# CMS32M67xx User Manual

**Ultra-low power 32-bit microcontrollers based on ARM® Cortex®-M0+**

**V1.0.0**

Please note the following CMS IP policy

* China Micro Semicon Co., Ltd. (hereinafter referred to as the Company) has applied for patents and holds absolute legal rights and interests. The patent rights associated with the Company's MCUs or other products have not been authorized for use, and any company, organization, or individual who infringes the Company's patent rights through improper means will be subject to all possible legal actions taken by the Company to curb the infringement and to recover any damages suffered by the Company as a result of the infringement or any illegal benefits obtained by the infringer.

* The name and logo of Cmsemicon are registered trademarks of the Company.

* The Company reserves the right to further explain the reliability, functionality and design improvements of the products in the data sheet. However, the Company is not responsible for the use of the Specification Contents. The applications mentioned herein are for illustrative purposes only and the Company does not warrant and does not represent that these applications can be applied without further modification, nor does it recommend that its products be used in places that may cause harm to persons due to malfunction or other reasons. The Company's products are not authorized for use as critical components in lifesaving, life-sustaining devices or systems. The Company reserves the right to modify the products without prior notice. For the latest information, please visit the official website at www.mcu.com.cn.

# Documentation Instructions

This manual is the technical reference manual for the CMS32M67xx controller product. The technical reference manual provides application notes on how to use this series of products. It contains detailed information on the structure, functional descriptions, operating modes, and register configurations of each functional module, with a dedicated chapter for each functional module.

The technical reference manual provides descriptions of all the functional modules for this series of products. For detailed information on the specific features of a particular model (i.e., the functionality it supports), please refer to the corresponding datasheet.

The data sheet information is as follows:

CMS32M67xx: CMS32M67xx Datasheet_vx.x.x.pdf

Typically, in the early stages of chip selection, the datasheet should be reviewed first to assess whether the product meets the functional requirements of the design. Once the required product is preliminarily selected, the technical reference manual should be consulted to confirm whether the operating modes of the functional modules meet the requirements. When the selection process moves into the programming and design phase, the technical reference manual should be studied in detail to understand the specific implementation methods and register configurations of each function. For hardware design, the datasheet can be referenced for information on voltage, current, drive capability, pin assignments, and other details.

For detailed information regarding the Cortex-M0+ core, SysTick timer, and NVIC (Nested Vector Interrupt Controller), please refer to the corresponding ARM documentation.

# Table of Contents

# Chapter 1 CPU

## 1.1 Overview

This chapter provides a brief introduction to the features and debugging features of the ARM Cortex-M0+ core. For details, please refer to the ARM related documentation.

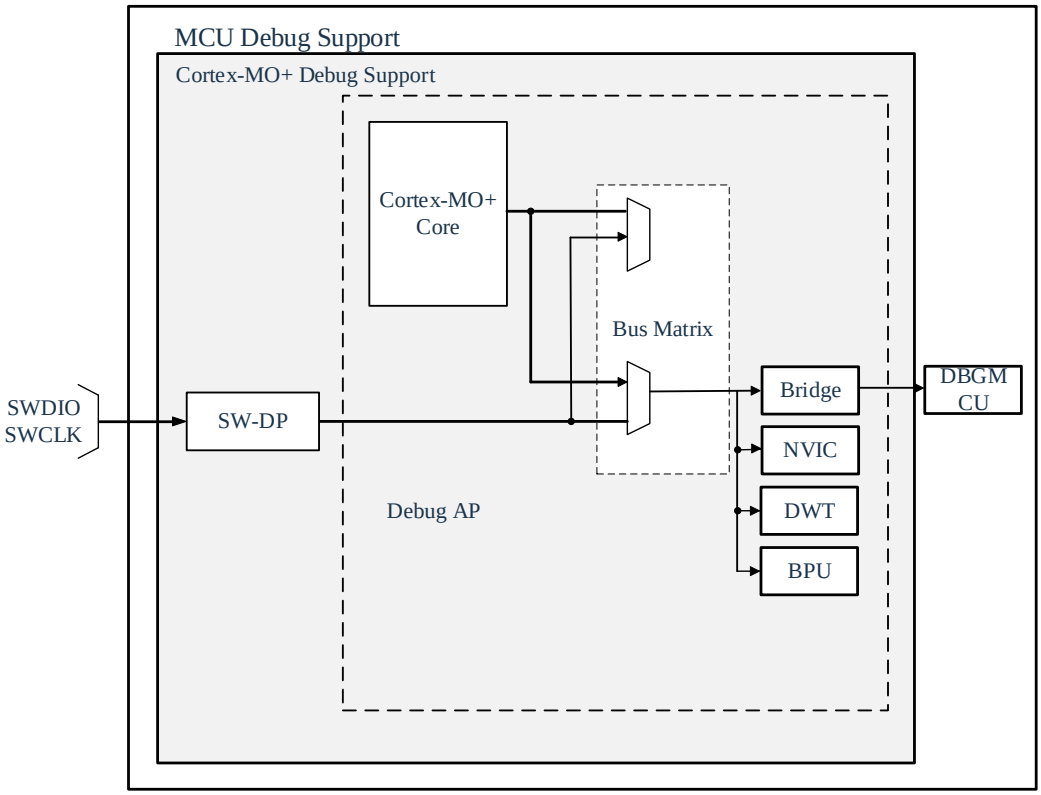
## 1.2 Cortex-M0+ Core Features

- ARM Cortex-M0+ processor is a 32-bit RISC core with a 2-stage pipeline that supports privileged mode only.
- 1-cycle hardware multiplier
- Nested vector interrupt controller (NVIC)
  - 1 non-maskable interrupt (NMI)
  - Support 23 maskable interrupt requests (IRQ)
  - 4 interrupt priority levels
- System Timer (SysTick) is a 24-bit countdown timer with a choice of $F_{CLK}$ or $F_{IL}$ count clock
- Vector table offset register (VTOR)
  - The software can write VTOR to relocate the vector table start address to a different location.
  - The default value of this register is 0x0000_0000, the lower 8 bits are ignored for writing and zero for reading, which means the offset is 256 bytes aligned.

## 1.3 Debugging Features

- 2-wire SWD debug interface
- Support for pausing, resuming and single-step execution of programs
- Access to the processor’s core registers and special function registers
- 4 hardware breakpoints (BPU)
- Unlimited software breakpoints (BKPT instruction)
- 2 data observation points (DWT)
- Accessing memory while the core is executing

Figure 1-1 Debug block diagram of Cortex-M0+




Note: SWD does not work in deep sleep mode, please do debug operation in active and sleep mode.

## 1.4 SWD Interface Pins

The 2 GPIOs of this product can be used as SWD interface pins, which exist in all packages.

Table 1-1 SWD debug port pins

| SWD port name | Debugging function | Pin assignment |
|---|---|---|
| SWCLK | Serial clock | P03 |
| SWDIO | Serial data input/output | P02 |

When the SWD function is not used, the SWD can be disabled by setting the debug stop control register (DBGSTOPCR).

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:25 | - | Reserved | - |
| 24 | SWDIS | SWD debug interface status<br>0: Enable the SWD debug interface. P02/P03 cannot be used as GPIO (because ENO and DOUT of this IOBUF are controlled by the debugger at this time).<br>1: Disable the SWD debug interface. P02/P03 can be used as GPIO. | 0 |
| 23:2 | - | Reserved | 0x0 |
| 1 | FRZEN1 | When the debugger is connected and the CPU is in debug state (HALTED=1), the peripheral module of the communication system acts/stops Note 2<br>0: Peripheral acts<br>1: Peripheral stops | 0 |
| 0 | FRZEN0 | When the debugger is connected and the CPU is in debug state (HALTED=1), the timer system peripheral module acts/stops Note 1<br>0: Peripheral acts<br>1: Peripheral stops | 0 |

Note 1: The timer system peripheral module of this product includes: general-purpose timer unit (Timer4).

Note 2: The communication system peripheral module of this product includes: serial communication unit, serial IICA.

## 1.5 ARM Reference Documents

The built-in debugging feature in the Cortex®-M0+ core is part of the ARM® CoreSight design suite. For documentation, refer to:

- Cortex®-M0+ Technical Reference Manual (TRM)
- ARM® Debug Interface V5
- ARM® CoreSight Design Suite Version r1p1 Technical Reference Manual

# Chapter 2 Pin Functions (GPIO)

## 2.1 Port Functions

For detailed information, please refer to the datasheet of the corresponding product series.

## 2.2 Port Alternate Function

For detailed configuration of each alternate function, please refer to the datasheet of the corresponding product series. See the table below for details of the port alternate function.

Table 2-1 Port alternate function digital mapping table

| Function name | Input | Alternate function (PmnCFG) | | | |
|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 |
| P00 | HALL_IN0/RXD0/CCP0A_I | | TXD0 | CCP0A_O | ADC_TRIG |
| P01 | NRST/RXD0/CCP0B_I | | TXD0 | CCP0B_O | ADC_TRIG |
| P02 | CCP1A_I | | - | CCP1A_O | - |
| P03 | CCP1B_I | | - | CCP1B_O | ADC_TRIG |
| P04 | CCP1A_I | | C0_O | CCP1A_O | SCL |
| P05 | CCP1B_I | | - | CCP1B_O | SDA |
| P06 | CCP0A_I | | NSS | CCP0A_O | SCL |
| P07 | CCP0B_I | | SCLK | CCP0B_O | SDA |
| P10 | ADET | | MOSI | - | - |
| P11 | CCP1A_I | | MISO | CCP1A_O | C0_O |
| P12 | CCP1B_I | | - | CCP1B_O | - |
| P13 | BKIN | -- | - | - | - |
| P14 | - | | MISO | PCUBZ0 | SCL |
| P15 | RXD0 | | MOSI | TXD0 | SDA |
| P16 | BKIN/RXD0 | | SCLK | TXD0 | - |
| P17 | - | | NSS | - | ADC_TRIG |
| P20 | HALL_IN0 | | - | - | - |
| P21 | HALL_IN1 | | - | - | ADC_TRIG |
| P22 | HALL_IN2 | | - | - | ADC_TRIG |
| P23 | BKIN/RXD1/CCP0B_I | | TXD1 | CCP0B_O | C1_O |
| P24 | RXD1/CCP0A_I | | TXD1 | CCP0A_O | ADC_TRIG |
| P25 | RXD1/CCP1A_I | | TXD1 | CCP1A_O | C1_O |
| P30 | - | | - | - | - |
| P31 | - | - | - | - | - |
| P32 | - | | - | - | - |
| P33 | - | | - | - | - |
| P34 | RXD0 | | TXD0 | - | SDA |
| P35 | RXD0 | | TXD0 | - | SCL |
| P36 | - | | EPWM0 | - | - |
| P37 | - | | EPWM1 | - | - |
| P40 | - | | EPWM2 | - | - |
| P41 | - | | EPWM3 | - | - |
| P42 | - | | EPWM4 | - | - |

| | | | | | |
|---|---|---|---|---|---|
| P43 | - | | EPWM5 | - | - |
| P44 | - | | EPWM6 | - | - |
| P45 | - | | EPWM7 | - | - |
| P46 | - | | - | - | - |
| P47 | - | | - | - | - |
| P50 | - | | - | - | - |
| P51 | - | | - | - | - |
| P52 | CCP0B_I | | - | CCP0B_O | - |
| P53 | RXD1/ CCP1B_I | | TXD1 | CCP1B_O | - |
| P54 | RXD1 | | TXD1 | - | SCL |
| P55 | BKIN/RXD1 | | TXD1 | - | SDA |
| P56 | HALL_IN2/CCP1A_I | | - | CCP1A_O | |
| P57 | HALL_IN1/ CCP1B_I | | - | CCP1B_O | |

Note 1: This product requires users to configure PMC, PM, and other registers separately for the IO alternate function.

Note 2: When selecting the IIC function, the open-drain function is automatically enabled.

Note 3: Regarding the multiplexing function, it can be used for both input and output. Once the PmnCFG is selected, the input channel is automatically enabled.

Table 2-2 Analog function and special function pins

| Pin | Analog | | | Special function pin |
|---|---|---|---|---|
| | ADC | ACMP | PGA | |
| P00 | AN7 | C0P0 | - | - |
| P01 | - | - | - | NRST |
| P02 | - | - | - | SWDDAT |
| P03 | - | - | - | SWDCLK |
| P04 | AN8 | - | - | OSCOUT |
| P05 | AN9 | - | - | OSCIN |
| P06 | AN10 | - | - | - |
| P07 | AN11 | - | - | - |
| P10 | - | - | -- | - |
| P11 | AN12 | - | - | - |
| P12 | AN13 | - | - | DAC_O |
| P13 | AN14 | - | - | - |
| P14 | AN15 | - | - | - |
| P15 | - | - | - | - |
| P16 | - | - | - | - |
| P17 | - | - | - | - |
| P20 | AN16 | C1P0 | - | - |
| P21 | AN17 | C1P1 | - | - |
| P22 | AN18 | C1P2 | - | - |
| P23 | - | C1P3 | - | - |
| P24 | AN19 | CIN | - | - |
| P25 | - | - | - | - |
| P30 | - | - | PGA3_P | - |
| P31 | - | - | PGA3_N | - |
| P32 | - | - | PGA2_P | - |
| P33 | - | - | PGA2_N | - |
| P34 | AN20 | - | - | - |
| P35 | - | - | - | - |
| P36 | - | - | - | - |
| P37 | - | - | - | - |
| P40 | - | - | - | - |
| P41 | - | - | - | - |
| P42 | - | - | - | - |
| P43 | - | - | - | - |
| P44 | | - | - | - |
| P45 | - | - | - | - |
| P46 | - | - | PGA1_P | - |
| P47 | - | - | PGA1_N | - |
| P50 | - | - | PGA0_P | - |
| P51 | - | - | PGA0_N | - |
| P52 | AN21 | - | PGA0_O | - |
| P53 | AN22 | - | PGA123_O | DAC_O |
| P54 | - | C0P3 | - | - |
| P55 | - | C0N | - | - |
| P56 | AN5 | C0P2 | - | - |
| P57 | AN6 | C0P1 | - | - |

## 2.3 Register Mapping

### 2.3.1 Control Function Register Mapping

(Base address of the port control register =0x40040000)  RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| P0 | 0x000 | R/W | Set the register for configuring the output latch value in 1-bit units; read this register in input mode to get the pin level, and in output mode to get the value of the port's output latch. | 0x00 |
| P1 | 0x001 | R/W | | 0x00 |
| P2 | 0x002 | R/W | | 0x00 |
| P3 | 0x003 | R/W | | 0x00 |
| P4 | 0x004 | R/W | | 0x00 |
| P5 | 0x005 | R/W | | 0x00 |
| PM0 | 0x020 | R/W | When the port is used as a digital channel, the registers for the input or output of the port are set in 1-bit units. | 0xFF |
| PM1 | 0x021 | R/W | | 0xFF |
| PM2 | 0x022 | R/W | | 0xFF |
| PM3 | 0x023 | R/W | | 0xFF |
| PM4 | 0x024 | R/W | | 0xFF |
| PM5 | 0x025 | R/W | | 0xFF |
| PU0 | 0x030 | R/W | The internal pull-up resistor selection register of the port can only be set when the corresponding PMCmn=0, the pull-up resistor is valid. The pull-up function of P01, P02 and P03 is enabled by default. | 0x0E |
| PU1 | 0x031 | R/W | | 0x00 |
| PU2 | 0x032 | R/W | | 0x00 |
| PU3 | 0x033 | R/W | | 0x00 |
| PU4 | 0x034 | R/W | | 0x00 |
| PU5 | 0x035 | R/W | | 0x00 |
| PD0 | 0x040 | R/W | The internal pull-down resistor selection register of the port can only be set when the corresponding PMCmn=0, the pull-down resistor is valid. P01 has no pull-down function. | 0x00 |
| PD1 | 0x041 | R/W | | 0x00 |
| PD2 | 0x042 | R/W | | 0x00 |
| PD3 | 0x043 | R/W | | 0x00 |
| PD4 | 0x044 | R/W | | 0x00 |
| PD5 | 0x045 | R/W | | 0x00 |
| POM0 | 0x050 | R/W | Open Drain Mode Register, N-Channel Open Drain will be turned on only when the port is configured for Output Mode. | 0x00 |
| POM1 | 0x051 | R/W | | 0x00 |
| POM2 | 0x052 | R/W | | 0x00 |
| POM3 | 0x053 | R/W | | 0x00 |
| POM4 | 0x054 | R/W | | 0x00 |
| POM5 | 0x055 | R/W | | 0x00 |
| PMC0 | 0x060 | R/W | Port Mode Register, sets the port to be used as a digital or analog channel in 1-bit units; P00, P01, P02, P03, P04, and P05 are used as digital channels by default. | 0xC1 |
| PMC1 | 0x061 | R/W | | 0xFF |
| PMC2 | 0x062 | R/W | | 0xFF |
| PMC3 | 0x063 | R/W | | 0xFF |
| PMC4 | 0x064 | R/W | | 0xFF |
| PMC5 | 0x065 | R/W | | 0xFF |
| PSET0 | 0x070 | W | Sets the registers of the port output latch in 1-bit units. | 0x00 |
| PSET1 | 0x071 | W | | 0x00 |
| PSET2 | 0x072 | W | | 0x00 |
| PSET3 | 0x073 | W | | 0x00 |
| PSET4 | 0x074 | W | | 0x00 |
| PSET5 | 0x075 | W | | 0x00 |

| | | | | |
|---|---|---|---|---|
| PCLR0 | 0x080 | W | Clear the port output latch registers in 1-bit units. | 0x00 |
| PCLR1 | 0x081 | W | | 0x00 |
| PCLR2 | 0x082 | W | | 0x00 |
| PCLR3 | 0x083 | W | | 0x00 |
| PCLR4 | 0x084 | W | | 0x00 |
| PCLR5 | 0x085 | W | | 0x00 |

## 2.3.2 Output-Input Alternate Function Register Mapping

(Base address of the output-input alternate function register=0x40040800)

RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| P00CFG | 0x00 | R/W | Port output alternate configuration register allows mapping the output functionality of peripheral modules to the corresponding ports. For specific pin function digit mapping, please refer to Table 2-1. The reset value of the port output alternate function register is 0x00, which corresponds to the default alternate function and GPIO function of the port. | 0x00 |
| P01CFG | 0x02 | R/W | | 0x00 |
| P02CFG | 0x04 | R/W | | 0x00 |
| P03CFG | 0x06 | R/W | | 0x00 |
| P04CFG | 0x08 | R/W | | 0x00 |
| P05CFG | 0x0A | R/W | | 0x00 |
| P06CFG | 0x0C | R/W | | 0x00 |
| P07CFG | 0x0E | R/W | | 0x00 |
| P10CFG | 0x10 | R/W | | 0x00 |
| P11CFG | 0x12 | R/W | | 0x00 |
| P12CFG | 0x14 | R/W | | 0x00 |
| P13CFG | 0x16 | R/W | | 0x00 |
| P14CFG | 0x18 | R/W | | 0x00 |
| P15CFG | 0x1A | R/W | | 0x00 |
| P16CFG | 0x1C | R/W | | 0x00 |
| P17CFG | 0x1E | R/W | | 0x00 |
| P20CFG | 0x20 | R/W | | 0x00 |
| P21CFG | 0x22 | R/W | | 0x00 |
| P22CFG | 0x24 | R/W | | 0x00 |
| P23CFG | 0x26 | R/W | | 0x00 |
| P24CFG | 0x28 | R/W | | 0x00 |
| P25CFG | 0x2A | R/W | | 0x00 |
| P30CFG | 0x30 | R/W | | 0x00 |
| P31CFG | 0x32 | R/W | | 0x00 |
| P32CFG | 0x34 | R/W | | 0x00 |
| P33CFG | 0x36 | R/W | | 0x00 |
| P34CFG | 0x38 | R/W | | 0x00 |
| P35CFG | 0x3A | R/W | | 0x00 |
| P36CFG | 0x3C | R/W | | 0x00 |
| P37CFG | 0x3E | R/W | | 0x00 |
| P40CFG | 0x40 | R/W | | 0x00 |
| P41CFG | 0x42 | R/W | | 0x00 |
| P42CFG | 0x44 | R/W | | 0x00 |
| P43CFG | 0x46 | R/W | | 0x00 |
| P44CFG | 0x48 | R/W | | 0x00 |
| P45CFG | 0x4A | R/W | | 0x00 |
| P46CFG | 0x4C | R/W | | 0x00 |
| P47CFG | 0x4E | R/W | | 0x00 |
| P50CFG | 0x50 | R/W | | 0x00 |
| P51CFG | 0x52 | R/W | | 0x00 |
| P52CFG | 0x54 | R/W | | 0x00 |
| P53CFG | 0x56 | R/W | | 0x00 |
| P54CFG | 0x58 | R/W | | 0x00 |

| | | | | |
|---|---|---|---|---|
| P55CFG | 0x5A | R/W | | 0x00 |
| P56CFG | 0x5C | R/W | | 0x00 |
| P57CFG | 0x5E | R/W | | 0x00 |
| $PS_{int0}$_CFG | 0x60 | R/W | External interrupt 0 input port selection register | 0x7f |
| $PS_{int1}$_CFG | 0x61 | R/W | External interrupt 1 input port selection register | 0x7f |
| $PS_{int2}$_CFG | 0x62 | R/W | External interrupt 2 input port selection register | 0x7f |
| $PS_{int3}$_CFG | 0x63 | R/W | External interrupt 3 input port selection register | 0x7f |
| $PS_{tau0tin0}$_CFG | 0x64 | R/W | TAU0 external input channel 0 input port selection register | 0x7f |
| $PS_{tau0tin1}$_CFG | 0x65 | R/W | TAU0 external input channel 1 input port selection register | 0x7f |
| $PS_{tau0tin2}$_CFG | 0x66 | R/W | TAU0 external input channel 2 input port selection register | 0x7f |
| $PS_{tau0tin3}$_CFG | 0x67 | R/W | TAU0 external input channel 3 input port selection register | 0x7f |
| $PS_{uart0rxd}$_CFG | 0x68 | R/W | RXD0 signal input port selection register for UART0 | 0x07 |
| $PS_{epwmnkin}$_CFG | 0x69 | R/W | EPWM external brake input port selection register | 0x07 |
| $PS_{ccp0ain}$_CFG | 0x6A | R/W | CCP0A channel capture input port selection register | 0x07 |
| $PS_{ccp0bin}$_CFG | 0x6B | R/W | CCP0B channel capture input port selection register | 0x07 |
| $PS_{ccp1ain}$_CFG | 0x6C | R/W | CCP1A channel capture input port selection register | 0x07 |
| $PS_{ccp1bin}$_CFG | 0x6D | R/W | CCP1B channel capture input port selection register | 0x07 |
| $PS_{uart1rxd}$_CFG | 0x6E | R/W | RXD1 signal input port selection register for UART1 | 0x07 |
| $PS_{hall_in0}$_CFG | 0x6F | R/W | HALL sensor input IN0 channel port selection register | 0x07 |
| $PS_{hall_in1}$_CFG | 0x70 | R/W | HALL sensor input IN1 channel port selection register | 0x07 |
| $PS_{hall_in2}$_CFG | 0x71 | R/W | HALL sensor input IN2 channel port selection register | 0x07 |
| P0TTLCFG | 0x74 | R/W | P0 port input level selection register | 0x00 |
| PMS | 0x7B | R/W | Port Read Mode Selection Register; this register selects the read port latch value or pin output level when the port is in output mode. See Chapter 29.3.5 for details. | 0x00 |

### 2.3.3 Special Function Port RESINB Control Register Mapping

(Register base address=0x40020400) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| RSTM | 0x0B | R/W | Select the RESINB (P02) port as the external reset port or GPIO port register | 0x00 |

## 2.4 Register Description

The ports are controlled via the following registers.

(1) Port register (Px)
(2) Port mode register (PMx)
(3) Pull-up resistor selection register (PUx)
(4) Pull-down resistor selection register (PDx)
(5) Port output mode register (POMx)
(6) Port mode control register (PMCx)
(7) Port set control register (PSETx)
(8) Port clear control register (PCLRx)
(9) Port output alternate configuration register (PxxCFG)
(10) Port input alternate configuration register (PSxx_CFG)
(11) Port level selection register (PxTTLCFG)
(12) Special function port RESINB control register (RSTM)

### 2.4.1 Port Register (Px)

This is a register Px (x=0 to 5) which sets the value of the port's output latch in 1-bit units. Reading this register in input mode gives the pin level, and reading it in output mode gives the value of the port's output latch. After a reset signal is generated, the value of the register changes to "00H". The register is described as follows:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | Px7 | Bit 7 of the port x mode register<br>0: Output "0" in output mode; input low in input mode<br>1: Output "1" in output mode; input high in input mode | 0 |
| 6 | Px6 | Bit 6 of the port x mode register<br>0: Output "0" in output mode; input low in input mode<br>1: Output "1" in output mode; input high in input mode | 0 |
| 5 | Px5 | Bit 5 of the port x mode register<br>0: Output "0" in output mode; input low in input mode<br>1: Output "1" in output mode; input high in input mode | 0 |
| 4 | Px4 | Bit 4 of the port x mode register<br>0: Output "0" in output mode; input low in input mode<br>1: Output "1" in output mode; input high in input mode | 0 |
| 3 | Px3 | Bit 3 of the port x mode register<br>0: Output "0" in output mode; input low in input mode<br>1: Output "1" in output mode; input high in input mode | 0 |
| 2 | Px2 | Bit 2 of the port x mode register<br>0: Output "0" in output mode; input low in input mode<br>1: Output "1" in output mode; input high in input mode | 0 |
| 1 | Px1 | Bit 1 of the port x mode register<br>0: Output "0" in output mode; input low in input mode<br>1: Output "1" in output mode; input high in input mode | 0 |
| 0 | Px0 | Bit 0 of the port x mode register<br>0: Output "0" in output mode; input low in input mode<br>1: Output "1" in output mode; input high in input mode | 0 |

Note: Be sure to set bits that are not mounted to their initial values.

## 2.4.2 Port Mode Register (PMx)

When the port is used as a digital channel, this is the register PMx (x=0~5) that sets its input/output in bits. After a reset signal is generated, all ports default to the input state. The register is described as follows:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | PMx7 | Bit 7 of the port x mode register<br>0: Output mode (used as output port (output buffer ON))<br>1: Input mode (used as input port (output buffer OFF)) | 1 |
| 6 | PMx6 | Bit 6 of the port x mode register<br>0: Output mode (used as output port (output buffer ON))<br>1: Input mode (used as input port (output buffer OFF)) | 1 |
| 5 | PMx5 | Bit 5 of the port x mode register<br>0: Output mode (used as output port (output buffer ON))<br>1: Input mode (used as input port (output buffer OFF)) | 1 |
| 4 | PMx4 | Bit 4 of the port x mode register<br>0: Output mode (used as output port (output buffer ON))<br>1: Input mode (used as input port (output buffer OFF)) | 1 |
| 3 | PMx3 | Bit 3 of the port x mode register<br>0: Output mode (used as output port (output buffer ON))<br>1: Input mode (used as input port (output buffer OFF)) | 1 |
| 2 | PMx2 | Bit 2 of the port x mode register<br>0: Output mode (used as output port (output buffer ON))<br>1: Input mode (used as input port (output buffer OFF)) | 1 |
| 1 | PMx1 | Bit 1 of the port x mode register<br>0: Output mode (used as output port (output buffer ON))<br>1: Input mode (used as input port (output buffer OFF)) | 1 |
| 0 | PMx0 | Bit 0 of the port x mode register<br>0: Output mode (used as output port (output buffer ON))<br>1: Input mode (used as input port (output buffer OFF)) | 1 |

Note: The P26 and P27 port are invalid, and bits 7 to 6 of the PM2 are set to 1.

## 2.4.3 Pull-Up Resistor Selection Register (PUx)

On-chip pull-up resistor selection register PUx (x=0 to 5). The pull-up resistor can only be set when the corresponding PMCx bit is equal to 0.

After a reset signal is generated, the pull-up function of the P01, P02 and P03 ports are turned on automatically, and the pull-up function of the other ports will not be turned on by default. The register is described as follows:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | PUx7 | Bit 7 of the Px pin on-chip pull-up resistor selection<br>0: No on-chip pull-up resistor is connected<br>1: Connect the on-chip pull-up resistor | 0 |
| 6 | PUx6 | Bit 6 of the Px pin on-chip pull-up resistor selection<br>0: No on-chip pull-up resistor is connected<br>1: Connect the on-chip pull-up resistor | 0 |
| 5 | PUx5 | Bit 5 of the Px pin on-chip pull-up resistor selection<br>0: No on-chip pull-up resistor is connected<br>1: Connect the on-chip pull-up resistor | 0 |
| 4 | PUx4 | Bit 4 of the Px pin on-chip pull-up resistor selection<br>0: No on-chip pull-up resistor is connected<br>1: Connect the on-chip pull-up resistor | 0 |
| 3 | PUx3 | Bit 3 of the Px pin on-chip pull-up resistor selection<br>0: No on-chip pull-up resistor is connected<br>1: Connect the on-chip pull-up resistor | x=0, the reset value is 1<br>x=other value, the reset value is 0 |
| 2 | PUx2 | Bit 2 of the Px pin on-chip pull-up resistor selection<br>0: No on-chip pull-up resistor is connected<br>1: Connect the on-chip pull-up resistor | x=0, the reset value is 1<br>x=other value, the reset value is 0 |
| 1 | PUx1 | Bit 1 of the Px pin on-chip pull-up resistor selection<br>0: No on-chip pull-up resistor is connected<br>1: Connect the on-chip pull-up resistor | x=0, the reset value is 1<br>x=other value, the reset value is 0 |
| 0 | PUx0 | Bit 0 of the Px pin on-chip pull-up resistor selection<br>0: No on-chip pull-up resistor is connected<br>1: Connect the on-chip pull-up resistor | 0 |

Note 1: P26 and P27 are invalid, and bits7~6 of the PU2 are set to 0.

Note 2: The internal pull-up resistor is 10K.

## 2.4.4 Pull-Down Resistor Selection Register (PDx)

On-chip pull-down resistor selection register PDx (x=0 to 5). The pull-down resistor can only be set when the corresponding PMCx bit is equal to 0; RESINB (P01) port has no pull-down function.

After a reset signal is generated, the pull-down function of other ports will not be turned on by default. The register is described as follows:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | PDx7 | Bit 7 of the Px pin on-chip pull-down resistor selection<br>0: No on-chip pull-down resistor is connected<br>1: Connect the on-chip pull-down resistor | 0 |
| 6 | PDx6 | Bit 6 of the Px pin on-chip pull-down resistor selection<br>0: No on-chip pull-down resistor is connected<br>1: Connect the on-chip pull-down resistor | 0 |
| 5 | PDx5 | Bit 5 of the Px pin on-chip pull-down resistor selection<br>0: No on-chip pull-down resistor is connected<br>1: Connect the on-chip pull-down resistor | 0 |
| 4 | PDx4 | Bit 4 of the Px pin on-chip pull-down resistor selection<br>0: No on-chip pull-down resistor is connected<br>1: Connect the on-chip pull-down resistor | 0 |
| 3 | PDx3 | Bit 3 of the Px pin on-chip pull-down resistor selection<br>0: No on-chip pull-down resistor is connected<br>1: Connect the on-chip pull-down resistor | 0 |
| 2 | PDx2 | Bit 2 of the Px pin on-chip pull-down resistor selection<br>0: No on-chip pull-down resistor is connected<br>1: Connect the on-chip pull-down resistor | 0 |
| 1 | PDx1 | Bit 1 of the Px pin on-chip pull-down resistor selection<br>0: No on-chip pull-down resistor is connected<br>1: Connect the on-chip pull-down resistor<br>Note: The RESINB(P01) does not have a pull-down function. | 0 |
| 0 | PDx0 | Bit 0 of the Px pin on-chip pull-down resistor selection<br>0: No on-chip pull-down resistor is connected<br>1: Connect the on-chip pull-down resistor | 0 |

Note: P26 and P27 are invalid, and bits 7 to 6 of the PD2 are set to 0.

## 2.4.5 Port Output Mode Register (POMx)

The Port Output Mode Register (POMx, x=0~5) will only be enabled when configured to output mode with N-channel open-drain. When ports P04, P05, P06, P07, P14, P15, P34, P35, P54, and P55 are multiplexed for IIC functionality, open-drain mode will be forced to enable.

After a reset signal is generated, the value of the register changes to “00H”.

Note: For the bits set to N-channel open-drain output mode (POMmn=1), the on-chip pull-up resistor is not connected.

The register is described as follows:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | POMx7 | Bit 7 of the Px pin output mode selection<br>0: Typical output mode<br>1: N-channel open-drain output mode | 0 |
| 6 | POMx6 | Bit 6 of the Px pin output mode selection<br>0: Typical output mode<br>1: N-channel open-drain output mode | 0 |
| 5 | POMx5 | Bit 5 of the Px pin output mode selection<br>0: Typical output mode<br>1: N-channel open-drain output mode | 0 |
| 4 | POMx4 | Bit 4 of the Px pin output mode selection<br>0: Typical output mode<br>1: N-channel open-drain output mode | 0 |
| 3 | POMx3 | Bit 3 of the Px pin output mode selection<br>0: Typical output mode<br>1: N-channel open-drain output mode | 0 |
| 2 | POMx2 | Bit 2 of the Px pin output mode selection<br>0: Typical output mode<br>1: N-channel open-drain output mode | 0 |
| 1 | POMx1 | Bit 1 of the Px pin output mode selection<br>0: Typical output mode<br>1: N-channel open-drain output mode | 0 |
| 0 | POMx0 | Bit 0 of the Px pin output mode selection<br>0: Typical output mode<br>1: N-channel open-drain output mode | 0 |

Note: P26 and P27 are invalid, and bits 7 to 6 of the POM2 are set to 0.

## 2.4.6 Port Mode Control Register (PMCx)

The Port Mode Control Register (PMCx, x=0~5) is used to configure each port as either a digital (input/output) or an analog (input) channel in 1-bit units.

After a reset signal is generated, P01, P02, P03, P04, and P05 are defaulted to be used as digital channels (with reset values of PMC01, PMC02, PMC03, PMC04, PMC05 set to 0). Other ports are defaulted to be used as analog channels, meaning the corresponding bit in PMCx is set to 1. The specific register description is as follows:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | PMCx7 | Bit 7 of the Px pin digital (inputs/outputs) or analog (inputs)<br>0: Digital inputs/outputs (alternate functions other than analog inputs)<br>1: Analog inputs | 1 |
| 6 | PMCx6 | Bit 6 of the Px pin digital (inputs/outputs) or analog (inputs)<br>0: Digital inputs/outputs (alternate functions other than analog inputs)<br>1: Analog inputs | 1 |
| 5 | PMCx5 | Bit 5 of the Px pin digital (inputs/outputs) or analog (inputs)<br>0: Digital inputs/outputs (alternate functions other than analog inputs)<br>1: Analog inputs | x=0: 0<br>x=other: 1 |
| 4 | PMCx4 | Bit 4 of the Px pin digital (inputs/outputs) or analog (inputs)<br>0: Digital inputs/outputs (alternate functions other than analog inputs)<br>1: Analog inputs | x=0: 0<br>x=other: 1 |
| 3 | PMCx3 | Bit 3 of the Px pin digital (inputs/outputs) or analog (inputs)<br>0: Digital inputs/outputs (alternate functions other than analog inputs)<br>1: Analog inputs | x=0: 0<br>x=other: 1 |
| 2 | PMCx2 | Bit 2 of the Px pin digital (inputs/outputs) or analog (inputs)<br>0: Digital inputs/outputs (alternate functions other than analog inputs)<br>1: Analog inputs | x=0: 0<br>x=other: 1 |
| 1 | PMCx1 | Bit 1 of the Px pin digital (inputs/outputs) or analog (inputs)<br>0: Digital inputs/outputs (alternate functions other than analog inputs)<br>1: Analog inputs | x=0: 0<br>x=other: 1 |
| 0 | PMCx0 | Bit 0 of the Px pin digital (inputs/outputs) or analog (inputs)<br>0: Digital inputs/outputs (alternate functions other than analog inputs)<br>1: Analog inputs | 1 |

Note: P26 and P27 are invalid, and bits 7 to 6 of the PMC2 are set to 1.

