values of EPFR during reset are "no output".

### Relocation Function

- Some input/output of peripheral functions have more than one pin (relocation pin).
  One of the pins can be selected by setting EPFR. Figure 2-2 show the schematic view of relocation function.

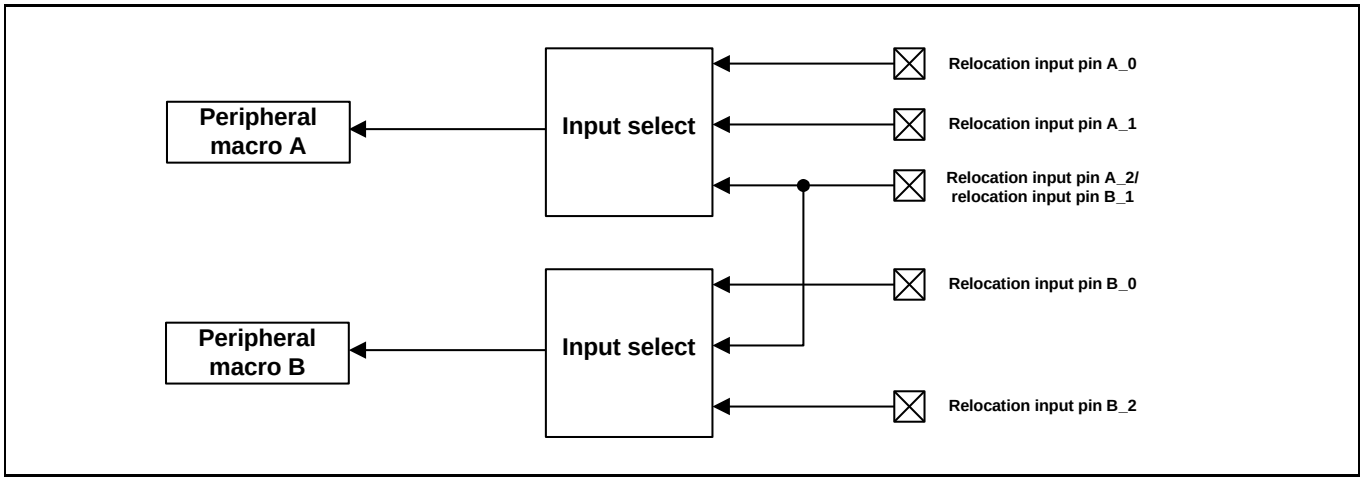
**Figure 2-2 Schematic View of Relocation Function**




Note: Which peripheral function is allocated to which pin depends on products.
See the pin function list of "Data Sheet" of the product used.

- Even if the input of one I/O port is connected to two or more peripheral functions, all peripheral inputs can be used by setting EPFR. For example, in Figure 2-3, by selecting input for both "Relocation input pin A_2" and "Relocation input pin B_1", simultaneous usage is possible. In this way, it is possible to use external interrupt and a multi-function serial input pin shared by one I/O port simultaneously.

**Figure 2-3 Multiple Peripheral Inputs**




- Even if an I/O pin is set as output, it can work as an input pin because input is not masked. For example, timer output can be used as external interrupt input which is shared.

### Fixed Priority of EPFR Outputs

Only one output pin function among two or more outputs is allocated to one I/O port.

By setting the EPFR register, if more than one output is set, fixed priority is applied and output pins are selected. Figure 2-4 shows output pins and fixed priority.

**Figure 2-4 Output Pins and Fixed Priority**

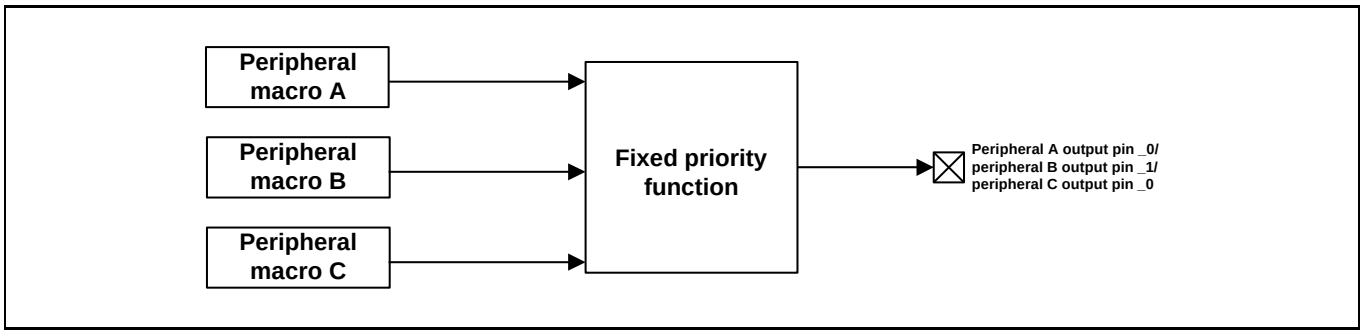



Table 2-4 describes the fixed priority of EPFR.

**Table 2-4 Fixed Priority of EPFR**

| Priority Higher | Peripheral function | Applied pin |
|---|---|---|
| ↓ | Special input | Serial Wire Debug input, NMI input |
| ↓ | Serial Wire Debug | I/O pin |
| ↓ | CAN | Output pin |
| ↓ | Multi-function serial | Output pin, I/O pin*1 |
| ↓ | Base timer output | I/O pin |
| ↓ | Multi-function timer | Output pin |
| ↓ | Internal CR waveform output | Output pin |
| ↓ | RTC Output | Output pin |
| Priority Lower | SUBCLK Output | Output pin |

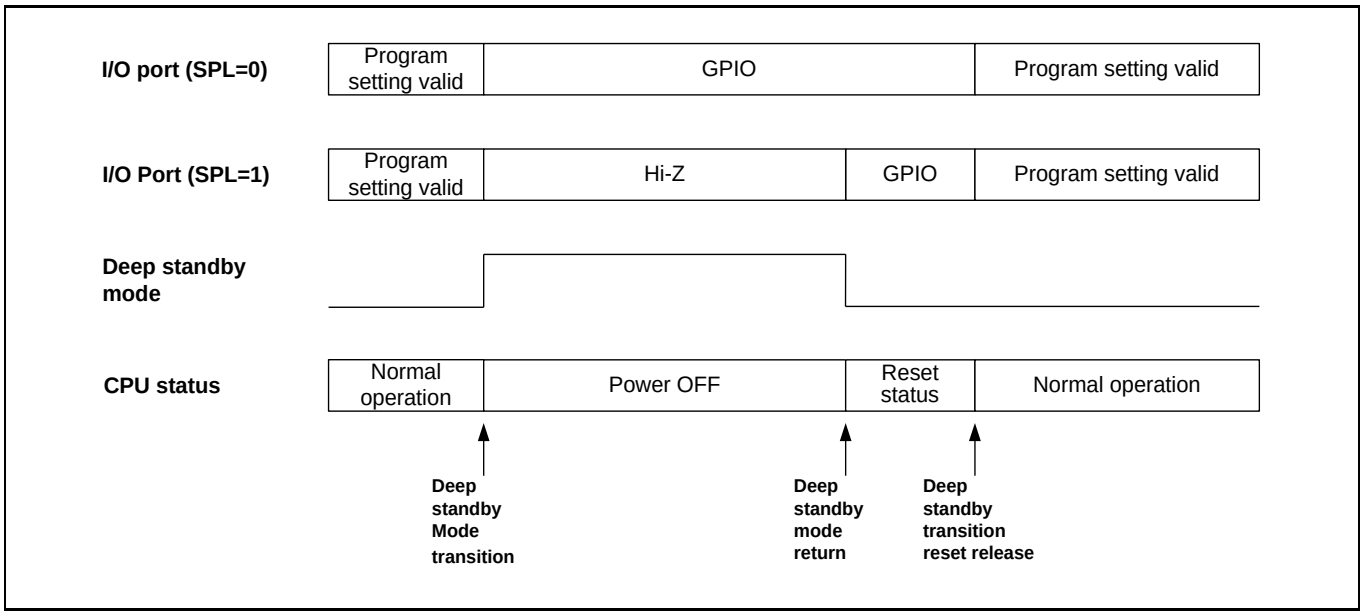
Note: The fixed priority is only applicable when "output" is set for more than one function. In case of "input", there is no fixed priority.
However, "Special input" has a higher priority than any other "output" setting. When "Special input" is set, the "output" setting allocated to the same port is invalid.
*1 : The priority is SOT(serial data output)> SCS(chip select output) in TYPE1 product.

- Due to output setting on the lower part of the priority, the EPFR register always includes "no output" setting.
- If you are going to use a pin as an external input pin of peripheral functions, disable all shared output settings. If every output of a pin is not selected by the EPFR register, the pin works as an external input pin.

### Operation in deep standby mode

GPIO function is selected in deep standby mode. Figure 2-5 shows I/O port operation in deep standby mode.

**Figure 2-5 I/O port operation in deep standby mode**




***Note:***

- *For the state of each pin in deep standby mode, refer to the pin state table in the "Data Sheet" of the product used.*

# 3. Setup Procedure Example

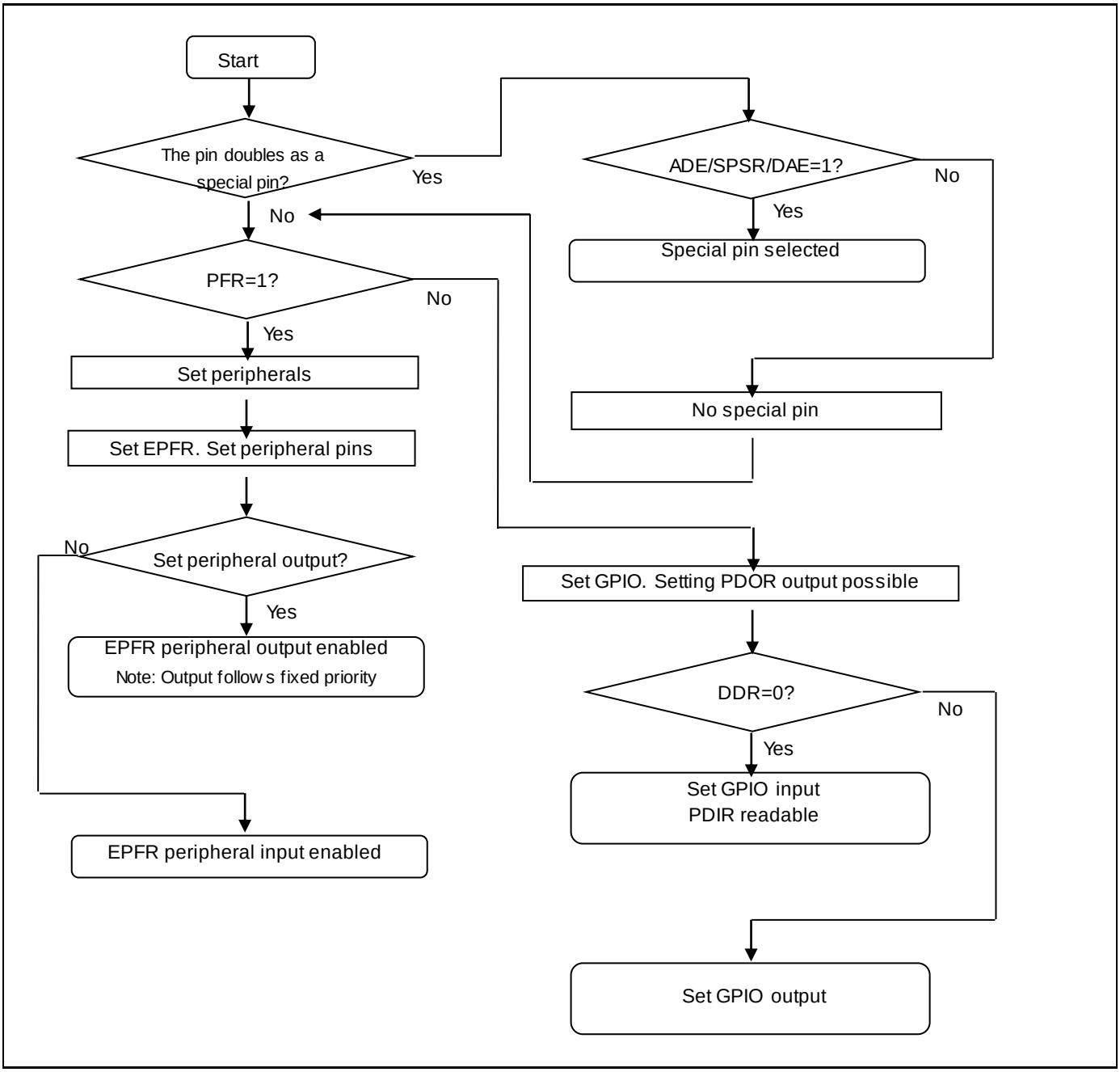
This section explains a procedure example of setting up the I/O port.

**Setup of the I/O Port**

By setting registers of the I/O port, select I/O direction and select GPIO/peripheral.

Figure 3-1 shows a setup procedure example.

**Figure 3-1 Setup Procedure Example of the I/O Port**



# 4. Registers

This section provides the register list of the I/O port.

Table 4-1 provides the register list.

**Table 4-1 Register List of the I/O Port**

| Abbreviation | Register name | Reference |
|---|---|---|
| PFR0 | Port function setting register 0 | 4.1 |
| PFR1 | Port function setting register 1 | |
| PFR2 | Port function setting register 2 | |
| PFR3 | Port function setting register 3 | |
| PFR4 | Port function setting register 4 | |
| PFR5 | Port function setting register 5 | |
| PFR6 | Port function setting register 6 | |
| PFR7 | Port function setting register 7 | |
| PFR8 | Port function setting register 8 | |
| PFR9 | Port function setting register 9 | |
| PFRA | Port function setting register A | |
| PFRB | Port function setting register B | |
| PFRC | Port function setting register C | |
| PFRD | Port function setting register D | |
| PFRE | Port function setting register E | |
| PFRF | Port function setting register F | |
| PCR0 | Pull-up setting register 0 | 4.2 |
| PCR1 | Pull-up setting register 1 | |
| PCR2 | Pull-up setting register 2 | |
| PCR3 | Pull-up setting register 3 | |
| PCR4 | Pull-up setting register 4 | |
| PCR5 | Pull-up setting register 5 | |
| PCR6 | Pull-up setting register 6 | |
| PCR7 | Pull-up setting register 7 | |
| PCR9 | Pull-up setting register 9 | |
| PCRA | Pull-up setting register A | |
| PCRB | Pull-up setting register B | |
| PCRC | Pull-up setting register C | 4.2 |
| PCRD | Pull-up setting register D | |
| PCRE | Pull-up setting register E | |
| PCRF | Pull-up setting register F | |

| Abbreviation | Register name | Reference |
|---|---|---|
| DDR0 | Port input/output direction setting register 0 | 4.3 |
| DDR1 | Port input/output direction setting register 1 | |
| DDR2 | Port input/output direction setting register 2 | |
| DDR3 | Port input/output direction setting register 3 | |
| DDR4 | Port input/output direction setting register 4 | |
| DDR5 | Port input/output direction setting register 5 | |
| DDR6 | Port input/output direction setting register 6 | |
| DDR7 | Port input/output direction setting register 7 | |
| DDR8 | Port input/output direction setting register 8 | |
| DDR9 | Port input/output direction setting register 9 | |
| DDRA | Port input/output direction setting register A | |
| DDRB | Port input/output direction setting register B | |
| DDRC | Port input/output direction setting register C | |
| DDRD | Port input/output direction setting register D | |
| DDRE | Port input/output direction setting register E | |
| DDRF | Port input/output direction setting register F | |
| PDIR0 | Port input data register 0 | 4.4 |
| PDIR1 | Port input data register 1 | |
| PDIR2 | Port input data register 2 | |
| PDIR3 | Port input data register 3 | |
| PDIR4 | Port input data register 4 | |
| PDIR5 | Port input data register 5 | |
| PDIR6 | Port input data register 6 | |
| PDIR7 | Port input data register 7 | |
| PDIR8 | Port input data register 8 | |
| PDIR9 | Port input data register 9 | |
| PDIRA | Port input data register A | |
| PDIRB | Port input data register B | |
| PDIRC | Port input data register C | |
| PDIRD | Port input data register D | 4.4 |
| PDIRE | Port input data register E | |
| PDIRF | Port input data register F | |
| PDOR0 | Port output data register 0 | 4.5 |
| PDOR1 | Port output data register 1 | |
| PDOR2 | Port output data register 2 | |
| PDOR3 | Port output data register 3 | |
| PDOR4 | Port output data register 4 | |
| PDOR5 | Port output data register 5 | |
| PDOR6 | Port output data register 6 | |
| PDOR7 | Port output data register 7 | |
| PDOR8 | Port output data register 8 | |
| PDOR9 | Port output data register 9 | |
| PDORA | Port output data register A | |
| PDORB | Port output data register B | |
| PDORC | Port output data register C | |
| PDORD | Port output data register D | |
| PDORE | Port output data register E | |
| PDORF | Port output data register F | |
| ADE | Analog input setting register | 4.6 |
| EPFR | Extended pin function setting register | 4.7 |

| Abbreviation | Register name | Reference |
|---|---|---|
| EPFR00 | Extended pin function setting register 00 | 4.8 |
| EPFR01 | Extended pin function setting register 01 | 4.9 |
| EPFR02 | Extended pin function setting register 02 | 4.10 |
| EPFR03 | Extended pin function setting register 03 | 4.11 |
| EPFR04 | Extended pin function setting register 04 | 4.12 |
| EPFR05 | Extended pin function setting register 05 | 4.13 |
| EPFR06 | Extended pin function setting register 06 | 4.14 |
| EPFR07 | Extended pin function setting register 07 | 4.15 |
| EPFR08 | Extended pin function setting register 08 | 4.16 |
| EPFR09 | Extended pin function setting register 09 | 4.17 |
| EPFR12 | Extended pin function setting register 12 | 4.18 |
| EPFR13 | Extended pin function setting register 13 | 4.19 |
| EPFR14 | Extended pin function setting register 14 | 4.20 |
| EPFR15 | Extended pin function setting register 15 | 4.21 |
| EPFR16 | Extended pin function setting register 16 | 4.22 |
| EPFR17 | Extended pin function setting register 17 | 4.23 |
| EPFR18 | Extended pin function setting register 18 | 4.24 |
| EPFR21 | Extended pin function setting register 21 | 4.25 |
| EPFR22 | Extended pin function setting register 22 | 4.26 |
| SPSR | Special Port Setting Register | 4.27 |
| PZR0 | Port pseudo open drain setting register 0 | 4.28 |
| PZR1 | Port pseudo open drain setting register 1 | |
| PZR2 | Port pseudo open drain setting register 2 | |
| PZR3 | Port pseudo open drain setting register 3 | |
| PZR4 | Port pseudo open drain setting register 4 | |
| PZR5 | Port pseudo open drain setting register 5 | |
| PZR6 | Port pseudo open drain setting register 6 | |
| PZR7 | Port pseudo open drain setting register 7 | |
| PZR8 | Port pseudo open drain setting register 8 | |
| PZR9 | Port pseudo open drain setting register 9 | |
| PZRA | Port pseudo open drain setting register A | |
| PZRB | Port pseudo open drain setting register B | |
| PZRC | Port pseudo open drain setting register C | |
| PZRD | Port pseudo open drain setting register D | |
| PZRE | Port pseudo open drain setting register E | |
| PZRF | Port pseudo open drain setting register F | |

## 4.1 Port Function Setting Register (PFRx)

The PFRx register selects usage of a pin.

### List of PFR Register Configuration

| bit 31 … 16 | bit 15 … 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | PFR0 | 0x000A | R/W | P0F to P00 |
| Reserved | PFR1 | 0x0000 | R/W | P1F to P10 |
| Reserved | PFR2 | 0x0000 | R/W | P2F to P20 |
| Reserved | PFR3 | 0x0000 | R/W | P3F to P30 |
| Reserved | PFR4 | 0x0000 | R/W | P4F to P40 |
| Reserved | PFR5 | 0x0000 | R/W | P5F to P50 |
| Reserved | PFR6 | 0x0000 | R/W | P6F to P60 |
| Reserved | PFR7 | 0x0000 | R/W | P7F to P70 |
| Reserved | PFR8 | 0x0000 | R/W | P8F to P80 |
| Reserved | PFR9 | 0x0000 | R/W | P9F to P90 |
| Reserved | PFRA | 0x0000 | R/W | PAF to PA0 |
| Reserved | PFRB | 0x0000 | R/W | PBF to PB0 |
| Reserved | PFRC | 0x0000 | R/W | PCF to PC0 |
| Reserved | PFRD | 0x0000 | R/W | PDF to PD0 |
| Reserved | PFRE | 0x0000 | R/W | PEF to PE0 |
| Reserved | PFRF | 0x0000 | R/W | PFF to PF0 |

### Detailed Register Configuration

| bit | 31 … 16 | 15 … 0 |
|---|---|---|
| Field | Reserved | PFRx |

### Register Function

**[bit31:16] Reserved: Reserved bits**

"0x0000" is read out from these bits.
When writing these bits, set them to "0x0000".

**[bit15:0] PFRx: Port Function Setting Register x**

Selects usage of a pin.

| bit15:0 | | Description |
|---|---|---|
| Reading | | Can read out the setting value of the register. |
| Writing | 0 | Uses a pin as a GPIO pin. |
| | 1 | Uses a pin as an input/output pin of peripheral functions. |

***Notes:***

- *The "x" of PFRx is a wildcard. PFRx indicates PFR0, PFR1, PFR2, etc.*
- *The "x" of Px0 and PxF is a wildcard. Px0 indicates P00, P10, P20, etc. PxF indicates P0F, P1F, P2F, etc.*
- *Functions can be set for 16 ports from PxF to Px0.*
- *Each bit in the register sets each pin individually. There is a one-to-one correspondence between bit assignment and the order of pins. For example, the 15th bit of PFR0 sets P0F, the 14th bit of PFR0 sets P0E, and the 0th bit of PFR0 sets P00.*
- *As a Serial Wire Debug pin is selected for P01 and P03, the initial value is "1".*
- *For a pin which is not available in your product, writing a value to the bit is invalid, and the read value is undefined.*
  *bit1 and bit3 of PFR0 register is not initialized by deep standby transition reset.*

## 4.2 Pull-up Setting Register (PCRx)

The PCRx register sets pull-up of a pin.

### List of PCR Register Configuration

| bit 31 16 | 15 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | PCR0 | 0x000A | R/W | P0F to P00 |
| Reserved | PCR1 | 0x0000 | R/W | P1F to P10 |
| Reserved | PCR2 | 0x0000 | R/W | P2F to P20 |
| Reserved | PCR3 | 0x0000 | R/W | P3F to P30 |
| Reserved | PCR4 | 0x0000 | R/W | P4F to P40 |
| Reserved | PCR5 | 0x0000 | R/W | P5F to P50 |
| Reserved | PCR6 | 0x0000 | R/W | P6F to P60 |
| Reserved | PCR7 | 0x0000 | R/W | P7F to P70 |
| Reserved | - | - | - | - |
| Reserved | PCR9 | 0x0000 | R/W | P9F to P90 |
| Reserved | PCRA | 0x0000 | R/W | PAF to PA0 |
| Reserved | PCRB | 0x0000 | R/W | PBF to PB0 |
| Reserved | PCRC | 0x0000 | R/W | PCF to PC0 |
| Reserved | PCRD | 0x0000 | R/W | PDF to PD0 |
| Reserved | PCRE | 0x0000 | R/W | PEF to PE0 |
| Reserved | PCRF | 0x0000 | R/W | PFF to PF0 |

### Detailed Register Configuration

| bit | 31 16 | 15 0 |
|---|---|---|
| Field | Reserved | PCRx |

### Register Function

**[bit31:16] Reserved: Register bits**

"0x0000" is read out from these bits.
When writing these bits, set them to "0x0000".

**[bit15:0] PCRx: Pull-up Setting Register x**

Sets pull-up of a pin

| bit15:0 | | Description |
|---|---|---|
| Reading | | Can read out the setting value of the register. |
| Writing | 0 | Disconnects the pull-up resistor of a pin. |
| | 1 | When a pin is in input status (for both GPIO and peripheral functions), the pull-up resistor is connected.<br>When a pin is in output status, the pull-up resistor is disconnected. |

***Notes:***

- *The "x" of PCRx is a wildcard. PCRx indicates PCR0, PCR1, PCR2, etc.*
- *The "x" of Px0 and PxF is a wildcard. Px0 indicates P00, P10, P20, etc. PxF indicates P0F, P1F, P2F, etc.*
- *One register allows setting 16 pull-ups from PxF to Px0.*
- *Each bit in the register sets each pin individually. There is a one-to-one correspondence between bit assignment and the order of pins. For example, the 15th bit of PCR0 sets P0F, the 14th bit of PCR0 sets P0E, and the 0th bit of PCR0 sets P00.*
- *As a Serial Debug pin is selected for P01 and P03, the initial value is "1".*
- *When using $I^2C$ function, use external pull-up by setting PCRx=0.*
- *PCR8 is not available.*
- *For a pin which is not available in your product, writing a value to the bit is invalid, and the read value is undefined.*
- *PE0, PE1 ports do not have a pull-up resistor. Because of this, writing a value to PE register is invalid. An initial value or a write value is read in this register.*
- *PCRx register is not initialized by deep standby transition reset.*

## 4.3 Port input/output Direction Setting Register (DDRx)

The DDRx register sets input/output direction of a pin.

### List of DDR Register Configuration

| bit 31 16 | bit 15 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | DDR0 | 0x0000 | R/W | P0F to P00 |
| Reserved | DDR1 | 0x0000 | R/W | P1F to P10 |
| Reserved | DDR2 | 0x0000 | R/W | P2F to P20 |
| Reserved | DDR3 | 0x0000 | R/W | P3F to P30 |
| Reserved | DDR4 | 0x0000 | R/W | P4F to P40 |
| Reserved | DDR5 | 0x0000 | R/W | P5F to P50 |
| Reserved | DDR6 | 0x0000 | R/W | P6F to P60 |
| Reserved | DDR7 | 0x0000 | R/W | P7F to P70 |
| Reserved | DDR8 | 0x0000 | R/W | P8F to P80 |
| Reserved | DDR9 | 0x0000 | R/W | P9F to P90 |
| Reserved | DDRA | 0x0000 | R/W | PAF to PA0 |
| Reserved | DDRB | 0x0000 | R/W | PBF to PB0 |
| Reserved | DDRC | 0x0000 | R/W | PCF to PC0 |
| Reserved | DDRD | 0x0000 | R/W | PDF to PD0 |
| Reserved | DDRE | 0x0000 | R/W | PEF to PE0 |
| Reserved | DDRF | 0x0000 | R/W | PFF to PF0 |

### Detailed Register Configuration

| bit | 31 16 | 15 0 |
|---|---|---|
| Field | Reserved | DDRx |

### Register Function

**[bit31:16] Reserved: Reserved bits**

"0x0000" is read out from these bits.
When writing these bits, set them to "0x0000".

**[bit15:0] DDRx: Port input/output Direction Setting Register x**

Sets input/output direction of a pin.

| bit15:0 | | Description |
|---|---|---|
| Reading | | Can read out the setting value of the register. |
| Writing | 0 | Uses GPIO in input direction.<br>If a pin is selected as an input/output pin of peripheral functions, this setting value is invalid. |
| | 1 | Uses GPIO in output direction.<br>If a pin is selected as an input/output pin of peripheral functions, this setting value is invalid. |

***Notes:***

- *The "x" of DDRx is a wildcard. DDRx indicates DDR0, DDR1, DDR2, etc.*
- *The "x" of Px0 and PxF is a wildcard. Px0 indicates P00, P10, P20, etc. PxF indicates P0F, P1F, P2F, etc.*
- *One register allows setting the input/output direction of 16 ports from PxF to Px0.*
- *Each bit in the register sets each pin individually. There is a one-to-one correspondence between bit assignment and the order of pins. For example, the 15th bit of DDR0 sets P0F, the 14th bit of DDR0 sets P0E, and the 0th bit of DDR0 sets P00.*
- *If the output RTO of a multifunction timer is selected, in an emergency stop due to DTTIX signal, a DDR controls pin status. For more information, see the chapter "Multifunction Timer" in "Timer Part".*
- *For a pin which is not available in your product, writing a value to the bit is invalid, and the read value is undefined.*
- *DDRx register is not initialized by deep standby transition reset.*

## 4.4 Port Input Data Register (PDIRx)

The PDIRx register indicates input data of a pin.

### List of PDIR Register Configuration

| bit 31 16 | bit 15 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | PDIR0 | 0xXXXX | R | P0F to P00 |
| Reserved | PDIR1 | 0xXXXX | R | P1F to P10 |
| Reserved | PDIR2 | 0xXXXX | R | P2F to P20 |
| Reserved | PDIR3 | 0xXXXX | R | P3F to P30 |
| Reserved | PDIR4 | 0xXXXX | R | P4F to P40 |
| Reserved | PDIR5 | 0xXXXX | R | P5F to P50 |
| Reserved | PDIR6 | 0xXXXX | R | P6F to P60 |
| Reserved | PDIR7 | 0xXXXX | R | P7F to P70 |
| Reserved | PDIR8 | 0xXXXX | R | P8F to P80 |
| Reserved | PDIR9 | 0xXXXX | R | P9F to P90 |
| Reserved | PDIRA | 0xXXXX | R | PAF to PA0 |
| Reserved | PDIRB | 0xXXXX | R | PBF to PB0 |
| Reserved | PDIRC | 0xXXXX | R | PCF to PC0 |
| Reserved | PDIRD | 0xXXXX | R | PDF to PD0 |
| Reserved | PDIRE | 0xXXXX | R | PEF to PE0 |
| Reserved | PDIRF | 0xXXXX | R | PFF to PF0 |

### Detailed Register Configuration

| bit | 31 16 | 15 0 |
|---|---|---|
| Field | Reserved | PDIRx |

### Register Function

**[bit31:16] Reserved: Reserved bits**

"0x0000" is read out from these bits.
When writing these bits, set them to "0x0000".

**[bit15:0] PDIRx: Port Input Data Register x**

Reads out input data of a pin.

| bit15:0 | | Description |
|---|---|---|
| Reading | 0 | Regardless of pin function settings (PFR/EPFR/DDR/PDOR), it indicates that a pin is in the status of "L" level input or "L" level output. When a special pin is selected by ADE/SPSR, as input is cut off, 0 is always read out. |
| | 1 | Regardless of pin function settings (PFR/EPFR/DDR/PDOR), it indicates that a pin is in the status of "H" level input or "H" level output. |
| Writing | | Writing does not affect anything. |

***Notes:***

- *The "x" of PDIRx is a wildcard. PDIRx indicates PDIR0, PDIR1, PDIR2, etc.*
- *The "x" of Px0 and PxF is a wildcard. Px0 indicates P00, P10, P20, etc. PxF indicates P0F, P1F, P2F, etc.*
- *One register allows reading input data of 16 ports from PxF to Px0.*
- *Each bit in the register indicates the status of each pin individually. There is a one-to-one correspondence between bit assignment and the order of pins. For example, the 15th bit of PDIR0 indicates P0F, the 14th bit of PDIR0 indicates P0E, and the 0th bit of PDIR0 indicates P00.*
- *"0" is always read for a bit value of the pin which is not available in your product.*
- *PDIRx register is not initialized by deep standby transition reset.*

## 4.5 Port Output Data Register x (PDORx)

The PDORx register sets output data to a pin.

### List of PDOR Register Configuration

| bit 31 16 | bit 15 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | PDOR0 | 0x0000 | R/W | P0F to P00 |
| Reserved | PDOR1 | 0x0000 | R/W | P1F to P10 |
| Reserved | PDOR2 | 0x0000 | R/W | P2F to P20 |
| Reserved | PDOR3 | 0x0000 | R/W | P3F to P30 |
| Reserved | PDOR4 | 0x0000 | R/W | P4F to P40 |
| Reserved | PDOR5 | 0x0000 | R/W | P5F to P50 |
| Reserved | PDOR6 | 0x0000 | R/W | P6F to P60 |
| Reserved | PDOR7 | 0x0000 | R/W | P7F to P70 |
| Reserved | PDOR8 | 0x0000 | R/W | P8F to P80 |
| Reserved | PDOR9 | 0x0000 | R/W | P9F to P90 |
| Reserved | PDORA | 0x0000 | R/W | PAF to PA0 |
| Reserved | PDORB | 0x0000 | R/W | PBF to PB0 |
| Reserved | PDORC | 0x0000 | R/W | PCF to PC0 |
| Reserved | PDORD | 0x0000 | R/W | PDF to PD0 |
| Reserved | PDORE | 0x0000 | R/W | PEF to PE0 |
| Reserved | PDORF | 0x0000 | R/W | PFF to PF0 |

### Detailed Register Configuration

| bit | 31 16 | 15 0 |
|---|---|---|
| Field | Reserved | PDORx |

### Register Function

**[bit31:16] Reserved: Reserved bits**

"0x0000" is read out from these bits.
When writing these bits, set them to "0x0000".

**[bit15:0] PDORx: Port Output Data Register x**

Sets output data of a pin.

| bit15:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Outputs "L" level to GPIO.<br>If a pin is selected as I/O input or peripheral functions input/output, a setting value is invalid. |
| | 1 | Outputs "H" level to GPIO.<br>If a pin is selected as I/O input or peripheral functions input/output, a setting value is invalid. |

***Notes:***

- *The "x" of PDORx is a wildcard. PDORx indicates PDOR0, PDOR1, PDOR2, etc.*
- *The "x" of Px0 and PxF is a wildcard. Px0 indicates P00, P10, P20, etc. PxF indicates P0F, P1F, P2F, etc.*
- *One register allows setting output data of 16 ports from PxF to Px0.*
- *Each bit in the register sets each pin individually. There is a one-to-one correspondence between bit assignment and the order of pins. For example, the 15th bit of PDOR0 sets P0F, the 14th bit of PDOR0 sets P0E, and the 0th bit of PDOR0 sets P00.*
- *For a pin which is not available in your product, writing a value to the bit is invalid, and the read value is undefined.*
- *PDORx register is not initialized by deep standby transition reset.*

## 4.6 Analog Input Setting Register (ADE)

The ADE register sets an external pin as an analog signal input pin of ADC.

### Register Configuration

| bit | 31 | 0 |
|---|---|---|
| Field | ADE | |
| Attribute | R/W | |
| Initial value | 0xFFFFFFFF | |

### Register Function

**[bit31:0] ADE: Analog Input Setting Register**

Sets as an analog signal input pin.

| bit31:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Uses an external pin not as analog input but digital input/output. |
| | 1 | Uses an external pin as analog input.<br>(An I/O cell will be in a state of input direction, input cut-off, and pull-up disconnection.) |

***Notes:***

- *This register sets analog input pins from AN31 to AN00.*
- *Each bit in the register sets each pin individually. There is a one-to-one correspondence between bit assignment and the order of pins. For example, the 31st bit of ADE sets AN31, the 14th bit of ADE sets AN14, and the 0th bit of ADE sets AN00. The port position of ANxx differs by each product. For correspondence, refer to the "Data Sheet" of the product used.*
- *This register is not initialized by deep standby transition reset.*

## 4.7 Extended Pin Function Setting Register (EPFRx)

The EPFRx register assigns functions to a pin if there is more than one function.

**List of EPFRx Register Configuration**

| bit 31 0 | Initial value | Attribute | Corresponding function |
|---|---|---|---|
| EPFR00 | 0x00010000 | R/W | System function |
| EPFR01 | 0x00000000 | R/W | Multi-function timer |
| EPFR02 | 0x00000000 | R/W | |
| EPFR03 | 0x00000000 | R/W | |
| EPFR04 | 0x00000000 | R/W | Base timer |
| EPFR05 | 0x00000000 | R/W | |
| EPFR06 | 0x00000000 | R/W | External interrupt |
| EPFR07 | 0x00000000 | R/W | Multi-function serial |
| EPFR08 | 0x00000000 | R/W | |
| EPFR09 | 0x00000000 | R/W | CAN/ADC trigger/QPRC |
| EPFR12 | 0x00000000 | R/W | Base timer |
| EPFR13 | 0x00000000 | R/W | |
| EPFR14 | 0x00000000 | R/W | QPRC/ HDMI-CEC, Remote Control Reception |
| EPFR15 | 0x00000000 | R/W | External interrupt |
| EPFR16 | 0x00000000 | R/W | Multi-function serial |
| EPFR17 | 0x00000000 | R/W | |
| EPFR18 | 0x00000000 | R/W | HDMI-CEC/Remote reception |
| EPFR21 | 0x00000000 | R/W | QPRC |
| EPFR22 | 0x00000000 | R/W | Multi-function serial |

EPFRx register is different depending on product TYPE.

For the correspondence between EPFRx register existence and product TYPE, SEE Table 4-2.

***Notes:***

- *EPFRx register is not initialized by deep standby transition reset.*

**Table 4-2 EPFRx Register Product TYPE Correspondence Table (TYPE1)**

| | TYPE1 |
|---|---|
| EPFR00 | ❍ |
| EPFR01 | - |
| EPFR02 | - |
| EPFR03 | - |
| EPFR04 | ❍ |
| EPFR05 | - |
| EPFR06 | ❍ |
| EPFR07 | ❍ |
| EPFR08 | - |
| EPFR09 | ❍ |
| EPFR12 | - |
| EPFR13 | - |
| EPFR14 | - |
| EPFR15 | - |
| EPFR16 | - |
| EPFR17 | - |
| EPFR18 | - |
| EPFR21 | ❍ |
| EPFR22 | ❍ |

## 4.8 Extended Pin Function Setting Register 00 (EPFR00)

The EPFR00 register assigns functions to a pin if there is more than one function.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | SWDEN |
| Attribute | - | | | | | | | R/W |
| Initial value | - | | | | | | | 1 |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | SUBOUTE | | RTCCOE | | Reserved | CROUTE | | NMIS |
| Attribute | R/W | | R/W | | - | R/W | | R/W |
| Initial value | 00 | | 00 | | - | 00 | | 0 |

### Register Function

**[bit31:17] Reserved: Reserved bits**

"0" is read out from these bits.
When writing these bits, set them to "0".

**[bit16] SWDEN: Serial Wire Debug Function Select bit 0**

Selects the function for SWCLK and SWDIO pins.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Does not use two pins of SWCLK and SWDIO.<br>(A shared pin is available.) |
| | 1 | Uses two pins of SWCLK and SWDIO. [Initial value] |

**[bit15:8] Reserved: Reserved bits**

"0" is read out from these bits.
When writing these bits, set them to "0".