## 2.4.7 Port Set Control Register (PSETx)

This is a register that sets the port output latch PSETx (x=0 to 5) in 1-bit units. After a reset signal is generated, the value of the register changes to “00H”. The register is described as follows:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | PSETx7 | Bit 7 of the Px pin set control<br>0: No operation<br>1: Corresponding Px7 set to 1 | 0 |
| 6 | PSETx6 | Bit 6 of the Px pin set control<br>0: No operation<br>1: Corresponding Px6 set to 1 | 0 |
| 5 | PSETx5 | Bit 5 of the Px pin set control<br>0: No operation<br>1: Corresponding Px5 set to 1 | 0 |
| 4 | PSETx4 | Bit 4 of the Px pin set control<br>0: No operation<br>1: Corresponding Px4 set to 1 | 0 |
| 3 | PSETx3 | Bit 3 of the Px pin set control<br>0: No operation<br>1: Corresponding Px3 set to 1 | 0 |
| 2 | PSETx2 | Bit 2 of the Px pin set control<br>0: No operation<br>1: Corresponding Px2 set to 1 | 0 |
| 1 | PSETx1 | Bit 1 of the Px pin set control<br>0: No operation<br>1: Corresponding Px1 set to 1 | 0 |
| 0 | PSETx0 | Bit 0 of the Px pin set control<br>0: No operation<br>1: Corresponding Px0 set to 1 | 0 |

Note: P26 and P27 are invalid, and bits 7 to 6 of the PSET2 are set to 0.

## 2.4.8 Port Clear Control Register (PCLRx)

This is a register that sets the port output latch PCLRx (x=0 to 5) in 1-bit units. After a reset signal is generated, the value of the register changes to “00H”. The register is described as follows:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | PCLRx7 | Bit 7 of the Px pin clear control<br>0: No operation<br>1: Corresponding Px7 set to 0 | 0 |
| 6 | PCLRx6 | Bit 6 of the Px pin clear control<br>0: No operation<br>1: Corresponding Px6 set to 0 | 0 |
| 5 | PCLRx5 | Bit 5 of the Px pin clear control<br>0: No operation<br>1: Corresponding Px5 set to 0 | 0 |
| 4 | PCLRx4 | Bit 4 of the Px pin clear control<br>0: No operation<br>1: Corresponding Px4 set to 0 | 0 |
| 3 | PCLRx3 | Bit 3 of the Px pin clear control<br>0: No operation<br>1: Corresponding Px3 set to 0 | 0 |
| 2 | PCLRx2 | Bit 2 of the Px pin clear control<br>0: No operation<br>1: Corresponding Px2 set to 0 | 0 |
| 1 | PCLRx1 | Bit 1 of the Px pin clear control<br>0: No operation<br>1: Corresponding Px1 set to 0 | 0 |
| 0 | PCLRx0 | Bit 0 of the Px pin clear control<br>0: No operation<br>1: Corresponding Px0 set to 0 | 0 |

Note: P26 and P27 are invalid, and bits 7 to 6 of the PCLR2 are set to 0.

## 2.4.9 Port Output Alternate Function Configuration Register ($P_{mn}$CFG)

The Port Output Alternate Function Configuration Register, $P_{mn}$CFG (where m=0, 1, 3, 4, 5 and n=0~7, and m=2 when n=0~5), allows the mapping of some peripheral module output functions to the port. The specific mapping functions are described in Table 2-1. The reset value of the port output alternate function configuration register is 00H, which means that the port is configured for default multi-functional and GPIO functions. The configuration steps are as follows:

1) Set PMCmn = 0 to choose digital input/output.
2) Set PMmn = 0 to set the output mode.
3) Set Pmn_CFG to select the corresponding pin's reset function output.

Note: The SDA, SCL of IIC and all SPI function pins can be used as both inputs and outputs. After setting PmnCFG, the input function will be automatically enabled. No additional operations are required beyond the three steps mentioned above.

The detailed register description is as follows:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PmnCFG[2:0] | Pmn pin output alternate function<br>0x00: Pmn's corresponding output mapping table =0x00<br>0x01: Pmn's corresponding output mapping table =0x01<br>0x02: Pmn's corresponding output mapping table =0x02<br>0x03: Pmn's corresponding output mapping table =0x03 | 0x0 |

### 2.4.10 Port Input Alternate Function Configuration Register (PSxx_CFG)

The INTP0, INTP1, INTP2, INTP3, TI00, TI01, TI02, TI03 can be mapped to any GPIO input. RXD0, RXD1, BKIN, CCP0AIN, CCP0BIN, CCP1AIN, CCP1BIN, HALL_IN0, HALL_IN1, and HALL_IN2 can be mapped to specific GPIO inputs. The specific input function mapping is detailed in Table 2-1. Since each function has multiple pin inputs, when using the input alternate function, it is necessary to select the specific GPIO input. The configuration steps are as follows:

4) Set PMCmn = 0
5) Set PMmn=1
6) Set PSxx_CFG to select the corresponding pin input.

PSintp0_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | -- | Reserved | -- |
| 6:0 | PSintp0_CFG[6:0] | INTP0 selects the GPIO input<br>0x00: Select P00 as INTP0 input<br>0x01: Select P01 as INTP0 input<br>… …<br>0x56: Select P56 as INTP0 input<br>0x57: Select P57 as INTP0 input<br>Other: Input low level | 0x7F |

PSintp1_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | -- | Reserved | -- |
| 6:0 | PSintp1_CFG[6:0] | INTP1 selects the GPIO input<br>0x00: Select P00 as INTP1 input<br>0x01: Select P01 as INTP1 input<br>… …<br>0x56: Select P56 as INTP1 input<br>0x57: Select P57 as INTP1 input<br>Other: Input low level | 0x7F |

PSintp2_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | -- | Reserved | -- |
| 6:0 | PSintp2_CFG[6:0] | INTP2 selects the GPIO input<br>0x00: Select P00 as INTP2 input<br>0x01: Select P01 as INTP2 input<br>… …<br>0x56: Select P56 as INTP2 input<br>0x57: Select P57 as INTP2 input<br>Other: Input low level | 0x7F |

PSintp3_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | -- | Reserved | -- |
| 6:0 | PSintp3_CFG[6:0] | INTP3 selects the GPIO input<br>0x00: Select P00 as INTP3 input<br>0x01: Select P01 as INTP3 input<br>… …<br>0x56: Select P56 as INTP3 input<br>0x57: Select P57 as INTP3 input<br>Other: Input low level | 0x7F |

PStau0tin0_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | -- | Reserved | -- |
| 6:0 | PStau0tin0_CFG[6:0] | TI00 selects the GPIO input<br>0x00: Select P00 as TI00 input<br>0x01: Select P01 as TI00 input<br>… …<br>0x56: Select P56 as TI00 input<br>0x57: Select P57 as TI00 input<br>Other: Input low level | 0x7F |

PStau0tin1_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | -- | Reserved | -- |
| 6:0 | PStau0tin1_CFG[6:0] | TI01 selects the GPIO input<br>0x00: Select P00 as TI01 input<br>0x01: Select P01 as TI01 input<br>… …<br>0x56: Select P56 as TI01 input<br>0x57: Select P57 as TI01 input<br>Other: Input low level | 0x7F |

PStau0tin2_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | -- | Reserved | -- |
| 6:0 | PStau0tin2_CFG[6:0] | TI02 selects the GPIO input<br>0x00: Select P00 as TI02 input<br>0x01: Select P01 as TI02 input<br>… …<br>0x56: Select P56 as TI02 input<br>0x57: Select P57 as TI02 input<br>Other: Input low level | 0x7F |

PStau0tin3_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | -- | Reserved | -- |
| 6:0 | PStau0tin3_CFG [6:0] | TI03 selects the GPIO input<br>0x00: Select P00 as TI03 input<br>0x01: Select P01 as TI03 input<br>… …<br>0x56: Select P56 as TI03 input<br>0x57: Select P57 as TI03 input<br>Other: Input low level | 0x7F |

PSuart0rxd_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PSuart0rxd_CFG[2:0] | UART0_RXD selects the GPIO input<br>0x00: Select P00 as the UART0_RXD input<br>0x01: Select P01 as the UART0_RXD input<br>0x02: Select P15 as the UART0_RXD input<br>0x03: Select P16 as the UART0_RXD input<br>0x04: Select P34 as the UART0_RXD input<br>0x05: Select P35 as the UART0_RXD input<br>Other: Input high level | 0x7 |

PSuart1rxd_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PSuart1rxd_CFG[2:0] | UART1_RXD selects the GPIO input<br>0x00: Select P23 as the UART1_RXD input<br>0x01: Select P24 as the UART1_RXD input<br>0x02: Select P25 as the UART1_RXD input<br>0x03: Select P53 as the UART1_RXD input<br>0x04: Select P54 as the UART1_RXD input<br>0x05: Select P55 as the UART1_RXD input<br>Other: Input high level | 0x7 |

PSepwmbkin_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PSepwmbkin_CFG [2:0] | EPWM_BKIN selects the GPIO input<br>0x00: Select P13 as the EPWM_BKIN input<br>0x01: Select P16 as the EPWM_BKIN input<br>0x02: Select P23 as the EPWM_BKIN input<br>0x03: Select P55 as the EPWM_BKIN input<br>Other: Input low level | 0x7 |

PSccp0ain_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PSccp0a_i_CFG [2:0] | CCP0AIN selects the GPIO input<br>0x00: Select P00 as the CCP0A_I input<br>0x01: Select P06 as the CCP0A_I input<br>0x02: Select P24 as the CCP0A_I input<br>Other: Input low level | 0x7 |

PSccp0bin_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PSccp0b_i_CFG [2:0] | CCP0BIN selects the GPIO input<br>0x00: Select P01 as the CCP0B_I input<br>0x01: Select P07 as the CCP0B_I input<br>0x02: Select P23 as the CCP0B_I input<br>0x03: Select P52 as the CCP0B_I input<br>Other: Input low level | 0x7 |

PSccp1ain_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PSccp1ain_CFG [2:0] | CCP1AIN selects the GPIO input<br>0x00: Select P02 as the CCP1A_I input<br>0x01: Select P04 as the CCP1A_I input<br>0x02: Select P11 as the CCP1A_I input<br>0x03: Select P25 as the CCP1A_I input<br>0x04: Select P56 as the CCP1A_I input<br>Other: Input low level | 0x7 |

PSccp1bin_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PSccp1bin_CFG [2:0] | CCP1BIN selects the GPIO input<br>0x00: Select P03 as the CCP1B_I input<br>0x01: Select P05 as the CCP1B_I input<br>0x02: Select P12 as the CCP1B_I input<br>0x03: Select P53 as the CCP1B_I input<br>0x04: Select P57 as the CCP1B_I input<br>Other: Input low level | 0x7 |

PShall_in0_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PShall_in0_CFG [2:0] | HALL_IN0 selects the GPIO input<br>0x00: Select P00 as the HALL_IN0 input<br>0x01: Select P20 as the HALL_IN0 input<br>Other: Input low level | 0x7 |

PShall_in1_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PShall_in1_CFG [2:0] | HALL_IN1 selects the GPIO input<br>0x00: Select P21 as the HALL_IN1 input<br>0x01: Select P57 as the HALL_IN1 input<br>Other: Input low level | 0x7 |

PShall_in2_CFG description:

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | -- | Reserved | -- |
| 2:0 | PShall_in2_CFG [2:0] | HALL_IN2 selects the GPIO input<br>0x00: Select P22 as the HALL_IN2 input<br>0x01: Select P56 as the HALL_IN2 input<br>Other: Input low level | 0x7 |

### 2.4.11 TTL and Schmitt Input Selection (P0TTLCFG)

The P0TTLCFG selection register is described as follows.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | P0TTL7 | P07 input level selection<br>0: Schmitt input<br>1: TTL input | 0 |
| 6 | P0TTL6 | P06 input level selection<br>0: Schmitt input<br>1: TTL input | 0 |
| 5 | P0TTL5 | P05 input level selection<br>0: Schmitt input<br>1: TTL input | 0 |
| 4 | P0TTL4 | P04 input level selection<br>0: Schmitt input<br>1: TTL input | 0 |
| 3 | P0TTL3 | P03 input level selection<br>0: Schmitt input<br>1: TTL input | 0 |
| 2 | P0TTL2 | P02 input level selection<br>0: Schmitt input<br>1: TTL input | 0 |
| 1 | P0TTL1 | P01 input level selection<br>0: Schmitt input<br>1: TTL input | 0 |
| 0 | P0TTL0 | P00 input level selection<br>0: Schmitt input<br>1: TTL input | 0 |

### 2.4.12 Special Function Port RESINB Description (RSTM)

The product's default behavior on power-up is that the RESINB (P01) pin is active. If you need to use this pin as a GPIO, you must disable the reset function through the register. The register description is as follows.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:1 | -- | Reserved, set to 0. | 0 |
| 0 | RSTM | RESINB pin external reset function mask<br>0: RESINB external reset pin<br>1: RESINB as the GPIO pin | 0 |

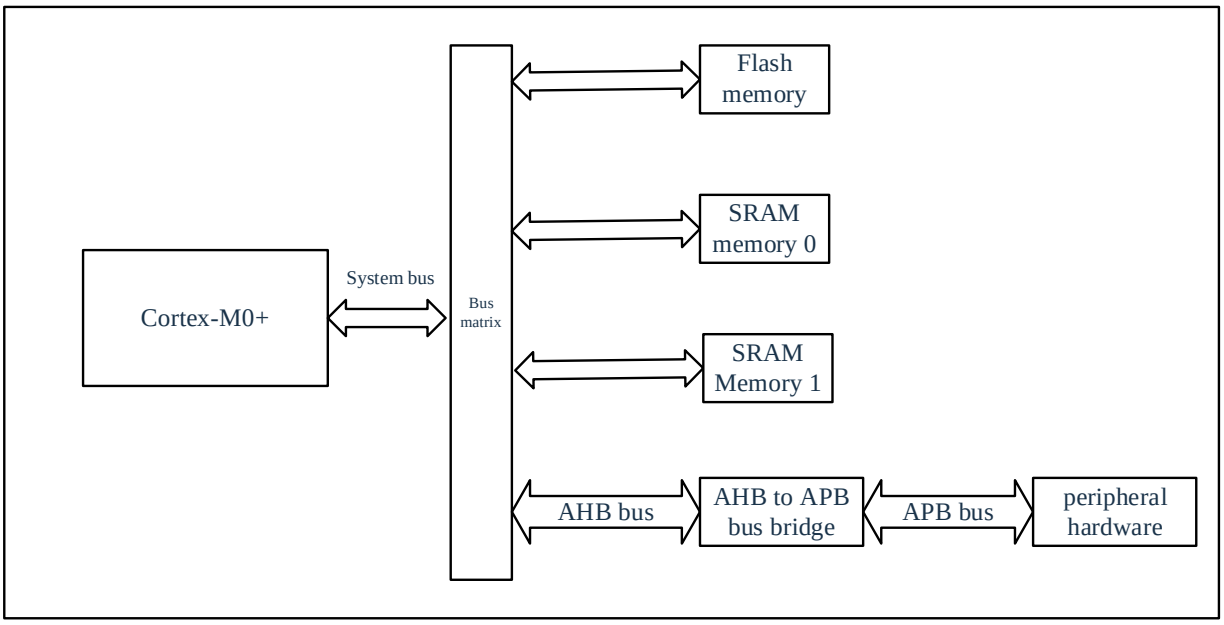
# Chapter 3 System Architecture

## 3.1 Overview

This product system consists of the following components:

- 1 AHB bus Master:
  - Cortex-M0+
- 4 AHB buses Slaves:
  - FLASH memory
  - SRAM memory 0
  - SRAM memory 1
  - AHB to APB Bridge, contains all APB interface peripherals.

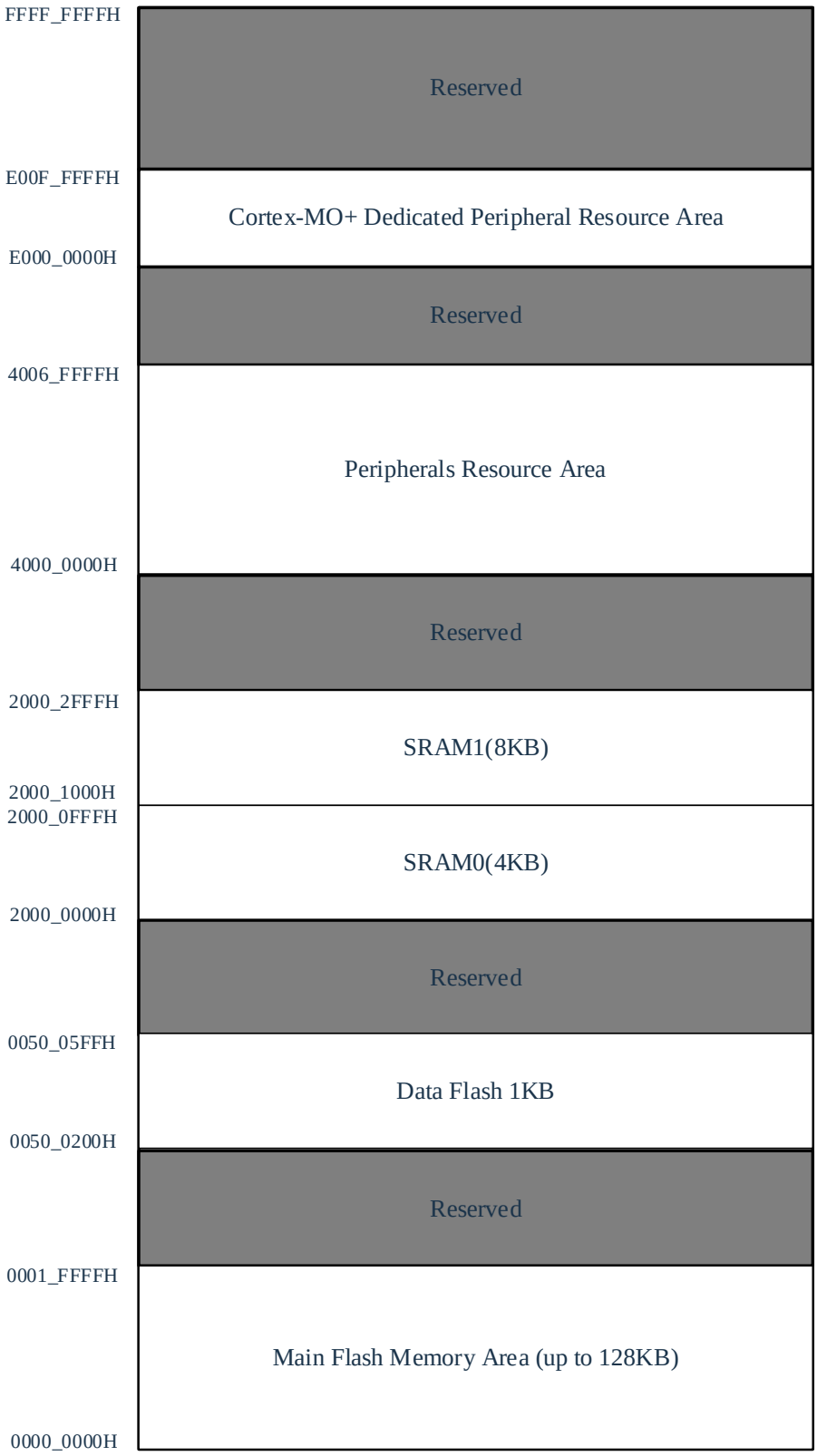
Figure 3-1 Block diagram of system architecture




1) System bus: This bus connects the system bus (peripheral bus) of the Cortex-M0+ core to the bus matrix.
2) Bus matrix: The bus matrix coordinates access to other buses on the core system bus.
3) AHB to APB Bridge: The AHB to APB Bridge provides a synchronous connection between the AHB and APB buses. Refer to Table 3-1 for address mapping of the different peripherals connected to each bridge.

## 3.2 System Address Partitioning

Figure 3-2 Map of address area



Peripheral Address Assignment

Table 3-1 Start address of peripheral register bank

| Start address | Peripheral | Remark |
|---|---|---|
| 0x4000_0000 - 0x4000_4FFF | Reserved | |
| 0x4000_5000 - 0x4000_5FFF | DMA | |
| 0x4000_6000 - 0x4000_6FFF | Reserved | |
| 0x4000_7000 - 0x4001_FFFF | Reserved | |
| 0x4002_0000 - 0x4002_03FF | FLASH control | |
| 0x4002_0400 - 0x4002_0FFF | Clock control | |
| 0x4002_1000 - 0x4002_1001 | Watchdog timer | |
| 0x4002_1002 - 0x4002_17FF | Reserved | |
| 0x4002_1800 - 0x4002_1BFF | High-speed CRC | See Chapter 29 Safety Function |
| 0x4002_1C00 - 0x4002_1FFF | Clock control | |
| 0x4002_2000 - 0x4003_FFFF | Reserved | |
| 0x4004_0000 - 0x4004_0F9F | GPIO | |
| 0x4004_0FA0 - 0x4004_10FF | Clock output | |
| 0x4004_1100 - 0x4004_1A2F | Reserved | |
| 0x4004_1A30 - 0x4004_1D7F | IICA | |
| 0x4004_1D80 - 0x4004_217F | Universal timer unit | |
| 0x4004_2180 - 0x4004_31FF | Reserved | |
| 0x4004_3200 - 0x4004_33FF | Universal CRC | See Chapter 29 Safety Function |
| 0x4004_3400 - 0x4004_4B4F | Reserved | |
| 0x4004_4B50 -0x4004_4B50 | LSITIMER | |
| 0x4004_4B51 - 0x4004_5AFF | Reserved | |
| 0x4004_5B00 - 0x4004_5BFF | External interrupt control | |
| 0x4004_5C00 - 0x4006_0FFF | Reserved | |
| 0x4006_1000 – 0x4006_1FFF | TIMER01 | |
| 0x4006_2000 – 0x4006_2FFF | Reserved | |
| 0x4006_3000 – 0x4006_3FFF | SPI | |
| 0x4006_4000 – 0x4006_40FF | UART0 | |
| 0x4006_4100 – 0x4006_41FF | UART1 | |
| 0x4006_4200 – 0x4006_42FF | EPWM | |
| 0x4006_4300 – 0x4006_437F | CCP | |
| 0x4006_4380 – 0x4006_447F | Reserved | |
| 0x4006_4480 – 0x4006_44BF | DIVSQRT | |
| 0x4006_44C0 – 0x4006_44FF | DIV | |
| 0x4006_4500 – 0x4006_453F | HALL | |
| 0x4006_4540 – 0x4006_7FFF | Reserved | |
| 0x4006_8000 – 0x4006_80FF | ADC | |
| 0x4006_8100 – 0x4006_81FF | Reserved | |

| | | |
|---|---|---|
| 0x4006_8200 – 0x4006_823F | ACMP0 | |
| 0x4006_8240 – 0x4006_82FF | Reserved | |
| 0x4006_8300 – 0x4006_831F | PGA0/1/2/3 | |
| 0x4006_8320 – 0x4006_833F | Reserved | |
| 0x4006_8340 – 0x4006_835F | ADCLDO | |
| 0x4006_8360 – 0x4006_836F | DAC | |
| 0x4006_8370 – 0x4006_8FFF | Reserved | |
| 0x4006_9000 – 0x4006_901F | TEST | See Chapter 33 |

# Chapter 4 Clock Generation Circuit

## 4.1 Functions of Clock Generation Circuit

The clock generation circuit is a circuit that generates a clock supplied to the CPU and peripheral hardware. There are the following 2 types of system clock and clock oscillation circuits.

(1) Main system clock

**X1 Oscillator Circuit**: The X1 pin and X2 pin can be connected to a resonator to generate an oscillation with a frequency of $f_X$ = 4 to 8 MHz. The oscillation can be stopped by entering deep sleep mode or setting the MSTOP bit (bit 7 of the CSC register).

**High-Speed On-Chip Oscillator (High-Speed OCO)**: The frequency can be selected from $f_{IH}$ = 72 MHz, 64 MHz, 36 MHz, 32 MHz, 18 MHz, 16 MHz, 9 MHz, 8 MHz, 4.5 MHz, 4 MHz, and 2 MHz (typical) via the option byte (000C2H). After reset release, the CPU starts running with the $f_{IH}$ clock. The oscillation can be stopped by entering deep sleep mode or setting the HIOSTOP bit (bit 0 of the CSC register). The frequency can be changed by the High-Speed On-Chip Oscillator Frequency Selection Register (HOCODIV) based on the option byte setting. For frequency settings, refer to setting of high-speed on-chip oscillator frequency selection register (HOCODIV).

**PLL (Phase-Locked Loop) Circuit**: The PLL can provide a system clock with a maximum of 72 MHz and a minimum of 48 MHz by multiplying the X1 oscillator frequency or the divided high-speed on-chip oscillator frequency. The oscillation and stop of the PLL are controlled by the PLL controller (PLLCR).

The switching between X1 clock and high-speed on-chip oscillator clock can be controlled by setting the MCM0 bit (bit 4 of the System Clock Control Register (CKC)).

(2) Subsystem clock

**Low-Speed On-Chip Oscillator (Low-Speed OCO)**: The low-speed on-chip oscillator oscillates when bit 4 (WDTON) of the option byte (000C0H) is 1, or when bit 4 (WUTMMCK0) of the 12-bit interval timer clock selection register (OSMC) is 1, or when bit 0 (SELLOSC) of the low-speed on-chip oscillator clock selection register (SUBCKSEL) is 1.

However, if bit WDTON is 1 and both bits WUTMMCK0 and SELLOSC are 0, and bit 0 (WDSTBYON) of the option byte (000C0H) is 0, the low-speed on-chip oscillator stops oscillating when entering deep sleep mode or sleep mode.

Note: $f_{HOCO}$: High-speed on-chip oscillator clock frequency

$f_{IH}$: High-speed internal oscillator divided clock frequency

$f_{IL}$: Low-speed internal oscillator clock frequency

$f_X$: High-speed external crystal oscillator clock oscillation frequency

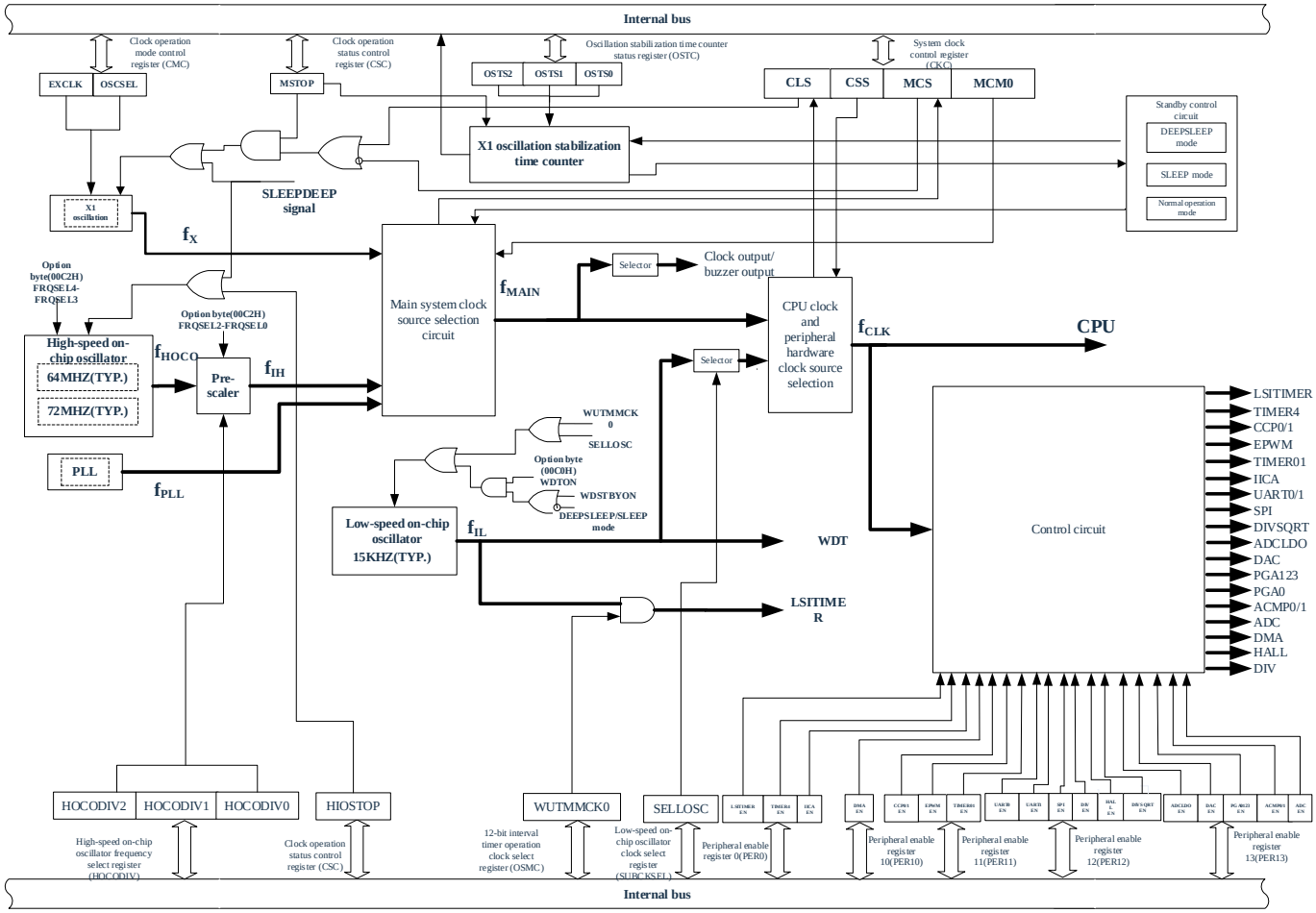
## 4.2 Configuration of Clock Generation Circuit

The clock generation circuit includes the following hardware.

Table 4-1 Configuration of clock generation circuit

| Item | Configuration |
|---|---|
| Control register | Clock operation mode control register (CMC)<br>System clock control register (CKC)<br>Clock operation status control register (CSC)<br>Oscillation stabilization time counter status register (OSTC)<br>Oscillation stabilization time selection register (OSTS)<br>Peripheral enable registers (PER0, PER10, PER11, PER12, PER13)<br>High-speed on-chip oscillator frequency selection register (HOCODIV)<br>High-speed internal oscillator trim register (HIOTRM) |
| Oscillatior circuit | High-speed on-chip oscillator<br>Low-speed on-chip oscillator<br>X1 oscillation circuit |

Figure 4-1 Block diagram of clock generation circuit




Note: $f_{HOCO}$ : High-speed on-chip oscillator clock frequency

$f_{IH}$ : High-speed internal oscillator divided clock frequency

$f_{MAIN}$: Main system clock frequency

$f_{CLK}$: CPU/peripheral hardware clock frequency

$f_{IL}$ : Low-speed internal oscillator clock frequency

$f_X$ : High-speed external crystal oscillator clock oscillation frequency

$f_{PLL}$: PLL clock frequency

## 4.3 Register Mapping

(Base address of the following registers = 0x4002_0400)

RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CMC | 0x000 | R/W | Clock operation mode control register | 0x00 |
| CSC | 0x001 | R/W | Clock operation status control register | 0xC0 |
| OSTC | 0x002 | RO | Oscillation stabilization time counter status register | 0x00 |
| OSTS | 0x003 | R/W | Oscillation stabilization time selection register | 0x07 |
| CKC | 0x004 | R/W | System clock control register | 0x00 |
| SUBCKSEL | 0x007 | R/W | Low-speed on-chip oscillator clock selection register | 0x00 |
| PMUKEY | 0x008 | WO | Power mode control protection register | 0x00 |
| PMUCTL | 0x00A | R/W | Power mode control register | 0x00 |
| PER0 | 0x020 | R/W | Peripheral enable register 0 | 0x00 |
| OSMC | 0x023 | R/W | 12-bit interval timer operation clock selection register | 0x00 |

(Base address of the following registers = 0x4002_0810)

RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| PER10 | 0x000 | R/W | Peripheral enable register 10 | 0x00 |
| PER11 | 0x001 | R/W | Peripheral enable register 11 | 0x00 |
| PER12 | 0x002 | R/W | Peripheral enable register 12 | 0x00 |
| PER13 | 0x003 | R/W | Peripheral enable register 13 | 0x00 |

(Base address of the following registers = 0x4002_1C00)

RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| HOCODIV | 0x020 | R/W | High-speed on-chip oscillator frequency selection register | The set values of the FRQSEL[2:0] bits of option byte (000C2H) |

(Base address of the following registers = 0x4002_0C00)

RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| MCKC | 0x000 | R/W | System clock master register | 0x00 |
| PLLCR | 0x004 | R/W | PLL control register | 0x00 |

## 4.4 Register Description

### 4.4.1 System Clock Control Register (CKC)

This is a register that selects the CPU/peripheral hardware clock and the main system clock.

The CKC register is set by an 8-bit memory manipulation instruction.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | CLS[Note1] | CPU/peripheral hardware clock ($F_{CLK}$) status<br>0: Main system clock ($F_{MAIN}$)<br>1: Low-speed on-chip oscillator clock ($F_{IL}$) | 0 |
| 6 | CSS | CPU/ peripheral hardware clock ($F_{CLK}$) selection<br>0: Main system clock ($F_{MAIN}$)<br>1: Low-speed on-chip oscillator clock ($F_{IL}$) | 0 |
| 5 | MCS[Note1] | Main system clock ($F_{MAIN}$) status<br>0: High-speed on-chip oscillator clock or PLL as main system clock ($F_{MAIN}$)<br>1: X1 oscillation clock as the main system clock ($F_{MAIN}$) | 0 |
| 4 | MCM0 | Main system clock ($F_{MAIN}$) selection<br>0: High-speed on-chip oscillator clock or PLL as main system clock ($F_{MAIN}$)<br>1: X1 oscillation clock as the main system clock ($F_{MAIN}$) | 0 |
| 3:0 | -- | Reserved | 0x0 |

Note 1: Bit5 and bit7 are read-only bits.

Note 2: Clocks set by the CSS bit are provided for the CPU and peripheral hardware. If you change the CPU clock, change the peripheral hardware clock at the same time (except for clock output and watchdog timer). Therefore, if you want to change the clock on the CPU/peripheral hardware, you must stop the peripheral functions.

## 4.4.2 Clock Operation Status Control Register (CSC)

This is a register that controls the operation of the high-speed on-chip oscillator clock. The CSC register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to “C0H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | MSTOP | X1 oscillation clock operation control<br>0: X1 oscillation circuit runs<br>1: X1 oscillation circuit stops | 1 |
| 6:1 | -- | Reserved | 0x20 |
| 0 | HIOSTOP | Operation control of high-speed on-chip oscillator clock<br>0: High-speed on-chip oscillator runs<br>1: High-speed on-chip oscillator stops | 0 |

Note 1: Do not stop the clock selected for the CPU peripheral hardware clock ($F_{CLK}$) with the CSC register.

Note 2: To use the X1 oscillating clock, configure the CMC register to X1 oscillation mode;

Note 3: For the register flag setting to stop clock oscillation and the conditions before stopping, refer to Table 4-2.