### [bit7:6] SUBOUTE: Sub clock divide output function select bit

Selects sub clock divide output.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Sub clock divide output is not executed. [initial value] |
| | 01 | SUBOUT_0 is used as the sub clock divide output pin. |
| | 10 | SUBOUT_1 is used as the sub clock divide output pin. |
| | 11 | SUBOUT_2 is used as the sub clock divide output pin. |

### [bit5:4] RTCCOE: RTC clock output select bit

Selects a RTC clock output.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | RTC clock output is not executed. [initial value] |
| | 01 | RTCCOE_0 is used as the RTC clock output pin. |
| | 10 | RTCCOE_1 is used as the RTC clock output pin. |
| | 11 | RTCCOE_2 is used as the RTC clock output pin. |

### [bit3] Reserved: Reserved bit

"0" is read out from this bit.
When writing this bit, set it to "0".

### [bit2:1] CROUTE: Internal high-speed CR Oscillation Output Function Select bit

Selects internal high-speed CR oscillation output.

| bit2:1 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce internal high-speed CR oscillation output. [Initial value] |
| | 01 | Uses CROUT_0 at the internal high-speed CR oscillation output pin. |
| | 10 | Uses CROUT_1 at the internal high-speed CR oscillation output pin. |
| | 11 | Uses CROUT_2 at the internal high-speed CR oscillation output pin. |

### [bit0] NMIS: NMIX Function Select bit

Selects a function for the NMIX pin.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Does not use the NMIX pin. [Initial value] |
| | 1 | Uses the NMIX pin. |

***Note:***

- *This register is not initialized by deep standby transition reset.*
- *When NMIX pin is used, set NMIS="1" and PFR="1".*

## 4.9 Extended Pin Function Setting Register 01 (EPFR01)

The EPFR01 register assigns functions to a pin of the multifunction timer Unit0.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | IC03S | | | IC02S | | | IC01S | |
| Attribute | R/W | | | R/W | | | R/W | |
| Initial value | 000 | | | 000 | | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | IC01S | IC00S | | | FRCK0S | | DTTI0S | |
| Attribute | R/W | R/W | | | R/W | | R/W | |
| Initial value | 0 | 000 | | | 00 | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | DTTI0C | RTO05E | | RTO04E | |
| Attribute | - | | | R/W | R/W | | R/W | |
| Initial value | - | | | 0 | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | RTO03E | | RTO02E | | RTO01E | | RTO00E | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

### Register Function

#### [bit31:29] IC03S: IC03 Input Select bits

Selects input for IC03.

| bit31:29 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses IC03_0 at the input pin of the input capture IC03. [Initial value] |
| | 001 | Same as Writing 000. |
| | 010 | Uses IC03_1 at the input pin of the input capture IC03. |
| | 011 | Uses IC03_2 at the input pin of the input capture IC03. |
| | 100 | Uses internal macro MFS ch.3 LSYN for input of the input capture IC03. |
| | 101 | Uses internal macro MFS ch.7 LSYN for input of the input capture IC03. |
| | 110 | Setting is prohibited. |
| | 111 | Uses the internal macro pin CRTRIM for input of the input capture IC03. |

### [bit28:26] IC02S: IC02 Input Select bits

Selects input for IC02.

| bit28:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses IC02_0 at the input pin of the input capture IC02. [Initial value] |
| | 001 | Same as Writing 000. |
| | 010 | Uses IC02_1 at the input pin of the input capture IC02. |
| | 011 | Uses IC02_2 at the input pin of the input capture IC02. |
| | 100 | Uses internal macro MFS ch.2 LSYN for input of the input capture IC02. |
| | 101 | Uses internal macro MFS ch.6 LSYN for input of the input capture IC02. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit25:23] IC01S: IC01 Input Select bits

Selects input for IC01.

| bit25:23 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses IC01_0 at the input pin of the input capture IC01. [Initial value] |
| | 001 | Same as Writing 000. |
| | 010 | Uses IC01_1 at the input pin of the input capture IC01. |
| | 011 | Uses IC01_2 at the input pin of the input capture IC01. |
| | 100 | Uses internal macro MFS ch.1 LSYN for input of the input capture IC01. |
| | 101 | Uses internal macro MFS ch.5 LSYN for input of the input capture IC01. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit22:20] IC00S: IC00 Input Select bits

Selects input for IC00.

| bit22:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses IC00_0 at the input pin of the input capture IC00. [Initial value] |
| | 001 | Same as Writing 000. |
| | 010 | Uses IC00_1 at the input pin of the input capture IC00. |
| | 011 | Uses IC00_2 at the input pin of the input capture IC00. |
| | 100 | Uses internal macro MFS ch.0 LSYN for input of the input capture IC00. |
| | 101 | Uses internal macro MFS ch.4 LSYN for input of the input capture IC00. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit19:18] FRCK0S: FRCK0 Input Select bits

Selects input for FRCK0.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses FRCK0_0 at the input pin of the free-run timer FRCK0. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses FRCK0_1 at the input pin of the free-run timer FRCK0. |
| | 11 | Uses FRCK0_2 at the input pin of the free-run timer FRCK0. |

### [bit17:16] DTTI0S: DTTIX0 Input Select bits

Selects input for DTTIX0.

| bit17:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses DTTIX0_0 at the input pin of the waveform generator DTTIX0. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses DTTIX0_1 at the input pin of the waveform generator DTTIX0. |
| | 11 | Uses DTTIX0_2 at the input pin of the waveform generator DTTIX0. |

### [bit15:13] Reserved: Reserved bits

"0b000" is read out from these bits.

When writing these bits, set them to "0b000".

### [bit12] DTTI0C: DTTIX0 Function Select bit

Selects a function for DTTIX0.

| bit12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Does not switch GPIO by DTTIF0 for output of pins RTO00 to RTO05. [Initial value] |
| | 1 | Switches GPIO by DTTIF0 for output of pins RTO00 to RTO05. |

### [bit11:10] RTO05E: RTO05 Output Select bits

Selects output for RTO05.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO05. [Initial value] |
| | 01 | Uses RTO05_0 at the output pin of the waveform generator RTO05. |
| | 10 | Uses RTO05_1 at the output pin of the waveform generator RTO05. |
| | 11 | Setting is prohibited. |

**[bit9:8] RTO04E: RTO04 Output Select bits**

Selects output for RTO04.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO04. [Initial value] |
| | 01 | Uses RTO04_0 at the output pin of the waveform generator RTO04. |
| | 10 | Uses RTO04_1 at the output pin of the waveform generator RTO04. |
| | 11 | Setting is prohibited. |

**[bit7:6] RTO03E: RTO03 Output Select bits**

Selects output for RTO03.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO03. [Initial value] |
| | 01 | Uses RTO03_0 at the output pin of the waveform generator RTO03. |
| | 10 | Uses RTO03_1 at the output pin of the waveform generator RTO03. |
| | 11 | Setting is prohibited. |

**[bit5:4] RTO02E: RTO02 Output Select bits**

Selects output for RTO02.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO02. [Initial value] |
| | 01 | Uses RTO02_0 at the output pin of the waveform generator RTO02. |
| | 10 | Uses RTO02_1 at the output pin of the waveform generator RTO02. |
| | 11 | Setting is prohibited. |

**[bit3:2] RTO01E: RTO01 Output Select bits**

Selects output for RTO01.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO01. [Initial value] |
| | 01 | Uses RTO01_0 at the output pin of the waveform generator RTO01. |
| | 10 | Uses RTO01_1 at the output pin of the waveform generator RTO01. |
| | 11 | Setting is prohibited. |

**[bit1:0] RTO00E: RTO00 Output Select bits**

Selects output for RTO00.

| bit1:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO00. [Initial value] |
| | 01 | Uses RTO00_0 at the output pin of the waveform generator RTO00. |
| | 10 | Uses RTO00_1 at the output pin of the waveform generator RTO00. |
| | 11 | Setting is prohibited. |

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.10 Extended Pin Function Setting Register 02 (EPFR02)

The EPFR02 register assigns functions to a pin of the multifunction timer Unit1.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | IC13S | | | IC12S | | | IC11S | |
| Attribute | R/W | | | R/W | | | R/W | |
| Initial value | 000 | | | 000 | | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | IC11S | IC10S | | | FRCK1S | | DTTI1S | |
| Attribute | R/W | R/W | | | R/W | | R/W | |
| Initial value | 0 | 000 | | | 00 | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | IGTRG0 | DTTI1C | RTO15E | | RTO14E | |
| Attribute | - | | R/W | R/W | R/W | | R/W | |
| Initial value | - | | 0 | 0 | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | RTO13E | | RTO12E | | RTO11E | | RTO10E | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

### Register Function

#### [bit31:29] IC13S: IC13 Input Select bits

Selects input for IC13.

| bit31:29 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses IC13_0 at the input pin of the input capture IC13. [Initial value] |
| | 001 | Same as Writing 000. |
| | 010 | Uses IC13_1 at the input pin of the input capture IC13. |
| | 011 | Setting is prohibited. |
| | 100 | Uses internal macro MFS ch.3 LSYN for input of the input capture IC13. |
| | 101 | Uses internal macro MFS ch.7 LSYN for input of the input capture IC13. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit28:26] IC12S: IC12 Input Select bits

Selects input for IC12.

| bit28:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses IC12_0 at the input pin of the input capture IC12. [Initial value] |
| | 001 | Same as Writing 000. |
| | 010 | Uses IC12_1 at the input pin of the input capture IC12. |
| | 011 | Setting is prohibited. |
| | 100 | Uses internal macro MFS ch.2 LSYN for input of the input capture IC12. |
| | 101 | Uses internal macro MFS ch.6 LSYN for input of the input capture IC12. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit25:23] IC11S: IC11 Input Select bits

Selects input for IC11.

| bit25:23 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses IC11_0 at the input pin of the input capture IC11. [Initial value] |
| | 001 | Same as Writing 000. |
| | 010 | Uses IC11_1 at the input pin of the input capture IC11. |
| | 011 | Setting is prohibited. |
| | 100 | Uses internal macro MFS ch.1 LSYN for input of the input capture IC11. |
| | 101 | Uses internal macro MFS ch.5 LSYN for input of the input capture IC11. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit22:20] IC10S: IC10 Input Select bits

Selects input for IC10.

| bit22:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses IC10_0 at the input pin of the input capture IC10. [Initial value] |
| | 001 | Same as Writing 000. |
| | 010 | Uses IC10_1 at the input pin of the input capture IC10. |
| | 011 | Setting is prohibited. |
| | 100 | Uses internal macro MFS ch.0 LSYN for input of the input capture IC10. |
| | 101 | Uses internal macro MFS ch.4 LSYN for input of the input capture IC10. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit19:18] FRCK1S: FRCK1 Input Select bits

Selects input for FRCK1.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses FRCK1_0 at the input pin of the free-run timer FRCK1. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses FRCK1_1 at the input pin of the free-run timer FRCK1. |
| | 11 | Setting is prohibited. |

### [bit17:16] DTTI1S: DTTIX1 Input Select bits

Select input for DTTIX1.

| bit17:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses DTTIX1_0 at the input pin of the waveform generator DTTIX1. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses DTTIX1_1 at the input pin of the waveform generator DTTIX1. |
| | 11 | Setting is prohibited. |

### [bit15:14] Reserved: Reserved bits

"0b00" is read out from these bits.

When writing these bits, set them to "0b00".

### [bit13] IGTRG0: IGTRG0 Input Select bit

Selects input for IGTRG0.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Uses IGTRG0_0 at the input pin of the PPG IGTRG. [Initial value] |
| | 1 | Uses IGTRG0_1 at the input pin of the PPG IGTRG. |

### [bit12] DTTI1C: DTTIX1 Function Select bit

Selects a function for DTTIX1.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Does not switch GPIO by DTTIF1 for output of pins RTO10 to RTO15. [Initial value] |
| | 1 | Switches GPIO by DTTIF1 for output of pins RTO10 to RTO15. |

### [bit11:10] RTO15E: RTO15 Output Select bits

Selects output for RTO15.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO15. [Initial value] |
| | 01 | Uses RTO15_0 at the output pin of the waveform generator RTO15. |
| | 10 | Uses RTO15_1 at the output pin of the waveform generator RTO15. |
| | 11 | Setting is prohibited. |

### [bit9:8] RTO14E: RTO14 Output Select bits

Selects output for RTO14.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO14. [Initial value] |
| | 01 | Uses RTO14_0 at the output pin of the waveform generator RTO14. |
| | 10 | Uses RTO14_1 at the output pin of the waveform generator RTO14. |
| | 11 | Setting is prohibited. |

### [bit7:6] RTO13E: RTO13 Output Select bits

Selects output for RTO13.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO13. [Initial value] |
| | 01 | Uses RTO13_0 at the output pin of the waveform generator RTO13. |
| | 10 | Uses RTO13_1 at the output pin of the waveform generator RTO13. |
| | 11 | Setting is prohibited. |

### [bit5:4] RTO12E: RTO12 Output Select bits

Selects output for RTO12.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO12. [Initial value] |
| | 01 | Uses RTO12_0 at the output pin of the waveform generator RTO12. |
| | 10 | Uses RTO12_1 at the output pin of the waveform generator RTO12. |
| | 11 | Setting is prohibited. |

### [bit3:2] RTO11E: RTO11 Output Select bits

Selects output for RTO11.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO11. [Initial value] |
| | 01 | Uses RTO11_0 at the output pin of the waveform generator RTO11. |
| | 10 | Uses RTO11_1 at the output pin of the waveform generator RTO11. |
| | 11 | Setting is prohibited. |

### [bit1:0] RTO10E: RTO10 Output Select bits

Selects output for RTO10.

| bit1:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for the waveform generator RTO10. [Initial value] |
| | 01 | Uses RTO10_0 at the output pin of the waveform generator RTO10. |
| | 10 | Uses RTO10_1 at the output pin of the waveform generator RTO10. |
| | 11 | Setting is prohibited. |

***Notes:***

- *This register is not initialized by deep standby transition reset.*

## 4.11 Extension Function Pin Setting Register 03 (EPFR03)

EPFR03 register sets the function assignment to the multi-function timer Unit2 pin.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | IC23S | | | IC22S | | | IC21S | |
| Attribute | R/W | | | R/W | | | R/W | |
| Initial value | 000 | | | 000 | | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | IC21S | IC20S | | | FRCK2S | | DTTI2S | |
| Attribute | R/W | R/W | | | R/W | | R/W | |
| Initial value | 0 | 000 | | | 00 | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | DTTI2C | RTO25E | | RTO24E | |
| Attribute | - | | | R/W | R/W | | R/W | |
| Initial value | - | | | 0 | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | RTO23E | | RTO22E | | RTO21E | | RTO20E | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

### Register Function

#### [bit31:29] IC23S: IC23 input select bits

Selects IC23 input.

| bit31:29 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Use IC23_0 as the input pin of input capture IC23. [initial value] |
| | 001 | Same as when writing "000" |
| | 010 | Use IC23_1 as the input pin of input capture IC23. |
| | 011 | Setting is prohibited. |
| | 100 | Use internal macro MFS ch.3 LSYN as input of input capture IC23. |
| | 101 | Use internal macro MFS ch.7 LSYN as input of input capture IC23. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit28:26] IC22S: IC22 input select bits

Selects IC22 input.

| bit28:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Use IC22_0 as the input pin of input capture IC22. [initial value] |
| | 001 | Same as when writing "000" |
| | 010 | Use IC22_1 as the input pin of input capture IC22. |
| | 011 | Setting is prohibited. |
| | 100 | Use internal macro MFS ch.2 LSYN as input of input capture IC22. |
| | 101 | Use internal macro MFS ch.6 LSYN as input of input capture IC22. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit25:23] IC21S: IC21 input select bits

Selects IC21 input.

| bit25:23 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Use IC21_0 as the input pin of input capture IC21. [initial value] |
| | 001 | Same as when writing "000" |
| | 010 | Use IC21_1 as the input pin of input capture IC21. |
| | 011 | Setting is prohibited. |
| | 100 | Use internal macro MFS ch.1 LSYN as input of input capture IC21. |
| | 101 | Use internal macro MFS ch.5 LSYN as input of input capture IC21. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit22:20] IC20S: IC20 input select bits

Selects IC20 input.

| bit22:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Use IC20_0 as the input pin of input capture IC20. [initial value] |
| | 001 | Same as when writing "000" |
| | 010 | Use IC20_1 as the input pin of input capture IC20. |
| | 011 | Setting is prohibited. |
| | 100 | Use internal macro MFS ch.0 LSYN as input of input capture IC20. |
| | 101 | Use internal macro MFS ch.4 LSYN as input of input capture IC20. |
| | 110 | Setting is prohibited. |
| | 111 | Setting is prohibited. |

### [bit19:18] FRCK2S: FRCK2 Input Select bits

Selects input for FRCK2.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Use FRCK2_0 as the input pin of free-run timer FRCK2. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Use FRCK2_1 a as the input pin of free-run timer FRCK2. |
| | 11 | Setting is prohibited. |

### [bit17:16] DTTI2S: DTTIX2 Input Select bits

Selects input for DTTIX2.

| bit17:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Use DTTIX2_0 as the input pin of waveform generator DTTIX2. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Use DTTIX2_1 as the input pin of waveform generator DTTIX2. |
| | 11 | Setting is prohibited. |

### [bit15:13] Reserved: Reserved bits

"0b000" is read from these bits.
When writing, set them to "0b000".

### [bit12] DTTI2C: DTTIX2 Function Select bit

Selects the function of DTTIX2.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Does not switch GPIO by DTTIF2 for outputs of the pins from RTO20 to RTO25. [Initial value] |
| | 1 | Switches GPIO by DTTIF2 for outputs of the pins from RTO20 to RTO25. |

### [bit11:10] RTO25E: RTO25 Output Select bits

Selects the output of RTO25.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not output waveform generator RTO25. [Initial value] |
| | 01 | Use RTO25_0 as the output pin of waveform generator RTO25. |
| | 10 | Use RTO25_1 as the output pin of waveform generator RTO25. |
| | 11 | Setting is prohibited. |

### [bit9:8] RTO24E: RTO24 Output Select bits

Selects output for RTO24.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not output waveform generator RTO24. [Initial value] |
| | 01 | Use RTO24_0 as the output pin of waveform generator RTO24. |
| | 10 | Use RTO24_1 as the output pin of waveform generator RTO24. |
| | 11 | Setting is prohibited. |

### [bit7:6] RTO23E: RTO23 Output Select bits

Selects output for RTO23.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not output waveform generator RTO23. [Initial value] |
| | 01 | Use RTO23_0 as the output pin of waveform generator RTO23. |
| | 10 | Use RTO23_1 as the output pin of waveform generator RTO23. |
| | 11 | Setting is prohibited. |

### [bit5:4] RTO22E: RTO22 Output Select bits

Selects output for RTO22.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not output waveform generator RTO22. [Initial value] |
| | 01 | Use RTO22_0 as the output pin of waveform generator RTO22. |
| | 10 | Use RTO22_1 as the output pin of waveform generator RTO22. |
| | 11 | Setting is prohibited. |

### [bit3:2] RTO21E: RTO21 Output Select bits

Selects output for RTO21.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not output waveform generator RTO21. [Initial value] |
| | 01 | Use RTO21_0 as the output pin of waveform generator RTO21. |
| | 10 | Use RTO21_1 as the output pin of waveform generator RTO21. |
| | 11 | Setting is prohibited. |

### [bit1:0] RTO20E: RTO20 Output Select bits

Selects output for RTO20.

| bit1:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not output waveform generator RTO20. [Initial value] |
| | 01 | Use RTO20_0 as the output pin of waveform generator RTO20. |
| | 10 | Use RTO20_1 as the output pin of waveform generator RTO20. |
| | 11 | Setting is prohibited. |

***Notes:***

- *This register is not initialized by deep standby transition reset.*

## 4.12 Extended Pin Function Setting Register 04 (EPFR04)

The EPFR04 register assigns functions to pins of ch.0, ch.1, ch.2, and ch.3 of the base timer.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB3S | | TIOA3E | | TIOA3S | |
| Attribute | - | | R/W | | R/W | | R/W | |
| Initial value | - | | 00 | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB2S | | TIOA2E | | Reserved | |
| Attribute | - | | R/W | | R/W | | - | |
| Initial value | - | | 00 | | 00 | | - | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB1S | | TIOA1E | | TIOA1S | |
| Attribute | - | | R/W | | R/W | | R/W | |
| Initial value | - | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | TIOB0S | | | TIOA0E | | Reserved | |
| Attribute | - | R/W | | | R/W | | - | |
| Initial value | - | 000 | | | 00 | | - | |

### Register Function

#### [bit31:30] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

#### [bit29:28] TIOB3S: TIOB3 Input Select bits

Selects input for TIOB3.

| bit29:28 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB3_0 at the input pin of BT ch.3 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB3_1 at the input pin of BT ch.3 TIOB. |
| | 11 | Uses TIOB3_2 at the input pin of BT ch.3 TIOB. |

#### [bit27:26] TIOA3E: TIOA3 Output Select bits

Selects output for TIOA3.

| bit27:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for BT ch.3 TIOA. [Initial value] |
| | 01 | Uses TIOA3_0 at the output pin of BT ch.3 TIOA. |
| | 10 | Uses TIOA3_1 at the output pin of BT ch.3 TIOA. |
| | 11 | Uses TIOA3_2 at the output pin of BT ch.3 TIOA. |

### [bit25:24] TIOA3S: TIOA3 Input Select bits

Selects input for TIOA3.

| bit25:24 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOA3_0 at the input pin of BT ch.3 TIOA. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOA3_1 at the input pin of BT ch.3 TIOA. |
| | 11 | Uses TIOA3_2 at the input pin of BT ch.3 TIOA. |

### [bit23:22] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

### [bit21:20] TIOB2S: TIOB2 Input Select bits

Selects input for TIOB2.

| bit21:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB2_0 at the input pin of BT ch.2 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB2_1 at the input pin of BT ch.2 TIOB. |
| | 11 | Uses TIOB2_2 at the input pin of BT ch.2 TIOB. |

### [bit19:18] TIOA2E: TIOA2 Output Select bits

Selects output for TIOA2.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for BT ch.2 TIOA. [Initial value] |
| | 01 | Uses TIOA2_0 at the output pin of BT ch.2 TIOA. |
| | 10 | Uses TIOA2_1 at the output pin of BT ch.2 TIOA. |
| | 11 | Uses TIOA2_2 at the output pin of BT ch.2 TIOA. |

### [bit17:14] Reserved: Reserved bits

"0b0000" is read out from these bits.
When writing these bits, set them to "0b0000".

### [bit13:12] TIOB1S: TIOB1 Input Select bits

Selects input for TIOB1.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB1_0 at the input pin of BT ch.1 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB1_1 at the input pin of BT ch.1 TIOB. |
| | 11 | Uses TIOB1_2 at the input pin of BT ch.1 TIOB. |

### [bit11:10] TIOA1E: TIOA1 Output Select bits

Selects output for TIOA1.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for BT ch.1 TIOA. [Initial value] |
| | 01 | Uses TIOA1_0 at the output pin of BT ch.1 TIOA. |
| | 10 | Uses TIOA1_1 at the output pin of BT ch.1 TIOA. |
| | 11 | Uses TIOA1_2 at the output pin of BT ch.1 TIOA. |

### [bit9:8] TIOA1S: TIOA1 Input Select bits

Selects input for TIOA1.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOA1_0 at the input pin of BT ch.1 TIOA. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOA1_1 at the input pin of BT ch.1 TIOA. |
| | 11 | Uses TIOA1_2 at the input pin of BT ch.1 TIOA. |

### [bit7] Reserved: Reserved bit

"0b0" is read out from this bit.

When writing this bit, set it to "0b0".

### [bit6:4] TIOB0S: TIOB0 Input Select bits

Selects input for TIOB0.

| bit6:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses TIOB0_0 at the input pin of BT ch.0 TIOB. [Initial value] |
| | 001 | Same as Writing 000. |
| | 010 | Uses TIOB0_1 at the input pin of BT ch.0 TIOB. |
| | 011 | Uses TIOB0_2 at the input pin of BT ch.0 TIOB. |
| | 100 | Setting is prohibited. |
| | 101 | Setting is prohibited. |
| | 110 | Uses SUBOUT at the input pin of BT ch.0 TIOB. |
| | 111 | Uses at the pin for measuring trimming of the high-speed CR frequency division clock. |

### [bit3:2] TIOA0E: TIOA0 Output Select bits

Selects output for TIOA0.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Produces output for BT ch.0 TIOA. [Initial value] |
| | 01 | Uses TIOA0_0 at the output pin of BT ch.0 TIOA. |
| | 10 | Uses TIOA0_1 at the output pin of BT ch.0 TIOA. |
| | 11 | Uses TIOA0_2 at the output pin of BT ch.0 TIOA. |

**[bit1:0] Reserved: Reserved bits**

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

***Notes:***

- *TIOA*
  *Even channels are for output only.*
  *Odd channels are for both input and output.*
- *TIOB*
  *Input only.*
- *TIOA1, TIOA3, TIOA5, TIOA7(odd number of "A") are not bidirectional pins so that choose either input pin or output pin for them.*
  *When output is chosen for odd TIOA channel, input setting will be ignored.*

  *Example1 : Use TIOA1 as an output pin:*
  *When TIOA1 is output to TIOA1_0, select EPFR04:TIOA1E = 01.*
  *When TIOA1 is output to TIOA1_1, select EPFR04:TIOA1E = 10.*
  *When TIOA1 is output to TIOA1_2, select EPFR04:TIOA1E = 11.*
  *Settings for EPFR04:TIOA1S will be ignored.*
  *Select ADE=0, PFR=1 for selected pins (DDR will be ignored).*
  *All the output of other peripheral function pins which are also used by selected pins must be OFF.*

  *Example2 : When TIOA1 is used as an input pin:*
  *Select EPFR04:TIOA1E = 00.*
  *When TIOA1 is input from TIOA1_0, select EPFR04:TIOA1S = 00 or 01.*
  *When TIOA1 is input from TIOA1_1, select EPFR04:TIOA1S = 10.*
  *When TIOA1 is input from TIOA1_2, select EPFR04:TIOA1S = 11.*
  *Select ADE=0, PFR=1 for selected pins (DDR will be ignored).*
  *All the output of other peripheral function pins which are also used by selected pins must be OFF.*

  ** When a pin is set to input, the output of the pin which is also used by selected pins (GPIO, other peripheral function pins) can be input to the base timer as Feedback with a setting not described above.*
  *This register is not initialized by deep standby transition reset.*

## 4.13 Extended Pin Function Setting Register 05 (EPFR05)

The EPFR05 register assigns functions to pins of ch.4, ch.5, ch.6, and ch.7 of the base timer.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB7S | | TIOA7E | | TIOA7S | |
| Attribute | - | | R/W | | R/W | | R/W | |
| Initial value | - | | 00 | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB6S | | TIOA6E | | Reserved | |
| Attribute | - | | R/W | | R/W | | - | |
| Initial value | - | | 00 | | 00 | | - | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB5S | | TIOA5E | | TIOA5S | |
| Attribute | - | | R/W | | R/W | | R/W | |
| Initial value | - | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB4S | | TIOA4E | | Reserved | |
| Attribute | - | | R/W | | R/W | | - | |
| Initial value | - | | 00 | | 00 | | - | |

### Register Function

#### [bit31:30] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

#### [bit29:28] TIOB7S: TIOB7 Input Select bits

Selects input for TIOB7.

| bit29:28 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB7_0 at the input pin of BT ch.7 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB7_1 at the input pin of BT ch.7 TIOB. |
| | 11 | Uses TIOB7_2 at the input pin of BT ch.7 TIOB. |

#### [bit27:26] TIOA7E: TIOA7 Output Select bits

Selects output for TIOA7.

| bit27:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.7 TIOA. [Initial value] |
| | 01 | Uses TIOA7_0 at the output pin of BT ch.7 TIOA. |
| | 10 | Uses TIOA7_1 at the output pin of BT ch.7 TIOA. |
| | 11 | Uses TIOA7_2 at the output pin of BT ch.7 TIOA. |

### [bit25:24] TIOA7S: TIOA7 Input Select bits

Selects input for TIOA7.

| bit25:24 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOA7_0 at the input pin of BT ch.7 TIOA. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOA7_1 at the input pin of BT ch.7 TIOA. |
| | 11 | Uses TIOA7_2 at the input pin of BT ch.7 TIOA. |

### [bit23:22] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

■ [bit21:20] TIOB6S: TIOB6 Input Select bits

Selects input for TIOB6.

| bit21:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB6_0 at the input pin of BT ch.6 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB6_1 at the input pin of BT ch.6 TIOB. |
| | 11 | Uses TIOB6_2 at the input pin of BT ch.6 TIOB. |

### [bit19:18] TIOA6E: TIOA6 Output Select bits

Selects output for TIOA6.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.6 TIOA. [Initial value] |
| | 01 | Uses TIOA6_0 at the output pin of BT ch.6 TIOA. |
| | 10 | Uses TIOA6_1 at the output pin of BT ch.6 TIOA. |
| | 11 | Uses TIOA6_2 at the output pin of BT ch.6 TIOA. |

### [bit17:14] Reserved: Reserved bits

"0b0000" is read out from these bits.
When writing these bits, set them to "0b0000".

### [bit13:12] TIOB5S: TIOB5 Input Select bits

Selects input for TIOB5.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB5_0 at the input pin of BT ch.5 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB5_1 at the input pin of BT ch.5 TIOB. |
| | 11 | Uses TIOB5_2 at the input pin of BT ch.5 TIOB. |

### [bit11:10] TIOA5E: TIOA5 Output Select bits

Selects output for TIOA5.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.5 TIOA. [Initial value] |
| | 01 | Uses TIOA5_0 at the output pin of BT ch.5 TIOA. |
| | 10 | Uses TIOA5_1 at the output pin of BT ch.5 TIOA. |
| | 11 | Uses TIOA5_2 at the output pin of BT ch.5 TIOA. |

### [bit9:8] TIOA5S: TIOA5 Input Select bits

Selects input for TIOA5.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOA5_0 at the input pin of BT ch.5 TIOA. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOA5_1 at the input pin of BT ch.5 TIOA. |
| | 11 | Uses TIOA5_2 at the input pin of BT ch.5 TIOA. |

### [bit7:6] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

### [bit5:4] TIOB4S: TIOB4 Input Select bits

Selects input for TIOB4.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB4_0 at the input pin of BT ch.4 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB4_1 at the input pin of BT ch.4 TIOB. |
| | 11 | Uses TIOB4_2 at the input pin of BT ch.4 TIOB. |

### [bit3:2] TIOA4E: TIOA4 Output Select bits

Selects output for TIOA4.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.4 TIOA. [Initial value] |
| | 01 | Uses TIOA4_0 at the output pin of BT ch.4 TIOA. |
| | 10 | Uses TIOA4_1 at the output pin of BT ch.4 TIOA. |
| | 11 | Uses TIOA4_2 at the output pin of BT ch.4 TIOA. |

### [bit1:0] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

***Notes:***

- *TIOA*
  *Even channels are for output only.*
  *Odd channels are for both input and output.*
- *TIOB*
  *Input only.*
- *TIOA1, TIOA3, TIOA5, TIOA7(odd number of "A") are not bidirectional pins so that choose either input pin or output pin for them.*
  *When output is chosen for odd TIOA channel, input setting will be ignored.*

  *Example1 : Use TIOA1 as an output pin:*
  *When TIOA1 is output to TIOA1_0, select EPFR04:TIOA1E = 01.*
  *When TIOA1 is output to TIOA1_1, select EPFR04:TIOA1E = 10.*
  *When TIOA1 is output to TIOA1_2, select EPFR04:TIOA1E = 11.*
  *Settings for EPFR04:TIOA1S will be ignored.*
  *Select ADE=0, PFR=1 for selected pins (DDR will be ignored).*
  *All the output of other peripheral function pins which are also used by selected pins must be OFF.*

  *Example2 : When TIOA1 is used as an input pin:*
  *Select EPFR04:TIOA1E = 00.*
  *When TIOA1 is input from TIOA1_0, select EPFR04:TIOA1S = 00 or 01.*
  *When TIOA1 is input from TIOA1_1, select EPFR04:TIOA1S = 10.*
  *When TIOA1 is input from TIOA1_2, select EPFR04:TIOA1S = 11.*
  *Select ADE=0, PFR=1 for selected pins (DDR will be ignored).*
  *All the output of other peripheral function pins which are also used by selected pins must be OFF.*

  ** When a pin is set to input, the output of the pin which is also used by selected pins (GPIO, other peripheral function pins) can be input to the base timer as Feedback with a setting not described above.*
  *This register is not initialized by deep standby transition reset.*

## 4.14 Extended Pin Function Setting Register 06 (EPFR06)

The EPFR06 register assigns functions to external interrupt pins.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | EINT15S | | EINT14S | | EINT13S | | EINT12S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | EINT11S | | EINT10S | | EINT09S | | EINT08S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | EINT07S | | EINT06S | | EINT05S | | EINT04S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | EINT03S | | EINT02S | | EINT01S | | EINT00S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

### Register Function

#### [bit31:30] EINT15S: External Interrupt Input Select bits

Selects input for EINT15.

| bit31:30 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT15_0 at the input pin of EINT ch.15. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT15_1 at the input pin of EINT ch.15. |
| | 11 | Uses INT15_2 at the input pin of EINT ch.15. |

#### [bit29:28] EINT14S: External Interrupt Input Select bits

Selects input for EINT14.

| bit29:28 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT14_0 at the input pin of EINT ch.14. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT14_1 at the input pin of EINT ch.14. |
| | 11 | Uses INT14_2 at the input pin of EINT ch.14. |

### [bit27:26] EINT13S: External Interrupt Input Select bits

Selects input for EINT13.

| bit27:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT13_0 at the input pin of EINT ch.13. [Initial value] |
| | 01 | Same as Writing 00 |
| | 10 | Uses INT13_1 at the input pin of EINT ch.13. |
| | 11 | Uses INT13_2 at the input pin of EINT ch.13. |

### [bit25:24] EINT12S: External Interrupt Input Select bits

Selects input for EINT12.

| bit25:24 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT12_0 at the input pin of EINT ch.12. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT12_1 at the input pin of EINT ch.12. |
| | 11 | Uses INT12_2 at the input pin of EINT ch.12. |

### [bit23:22] EINT11S: External Interrupt Input Select bits

Selects input for EINT11.

| bit23:22 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT11_0 at the input pin of EINT ch.11. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT11_1 at the input pin of EINT ch.11. |
| | 11 | Uses INT11_2 at the input pin of EINT ch.11. |

### [bit21:20] EINT10S: External Interrupt Input Select bits

Selects input for EINT10.

| bit21:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT10_0 at the input pin of EINT ch.10. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT10_1 at the input pin of EINT ch.10. |
| | 11 | Uses INT10_2 at the input pin of EINT ch.10. |

### [bit19:18] EINT09S: External Interrupt Input Select bits

Selects input for EINT09.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT09_0 at the input pin of EINT ch.9. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT09_1 at the input pin of EINT ch.9. |
| | 11 | Uses INT09_2 at the input pin of EINT ch.9. |

### [bit17:16] EINT08S: External Interrupt Input Select bits

Selects input for EINT08.

| bit17:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT08_0 at the input pin of EINT ch.8. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT08_1 at the input pin of EINT ch.8. |
| | 11 | Uses INT08_2 at the input pin of EINT ch.8. |

### [bit15:14] EINT07S: External Interrupt Input Select bits

Selects input for EINT07.

| bit15:14 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT07_0 at the input pin of EINT ch.7. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT07_1 at the input pin of EINT ch.7. |
| | 11 | Uses INT07_2 at the input pin of EINT ch.7. |

### [bit13:12] EINT06S: External Interrupt Input Select bits

Selects input for EINT06.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT06_0 at the input pin of EINT ch.6. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT06_1 at the input pin of EINT ch.6. |
| | 11 | Uses INT06_2 at the input pin of EINT ch.6. |

### [bit11:10] EINT05S: External Interrupt Input Select bits

Selects input for EINT05.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT05_0 at the input pin of EINT ch.5. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT05_1 at the input pin of EINT ch.5. |
| | 11 | Uses INT05_2 at the input pin of EINT ch.5. |

### [bit9:8] EINT04S: External Interrupt Input Select bits

Selects input for EINT04.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT04_0 at the input pin of EINT ch.4. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT04_1 at the input pin of EINT ch.4. |
| | 11 | Uses INT04_2 at the input pin of EINT ch.4. |

### [bit7:6] EINT03S: External Interrupt Input Select bits

Selects input for EINT03.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT03_0 at the input pin of EINT ch.3. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT03_1 at the input pin of EINT ch.3. |
| | 11 | Uses INT03_2 at the input pin of EINT ch.3. |

### [bit5:4] EINT02S: External Interrupt Input Select bits

Selects input for EINT02.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT02_0 at the input pin of EINT ch.2. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT02_1 at the input pin of EINT ch.2. |
| | 11 | Uses INT02_2 at the input pin of EINT ch.2. |

### [bit3:2] EINT01S: External Interrupt Input Select bits

Selects input for EINT01.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT01_0 at the input pin of EINT ch.1. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT01_1 at the input pin of EINT ch.1. |
| | 11 | Uses INT01_2 at the input pin of EINT ch.1. |

### [bit1:0] EINT00S: External Interrupt Input Select bits

Selects input for EINT00.

| bit1:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT00_0 at the input pin of EINT ch.0. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT00_1 at the input pin of EINT ch.0. |
| | 11 | Uses INT00_2 at the input pin of EINT ch.0. |

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.15 Extended Pin Function Setting Register 07 (EPFR07)

The EPFR07 register assigns functions of multi-function serial ch.0 to ch.3.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | SCK3B | | SOT3B | |
| Attribute | - | | | | R/W | | R/W | |
| Initial value | - | | | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | SIN3S | | SCK2B | | SOT2B | | SIN2S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | SCK1B | | SOT1B | | SIN1S | | SCK0B | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | SOT0B | | SIN0S | | Reserved | | | |
| Attribute | R/W | | R/W | | - | | | |
| Initial value | 00 | | 00 | | - | | | |

### Register Function

**[bit31:28] Reserved: Reserved bits**

"0b0000" is read from these bits.
When writing these bits, set them to "0b0000".

**[bit27:26] SCK3B: SCK3 Input/Output Select bits**

Selects input/output for SCK3.

| bit27:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK3_0 at the input pin of MFS ch.3 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK3_0 at the input pin of MFS ch.3 SCK.<br>Uses SCK3_0 at the output pin. |
| | 10 | Uses SCK3_1 at the input pin of MFS ch.3 SCK.<br>Uses SCK3_1 at the output pin. |
| | 11 | Uses SCK3_2 at the input pin of MFS ch.3 SCK.<br>Uses SCK3_2 at the output pin. |

### [bit25:24] SOT3B: SOT3 Input/Output Select bits

Selects input/output for SOT3.

| bit25:24 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT3_0 at the input pin of MFS ch.3 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT3_0 at the input pin of MFS ch.3 SOT.<br>Uses SOT3_0 at the output pin. |
| | 10 | Uses SOT3_1 at the input pin of MFS ch.3 SOT.<br>Uses SOT3_1 at the output pin. |
| | 11 | Uses SOT3_2 at the input pin of MFS ch.3 SOT.<br>Uses SOT3_2 at the output pin. |

### [bit23:22] SIN3S: SIN3 Input Select bits

Selects input for SIN3.

| bit23:22 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN3_0 at the input pin of MFS ch.3 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN3_1 at the input pin of MFS ch.3 SIN. |
| | 11 | Uses SIN3_2 at the input pin of MFS ch.3 SIN. |

### [bit21:20] SCK2B: SCK2 Input/Output Select bits

Selects input/output for SCK2.

| bit21:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK2_0 at the input pin of MFS ch.2 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK2_0 at the input pin of MFS ch.2 SCK.<br>Uses SCK2_0 at the output pin. |
| | 10 | Uses SCK2_1 at the input pin of MFS ch.2 SCK.<br>Uses SCK2_1 at the output pin. |
| | 11 | Uses SCK2_2 at the input pin of MFS ch.2 SCK.<br>Uses SCK2_2 at the output pin. |

### [bit19:18] SOT2B: SOT2 Input/Output Select bits

Selects input/output for SOT2.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT2_0 at the input pin of MFS ch.2 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT2_0 at the input pin of MFS ch.2 SOT.<br>Uses SOT2_0 at the output pin. |
| | 10 | Uses SOT2_1 at the input pin of MFS ch.2 SOT.<br>Uses SOT2_1 at the output pin. |
| | 11 | Uses SOT2_2 at the input pin of MFS ch.2 SOT.<br>Uses SOT2_2 at the output pin. |

### [bit17:16] SIN2S: SIN2 Input Select bits

Selects input for SIN2.

| bit17:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN2_0 at the input pin of MFS ch.2 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN2_1 at the input pin of MFS ch.2 SIN. |
| | 11 | Uses SIN2_2 at the input pin of MFS ch.2 SIN. |

### [bit15:14] SCK1B: SCK1 Input/Output Select bits

Selects input/output for SCK1.

| bit15:14 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK1_0 at the input pin of MFS ch.1 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK1_0 at the input pin of MFS ch.1 SCK.<br>Uses SCK1_0 at the output pin. |
| | 10 | Uses SCK1_1 at the input pin of MFS ch.1 SCK.<br>Uses SCK1_1 at the output pin. |
| | 11 | Uses SCK1_2 at the input pin of MFS ch.1 SCK.<br>Uses SCK1_2 at the output pin. |

### [bit13:12] SOT1B: SOT1 Input/Output Select bits

Selects input/output for SOT1.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT1_0 at the input pin of MFS ch.1 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT1_0 at the input pin of MFS ch.1 SOT.<br>Uses SOT1_0 at the output pin. |
| | 10 | Uses SOT1_1 at the input pin of MFS ch.1 SOT.<br>Uses SOT1_1 at the output pin. |
| | 11 | Uses SOT1_2 at the input pin of MFS ch.1 SOT.<br>Uses SOT1_2 at the output pin. |

### [bit11:10] SIN1S: SIN1 Input Select bits

Selects input for SIN1.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN1_0 at the input pin of MFS ch.1 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN1_1 at the input pin of MFS ch.1 SIN. |
| | 11 | Uses SIN1_2 at the input pin of MFS ch.1 SIN. |

### [bit9:8] SCK0B: SCK0 Input/Output Select bits

Selects input/output for SCK0.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK0_0 at the input pin of MFS ch.0 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK0_0 at the input pin of MFS ch.0 SCK.<br>Uses SCK0_0 at the output pin. |
| | 10 | Uses SCK0_1 at the input pin of MFS ch.0 SCK.<br>Uses SCK0_1 at the output pin. |
| | 11 | Uses SCK0_2 at the input pin of MFS ch.0 SCK.<br>Uses SCK0_2 at the output pin. |

### [bit7:6] SOT0B: SOT0 Input/Output Select bits

Selects input/output for SOT0.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT0_0 at the input pin of MFS ch.0 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT0_0 at the input pin of MFS ch.0 SOT.<br>Uses SOT0_0 at the output pin. |
| | 10 | Uses SOT0_1 at the input pin of MFS ch.0 SOT.<br>Uses SOT0_1 at the output pin. |
| | 11 | Uses SOT0_2 at the input pin of MFS ch.0 SOT.<br>Uses SOT0_2 at the output pin. |

### [bit5:4] SIN0S: SIN0 Input Select bits

Selects input for SIN0.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN0_0 at the input pin of MFS ch.0 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN0_1 at the input pin of MFS ch.0 SIN. |
| | 11 | Uses SIN0_2 at the input pin of MFS ch.0 SIN. |

### [bit3:0] Reserved: Reserved bits

"0b0000" is read from these bits.
When writing these bits, set them to "0b0000".

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.16 Extended Pin Function Setting Register 08 (EPFR08)

The EPFR08 register assigns functions of multi-function serial ch.4 to ch.7.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | SCK7B | | SOT7B | |
| Attribute | - | | | | R/W | | R/W | |
| Initial value | - | | | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | SIN7S | | SCK6B | | SOT6B | | SIN6S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | SCK5B | | SOT5B | | SIN5S | | SCK4B | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | SOT4B | | SIN4S | | CTS4S | | RTS4E | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

### Register Function

**[bit31:28] Reserved: Reserved bits**

"0b0000" is read from these bits.
When writing these bits, set them to "0b0000".

**[bit27:26] SCK7B: SCK7 Input/Output Select bits**

Selects input/output for SCK7.

| bit27:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK7_0 at the input pin of MFS ch.7 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK7_0 at the input pin of MFS ch.7 SCK.<br>Uses SCK7_0 at the output pin. |
| | 10 | Uses SCK7_1 at the input pin of MFS ch.7 SCK.<br>Uses SCK7_1 at the output pin. |
| | 11 | Uses SCK7_2 at the input pin of MFS ch.7 SCK.<br>Uses SCK7_2 at the output pin. |

### [bit25:24] SOT7B: SOT7 Input/Output Select bits

Selects input/output for SOT7.

| bit25:24 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT7_0 at the input pin of MFS ch.7 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT7_0 at the input pin of MFS ch.7 SOT.<br>Uses SOT7_0 at the output pin. |
| | 10 | Uses SOT7_1 at the input pin of MFS ch.7 SOT.<br>Uses SOT7_1 at the output pin. |
| | 11 | Uses SOT7_2 at the input pin of MFS ch.7 SOT.<br>Uses SOT7_2 at the output pin. |

### [bit23:22] SIN7S: SIN7 Input Select bits

Selects input for SIN7.

| bit23:22 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN7_0 at the input pin of MFS ch.7 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN7_1 at the input pin of MFS ch.7 SIN. |
| | 11 | Uses SIN7_2 at the input pin of MFS ch.7 SIN. |

### [bit21:20] SCK6B: SCK6 Input/Output Select bits

Selects input/output for SCK6.

| bit21:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK6_0 at the input pin of MFS ch.6 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK6_0 at the input pin of MFS ch.6 SCK.<br>Uses SCK6_0 at the output pin. |
| | 10 | Uses SCK6_1 at the input pin of MFS ch.6 SCK.<br>Uses SCK6_1 at the output pin. |
| | 11 | Uses SCK6_2 at the input pin of MFS ch.6 SCK.<br>Uses SCK6_2 at the output pin. |

### [bit19:18] SOT6B: SOT6 Input/Output Select bits

Selects input/output for SOT6.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT6_0 at the input pin of MFS ch.6 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT6_0 at the input pin of MFS ch.6 SOT.<br>Uses SOT6_0 at the output pin. |
| | 10 | Uses SOT6_1 at the input pin of MFS ch.6 SOT.<br>Uses SOT6_1 at the output pin. |
| | 11 | Uses SOT6_2 at the input pin of MFS ch.6 SOT.<br>Uses SOT6_2 at the output pin. |

### [bit17:16] SIN6S: SIN6 Input Select bits

Selects input for SIN6.

| bit17:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN6_0 at the input pin of MFS ch.6 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN6_1 at the input pin of MFS ch.6 SIN. |
| | 11 | Uses SIN6_2 at the input pin of MFS ch.6 SIN. |

### [bit15:14] SCK5B: SCK5 Input/Output Select bits

Selects input/output for SCK5.

| bit15:14 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK5_0 at the input pin of MFS ch.5 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK5_0 at the input pin of MFS ch.5 SCK.<br>Uses SCK5_0 at the output pin. |
| | 10 | Uses SCK5_1 at the input pin of MFS ch.5 SCK.<br>Uses SCK5_1 at the output pin. |
| | 11 | Uses SCK5_2 at the input pin of MFS ch.5 SCK.<br>Uses SCK5_2 at the output pin. |

### [bit13:12] SOT5B: SOT5 Input/Output Select bits

Selects input/output for SOT5.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT5_0 at the input pin of MFS ch.5 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT5_0 at the input pin of MFS ch.5 SOT.<br>Uses SOT5_0 at the output pin. |
| | 10 | Uses SOT5_1 at the input pin of MFS ch.5 SOT.<br>Uses SOT5_1 at the output pin. |
| | 11 | Uses SOT5_2 at the input pin of MFS ch.5 SOT.<br>Uses SOT5_2 at the output pin. |

### [bit11:10] SIN5S: SIN5 Input Select bits

Selects input for SIN5.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN5_0 at the input pin of MFS ch.5 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN5_1 at the input pin of MFS ch.5 SIN. |
| | 11 | Uses SIN5_2 at the input pin of MFS ch.5 SIN. |

### [bit9:8] SCK4B: SCK4 Input/Output Select bits

Selects input/output for SCK4.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK4_0 at the input pin of MFS ch.4 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK4_0 at the input pin of MFS ch.4 SCK.<br>Uses SCK4_0 at the output pin. |
| | 10 | Uses SCK4_1 at the input pin of MFS ch.4 SCK.<br>Uses SCK4_1 at the output pin. |
| | 11 | Uses SCK4_2 at the input pin of MFS ch.4 SCK.<br>Uses SCK4_2 at the output pin. |

### [bit7:6] SOT4B: SOT4 Input/Output Select bits

Selects input/output for SOT4.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT4_0 at the input pin of MFS ch.4 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT4_0 at the input pin of MFS ch.4 SOT.<br>Uses SOT4_0 at the output pin. |
| | 10 | Uses SOT4_1 at the input pin of MFS ch.4 SOT.<br>Uses SOT4_1 at the output pin. |
| | 11 | Uses SOT4_2 at the input pin of MFS ch.4 SOT.<br>Uses SOT4_2 at the output pin. |

### [bit5:4] SIN4S: SIN4 Input Select bits

Selects input for SIN4.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN4_0 at the input pin of MFS ch.4 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN4_1 at the input pin of MFS ch.4 SIN. |
| | 11 | Uses SIN4_2 at the input pin of MFS ch.4 SIN. |

### [bit3:2] CTS4S: CTS4 Input Select bits

Selects input for CTS4.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses CTS4_0 at the input pin of MFS ch.4 CTS. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses CTS4_1 at the input pin of MFS ch.4 CTS. |
| | 11 | Uses CTS4_2 at the input pin of MFS ch.4 CTS. |

**[bit1:0] RTS4E: RTS4 Output Select bits**

Selects output for RTS4.

| bit1:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for MFS ch.4 RTS. [Initial value] |
| | 01 | Uses RTS4_0 at the output pin of MFS ch.4 RTS. |
| | 10 | Uses RTS4_1 at the output pin of MFS ch.4 RTS. |
| | 11 | Uses RTS4_2 at the output pin of MFS ch.4 RTS. |

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.17 Extended Pin Function Setting Register 09 (EPFR09)

The EPFR09 register assigns functions to CAN, ADC trigger, and QPRC peripheral pins.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | CTX1E | | CRX1S | | CTX0E | | CRX0S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | ADTRG2S | | | | ADTRG1S | | | |
| Attribute | R/W | | | | R/W | | | |
| Initial value | 0000 | | | | 0000 | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | ADTRG0S | | | | QZIN1S | | QBIN1S | |
| Attribute | R/W | | | | R/W | | R/W | |
| Initial value | 0000 | | | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | QAIN1S | | QZIN0S | | QBIN0S | | QAIN0S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

### Register Function

#### [bit31:30] CTX1E: CTX1E Output Select bits

Selects output for CAN TX1.

| bit31:30 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for CAN ch.1 TX. [Initial value] |
| | 01 | Sets the output pin of CAN ch.1 TX to TX1_0. |
| | 10 | Sets the output pin of CAN ch.1 TX to TX1_1. |
| | 11 | Sets the output pin of CAN ch.1 TX to TX1_2. |

#### [bit29:28] CRX1S: CRX1S Input Select bits

Selects input for CAN RX1.

| bit29:28 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Sets the input pin of CAN ch.1 RX to RX1_0. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Sets the input pin of CAN ch.1 RX to RX1_1. |
| | 11 | Sets the input pin of CAN ch.1 RX to RX1_2. |

### [bit27:26] CTX0E: CTX0E Output Select bits

Selects output for CAN TX0.

| bit27:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for CAN ch.0 TX. [Initial value] |
| | 01 | Sets the output pin of CAN ch.0 TX to TX0_0. |
| | 10 | Sets the output pin of CAN ch.0 TX to TX0_1. |
| | 11 | Sets the output pin of CAN ch.0 TX to TX0_2. |

### [bit25:24] CRX0S: CRX0S Input Select bits

Selects input for CAN RX0.

| bit25:24 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Sets the input pin of CAN ch.0 RX to RX0_0. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Sets the input pin of CAN ch.0 RX to RX0_1. |
| | 11 | Sets the input pin of CAN ch.0 RX to RX0_2. |

### [bit23:20] ADTRG2S: ADTRG2 Input Select bits

Selects input for ADTRG2.

| bit23:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0000 | Uses ADTG_0 at the input pin of ADC unit 2's startup trigger. [Initial value] |
| | 0001 | Same as Writing 0000. |
| | 0010 | Uses ADTG_1 at the input pin of ADC unit 2's startup trigger. |
| | 0011 | Uses ADTG_2 at the input pin of ADC unit 2's startup trigger. |
| | 0100 | Uses ADTG_3 at the input pin of ADC unit 2's startup trigger. |
| | 0101 | Uses ADTG_4 at the input pin of ADC unit 2's startup trigger. |
| | 0110 | Uses ADTG_5 at the input pin of ADC unit 2's startup trigger. |
| | 0111 | Uses ADTG_6 at the input pin of ADC unit 2's startup trigger. |
| | 1000 | Uses ADTG_7 at the input pin of ADC unit 2's startup trigger. |
| | 1001 | Uses ADTG_8 at the input pin of ADC unit 2's startup trigger. |
| Writing other data | | Setting is prohibited. |

### [bit19:16] ADTRG1S: ADTRG1 Input Select bits

Selects input for ADTRG1.

| bit19:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0000 | Uses ADTG_0 at the input pin of ADC unit 1's startup trigger. [Initial value] |
| | 0001 | Same as Writing 0000. |
| | 0010 | Uses ADTG_1 at the input pin of ADC unit 1's startup trigger. |
| | 0011 | Uses ADTG_2 at the input pin of ADC unit 1's startup trigger. |
| | 0100 | Uses ADTG_3 at the input pin of ADC unit 1's startup trigger. |
| | 0101 | Uses ADTG_4 at the input pin of ADC unit 1's startup trigger. |
| | 0110 | Uses ADTG_5 at the input pin of ADC unit 1's startup trigger. |
| | 0111 | Uses ADTG_6 at the input pin of ADC unit 1's startup trigger. |
| | 1000 | Uses ADTG_7 at the input pin of ADC unit 1's startup trigger. |
| | 1001 | Uses ADTG_8 at the input pin of ADC unit 1's startup trigger. |
| Writing other data | | Setting is prohibited. |

### [bit15:12] ADTRG0S: ADTRG0 Input Select bits

Selects input for ADTRG0.

| bit15:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0000 | Uses ADTG_0 at the input pin of ADC unit 0's startup trigger. [Initial value] |
| | 0001 | Same as Writing 0000. |
| | 0010 | Uses ADTG_1 at the input pin of ADC unit 0's startup trigger. |
| | 0011 | Uses ADTG_2 at the input pin of ADC unit 0's startup trigger. |
| | 0100 | Uses ADTG_3 at the input pin of ADC unit 0's startup trigger. |
| | 0101 | Uses ADTG_4 at the input pin of ADC unit 0's startup trigger. |
| | 0110 | Uses ADTG_5 at the input pin of ADC unit 0's startup trigger. |
| | 0111 | Uses ADTG_6 at the input pin of ADC unit 0's startup trigger. |
| | 1000 | Uses ADTG_7 at the input pin of ADC unit 0's startup trigger. |
| | 1001 | Uses ADTG_8 at the input pin of ADC unit 0's startup trigger. |
| Writing other data | | Setting is prohibited. |

### [bit11:10] QZIN1S: QZIN1S Input Select bits

Selects input for QPRC ZIN1.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses ZIN1_0 at the input pin of QPRC ch.1's ZIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses ZIN1_1 at the input pin of QPRC ch.1's ZIN. |
| | 11 | Uses ZIN1_2 at the input pin of QPRC ch.1's ZIN. |

### [bit9:8] QBIN1S: QBIN1S Input Select bits

Selects input for QPRC BIN1.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses BIN1_0 at the input pin of QPRC ch.1's BIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses BIN1_1 at the input pin of QPRC ch.1's BIN. |
| | 11 | Uses BIN1_2 at the input pin of QPRC ch.1's BIN. |

### [bit7:6] QAIN1S: QAIN1S Input Select bits

Selects input for QPRC AIN1.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses AIN1_0 at the input pin of QPRC ch.1's AIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses AIN1_1 at the input pin of QPRC ch.1's AIN. |
| | 11 | Uses AIN1_2 at the input pin of QPRC ch.1's AIN. |

### [bit5:4] QZIN0S: QZIN0S Input Select bits

Selects input for QPRC ZIN0. If ZIN0_3 is used, set this bit together with QZIN0S[2] bit in Extended Pin Function Setting Register 21 (EPFR21).

| EPFR21[2],bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses ZIN0_0 at the input pin of QPRC ch.0's ZIN. [Initial value] |
| | 001 | Same as Writing 00. |
| | 010 | Uses ZIN0_1 at the input pin of QPRC ch.0's ZIN. |
| | 011 | Uses ZIN0_2 at the input pin of QPRC ch.0's ZIN. |
| | 100 | Uses ZIN0_3 at the input pin of QPRC ch.0's ZIN. |
| | Others | Setting is prohibited. |

### [bit3:2] QBIN0S: QBIN0S Input Select bits

Selects input for QPRC BIN0. If BIN0_3 is used, set this bit together with QBIN0S[2] bit in Extended Pin Function Setting Register 21 (EPFR21).

| EPFR21[1],bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses BIN0_0 at the input pin of QPRC ch.0's BIN. [Initial value] |
| | 001 | Same as Writing 00. |
| | 010 | Uses BIN0_1 at the input pin of QPRC ch.0's BIN. |
| | 011 | Uses BIN0_2 at the input pin of QPRC ch.0's BIN. |
| | 100 | Uses BIN0_3 at the input pin of QPRC ch.0's BIN. |
| | Others | Setting is prohibited |

**[bit1:0] QAIN0S: QAIN0S Input Select bits**

Selects input for QPRC AIN0. If AIN0_3 is used, set this bit together with QAIN0S[2] bit in Extended Pin Function Setting Register 21 (EPFR21)

| EPFR21[0],bit1:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | Uses AIN0_0 at the input pin of QPRC ch.0's AIN. [Initial value] |
| | 001 | Same as Writing 00. |
| | 010 | Uses AIN0_1 at the input pin of QPRC ch.0's AIN. |
| | 011 | Uses AIN0_2 at the input pin of QPRC ch.0's AIN. |
| | 100 | Uses AIN0_3 at the input pin of QPRC ch.0's AIN. |
| | Others | Setting is prohibited. |

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.18 Extended Pin Function Setting Register 12 (EPFR12)

The EPFR12 register assigns functions to pins of ch.8, ch.9, ch.10, and ch.11 of the base timer.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB11S | | TIOA11E | | TIOA11S | |
| Attribute | - | | R/W | | R/W | | R/W | |
| Initial value | - | | 00 | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB10S | | TIOA10E | | Reserved | |
| Attribute | - | | R/W | | R/W | | - | |
| Initial value | - | | 00 | | 00 | | - | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB9S | | TIOA9E | | TIOA9S | |
| Attribute | - | | R/W | | R/W | | R/W | |
| Initial value | - | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB8S | | TIOA8E | | Reserved | |
| Attribute | - | | R/W | | R/W | | - | |
| Initial value | - | | 00 | | 00 | | - | |

### Register Function

#### [bit31:30] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

#### [bit29:28] TIOB11S: TIOB11 Input Select bits

Selects input for TIOB11.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB11_0 at the input pin of BT ch.11 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB11_1 at the input pin of BT ch.11 TIOB. |
| | 11 | Uses TIOB11_2 at the input pin of BT ch.11 TIOB. |

#### [bit27:26] TIOA11E: TIOA11 Output Select bits

Selects output for TIOA11.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.11 TIOA. [Initial value] |
| | 01 | Uses TIOA11_0 at the output pin of BT ch.11 TIOA. |
| | 10 | Uses TIOA11_1 at the output pin of BT ch.11 TIOA. |
| | 11 | Uses TIOA11_2 at the output pin of BT ch.11 TIOA. |

### [bit25:24] TIOA11S: TIOA11 Input Select bits

Selects input for TIOA11.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOA11_0 at the input pin of BT ch.11 TIOA. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOA11_1 at the input pin of BT ch.11 TIOA. |
| | 11 | Uses TIOA11_2 at the input pin of BT ch.11 TIOA. |

### [bit23:22] Reserved: Reserved bits

"0b00" is read out from these bits.

When writing these bits, set them to "0b00".

### [bit21:20] TIOB10S: TIOB10 Input Select bits

Selects input for TIOB10.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB10_0 at the input pin of BT ch.10 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB10_1 at the input pin of BT ch.10 TIOB. |
| | 11 | Uses TIOB10_2 at the input pin of BT ch.10 TIOB. |

### [bit19:18] TIOA10E: TIOA10 Output Select bits

Selects output for TIOA10.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.10 TIOA. [Initial value] |
| | 01 | Uses TIOA10_0 at the output pin of BT ch.10 TIOA. |
| | 10 | Uses TIOA10_1 at the output pin of BT ch.10 TIOA. |
| | 11 | Uses TIOA10_2 at the output pin of BT ch.10 TIOA. |

### [bit17:14] Reserved: Reserved bits

"0b0000" is read out from these bits.

When writing these bits, set them to "0b0000".

### [bit13:12] TIOB9S: TIOB9 Input Select bits

Selects input for TIOB9.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB9_0 at the input pin of BT ch.9 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB9_1 at the input pin of BT ch.9 TIOB. |
| | 11 | Uses TIOB9_2 at the input pin of BT ch.9 TIOB. |

### [bit11:10] TIOA9E: TIOA9 Output Select bits

Selects output for TIOA9.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.9 TIOA. [Initial value] |
| | 01 | Uses TIOA9_0 at the output pin of BT ch.9 TIOA. |
| | 10 | Uses TIOA9_1 at the output pin of BT ch.9 TIOA. |
| | 11 | Uses TIOA9_2 at the output pin of BT ch.9 TIOA. |

### [bit9:8] TIOA9S: TIOA9 Input Select bits

Selects input for TIOA9.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOA9_0 at the input pin of BT ch.9 TIOA. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOA9_1 at the input pin of BT ch.9 TIOA. |
| | 11 | Uses TIOA9_2 at the input pin of BT ch.9 TIOA. |

### [bit7:6] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

### [bit5:4] TIOB8S: TIOB8 Input Select bits

Selects input for TIOB8.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB8_0 at the input pin of BT ch.8 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB8_1 at the input pin of BT ch.8 TIOB. |
| | 11 | Uses TIOB8_2 at the input pin of BT ch.8 TIOB. |

### [bit3:2] TIOA8E: TIOA8 Output Select bits

Selects output for TIOA8.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.8 TIOA. [Initial value] |
| | 01 | Uses TIOA8_0 at the output pin of BT ch.8 TIOA. |
| | 10 | Uses TIOA8_1 at the output pin of BT ch.8 TIOA. |
| | 11 | Uses TIOA8_2 at the output pin of BT ch.8 TIOA. |

### [bit1:0] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

***Notes:***

- *TIOA*
  *Even channels are for output only.*
  *Odd channels are for both input and output.*
- *TIOB*
  *Input only.*
- *TIOA9, TIOA11, TIOA13, TIOA15 (odd number of "A") are not bidirectional pins so that choose either input pin or output pin for them.*
  *When output is chosen for odd TIOA channel, input setting will be ignored.*

  *Example1 : Use TIOA11 as an output pin:*
  *When TIOA11 is output to TIOA11_0, select EPFR12:TIOA11E = 01.*
  *When TIOA11 is output to TIOA11_1, select EPFR12:TIOA11E = 10.*
  *When TIOA11 is output to TIOA11_2, select EPFR12:TIOA11E = 11.*
  *Settings for EPFR12:TIOA11S will be ignored.*
  *Select ADE=0, PFR=1 for selected pins (DDR will be ignored).*
  *All the output of other peripheral function pins which are also used by selected pins must be OFF.*

  *Example2 : When TIOA11 is used as an input pin:*
  *Select EPFR12:TIOA11E = 00.*
  *When TIOA11 is input from TIOA11_0, select EPFR12:TIOA11S = 00 or 01.*
  *When TIOA11 is input from TIOA11_1, select EPFR12:TIOA11S = 10.*
  *When TIOA11 is input from TIOA11_2, select EPFR12:TIOA11S = 11.*
  *Select ADE=0, PFR=1 for selected pins (DDR will be ignored).*
  *All the output of other peripheral function pins which are also used by selected pins must be OFF.*

  ** When a pin is set to input, the output of the pin which is also used by selected pins (GPIO, other peripheral function pins) can be input to the base timer as Feedback with a setting not described above.*
- *This register is not initialized by deep standby transition reset.*

## 4.19 Extended Pin Function Setting Register 13 (EPFR13)

The EPFR13 register assigns functions to pins of ch.12, ch.13, ch.14, and ch.15 of the base timer.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB15S | | TIOA15E | | TIOA15S | |
| Attribute | - | | R/W | | R/W | | R/W | |
| Initial value | - | | 00 | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB14S | | TIOA14E | | Reserved | |
| Attribute | - | | R/W | | R/W | | - | |
| Initial value | - | | 00 | | 00 | | - | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB13S | | TIOA13E | | TIOA13S | |
| Attribute | - | | R/W | | R/W | | R/W | |
| Initial value | - | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | TIOB12S | | TIOA12E | | Reserved | |
| Attribute | - | | R/W | | R/W | | - | |
| Initial value | - | | 00 | | 00 | | - | |

### Register Function

**[bit31:30] Reserved: Reserved bits**

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

**[bit29:28] TIOB15S: TIOB15 Input Select bits**

Selects input for TIOB15.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB15_0 at the input pin of BT ch.15 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB15_1 at the input pin of BT ch.15 TIOB. |
| | 11 | Uses TIOB15_2 at the input pin of BT ch.15 TIOB. |

**[bit27:26] TIOA15E: TIOA15 Output Select bits**

Selects output for TIOA15.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.15 TIOA. [Initial value] |
| | 01 | Uses TIOA15_0 at the output pin of BT ch.15 TIOA. |
| | 10 | Uses TIOA15_1 at the output pin of BT ch.15 TIOA. |
| | 11 | Uses TIOA15_2 at the output pin of BT ch.15 TIOA. |

### [bit25:24] TIOA15S: TIOA15 Input Select bits

Selects input for TIOA15.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOA15_0 at the input pin of BT ch.15 TIOA. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOA15_1 at the input pin of BT ch.15 TIOA. |
| | 11 | Uses TIOA15_2 at the input pin of BT ch.15 TIOA. |

### [bit23:22] Reserved: Reserved bits

"0b00" is read out from these bits.

When writing these bits, set them to "0b00".

### [bit21:20] TIOB14S: TIOB14 Input Select bits

Selects input for TIOB14.

| bit21:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB14_0 at the input pin of BT ch.14 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB14_1 at the input pin of BT ch.14 TIOB. |
| | 11 | Uses TIOB14_2 at the input pin of BT ch.14 TIOB. |

### [bit19:18] TIOA14E: TIOA14 Output Select bits

Selects output for TIOA14.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.14 TIOA. [Initial value] |
| | 01 | Uses TIOA14_0 at the output pin of BT ch.14 TIOA. |
| | 10 | Uses TIOA14_1 at the output pin of BT ch.14 TIOA. |
| | 11 | Uses TIOA14_2 at the output pin of BT ch.14 TIOA. |

### [bit17:14] Reserved: Reserved bits

"0b0000" is read out from these bits.

When writing these bits, set them to "0b0000".

### [bit13:12] TIOB13S: TIOB13 Input Select bits

Selects input for TIOB13.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB13_0 at the input pin of BT ch.13 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB13_1 at the input pin of BT ch.13 TIOB. |
| | 11 | Uses TIOB13_2 at the input pin of BT ch.13 TIOB. |

### [bit11:10] TIOA13E: TIOA13 Output Select bits

Selects output for TIOA13.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.13 TIOA. [Initial value] |
| | 01 | Uses TIOA13_0 at the output pin of BT ch.13 TIOA. |
| | 10 | Uses TIOA13_1 at the output pin of BT ch.13 TIOA. |
| | 11 | Uses TIOA13_2 at the output pin of BT ch.13 TIOA. |

### [bit9:8] TIOA13S: TIOA13 Input Select bits

Selects input for TIOA13.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOA13_0 at the input pin of BT ch.13 TIOA. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOA13_1 at the input pin of BT ch.13 TIOA. |
| | 11 | Uses TIOA13_2 at the input pin of BT ch.13 TIOA. |

### [bit7:6] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

### [bit5:4] TIOB12S: TIOB12 Input Select bits

Selects input for TIOB12.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses TIOB12_0 at the input pin of BT ch.12 TIOB. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses TIOB12_1 at the input pin of BT ch.12 TIOB. |
| | 11 | Uses TIOB12_2 at the input pin of BT ch.12 TIOB. |

### [bit3:2] TIOA12E: TIOA12 Output Select bits

Selects output for TIOA12.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce the output of the BT ch.12 TIOA. [Initial value] |
| | 01 | Uses TIOA12_0 at the output pin of BT ch.12 TIOA. |
| | 10 | Uses TIOA12_1 at the output pin of BT ch.12 TIOA. |
| | 11 | Uses TIOA12_2 at the output pin of BT ch.12 TIOA. |

### [bit1:0] Reserved: Reserved bits

"0b00" is read out from these bits.
When writing these bits, set them to "0b00".

***Notes:***

- *TIOA*
  *Even channels are for output only.*
  *Odd channels are for both input and output.*
- *TIOB*
  *Input only.*
- *TIOA9, TIOA11, TIOA13, TIOA15 (odd number of "A") are not bidirectional pins so that choose either input pin or output pin for them.*
  *When output is chosen for odd TIOA channel, input setting will be ignored.*

  *Example1 : Use TIOA11 as an output pin:*
  *When TIOA11 is output to TIOA11_0, select EPFR12:TIOA11E = 01.*
  *When TIOA11 is output to TIOA11_1, select EPFR12:TIOA11E = 10.*
  *When TIOA11 is output to TIOA11_2, select EPFR12:TIOA11E = 11.*
  *Settings for EPFR12:TIOA11S will be ignored.*
  *Select ADE=0, PFR=1 for selected pins (DDR will be ignored).*
  *All the output of other peripheral function pins which are also used by selected pins must be OFF.*

  *Example2 : When TIOA11 is used as an input pin:*
  *Select EPFR12:TIOA11E = 00.*
  *When TIOA11 is input from TIOA11_0, select EPFR12:TIOA11S = 00 or 01.*
  *When TIOA11 is input from TIOA11_1, select EPFR12:TIOA11S = 10.*
  *When TIOA11 is input from TIOA11_2, select EPFR12:TIOA11S = 11.*
  *Select ADE=0, PFR=1 for selected pins (DDR will be ignored).*
  *All the output of other peripheral function pins which are also used by selected pins must be OFF.*

  ** When a pin is set to input, the output of the pin which is also used by selected pins (GPIO, other peripheral function pins) can be input to the base timer as Feedback with a setting not described above.*
- *This register is not initialized by deep standby transition reset.*

## 4.20 Extended Pin Function Setting Register 14 (EPFR14)

EPFR14 register sets the function assignment to QPRC pins.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | QZIN2S | | QBIN2S | | QAIN2S | |
| Attribute | - | | R/W | | R/W | | R/W | |
| Initial value | - | | 00 | | 00 | | 00 | |

### Register Function

**[bit31:6] Reserved: Reserved bits**

"0" is read out from these bits.
When writing these bits, set them to "0".

**[bit5:4] QZIN2S: QPRC-ch.2 ZIN Input Pin bits**

Selects input for QPRC-ch.2 as ZIN.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | ZIN2_0 is used as ZIN, the input pin of QPRC ch.2. [Initial value] |
| | 01 | ZIN2_0 is used as ZIN, the input pin of QPRC ch.2. |
| | 10 | ZIN2_1 is used as ZIN, the input pin of QPRC ch.2. |
| | 11 | ZIN2_2 is used as ZIN, the input pin of QPRC ch.2. |

**[bit3:2] QBIN2S: QPRC-ch.2 BIN Input Pin bits**

Selects input for QPRC-ch.2 as BIN.

| bit | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | BIN2_0 is used as BIN, the input pin of QPRC ch.2. [Initial value] |
| | 01 | BIN2_0 is used as BIN, the input pin of QPRC ch.2. |
| | 10 | BIN2_1 is used as BIN, the input pin of QPRC ch.2. |
| | 11 | BIN2_2 is used as BIN, the input pin of QPRC ch.2. |

**[bit1:0] QAIN2S: QPRC-ch.2 AIN Input Pin bits**

Selects input for QPRC-ch.2 as AIN.

| bit1:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | AIN2_0 is used as AIN, the input pin of QPRC ch.2. [Initial value] |
| | 01 | AIN2_0 is used as AIN, the input pin of QPRC ch.2. |
| | 10 | AIN2_1 is used as AIN, the input pin of QPRC ch.2. |
| | 11 | AIN2_2 is used as AIN, the input pin of QPRC ch.2. |

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.21 Extended Pin Function Setting Register 15 (EPFR15)

EPFR15 register sets the function assignment to external interrupt pins.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | EINT31S | | EINT30S | | EINT29S | | EINT28S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | EINT27S | | EINT26S | | EINT25S | | EINT24S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | EINT23S | | EINT22S | | EINT21S | | EINT20S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | EINT19S | | EINT18S | | EINT17S | | EINT16S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

### Register Function

#### [bit31:30] EINT31S: External Interrupt Input Select bits

Selects input for EINT31.

| bit31:30 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT31_0 at the input pin of EINT ch.31. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT31_1 at the input pin of EINT ch.31. |
| | 11 | Uses INT31_2 at the input pin of EINT ch.31. |

#### [bit29:28] EINT30S: External Interrupt Input Select bits

Selects input for EINT30.

| bit29:28 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT30_0 at the input pin of EINT ch.30. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT30_1 at the input pin of EINT ch.30. |
| | 11 | Uses INT30_2 at the input pin of EINT ch.30. |

### [bit27:26] EINT29S: External Interrupt Input Select bits

Selects input for EINT29.

| bit27:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT29_0 at the input pin of EINT ch.29. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT29_1 at the input pin of EINT ch.29. |
| | 11 | Uses INT29_2 at the input pin of EINT ch.29. |

### [bit25:24] EINT28S: External Interrupt Input Select bits

Selects input for EINT28.

| bit25:24 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT28_0 at the input pin of EINT ch.28. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT28_1 at the input pin of EINT ch.28. |
| | 11 | Uses INT28_2 at the input pin of EINT ch.28. |

### [bit23:22] EINT27S: External Interrupt Input Select bits

Selects input for EINT27.

| bit23:22 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT27_0 at the input pin of EINT ch.27. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT27_1 at the input pin of EINT ch.27. |
| | 11 | Uses INT27_2 at the input pin of EINT ch.27. |

### [bit21:20] EINT26S: External Interrupt Input Select bits

Selects input for EINT26.

| bit21:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT26_0 at the input pin of EINT ch.26. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT26_1 at the input pin of EINT ch.26. |
| | 11 | Uses INT26_2 at the input pin of EINT ch.26. |

### [bit19:18] EINT25S: External Interrupt Input Select bits

Selects input for EINT25.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT25_0 at the input pin of EINT ch.25. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT25_1 at the input pin of EINT ch.25. |
| | 11 | Uses INT25_2 at the input pin of EINT ch.25. |

### [bit17:16] EINT24S: External Interrupt Input Select bits

Selects input for EINT24.

| bit17:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT24_0 at the input pin of EINT ch.24. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT24_1 at the input pin of EINT ch.24. |
| | 11 | Uses INT24_2 at the input pin of EINT ch.24. |

### [bit15:14] EINT23S: External Interrupt Input Select bits

Selects input for EINT23.

| bit15:14 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT23_0 at the input pin of EINT ch.23. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT23_1 at the input pin of EINT ch.23. |
| | 11 | Uses INT23_2 at the input pin of EINT ch.23. |

### [bit13:12] EINT22S: External Interrupt Input Select bits

Selects input for EINT22.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT22_0 at the input pin of EINT ch.22. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT22_1 at the input pin of EINT ch.22. |
| | 11 | Uses INT22_2 at the input pin of EINT ch.22. |

### [bit11:10] EINT21S: External Interrupt Input Select bits

Selects input for EINT21.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT21_0 at the input pin of EINT ch.21. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT21_1 at the input pin of EINT ch.21. |
| | 11 | Uses INT21_2 at the input pin of EINT ch.21. |

### [bit9:8] EINT20S: External Interrupt Input Select bits

Selects input for EINT20.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT20_0 at the input pin of EINT ch.20. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT20_1 at the input pin of EINT ch.20. |
| | 11 | Uses INT20_2 at the input pin of EINT ch.20. |

### [bit7:6] EINT19S: External Interrupt Input Select bits

Selects input for EINT19.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT19_0 at the input pin of EINT ch.19. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT19_1 at the input pin of EINT ch.19. |
| | 11 | Uses INT19_2 at the input pin of EINT ch.19. |

### [bit5:4] EINT18S: External Interrupt Input Select bits

Selects input for EINT18.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT18_0 at the input pin of EINT ch.18. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT18_1 at the input pin of EINT ch.18. |
| | 11 | Uses INT18_2 at the input pin of EINT ch.18. |

### [bit3:2] EINT17S: External Interrupt Input Select bits

Selects input for EINT17.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT17_0 at the input pin of EINT ch.17. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT17_1 at the input pin of EINT ch.17. |
| | 11 | Uses INT17_2 at the input pin of EINT ch.17. |

### [bit1:0] EINT16S: External Interrupt Input Select bits

Selects input for EINT16.

| bit1:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses INT16_0 at the input pin of EINT ch.16. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses INT16_1 at the input pin of EINT ch.16. |
| | 11 | Uses INT16_2 at the input pin of EINT ch.16. |

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.22 Extended Pin Function Setting Register 16 (EPFR16)

The EPFR16 register assigns functions of multi-function serial channel 8, channel 9, channel 10, and channel 11.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | SCK11B | | SOT11B | |
| Attribute | - | | | | R/W | | R/W | |
| Initial value | - | | | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | SIN11S | | SCK10B | | SOT10B | | SIN10S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | SCK9B | | SOT9B | | SIN9S | | SCK8B | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | SOT8B | | SIN8S | | Reserved | | | |
| Attribute | R/W | | R/W | | - | | | |
| Initial value | 00 | | 00 | | - | | | |

### Register Function

**[bit31:28] Reserved: Reserved bits**

"0b0000" is read from these bits.
When writing these bits, set them to "0b0000".