Table 4-2 Conditions before stopping clock oscillation

| Clock | Condition before stopping clock | Setting of CSC register flags |
|---|---|---|
| High-speed on-chip oscillator clock | CPU/peripheral hardware clock runs on a clock other than the high-speed on-chip oscillator clock or PLL clock (CLS=1) | HIOSTOP=1 |

## 4.4.3 PLL Control Register (PLLCR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7 | PLLCKSEL | PLL input clock selection:<br>0: High-speed internal oscillator divided clock<br>1: X1 oscillation clock | 0 |
| 6:4 | - | Reserved | - |
| 3:1 | PLLP | PLL clock multiplier select bit<br>000: 6x<br>001: 9x<br>010: 12x<br>011: 18x<br>100: 8x<br>101: 12x<br>110: 16x<br>111: 24x | 0x0 |
| 0 | PLLON | PLL module enable:<br>0: Disable<br>1: Enable | 0 |

## 4.4.4 System Clock Master Register (MCKC)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7 | PLLCKSTR | System clock selection PLL switch success status bit:<br>0: System clock selection: $F_{IH}$ output<br>1: System clock selection: PLL output | 0 |
| 6:3 | - | Reserved | - |
| 2:1 | RDIV | PLL output clock division factor selection bit:<br>00: No division<br>01: Divided by 2<br>10: Divided by 4<br>11: Divided by 8 | 0x0 |
| 0 | MCKSEL | System clock selection bit:<br>0: Select $F_{IH}$ output<br>1: Select PLL output | 0 |

Note: The clock output range of PLL is 48MHz to 72MHz.

## 4.4.5 Clock Operation Mode Control Register (CMC)

This is the register for setting the operating mode of the X1 and X2 pins.

After the reset is released, the CMC register can only be written once with an 8-bit memory manipulation instruction. It can be read using an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register is set to “00H”.

| Bit | Symbol | Description | | | | | Reset value |
|---|---|---|---|---|---|---|---|
| | | X1, X2 pin operation mode control | | | | | |
| | | EXCLK | OSCSEL | High-speed system pin operating mode | X1 pin | X2/ECLK pin | |
| 7 | EXCLK | 0 | 0 | Port mode | Input/output port | | 0 |
| | | 0 | 1 | X1 oscillator mode | Crystal or ceramic resonator connection | | |
| 6 | OSCSEL | 1 | 0 | Port mode | Input/output port | | 0 |
| | | 1 | 1 | Settings are disabled | | | |
| 3:0 | - | Reserved | | | | | 0x0 |

Note 1: After the reset is released, the CMC register can only be written once using an 8-bit memory manipulation instruction. When the CMC register is used with the initial value (00H), in order to prevent erroneous actions in case of program malfunction (if a value other than 00H is mistakenly written, it cannot be recovered), the CMC register must be set to 00H after the reset is released.

Note 2: After the reset is released and before the X1 oscillator starts through the clock operation status control register (CSC), the CMC register must be set.

Note 3: The system clock frequency limit is 72 MHz, but the frequency limit of the X1 oscillator circuit is 8 MHz.

## 4.4.6 Oscillation Stabilization Time Selection Register (OSTS)

This is the register for selecting the oscillation stabilization time of the X1 clock.

If the X1 clock oscillates, the system will automatically wait for the time set by the OSTS register after the X1 oscillator circuit starts (MSTOP=0).

If the CPU clock is switched from the high-speed on-chip oscillator clock or low-speed on-chip oscillator clock to the X1 clock, or if the CPU clock is using the high-speed on-chip oscillator clock and then switches to deep sleep mode while the X1 clock is oscillating, and later exits deep sleep mode, it is necessary to confirm whether the oscillation stabilization time has elapsed by checking the state register of the oscillation stabilization time counter (OSTC).

The OSTC register can confirm the time previously set by the OSTS register.

After a reset signal is generated, the value of this register will become “07H”.

| Bit | Symbol | Description | | | | | | Reset value |
|---|---|---|---|---|---|---|---|---|
| 7:3 | - | Reserved | | | | | | 0x0 |
| 2:0 | OSTS[2:0] | Selection of oscillation stabilization time | | | | | | 0x7 |
| | | OSTS2 | OSTS1 | OSTS0 | | $f_X$=4MHz | $f_X$=8MHz | |
| | | 0 | 0 | 0 | $2^8/f_X$ | 64us | 32us | |
| | | 0 | 0 | 1 | $2^9/f_X$ | 128us | 64us | |
| | | 0 | 1 | 0 | $2^{10}/f_X$ | 256us | 128us | |
| | | 0 | 1 | 1 | $2^{11}/f_X$ | 512us | 256us | |
| | | 1 | 0 | 0 | $2^{13}/f_X$ | 2.04ms | 1.02ms | |
| | | 1 | 0 | 1 | $2^{15}/f_X$ | 8.17ms | 4.07ms | |
| | | 1 | 1 | 0 | $2^{17}/f_X$ | 32.7ms | 16.3ms | |
| | | 1 | 1 | 1 | $2^{18}/f_X$ | 65.5ms | 32.7ms | |

Note 1: To change the setting of the OSTS register, you must make the change before the MSTOP bit of the Clock Operation Status Control Register (CSC) set to 0.

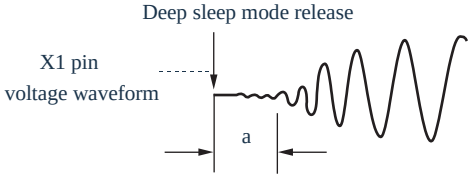
Note 2: The oscillation stable time counter is counted only in that oscillation stable time set in the OSTS register.

In the following cases, the setting value of the oscillation stability time of the OSTS register must be greater than the count value confirmed by the OSTC register after the start of the oscillation.

• When the CPU clock is a high-speed on-chip oscillator clock or a subsystem clock and the X1 clock is to start oscillating

• When the CPU clock is a high-speed on-chip oscillator clock and is released from deep sleep mode after shifting to deep sleep mode in the state of X1 clock oscillation (therefore, it must be noted that the OSTC register after release from deep sleep mode only sets the state within the oscillation stabilization time set by the OSTS register)

Note 3: The oscillation stable time of the X1 clock does not include the time before the clock starts to oscillate (Figure a below).



## 4.4.7 Oscillation Stabilization Time Counter Status Register (OSTC)

This register represents the count status of the X1 clock oscillation stabilization time counter. It can confirm the X1 clock's oscillation stabilization time in the following situations:

When the CPU clock is using the high-speed on-chip oscillator clock or low-speed on-chip oscillator clock, and the X1 clock oscillation begins.

When the CPU clock is using the high-speed on-chip oscillator clock, and the sleep mode is released after transitioning to deep sleep mode while the X1 clock is oscillating.

The OSTC register can be read using an 8-bit memory manipulation instruction.

The value of this register will become "00H" after a reset signal is generated, entering deep sleep mode, or when the MSTOP bit (bit 7 of the clock operation status control register (CSC)) is set to "1".

Remark: The oscillation stabilization time counter starts counting in the following cases:

1) When the X1 clock starts oscillating (EXCLK, OSCSEL=0,1, MSTOP=0).
2) When waking up from deep sleep mode.

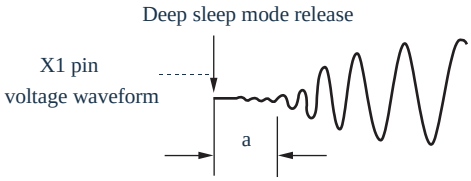
| Bit | Symbol | Description | | | | | | | | | | | Reset value |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Oscillation stabilization time state | | | | | | | | | | | |
| | | MOST8 | MOST9 | MOST10 | MOST11 | MOST13 | MOST15 | MOST17 | MOST18 | | $f_X$=4MHz | $f_X$=8MHz | |
| 7 | MOST8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Max.$2^8/f_X$ | Max.64us | $f_X$=8MHz | 0 |
| 6 | MOST9 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Min.$2^8/f_X$ | Min.64us | Max.32us | 0 |
| 5 | MOST10 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | Min.$2^9/f_X$ | Min.128us | Min.32us | 0 |
| 4 | MOST11 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | Min.$2^{10}/f_X$ | Min.256us | Min.64us | 0 |
| 3 | MOST13 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | Min.$2^{11}/f_X$ | Min.512us | Min.128us | 0 |
| 2 | MOST15 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | Min.$2^{13}/f_X$ | Min.2.04ms | Min.256us | 0 |
| 1 | MOST17 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | Min.$2^{15}/f_X$ | Min.8.17ms | Min.1.02ms | 0 |
| 0 | MOST18 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | Min.$2^{17}/f_X$ | Min.32.7ms | Min.4.07ms | 0 |
| | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | Min.$2^{18}/f_X$ | Min.65.5ms | Min.16.3ms | |

Note 1: After the above-mentioned time has elapsed, each bit starting from the MOST8 bit will sequentially change to 1 and remain the state.

Note 2: The oscillation stabilization time counter counts up to the oscillation stabilization time set by the OSTS register. In the following cases, set the oscillation stabilization time of the OSTS register to the value greater than the count value which is to be checked by the OSTC register after the oscillation starts.

- If the X1 clock starts oscillation while the high-speed on-chip oscillator clock or low-speed on-chip oscillator is being used as the CPU clock.
- If the deep sleep mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock oscillating. (Note, therefore, that only the status up to the oscillation stabilization time set by the OSTS register is set to the OSTC register after the deep sleep mode is released.)

Note 3: The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts (see "a" below)


Deep sleep mode release
X1 pin
voltage waveform
a

## 4.4.8 Peripheral Enable Registers (PER0, PER10, PER11, PER12, PER13)

These registers are used to enable or disable supplying the clock to the peripheral hardware. Clock supply to the hardware that is not used is also stopped so as to decrease the power consumption and noise. When using the following peripheral functions controlled by these registers, the corresponding bits must be set to "1" before initial settings of the peripheral functions.

| | |
|---|---|
| PER0 | LSITIMER, IICA, TIMER4 |
| PER10 | DMA |
| PER11 | CCP0/1, EPWM, TMER01 |
| PER12 | HALL, DIV, UART1, UART0, SPI, DIVSQRT |
| PER13 | ADCLDO, DAC, PGA, ACMP, ADC |

The PER0, PER10, PER11, PER12, and PER13 registers can be set using 8-bit memory manipulation instructions.

After a reset signal is generated, the values of these registers will become "00H".

Peripheral enable register 0 (PER0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | LSITIMEREN | Control of LSITIMER input clock supply (power-down sleep is possible)<br>0: Stop to supply the input clock, the SFR used by the LSITIMER cannot be written.<br>1: Supply the input clock, the SFR used by the LSITIMER can be written. | 0 |
| 6:5 | -- | Reserved | 0x0 |
| 4 | IICAEN | Control of IICA input clock supply<br>0: Stop to supply the input clock, the SFR used by the IICA cannot be written.<br>1: Supply the input clock, the SFR used by the IICA can be written. | 0 |
| 3:1 | -- | Reserved | 0x0 |
| 0 | TM40EN | Control of Timer Unit 0 input clock supply<br>0: Stop to supply the input clock, the SFR used by the Timer Unit cannot be written.<br>1: Supply the input clock, the SFR used by the Timer Unit 0 can be written. | 0 |

Peripheral enable register 10 (PER10)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:4 | -- | Reserved | 0x0 |
| 3 | DMAEN | Control of DMA input clock supply<br>0: Stop to supply the input clock, DMA can not run.<br>1: Supply the input clock, DMA can run. | 0 |
| 2:0 | -- | Reserved | 0x0 |

Peripheral enable register 11 (PER11)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:5 | -- | Reserved | 0x0 |
| 4 | CCPEN | Control of CCP input clock supply<br>0: Stop to supply the input clock, CCP can not run.<br>1: Supply the input clock, CCP can run. | 0 |
| 3 | EPWMEN | Control of EPWM input clock supply<br>0: Stop to supply the input clock, EPWM can not run.<br>1: Supply the input clock, EPWM can run | 0 |
| 2:1 | -- | Reserved | 0x0 |
| 0 | TIMER01EN | Control of TIMER01 input clock supply<br>0: Stop to supply the input clock, TIMER01 can not run.<br>1: Supply the input clock, TIMER01 can run. | 0 |

Peripheral enable register 12 (PER12)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | TESTEN | Test register clock enable<br>0: Stop to supply the input clock, and TEST can not run.<br>1: Supply the input clock, and TEST can run. | 0 |
| 6 | HALLEN | Control of HALL input clock supply<br>0: Stop to supply the input clock, and HALL can not run.<br>1: Supply the input clock, and HALL can run. | 0 |
| 5 | DIVEN | Control of DIV input clock supply<br>0: Stop to supply the input clock, and DIV can not run.<br>1: Supply the input clock, and DIV can run. | 0 |
| 4 | -- | Reserved | -- |
| 3 | UART1EN | Control of UART input clock supply<br>0: Stop to supply the input clock, and UART can not run.<br>1: Supply the input clock, and UART can run. | 0 |
| 2 | UART0EN | Control of UART input clock supply<br>0: Stop to supply the input clock, and UART can not run.<br>1: Supply the input clock, and UART can run. | 0 |
| 1 | SPIEN | Control of SPI input clock supply | 0 |

| | | | |
|---|---|---|---|
| | | 0: Stop to supply the input clock, SPI can not run.<br>1: Supply the input clock, SPI can run. | |
| 0 | DIVSQRTEN | Control of DIVSQRT input clock supply<br>0: Stop to supply the input clock, and DIVSQRT can not run.<br>1: Supply the input clock, and DIVSQRT can run. | 0 |

Peripheral enable register 13 (PER13)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | - | Reserved | - |
| 6 | ADCLDOEN | Control of ADCLDO input clock supply<br>0: Stop to supply the input clock, and ADCLDO can not run.<br>1: Supply the input clock, and ADCLDO can run. | 0 |
| 5 | DACEN | Control of DAC input clock supply<br>0: Stop to supply the input clock, and DAC can not run.<br>1: Supply the input clock, and DAC can run. | 0 |
| 4 | - | Reserved | - |
| 3 | PGAEN | Control of PGA0/1/2/3 input clock supply<br>0: Stop to supply the input clock, and PGA0/1/2/3 can not run.<br>1: Supply the input clock, and PGA0/1/2/3 can run. | 0 |
| 2 | - | Reserved | - |
| 1 | ACMPEN | Control of ACMP0/1 input clock supply<br>0: Stop to supply the input clock, and ACMP0/1 can not run.<br>1: Supply the input clock, and ACMP0/1 can run. | 0 |
| 0 | ADCEN | Control of ADC input clock supply<br>0: Stop to supply the input clock, and ADC can not run.<br>1: Supply the input clock, and ADC can run. | 0 |

## 4.4.9 12-Bit Interval Timer Operation Clock Selection Register (OSMC)

The OSMC register is used to select the operating clock for the 12-bit interval timer LSITIMER.

The OSMC register can be set using an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register will become “00H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:5 | -- | Reserved | 0x0 |
| 4 | WUTMMCK0 | Selection of 12-bit interval timer operation clock:<br>0: The low-speed on-chip oscillator clock stops providing the clock to the 12-bit interval timer LSITIMER.<br>1: The low-speed on-chip oscillator clock provides the clock to the 12-bit interval timer LSITIMER. | 0 |
| 3:0 | -- | Reserved | 0x0 |

## 4.4.10 High-Speed On-Chip Oscillator Frequency Selection Register (HOCODIV)

This is the register that sets the frequency of the high-speed on-chip oscillator through the option byte (000C2H). However, the selectable frequencies vary depending on the values of the FRQSEL[4:3] bits in the option byte (000C2H).

The HOCODIV register can be set using an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register will become the value set by the FRQSEL[2:0] bits in the option byte (000C2H).

| Bit | Symbol | Description | | | | Reset value |
|---|---|---|---|---|---|---|
| 7:3 | -- | Reserved | | | | 0 |
| 2:0 | HOCODIV[2:0] | Selection of high-speed on-chip oscillator clock frequency | | | | Setting values of the FRQSEL[2:0] of the option byte (000C2H) |
| | | | FSQSEL[4:3]=11/10 | | FSQSEL[4:3]=01/00 | |
| | | 000 | $f_{IH}$=72MHZ | $f_{HOCO}$=72MHZ | $f_{IH}$=64MHZ | $f_{HOCO}$=64MHZ |
| | | 001 | $f_{IH}$=36MHZ | $f_{HOCO}$=72MHZ | $f_{IH}$=32MHZ | $f_{HOCO}$=64MHZ |
| | | 010 | $f_{IH}$=18MHZ | $f_{HOCO}$=72MHZ | $f_{IH}$=16MHZ | $f_{HOCO}$=64MHZ |
| | | 011 | $f_{IH}$=9MHZ | $f_{HOCO}$=72MHZ | $f_{IH}$=8MHZ | $f_{HOCO}$=64MHZ |
| | | 100 | $f_{IH}$=4.5MHZ | $f_{HOCO}$=72MHZ | $f_{IH}$=4MHZ | $f_{HOCO}$=64MHZ |
| | | 101 | Settings are disabled | | $f_{IH}$=2MHZ | $f_{HOCO}$=64MHZ |
| | | Other than the above | Settings are disabled | | | |

Note 1: The HOCODIV register must be set in a state where the high-speed on-chip oscillator clock ($F_{IH}$) is selected as the CPU/peripheral hardware clock ($F_{CLK}$).

Note 2: After changing the frequency through the HOCODIV register, the frequency switch is performed after the following transition time:

- Run up to 3 clocks at the frequency before the change.
- Wait for up to 3 CPU/peripheral hardware clocks at changed frequencies.

### 4.4.11 Low-Speed On-Chip Oscillator Clock Selection Register (SUBCKSEL)

The SUBCKSEL register is the register that selects the low-speed on-chip oscillator clock $f_{IL}$.

The SUBCKSEL register can be set using an 8-bit memory operation instruction.

After a reset signal is generated, the value of this register will become “00H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:1 | -- | Reserved | 0x0 |
| 0 | SELLOCO | Low-speed on-chip oscillator clock selection<br>0: Selecting the low-speed on-chip oscillator clock is disabled<br>1: Select the low-speed on-chip oscillator clock | 0 |

### 4.4.12 Power Supply Mode Control Protection Register (PMUKEY)

The PMUKEY register is the register that protects the power mode control PMUCTL.

The PMUKEY register can be set using a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of this register will become “0000H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:0 | PMUKEY | Power supply mode control protection register selection<br>• To release PMUCTL write protection, write 192AH and 3E4FH sequentially to the PMUKEY to enable the write control of the PWDNEN bit in PMUCTL.<br>• Other. The PMUCTL write settings will be invalid. | 0x0000 |

### 4.4.13 Power Supply Mode Control Register (PMUCTL)

The PMUCTL register is the register that controls the enablement of the power control mode.