**[bit27:26] SCK11B: SCK11 Input/Output Select bits**

Selects input/output for SCK11.

| bit27:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK11_0 at the input pin of MFS ch.11 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK11_0 at the input pin of MFS ch.11 SCK.<br>Uses SCK11_0 at the output pin. |
| | 10 | Uses SCK11_1 at the input pin of MFS ch.11 SCK.<br>Uses SCK11_1 at the output pin. |
| | 11 | Uses SCK11_2 at the input pin of MFS ch.11 SCK.<br>Uses SCK11_2 at the output pin. |

### [bit25:24] SOT11B: SOT11 Input/Output Select bits

Selects input/output for SOT11.

| bit25:24 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT11_0 at the input pin of MFS ch.11 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT11_0 at the input pin of MFS ch.11 SOT.<br>Uses SOT11_0 at the output pin. |
| | 10 | Uses SOT11_1 at the input pin of MFS ch.11 SOT.<br>Uses SOT11_1 at the output pin. |
| | 11 | Uses SOT11_2 at the input pin of MFS ch.11 SOT.<br>Uses SOT11_2 at the output pin. |

### [bit23:22] SIN11S: SIN11 Input Select bits

Selects input for SIN11.

| bit23:22 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN11_0 at the input pin of MFS ch.11 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN11_1 at the input pin of MFS ch.11 SIN. |
| | 11 | Uses SIN11_2 at the input pin of MFS ch.11 SIN. |

### [bit21:20] SCK10B: SCK10 Input/Output Select bits

Selects input/output for SCK10.

| bit21:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK10_0 at the input pin of MFS ch.10 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK10_0 at the input pin of MFS ch.10 SCK.<br>Uses SCK10_0 at the output pin. |
| | 10 | Uses SCK10_1 at the input pin of MFS ch.10 SCK.<br>Uses SCK10_1 at the output pin. |
| | 11 | Uses SCK10_2 at the input pin of MFS ch.10 SCK.<br>Uses SCK10_2 at the output pin. |

### [bit19:18] SOT10B: SOT10 Input/Output Select bits

Selects input/output for SOT10.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT10_0 at the input pin of MFS ch.10 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT10_0 at the input pin of MFS ch.10 SOT.<br>Uses SOT10_0 at the output pin. |
| | 10 | Uses SOT10_1 at the input pin of MFS ch.10 SOT.<br>Uses SOT10_1 at the output pin. |
| | 11 | Uses SOT10_2 at the input pin of MFS ch.10 SOT.<br>Uses SOT10_2 at the output pin. |

### [bit17:16] SIN10S: SIN10 Input Select bits

Selects input for SIN10.

| bit17:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN10_0 at the input pin of MFS ch.10 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN10_1 at the input pin of MFS ch.10 SIN. |
| | 11 | Uses SIN10_2 at the input pin of MFS ch.10 SIN. |

### [bit15:14] SCK9B: SCK9 Input/Output Select bits

Selects input/output for SCK9.

| bit15:14 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK9_0 at the input pin of MFS ch.9 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK9_0 at the input pin of MFS ch.9 SCK.<br>Uses SCK9_0 at the output pin. |
| | 10 | Uses SCK9_1 at the input pin of MFS ch.9 SCK.<br>Uses SCK9_1 at the output pin. |
| | 11 | Uses SCK9_2 at the input pin of MFS ch.9 SCK.<br>Uses SCK9_2 at the output pin. |

### [bit13:12] SOT9B: SOT9 Input/Output Select bits

Selects input/output for SOT9.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT9_0 at the input pin of MFS ch.9 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT9_0 at the input pin of MFS ch.9 SOT.<br>Uses SOT9_0 at the output pin. |
| | 10 | Uses SOT9_1 at the input pin of MFS ch.9 SOT.<br>Uses SOT9_1 at the output pin. |
| | 11 | Uses SOT9_2 at the input pin of MFS ch.9 SOT.<br>Uses SOT9_2 at the output pin. |

### [bit11:10] SIN9S: SIN9 Input Select bits

Selects input for SIN9.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN9_0 at the input pin of MFS ch.9 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN9_1 at the input pin of MFS ch.9 SIN. |
| | 11 | Uses SIN9_2 at the input pin of MFS ch.9 SIN. |

### [bit9:8] SCK8B: SCK8 Input/Output Select bits

Selects input/output for SCK8.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK8_0 at the input pin of MFS ch.8 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK8_0 at the input pin of MFS ch.8 SCK.<br>Uses SCK8_0 at the output pin. |
| | 10 | Uses SCK8_1 at the input pin of MFS ch.8 SCK.<br>Uses SCK8_1 at the output pin. |
| | 11 | Uses SCK8_2 at the input pin of MFS ch.8 SCK.<br>Uses SCK8_2 at the output pin. |

### [bit7:6] SOT8B: SOT8 Input/Output Select bits

Selects input/output for SOT8.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT8_0 at the input pin of MFS ch.8 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT8_0 at the input pin of MFS ch.8 SOT.<br>Uses SOT8_0 at the output pin. |
| | 10 | Uses SOT8_1 at the input pin of MFS ch.8 SOT.<br>Uses SOT8_1 at the output pin. |
| | 11 | Uses SOT8_2 at the input pin of MFS ch.8 SOT.<br>Uses SOT8_2 at the output pin. |

### [bit5:4] SIN8S: SIN8 Input Select bits

Selects input for SIN4.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN8_0 at the input pin of MFS ch.8 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN8_1 at the input pin of MFS ch.8 SIN. |
| | 11 | Uses SIN8_2 at the input pin of MFS ch.8 SIN. |

### [bit3:0] Reserved: Reserved bits

"0b0000" is read from these bits.

When writing these bits, set them to "0b0000".

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.23 Extended Pin Function Setting Register 17 (EPFR17)

The EPFR17 register assigns functions of multi-function serial channel 12, channel 13, channel 14, and channel 15.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | SCK15B | | SOT15B | |
| Attribute | - | | | | R/W | | R/W | |
| Initial value | - | | | | 00 | | 00 | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | SIN15S | | SCK14B | | SOT14B | | SIN14S | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | SCK13B | | SOT13B | | SIN13S | | SCK12B | |
| Attribute | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00 | | 00 | | 00 | | 00 | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | SOT12B | | SIN12S | | Reserved | | | |
| Attribute | R/W | | R/W | | - | | | |
| Initial value | 00 | | 00 | | - | | | |

### Register Function

**[bit31:28] Reserved: Reserved bits**

"0b0000" is read from these bits.
When writing these bits, set them to "0b0000".

**[bit27:26] SCK15B: SCK15 Input/Output Select bits**

Selects input/output for SCK15.

| bit27:26 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK15_0 at the input pin of MFS ch.15 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK15_0 at the input pin of MFS ch.15 SCK.<br>Uses SCK15_0 at the output pin. |
| | 10 | Uses SCK15_1 at the input pin of MFS ch.15 SCK.<br>Uses SCK15_1 at the output pin. |
| | 11 | Uses SCK15_2 at the input pin of MFS ch.15 SCK.<br>Uses SCK15_2 at the output pin. |

### [bit25:24] SOT15B: SOT15 Input/Output Select bits

Selects input/output for SOT15.

| bit25:24 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT15_0 at the input pin of MFS ch.15 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT15_0 at the input pin of MFS ch.15 SOT.<br>Uses SOT15_0 at the output pin. |
| | 10 | Uses SOT15_1 at the input pin of MFS ch.15 SOT.<br>Uses SOT15_1 at the output pin. |
| | 11 | Uses SOT15_2 at the input pin of MFS ch.15 SOT.<br>Uses SOT15_2 at the output pin. |

### [bit23:22] SIN15S: SIN15 Input Select bits

Selects input for SIN15.

| bit23:22 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN15_0 at the input pin of MFS ch.15 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN15_1 at the input pin of MFS ch.15 SIN. |
| | 11 | Uses SIN15_2 at the input pin of MFS ch.15 SIN. |

### [bit21:20] SCK14B: SCK14 Input/Output Select bits

Selects input/output for SCK14.

| bit21:20 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK14_0 at the input pin of MFS ch.14 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK14_0 at the input pin of MFS ch.14 SCK.<br>Uses SCK14_0 at the output pin. |
| | 10 | Uses SCK14_1 at the input pin of MFS ch.14 SCK.<br>Uses SCK14_1 at the output pin. |
| | 11 | Uses SCK14_2 at the input pin of MFS ch.14 SCK.<br>Uses SCK14_2 at the output pin. |

### [bit19:18] SOT14B: SOT14 Input/Output Select bits

Selects input/output for SOT14.

| bit19:18 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT14_0 at the input pin of MFS ch.14 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT14_0 at the input pin of MFS ch.14 SOT.<br>Uses SOT14_0 at the output pin. |
| | 10 | Uses SOT14_1 at the input pin of MFS ch.14 SOT.<br>Uses SOT14_1 at the output pin. |
| | 11 | Uses SOT14_2 at the input pin of MFS ch.14 SOT.<br>Uses SOT14_2 at the output pin. |

### [bit17:16] SIN14S: SIN14 Input Select bits

Selects input for SIN14.

| bit17:16 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN14_0 at the input pin of MFS ch.14 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN14_1 at the input pin of MFS ch.14 SIN. |
| | 11 | Uses SIN14_2 at the input pin of MFS ch.14 SIN. |

### [bit15:14] SCK13B: SCK13 Input/Output Select bits

Selects input/output for SCK13.

| bit15:14 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK13_0 at the input pin of MFS ch.13 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK13_0 at the input pin of MFS ch.13 SCK.<br>Uses SCK13_0 at the output pin. |
| | 10 | Uses SCK13_1 at the input pin of MFS ch.13 SCK.<br>Uses SCK13_1 at the output pin. |
| | 11 | Uses SCK13_2 at the input pin of MFS ch.13 SCK.<br>Uses SCK13_2 at the output pin. |

### [bit13:12] SOT13B: SOT13 Input/Output Select bits

Selects input/output for SOT13.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT13_0 at the input pin of MFS ch.13 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT13_0 at the input pin of MFS ch.13 SOT.<br>Uses SOT13_0 at the output pin. |
| | 10 | Uses SOT13_1 at the input pin of MFS ch.13 SOT.<br>Uses SOT13_1 at the output pin. |
| | 11 | Uses SOT13_2 at the input pin of MFS ch.13 SOT.<br>Uses SOT13_2 at the output pin. |

### [bit11:10] SIN13S: SIN13 Input Select bits

Selects input for SIN13.

| bit11:10 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN13_0 at the input pin of MFS ch.13 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN13_1 at the input pin of MFS ch.13 SIN. |
| | 11 | Uses SIN13_2 at the input pin of MFS ch.13 SIN. |

### [bit9:8] SCK12B: SCK12 Input/Output Select bits

Selects input/output for SCK12.

| bit9:8 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCK12_0 at the input pin of MFS ch.12 SCK.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCK12_0 at the input pin of MFS ch.12 SCK.<br>Uses SCK12_0 at the output pin. |
| | 10 | Uses SCK12_1 at the input pin of MFS ch.12 SCK.<br>Uses SCK12_1 at the output pin. |
| | 11 | Uses SCK12_2 at the input pin of MFS ch.12 SCK.<br>Uses SCK12_2 at the output pin. |

### [bit7:6] SOT12B: SOT12 Input/Output Select bits

Selects input/output for SOT12.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SOT12_0 at the input pin of MFS ch.12 SOT.<br>Does not produce output. [Initial value] |
| | 01 | Uses SOT12_0 at the input pin of MFS ch.12 SOT.<br>Uses SOT12_0 at the output pin. |
| | 10 | Uses SOT12_1 at the input pin of MFS ch.12 SOT.<br>Uses SOT12_1 at the output pin. |
| | 11 | Uses SOT12_2 at the input pin of MFS ch.12 SOT.<br>Uses SOT12_2 at the output pin. |

### [bit5:4] SIN12S: SIN12 Input Select bits

Selects input for SIN12.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SIN12_0 at the input pin of MFS ch.12 SIN. [Initial value] |
| | 01 | Same as Writing 00. |
| | 10 | Uses SIN12_1 at the input pin of MFS ch.12 SIN. |
| | | Uses SIN12_2 at the input pin of MFS ch.12 SIN. |

### [bit3:0] Reserved: Reserved bits

"0b0000" is read from these bits.

When writing these bits, set them to "0b0000".

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.24 Extended Pin Function Setting Register 18 (EPFR18)

The EPFR18 register assigns functions of HDMI-CEC pins.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | CECR1B | | CECR0B | |
| Attribute | - | | | | R/W | | R/W | |
| Initial value | - | | | | 00 | | 00 | |

### Register Function

**[bit31:4] Reserved: Reserved bits**

"0" is read from these bits.
When writing these bits, set them to "0".

**[bit3:2] CECR1B : CEC1 input/output selection bits**

It selects I/O for I/O pin CEC1 of HDMI-CEC/Remote Control Reception ch.1.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | HDMI-CEC/remote control reception ch.1 is not input/output. [Initial value] |
| | 01 | CEC1_0 is used, HDMI-CEC/remote control reception ch.1 is input/output. |
| | 10 | CEC1_1 is used, HDMI-CEC/remote control reception ch.1 is input/output. |
| | 11 | Setting is prohibited. |

**[bit1:0] CECR0B: CEC0 input/output selection bits**

It selects I/O for I/O pin CEC0 of HDMI-CEC/Remote Control Reception ch.0.

| bit1:0 | Description |
|---|---|
| Reading | Reads out the register value. |
| Writing 00 | HDMI-CEC/remote control reception ch.0 is not input/output. [Initial value] |
| Writing 01 | CEC0_0 is used, HDMI-CEC/remote control reception ch.0 is input/output. |
| Writing 10 | CEC0_1 is used, HDMI-CEC/remote control reception ch.0 is input/output. |
| Writing 11 | Setting is prohibited. |

***Notes:***

- *This register is not initialized by deep standby transition reset.*

## 4.25 Extended Pin Function Setting Register 21 (EPFR21)

EPFR21 register sets the function assignment to QPRC.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | QZIN0S[2] | QBIN0S[2] | QAIN0S[2] |
| Attribute | - | | | | | R/W | R/W | R/W |
| Initial value | - | | | | | 0 | 0 | 0 |

### Register Function

**[bit31:3] Reserved: Reserved bits**

"0" is read out from these bits.
When writing these bits, set them to "0".

**[bit2] QZIN0S[2]: QPRC-ch.0 ZIN Input Pin bits**

Selects input for QPRC-ch.0 as ZIN. If ZIN0_3 is used, set this bit together with QZIN0S bit in Extended Pin Function Setting Register 9 (EPFR09).

| bit[2],EPFR09[5:4] | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | ZIN0_0 is used as ZIN, the input pin of QDU ch.0. [Initial value] |
| | 001 | ZIN0_0 is used as ZIN, the input pin of QDU ch.0. |
| | 010 | ZIN0_1 is used as ZIN, the input pin of QDU ch.0. |
| | 011 | ZIN0_2 is used as ZIN, the input pin of QDU ch.0. |
| | 100 | ZIN0_3 is used as ZIN, the input pin of QDU ch.0. |
| | Others | Setting is prohibited. |

### [bit1] QBIN0S[2]: QPRC-ch.0 BIN Input Pin bits

Selects input for QPRC-ch.0 as BIN. If BIN0_3 is used, set this bit together with QBIN0S bit in Extended Pin Function Setting Register 9 (EPFR09).

| Bit[1],EPFR09[3:2] | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | BIN0_0 is used as BIN, the input pin of QDU ch.0. [Initial value] |
| | 001 | BIN0_0 is used as BIN, the input pin of QDU ch.0. |
| | 010 | BIN0_1 is used as BIN, the input pin of QDU ch.0. |
| | 011 | BIN0_2 is used as BIN, the input pin of QDU ch.0. |
| | 100 | BIN0_3 is used as BIN, the input pin of QDU ch.0. |
| | Others | Setting is prohibited. |

### [bit0] QAIN0S[2]: QPRC-ch.0 AIN Input Pin bits

Selects input for QPRC-ch.0 as AIN. If AIN0_3 is used, set this bit together with QAIN0S bit in Extended Pin Function Setting Register 9 (EPFR09)

| Bit[0],EPFR09[1:0] | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 000 | AIN0_0 is used as AIN, the input pin of QDU ch.0. [Initial value] |
| | 001 | AIN0_0 is used as AIN, the input pin of QDU ch.0. |
| | 010 | AIN0_1 is used as AIN, the input pin of QDU ch.0. |
| | 011 | AIN0_2 is used as AIN, the input pin of QDU ch.0. |
| | 100 | AIN0_3 is used as AIN, the input pin of QDU ch.0. |
| | Others | Setting is prohibited. |

***Note:***

- *This register is not initialized by deep standby transition reset.*

## 4.26 Extended Pin Function Setting Register 22 (EPFR22)

The EPFR22 register assigns functions of multi-function serial channel 0, channel 1, channel 2, channel 3.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | SCS31E | | SCS30B | | Reserved | | | |
| R/W | R/W | | R/W | | R/W | | | |
| 00 | 00 | | 00 | | 00 | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | SCS11E | | SCS10B | | Reserved | | | |
| Attribute | R/W | | R/W | | - | | | |
| Initial value | 00 | | 00 | | - | | | |

### Register Function

**[bit31:16] Reserved: Reserved bits**

"0" is read from these bits.
When writing these bits, set them to "0".

**[bit15:14] SCS31E: SCS31 Output Select bits.**

Selects output for SCS31.

| bit15:14 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for MFS ch.3 SCS31. [Initial value] |
| | 01 | Uses SCS31_0 at the output pin of MFS ch 3 SCS31. |
| | 10 | Uses SCS31_1 at the output pin of MFS ch 3 SCS31. |
| | 11 | Uses SCS31_2 at the output pin of MFS ch 3 SCS31. |

**[bit13:12] SCS30B: SCS30 Input/Output Select bits.**
Selects input/output for SCS30.

| bit13:12 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCS30_0 at the input pin of MFS ch.3 SCS30.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCS30_0 at the input pin of MFS ch.3 SCS30.<br>Uses SCS30_0 at the output pin. |
| | 10 | Uses SCS30_1 at the input pin of MFS ch.3 SCS30.<br>Uses SCS30_1 at the output pin. |
| | 11 | Uses SCS30_2 at the input pin of MFS ch.3 SCS30.<br>Uses SCS30_2 at the output pin. |

**[bit11:8] Reserved: Reserved bits**
"0" is read from these bits.
When writing these bits, set them to "0".

**[bit7:6] SCS11E: SCS11 Output Select bits.**
Selects output for SCS11.

| bit7:6 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not produce output for MFS ch.1 SCS11. [Initial value] |
| | 01 | Uses SCS11_0 at the output pinof MFS ch 1 SCS11. |
| | 10 | Uses SCS11_1 at the output pinof MFS ch 1 SCS11. |
| | 11 | Uses SCS11_2 at the output pinof MFS ch 1 SCS11. |

**[bit5:4] SCS10B: SCS10 Input/Output Select bits.**
Selects input/output for SCS10.

| bit5:4 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Uses SCS10_0 at the input pin of MFS ch.1 SCS10.<br>Does not produce output. [Initial value] |
| | 01 | Uses SCS10_0 at the input pin of MFS ch.1 SCS10.<br>Uses SCS10_0 at the output pin. |
| | 10 | Uses SCS10_1 at the input pin of MFS ch.1 SCS10.<br>Uses SCS10_1 at the output pin. |
| | 11 | Uses SCS10_2 at the input pin of MFS ch.1 SCS10.<br>Uses SCS10_2 at the output pin. |

**[bit3:0] Reserved: Reserved bits**
"0" is read from these bits.
When writing these bits, set them to "0".

***Notes:***

- *This register is not initialized by deep standby transition reset.*

## 4.27 Special Port Setting Register (SPSR)

The SPSR register sets a pin as a signal pin of special functions.

### Register Configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |
| Initial value | - | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | MAINXC | | SUBXC | |
| Attribute | - | | | | R/W | | R/W | |
| Initial value | - | | | | 01 | | 01 | |

### Register Function

**[bit31:4] Reserved: Reserved bits**

"0" is read from these bits.
When writing these bits, set them to "0".

**[bit3:2] MAINXC : Main Clock (Oscillation) Pin Setting Register**

This bit sets a pin as a main clock (oscillation) pin.

| bit3:2 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not use two pins of X0 and X1 as main clock (oscillation) pins but as digital input/output pins. |
| | 01 | Uses two pins of X0 and X1 as main clock (oscillation) pins. [Initial value]<br>(An I/O cell will be in a state of input direction, input cut-off, and pull-up disconnection.) |
| | 10 | Setting is prohibited. |
| | 11 | Uses X0 pin as an external clock input pin.<br>Uses X1 pin as a digital input/output. |

**[bit1:0] SUBXC : Sub Clock (Oscillation) Pin Setting Register**

This bit sets a pin as a sub clock (oscillation) pin.

| bit1:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 00 | Does not use two pins of X0A and X1A as sub clock (oscillation) pins but as digital input/output pins. |
| | 01 | Uses two pins of X0A and X1A as sub clock (oscillation) pins. [Initial value]<br>(An I/O cell will be in a state of input direction, input cut-off, and pull-up disconnection.) |
| | 10 | Setting is prohibited. |
| | 11 | Uses X0A pin as an external clock input pin.<br>Uses X1A pin as a digital input/output. |

***Notes:***

- *Only writing "01" to the MAINXC bit does not make a main clock start oscillation.*
  *To start oscillation, enable oscillation by the MOSCE bit of the System Clock Mode Control Register (SCM_CTL), which is described in the chapter "Clock", after writing "01" to the MAINXC bit.*
- *Only writing "01" to the SUBXC bit does not make a sub clock start oscillation.*
  *To start oscillation, enable oscillation by the SOSCE bit of the System Clock Mode Control Register (SCM_CTL), which is described in the chapter "Clock", after writing "01" to the SUBXC bit.*
- *To use external clock, refer to "Using an external clock" of "HANDLING PRECAUTIONS" in Datasheet of each product.*
- *This register is not initialized by deep standby transition reset.*

## 4.28 Port Pseudo Open Drain Setting Register (PZRx)

PZRx register makes I/O port Hi-Z when output is High level and sets pseudo open drain control.

### List of PZR register configuration

| bit 31 – 16 | bit 15 – 0 | Initial value | Attribute |
|---|---|---|---|
| Reserved | PZR0 | 0x0000 | R/W |
| Reserved | PZR1 | 0x0000 | R/W |
| Reserved | PZR2 | 0x0000 | R/W |
| Reserved | PZR3 | 0x0000 | R/W |
| Reserved | PZR4 | 0x0000 | R/W |
| Reserved | PZR5 | 0x0000 | R/W |
| Reserved | PZR6 | 0x0000 | R/W |
| Reserved | PZR7 | 0x0000 | R/W |
| Reserved | PZR8 | 0x0000 | R/W |
| Reserved | PZR9 | 0x0000 | R/W |
| Reserved | PZRA | 0x0000 | R/W |
| Reserved | PZRB | 0x0000 | R/W |
| Reserved | PZRC | 0x0000 | R/W |
| Reserved | PZRD | 0x0000 | R/W |
| Reserved | PZRE | 0x0000 | R/W |
| Reserved | PZRF | 0x0000 | R/W |

### Details of Register Configuration

| bit | 31 – 16 | 15 – 0 |
|---|---|---|
| Field | Reserved | PZRx |

### Register Function

**[bit31:16] Reserved: Reserved bits**

"0x0000" is read from these bits.
When writing these bits, set them to "0x0000".

**[bit15:0] PZRx: Port Pseudo Open Drain Setting Register x**

Sets the pseudo open drain of the pin.

| bit15:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Set the pin to High level when outputting digital High level by GPIO or peripheral macro. |
| | 1 | Set the pin to Hi-Z when outputting digital High level by GPIO or peripheral macro.<br>Disconnect the pull-up resistor regardless of the PCR setting. |

***Notes:***

- *The "x" description of PZRx is wildcard. It shows PZR0, PZR1, PZR2, and so on.*
- *The function of the PZR register is implemented only in some specific pins.*
  *Only pins described as "PZR register control is enabled" in remarks column of "I/O circuit type" of Data Sheet can control open drain.*

- *PZR register does not exist in all pins. However, even the pins that do not have PZR registers can control pseudo open drain by the setting of DDR register if they are used as GPIO.*
  *In such a case, after setting PFR = 0 (GPIO setting) and PDOR = 0,*
  *When setting L output: used as DDR = 1 (output direction).*
  *When setting Hi-Z output: used as DDR = 0 (input direction).*

- *However, in open drain by the GPIO setting, you cannot apply voltage that exceeds VCC at Hi-Z.*

- *This register is not initialized by deep standby transition reset.*

# 5. Usage Precautions

This section describes precautions for using the I/O port.

■ ON/OFF of the Pull-up Resistance When SPL=1

SPL is a signal for turning a pin into Hi-Z state during standby mode.

- When SPL=0 Normal operations
- When SPL=1 Pin Hi-Z, input cut-off, pull-up disconnection
  However, the SPL bit cannot be used for setting external interrupts, NMIX, SWD pins.

For details of the SPL bit, see Chapter "Low Power Consumption Mode".

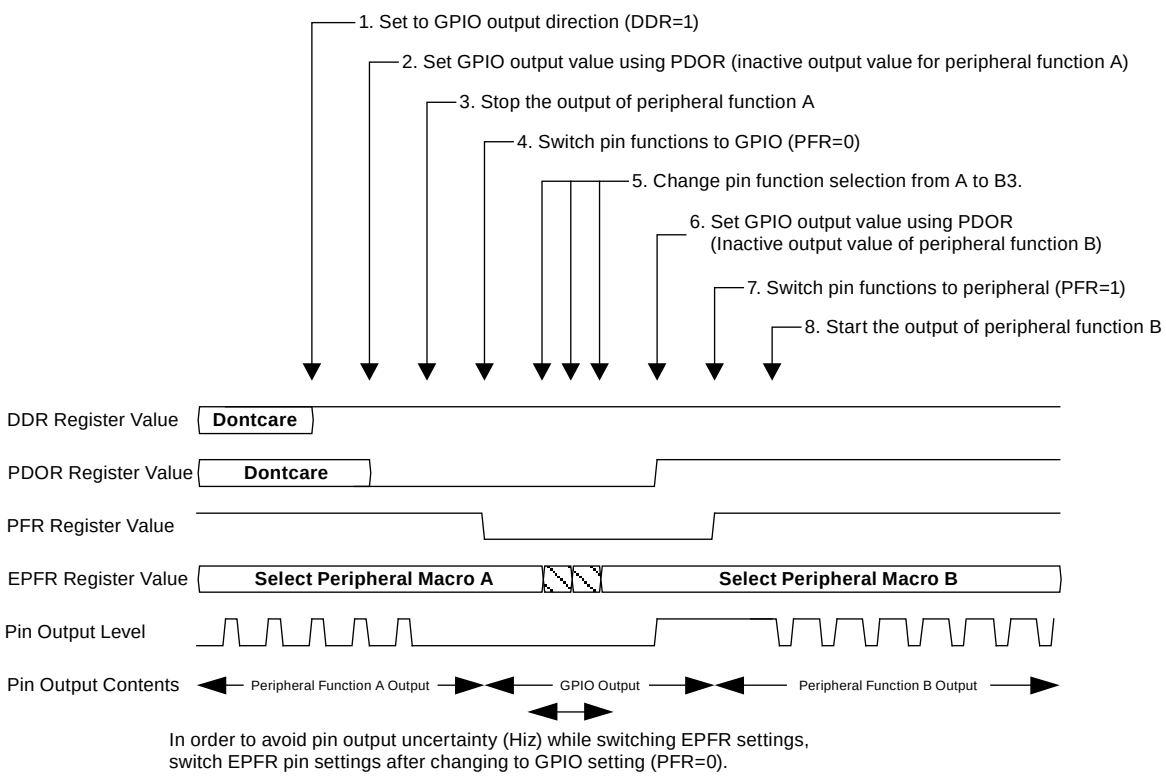
■ DTTIX Input

DTTI input is an input signal for switching the dual-purpose motor control PWM output (RTO) setting output pin to its other GPIO pin setting to address a motor stop demand in an emergency.
To use this function, enable switching by EPFR.

■ Procedures for Switching Pin Functions

When switching the outputs for peripheral functions using the EPFR register, to prevent pin uncertain output (Hi-Z), switch settings according to the procedures similar to the switching example shown in the following Figure 5-1.

**Figure 5-1 Procedures for Switching Pin Functions**



■ Reserved bit

This bit is read out as "0" except for that of ADE register. When writing, always write "0". The ADE reserved bit is read out as "1". When writing, always write "1".

■ Multi-function Serial Pin Group

When there are some multi-function serial inputs/outputs, set each input/output to the port of the same group. "The port of the same group" means that relocate function numbers attached to the pin name are the same, just like "xxx_0" or "yyy_1".

Table 5-1 shows an example setting.

**Table 5-1 Multi-function Serial Interface example setting**

| Serial Data Output | Serial Clock Input/Output | Serial Data Input | Effective Port |
|---|---|---|---|
| Pin SOT1_0 (Port 0) | Pin SCK1_0 (Port 0) | Pin SIN1_0 (Port 0) | Port 0 |
| | | Pin SIN1_1 (Port 1) | Setting is prohibited. |
| | Pin SCK1_1 (Port 1) | Pin SIN1_0 (Port 0) | |
| | | Pin SIN1_1 (Port 1) | |
| Pin SOT1_1 (Port 1) | Pin SCK1_0 (Port 0) | Pin SIN1_0 (Port 0) | |
| | | Pin SIN1_1 (Port 1) | |
| | Pin SCK1_1 (Port 1) | Pin SIN1 (Port 0) | |
| | | Pin SIN1_1 (Port 1) | Port 1 |

■ Peripheral Function Output

As output pins for peripheral functions are uniquely determined by EPFR settings, Output for peripheral functions cannot be assigned to separate pins.

(Disabled example) Assign multifunction serial output SOT1_0 and SOT1_1 to the same output.

■ Pin Settings and Operation Mode

For SWD settings, see Chapter "Debug Interface".

For state of each pin during standby mode or reset, see "Data Sheet" of the product used.

■ Product Specifications and Peripheral Function Pin Assignment

Functions which are assigned to pins (GPO, peripheral I/O and special I/O) vary in different products. Please see the pin function table of "Data Sheet" to confirm the pin function of each product. Do not select a function for a pin which is not available in your product by using the EPFR register setting.

■ When PE0 pin is used as GPIO

To use PE0 pin, the following settings are required.

Input: By reading PDIR, the value is read.

Output: Only L output is available because I/O of PE0 pin is Nch open drain pin.
PFR=0 (Used as GPIO.)
DDR=1(Used as output)
PDOR=0 (Output data is "0".)
SPL=0 (GPIO status is retained in STOP mode.)

■ External Interrupt Pin Settings in Standby Mode

When the mode is transferred to the Standby mode under the setting of SPL=1, set PFR=1 and select peripheral functions to enable the external interrupt assignment pin for returning.

If the setting of a pin used for external interrupt is remained PFR=0, unintended operation occurs.

# CHAPTER10-2: Fast GPIO

**This chapter explains the Fast GPIO port.**



CODE: 9AF_FastGPIO-E01.0

# 1. Overview

This section provides an overview of the Fast GPIO.

**1 cycle access**

- GPIO can access by 1 cycle is called Fast GPIO.
- Fast GPIO can read an input level and set an output level from the CPU within one cycle clock of HCLK.
- GPIO is selected in initial state. It is needed to select Fast GPIO by register setting to use Fast GPIO.

**Pin assignment**

All I/O ports have both normal GPIO and Fast GPIO.

**Bit manipulation**

The Fast GPIO support bit manipulation by dedicated circuit.
Even when bit operation, it can be accessed by 1cycle.

**Simultaneous access function**

Maximum 8 ports can be accessed together at the same time by the mirror register which bundles some ports at different group. It is enabling the effective access in each product.

# 2. Configuration

This section explains the configuration, block diagram, and operation of the Fast GPIO.

**Configuration of Fast GPIO**

Figure 2-1 shows the Fast GPIO block diagram.

I/O port can be switched to normal GPIO and Fast GPIO by FPOER.

Setting PCRx, DDRx, ADE is shared with normal GPIO.

**Figure 2-1 Block diagram of Fast GPIO port**

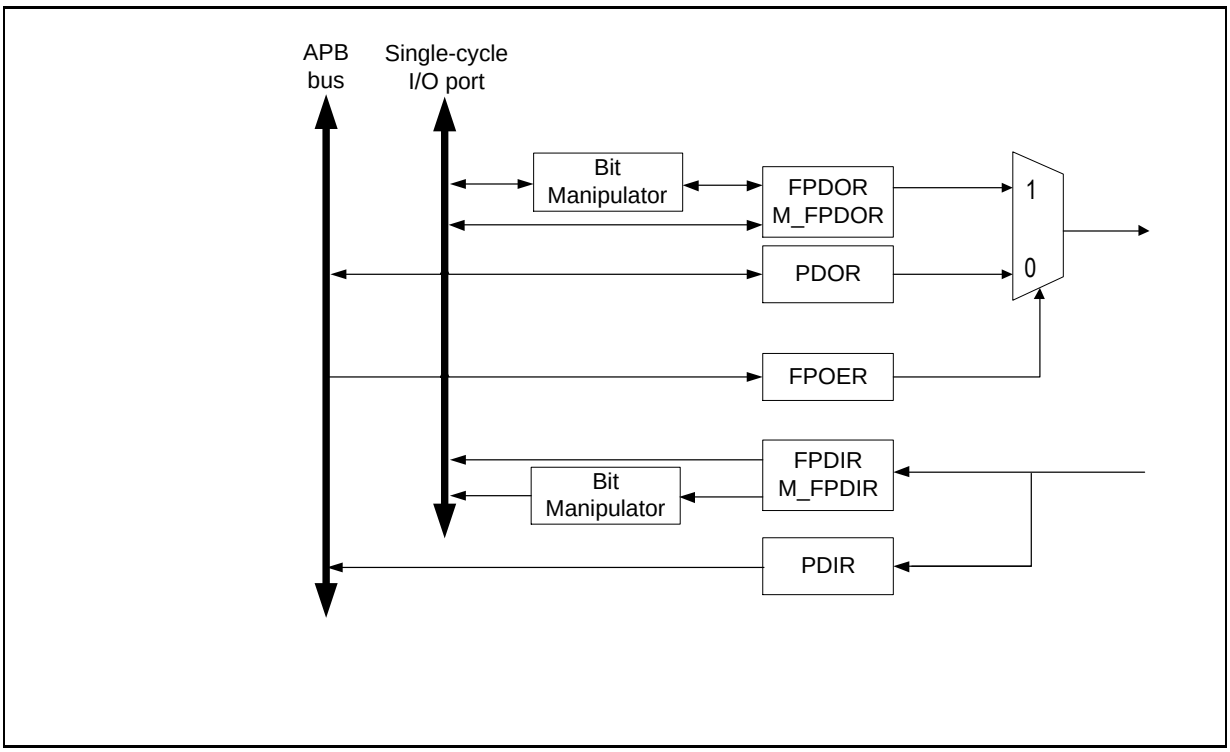


**Table 2-1 Functions of registers**

| Register name | Function |
|---|---|
| FPDIR | A register to read the level status of I/O port. This is connected to Single-cycle I/O port, can be accessed by 1 cycle.<br>- If the Fast GPIO port is used as a digital input pin, it reads input level.<br>- If the Fast GPIO port is used as a digital output pin, it reads output level.<br>- If the Fast GPIO port is used as a special pin, it always reads "0". |
| FPDOR | A register to set output level if the I/O port is used as Fast GPIO output pin. This is connected to Single-cycle I/O port, can be accessed by 1 cycle.<br>- When "0" is set, it outputs Low level.<br>- When "1" is set, it outputs High level.<br>Notes:<br>- If a pin is selected as input state or GPIO function or input/output of peripheral functions, the setting value is invalid.<br>- Setting these registers will affect the value of related bit of M_FPDOR. |
| M_FPDIR | A Mirror Register to read the level status of FPDIR.<br>Some Fast GPIO input value can be read simultaneously.<br>- The register function is the same as FPDIR. |
| M_FPDOR | A Mirror Register to set output level of FPDOR.<br>Some Fast GPIO output value can be set simultaneously.<br>- The register function is the same as FPDOR.<br>- Setting these register will affect the value of related bit of FPDOR. |
| FPOER | A register to select the output between normal GPIO and Fast GPIO.<br>- When “0” select the output of normal GPIO.<br>- When “1” select the output of Fast GPIO.<br>- The setting of this register doesn’t affect the FPDIR/M_FPDIR. |

### Fast GPIO access

- After configured the GPIO as Fast GPIO, each pin can be read by FPDIR and set output level by FPDOR. The usage is same as PDIR/PDOR.
- It is one-to-one correspondence between the bit of FPDIR/PDIR and FPDOR/PDOR.
- For example,

  If P00 is set as input state, access bit0 of FPDIR0 for reading the input value of P00.

  If P15 is set as Fast GPIO and output state, access bit5 of FPDOR1 for setting the output value of P15.

### Fast GPIO mirror access

- Some pins are re-assigned in a special group for flexibility.
- A group of some port assigned to mirror register(M_FPDIR/M_FPDOR) can be accessed simultaneously.
- Table 2-2 shows the configuration of M_FPDIR0/M_FPDOR0. P10-15 and P22-23 can be accessed simultaneously. No need to read FPDIR1 and FPDIR2 separately.
- Table 2-3 shows the configuration of M_FPDIR1/M_FPDOR1. P3A-P3F and P46-47 can be accessed simultaneously. No need to read FPDIR3 and PFPDIR4 separately.

**Table 2-2 M_FPDIR0/M_FPDOR0 configuration (Type 1 product)**

| Mirror port name | Port | Corresponding FPDIR bit | Corresponding FPDOR bit |
|---|---|---|---|
| M_FP00 | P10 | FPDIR1[0] | FPDOR1[0] |
| M_FP01 | P11 | FPDIR1[1] | FPDOR1[1] |
| M_FP02 | P12 | FPDIR1[2] | FPDOR1[2] |
| M_FP03 | P13 | FPDIR1[3] | FPDOR1[3] |
| M_FP04 | P14 | FPDIR1[4] | FPDOR1[4] |
| M_FP05 | P15 | FPDIR1[5] | FPDOR1[5] |
| M_FP06 | P23 | FPDIR2[3] | FPDOR2[3] |
| M_FP07 | P22 | FPDIR2[2] | FPDOR2[2] |

**Table 2-3 M_FPDIR1/M_FPDOR1 configuration(Type 1 product)**

| Mirror port name | Port | Corresponding FPDIR bit | Corresponding FPDOR bit |
|---|---|---|---|
| M_FP10 | P3A | FPDIR3[10] | FPDOR3[10] |
| M_FP11 | P3B | FPDIR3[11] | FPDOR3[11] |
| M_FP12 | P3C | FPDIR3[12] | FPDOR3[12] |
| M_FP13 | P3D | FPDIR3[13] | FPDOR3[13] |
| M_FP14 | P3E | FPDIR3[14] | FPDOR3[14] |
| M_FP15 | P3F | FPDIR3[15] | FPDOR3[15] |
| M_FP16 | P46 | FPDIR4[6] | FPDOR4[6] |
| M_FP17 | P47 | FPDIR4[7] | FPDOR4[7] |

***Notes:***

- *Write the bit of M_FPDORx will affect the value in related bit of FPDORx. For example, after set the bit5 of M_FPDOR1 as "1", the value of bit15 of FPDOR3 is also changed to "1".*

### Bit manipulation

■ Overview

The Fast GPIO is mounted on Single-cycle I/O port. Single-cycle I/O port can access data with one-cycle . However, Single-cycle I/O port cannot support the bit-band operation.

The Fast GPIO supports the bit manipulation by special operation. Each bit of Fast GPIO can be set/clear independently. Similar with the bit-band, there is additional alias area for bit manipulation.

The bit manipulation just support to access the register FPDIR, FPDOR, M_FPDIR and M_FPDOR.

■ How to Use

This explains the address for bit manipulation.

- Alias address

  Alias address is accessed for bit manipulation. The formula is as following.

  Alias address = Base address + Offset address

- Base address

  Base address is decided by each FPDIR, FPDOR, M_FPDIR, M_FPDOR. Please refer to 5.Bit manipulation base address.

- Offset address

  Offset address is accessed to set access bit.

**Table 2-4 Offset address of bit manipulation**

| Access bit | Offset address | Access bit | Offset address |
|---|---|---|---|
| bit8 | 0x0004+1 | bit0 | 0x0004 |
| bit9 | 0x0008+1 | bit1 | 0x0008 |
| bit10 | 0x0010+1 | bit2 | 0x0010 |
| bit11 | 0x0020+1 | bit3 | 0x0020 |
| bit12 | 0x0040+1 | bit4 | 0x0040 |
| bit13 | 0x0080+1 | bit5 | 0x0080 |
| bit14 | 0x0100+1 | bit6 | 0x0100 |
| bit15 | 0x0200+1 | bit7 | 0x0200 |
| Setting is prohibited | Others | Setting is prohibited | Others |

FPDIR read value is reflected to only "Access bit", other bits are "0". For example, when FPDIR8 is "0x07", "FPDIR8+Base address+0x0010" read value is "0x04".

Write to FPDOR is applicable to only "Access bit", other bits are ignored. For example, when FPDOR1 is "0x00" and write "0xFF" to "FPDOR1+Base address+0x0008", FPDOR1 is changed to "0x02".

■ Example of Alias address calculation

- When setting bit3 of FPDOR0

| Access register | Access bit | Access size | Base address | Offset address | Alias address |
|---|---|---|---|---|---|
| FPDOR0 | bit3 | Byte | 0xF801_4000 | 0x0020 | 0xF801_4020 |

- When setting bit13 of FPDOR0

| Access register | Access bit | Access size | Base address | Offset address | Alias address |
|---|---|---|---|---|---|
| FPDOR0 | bit13 | Byte | 0xF801_4000 | 0x0080+1 | 0xF801_4081 |

In the case of half word access, both upper byte and lower byte are masked.

- When setting bit13 of FPDOR0

| Access register | Access bit | Access size | Base address | Offset address | Alias address |
|---|---|---|---|---|---|
| FPDOR0 | bit13, bit5 | Half word | 0xF801_4000 | 0x0080 | 0xF801_4080 |

In the case of word access, it is same with half word access.

# 3. Setup Procedure Example

This section explains an example of procedure for setting up the Fast GPIO.

**Setting up Fast GPIO**

1. When Fast GPIO is used as output, choose the Fast GPIO by setting the FPOER register.
2. Setting the I/O port register as normal GPIO.

Figure 3-1 shows an example of procedure for setting up output.
Figure 3-2 shows an example of procedure for setting up input.

**Figure 3-1 Example of procedure for setting up output of Fast GPIO**

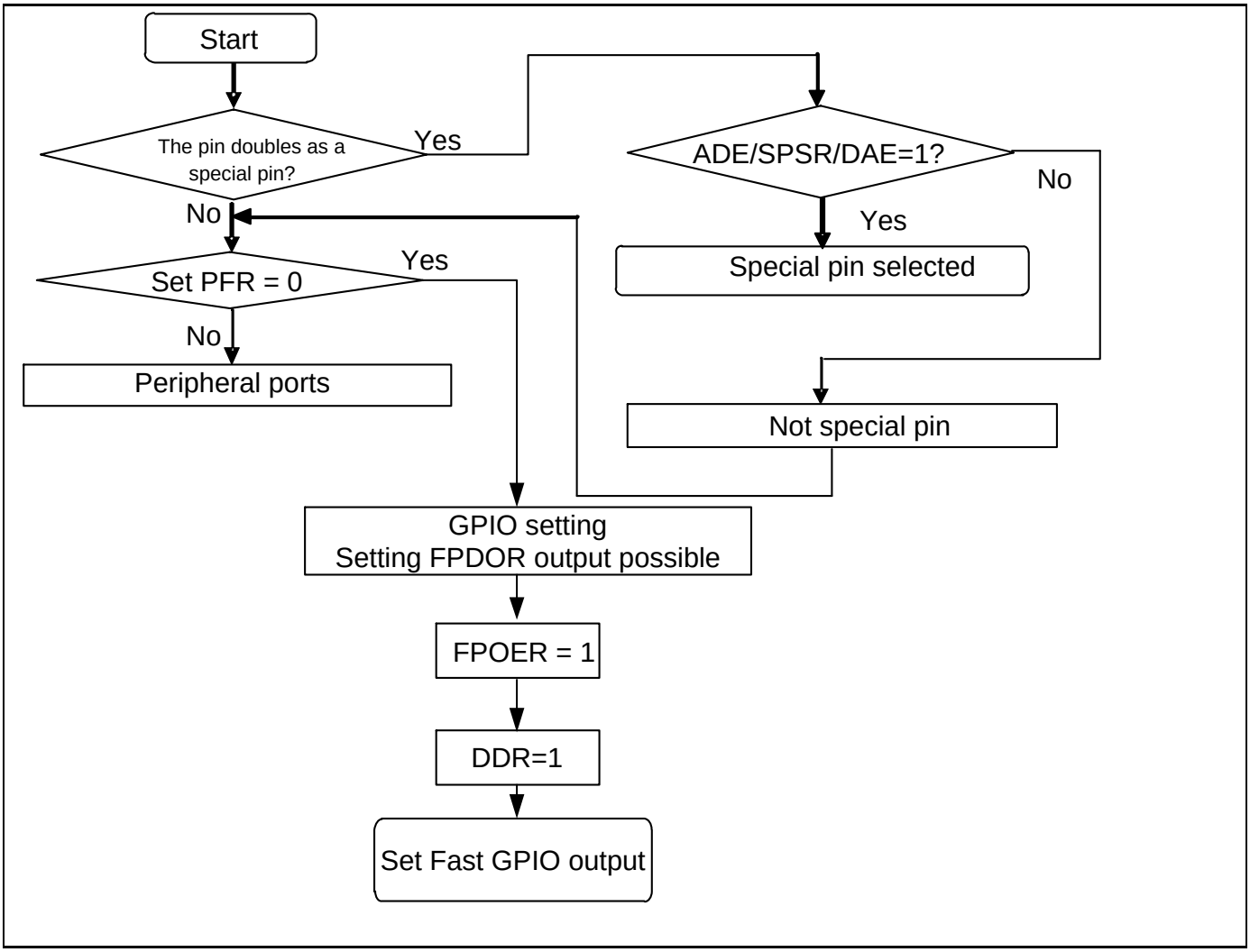
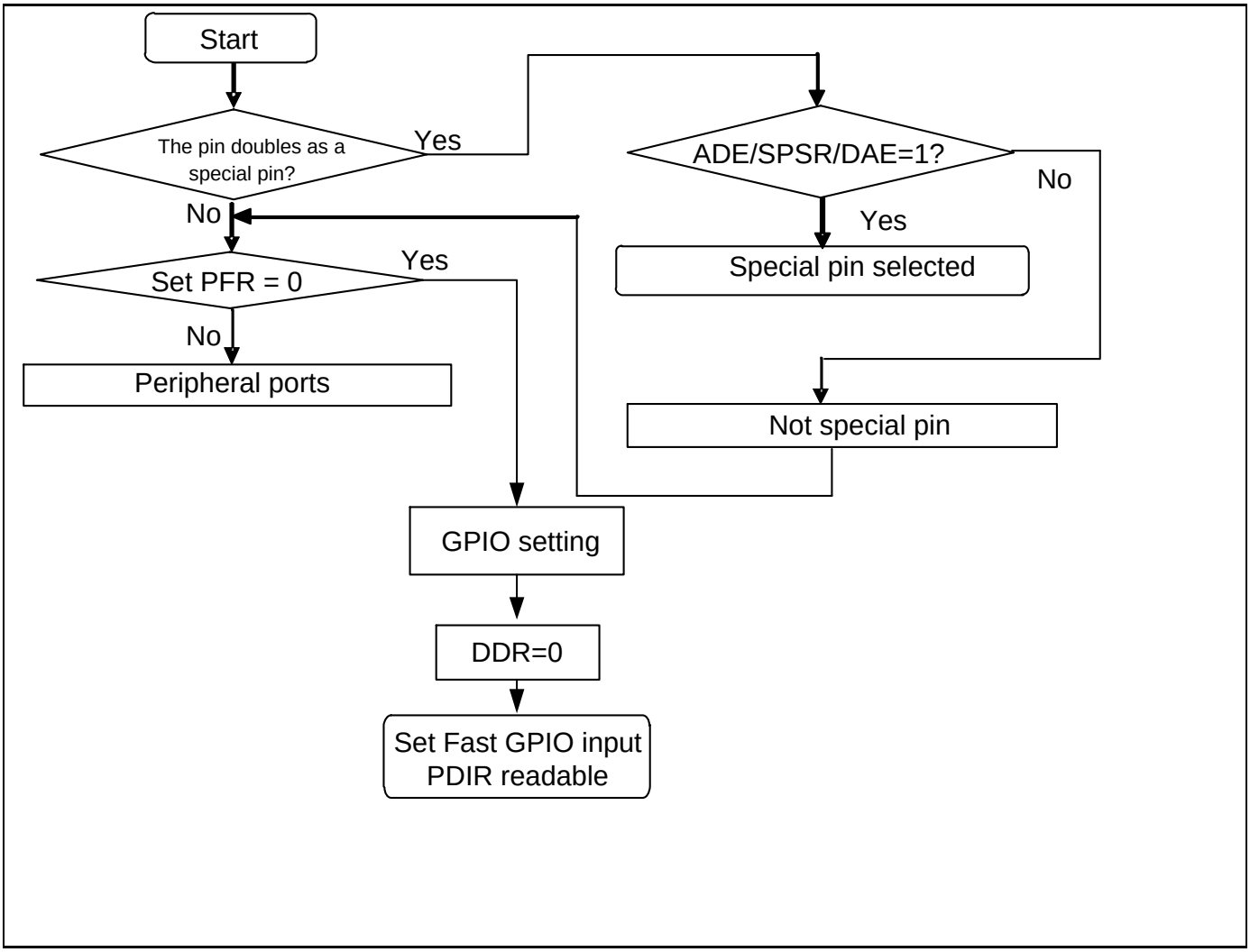


**Figure 3-2 Example of procedure for setting up input of Fast GPIO**



# 4. Registers

This section describes the registers of the Fast GPIO port.

### List of registers of Fast GPIO port

**Table 4-1 List of registers of Fast GPIO**

| Abbreviation | Register name | Reference |
|---|---|---|
| FPDIR0 | Fast GPIO Input Data Register 0 | 4.1 |
| FPDIR1 | Fast GPIO Input Data Register 1 | |
| FPDIR2 | Fast GPIO Input Data Register 2 | |
| FPDIR3 | Fast GPIO Input Data Register 3 | |
| FPDIR4 | Fast GPIO Input Data Register 4 | |
| FPDIR5 | Fast GPIO Input Data Register 5 | |
| FPDIR6 | Fast GPIO Input Data Register 6 | |
| FPDIR7 | Fast GPIO Input Data Register 7 | |
| FPDIR8 | Fast GPIO Input Data Register 8 | |
| FPDIR9 | Fast GPIO Input Data Register 9 | |
| FPDIRA | Fast GPIO Input Data Register A | |
| FPDIRB | Fast GPIO Input Data Register B | |
| FPDIRC | Fast GPIO Input Data Register C | |
| FPDIRD | Fast GPIO Input Data Register D | |
| FPDIRE | Fast GPIO Input Data Register E | |
| FPDIRF | Fast GPIO Input Data Register F | |
| FPDOR0 | Fast GPIO Output Data Register 0 | 4.2 |
| FPDOR1 | Fast GPIO Output Data Register 1 | |
| FPDOR2 | Fast GPIO Output Data Register 2 | |
| FPDOR3 | Fast GPIO Output Data Register 3 | |
| FPDOR4 | Fast GPIO Output Data Register 4 | |
| FPDOR5 | Fast GPIO Output Data Register 5 | |
| FPDOR6 | Fast GPIO Output Data Register 6 | |
| FPDOR7 | Fast GPIO Output Data Register 7 | |
| FPDOR8 | Fast GPIO Output Data Register 8 | |
| FPDOR9 | Fast GPIO Output Data Register 9 | |
| FPDORA | Fast GPIO Output Data Register A | |
| FPDORB | Fast GPIO Output Data Register B | |
| FPDORC | Fast GPIO Output Data Register C | |
| FPDORD | Fast GPIO Output Data Register D | |
| FPDORE | Fast GPIO Output Data Register E | |
| FPDORF | Fast GPIO Output Data Register F | |

| Abbreviation | Register name | Reference |
|---|---|---|
| M_FPDIR0 | Mirror of Fast GPIO Input Data Register 0 | 4.3 |
| M_FPDIR1 | Mirror of Fast GPIO Input Data Register 1 | |
| M_FPDIR2 | Mirror of Fast GPIO Input Data Register 2 | |
| M_FPDIR3 | Mirror of Fast GPIO Input Data Register 3 | |
| M_FPDIR4 | Mirror of Fast GPIO Input Data Register 4 | |
| M_FPDIR5 | Mirror of Fast GPIO Input Data Register 5 | |
| M_FPDIR6 | Mirror of Fast GPIO Input Data Register 6 | |
| M_FPDIR7 | Mirror of Fast GPIO Input Data Register 7 | |
| M_FPDIR8 | Mirror of Fast GPIO Input Data Register 8 | |
| M_FPDIR9 | Mirror of Fast GPIO Input Data Register 9 | |
| M_FPDIRA | Mirror of Fast GPIO Input Data Register A | |
| M_FPDIRB | Mirror of Fast GPIO Input Data Register B | |
| M_FPDIRC | Mirror of Fast GPIO Input Data Register C | |
| M_FPDIRD | Mirror of Fast GPIO Input Data Register D | |
| M_FPDIRE | Mirror of Fast GPIO Input Data Register E | |
| M_FPDIRF | Mirror of Fast GPIO Input Data Register F | |
| M_FPDOR0 | Mirror of Fast GPIO Output Data Register 0 | 4.4 |
| M_FPDOR1 | Mirror of Fast GPIO Output Data Register 1 | |
| M_FPDOR2 | Mirror of Fast GPIO Output Data Register 2 | |
| M_FPDOR3 | Mirror of Fast GPIO Output Data Register 3 | |
| M_FPDOR4 | Mirror of Fast GPIO Output Data Register 4 | |
| M_FPDOR5 | Mirror of Fast GPIO Output Data Register 5 | |
| M_FPDOR6 | Mirror of Fast GPIO Output Data Register 6 | |
| M_FPDOR7 | Mirror of Fast GPIO Output Data Register 7 | |
| M_FPDOR8 | Mirror of Fast GPIO Output Data Register 8 | |
| M_FPDOR9 | Mirror of Fast GPIO Output Data Register 9 | |
| M_FPDORA | Mirror of Fast GPIO Output Data Register A | |
| M_FPDORB | Mirror of Fast GPIO Output Data Register B | |
| M_FPDORC | Mirror of Fast GPIO Output Data Register C | |
| M_FPDORD | Mirror of Fast GPIO Output Data Register D | |
| M_FPDORE | Mirror of Fast GPIO Output Data Register E | |
| M_FPDORF | Mirror of Fast GPIO Output Data Register F | |
| FPOER0 | Fast GPIO Output Enable Register 0 | 4.5 |
| FPOER1 | Fast GPIO Output Enable Register 1 | |
| FPOER2 | Fast GPIO Output Enable Register 2 | |
| FPOER3 | Fast GPIO Output Enable Register 3 | |
| FPOER4 | Fast GPIO Output Enable Register 4 | |
| FPOER5 | Fast GPIO Output Enable Register 5 | |
| FPOER6 | Fast GPIO Output Enable Register 6 | |
| FPOER7 | Fast GPIO Output Enable Register 7 | |
| FPOER8 | Fast GPIO Output Enable Register 8 | |
| FPOER9 | Fast GPIO Output Enable Register 9 | |
| FPOERA | Fast GPIO Output Enable Register A | |
| FPOERB | Fast GPIO Output Enable Register B | |
| FPOERC | Fast GPIO Output Enable Register C | |
| FPOERD | Fast GPIO Output Enable Register D | |
| FPOERE | Fast GPIO Output Enable Register E | |
| FPOERF | Fast GPIO Output Enable Register F | |

## 4.1 Fast GPIO Input Data Register (FPDIRx)

The Fast GPIO Input Data Register (FPDIRx) indicates input data of a pin.

**List of FPDIRx Register configuration**

| bit 31 16 | 15 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | FPDIR0 | 0xXXXX | R | P0F to P00 |
| Reserved | FPDIR1 | 0xXXXX | R | P1F to P10 |
| Reserved | FPDIR2 | 0xXXXX | R | P2F to P20 |
| Reserved | FPDIR3 | 0xXXXX | R | P3F to P30 |
| Reserved | FPDIR4 | 0xXXXX | R | P4F to P40 |
| Reserved | FPDIR5 | 0xXXXX | R | P5F to P50 |
| Reserved | FPDIR6 | 0xXXXX | R | P6F to P60 |
| Reserved | FPDIR7 | 0xXXXX | R | P7F to P70 |
| Reserved | FPDIR8 | 0xXXXX | R | P8F to P80 |
| Reserved | FPDIR9 | 0xXXXX | R | P9F to P90 |
| Reserved | FPDIRA | 0xXXXX | R | PAF to PA0 |
| Reserved | FPDIRB | 0xXXXX | R | PBF to PB0 |
| Reserved | FPDIRC | 0xXXXX | R | PCF to PC0 |
| Reserved | FPDIRD | 0xXXXX | R | PDF to PD0 |
| Reserved | FPDIRE | 0xXXXX | R | PEF to PE0 |
| Reserved | FPDIRF | 0xXXXX | R | PFF to PF0 |

**Detailed register configuration**

| bit | 31 16 | 15 0 |
|---|---|---|
| Field | Reserved | FPDIRx |

**Register functions**

**[bit31:16] Reserved: Reserved bits**

"0x0000" is read out from these bits.

When writing these bits, set them to "0x0000".

**[bit15:0] FPDIRx: Fast GPIO Input Data Register x**

Reads out input data of Fast GPIO.

| bit15:0 | | Description |
|---|---|---|
| Reading | 0 | Regardless of pin function settings (PFR/EPFR/DDR/FPDOR/M_FPDOR), it indicates that a pin is in the status of "L" level input or "L" level output. When a special pin is selected by ADE/SPSR, as input is cut off, 0 is always read out. |
| | 1 | Regardless of pin function settings (PFR/EPFR/DDR/FPDOR/M_FPDOR), it indicates that a pin is in the status of "H" level input or "H" level output. |
| Writing | | Writing does not affect anything. |

***Notes:***

- *The "x" of FPDIRx is a wildcard. FPDIRx indicates FPDIR0, FPDIR1, FPDIR2, etc.*
- *The "x" of Px0 and PxF is a wildcard. Px0 indicates P00, P10, P20, etc. PxF indicates P0F, P1F, P2F, etc.*
- *One register allows reading input data of 16 ports from PxF to Px0.*
- *Each bit in the register indicates the status of each pin individually. There is a one-to-one correspondence between bit assignment and the order of pins. For example, the 15th bit of FPDIR0 indicates P0F, the 14th bit of FPDIR0 indicates P0E, and the 0th bit of FPDIR0 indicates P00.*
- *"0" is always read for a bit value of the pin which is not available in your product.*
- *FPDIRx register is not initialized by deep standby transition reset.*

## 4.2 Fast GPIO Output Data Register x (FPDORx)

The Fast GPIO Output Data Register x (FPDORx) sets output data to a pin.

### List of FPDORx Register configuration

| bit 31 16 | bit 15 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | FPDOR0 | 0x0000 | R/W | P0F to P00 |
| Reserved | FPDOR1 | 0x0000 | R/W | P1F to P10 |
| Reserved | FPDOR2 | 0x0000 | R/W | P2F to P20 |
| Reserved | FPDOR3 | 0x0000 | R/W | P3F to P30 |
| Reserved | FPDOR4 | 0x0000 | R/W | P4F to P40 |
| Reserved | FPDOR5 | 0x0000 | R/W | P5F to P50 |
| Reserved | FPDOR6 | 0x0000 | R/W | P6F to P60 |
| Reserved | FPDOR7 | 0x0000 | R/W | P7F to P70 |
| Reserved | FPDOR8 | 0x0000 | R/W | P8F to P80 |
| Reserved | FPDOR9 | 0x0000 | R/W | P9F to P90 |
| Reserved | FPDORA | 0x0000 | R/W | PAF to PA0 |
| Reserved | FPDORB | 0x0000 | R/W | PBF to PB0 |
| Reserved | FPDORC | 0x0000 | R/W | PCF to PC0 |
| Reserved | FPDORD | 0x0000 | R/W | PDF to PD0 |
| Reserved | FPDORE | 0x0000 | R/W | PEF to PE0 |
| Reserved | FPDORF | 0x0000 | R/W | PFF to PF0 |

### Detailed register configuration

| bit | 31 16 | 15 0 |
|---|---|---|
| Field | Reserved | FPDORx |

### Register functions

**[bit31:16] Reserved: Reserved bits**

"0x0000" is read out from these bits.

When writing these bits, set them to "0x0000".

**[bit15:0] FPDORx: Fast GPIO Output Data Register x**

Sets output data of Fast GPIO.

| bit15:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Outputs "L" level to Fast GPIO.<br>If a pin is selected as I/O input or normal GPIO or peripheral functions input/output, a setting value is invalid. |
| | 1 | Outputs "H" level to Fast GPIO.<br>If a pin is selected as I/O input or normal GPIO or peripheral functions input/output, a setting value is invalid. |

***Notes:***

- *The "x" of FPDORx is a wildcard. FPDORx indicates FPDOR0, FPDOR1, FPDOR2, etc.*
- *The "x" of Px0 and PxF is a wildcard. Px0 indicates P00, P10, P20, etc. PxF indicates P0F, P1F, P2F, etc.*
- *One register allows setting output data of 16 ports from PxF to Px0.*
- *Each bit in the register sets each pin individually. There is a one-to-one correspondence between bit assignment and the order of pins. For example, the 15th bit of FPDOR0 sets P0F, the 14th bit of FPDOR0 sets P0E, and the 0th bit of FPDOR0 sets P00.*
- *For a pin which is not available in your product, writing a value to the bit is invalid, and the read value is undefined.*
- *FPDORx register is not initialized by deep standby transition reset.*

## 4.3 Mirror Fast GPIO Input Data Register (M_FPDIRx)

The Mirror Fast GPIO Input Data Register (M_FPDIRx) indicates input data of a pin.

**List of M_FPDIRx Register configuration**

| bit 31 … 8 | bit 7 … 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | M_FPDIR0 | 0xXX | R | M_FP07 to M_FP00 |
| Reserved | M_FPDIR1 | 0xXX | R | M_FP17 to M_FP10 |
| Reserved | M_FPDIR2 | 0xXX | R | M_FP27 to M_FP20 |
| Reserved | M_FPDIR3 | 0xXX | R | M_FP37 to M_FP30 |
| Reserved | M_FPDIR4 | 0xXX | R | M_FP47 to M_FP40 |
| Reserved | M_FPDIR5 | 0xXX | R | M_FP57 to M_FP50 |
| Reserved | M_FPDIR6 | 0xXX | R | M_FP67 to M_FP60 |
| Reserved | M_FPDIR7 | 0xXX | R | M_FP77 to M_FP70 |
| Reserved | M_FPDIR8 | 0xXX | R | M_FP87 to M_FP80 |
| Reserved | M_FPDIR9 | 0xXX | R | M_FP97 to M_FP90 |
| Reserved | M_FPDIRA | 0xXX | R | M_FPA7 to M_FPA0 |
| Reserved | M_FPDIRB | 0xXX | R | M_FPB7 to M_FPB0 |
| Reserved | M_FPDIRC | 0xXX | R | M_FPC7 to M_FPC0 |
| Reserved | M_FPDIRD | 0xXX | R | M_FPD7 to M_FPD0 |
| Reserved | M_FPDIRE | 0xXX | R | M_FPE7 to M_FPE0 |
| Reserved | M_FPDIRF | 0xXX | R | M_FPF7 to M_FPF0 |

**Detailed register configuration**

| bit | 31 … 8 | 7 … 0 |
|---|---|---|
| Field | Reserved | M_FPDIRx |

**Register functions**

**[bit31:8] Reserved: Reserved bits**

"0x0000" is read out from these bits. When writing these bits, set them to "0x0000".

**[bit7:0] M_FPDIRx: Mirror Fast GPIO Input Data Register x**

Reads out input data of Fast GPIO.

| bit7:0 | | Description |
|---|---|---|
| Reading | 0 | Regardless of pin function settings (PFR/EPFR/DDR/FPDOR/M_FPDOR), it indicates that a pin is in the status of "L" level input or "L" level output. When a special pin is selected by ADE/SPSR, as input is cut off, 0 is always read out. |
| | 1 | Regardless of pin function settings (PFR/EPFR/DDR/FPDOR/M_FPDOR), it indicates that a pin is in the status of "H" level input or "H" level output. |
| Writing | | Writing does not affect anything. |

***Notes:***

- *The "x" of M_FPDIRx is a wildcard. M_FPDIRx indicates M_FPDIR0, M_FPDIR1, etc.*
- *"0" is always read for a bit value of the pin which is not available in your product.*
- *M_FPDIRx register is not initialized by deep standby transition reset.*

## 4.4 Mirror of Fast GPIO Output Data Register (M_FPDORx)

The Mirror of Fast GPIO Output Data Register (M_FPDORx) sets output data to a pin.

### List of M_FPDORx Register configuration

| bit 31 … 8 | 7 … 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | M_FPDOR0 | 0x00 | R/W | M_FP07 to M_FP00 |
| Reserved | M_FPDOR1 | 0x00 | R/W | M_FP17 to M_FP10 |
| Reserved | M_FPDOR2 | 0x00 | R/W | M_FP27 to M_FP20 |
| Reserved | M_FPDOR3 | 0x00 | R/W | M_FP37 to M_FP30 |
| Reserved | M_FPDOR4 | 0x00 | R/W | M_FP47 to M_FP40 |
| Reserved | M_FPDOR5 | 0x00 | R/W | M_FP57 to M_FP50 |
| Reserved | M_FPDOR6 | 0x00 | R/W | M_FP67 to M_FP60 |
| Reserved | M_FPDOR7 | 0x00 | R/W | M_FP77 to M_FP70 |
| Reserved | M_FPDOR8 | 0x00 | R/W | M_FP87 to M_FP80 |
| Reserved | M_FPDOR9 | 0x00 | R/W | M_FP97 to M_FP90 |
| Reserved | M_FPDORA | 0x00 | R/W | M_FPA7 to M_FPA0 |
| Reserved | M_FPDORB | 0x00 | R/W | M_FPB7 to M_FPB0 |
| Reserved | M_FPDORC | 0x00 | R/W | M_FPC7 to M_FPC0 |
| Reserved | M_FPDORD | 0x00 | R/W | M_FPD7 to M_FPD0 |
| Reserved | M_FPDORE | 0x00 | R/W | M_FPE7 to M_FPE0 |
| Reserved | M_FPDORF | 0x00 | R/W | M_FPF7 to M_FPF0 |

### Detailed register configuration

| bit | 31 … 8 | 7 … 0 |
|---|---|---|
| Field | Reserved | M_FPDORx |

### Register functions

#### [bit31:8] Reserved: Reserved bits

"0x0000" is read out from these bits. When writing these bits, set them to "0x0000".

#### [bit7:0] M_FPDORx: Mirror Fast GPIO Output Data Register x

Sets output data of Fast GPIO.

| bit7:0 | | Description |
|---|---|---|
| Reading | | Reads out the register value. |
| Writing | 0 | Outputs "L" level to Fast GPIO.<br>If a pin is selected as I/O input or normal GPIO or peripheral functions input/output, a setting value is invalid. |
| | 1 | Outputs "H" level to Fast GPIO.<br>If a pin is selected as I/O input or normal GPIO or peripheral functions input/output, a setting value is invalid. |

***Notes:***

- *The "x" of M_FPDORx is a wildcard. M_FPDORx indicates M_FPDOR0, M_FPDOR1, etc.*
- *For a pin which is not available in your product, writing a value to the bit is invalid, and the read value is undefined.*
- *M_FPDORx register is not initialized by deep standby transition reset.*

## 4.5 Fast GPIO Output Enable Register (FPOERx)

The Fast GPIO Output Enable Register (FPOERx) selects the output of the normal GPIO/Fast GPIO.

### List of FPOER Register configuration

| bit 31 16 | 15 0 | Initial value | Attribute | Corresponding port |
|---|---|---|---|---|
| Reserved | FPOER0 | 0x0000 | W | P0F to P00 |
| Reserved | FPOER1 | 0x0000 | W | P1F to P10 |
| Reserved | FPOER2 | 0x0000 | W | P2F to P20 |
| Reserved | FPOER3 | 0x0000 | W | P3F to P30 |
| Reserved | FPOER4 | 0x0000 | W | P4F to P40 |
| Reserved | FPOER5 | 0x0000 | W | P5F to P50 |
| Reserved | FPOER6 | 0x0000 | W | P6F to P60 |
| Reserved | FPOER7 | 0x0000 | W | P7F to P70 |
| Reserved | FPOER8 | 0x0000 | W | P8F to P80 |
| Reserved | FPOER9 | 0x0000 | W | P9F to P90 |
| Reserved | FPOERA | 0x0000 | W | PAF to PA0 |
| Reserved | FPOERB | 0x0000 | W | PBF to PB0 |
| Reserved | FPOERC | 0x0000 | W | PCF to PC0 |
| Reserved | FPOERD | 0x0000 | W | PDF to PD0 |
| Reserved | FPOERE | 0x0000 | W | PEF to PE0 |
| Reserved | FPOERF | 0x0000 | W | PFF to PF0 |

### Detailed register configuration

| bit | 31 16 | 15 0 |
|---|---|---|
| Field | Reserved | FPOERx |

### Register functions

**[bit31:16] Reserved: Reserved bits**

"0x0000" is read out from these bits.

When writing these bits, set them to "0x0000".

**[bit15:0] FPOERx: Fast GPIO Output Enable Register x**

Set the I/O port output to normal GPIO or Fast GPIO.

| bit15:0 | | Description |
|---|---|---|
| Reading | | The read value is undefined |
| Writing | 0 | Select the output of PDOR to the pin.<br>If a pin is selected as I/O input or peripheral functions input/output, a setting value is invalid. |
| | 1 | Select the output of FPDOR to the pin.<br>If a pin is selected as I/O input or peripheral functions input/output, a setting value is invalid. |

***Notes:***

- *The FPOERx registers are mounted on APB bus. Please note these registers cannot support the bit-band access because these registers are write only.*
- *The "x" of FPOERx is a wildcard. FPOERx indicates FPOER0, FPOER1, FPOER2, etc.*
- *The "x" of Px0 and PxF is a wildcard. Px0 indicates P00, P10, P20, etc. PxF indicates P0F, P1F, P2F, etc.*
- *For a pin which is not available in your product, writing a value to the bit is invalid, and the read value is undefined.*
- *FPOERx register is not initialized by deep standby transition reset.*
- *No matter what the value of FPOERx, the value of the pin can be read by FPDIRx.*

# 5. Bit Manipulation Base Address

This section describes the base address for bit manipulation to FPDIRx, FPDORx, M_FPDIRx and M_FPDORx.

### List of Base address

Table 5-1 shows the base address for bit manipulation.

**Table 5-1 List of Alias area**

| name | Address | name | Address |
|---|---|---|---|
| FPDIR0 base address | 0xF801_0000 | FPDOR0 base address | 0xF801_4000 |
| FPDIR1 base address | 0xF801_0400 | FPDOR1 base address | 0xF801_4400 |
| FPDIR2 base address | 0xF801_0800 | FPDOR2 base address | 0xF801_4800 |
| FPDIR3 base address | 0xF801_0C00 | FPDOR3 base address | 0xF801_4C00 |
| FPDIR4 base address | 0xF801_1000 | FPDOR4 base address | 0xF801_5000 |
| FPDIR5 base address | 0xF801_1400 | FPDOR5 base address | 0xF801_5400 |
| FPDIR6 base address | 0xF801_1800 | FPDOR6 base address | 0xF801_5800 |
| FPDIR7 base address | 0xF801_1C00 | FPDOR7 base address | 0xF801_5C00 |
| FPDIR8 base address | 0xF801_2000 | FPDOR8 base address | 0xF801_6000 |
| FPDIR9 base address | 0xF801_2400 | FPDOR9 base address | 0xF801_6400 |
| FPDIRA base address | 0xF801_2800 | FPDORA base address | 0xF801_6800 |
| FPDIRB base address | 0xF801_2C00 | FPDORB base address | 0xF801_6C00 |
| FPDIRC base address | 0xF801_3000 | FPDORC base address | 0xF801_7000 |
| FPDIRD base address | 0xF801_3400 | FPDORD base address | 0xF801_7400 |
| FPDIRE base address | 0xF801_3800 | FPDORE base address | 0xF801_7800 |
| FPDIRF base address | 0xF801_3C00 | FPDORF base address | 0xF801_7C00 |
| M_FPDIR0 base address | 0xF801_8000 | M_FPDOR0 base address | 0xF801_C000 |
| M_FPDIR1 base address | 0xF801_8400 | M_FPDOR1 base address | 0xF801_C400 |
| M_FPDIR2 base address | 0xF801_8800 | M_FPDOR2 base address | 0xF801_C800 |
| M_FPDIR3 base address | 0xF801_8C00 | M_FPDOR3 base address | 0xF801_CC00 |
| M_FPDIR4 base address | 0xF801_9000 | M_FPDOR4 base address | 0xF801_D000 |
| M_FPDIR5 base address | 0xF801_9400 | M_FPDOR5 base address | 0xF801_D400 |
| M_FPDIR6 base address | 0xF801_9800 | M_FPDOR6 base address | 0xF801_D800 |
| M_FPDIR7 base address | 0xF801_9C00 | M_FPDOR7 base address | 0xF801_DC00 |
| M_FPDIR8 base address | 0xF801_A000 | M_FPDOR8 base address | 0xF801_E000 |
| M_FPDIR9 base address | 0xF801_A400 | M_FPDOR9 base address | 0xF801_E400 |
| M_FPDIRA base address | 0xF801_A800 | M_FPDORA base address | 0xF801_E800 |
| M_FPDIRB base address | 0xF801_AC00 | M_FPDORB base address | 0xF801_EC00 |
| M_FPDIRC base address | 0xF801_B000 | M_FPDORC base address | 0xF801_F000 |
| M_FPDIRD base address | 0xF801_B400 | M_FPDORD base address | 0xF801_F400 |
| M_FPDIRE base address | 0xF801_B800 | M_FPDORE base address | 0xF801_F800 |
| M_FPDIRF base address | 0xF801_BC00 | M_FPDORF base address | 0xF801_FC00 |

# CHAPTER11: CRC (Cyclic Redundancy Check)

**This chapter explains the CRC functions.**



CODE: FS15-E02.3

# 1. Overview of CRC

The CRC (Cyclic Redundancy Check) is an error detection system. The CRC code is a remainder after an input data string is divided by the pre-defined generator polynomial, assuming the input data string is a high order polynomial. Ordinarily, a data string is suffixed by a CRC code when being sent, and the received data is divided by a generator polynomial as described above. If the received data is dividable, it is judged that the data is correctly received.