The PMUCTL register can be set using an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register will become “00H”, and write protection will be enabled. Write control can be released through the PMUKEY.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:1 | -- | Reserved | 0x0 |
| 0 | PWDNEN | Power supply mode control register selection<br>0: Disable partial power-down mode<br>1: Enable partial power-down mode | 0 |

Note: The write protection of PMUCTL can be released through the PMUKEY.

# 4.5 System Clock Oscillation Circuit

## 4.5.1 X1 Oscillation Circuit

The X1 oscillator circuit operates through a crystal or ceramic resonator (4-8 MHz) connected between the X1 and X2 pins.

When using the X1 oscillator circuit, the bit 7 and bit 6 (EXCLK, OSCSEL) of the Clock Operation Mode Control Register (CMC) must be set as follows:

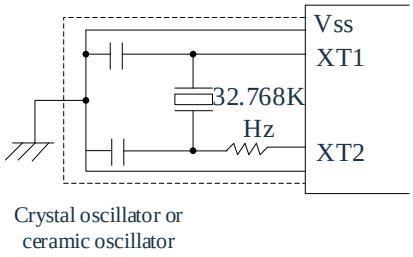
For crystal or ceramic oscillators: EXCLK, OSCSEL = 0, 1

When not using the X1 oscillator circuit, it must be set to port mode (EXCLK, OSCSEL = 0, 0).

An example of the external circuit for the X1 oscillator is shown in the figure below.

Figure 4-2 Example of an external circuit for X1 oscillation circuit

(a) Crystal or ceramic oscillator




Note: When using the X1 oscillator circuit, in order to avoid the impact of parasitic capacitance and other factors, the wiring of the dashed part in Figure 4-2 must be done as follows:

a) Keep the wiring length as short as possible.

b) Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flow.

c) Always make the ground point of the oscillator capacitor the same potential as VSS. Do not ground the capacitor to a ground pattern through which a high current flow.

d) Do not fetch signals from the oscillator.

## 4.5.2 High-Speed On-Chip Oscillator

The CMS32M67xx has a built-in high speed on-chip oscillator. Frequency can be selected from 72MHz, 64MHz, 36MHz, 32MHz, 18MHz, 16MHz, 9MHz, 8MHz, 4.5MHz, 4MHz, and 2MHz via option bytes (000C2H). The oscillation can be controlled by the bit0 (HIOSTOP) of the clock operation status control register (CSC).

After a reset is released, the high-speed on-chip oscillator automatically starts to oscillate.

### 4.5.3 Low-Speed On-Chip Oscillator

The CMS32M67xx has a built-in low-speed on-chip oscillator.

The low-speed on-chip oscillator clock is used as the external reference clock for the watchdog timer, LSITIMER, and SysTick timer. It can also be used as the CPU clock and peripheral module clock.

When bit 4 (WDTON) of the option byte (000C0H), or bit 4 (WUTMMCK0) of the 12-bit interval timer operation clock selection register (OSMC) is set to 1, or bit 0 (SELLOSC) of the low-speed on-chip oscillator clock selection register (SUBCKSEL) is set to 1, the low-speed on-chip oscillator oscillates.

When the watchdog timer stops running and the WUTMMCK0 bit is not 0, the low-speed on-chip oscillator continues to oscillate. However, when the WUTMMCK0 is 0 and the SELLOSC is 0, and the WDTON is 1 and bit 0 (WDSTBYON) of the option byte (000C0H) is 0, the low-speed on-chip oscillator stops oscillating if the system enters deep sleep or sleep mode. During the operation of the watchdog timer, even if the program malfunctions, the low-speed on-chip oscillator clock continues to run.

# 4.6 Operation of Clock Generation Circuit

The clock generation circuit generates various clocks as shown below and controls the operation mode of the CPU such as the standby mode (refer to Figure 4-1).

$F_{MAIN}$: Main system clock

$F_{IH}$: High-speed on-chip oscillator clock

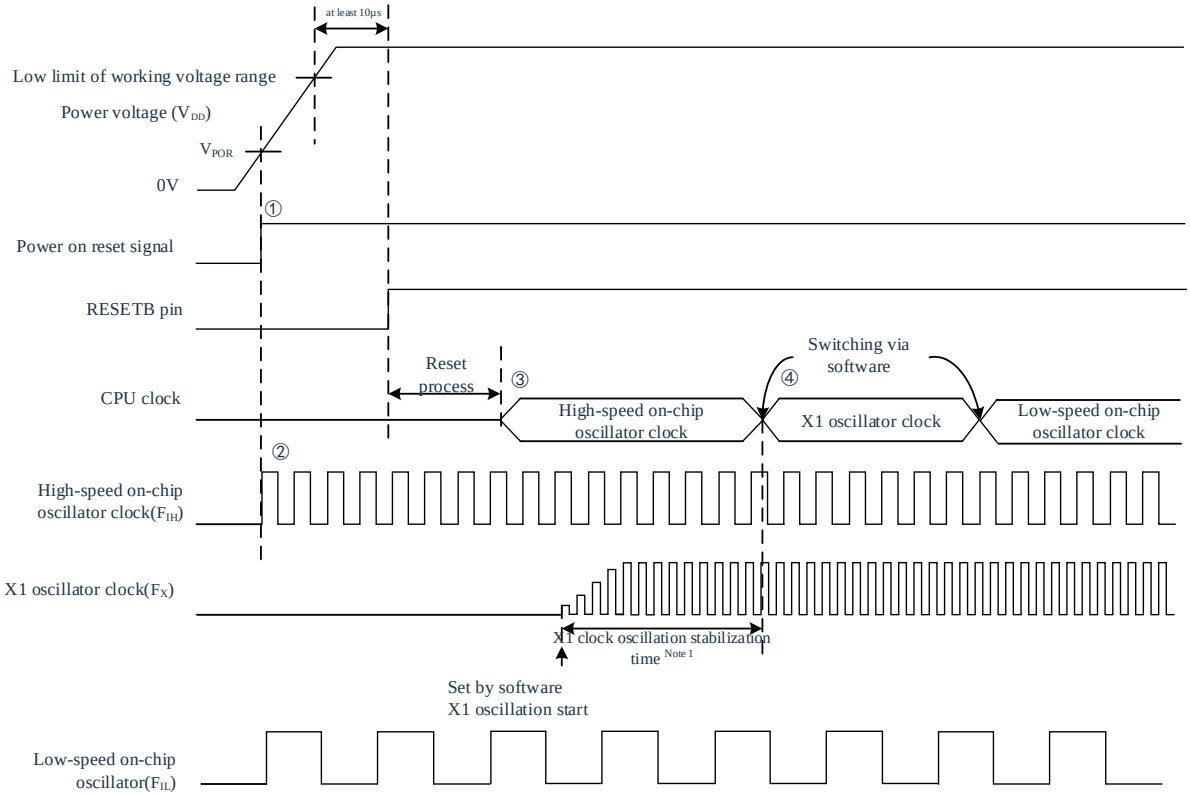
$F_{IL}$: Low-speed on-chip oscillator clock

$F_{CLK}$: CPU/peripheral hardware clock

$F_X$ : High-speed external crystal oscillator clock oscillation frequency

After the CMS32M67xx is released from reset, the CPU begins operation through the output of the high-speed on-chip oscillator. The operation of the clock generation circuit when the power is turned on is shown in Figure 4-3.

Figure 4-3 Operation of the clock generation circuit when the power is turned on




1) After power is turned on, an internal reset signal is generated through the power-on reset (POR) circuit.
2) However, the reset state is maintained by a voltage detection circuit or an external reset until the operating voltage range shown in the AC characteristics of the datasheet is reached (the above figure shows an example when an external reset is used).
3) The high-speed on-chip oscillator starts oscillating automatically after the reset is released.
4) After the reset is released, voltage stabilization waiting and reset processing are performed, and then the CPU starts running with a high-speed on-chip oscillator clock.
5) The X1 clock must be started by software to begin oscillation (refer to 4.7.2 Example of X1 Oscillator Circuit Setup).
6) If the CPU clock is to be switched to the X1 clock, it must be switched by software after waiting for the clock

oscillation to stabilize (refer to 4.7.2 Example of X1 Oscillator Circuit Setup).

Note: When resetting is released, the oscillation stabilization time of the X1 clock must be confirmed through the oscillation stabilization time counter status register (OSTC).

# 4.7 Clock Control

## 4.7.1 Example of Setting Up a High-Speed On-Chip Oscillator

The CPU/peripheral hardware clock ($F_{CLK}$) must run at the high-speed on-chip oscillator clock after the reset is released. The frequency of the high-speed on-chip oscillator can be selected from 72MHz, 64MHz, 36MHz, 32MHz, 18MHz, 16MHz, 9MHz, 8MHz, 4.5MHz, 4MHz and 2MHz by using bits FRQSEL0 to FRQSEL4 of the option byte (000C2H). In addition, the frequency can be changed by the high-speed on-chip oscillator register (HOCODIV).

[Option byte 000C2 setting]

| Bit | Symbol | Description | | | Reset value |
|---|---|---|---|---|---|
| 7:5 | -- | Reserved | | | -- |
| 4:0 | FRQSEL[4:0] | Selection of high-speed on-chip oscillator clock frequency | | | -- |
| | | FRQSEL[4:0] value | $f_{HOCO}$ | $f_{IH}$ | |
| | | 5'b1x000 | 72MHZ | 72MHZ | |
| | | 5'b1x001 | 72MHZ | 36MHZ | |
| | | 5'b1x010 | 72MHZ | 18MHZ | |
| | | 5'b1x011 | 72MHZ | 9MHZ | |
| | | 5'b1x100 | 72MHZ | 4.5MHZ | |
| | | 5'b0x000 | 64MHZ | 64MHZ | |
| | | 5'b0x001 | 64MHZ | 32MHZ | |
| | | 5'b0x010 | 64MHZ | 16MHZ | |
| | | 5'b0x011 | 64MHZ | 8MHZ | |
| | | 5'b0x100 | 64MHZ | 4MHZ | |
| | | 5'b0x101 | 64MHZ | 2MHZ | |
| | | Other than the above | Settings are prohibited. | | |

[Setting of high-speed on-chip oscillator frequency selection register (HOCODIV)]

| Bit | Symbol | Description | | | Reset value |
|---|---|---|---|---|---|
| 7:3 | -- | Reserved | | | 0 |
| 2:0 | HOCODIV[2:0] | Selection of high-speed on-chip oscillator clock frequency | | | Set value of FRQSEL2 to FRQSEL0 bits of option byte (000C2H) |
| | | HOCODIV[2:0] values | FSQSEL[4,3]=11/10 | FSQSEL[4,3]=01/00 | |
| | | 3'b000 | $f_{IH}$=72MHZ $f_{HOCO}$=72MHZ | $f_{IH}$=64MHZ $f_{HOCO}$=64MHZ | |
| | | 3'b001 | $f_{IH}$=36MHZ $f_{HOCO}$=72MHZ | $f_{IH}$=32MHZ $f_{HOCO}$=64MHZ | |
| | | 3'b010 | $f_{IH}$=18MHZ $f_{HOCO}$=72MHZ | $f_{IH}$=16MHZ $f_{HOCO}$=64MHZ | |
| | | 3'b011 | $f_{IH}$=9MHZ $f_{HOCO}$=72MHZ | $f_{IH}$=8MHZ $f_{HOCO}$=64MHZ | |
| | | 3'b100 | $f_{IH}$=4.5MHZ $f_{HOCO}$=72MHZ | $f_{IH}$=4MHZ $f_{HOCO}$=64MHZ | |
| | | 3'b101 | Settings are prohibited | $f_{IH}$=2MHZ $f_{HOCO}$=64MHZ | |
| | | Other than the above | Settings are prohibited | | |

## 4.7.2 Example of Setting X1 Oscillation Circuit

After a reset release, the CPU/peripheral hardware clock ($F_{CLK}$) always starts operating with the high-speed on-chip oscillator clock. To subsequently change the clock to the X1 oscillation clock, set the oscillator and start oscillation by using the oscillation stabilization time select register (OSTS) and clock operation mode control register (CMC) and clock operation status control register (CSC) and wait for oscillation to stabilize by using the oscillation stabilization time counter status register (OSTC). After the oscillation stabilizes, set the X1 oscillation clock to $F_{CLK}$ by using the system clock control register (CKC).

[Register settings] The registers must be set in the order of ① to ⑤.

① Set the EXCLK, OSCSEL of the CMC register to 2'b01, and select the corresponding clock input port as X1 oscillation mode.

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CMC | EXCLK | OSCSEL | 0 | 0 | 0 | 0 | 0 | 0 |

② Using the OSTS register, select the oscillation stabilization time of the X1 oscillator at releasing of the deep sleep mode.

Example) Setting values when a wait of at least 128μs is set based on a 8MHz resonator. Set the OSTS register to 0x02.

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSTS | 0 | 0 | 0 | 0 | 0 | OSTS2 | OSTS1 | OSTS0 |

③ Clear the MSTOP bit of the CSC register to start oscillating the X1 oscillator.

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CSC | MSTOP | 0 | 0 | 0 | 0 | 0 | 0 | HIOSTOP |

④ Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize.

Example) Wait until the bits rof the OSTC to 0xE0 when a wait of at least 128μs is set based on a 8MHz resonator.

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSTC | MOST8 | MOST9 | MOST10 | MOST11 | MOST13 | MOST15 | MOST17 | MOST18 |

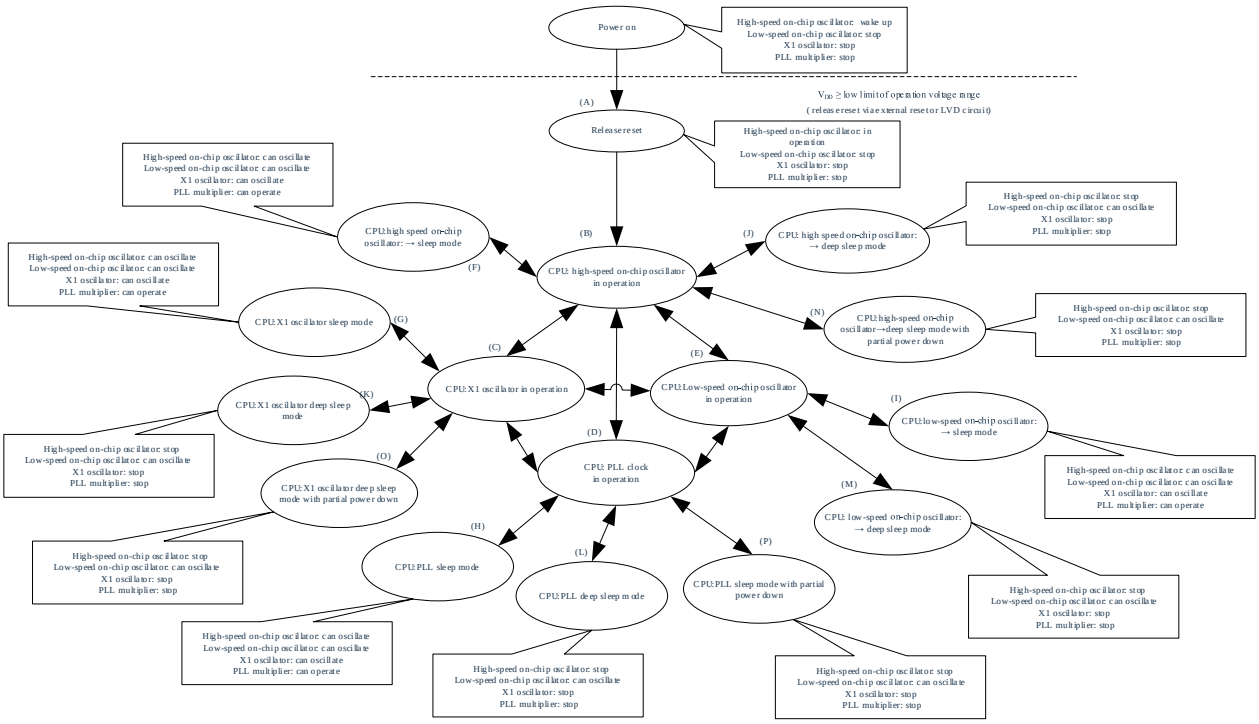
⑤ Use the MCM0 bit of the CKC register to specify the X1 oscillation clock as the CPU/peripheral hardware clock.

| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CKC | CLS | CSS | MCS | MCM0 | 0 | 0 | 0 | 0 |

# 4.7.3 State Transition Graph of the CPU Clock

Figure 4-4 shows the CPU clock status transition diagram of this product.

Figure 4-4 CPU clock status transition diagram




Examples of CPU clock transfer and SFR register setting are in Table 4-3.