### CRC functions

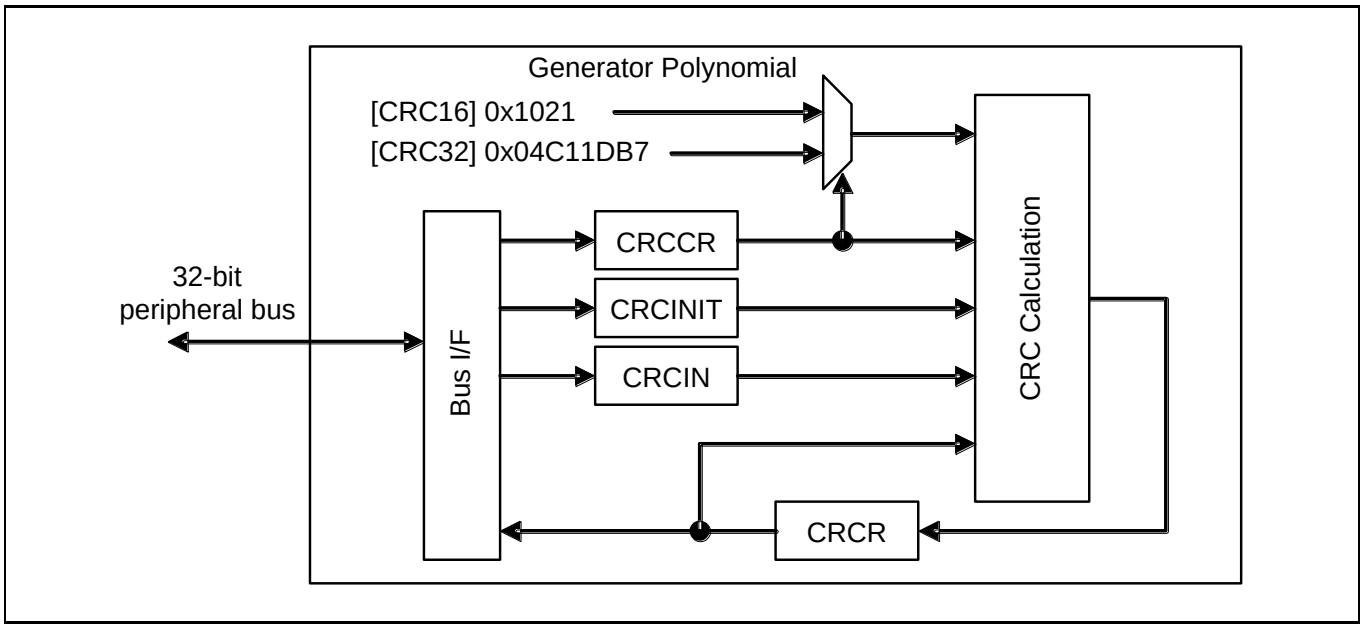
This module enables the calculation in both CCITT CRC16 and IEEE-802.3 CRC32. In this module, the generator polynomial is fixed to the numeric values for those two modes; therefore, the CRC value based on other generator polynomials cannot be calculated.

- CCITT CRC16 generator polynomial: 0x1021
- IEEE-802.3 CRC32 generator polynomial: 0x04C11DB7

### CRC block diagram

Figure 1-1 shows the CRC block diagram.

**Figure 1-1 CRC block diagram**




- CRCCR (CRC Control Register)
  Used to control CRC calculation.
- CRCINIT (CRC Initial Value Register)
  Used to specify the initial values for CRC calculation.
- CRCIN (Input Data Register)
  Used to set input data for CRC calculation.
- CRCR (CRC Register)
  Used to output the CRC calculation result.
- CRC Calculation
  A circuit to perform CRC calculation.

# 2. CRC Operations

This section provides an overview of CRC operations.

### CRC definition

[CCITT CRC16 Standard]

| | | |
|---|---|---|
| Generator polynomial | 0x1021 | (CRCCR:CRC32=0) |
| Initial value | 0xFFFF | |
| Final XOR value | 0x0000 | (CRCCR:FXOR=0) |
| bit order | MSB First | (CRCCR:LSBFST=0) |
| Output bit order | MSB First | (CRCCR:CRCLSF=0) |

(The input-output byte order can be specified arbitrarily.)

[IEEE-802.3 CRC32 Ethernet Standard]

| | | |
|---|---|---|
| Generator polynomial | 0x04C11DB7 | (CRCCR:CRC32=1) |
| Initial value | 0xFFFFFFFF | |
| Final XOR value | 0xFFFFFFFF | (CRCCR:FXOR=1) |
| bit order | LSB First | (CRCCR:LSBFST=1) |
| Output bit order | LSB First | (CRCCR:CRCLSF=1) |

(The input-output byte order can be specified arbitrarily.)

### Reset operations

When resetting, the Initial Value Register (CRCINIT) and CRC Register (CRCR) are set to 0xFFFFFFFF. Other registers are cleared to "0".

### Initialization

Initializing with the initialization bit (INIT) of the CRCCR register loads the value of the Initial Value Register to the CRC Register (CRCR).

### Processing byte and bit orders

The following shows how to process byte and bit orders, using examples.

Input the following one word to the CRC computing unit.

133.82.171.1 = 10000101 01010010 10101011 00000001

If the byte order is set to big endian (CRCCR:LTLEND=0), the sending sequence in bytes is configured as shown below.

10000101 01010010 10101011 00000001
(1st) (2nd) (3rd) (4th)

If the bit order is set to Little endian (CRCCR:LSBFST=1), the sending sequence in bits is configured as shown below.

10100001 01001010 11010101 10000000
(Head) (End)

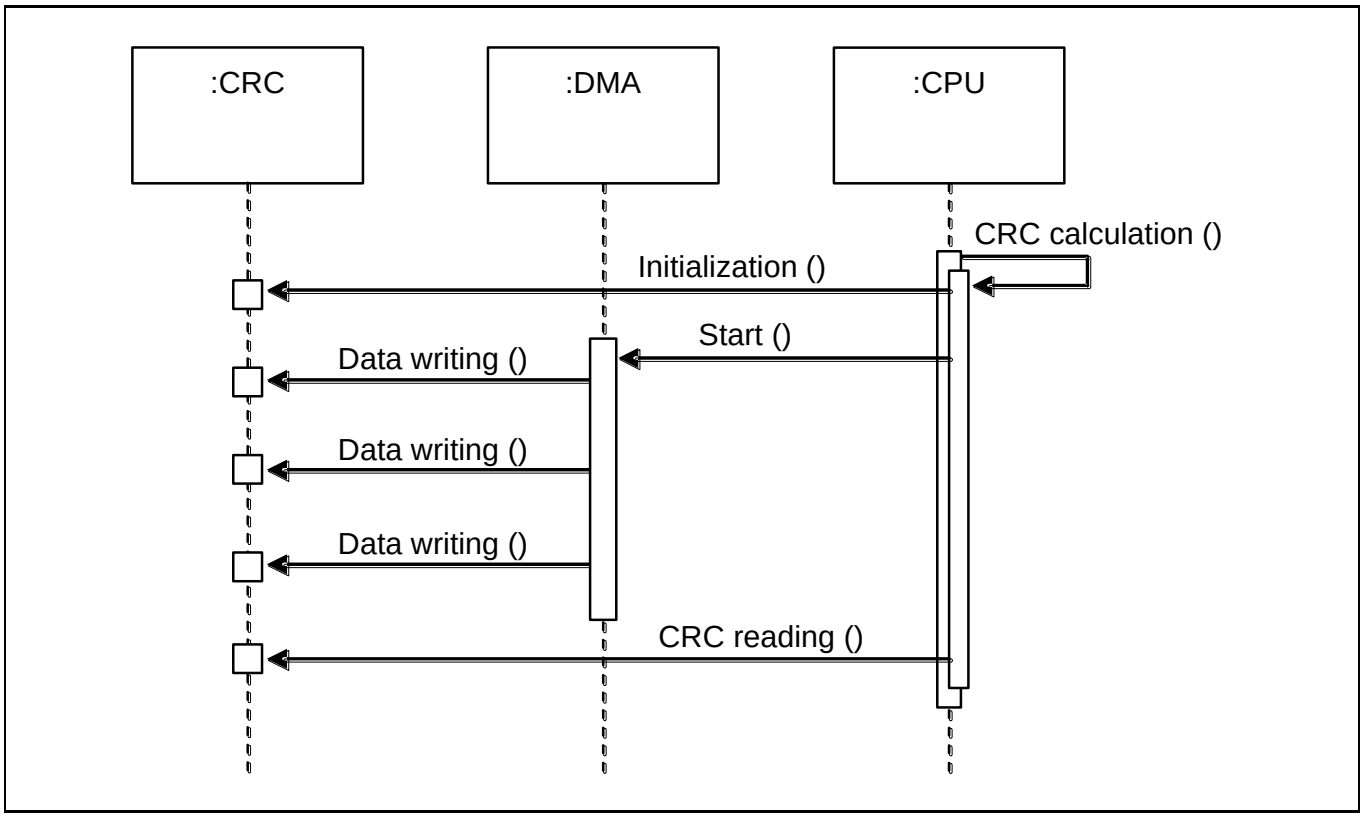
***Note:***

- *At CRCCR:CRCLTE=1, the CRC result is rearranged in bytes with the 32-bit width in both CRC16 and CRC32.*
  *In particular, in CRC16 mode, note that data is output to bit 31 to bit 16.*

## 2.1 CRC calculation sequence

Figure 2-1 shows the CRC calculation sequence. In this section, it is assumed that the Initial Value Register (CRCINIT) setting, CRC16 or CRC32 mode selection (CRCCR:CRC32), and byte- or bit-order setting (CRCCR:LTLEND, CRCCR:LSBFST) have already been configured.
(If the initial value can be set to ALL"H", the Initial Value Register (CRCINIT) setting can be omitted.)

**Figure 2-1 CRC calculation sequence**




- To perform initialization, write "1" to the initial value bit (CRCCR.INIT). The value of the Initial Value Register (CRCINT) is loaded to the CRC Register (CRCR).
- To write input data, write to the Input Data Register (CRCIN). This then starts CRC calculation. If necessary, input data can be written continuously. Furthermore, different bit widths can be used in a sequence to write input data.
- To obtain a CRC code, read the CRC Register (CRCR).

## 2.2 CRC use examples

Figure 2-2 to Figure 2-5 show CRC use examples.

**Use example 1 CRC16, Byte input fixed**

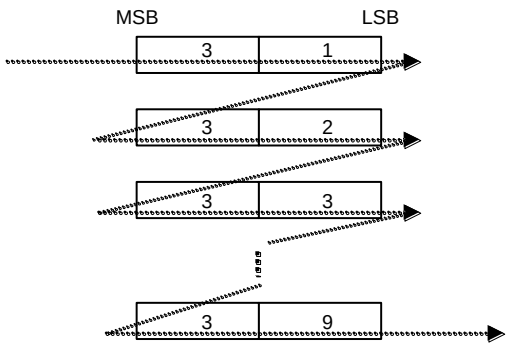
**Figure 2-2 Use example 1 (CRC16, byte input fixed, core byte order : Big endian)**

```
//*************************************************************
//  CRC16 (CRC ITU-T)
//     polynomial:          0x1021
//     initial value:       0xFFFF
//     CRCCR.CRC32:         0 //CRC16
//     CRCCR.LTLEND:        0 //big endian
//     CRCCR.LSBFST:        0 //MSB First
//     CRCCR.CRCLTE:        0 //CRC big endian
//     CRCCR.CRCLSF:        0 //CRC MSB First
//     CRCCR.FXOR:          0 //CRC Final XOR off
//*************************************************************

//
// Example 1-1 (Byte-base writing)
//

// Initialization
B_WRITE (CRCCR, 0x01);

// data write “123456789”
B_WRITE (CRCIN, 0x31);
B_WRITE (CRCIN, 0x32);
B_WRITE (CRCIN, 0x33);
B_WRITE (CRCIN, 0x34);
B_WRITE (CRCIN, 0x35);
B_WRITE (CRCIN, 0x36);
B_WRITE (CRCIN, 0x37);
B_WRITE (CRCIN, 0x38);
B_WRITE (CRCIN, 0x39);

// read result
H_READ (CRCR+2, data);

// check result
assert (data == 0x29B1);

//
// Example 1-2 (CRC check)
//

// Initialization
B_WRITE (CRCCR, 0x01);

// data write "123456789" + CRC
B_WRITE (CRCIN, 0x31);
B_WRITE (CRCIN, 0x32);
B_WRITE (CRCIN, 0x33);
B_WRITE (CRCIN, 0x34);
B_WRITE (CRCIN, 0x35);
B_WRITE (CRCIN, 0x36);
B_WRITE (CRCIN, 0x37);
B_WRITE (CRCIN, 0x38);
B_WRITE (CRCIN, 0x39);
B_WRITE (CRCIN, 0x29); // <-- CRC
B_WRITE (CRCIN, 0xB1); // <-- CRC

// read result
H_READ (CRCR+2, data);

// check result
assert (data == 0x0000);
```

(Assumed as follows.)

| | |
|---|---|
| B_WRITE | -- Byte writing |
| H_WRITE | -- Half-word writing |
| W_WRITE | -- Word writing |
| B_READ | -- Byte reading |
| H_READ | -- Half-word reading |
| W_READ | -- Word reading |
| CRCCR | -- Control Register address |
| CRCINIT | -- Initial Value Register address |
| CRCIN | -- Input Data Register address |
| CRCR | -- Current CRC Register address |

CRC computing unit input sequence image




- The byte and half-word writing positions are arbitrary. In this example, data is written continuously at position +0.
- Table 2-1 shows the CPU, CRC result byte order, CRCR (CRC Register) output position, and read address in CRC16 mode.

**Table 2-1 CPU, CRC result byte order, and CRCR read address**

| Core byte order | CRC result byte order | Output position to CRCR | CRCR H_READ address |
|---|---|---|---|
| Big endian | Big endian | bit 15 to bit 0 | CRCR +2 |
| Big endian | Little endian | bit 31 to bit 16 | CRCR +0 |
| Little endian | Big endian | bit 15 to bit 0 | CRCR +0 |
| Little endian | Little endian | bit 31 to bit 16 | CRCR +2 |

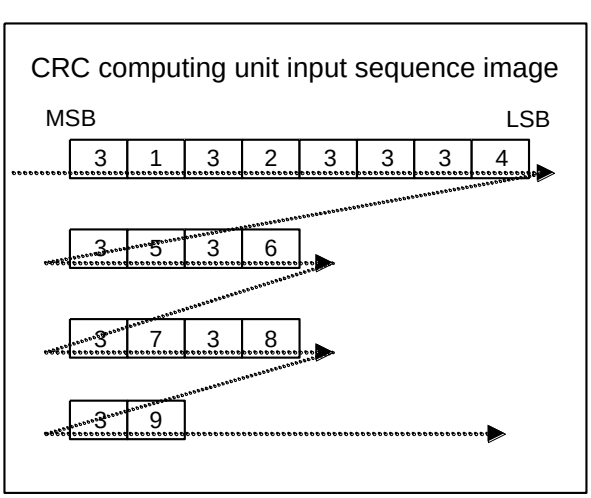
### Use example 2 CRC16, different input bit widths mixed

**Figure 2-3 Use example 2 (CRC16, different input bit widths mixed, core byte order: Big endian)**

```
//*************************************************************
//  CRC16 (CRC ITU-T)
//     polynomial:          0x1021
//     initial value:       0xFFFF
//     CRCCR.CRC32:         0 //CRC16
//     CRCCR.LTLEND:        0 //big endian
//     CRCCR.LSBFST:        0 //MSB First
//     CRCCR.CRCLTE:        0 //CRC big endian
//     CRCCR.CRCLSF:        0 //CRC MSB First
//     CRCCR.FXOR:          0 //CRC Final XOR off
//*************************************************************

//
// Example 2-1 (Writing widths mixed)
//

// Initialization
B_WRITE (CRCCR, 0x01);

// data write "123456789"
W_WRITE (CRCIN, 0x31323334);
H_WRITE (CRCIN, 0x3536);
H_WRITE (CRCIN+2, 0x3738);
B_WRITE (CRCIN+3, 0x39);

// read result
H_READ (CRCR+2, data);

// check result
assert  (data == 0x29B1);

//
// Example 2-2 (CRC check)
//

// Initialization
B_WRITE (CRCCR, 0x01);

// data write "123456789" + CRC
W_WRITE (CRCIN, 0x31313334);
W_WRITE (CRCIN, 0x35363738);
H_WRITE (CRCIN, 0x3929); // <-- CRC(0x29)
B_WRITE (CRCIN, 0xB1); // <-- CRC(0xB1)

// read result
H_READ (CRCR+2, data);

// check result
assert (data == 0x0000);
```

(Assumed as follows.)

| | |
|---|---|
| B_WRITE | -- Byte writing |
| H_WRITE | -- Half-word writing |
| W_WRITE | -- Word writing |
| B_READ | -- Byte reading |
| H_READ | -- Half-word reading |
| W_READ | -- Word reading |
| CRCCR | -- Control Register address |
| CRCINIT | -- Initial Value Register address |
| CRCIN | -- Input Data Register address |
| CRCR | -- Current CRC Register address |




- If the byte or bit order setting is correct and the bit input sequence to the CRC computing unit is the same, the writing width can be specified arbitrarily.
  For example, if a 1-, 2-, or 3-byte fraction is finally obtained in the word-base writing mode, both byte and half-word writings may be enabled.

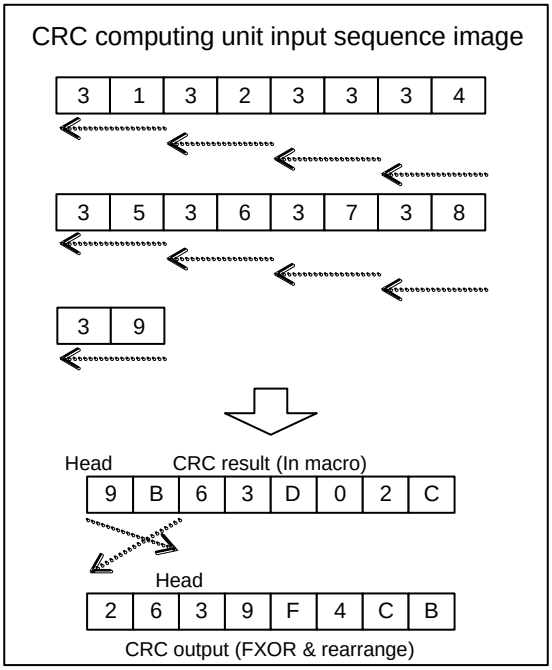
### Use example 3 CRC32, byte order: Big endian

**Figure 2-4 Use example 3 (CRC32, byte order: Big endian)**

```
//*****************************************************************
// CRC32 (IEEE-802.3)
// polynomial:          0x04C11DB7
// initial value:       0xFFFF_FFFF
// CRCCR.CRC32          1 // CRC32
// CRCCR.LTLEND:        0 // big endian
// CRCCR.LSBFST:        1 // LSB First
// CRCCR.CRCLTE:        0 // CRC big endian
// CRCCR.CRCLSF:        1 // CRC LSB First
// CRCCR.FXOR:          1 // CRC Final XOR on
//*****************************************************************

//
// Example 3-1 (CRC32)
//

// Initialization
B_WRITE (CRCCR, 0x6B);

// data write "123456789"
W_WRITE (CRCIN, 0x31323334);
W_WRITE (CRCIN, 0x35363738);
B_WRITE (CRCIN, 0x39);

// read result
W_READ (CRCR, data);

// check CRC result
assert (data == 0x2639F4CB); // <- big endian & LSB First
```

(Assumed as follows.)

| | |
|---|---|
| B_WRITE | -- Byte writing |
| H_WRITE | -- Half-word writing |
| W_WRITE | -- Word writing |
| B_READ | -- Byte reading |
| H_READ | -- Half-word reading |
| W_READ | -- Word reading |
| CRCCR | -- Control Register address |
| CRCINIT | -- Initial Value Register address |
| CRCIN | -- Input Data Register address |
| CRCR | -- Current CRC Register address |




- In CRC32 (IEEE-802.3) mode, the bit order is set to LSB First. This CRC computing unit supports both the big endian and little endian as the byte order. Figure 2-4 shows an example for big endian.

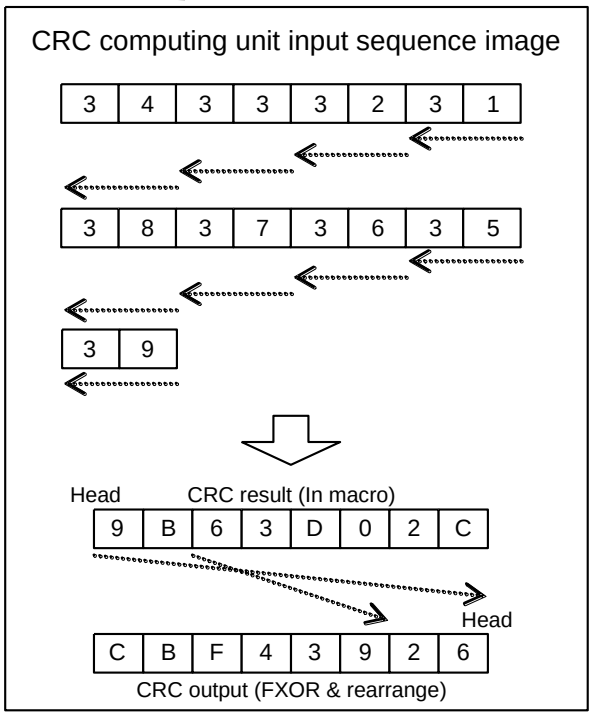
### Use example 4 CRC32, byte order: Little endian

**Figure 2-5 Use example 4 (CRC32, byte order: Little endian)**

```
//*************************************************************
// CRC32 (IEEE-802.3)
// polynomial:          0x04C11DB7
// initial value:       0xFFFF_FFFF
// CRCCR.CRC32          1 // CRC32
// CRCCR.LTLEND:        1 // big endian
// CRCCR.LSBFST:        1 // LSB First
// CRCCR.CRCLTE:        1 // CRC little endian
// CRCCR.CRCLSF:        1 // CRC LSB First
// CRCCR.FXOR:          1 // CRC Final XOR on
//*************************************************************

//
// Example 4-1 (CRC32)
//

// Initialization
B_WRITE (CRCCR, 0x7F);

// data write "123456789"
W_WRITE (CRCIN, 0x34333231);
W_WRITE (CRCIN, 0x38373635);
B_WRITE (CRCIN, 0x39);

// read result
W_READ (CRCR, data);

// check result
assert (data == 0xCBF43926); // <- little endian & LSB First
```

(Assumed as follows.)

| | |
|---|---|
| B_WRITE | -- Byte writing |
| H_WRITE | -- Half-word writing |
| W_WRITE | -- Word writing |
| B_READ | -- Byte reading |
| H_READ | -- Half-word reading |
| W_READ | -- Word reading |
| CRCCR | -- Control Register address |
| CRCINIT | -- Initial Value Register address |
| CRCIN | -- Input Data Register address |
| CRCR | -- Current CRC Register address |




- In CRC32 (IEEE-802.3) mode, the bit order is set to LSB First. This CRC computing unit supports both the big endian and little endian as the byte order. Figure 2-5 shows an example for little endian.

- If bit inversion is not required for the CRC result, perform either one of the following processes to release the bit inversion for the current result.
  - Initialize with 0x3F before calculation.
  - After data was input, set the CRCCR:FXOR bit to "0" (for example, CRCCR=0x3E).

# 3. CRC Registers

This section provides a list of CRC registers.

### CRC registers

**Table 3-1 CRC register list**

| Abbreviation | Register name | Reference |
|---|---|---|
| CRCCR | CRC Control Register | 3.1 |
| CRCINIT | Initial Value Register | 3.2 |
| CRCIN | Input Data Register | 3.3 |
| CRCR | CRC Register | 3.4 |

## 3.1 CRC Control Register (CRCCR)

The CRC Control Register (CRCCR) is used to control CRC calculation.

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | FXOR | CRCLSF | CRCLTE | LSBFST | LTLEND | CRC32 | INIT |
| Attribute | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**[bit7] Reserved: Reserved bit**

The read value is "0".

Be sure to write "0" to this bit.

**[bit6] FXOR: Final XOR control bit**

This bit is used to output the CRC result as the XOR value or XOR.

The OR value is set to ALL"H". This bit is inverted at FXOR=1.

This processing is performed in the latter part of the CRC Register processing. The CRC result is therefore reflected on the read value immediately after this bit was set.

| bit | Description |
|---|---|
| 0 | None |
| 1 | Yes |

**[bit5] CRCLSF: CRC result bit-order setting bit**

This is a bit-order setting bit for CRC result.

This bit is used to rearrange bits within each byte. Set "0" to specify MSB First and set "1" to specify LSB First.

This processing is performed in the latter part of the CRC Register processing. The CRC result is therefore reflected on the read value immediately after this bit was set.

| bit | Description |
|---|---|
| 0 | MSB First |
| 1 | LSB First |

### [bit4] CRCLTE: CRC result byte-order setting bit

This is a byte-order setting bit for CRC result.

This bit is used to rearrange the byte order in each word. Set "0" to specify big endian and set "1" to specify little endian.

This processing is performed in the latter part of the CRC Register processing. The CRC result is therefore reflected on the read value immediately after this bit was set.

If this bit is set to "1" in CRC16 mode, data is output to bit31:16.

| bit | Description |
|---|---|
| 0 | Big endian |
| 1 | Little endian |

### [bit3] LSBFST: bit-order setting bit

This is a bit-order setting bit.

This bit is used to specify the head bit of a byte (8 bits). Set "0" to specify MSB First and set "1" to specify LSB First.

Four types of processing orders can be specified when this bit is combined with the LTLEND bit setting.

| bit | Description |
|---|---|
| 0 | MSB First |
| 1 | LSB First |

### [bit2] LTLEND: Byte-order setting bit

This is a byte-order setting bit.

This bit is used to specify the byte order with the write width. Set "0" to specify big endian and set "1" to specify little endian.

| bit | Description |
|---|---|
| 0 | Big endian |
| 1 | Little endian |

### [bit1] CRC32: CRC mode selection bit

This bit is used to select the CRC16 or CRC32 mode.

| bit | Description |
|---|---|
| 0 | CRC16 |
| 1 | CRC32 |

### [bit0] INIT: Initialization bit

This is an initialization bit. Writing "1" to this bit initializes data. This bit does not have a value, and always returns "0" at reading.
At initialization, the value of the Initial Value Register is loaded to the CRC Register.
Initialization must be performed once at the start of CRC calculation.

| bit | Description | |
|---|---|---|
| | Write | Read |
| 0 | Invalid | Always reads "0". |
| 1 | Initialization | |

## 3.2 Initial Value Register (CRCINIT)

The Initial Value Register (CRCINIT) is used to save the initial values for CRC calculation.

| bit | 31 | 0 |
|---|---|---|
| Field | D[31:0] | |
| Attribute | R/W | |
| Initial value | 0xFFFFFFFF | |

**[bit31:0] D[31:0] : Initial value bits**

These bits are used to save the initial values for CRC calculation.

Write the initial values for CRC calculation to this register.

(0xFFFFFFFF at resetting)

In CRC16 mode, D15 to D0 are used while D31 to D16 are ignored.

## 3.3 Input Data Register (CRCIN)

The Input Data Register (CRCIN) is used to set input data for CRC calculation.

| bit | 31 | 0 |
|---|---|---|
| Field | D[31:0] | |
| Attribute | R/W | |
| Initial value | 0x00000000 | |

**[bit31:0] D[31:0] : Input data bits**

These bits are used to set input data for CRC calculation.

Write input data for CRC calculation to this register. There are three types of bit widths: 8-bit, 16-bit, and 32-bit (byte, half word, word), which can be specified together.

The byte and half-word writing positions are arbitrary. The available address positions are as follows.

Byte writing : +0, +1, +2, +3

Half-word writing : +0, +2

## 3.4 CRC Register (CRCR)

The CRC Register (CRCR) is used to output the CRC calculation result. This register must be initialized before start calculating.

| bit | 31 | 0 |
|---|---|---|
| Field | D[31:0] | |
| Attribute | R | |
| Initial value | 0xFFFFFFFF | |

**[bit31:0] D[31:0] : CRC bits**

These bits are used to read the CRC calculation result. If "1" is written to the initialization bit (CRCCR :INIT), the value of the Initial Value Register (CRCINIT) is loaded to this register.
If input data for CRC calculation is written to the Input Data Register (CRCIN), the CRC calculation result is set to this register after one machine clock cycle has elapsed. When all input data writing has been completed, this register holds the final CRC code.
In CRC16 mode, when the byte order is set to big endian (CRCLTE=0), the result is output to D15 to D0. When the byte order is set to little endian (CRCLTE=1), the result is output to D31 to D16.

# CHAPTER12: Debug Interface

**This chapter explains the function and operation of the debug interface.**



CODE: 9AFDEBUG-E01.0

# 1. Overview

This Family contains a Serial Wire Debug Port (SW-DP).

Connecting an ICE to the SW-DP allows system debugging.

This series also contains a Micro Trace Buffer (MTB) for recoding changes in a program flow.

This section describes the debugging interface.

For details on the SW-DP and system debug, see "Cortex-M0+ Technical Reference Manual".

### Features

Two pins are assigned to the SW-DP.

The initial function of these two pins is serial wire debug.

# 2. Pin Description

This section explains pins.

## 2.1 Pins for Debug Purposes

Two pins (SWCLK and SWDIO) are assigned to the serial wire.

Table 2-1 shows a list of pin functions.

**Table 2-1 SW-DP pin functions in debug mode**

| Pin | Function |
|---|---|
| SWCLK | Serial Wire Clock signal |
| SWDIO | Serial Wire Data Input/Output signal |

## 2.2 Functions Initially Assigned to Pins

The two SW-DP pins are also used as GPIO.
The initial function of the SW-DP pins is debugging.

Note: For details on how to set the debug function, see Chapter "I/O Port"

Table 2-2 shows initial states after resets are cleared and the functions that can be changed by setting PFRs (Port function registers).

Note: For details on the PFRs, see chapter "I/O Port".

**Table 2-2 Functions initially assigned to pins for debugging purposes and change of functions**

| | Pin name | Initially assigned pin function | Change of functions by setting the PFRs |
|---|---|---|---|
| SW-DP pins | SWCLK | SWCLK | GPIO |
| | SWDIO | SWDIO | GPIO |

## 2.3 Internal Pull-up of SW-DP Pins

As specified in the ARM Standard, this Family provides the Debug pins that have internal pull-ups. The user can control pull-up by setting the appropriate registers in the GPIO.

**Table 2-3 Enabled or disabled state of internal pull-up of SW-DP pins**

| Pin name | Pull-up with debug pins enabled* |
|---|---|
| SWCLK | Enabled |
| SWDIO | Enabled |

*: Pull-up is enabled even at a reset.

# CHAPTER13: Micro Trace Buffer Data Watchpoint and Trace

**This chapter explains the functions and operations of the MTB_DWT(Micro Trace Buffer Data Watchpoint and Trace).**



CODE: 9AFEXTINT-E01.0_FW12-E0.14

# 1. Overview

The MTB_DWT is to generate TSTOP and TSTART signals of ARM Core-sight MTB. The MTB_DWT function monitors the processor address and data buses when accessing data phase, configurable watch points can be detected to trigger the appropriate response in the MTB recording.

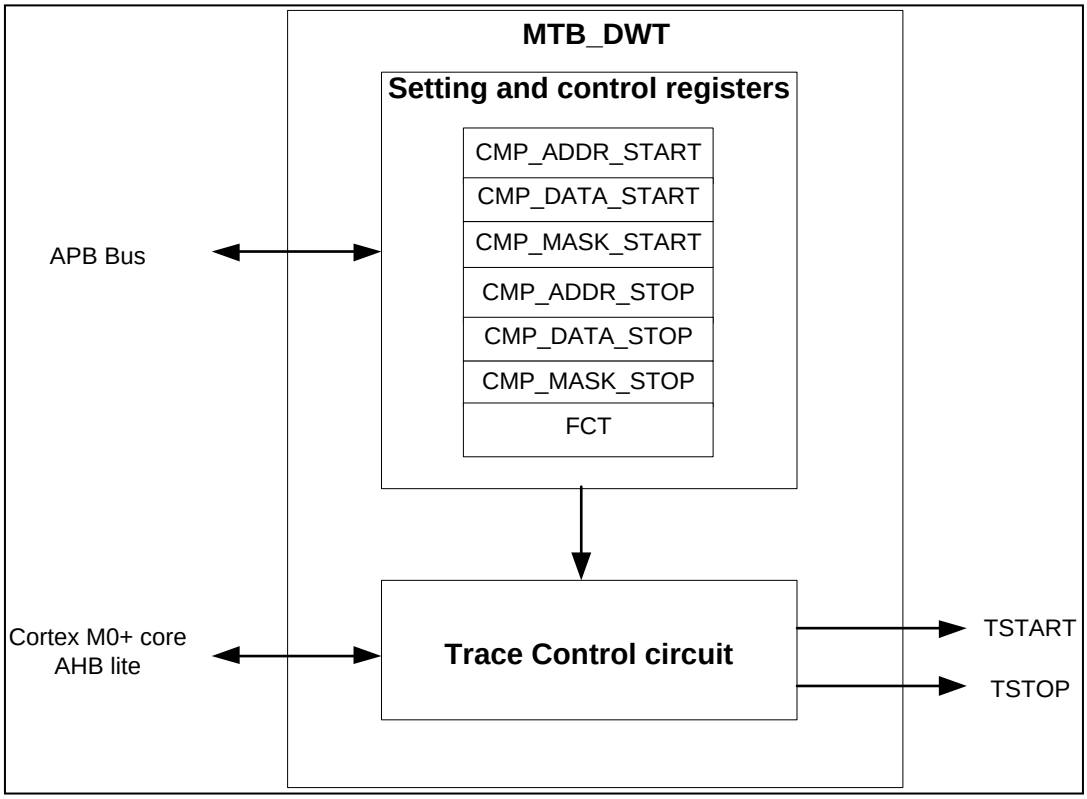
**Features of MTB_DWT**

- Support addresses and address + data programmable start/stop recording.
- Support masking each bit of data.
- Support write/read and read or write operations monitor.
- Support byte, half-word and word monitor.
- Support data phase monitor function.

# 2. Block Diagram

The following shows the block diagram of the MTB_DWT.

**Figure 2-1 Block diagram of MTB_DWT**



# 3. Configuration and Setting Procedure Examples

This section explains configurations and setting procedure examples.

## 3.1 Configurations of MTB_DWT

This section shows the configurations of the MTB_DWT

### Overview of configuration in MTB_DWT

MTB_DWT only supports little-endian data format. When the byte or half-word access is enabled, the unused bit of MTB_DWT Data Compare Start trace Register (CMP_DATA_START) and MTB_DWT Data Compare Stop trace Register (CMP_DATA_STOP) is shown as Table 3-1.The sign of "○" in the table shows that it's valid data. The sign of "-" in the table shows that it's an invalid data.
Mask the corresponding bit with invalid data by setting CMP_MASK_START/CMP_MASK_STOP.

**Table 3-1 AHB-Lite byte lane definition**

| Access | Address phase | Corresponding data phase | | | |
|---|---|---|---|---|---|
| | CMP_ADDRx* [1:0] | CMP_DATAx*2 [31:24] | CMP_DATAx*2 [23:16] | CMP_DATAx*2 [15:8] | CMP_DATAx*2 [7:0] |
| Byte | 00 | - | - | - | ○ |
| | 01 | - | - | ○ | - |
| | 10 | - | ○ | - | - |
| | 11 | ○ | - | - | - |
| Half-word | 00 | - | - | ○ | ○ |
| | 10 | ○ | ○ | - | - |
| Word | 00 | ○ | ○ | ○ | ○ |

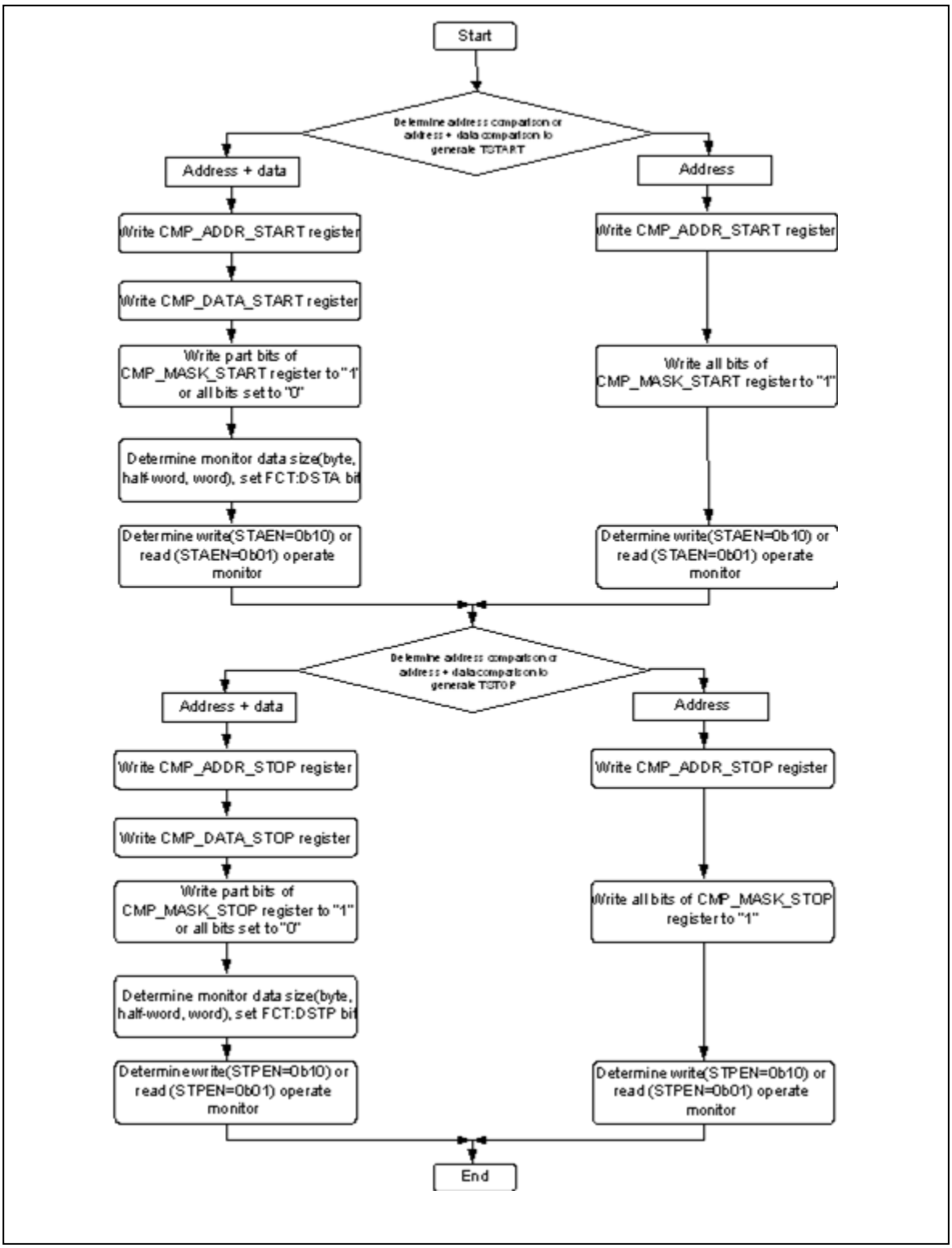
*1: CMP_ADDRx: CMP_ADDR_START register or CMP_ADDR_STOP register.

*2: CMP_DATAx: CMP_DATA_START register or CMP_DATA_STOP register.

## 3.2 Setting procedure

For MTB_DWT setting procedure example, see Figure 3-1

**Figure 3-1 MTB_DWT setting procedure**



# 4. Registers

This section describes the registers of the MTB_DWT

### List of registers of the MTB_DWT

**Table 4-1 List of registers of the MTB_DWT**

| Abbreviation | Register Name | Reference |
|---|---|---|
| CMP_ADDR_START | MTB_DWT Address Compare Start trace Register | 4.1 |
| CMP_DATA_START | MTB_DWT Data Compare Start trace Register | 4.2 |
| CMP_MASK_START | MTB_DWT Mask Data Compare Start trace Register | 4.3 |
| CMP_ADDR_STOP | MTB_DWT Address Compare Stop trace Register | 4.4 |
| CMP_DATA_STOP | MTB_DWT Data Compare Stop trace Register | 4.5 |
| CMP_MASK_STOP | MTB_DWT Mask Data Compare Stop trace Register | 4.6 |
| FCT | MTB_DWT Function Register | 4.7 |
| PID4 | Peripheral ID4 Register | 4.8 |
| PID5 | Peripheral ID5 Register | |
| PID6 | Peripheral ID6 Register | |
| PID7 | Peripheral ID7 Register | |
| PID0 | Peripheral ID0 Register | |
| PID1 | Peripheral ID1 Register | |
| PID2 | Peripheral ID2 Register | |
| PID3 | Peripheral ID3 Register | |
| CID0 | Component ID0 Register | 4.9 |
| CID1 | Component ID1 Register | |
| CID2 | Component ID2 Register | |
| CID3 | Component ID3 Register | |

## 4.1 MTB_DWT Address Compare Start trace Register (CMP_ADDR_START)

The MTB_DWT Address Compare Start trace Register (CMP_ADDR_START) provide a reference address value for generating start trigger signal TSTART.

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | ADCMP_STA[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | ADCMP_STA[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

**Register functions**

**[bit31:0] ADCMP_STA[31:0]: MTB_DWT address comparison start trace bits**
Reference value for address comparison is to generate MTB start trigger.

***Notes:***

- *When word accessing, the bit[3:0] must set to one of 0x0, 0x4, 0x8 and 0xC.*
- *When half-word accessing, the bit[1:0] must set to one of 0x0 and 0x2.*
- *This register can be cleared by PRESET0. After PRESET0, this register should be configured again.*

## 4.2 MTB_DWT Data Compare Start trace Register (CMP_DATA_START)

The MTB_DWT Data Compare Start trace Register (CMP_DATA_START) provides a reference data value for generating start trigger signal TSTART.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | DTCMP_STA[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | DTCMP_STA[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

### Register functions

**[bit31:0] DTCMP_STA[31:0] : MTB_DWT data comparison start trace bits**

Reference value for data comparison is to generate MTB start trigger.

When all bits are masked by CMP_MASK_START, the MTB TSTART signal is only determined by comparative address result, otherwise the MTB TSTART signal is determined by both data CMP_DATA_START and CMP_ADDR_START comparative result.

***Notes:***

- *When byte or half-word accessing, set CMP_MASK_START register following Table 3-1 to ignore unused bit in CMP_DATA_START register.*
- *This register can be cleared by PRESET0. After PRESET0, this register should be configured again.*

## 4.3 MTB_DWT Mask Data Compare Start Trace Register (CMP_MASK_START)

The MTB_DWT Mask Data Compare Start trace Register (CMP_MASK_START) defines the ignored bit of MTB_DWT Data Compare Start trace Register (CMP_DATA_START).

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | MSK_STA[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | MSK_STA [15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

### Register functions

**[bit31:0] MSK_STA[31:0] : MTB_DWT data compare start trace register mask bits**

This register masks reference data value for starting MTB.

| bit | Function |
|---|---|
| 0 | No effect on operation [Initial value] |
| 1 | Mask corresponding bits |

***Notes:***

- *When all bits of this register are set to "1", the value of CMP_DATA_START register and the value of DSTA bit of FCT register will be ignored.*
- *This register can be cleared by PRESET0. After PRESET0, this register should be configured again.*

## 4.4 MTB_DWT Address Compare Stop trace Register (CMP_ADDR_STOP)

The MTB_DWT Address Compare Stop trace Register (CMP_ADDR_STOP) provide a reference address value for generating stop trigger signal TSTOP.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | ADCMP_STO[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | ADCMP_STO[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

### Register functions

**[bit31:0] ADCMP_STO[31:0]: MTB_DWT address comparison stop trace bits**
Reference value for address comparison is to generate MTB stop trigger.

***Notes:***

- *When word accessing, the bit[3:0] must set to one of 0x0, 0x4, 0x8 and 0xC.*
- *When half-word accessing, the bit[1:0] must set to one of 0x0 and 0x2.*
- *This register can be cleared by PRESET0. After PRESET0, this register should be configured again.*

## 4.5 MTB_DWT Data Compare Stop trace Register (CMP_DATA_STOP)

The MTB_DWT Data Compare Stop trace Register (CMP_DATA_STOP) provides a reference data value for generating stop trigger signal TSTOP.

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | DTCMP_STO[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | DTCMP_STO[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

**Register functions**

**[bit31:0] DTCMP_STO[31:0]: MTB_DWT data comparison stop trace bits**
Reference value for data comparison is to generate MTB stop trigger.

When all bits are masked by MTB_DWT Mask data Compare Stop trace Register, the MTB TSTOP signal is only determined by comparative address result, otherwise the MTB TSTOP signal is determined by both data (MTB_DWT Data Compare Stop trace Register) and address (MTB_DWT Address Compare Stop trace Register) comparative result.

***Notes:***

- *When byte or half-word accessing, set CMP_MASK_STOP register following Table 3-1 <Not to ignore unused bit in CMP_DATA_STOP register.*
- *This register can be cleared by PRESET0. After PRESET0, this register should be configured again.*

## 4.6 MTB_DWT Mask Data Compare Stop Trace Register (CMP_MASK_STOP)

The MTB_DWT Mask Data Compare Stop Trace Register (CMP_MASK_STOP) defines the ignored bit of register MTB_DWT Data Compare Stop trace Register (CMP_DATA_STOP).

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | MSK_STO[31:16] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | MSK_STO[15:0] | | | | | | | | | | | | | | | |
| Attribute | R/W | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

### Register functions

**[bit31:0] MSK_STO[31:0]: MTB_DWT data compare stop trace register mask bits**

This register masks reference data value for stopping MTB.

| bit | Function |
|---|---|
| 0 | No effect on operation. [Initial value] |
| 1 | Mask corresponding bits. |

***Notes:***

- *When all bits of this register are set to "1", the value of CMP_DATA_STOP register and the value of DSTP bit of FCT register will be ignored.*
- *This register can be cleared by PRESET0. After PRESET0, this register should be configured again.*

## 4.7 MTB_DWT Function Register (FCT)

The MTB_DWT Function Register (FCT) controls read/write operation and data size.

**Register configuration**

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | | | | | | | | |
| Attribute | - | | | | | | | | | | | | | | | |
| Initial value | 0x0000 | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | | DSTP | | DSTA | | STPEN | | STAEN | |
| Attribute | - | | | | | | | | R/W | | R/W | | R/W | | R/W | |
| Initial value | 00000000 | | | | | | | | 00 | | 00 | | 00 | | 00 | |

**Register functions**

**[bit31:8] Reserved: Reserved bits**

The read value is “0”. They have no effect in write mode.

**[bit7:6] DSTP: Data size stop bits**

These bits define the data value size to stop MTB function.

| bit7 | bit6 | Function |
|---|---|---|
| 0 | 0 | Byte [Initial value] |
| 0 | 1 | Half-word |
| 1 | 0 | Word |
| 1 | 1 | Reserved. |

**[bit5:4] DSTA: Data size start bits**

These bits define the data value size to start MTB function.

| bit5 | bit4 | Function |
|---|---|---|
| 0 | 0 | Byte [Initial value] |
| 0 | 1 | Half-word |
| 1 | 0 | Word |
| 1 | 1 | Reserved. |

**[bit3:2] STPEN: Enable MTB_DWT stop MTB function bits**

These bits enable MTB_DWT stop MTB function.

| bit3 | bit2 | Function |
|---|---|---|
| 0 | 0 | Disabled MTB_DWT stop MTB function. [Initial value] |
| 0 | 1 | Data read operation to stop MTB. |
| 1 | 0 | Data write operation to stop MTB. |
| 1 | 1 | Data write or read operation to stop MTB. |

**[bit1:0] STAEN: Enable MTB_DWT start MTB function bits**

These bits enable MTB_DWT start MTB function.

| bit1 | bit0 | Function |
|---|---|---|
| 0 | 0 | Disabled MTB_DWT start MTB function. [Initial value] |
| 0 | 1 | Data read operation to start MTB. |
| 1 | 0 | Data write operation to start MTB. |
| 1 | 1 | Data write or read operation to start MTB. |

***Notes:***

- *This register can be cleared by PRESET0. After PRESET0, this register should be configured again.*

## 4.8 Peripheral ID0-7 Register (PID0-7)

The Peripheral ID0-7 Registers (PID0-7) indicate the peripheral IDs.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | PERID[31:16] | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0xXXXX | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | PERID[15:0] | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0xXXXX | | | | | | | | | | | | | | | |

### Register functions

**[bit31:0] PERID[31:0]: Peripheral ID bits**

They are hardwired to specific values used during the auto-discovery process by an external debug agent.

| bit | Function |
|---|---|
| When Read | A specified value is read.<br>PID0: 0x00000016<br>PID1: 0x00000048<br>PID2: 0x00000008<br>PID3-7: 0x00000000 |
| When Write | No effect on operation |

## 4.9 Component ID0-3 Register (CID0-3)

The Component ID0-3 Registers (CID0-3) indicate the component IDs.

### Register configuration

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | CPNTID[31:16] | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0xXXXX | | | | | | | | | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | CPNTID[15:0] | | | | | | | | | | | | | | | |
| Attribute | R | | | | | | | | | | | | | | | |
| Initial value | 0xXXXX | | | | | | | | | | | | | | | |

### Register functions

**[bit31:0] CPNTID[31:0]: Component ID bits**

They are hardwired to specific values used during the auto-discovery process by an external debug agent.

| bit | Function |
|---|---|
| When Read | A specified value is read.<br>CID0: 0x0000000D;<br>CID1: 0x00000090;<br>CID2: 0x00000005;<br>CID3: 0x000000B1. |
| When Write | No effect on operation |

# CHAPTER14: Flash Memory

**For the flash memory, refer to the "FLASH PROGRAMMING MANUAL" of the product to be used.**

CODE: 9xFLASHTOP-E01.1

# CHAPTER15: Unique ID Register

**Functions and operations of Unique ID Register are explained as follows.**



CODE: 9BFUNIQID-J01.0

# 1. Overview

Overview of this function is explained as follows.

41 bits of preset device unique values have been set to the Unique ID Register.

These values are different from each other in all of the devices which allow using these bits for various purposes such as security enhancement and product serial number.

This register is a read-only register which cannot be written by the user. Also, these values will not be changed due to reset or power on/off.

# 2. Registers

Configuration and functions of registers are explained as follows.

### Registers list

| Abbreviated Name | Register Name | Reference |
|---|---|---|
| UIDR0 | Unique ID Register 0 | 2.1 |
| UIDR1 | Unique ID Register 1 | 2.2 |

## 2.1 Unique ID Register 0 (UIDR0: Unique ID Register 0)

Unique ID Register 0 is explained as follows.

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | UID[27:20] | | | | | | | |
| Attribute | R | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | UID[19:12] | | | | | | | |
| Attribute | R | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | UID[11:4] | | | | | | | |
| Attribute | R | | | | | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | UID[3:0] | | | | Reserved | | | |
| Attribute | R | | | | - | | | |

**[bit31:4] UID[27:0] : Unique ID 27 through Unique ID 0**
bit27 through bit0 of the unique ID.

**[bit3:0] Reserved : Reserved bits**
Reserved bits. Read values have no meaning.

## 2.2 Unique ID Register 1 (UIDR1: Unique ID Register 1)

Unique ID Register 1 is explained as follows.

| bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |

| bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | | | | |
| Attribute | - | | | | | | | |

| bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | UID[40:36] | | | |
| Attribute | - | | | | R | | | |

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | UID[35:28] | | | | | | | |
| Attribute | R | | | | | | | |

**[bit31:13] Reserved : Reserved bits**
Reserved bits. Read values have no meaning.

**[bit12:0] UID[40:28] : Unique ID 40 through Unique ID 28**
bit 40 through bit28 of unique ID.

# APPENDIXES

**This chapter shows the register map and list of notes.**

A. Register Map

B. List of Notes

CODE: 9AFAPPENDIXES-E01.0

# A. Register Map

**This chapter shows the register map.**



CODE: 9AFREGMAP-E01.0

# 1. Register Map

Register map is shown on the table every module/function.

**[How to read the each table]**

Module/function name and its base address

**Clock/Reset** **Base_Address : 0x4001_0000**

| Base_Address + Address | Register | | | |
|---|---|---|---|---|
| | +3 | +2 | +1 | +0 |
| 0x000 | - | - | - | SCM_CTL[B,H,W]<br>00000-01 |
| 0x004 | - | - | - | SCM_STR[B,H,W]<br>00000-01 |
| 0x008 | STB_CTL[B,H,W]<br>00000000 00000000 -------- ---0--00 | | | |
| 0x00C | - | - | RST_STR[B,H,W]<br>-------0 00000-01 | |
| | | | | |

- : Reserved area
* : Test register area

Initial value after reset

| | | |
|---|---|---|
| "1" | : | Initial value is "1" |
| "0" | : | Initial value is "0" |
| "X" | : | Initial value is undefined |
| " - " | : | Reserved bit |

Register name

Access unit
(B : byte, H : half word, W : word)

Rightmost register address (For word-length access, the "+0" column of the register is the LSB of the data.)

***Notes:***

- The register table is represented in the little-endian.
- When performing a data access, the addresses should be as below according to the access size.
  - Word access : Address should be multiples of 4 (least significant 2 bits should be "0x00")
  - Half word access : Address should be multiples of 2 (least significant bit should be "0x0")
  - Byte access : -
- Do not access the test register area.
- Do not access the area that is not written in the register table.

### Flash I/F Base_Address : 0x4000_0000

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | - | - |
| 0x004 | FRWTR[B,H,W] | | | |
| 0x008 | FSTR[B,H,W] | | | |
| 0x00C | | | | |
| 0x010 | FSYNDN[B,H,W] | | | |
| 0x014 - 0x01C | - | - | - | - |
| 0x020 | FICR[B,H,W] | | | |
| 0x024 | FISR[B,H,W] | | | |
| 0x028 | FICLR[B,H,W] | | | |
| 0x02C - 0x0FC | - | - | - | - |
| 0x100 | CRTRMM[B,H,W] | | | |
| 0x104 - 0x1FC | - | - | - | - |

***Note:***

- *For details of Flash I/F registers, refer to "FLASH PROGRAMMING MANUAL" of the product used.*

### Unique ID Base_Address : 0x4000_0200

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | UIDR0[W] | | | |
| | XXXXXXXX XXXXXXXX XXXXXXXX XXXX---- | | | |
| 0x004 | UIDR1[W] | | | |
| | -------- -------- ---XXXXX XXXXXXXX | | | |
| 0x008 – 0xDFC | - | - | - | - |

**Clock/Reset Base_Address : 0x4001_0000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | - | SCM_CTL[W] |
| | | | | 00000-01 |
| 0x004 | - | - | - | SCM_STR[W] |
| | | | | 00000-01 |
| 0x008 | STB_CTL[W] | | | |
| | 00000000 00000000 -------- ---0-000 | | | |
| 0x00C | - | - | RST_STR[W] | |
| | | | -------0 0000--01 | |
| 0x010 | - | - | - | BSC_PSR[W] |
| | | | | -----000 |
| 0x014 | - | - | - | APBC0_PSR[W] |
| | | | | ------00 |
| 0x018 | - | - | - | APBC1_PSR[W] |
| | | | | 1--0--00 |
| 0x01C | - | - | - | - |
| 0x020 | - | - | - | SWC_PSR[W] |
| | | | | X-----00 |
| 0x024 - 0x02C | - | - | - | - |
| 0x030 | - | - | - | CSW_TMR[W] |
| | | | | 00000000 |
| 0x034 | - | - | - | PSW_TMR[W] |
| | | | | ---0-000 |
| 0x038 | - | - | - | PLL_CTL1[W] |
| | | | | 00000000 |
| 0x03C | - | - | - | PLL_CTL2[W] |
| | | | | --000000 |
| 0x040 | - | - | CSV_CTL[W] | |
| | | | -111--00 ------11 | |
| 0x044 | - | - | - | CSV_STR[W] |
| | | | | ------00 |
| 0x048 | - | - | FCSWH_CTL[W] | |
| | | | 11111111 11111111 | |
| 0x04C | - | - | FCSWL_CTL[W] | |
| | | | 00000000 00000000 | |
| 0x050 | - | - | FCSWD_STR[W] | |
| | | | 00000000 00000000 | |
| 0x054 | - | - | - | DBWDT_CTL[W] |
| | | | | 0-0----- |
| 0x058 | - | - | - | * |
| 0x05C | - | - | - | - |
| 0x060 | - | - | - | INT_ENR[W] |
| | | | | --0--000 |
| 0x064 | - | - | - | INT_STR[W] |
| | | | | --0--000 |
| 0x068 | - | - | - | INT_CLR[W] |
| | | | | --0--000 |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x06C - 0xFFC | - | - | - | - |

**HW WDT Base_Address : 0x4001_1000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | WDG_LDR[W] | | | |
| | 00000000 00000000 11111111 11111111 | | | |
| 0x004 | WDG_VLR[W] | | | |
| | XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0x008 | - | - | - | WDG_CTL[W] |
| | | | | ------11 |
| 0x00C | - | - | - | WDG_ICL[W] |
| | | | | XXXXXXXX |
| 0x010 | - | - | - | WDG_RIS[W] |
| | | | | -------0 |
| 0x014 - 0xBFC | - | - | - | - |
| 0xC00 | WDG_LCK[W] | | | |
| | 00000000 00000000 00000000 00000001 | | | |
| 0xC04 - 0xFFC | - | - | - | - |

**SW WDT Base_Address : 0x4001_2000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | WdogLoad[W] | | | |
| | 11111111 11111111 11111111 11111111 | | | |
| 0x004 | WdogValue[W] | | | |
| | 11111111 11111111 11111111 11111111 | | | |
| 0x008 | - | - | - | WdogControl[W] |
| | | | | ---00000 |
| 0x00C | WdogIntClr[W] | | | |
| | XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0x010 | - | - | - | WdogRIS[W] |
| | | | | -------0 |
| 0x014 | * | | | |
| 0x018 | - | - | - | WdogSPMC[W] |
| | | | | -------0 |
| 0x01C - 0xBFC | - | - | - | - |
| 0xC00 | WdogLock[W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0xC04 - 0xDFC | - | - | - | - |
| 0xF00 | * | | | |
| 0xF08 - 0xFDF | - | - | - | - |
| 0xFE0 - 0xFFC | * | | | |

**Dual Timer** **Base_Address : 0x4001_5000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | Timer1Load[W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x004 | Timer1Value[W] | | | |
| | 11111111 11111111 11111111 11111111 | | | |
| 0x008 | Timer1Control[W] | | | |
| | -------- -------- -------- 00100000 | | | |
| 0x00C | Timer1IntClr[W] | | | |
| | XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0x010 | Timer1RIS[W] | | | |
| | -------- -------- -------- -------0 | | | |
| 0x014 | Timer1MIS[W] | | | |
| | -------- -------- -------- -------0 | | | |
| 0x018 | Timer1BGLoad[W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x020 | Timer2Load[W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x024 | Timer2Value[W] | | | |
| | 11111111 11111111 11111111 11111111 | | | |
| 0x028 | Timer2Control[W] | | | |
| | -------- -------- -------- 00100000 | | | |
| 0x02C | Timer2IntClr[W] | | | |
| | XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0x030 | Timer2RIS[W] | | | |
| | -------- -------- -------- -------0 | | | |
| 0x034 | Timer2MIS[W] | | | |
| | -------- -------- -------- -------0 | | | |
| 0x038 | Timer2BGLoad[W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x040 - 0xFFC | - | - | - | - |

**MFT unit0 Base_Address : 0x4002_0000**

**MFT unit1 Base_Address : 0x4002_1000**

**MFT unit2 Base_Address : 0x4002_2000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x100 | OCCP0[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x104 | OCCP1[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x108 | OCCP2[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x10C | OCCP3[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x110 | OCCP4[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x114 | OCCP5[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x118 | - | OCSD10[B,H,W] | OCSB10[B,H,W] | OCSA10[B,H,W] |
| | | 00000000 | 00000000 | 00000000 |
| 0x11C | - | OCSD32[B,H,W] | OCSB32[B,H,W] | OCSA32[B,H,W] |
| | | 00000000 | 00000000 | 00000000 |
| 0x120 | - | OCSD54[B,H,W] | OCSB54[B,H,W] | OCSA54[B,H,W] |
| | | 00000000 | 00000000 | 00000000 |
| 0x124 | - | - | OCSC[B,H,W] | - |
| | | | --000000 | |
| 0x128 | - | - | OCSE0[H,W] | |
| | | | 00000000 00000000 | |
| 0x12C | OCSE1[H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x130 | - | - | OCSE2[H,W] | |
| | | | 00000000 00000000 | |
| 0x134 | OCSE3[H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x138 | - | - | OCSE4[H,W] | |
| | | | 00000000 00000000 | |
| 0x13C | OCSE5[H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x140 | TCCP0[H,W] | | - | - |
| | 11111111 11111111 | | | |
| 0x144 | TCDT0[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x148 | TCSC0[B,H,W] | | TCSA0[B,H,W] | |
| | 00000000 00000000 | | 000---00 01000000 | |
| 0x14C | TCCP1[H,W] | | - | - |
| | 11111111 11111111 | | | |
| 0x150 | TCDT1[H,W] | | - | - |
| | 00000000 00000000 | | | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x154 | TCSC1[B,H,W] | | TCSA1[B,H,W] | |
| | 00000000 00000000 | | 000---00 01000000 | |
| 0x158 | TCCP2[H,W] | | - | - |
| | 11111111 11111111 | | | |
| 0x15C | TCDT2[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x160 | TCSC2[B,H,W] | | TCSA2[B,H,W] | |
| | 00000000 00000000 | | 000---00 01000000 | |
| 0x164 | TCAL[B,H,W] (only in unit 0) | | | |
| | 00000000 00000000 11111111 11111111 | | | |
| 0x168 | - | OCFS54[B,H,W] | OCFS32[B,H,W] | OCFS10[B,H,W] |
| | | 00000000 | 00000000 | 00000000 |
| 0x16C | - | - | ICFS32[B,H,W] | ICFS10[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x170 | - | ACFS54[B,H,W] | ACFS32[B,H,W] | ACFS10[B,H,W] |
| | | 00000000 | 00000000 | 00000000 |
| 0x174 | ICCP0[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x178 | ICCP1[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x17C | ICCP2[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x180 | ICCP3[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x184 | - | - | ICSB10[B,H,W] | ICSA10[B,H,W] |
| | | | ------00 | 00000000 |
| 0x188 | - | - | ICSB32[B,H,W] | ICSA32[B,H,W] |
| | | | ------00 | 00000000 |
| 0x18C | WFTF10[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x190 | WFTB10[H,W] | | WFTA10[H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x194 | WFTF32[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x198 | WFTB32[H,W] | | WFTA32[H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x19C | WFTF54[H,W] | | - | - |
| | 00000000 00000000 | | | |
| 0x1A0 | WFTB54[H,W] | | WFTA54[H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x1A4 | - | - | WFSA10[H,W] | |
| | | | ---00000 000000 | |
| 0x1A8 | - | - | WFSA32[H,W] | |
| | | | ---00000 000000 | |
| 0x1AC | - | - | WFSA54[H,W] | |
| | | | ---00000 000000 | |
| 0x1B0 | - | - | WFIR[H,W] | |
| | | | 00000000 00000000 | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x1B4 | - | - | NZCL[H,W] | |
| | | | -000--00 ---00000 | |
| 0x1B8 | ACMP0 | | - | - |
| | 00000000 00000000 | | | |
| 0x1BC | ACMP1 | | - | - |
| | 00000000 00000000 | | | |
| 0x1C0 | ACMP2 | | - | - |
| | 00000000 00000000 | | | |
| 0x1C4 | ACMP3 | | - | - |
| | 00000000 00000000 | | | |
| 0x1C8 | ACMP4 | | - | - |
| | 00000000 00000000 | | | |
| 0x1CC | ACMP5 | | - | - |
| | 00000000 00000000 | | | |
| 0x1D0 | - | - | ACSA[B,H,W] | |
| | | | --000000 --000000 | |
| 0x1D4 | - | - | ACSD0[B,H,W] | ACSC0[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x1D8 | - | - | ACSD1[B,H,W] | ACSC1[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x1DC | - | - | ACSD2[B,H,W] | ACSC2[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x1E0 | - | - | ACSD3[B,H,W] | ACSC3[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x1E4 | - | - | ACSD4[B,H,W] | ACSC4[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x1E8 | - | - | ACSD5[B,H,W] | ACSC5[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x1EC - 0xFFC | - | - | - | - |

**PPG Base_Address : 0x4002_4000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | TTCR0[B,H,W]<br>11110000 | - |
| 0x004 | - | - | - | * |
| 0x008 | - | - | COMP0[B,H,W]<br>00000000 | - |
| 0x00C | - | - | - | COMP2[B,H,W]<br>00000000 |
| 0x010 | - | - | COMP4[B,H,W]<br>00000000 | - |
| 0x014 | - | - | - | COMP6[B,H,W]<br>00000000 |
| 0x018 - 0x01C | - | - | - | - |
| 0x020 | - | - | TTCR1[B,H,W]<br>11110000 | - |
| 0x024 | - | - | - | * |
| 0x028 | - | - | COMP1[B,H,W]<br>00000000 | - |
| 0x02C | - | - | - | COMP3[B,H,W]<br>00000000 |
| 0x030 | - | - | COMP5[B,H,W]<br>00000000 | - |
| 0x034 | - | - | - | COMP7[B,H,W]<br>00000000 |
| 0x038 - 0x03C | - | - | - | - |
| 0x040 | - | - | TTCR2[B,H,W]<br>11110000 | - |
| 0x044 | - | - | - | * |
| 0x048 | - | - | COMP8[B,H,W]<br>00000000 | - |
| 0x04C | - | - | - | COMP10[B,H,W]<br>00000000 |
| 0x050 | - | - | COMP12[B,H,W]<br>00000000 | - |
| 0x054 | - | - | - | COMP14[B,H,W]<br>00000000 |
| 0x058 - 0x0FC | - | - | - | - |
| 0x100 | - | - | TRG0[B,H,W]<br>00000000 00000000 | |
| 0x104 | - | - | REVC0[B,H,W]<br>00000000 00000000 | |
| 0x108 - 0x13C | - | - | - | - |
| 0x140 | - | - | TRG1[B,H,W]<br>-------- 00000000 | |
| 0x144 | - | - | REVC1[B,H,W]<br>-------- 00000000 | |
| 0x148 - 0x1FC | - | - | - | - |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x200 | - | - | PPGC0[B,H,W] | PPGC1[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x204 | - | - | PPGC2[B,H,W] | PPGC3[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x208 | - | - | PRLH0[B,H,W] | PRLL0[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x20C | - | - | PRLH1[B,H,W] | PRLL1[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x210 | - | - | PRLH2[B,H,W] | PRLL2[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x214 | - | - | PRLH3[B,H,W] | PRLL3[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x218 | - | - | - | GATEC0[B,H,W] |
| | | | | --00---00 |
| 0x21C - 0x23C | - | - | - | - |
| 0x240 | - | - | PPGC4[B,H,W] | PPGC5[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x244 | - | - | PPGC6[B,H,W] | PPGC7[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x248 | - | - | PRLH4[B,H,W] | PRLL4[B.H.W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x24C | - | - | PRLH5[B,H,W] | PRLL5[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x250 | - | - | PRLH6[B,H,W] | PRLL6[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x254 | - | - | PRLH7[B,H,W] | PRLL7[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x258 | - | - | - | GATEC4[B,H,W] |
| | | | | --00--00 |
| 0x25C - 0x27C | - | - | - | - |
| 0x280 | - | - | PPGC8[B,H,W] | PPGC9[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x284 | - | - | PPGC10[B,H,W] | PPGC11[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x288 | - | - | PRLH8[B,H,W] | PRLL8[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x28C | - | - | PRLH9[B,H,W] | PRLL9[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x290 | - | - | PRLH10[B,H,W] | PRLL10[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x294 | - | - | PRLH11[B,H,W] | PRLL11[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x298 | - | - | - | GATEC8[B,H,W] |
| | | | | --00--00 |
| 0x29C - 0x2BC | - | - | - | - |
| 0x2C0 | - | - | PPGC12[B,H,W] | PPGC13[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x2C4 | - | - | PPGC14[B,H,W] | PPGC15[B,H,W] |
| | | | 00000000 | 00000000 |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x2C8 | - | - | PRLH12[B,H,W] | PRLL12[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x2CC | - | - | PRLH13[B,H,W] | PRLL13[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x2D0 | - | - | PRLH14[B,H,W] | PRLL14[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x2D4 | - | - | PRLH15[B,H,W] | PRLL15[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x2D8 | - | - | - | GATEC12[B,H,W] |
| | | | | --00--00 |
| 0x2DC - 0x2FC | - | - | - | - |
| 0x300 | - | - | PPGC16[B,H,W] | PPGC17[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x304 | - | - | PPGC18[B,H,W] | PPGC19[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x308 | - | - | PRLH16[B,H,W] | PRLL16[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x30C | - | - | PRLH17[B,H,W] | PRLL17[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x310 | - | - | PRLH18[B,H,W] | PRLL18[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x314 | - | - | PRLH19[B,H,W] | PRLL19[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x318 | - | - | - | GATEC16[B,H,W] |
| | | | | --00---00 |
| 0x31C - 0x33C | - | - | - | - |
| 0x340 | - | - | PPGC20[B,H,W] | PPGC21[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x344 | - | - | PPGC22[B,H,W] | PPGC23[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x348 | - | - | PRLH20[B,H,W] | PRLL20[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x34C | - | - | PRLH21[B,H,W] | PRLL21[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x350 | - | - | PRLH22[B,H,W] | PRLL22[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x354 | - | - | PRLH23[B,H,W] | PRLL23[B,H,W] |
| | | | XXXXXXXX | XXXXXXXX |
| 0x358 | - | - | - | GATEC20[B,H,W] |
| | | | | --00--00 |
| 0x35C - 0x37C | - | - | - | - |
| 0x380 | - | - | - | IGBTC[B,H,W] |
| | | | | 00000000 |
| 0x384 - 0xFFC | - | - | - | - |

**Base Timer ch.0 Base Address : 0x4002_5000**

**Base Timer ch.1 Base Address : 0x4002_5040**

**Base Timer ch.2 Base Address : 0x4002_5080**

**Base Timer ch.3 Base Address : 0x4002_50C0**

**Base Timer ch.4 Base Address : 0x4002_5200**

**Base Timer ch.5 Base Address : 0x4002_5240**

**Base Timer ch.6 Base Address : 0x4002_5280**

**Base Timer ch.7 Base Address : 0x4002_52C0**

**Base Timer ch.8 Base Address : 0x4002_5400**

**Base Timer ch.9 Base Address : 0x4002_5440**

**Base Timer ch.10 Base Address : 0x4002_5480**

**Base Timer ch.11 Base Address : 0x4002_54C0**

**Base Timer ch.12 Base Address : 0x4002_5600**

**Base Timer ch.13 Base Address : 0x4002_5640**

**Base Timer ch.14 Base Address : 0x4002_5680**

**Base Timer ch.15 Base Address : 0x4002_56C0**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | PCSR/PRLL[H,W] | |
| | | | XXXXXXXX XXXXXXXX | |
| 0x004 | - | - | PDUT/PRLH/DTBF[H,W] | |
| | | | XXXXXXXX XXXXXXXX | |
| 0x008 | - | - | TMR[H,W] | |
| | | | 00000000 00000000 | |
| 0x00C | - | - | TMCR[B,H,W] | |
| | | | -0000000 00000000 | |
| 0x010 | - | - | TMCR2[B,H,W] | STC[B,H,W] |
| | | | -------0 | 0000-000 |
| 0x014 - 0x03C | - | - | - | - |

**IO Selector for ch.0-ch.3 (Base Timer)** **Base Address : 0x4002_5100**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | BTSEL0123[B,H,W]<br>00000000 | - |
| 0x004 - 0x0FC | - | - | - | - |

**IO Selector for ch.4-ch.7 (Base Timer)** **Base Address : 0x4002_5300**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | BTSEL4567[B,H,W]<br>00000000 | - |
| 0x004 - 0x0FC | - | - | - | - |

**IO Selector for ch.8-ch.11 (Base Timer)** **Base Address : 0x4002_5500**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | BTSEL89AB[B,H,W]<br>00000000 | - |
| 0x004 - 0x0FC | - | - | - | - |

**IO Selector for ch.12-ch.15 (Base Timer)** **Base Address : 0x4002_5700**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | BTSELCDEF[B,H,W]<br>00000000 | - |
| 0x004 - 0x0FC | - | - | - | - |

**Software-based Simulation Startup (Base Timer)** **Base Address : 0x4002_5F00**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 - 0x0FB | - | - | - | - |
| 0x0FC | - | - | BTSSSR[B,H,W]<br>XXXXXXXX XXXXXXXX | |

**QPRC ch.0 Base Address : 0x4002_6000**

**QPRC ch.1 Base Address : 0x4002_6040**

**QPRC ch.2 Base Address : 0x4002_6080**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x0000 | - | - | QPCR[H,W] | |
| | | | 00000000 00000000 | |
| 0x0004 | - | - | QRCR[H,W] | |
| | | | 00000000 00000000 | |
| 0x0008 | - | - | QPCCR[H,W] | |
| | | | 00000000 00000000 | |
| 0x000C | - | - | QPRCR[H,W] | |
| | | | 00000000 00000000 | |
| 0x0010 | - | - | QMPR[H,W] | |
| | | | 11111111 11111111 | |
| 0x0014 | - | - | QICRH[B,H,W] | QICRL[B,H,W] |
| | | | --000000 | 00000000 |
| 0x0018 | - | - | QCRH[B,H,W] | QCRL[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x001C | - | - | QECR[B,H,W] | |
| | | | -------- -----000 | |
| 0x0020 - 0x0038 | - | - | - | - |
| 0x003C | QPCRR[B,H,W] | | QRCRR[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |

**QPRC ch.0 NF Base Address : 0x4002_6100**

**QPRC ch.1 NF Base Address : 0x4002_6110**

**QPRC ch.2 NF Base Address : 0x4002_6120**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x0000 | NFCTLA[B,H,W] | | | |
| | -------- -------- -------- --00-000 | | | |
| 0x0004 | NFCTLB[B,H,W] | | | |
| | -------- -------- -------- --00-000 | | | |
| 0x0008 | NFCTLC[B,H,W] | | | |
| | -------- -------- -------- --00-000 | | | |
| 0x000C | - | - | - | - |

**12-bit A/DC unit0 Base_Address : 0x4002_7000**

**12-bit A/DC unit1 Base_Address : 0x4002_7100**

**12-bit A/DC unit2 Base_Address : 0x4002_7200**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | ADCR[B,H,W] | ADSR[B,H,W] |
| | | | 000-0000 | 00---000 |
| 0x004 | - | - | - | * |
| 0x008 | - | - | SCCR[B,H,W] | SFNS[B,H,W] |
| | | | 1000-000 | ----0000 |
| 0x00C | SCFD[B,H,W] | | | |
| | XXXXXXXX XXXX---- ---X--XX ---XXXXX | | | |
| 0x010 | - | - | SCIS3[B,H,W] | SCIS2[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x014 | - | - | SCIS1[B,H,W] | SCIS0[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x018 | - | - | PCCR[B,H,W] | PFNS[B,H,W] |
| | | | 1000-000 | --XX--00 |
| 0x01C | PCFD[B,H,W] | | | |
| | XXXXXXXX XXXX---- ---X-XXX ---XXXXX | | | |
| 0x020 | - | - | - | PCIS[B,H,W] |
| | | | | 00000000 |
| 0x024 | CMPD[B,H,W] | | - | CMPCR[B,H,W] |
| | 00000000 00------ | | | 00000000 |
| 0x028 | - | - | ADSS3[B,H,W] | ADSS2[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x02C | - | - | ADSS1[B,H,W] | ADSS0[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x030 | - | - | ADST0[B,H,W] | ADST1[B,H,W] |
| | | | 00010000 | 00010000 |
| 0x034 | - | - | - | ADCT[B,H,W] |
| | | | | 00000111 |
| 0x038 | - | - | SCTSL[B,H,W] | PRTSL[B,H,W] |
| | | | ----0000 | ----0000 |
| 0x03C | - | - | ADCEN[B,H,W] | |
| | | | 11111111 ------00 | |
| 0x040 | * | | | |
| 0x044 | WCMRCIF[B,H,W] | | | |
| | -------- -------- -------- -------0 | | | |
| 0x048 | WCMRCOT[B,H,W] | | | |
| | -------- -------- -------- -------0 | | | |
| 0x04C | - | - | WCMPSR[B,H,W] | WCMRPCR[B,H,W] |
| | | | --000000 | 001000-- |
| 0x050 | WCMPDH[B,H,W] | | WCMPDL[B,H,W] | |
| | 00000000 00------ | | 00000000 00------ | |
| 0x054 - 0x0FC | - | - | - | - |

**10-bit D/AC  Base_Address : 0x4002_8000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x00 | - | DACR0[B,H,W] | DADR0[B,H,W] | |
| | | -------0 | ------XX XXXXXXXX | |
| 0x04 | - | DACR1[B,H,W] | DADR1[B,H,W] | |
| | | -------0 | ------XX XXXXXXXX | |
| 0x08 - 0xFC | - | - | - | - |