Table 4-3 CPU clock transition and SFR register setting examples(1/5)

(1) CPU operating with high-speed on-chip oscillator clock (B) after reset release (A)

| Status transition | SFR register setting |
|---|---|
| (A)→(B) | SFR registers do not have to be set (default status after reset release). |

(2) CPU operating with high-speed system clock (C) after reset release (A)

| Status transition | SFR register setting |
|---|---|
| (B)→(C)<br>(D)→(C)<br>(E)→(C) | Register configuration sequence: |
| | ① Set MCM0 in the CKC register to 1 to select the X1 oscillation clock as the main clock output. |
| | ② Check the MCS bit in the CKC register and wait for this bit to be 1. |
| | ③ Set CSS in the CKC register to 0 to select the main clock as the CPU and peripheral circuit clock. |
| | ④ Check the CLS bit in the CKC register and wait for this bit to be 0. |

Table 4-3 CPU clock transition and SFR register setting examples(2/5)

(3) CPU changing to PLL clock (D) after reset release (A)

| Status transition | SFR register setting |
|---|---|
| (B)→(D)<br>(D)→(D)<br>(E)→(D) | Register configuration sequence: |
| | ① Configure the PLLCR register's PLLCKSEL, PLLP, and PLLON bits to select the PLL input clock source, set the frequency multiplier, and enable PLL. |
| | ② Set the MCKSEL bit in the MCKC register to 1 to select the PLL clock as the output. |
| | ③ Check the PLLCKSTR bit in the MCKC register and wait for this bit to be 1. |
| | ④ Set MCM0 in the CKC register to 0 to select PLL as the main clock output. |
| | ⑤ Check the MCS bit in the CKC register and wait for this bit to be 0. |
| | ⑥ Set CSS in the CKC register to 0 to select the main clock as the CPU and peripheral circuit clock. |
| | ⑦ Check the CLS bit in the CKC register and wait for this bit to be 0. |

(4) CPU clock changing to low-speed on-chip oscillator clock (E) after reset release (A)

| Status transition | SFR register setting |
|---|---|
| (B)→(E)<br>(D)→(E)<br>(C)→(E) | Register configuration sequence: |
| | ① Set the SELLOCO bit in the SUBCKSEL register to 1. |
| | ② Set the CSS bit in the CKC register to 1 to select the low-speed on-chip oscillator clock as the CPU and peripheral circuit clock. |
| | ③ Check the CLS bit in the CKC register and wait for this bit to be 1. |

(5) CPU clock changing from other clocks to high-speed on-chip oscillator clock (B)

| Status transition | SFR register setting |
|---|---|
| (C)→(B)<br>(D)→(B)<br>(E)→(B) | Register configuration sequence: |
| | ① Set the MCKSEL bit in the MCKC register to 0 to select the on-chip high-speed oscillator clock as the output. |
| | ② Check the PLLCKSTR bit in the MCKC register and wait for this bit to be 0. |
| | ③ Set the MCM0 in the CKC register to 0 to select HOCO as the main clock output. |
| | ④ Check the MCS bit in the CKC register and wait for this bit to be 0. |
| | ⑤ Set the CSS in the CKC register to 0 to select the main clock as the CPU and peripheral circuit clock. |
| | ⑥ Check the CLS bit in the CKC register and wait for this bit to be 0. |

Note 1: (A) to (P) of Table 4-3 correspond to (A) to (P) of Figure 4-4.

Note 2: The oscillation accuracy stabilization wait of the high-speed on-chip oscillator clock may vary due to temperature conditions during deep sleep mode.

Table 4-3 CPU clock transition and SFR register setting examples(3/5)

(6) CPU clock changing from high-speed on-chip clock operation (B) to sleep mode (F).
CPU clock changing from high-speed external clock operation (C) to sleep mode (G).
CPU clock changing from PLL clock operation (D) to sleep mode (H).
CPU clock changing from low-speed on-chip clock operation (E) to sleep mode (I).

| Status transition | SFR register setting |
|---|---|
| (B)→(F)<br>(C)→(G)<br>(D)→(H) | Execute WFI. |

Note: (A) to (P) of Table 4-3 correspond to (A) to (P) of Figure 4-4.

Table 4-3 CPU clock transition and SFR register setting examples(4/5)

(7) CPU clock changing from high-speed on-chip clock operation (B) to deep sleep mode (J).
CPU clock changing from high-speed external clock operation (C) to deep sleep mode (K).
CPU clock changing from PLL clock operation (D) to deep sleep mode (L).
CPU clock changing from low-speed on-chip clock operation (E) to deep sleep mode (M).

(SFR register setting order) ⟶

| Status transition | SFR register setting | |
|---|---|---|
| (B)→(J)<br>(C)→(K)<br>(D)→(L)<br>(E)→(M) | Stop peripheral functions that cannot operate in deep sleep mode. | Set the bit2 (SLEEPDEEP) of the SCR register to 1, and execute the WFI instruction. |

Note: (A) to (P) of Table 4-3 correspond to (A) to (P) of Figure 4-4.

Table 4-3 CPU clock transition and SFR register setting examples(5/5)

(8) CPU clock changing from high-speed on-chip clock operation (B) to deep sleep mode with partial power-down (N).
CPU clock changing from high-speed external clock operation (C) to deep sleep mode with partial power-down (O).
CPU clock changing from PLL clock operation (D) to deep sleep mode with partial power-down (P).

(SFR register setting order) ⟶

| Status transition | SFR register setting | | |
|---|---|---|---|
| (B)→(N)<br>(C)→(O)<br>(D)→(P) | Stop peripheral functions that cannot operate in deep sleep mode. | PMUKEY=0x192A;<br>PMUKEY=0x3E4F;<br>PMUCTL=0x01; | Set the bit2 (SLEEPDEEP) of the SCR register to 1, and execute the WFI instruction. |

Note: (A) to (P) of Table 4-3 correspond to (A) to (P) of Figure 4-4.

### 4.7.4 Conditions Before CPU Clock Transfer and Post-Transfer Processing

The conditions before the CPU clock transfer and the processing after the transfer are shown below.

Table 4-4 Transfer of CPU clocks

| CPU clock | | Conditions before transfer | Post-transfer processing |
|---|---|---|---|
| Before transfer | After transfer | | |
| High-speed on-chip oscillator clock | Low-speed on-chip oscillator clock | Select low-speed on-chip oscillator clock<br>SELLOCO=1 | If the high-speed on-chip oscillator is stopped (HIOSTOP=1), the operating current can be reduced. When the PLL clock source is the high-speed on-chip oscillator, the high-speed on-chip oscillator cannot be stopped when switching the CPU clock to PLL. |
| | PLL clock | Enable PLL operation<br>• PLLON=1<br>• PLL input clock source oscillation stabilization | |
| | X1 oscillator clock | X1 oscillation stabilization<br>• OSCSEL=1,EXCLK=0,MSTOP=0<br>• After oscillation stabilization time | |
| Low-speed on-chip oscillator clock | High-speed on-chip oscillator clock | Enable high-speed on-chip oscillator oscillation<br>• HIOSTOP=0<br>• After oscillation stabilization time | If the watchdog and LSITIMER are not running, the low-speed on-chip clock oscillator can be turned off (SELLOCO=0) to reduce the operating current. |
| | PLL clock | Enable PLL operation<br>• PLLON=1<br>• PLL input clock source oscillation stabilization | |
| | X1 oscillator clock | X1 oscillation stabilization<br>• OSCSEL=1,EXCLK=0,MSTOP=0<br>• After oscillation stabilization time | |
| PLL clock | High-speed on-chip oscillator clock | Enable high-speed on-chip oscillator oscillation<br>• HIOSTOP=0<br>• After oscillation stabilization time | The PLL can be disabled (PLLON=0) to reduce the operating current. |
| | Low-speed on-chip oscillator clock | Select low-speed on-chip oscillator clock<br>SELLOCO=1 | |
| | X1 oscillator clock | X1 oscillation stabilization<br>• OSCSEL=1,EXCLK=0,MSTOP=0<br>• After oscillation stabilization time | |
| X1 oscillator clock | High-speed on-chip oscillator clock | Enable high-speed on-chip oscillator oscillation<br>• HIOSTOP=0<br>• After oscillation stabilization time | If the X1 oscillation clock is stopped (MSTOP=1), the operating current can be reduced. When the PLL clock source is the X1 oscillation clock, the X1 clock oscillator cannot be stopped when switching the CPU clock to PLL. |
| | Low-speed on-chip oscillator clock | Select low-speed on-chip oscillator clock<br>SELLOCO=1 | |
| | PLL clock | Enable PLL operation<br>• PLLON=1<br>• PLL input clock source oscillation stabilization | |

## 4.7.5 Time Required To Switch CPU Clock and Main System Clock

It can switch CPU clock (main system clock↔low-speed on-chip oscillator clock, i.e., high speed on-chip oscillator clock↔ low-speed on-chip oscillator clock) by setting the bit6 (CSS) of system clock control register (CKC).

The actual switchover does not occur immediately after the CKC register is overridden, but several clocks continue to run with the clock before the switchover after the CKC register is changed (see Table 4-5).

The CPU can be determined by the bit7 (CLS), bit (MCS) of the CKC register or the bit7 (PLLCKSTR) of the MCKC register whether the CPU is the running clock.

If you switch the CPU clock, switch the peripheral hardware clock at the same time.

Table 4-5 Number of clocks required for clock switching

| Clock after switching / Clock before switching | High speed on-chip oscillator clock ($f_{IH}$) | X1 oscillator clock ($f_X$) | PLL ($f_{PLL}$) | Low speed on-chip oscillator clock ($f_{IL}$) |
|---|---|---|---|---|
| High-speed on-chip oscillator clock ($f_{IH}$) | | 2.5 $f_{IH}$ / $f_X$ clocks | 2.5 $f_{IH}$ /$f_{PLL}$ clocks | 2.5 $f_{IH}$ / $f_{IL}$ clocks |
| X1 oscillator clock ($f_X$) | 2 clocks | | 2.5 $f_X$ /$f_{PLL}$ clocks | 2.5 $f_X$ /$f_{IL}$ clocks |
| PLL($f_{PLL}$) | 2 clocks | 2.5 $f_{PLL}$/ $f_X$ clocks | | 2.5 $f_{PLL}$/ $f_{IL}$ clocks |
| Low-speed on-chip oscillator clock ($f_{IL}$) | 2 clocks | 2 clocks | 2 clocks | |

Note 1: The number of clocks in Table 4-5 is the number of CPU clocks before the switch.

Note 2: The number of clocks Table 4-5 is the number of clocks rounded to the decimal portion.

Example: When switching CPU from the main system clock to sub-system clock (oscillation with $F_{IH}$=2MHz, $F_{SUB}$= $F_{IL}$ =15KHz)

$2.5F_{MAIN}/F_{SUB}=2.5(2000/15)=333.3\approx334$ clocks

## 4.7.6 Conditions Before Clock Oscillation Is Stopped

The following lists the register flag settings for stopping the clock oscillation and conditions before the clock oscillation is stopped.

Table 4-6 Conditions and flag settings before clock oscillation stops

| Clock | Conditions before clock oscillation is stopped | Flag settings of SFR register |
|---|---|---|
| High-speed on-chip oscillator clock | CLS=1<br>(CPU runs at a clock other than the high-speed on-chip oscillator clock or PLL clock) | HIOSTOP=1 |
| X1 oscillator clock | MCS=0 or CLS=1; when runs on the PLL clock, PLLCKSEL=0<br>(The CPU runs on a clock other than the high-speed system clock, and when running on the PLL clock, the clock source is not the X1 oscillating clock.) | MSTOP=1 |
| PLL clock | MCS=0 or CLS=1 or PLLCKSEL=0<br>(CPU runs on a clock other than the PLL clock) | PLLON=0 |
| Low-speed on-chip oscillator clock | CLS=0,WDTON=0,WUMMCK0=0<br>(CPU runs at a clock other than the low-speed on-chip oscillator clock) | SELLOCO=0 |

# Chapter 5 Universal Timer Unit

## 5.1 Functions of Universal Timer Unit

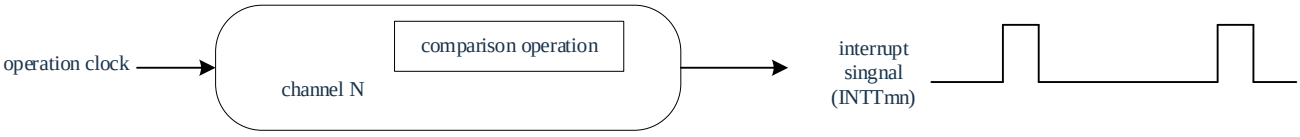
The universal timer unit has the following functions.

### 5.1.1 Independent Channel Operation Function

The independent channel operation function is a function that enables independent use of any channel without being affected by other channel operation modes.

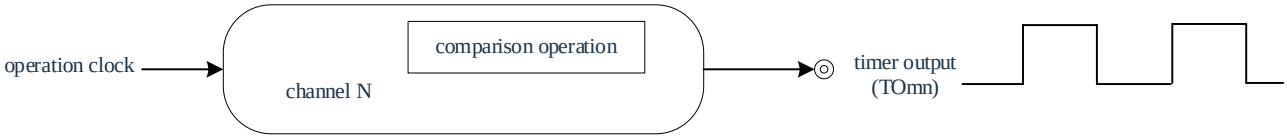
(1) Interval timer

Each timer of a unit can be used as a reference timer that generates an interrupt (INTTMmn) at fixed intervals.



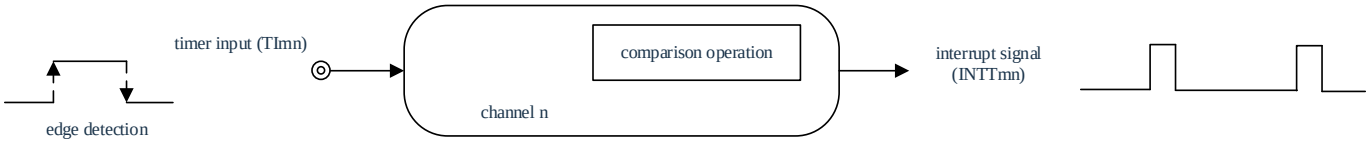

(2) Square wave output

A toggle operation is performed each time INTTMmn interrupt is generated and a square wave with a duty cycle of 50% is output from a timer output pin (TOmn).



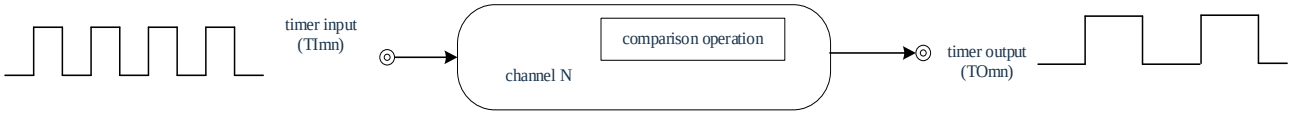

(3) External event counter

The valid edge of the input signal of the timer input pin (TImn) is counted, and if the specified number of times is reached, it can be used as an event counter for generating interrupts.



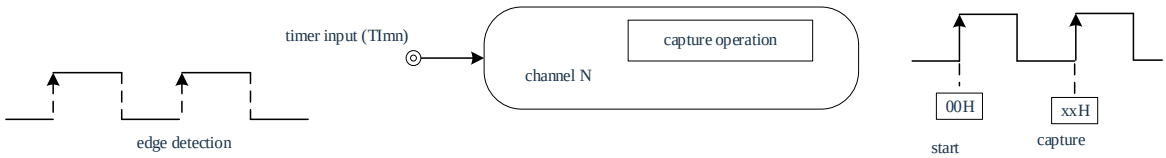

(4) Frequency divider function (channel 0 of unit 0 only)

The input clock from the timer input pin (TI00) is divided and then output from the output pin (TO00).



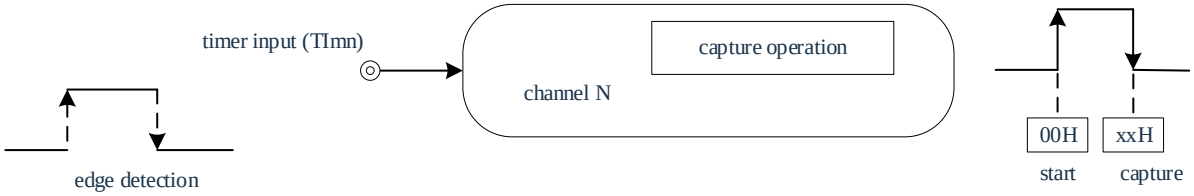

(5) Measurement of input pulse interval

The interval between input pulses is measured by starting counting at the active edge of the input pulse signal at the timer input pin (TImn) and capturing the count value at the active edge of the next pulse.



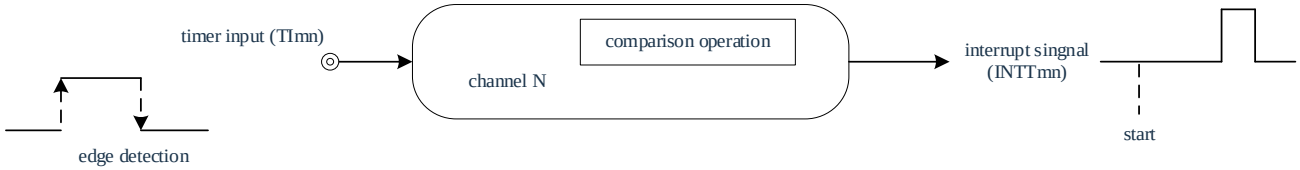

(6) Measurement of the high-/low-level width of the input signal

The high- and low-level width of the input signal is measured by starting the count on one edge of the input signal at the timer input pin (TImn) and capturing the count value on the other edge.




(7) Delay counter

Counting begins on the active edge of the input signal to the timer input pin (TImn) and an interrupt is generated after an arbitrary delay period.




Note 1: m: unit number (m=0) n: channel number (n=0~3)

Note 2: Please refer to Chapter 2 Pin Functions for the configurable timer input/output pins of channel 0~3.

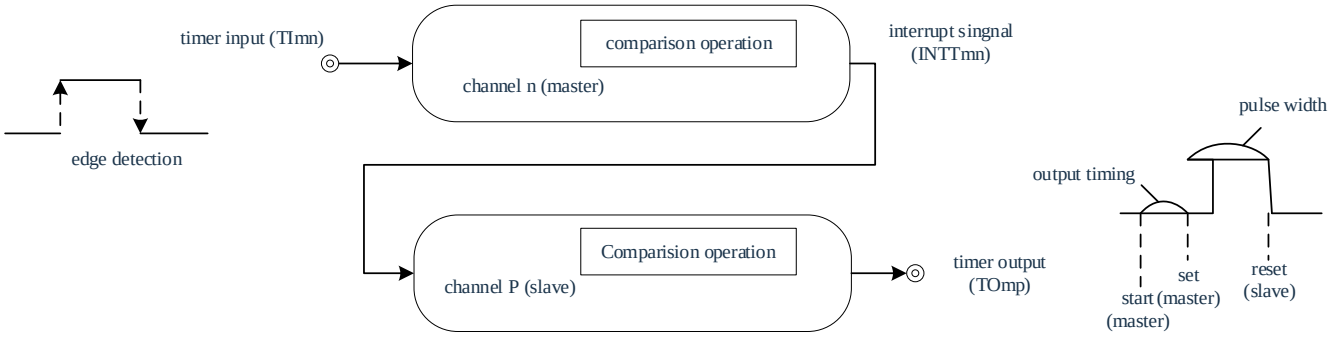
## 5.1.2 Multi-Channel Linkage Operation Functions

The multi-channel linked operation function is a combination of a master channel (the reference timer for the master control cycle) and a slave channel (a timer that operates in compliance with the master channel).