**CR Trim  Base_Address : 0x4002_E000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | - | MCR_PSR[B,H,W] |
| | | | | ------001 |
| 0x004 | - | - | MCR_FTRM[B,H,W] | |
| | | | ------01 11101111 | |
| 0x008 | - | - | - | MCR_TTRM[B,H,W] |
| | | | | ---10000 |
| 0x00C | MCR_RLR[B,H,W] | | | |
| | 00000000 00000000 00000000 00000001 | | | |
| 0x010 - 0x0FC | - | - | - | - |

**EXTI  Base_Address : 0x4003_0000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | ENIR[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x004 | EIRR[B,H,W] | | | |
| | XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0x008 | EICL[B,H,W] | | | |
| | 11111111 11111111 11111111 11111111 | | | |
| 0x00C | ELVR[R/W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x010 | ELVR1[R/W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x014 | - | - | NMIRR[B,H,W] | |
| | | | -------- -------0 | |
| 0x018 | - | - | NMICL[B,H,W] | |
| | | | -------- -------1 | |
| 0x01C - 0x0FC | - | - | - | - |

**INT-Req. READ Base_Address : 0x4003_1000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | DRQSEL[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x004 | * | | | |
| 0x008 - 0x00B | - | - | - | - |
| 0x00C | - | - | - | IRQCMODE[B,H,W] |
| | | | | -------0 |
| 0x010 | EXC02MON[B,H,W] | | | |
| | -------- -------- -------- ------00 | | | |
| 0x014 | IRQ00MON[B,H,W] | | | |
| | -------- -------- -------- -------0 | | | |
| 0x018 | IRQ01MON[B,H,W] | | | |
| | -------- -------- -------- -------0 | | | |
| 0x01C | IRQ02MON[B,H,W] | | | |
| | -------- -------- -------- -------0 | | | |
| 0x020 | IRQ03MON[B,H,W] | | | |
| | -------- -------- ----0000 00000000 | | | |
| 0x024 | IRQ04MON[B,H,W] | | | |
| | -------- -------- -------- 00000000 | | | |
| 0x028 | IRQ05MON[B,H,W] | | | |
| | -------- 00000000 00000000 00000000 | | | |
| 0x02C | IRQ06MON[B,H,W] | | | |
| | -------- ----0000 00000000 00000000 | | | |
| 0x030 | IRQ07MON[B,H,W] | | | |
| | -------- -------- -------- ------00 | | | |
| 0x034 | IRQ08MON[B,H,W] | | | |
| | -------- -------- -------- ----0000 | | | |
| 0x038 | IRQ09MON[B,H,W] | | | |
| | -------- -------- -------- ------00 | | | |
| 0x03C | IRQ10MON[B,H,W] | | | |
| | -------- -------- -------- ----0000 | | | |
| 0x040 | IRQ11MON[B,H,W] | | | |
| | -------- -------- -------- ------00 | | | |
| 0x044 | IRQ12MON[B,H,W] | | | |
| | -------- -------- -------- ----0000 | | | |
| 0x048 | IRQ13MON[B,H,W] | | | |
| | -------- -------- -------- ------00 | | | |
| 0x04C | IRQ14MON[B,H,W] | | | |
| | -------- -------- -------- ----0000 | | | |
| 0x050 | IRQ15MON[B,H,W] | | | |
| | -------- -------- -------- ------00 | | | |
| 0x054 | IRQ16MON[B,H,W] | | | |
| | -------- -------- -------- ----0000 | | | |
| 0x058 | IRQ17MON[B,H,W] | | | |
| | -------- -------- -------- ------00 | | | |
| 0x05C | IRQ18MON[B,H,W] | | | |
| | -------- -------- -------- ----0000 | | | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x060 | IRQ19MON[B,H,W] | | | |
| | -------- -------- -------- ---0--00 | | | |
| 0x064 | IRQ20MON[B,H,W] | | | |
| | -------- -------- -------- ---00000 | | | |
| 0x068 | IRQ21MON[B,H,W] | | | |
| | -------- -------- -------- ---0--00 | | | |
| 0x06C | IRQ22MON[B,H,W] | | | |
| | -------- -------- -------- ---00000 | | | |
| 0x070 | IRQ23MON[B,H,W] | | | |
| | -------- -------- -------0 00000000 | | | |
| 0x074 | IRQ24MON[B,H,W] | | | |
| | -------- -------- -------- --00-000 | | | |
| 0x078 | IRQ25MON[B,H,W] | | | |
| | -------- -------- -------- ---00000 | | | |
| 0x07C | IRQ26MON[B,H,W] | | | |
| | -------- -------- -------- ---00000 | | | |
| 0x080 | IRQ27MON[B,H,W] | | | |
| | -------- -------- -------- --000000 | | | |
| 0x084 | IRQ28MON[B,H,W] | | | |
| | -------- ------00 00000000 00000000 | | | |
| 0x088 | IRQ29MON[B,H,W] | | | |
| | -------- -------- ----0000 00000000 | | | |
| 0x08C | IRQ30MON[B,H,W] | | | |
| | -------- ------00 00000000 00000000 | | | |
| 0x090 | IRQ31MON[B,H,W] | | | |
| | ----0--- -------- 00000000 00000000 | | | |
| 0x094 - 0x20C | - | - | - | - |
| 0x210 | RCINTSEL0[B,H,W]<br>00000000 00000000 00000000 00000000 | | | |
| 0x214 | RCINTSEL1[B,H,W]<br>00000000 00000000 00000000 00000000 | | | |
| 0x218 - 0xFFC | - | - | - | - |

**LCDC Base_Address : 0x4003_2000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x00 | - | LCDCC3[B,H,W] | LCDCC2[B,H,W] | LCDCC1[B,H,W] |
| | | 0011111- | --010100 | -00000-- |
| 0x04 | LCDC_PSR[B,H,W] | | | |
| | -------- -0000000 00000000 00000000 | | | |
| 0x08 | LCDC_COMEN[B,H,W] | | | |
| | -------- -------- -------- 00000000 | | | |
| 0x0C | LCDC_SEGEN1[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x10 | LCDC_SEGEN2[B,H,W] | | | |
| | -------- -------- -------- 00000000 | | | |
| 0x14 | - | - | LCDC_BLINK[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x18 | - | - | - | - |
| 0x1C | LCDRAM03[B,H,W] | LCDRAM02[B,H,W] | LCDRAM01[B,H,W] | LCDRAM00[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x20 | LCDRAM07[B,H,W] | LCDRAM06[B,H,W] | LCDRAM05[B,H,W] | LCDRAM04[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x24 | LCDRAM11[B,H,W] | LCDRAM10[B,H,W] | LCDRAM09[B,H,W] | LCDRAM08[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x28 | LCDRAM15[B,H,W] | LCDRAM14[B,H,W] | LCDRAM13[B,H,W] | LCDRAM12[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x2C | LCDRAM19[B,H,W] | LCDRAM18[B,H,W] | LCDRAM17[B,H,W] | LCDRAM16[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x30 | LCDRAM23[B,H,W] | LCDRAM22[B,H,W] | LCDRAM21[B,H,W] | LCDRAM20[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x34 | LCDRAM27[B,H,W] | LCDRAM26[B,H,W] | LCDRAM25[B,H,W] | LCDRAM24[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x38 | LCDRAM31[B,H,W] | LCDRAM30[B,H,W] | LCDRAM29[B,H,W] | LCDRAM28[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x3C | LCDRAM35[B,H,W] | LCDRAM34[B,H,W] | LCDRAM33[B,H,W] | LCDRAM32[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x40 | LCDRAM39[B,H,W] | LCDRAM38[B,H,W] | LCDRAM37[B,H,W] | LCDRAM36[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x44 - 0xFC | - | - | - | - |

**GPIO  Base_Address : 0x4003_3000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | PFR0[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 1010 | | | |
| 0x004 | PFR1[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x008 | PFR2[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x00C | PFR3[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x010 | PFR4[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x014 | PFR5[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x018 | PFR6[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x01C | PFR7[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x020 | PFR8[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x024 | PFR9[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x028 | PFRA[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x02C | PFRB[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x030 | PFRC[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x034 | PFRD[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x038 | PFRE[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x03C | PFRF[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x040 - 0x0FC | - | - | - | - |
| 0x100 | PCR0[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 1010 | | | |
| 0x104 | PCR1[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x108 | PCR2[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x10C | PCR3[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x110 | PCR4[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x114 | PCR5[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x118 | PCR6[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x11C | PCR7[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x120 | - | - | - | - |
| 0x124 | PCR9[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x128 | PCRA[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x12C | PCRB[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x130 | PCRC[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x134 | PCRD[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x138 | PCRE[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x13C | PCRF[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x140 - 0x1FC | - | - | - | - |
| 0x200 | DDR0[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x204 | DDR1[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x208 | DDR2[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x20C | DDR3[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x210 | DDR4[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x214 | DDR5[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x218 | DDR6[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x21C | DDR7[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x220 | DDR8[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x224 | DDR9[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x228 | DDRA[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x22C | DDRB[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x230 | DDRC[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x234 | DDRD[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x238 | DDRE[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x23C | DDRF[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x240 - 0x2FC | - | - | - | - |
| 0x300 | PDIR0[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x304 | PDIR1[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x308 | PDIR2[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x30C | PDIR3[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x310 | PDIR4[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x314 | PDIR5[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x318 | PDIR6[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x31C | PDIR7[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x320 | PDIR8[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x324 | PDIR9[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x328 | PDIRA[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x32C | PDIRB[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x330 | PDIRC[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x334 | PDIRD[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x338 | PDIRE[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x33C | PDIRF[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x340 - 0x3FC | - | - | - | - |
| 0x400 | PDOR0[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x404 | PDOR1[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x408 | PDOR2[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x40C | PDOR3[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x410 | PDOR4[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x414 | PDOR5[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x418 | PDOR6[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x41C | PDOR7[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x420 | PDOR8[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x424 | PDOR9[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x428 | PDORA[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x42C | PDORB[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x430 | PDORC[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x434 | PDORD[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x438 | PDORE[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x43C | PDORF[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x440 - 0x4FC | - | - | - | - |
| 0x500 | ADE[B,H,W] | | | |
| | 1111 1111 1111 1111 1111 1111 1111 1111 | | | |
| 0x504 - 0x57C | - | - | - | - |
| 0x580 | SPSR[B,H,W] | | | |
| | ---- ---- ---- ---- ---- ---- ---- 0101 | | | |
| 0x584 - 0x5FC | - | - | - | - |
| 0x600 | EPFR00[B,H,W] | | | |
| | ---- ---- ---- ---1 ---- ---- 0000 -000 | | | |
| 0x604 | EPFR01[B,H,W] | | | |
| | 0000 0000 0000 0000 ---0 0000 0000 0000 | | | |
| 0x608 | EPFR02[B,H,W] | | | |
| | 0000 0000 0000 0000 ---0 0000 0000 0000 | | | |
| 0x60C | EPFR03[B,H,W] | | | |
| | 0000 0000 0000 0000 ---0 0000 0000 0000 | | | |
| 0x610 | EPFR04[B,H,W] | | | |
| | --00 0000 --00 00-- --00 0000 -000 00-- | | | |
| 0x614 | EPFR05[B,H,W] | | | |
| | --00 0000 --00 00-- --00 0000 --00 00-- | | | |
| 0x618 | EPFR06[B,H,W] | | | |
| | 0000 0000 0000 0000 0000 0000 0000 0000 | | | |
| 0x61C | EPFR07[B,H,W] | | | |
| | ---- 0000 0000 0000 0000 0000 0000 ---- | | | |
| 0x620 | EPFR08[B,H,W] | | | |
| | ---- 0000 0000 0000 0000 0000 0000 0000 | | | |
| 0x624 | EPFR09[B,H,W] | | | |
| | 0000 0000 0000 0000 0000 0000 0000 0000 | | | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x628 - 0x62C | - | - | - | - |
| 0x630 | EPFR12[B,H,W] | | | |
| | --00 0000 --00 00-- --00 0000 --00 00-- | | | |
| 0x634 | EPFR13[B,H,W] | | | |
| | --00 0000 --00 00-- --00 0000 --00 00-- | | | |
| 0x638 | EPFR14[B,H,W] | | | |
| | ---- ---- ---- ---- ---- ---- --00 0000 | | | |
| 0x63C | EPFR15[B,H,W] | | | |
| | 0000 0000 0000 0000 0000 0000 0000 0000 | | | |
| 0x640 | EPFR16[B,H,W] | | | |
| | ---- 0000 0000 0000 0000 0000 0000 ---- | | | |
| 0x644 | EPFR17[B,H,W] | | | |
| | ---- 0000 0000 0000 0000 0000 0000 ---- | | | |
| 0x648 | EPFR18[B,H,W] | | | |
| | ---- ---- ---- ---- ---- ---- ---- 0000 | | | |
| 0x64C - 0x650 | - | - | - | - |
| 0x654 | EPFR21[B,H,W] | | | |
| | ---- ---- ---- ---- ---- ---- ---- -000 | | | |
| 0x658 | EPFR22[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 ---- 0000 ---- | | | |
| 0x65C - 0x6FC | - | - | - | - |
| 0x700 | PZR0[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x704 | PZR1[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x708 | PZR2[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x70C | PZR3[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x710 | PZR4[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x714 | PZR5[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x718 | PZR6[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x71C | PZR7[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x720 | PZR8[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x724 | PZR9[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x728 | PZRA[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x72C | PZRB[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x730 | PZRC[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x734 | PZRD[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x738 | PZRE[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x73C | PZRF[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x740 - 0x7FC | - | - | - | - |
| 0x800 | * | | | |
| 0x804 | * | | | |
| 0x808 - 0x8FC | - | - | - | - |
| 0x900 | FPOER0[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x904 | FPOER1[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x908 | FPOER2[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x90C | FPOER3[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x910 | FPOER4[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x914 | FPOER5[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x918 | FPOER6[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x91C | FPOER7[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x920 | FPOER8[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x924 | FPOER9[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x928 | FPOERA[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x92C | FPOERB[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x930 | FPOERC[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x934 | FPOERD[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x938 | FPOERE[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x93C | FPOERF[B,H,W] | | | |
| | ---- ---- ---- ---- 0000 0000 0000 0000 | | | |
| 0x940 - 0xFFC | - | - | - | - |

**HDMI-CEC/Remote Control Receiver ch.0 Base_Address : 0x4003_4000**

**HDMI-CEC/Remote Control Receiver ch.1 Base_Address : 0x4003_4100**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x00 | - | - | - | TXCTRL[B,H,W]<br>--0000-0 |
| 0x04 | - | - | - | TXDATA[B,H,W]<br>00000000 |
| 0x08 | - | - | - | TXSTS[B,H,W]<br>--00---0 |
| 0x0C | - | - | - | SFREE[B,H,W]<br>----0000 |
| 0x10 - 0x3F | - | - | - | - |
| 0x40 | - | - | RCCR[B,H,W]<br>0---0000 | RCST[B,H,W]<br>00000000 |
| 0x44 | - | - | RCSHW[B,H,W]<br>00000000 | RCDAHW[B,H,W]<br>00000000 |
| 0x48 | - | - | RCDBHW[B,H,W]<br>00000000 | - |
| 0x4C | - | - | RCADR1[B,H,W]<br>---00000 | RCADR2[B,H,W]<br>---00000 |
| 0x50 | - | - | RCDTHH[B,H,W]<br>00000000 | RCDTHL[B,H,W]<br>00000000 |
| 0x54 | - | - | RCDTLH[B,H,W]<br>00000000 | RCDTLL[B,H,W]<br>00000000 |
| 0x58 | - | - | RCCKD[H,W]<br>---00000 00000000 | |
| 0x5C | - | - | RCRC[B,H,W]<br>---0---0 | RCRHW[B,H,W]<br>00000000 |
| 0x60 | - | - | RCLE[B,H,W]<br>00000-00 | - |
| 0x64 | - | - | RCLELW[B,H,W]<br>00000000 | RCLESW[B,H,W]<br>00000000 |
| 0x68 - 0xFC | - | - | - | - |

**LVD Base_Address : 0x4003_5000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | LVD_CTL[B,H,W] | |
| | | | 100000-- 000011-- | |
| 0x004 | - | - | - | LVD_STR[B,H,W] |
| | | | | 0------- |
| 0x008 | - | - | - | LVD_CLR[B,H,W] |
| | | | | 1------- |
| 0x00C | LVD_RLR[W] | | | |
| | 00000000 00000000 00000000 00000001 | | | |
| 0x010 | - | - | - | LVD_STR2 |
| | | | | 01------ |
| 0x014 - 0x0FC | - | - | - | - |

**DS Mode Base_Address : 0x4003_5100**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | - | REG_CTL[B,H,W] |
| | | | | -------0 |
| 0x004 | - | - | - | RCK_CTL[B,H,W] |
| | | | | ------01 |
| 0x008 - 0x6FC | - | - | - | - |
| 0x700 | - | - | - | PMD_CTL[B,H,W] |
| | | | | -------0 |
| 0x704 | - | - | - | WRFSR[B,H,W] |
| | | | | ------00 |
| 0x708 | - | - | WIFSR[B,H,W] | |
| | | | ------00 00000000 | |
| 0x70C | - | - | WIER[B,H,W] | |
| | | | ------00 00000-00 | |
| 0x710 | - | - | - | WILVR[B,H,W] |
| | | | | -----000 |
| 0x714 | - | - | - | DSRAMR[B,H,W] |
| | | | | ------00 |
| 0x718 - 0x7FC | - | - | - | - |
| 0x800 | BUR04[B,H,W] | BUR03[B,H,W] | BUR02[B,H,W] | BUR01[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x804 | BUR08[B,H,W] | BUR07[B,H,W] | BUR06[B,H,W] | BUR05[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x808 | BUR012[B,H,W] | BUR11[B,H,W] | BUR10[B,H,W] | BUR09[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x80C | BUR16[B,H,W] | BUR15[B,H,W] | BUR14[B,H,W] | BUR13[B,H,W] |
| | 00000000 | 00000000 | 00000000 | 00000000 |
| 0x810 - 0xEFC | - | - | - | - |

**CAN Prescaler Base_Address : 0x4003_7000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | - | CANPRE[B,H,W] |
| | | | | ----1011 |
| 0x004 - 0xFFC | - | - | - | - |

**MFS**

**MFS ch.0 Base_Address : 0x4003_8000**

**MFS ch.1 Base_Address : 0x4003_8100**

**MFS ch.2 Base_Address : 0x4003_8200**

**MFS ch.3 Base_Address : 0x4003_8300**

**MFS ch.4 Base_Address : 0x4003_8400**

**MFS ch.5 Base_Address : 0x4003_8500**

**MFS ch.6 Base_Address : 0x4003_8600**

**MFS ch.7 Base_Address : 0x4003_8700**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | SCR/IBCR[B,H,W] | SMR[B,H,W] |
| | | | 0--00000 | 000-00-0 |
| 0x004 | - | - | SSR[B,H,W] | ESCR/IBSR[B,H,W] |
| | | | 0-000011 | 00000000 |
| 0x008 | - | - | RDR/TDR[H,W] | |
| | | | 00000000 00000000 | |
| 0x00C | - | - | BGR1[B,H,W] | BGR0[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x010 | - | - | ISMK[B,H,W] | ISBA[B,H,W] |
| | | | -------- | -------- |
| 0x014 | - | - | FCR1[B,H,W] | FCR0[B,H,W] |
| | | | ---00100 | -0000000 |
| 0x018 | - | - | FBYTE2[B,H,W] | FBYTE1[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x01C | - | - | SCSTR1/ EIBCR[B,H,W] | SCSTR0/ NFCR[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x020 | - | - | SCSTR3[B,H,W] | SCSTR2[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x024 | - | - | SACSR[B,H,W] | |
| | | | --000--0 00-00000 | |
| 0x028 | - | - | STMR[B,H,W] | |
| | | | 00000000 00000000 | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x02C | - | - | STMCR[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x030 | - | - | SCSCR[B,H,W] | |
| | | | 00000000 00100000 | |
| 0x034 | - | - | SCSFR1[B,H,W] 10000000 | SCSFR0[B,H,W] 10000000 |
| 0x038 | - | - | - | SCSFR2[B,H,W] 10000000 |
| 0x03C | - | - | TBYTE1[B,H,W] | TBYTE0[B,H,W] |
| | | | 00000000 | 00000000 |
| 0x040 | - | - | TBYTE3[B,H,W] 00000000 | TBYTE2[B,H,W] 00000000 |
| 0x044 - 0x0FC | - | - | - | - |

**CRC Base_Address : 0x4003_9000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | - | CRCCR[B,H,W] |
| | | | | -0000000 |
| 0x004 | CRCINIT[B,H,W] | | | |
| | 11111111 11111111 11111111 11111111 | | | |
| 0x008 | CRCIN[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x00C | CRCR[B,H,W] | | | |
| | 11111111 11111111 11111111 11111111 | | | |

**Watch Counter Base_Address : 0x4003_A000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | WCCR[B,H,W] | WCRL[B,H,W] | WCRD[B,H,W] |
| | | 00--0000 | --000000 | --000000 |
| 0x004 - 0x00C | - | - | - | - |
| 0x010 | - | - | CLK_SEL[B,H,W] | |
| | | | -----000 ------00 | |
| 0x014 | - | - | - | CLK_EN[B,H,W] |
| | | | | ------00 |
| 0x018 - 0xFFC | - | - | - | - |

**RTC  Base_Address : 0x4003_B000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | WTCR1[B,H,W] | | | |
| | 00000000 00000000 ---00000 -00000-0 | | | |
| 0x004 | WTCR2[B,H,W] | | | |
| | -------- -------- -----000 -------0 | | | |
| 0x008 | WTBR[B,H,W] | | | |
| | -------- 00000000 00000000 00000000 | | | |
| 0x00C | WTDR[B,H,W] | WTHR[B,H,W] | WTMIR[B,H,W] | WTSR[B,H,W] |
| | --000000 | --000000 | -0000000 | -0000000 |
| 0x010 | - | WTYR[B,H,W] | WTMOR[B,H,W] | WTDW[B,H,W] |
| | | 00000000 | ---00000 | -----000 |
| 0x014 | ALDR[B,H,W] | ALHR[B,H,W] | ALMIR[B,H,W] | - |
| | --000000 | --000000 | -0000000 | |
| 0x018 | - | ALYR[B,H,W] | ALMOR[B,H,W] | - |
| | | 00000000 | ---00000 | |
| 0x01C | WTTR[B,H,W] | | | |
| | -------- ------00 00000000 00000000 | | | |
| 0x020 | - | - | WTCLKM[B,H,W] | WTCLKS[B,H,W] |
| | | | ------00 | -------0 |
| 0x024 | - | WTCALEN[B,H,W] | WTCAL[B,H,W] | |
| | | -------0 | ------00 00000000 | |
| 0x028 | - | - | WTDIVEN[B,H,W] | WTDIV[B,H,W] |
| | | | ------00 | ----0000 |
| 0x02C | - | - | - | WTCALPRD[B,H,W] |
| | | | | --010011 |
| 0x030 | - | - | - | WTCOSEL[B,H,W] |
| | | | | -------0 |
| 0x034 - 0xFFC | - | - | - | - |

**Low-speed CR Prescaler  Base_Address : 0x4003_C000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | - | LCR_PRSLD[B,H,W] |
| | | | | --000000 |
| 0x000 - 0x0FC | - | - | - | - |

**Peripheral Clock Gating Base_Address : 0x4003_C100**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | CKEN0[B,H,W] | | | |
| | ---1---1 ----1111 11111111 11111111 | | | |
| 0x004 | MRST0[B,H,W] | | | |
| | -------0 ----0000 00000000 00000000 | | | |
| 0x008 - 0x00C | - | - | - | - |
| 0x010 | CKEN1[B,H,W] | | | |
| | -------- ----1111 ----1111 ----1111 | | | |
| 0x014 | MRST1[B,H,W] | | | |
| | -------- ----0000 ----0000 ----0000 | | | |
| 0x018 - 0x01C | - | - | - | - |
| 0x020 | CKEN2[B,H,W] | | | |
| | -------- -------- -------- --**----<br>Products with CAN : *="1"<br>Products without CAN : *="0" | | | |
| 0x024 | MRST2[B,H,W] | | | |
| | -------- -------- -------- --00---- | | | |
| 0x028 - 0x0FC | - | - | - | - |

**DMAC Base_Address : 0x4006_0000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x0000 | DMACR[B,H,W] | | | |
| | 00-00000 -------- -------- -------- | | | |
| 0x0010 | DMACA0[B,H,W] | | | |
| | 00000000 0---0000 00000000 00000000 | | | |
| 0x0014 | DMACB0[B,H,W] | | | |
| | --000000 00000000 00000000 -------0 | | | |
| 0x0018 | DMACSA0[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x001C | DMACDA0[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x0020 | DMACA1[B,H,W] | | | |
| | 00000000 0---0000 00000000 00000000 | | | |
| 0x0024 | DMACB1[B,H,W] | | | |
| | --000000 00000000 00000000 -------0 | | | |
| 0x0028 | DMACSA1[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x002C | DMACDA1[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x0030 | DMACA2[B,H,W] | | | |
| | 00000000 0---0000 00000000 00000000 | | | |
| 0x0034 | DMACB2[B,H,W] | | | |
| | --000000 00000000 00000000 -------0 | | | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x0038 | DMACSA2[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x003C | DMACDA2[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x0040 | DMACA3[B,H,W] | | | |
| | 00000000 0---0000 00000000 00000000 | | | |
| 0x0044 | DMACB3[B,H,W] | | | |
| | --000000 00000000 00000000 -------0 | | | |
| 0x0048 | DMACSA3[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x004C | DMACDA3[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x0050 | DMACA4[B,H,W] | | | |
| | 00000000 0---0000 00000000 00000000 | | | |
| 0x0054 | DMACB4[B,H,W] | | | |
| | --000000 00000000 00000000 -------0 | | | |
| 0x0058 | DMACSA4[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x005C | DMACDA4[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x0060 | DMACA5[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x0064 | DMACB5[B,H,W] | | | |
| | --000000 00000000 00000000 -------0 | | | |
| 0x0068 | DMACSA5[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x006C | DMACDA5[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x0070 | DMACA6[B,H,W] | | | |
| | 00000000 0---0000 00000000 00000000 | | | |
| 0x0074 | DMACB6[B,H,W] | | | |
| | --000000 00000000 00000000 -------0 | | | |
| 0x0078 | DMACSA6[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x007C | DMACDA6[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x0080 | DMACA7[B,H,W] | | | |
| | 00000000 0---0000 00000000 00000000 | | | |
| 0x0084 | DMACB7[B,H,W] | | | |
| | --000000 00000000 00000000 -------0 | | | |
| 0x0088 | DMACSA7[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x008C | DMACDA7[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x0090 - 0x00FC | - | - | - | - |

**CAN ch.0 Base_Address : 0x4006_2000**

**CAN ch.1 Base_Address : 0x4006_3000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x0000 | STATR[B,H,W] | | CTRLR[B,H,W] | |
| | -------- 00000000 | | -------- 000-0001 | |
| 0x0004 | BTR[B,H,W] | | ERRCNT[B,H,W] | |
| | -0100011 00000001 | | 00000000 00000000 | |
| 0x0008 | TESTR[B,H,W] | | INTR[B,H,W] | |
| | -------- X00000-- | | 00000000 00000000 | |
| 0x000C | - | - | BRPER[B,H,W] | |
| | | | -------- ----0000 | |
| 0x0010 | IF1CMSK[B,H,W] | | IF1CREQ[B,H,W] | |
| | -------- 00000000 | | 0------- 00000001 | |
| 0x0014 | IF1MSK2[B,H,W] | | IF1MSK1[B,H,W] | |
| | 11-11111 11111111 | | 11111111 11111111 | |
| 0x0018 | IF1ARB2[B,H,W] | | IF1ARB1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x001C | - | - | IF1MCTR[B,H,W] | |
| | | | 00000000 0---0000 | |
| 0x0020 | IF1DTA2[B,H,W] | | IF1DTA1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0024 | IF1DTB2[B,H,W] | | IF1DTB1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0028 - 0x002F | - | - | - | - |
| 0x0030 | IF1DTA1[B,H,W] | | IF1DTA2[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0034 | IF1DTB1[B,H,W] | | IF1DTB2[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0038 - 0x003C | - | - | - | - |
| 0x0040 | IF2CMSK[B,H,W] | | IF2CREQ[B,H,W] | |
| | -------- 00000000 | | 0------- 00000001 | |
| 0x0044 | IF2MSK2[B,H,W] | | IF2MSK1[B,H,W] | |
| | 11-11111 11111111 | | 11111111 11111111 | |
| 0x0048 | IF2ARB2[B,H,W] | | IF2ARB1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x004C | - | - | IF2MCTR[B,H,W] | |
| | | | 00000000 0---0000 | |
| 0x0050 | IF2DTA2[B,H,W] | | IF2DTA1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0054 | IF2DTB2[B,H,W] | | IF2DTB1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0058 - 0x005C | - | - | - | - |
| 0x0060 | IF2DTA1[B,H,W] | | IF2DTA2[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0064 | IF2DTB1[B,H,W] | | IF2DTB2[B,H,W] | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0068 - 0x007C | - | - | - | - |
| 0x0080 | TREQR2[B,H,W] | | TREQR1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0084 - 0x008F | - | - | - | - |
| 0x0090 | NEWDT2[B,H,W] | | NEWDT1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x0094 - 0x009F | - | - | - | - |
| 0x00A0 | INTPND2[B,H,W] | | INTPND1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x00A4 - 0x00AF | - | - | - | - |
| 0x00B0 | MSGVAL2[B,H,W] | | MSGVAL1[B,H,W] | |
| | 00000000 00000000 | | 00000000 00000000 | |
| 0x00B4 - 0x0FFC | - | - | - | - |

**MTB_DWT Base_Address : 0xF000_1000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | CMP_ADDR_START[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x004 | CMP_DATA_START[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x008 | CMP_MASK_START[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x00C | - | - | - | - |
| 0x010 | CMP_ADDR_STOP[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x014 | CMP_DATA_STOP[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x018 | CMP_MASK_STOP[B,H,W] | | | |
| | 00000000 00000000 00000000 00000000 | | | |
| 0x01C | - | - | - | - |
| 0x020 | - | - | - | FCT[B,H,W] |
| | | | | 00000000 |
| 0x024 - 0xFCC | - | - | - | - |
| 0xFD0 | PID4[B,H,W] | | | |
| | XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFD4 | PID5[B,H,W] | | | |
| | XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFD8 | PID6[B,H,W] | | | |
| | XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0xFDC | PID7[B,H,W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFE0 | PID0[B,H,W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFE4 | PID1[B,H,W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFE8 | PID2[B,H,W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFEC | PID3[B,H,W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFF0 | CID0[B,H,W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFF4 | CID1[B,H,W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFF8 | CID2[B,H,W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |
| 0xFFC | CID3[B,H,W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | |

**Fast GPIO Base_Address : 0xF800_0000**

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x000 | - | - | FPDIR0[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x004 | - | - | FPDIR1[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x008 | - | - | FPDIR2[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x00C | - | - | FPDIR3[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x010 | - | - | FPDIR4[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x014 | - | - | FPDIR5[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x018 | - | - | FPDIR6[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x01C | - | - | FPDIR7[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x020 | - | - | FPDIR8[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x024 | - | - | FPDIR9[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x028 | - | - | FPDIRA[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x02C | - | - | FPDIRB[B,H,W]<br>XXXXXXXX XXXXXXXX | |
| 0x030 | - | - | FPDIRC[B,H,W]<br>XXXXXXXX XXXXXXXX | |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x034 | - | - | FPDIRD[B,H,W] | |
| | | | XXXXXXXX XXXXXXXX | |
| 0x038 | - | - | FPDIRE[B,H,W] | |
| | | | XXXXXXXX XXXXXXXX | |
| 0x03C | - | - | FPDIRF[B,H,W] | |
| | | | XXXXXXXX XXXXXXXX | |
| 0x040 | - | - | FPDOR0[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x044 | - | - | FPDOR1[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x048 | - | - | FPDOR2[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x04C | - | - | FPDOR3[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x050 | - | - | FPDOR4[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x054 | - | - | FPDOR5[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x058 | - | - | FPDOR6[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x05C | - | - | FPDOR7[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x060 | - | - | FPDOR8[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x064 | - | - | FPDOR9[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x068 | - | - | FPDORA[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x06C | - | - | FPDORB[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x070 | - | - | FPDORC[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x074 | - | - | FPDORD[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x078 | - | - | FPDORE[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x07C | - | - | FPDORF[B,H,W] | |
| | | | 00000000 00000000 | |
| 0x080 | - | - | - | M_FPDIR0[B,H,W] |
| | | | | XXXXXXXX |
| 0x084 | - | - | - | M_FPDIR1[B,H,W] |
| | | | | XXXXXXXX |
| 0x088 | - | - | - | M_FPDIR2[B,H,W] |
| | | | | XXXXXXXX |
| 0x08C | - | - | - | M_FPDIR3[B,H,W] |
| | | | | XXXXXXXX |
| 0x090 | - | - | - | M_FPDIR4[B,H,W] |
| | | | | XXXXXXXX |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x094 | - | - | - | M_FPDIR5[B,H,W]<br>XXXXXXXX |
| 0x098 | - | - | - | M_FPDIR6[B,H,W]<br>XXXXXXXX |
| 0x09C | - | - | - | M_FPDIR7[B,H,W]<br>XXXXXXXX |
| 0x0A0 | - | - | - | M_FPDIR8[B,H,W]<br>XXXXXXXX |
| 0x0A4 | - | - | - | M_FPDIR9[B,H,W]<br>XXXXXXXX |
| 0x0A8 | - | - | - | M_FPDIRA[B,H,W]<br>XXXXXXXX |
| 0x0AC | - | - | - | M_FPDIRB[B,H,W]<br>XXXXXXXX |
| 0x0B0 | - | - | - | M_FPDIRC[B,H,W]<br>XXXXXXXX |
| 0x0B4 | - | - | - | M_FPDIRD[B,H,W]<br>XXXXXXXX |
| 0x0B8 | - | - | - | M_FPDIRE[B,H,W]<br>XXXXXXXX |
| 0x0BC | - | - | - | M_FPDIRF[B,H,W]<br>XXXXXXXX |
| 0x0C0 | - | - | - | M_FPDOR0[B,H,W]<br>00000000 |
| 0x0C4 | - | - | - | M_FPDOR1[B,H,W]<br>00000000 |
| 0x0C8 | - | - | - | M_FPDOR2[B,H,W]<br>00000000 |
| 0x0CC | - | - | - | M_FPDOR3[B,H,W]<br>00000000 |
| 0x0D0 | - | - | - | M_FPDOR4[B,H,W]<br>00000000 |
| 0x0D4 | - | - | - | M_FPDOR5[B,H,W]<br>00000000 |
| 0x0D8 | - | - | - | M_FPDOR6[B,H,W]<br>00000000 |
| 0x0DC | - | - | - | M_FPDOR7[B,H,W]<br>00000000 |
| 0x0E0 | - | - | - | M_FPDOR8[B,H,W]<br>00000000 |
| 0x0E4 | - | - | - | M_FPDOR9[B,H,W]<br>00000000 |
| 0x0E8 | - | - | - | M_FPDORA[B,H,W]<br>00000000 |
| 0x0EC | - | - | - | M_FPDORB[B,H,W]<br>00000000 |
| 0x0F0 | - | - | - | M_FPDORC[B,H,W]<br>00000000 |

| Base_Address | Register | | | |
|---|---|---|---|---|
| + Address | +3 | +2 | +1 | +0 |
| 0x0F4 | - | - | - | M_FPDORD[B,H,W] |
| | | | | 00000000 |
| 0x0F8 | - | - | - | M_FPDORE[B,H,W] |
| | | | | 00000000 |
| 0x0FC | - | - | - | M_FPDORF[B,H,W] |
| | | | | 00000000 |
| 0x100 - 0xFFC | - | - | - | - |

# B. List of Notes

**This section explains notes for each function.**

1. Notes when High-speed CR Is Used for Master Clock

CODE: 9APRECAUTION-E01.0

# 1. Notes when High-speed CR Is Used for Master Clock

This section explains notes when the high-speed CR is used for the master clock.

The frequency of the high-speed CR varies depending on the temperature and/or the power supply voltage. The following table shows notes on each function macro when the high-speed CR is used for the master clock.

Furthermore, pay attention to notes when the high-speed CR is used as an input clock of the PLL and the master clock is selected for PLL.

■ Notes on each macro

| Macro | Function/mode | Notes |
|---|---|---|
| Internal Bus Clock | HCLK/FCLK/PCLK0/PCLK1 | The maximum frequency of the high-speed CR shall not exceed the upper limit of the internal operation clock frequency specified in the "Data Sheet" of the product used. |
| Timer | Multi-function Timer<br>Base Timer<br>Watch Timer<br>Dual Timer<br>Watch Dog Timer<br>Quadrature | The frequency variation of the high-speed CR should be considered for the timer count value of each macro. |
| A/D Converter | Sampling Time<br>Compare Tim | Considering the frequency variation of the high-speed CR, the sampling time and the compare time of the A/D converter shall satisfy the specification specified in the "Data Sheet" of the product used. |
| CAN | - | As the frequency accuracy does not meet the required specification, these macros cannot be used when the high-speed CR is used for the master clock. |
| Multi Function Serial Interface | UART | Even if the frequency of the high-speed CR is the minimum or the maximum value, the baud rate error should be considered.<br>The baud rate error shall not exceed the limit. |
| | CSIO | The frequency variation of the high-speed CR should be considered for the communication of each macro. |
| | I2C | |
| | LIN | As the required frequency accuracy cannot be met, this function cannot be used as master.<br>As slave, this function can be used.<br>As a slave, the specified baud rate has more error at the maximum/minimum frequency of high-speed clock. So, if the error limit of the baud rate is exceeded, this function cannot be used. |

# MAJOR CHANGES IN THIS EDITION

| Page | Section | Change Results |
|---|---|---|
| Rev. 1.0 | | |
| - | - | Initial release |

MN710-00001-1v0-E

**Spansion・Controller Manual**

FM0+ Family
32-BIT MICROCONTROLLER
PERIPHERAL MANUAL

January 2014 Rev. 1.0

Published : Spansion Inc.
Edited : Marketing Communication Dept.

**Colophon**

The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for any use that includes fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for any use where chance of failure is intolerable (i.e., submersible repeater and artificial satellite). Please note that Spansion will not be liable to you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products. Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions. If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the US Export Administration Regulations or the applicable laws of any other country, the prior authorization by the respective government entity will be required for export of those products.

**Trademarks and Notice**

The contents of this document are subject to change without notice. This document may contain information on a Spansion product under development by Spansion. Spansion reserves the right to change or discontinue work on any product without notice. The information in this document is provided as is without warranty or guarantee of any kind as to its accuracy, completeness, operability, fitness for particular purpose, merchantability, non-infringement of third-party rights, or any other warranty, express, implied, or statutory. Spansion assumes no liability for any damages of any kind arising out of the use of the information in this document.