The multi-channel linkage operation function can be used as the following modes.

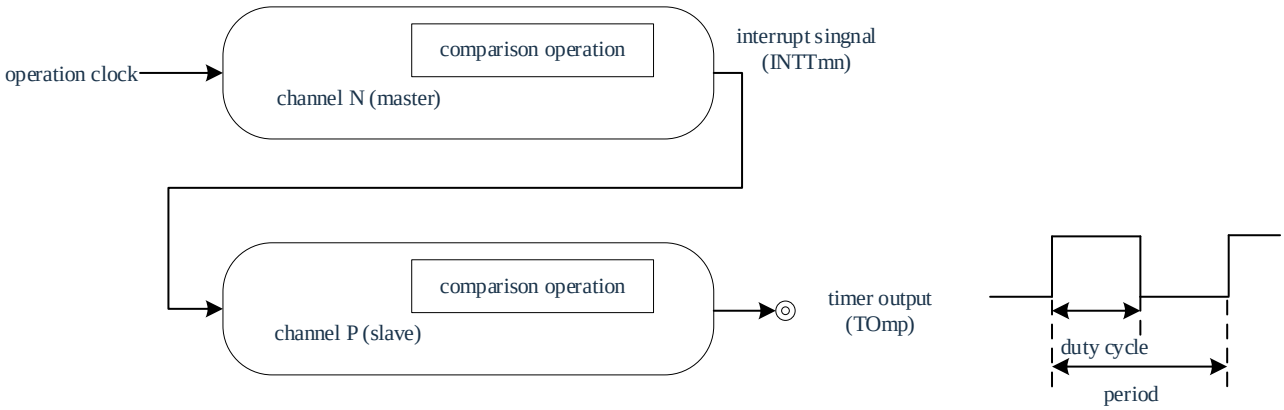
(1) Single trigger pulse output

Using the 2 channels in pairs, a single trigger pulse with arbitrary output timing and pulse width can be generated.



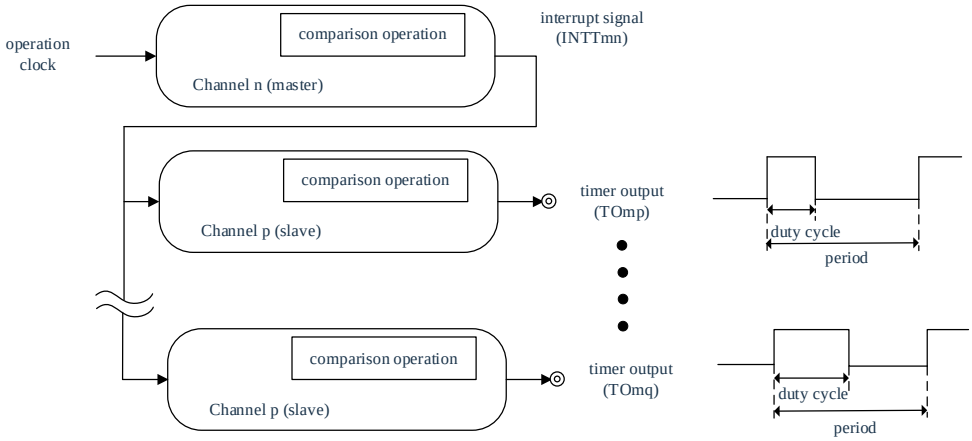

(2) PWM (Pulse Width Modulation) output

Using the 2 channels in pairs, pulses with arbitrary period and duty cycle can be generated.




(3) Multiple PWM (Pulse Width Modulation) output

The PWM function can be extended to generate up to 3 PWM signals of any duty cycle with a fixed period using one master channel and multiple slave channels.




Note 1: Please refer to “5.3.1 Basic rules of multi-channel linkage operation function” for the rule details of multi-channel linkage operation

function.

Note 2: m: unit number (m=0) n: channel number (n=0~3) p, q: slave channel number (n<p<q≤3)

## 5.1.3 8-Bit Timer Operation Function (Channels 1 and 3 Only)

The 8-bit timer operation function makes it possible to use a 16-bit timer channel in a configuration consisting of two 8-bit timer channels. This function can only be used for channels 1 and 3.

Caution: There are several rules for using 8-bit timer operation function.
For details, see 5.3.2 Basic rules of 8-bit timer operation function (channels 1 and 3 only).

## 5.2 Structure of Universal Timer Unit

The universal timer unit consists of the following hardware.

Table 5-1 Structure of universal timer unit

| Item | Structure |
|---|---|
| Counter | Timer count register mn (TCRmn) |
| Register | Timer data register mn (TDRmn) |
| Timer input | TI00~TI03Note1 |
| Timer output | TO00~TO03 Note 1, output control circuit |
| Control register | ＜Registers of unit setting section＞<br>• Peripheral enable register 0 (PER0)<br>• Timer clock select register m (TPSm)<br>• The timer channel enable status register m (TEm)<br>• Timer channel start register m (TSm)<br>• Timer channel stop register m (TTm)<br>• Timer input select register 0 (TIS0) Note2<br>• Timer output enable register m (TOEm)<br>• Timer output register m (TOm)<br>• Timer output level register m (TOLm)<br>• Timer output mode register m(TOMm) |
| | <Register of each channel><br>• Timer mode register mn(TMRmn)<br>• Timer status register mn(TSRmn)<br>• Port mode control register (PMCxx) Note 3<br>• Port mode register (PMxx) Note 3<br>• Port output alternate function configuration register (PxxCFG) Note 3<br>• Port input alternate function configuration register (PStau0tinnCFG) Note 3 |

Note 1: The input/output pins of Universal Timer Unit 0 are multiplexed to the fixed port. For details, refer to “Chapter 2 Pin Functions”.

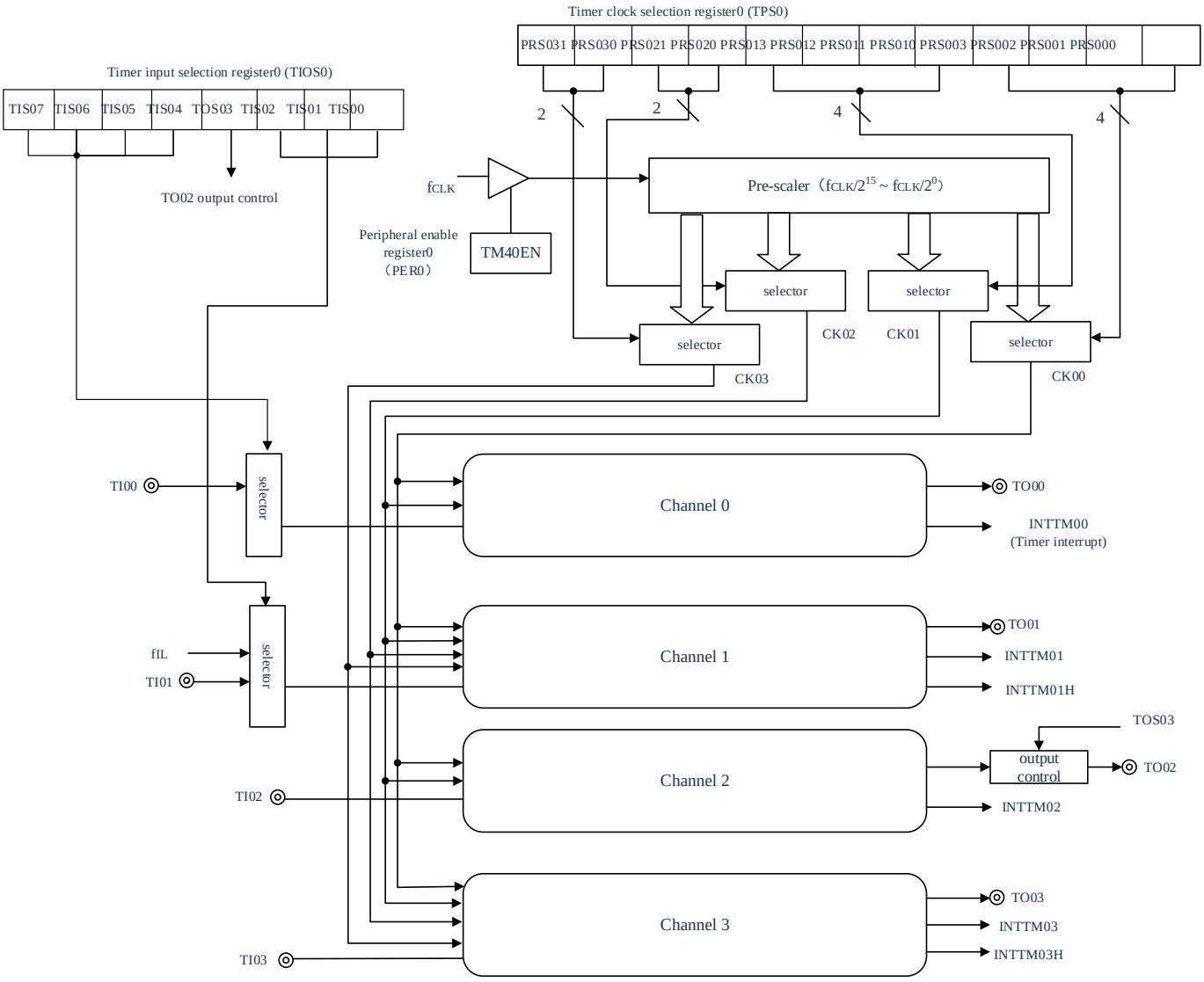
Note 2: Used only for channel selection in Unit 0.

Note 3: Used for the configuration of timer input/output pins of channels 0 to 3. For details, refer to “Chapter 2 Pin Functions”.

Note 4: m: unit number (m=0) n: channel number (n=0 to 3).

The block diagram of the universal timer unit is shown in Figure 5-1.

Figure 5-1 Overall block diagram of universal timer unit 0



## 5.2.1 Register Mapping

(Bass address = 0x4004_1D80) RO: Read only, WO: Write only, R/W: Read/Write

| Register name | Offset address | R/W | Bit width | Description | Reset value |
|---|---|---|---|---|---|
| TCR00 | 0x000 | R | 16 | Timer channel 0 count register | 0xFFFF |
| TCR01 | 0x002 | R | 16 | Timer channel 1 count register | 0xFFFF |
| TCR02 | 0x004 | R | 16 | Timer channel 2 count register | 0xFFFF |
| TCR03 | 0x006 | R | 16 | Timer channel 3 count register | 0xFFFF |
| TMR00 | 0x010 | R/W | 16 | Timer channel 0 mode register | 0x0000 |
| TMR01 | 0x012 | R/W | 16 | Timer channel 1 mode register | 0x0000 |
| TMR02 | 0x014 | R/W | 16 | Timer channel 2 mode register | 0x0000 |
| TMR03 | 0x016 | R/W | 16 | Timer channel 3 mode register | 0x0000 |
| TSR00 | 0x020 | R | 16 | Timer channel 0 status register | 0x0000 |
| TSR01 | 0x022 | R | 16 | Timer channel 1 status register | 0x0000 |
| TSR02 | 0x024 | R | 16 | Timer channel 2 status register | 0x0000 |
| TSR03 | 0x026 | R | 16 | Timer channel 3 status register | 0x0000 |
| TE0 | 0x030 | R | 16 | Timer channel enable status register | 0x0000 |
| TS0 | 0x032 | R/W | 16 | Timer channel start register | 0x0000 |
| TT0 | 0x034 | R/W | 16 | Timer channel stop register | 0x0000 |
| TPS0 | 0x036 | R/W | 16 | Timer clock selection register | 0x0000 |
| TO0 | 0x038 | R/W | 16 | Timer output register | 0x0000 |
| TOE0 | 0x03A | R/W | 16 | Timer output enable register | 0x0000 |
| TOL0 | 0x03C | R/W | 16 | Timer output level register | 0x0000 |
| TOM0 | 0x03E | R/W | 16 | Timer output mode register | 0x0000 |
| TOM0L | 0x03E | R/W | 8 | Timer output mode register lower 8 bits | 0x00 |
| TDR00 | 0x198 | R/W | 16 | Timer channel 0 data register | 0x0000 |
| TDR01 | 0x19A | R/W | 16 | Timer channel 1 data register | 0x0000 |
| TDR02 | 0x1E4 | R/W | 16 | Timer channel 2 data register | 0x0000 |
| TDR03 | 0x1E6 | R/W | 16 | Timer channel 3 data register | 0x0000 |

(Bass address = 0x4004_0470) RO: Read only, WO: Write only, R/W: Read/Write

| Register name | Offset address | R/W | Bit width | Description |
|---|---|---|---|---|
| TIOS0 | 0x004 | R/W | Timer input/output selection register | 0x0 |

## 5.2.2 Timer Count Register mn (TCRmn)

The TCRmn register is a 16-bit read-only register that counts the count clock. The count is incremented or decremented synchronously with the rising edge of the count clock.

The operation mode is selected by the MDmn3 to MDmn0 bits of the Timer Mode Register mn (TMRmn) to switch between incremental and decremental counting (refer to 5.2.6: Timer Mode Register mn (TMRmn)).

Table 5-2 Table of Timer count register mn (TCRmn)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:0 | TCRmn | Count clock count register (read-only) | 0xFFFF |

Note: m: unit number (m=0) n: channel number (n=0 to 3)

The count value can be read by reading the timer count register mn (TCRmn).

The count value becomes “FFFFH” in the following cases.

- When a reset signal is generated
- When clearing the TM40EN bit of the peripheral enable register 0 (PER0)
- At the end of the count of the slave channel in PWM output mode
- At the end of the count of the slave channel in delayed count mode
- At the end of counting of master/slave channels in single trigger pulse output mode
- At the end of the count of the slave channel in the multiple PWM output mode

The count value becomes “0000H” in the following cases.

- When input starts triggering in capture mode
- At the end of the capture in capture mode

Caution: Even if the TCRmn register is read, the count value is not captured to the timer data register mn(TDRmn).

As shown below, the read values of the TCRmn register vary depending on the operating mode and operating state.

Table 5-3 Read value of timer count register mn (TCRmn) in each running mode

| Operation mode | Counting method | Timer count register mn (TCRmn) read value[Note] | | | |
|---|---|---|---|---|---|
| | | Value if the operation mode was changed after releasing reset | Counting pause Value at (TTmn = 1) | Counting pause (TTmn=1) after changing the value of the operating mode | Wait after a single count The value at the start of the trigger |
| Interval timer mode | Count down | 0xFFFF | value when stopped | undefined | — |
| Capture mode | Count up | 0x0000 | value when stopped | undefined | — |
| Event count mode | Count down | 0xFFFF | value when stopped | undefined | — |
| Single count mode | Count down | 0xFFFF | value when stopped | undefined | 0xFFFF |
| Capture & single count mode | Count up | 0x0000 | value when stopped | undefined | TDRmn register capture value +1 |

Note 1: m: unit number (m=0) n: channel number (n=0 to 3)

Note 2: It indicates the read value of the TCRmn register when channel n is in the timer stop state (TEmn=0) and the count enable state (TSmn=1). Hold this value in the TCRmn register until counting starts.

## 5.2.3 Timer Data Register mn (TDRmn)

This is a 16-bit register from which a capture function and a compare function can be selected. The capture or compare function can be switched by selecting an operation mode by using the MDmn3 to MDmn0 bits of timer mode register mn (TMRmn).

The value of the TDRmn register can be changed at any time.

This register can be read or written in 16-bit units.

In addition, for the TDRm1 and TDRm3 registers, while in the Timer4 8-bit timer mode (when the SPLIT bits of timer mode registers m1 and m3 (TMRm1, TMRm3) are 1), it is possible to rewrite the data in 8-bit units, with TDRm1H and TDRm3H used as the higher 8 bits, and TDRm1L and TDRm3L used as the lower 8 bits.

Reset signal generation clears this register to 0000H.

Table 5-4 Channel 0 timer data register TDR00

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:0 | TDR00 | Timer channel 0 data register | 0x0000 |

Table 5-5 Channel 1 timer data register TDR01

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:8 | TDR01H | Timer channel 1 data register bit15:8 | 0x00 |
| 7:0 | TDR01L | Timer channel 1 data register bit7:0 | 0x00 |

Table 5-6 Channel 2 timer data register TDR02

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:0 | TDR02 | Timer channel 2 data register | 0x00 |

Table 5-7 Channel 3 timer data register TDR03

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:8 | TDR03H | Timer channel 3 data register bit15:8 | 0x00 |
| 7:0 | TDR03L | Timer channel 3 data register bit7:0 | 0x00 |

(1) When timer data register mn (TDRmn) is used as compare register

Counting down is started from the value set to the TDRmn register. When the count value reaches 0000H, an interrupt signal (INTTMmn) is generated. The TDRmn register holds its value until it is rewritten.

Note: Even if a capture trigger signal is input, the TDRmn register set to the compare function does not perform capture operation.

(2) When timer data register mn (TDRmn) is used as capture register

The count value of timer count register mn (TCRmn) is captured to the TDRmn register when the capture trigger is input.

A valid edge of the TImn pin can be selected as the capture trigger. This selection is made by timer mode register mn (TMRmn).

Note: m: unit number (m=0) n: channel number (n=0~3)

## 5.2.4 Peripheral Enable Register 0 (PER0)

The PER0 register is a register that sets whether to enable or disable the supply of clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to hardware that is not in use.

To use universal timer unit 0, bit0 (TM40EN) must be set to "1". The PER0 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the PER0 register changes to "00H".

Table 5-8 Table of peripheral enable register 0 (PER0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | LSITIMEREN | Control of LSITIMER input clock supply (available in power down sleep mode)<br>0: Stop to supply the input clock, the SFR used by the LSITIMER cannot be written.<br>1: Supply the input clock, the SFR used by the LSITIMER can be written. | 0 |
| 6:5 | -- | Reserved | 0x0 |
| 4 | IICAEN | Control of IICA input clock supply<br>0: Stop to supply the input clock, the SFR used by the IICA cannot be written.<br>1: Supply the input clock, the SFR used by the IICA can be written. | 0 |
| 3:1 | -- | Reserved | 0x0 |
| 0 | TM40EN | Control of universal timer unit 0 input clock supply<br>0: Stop to supply the input clock, the SFR used by the universal timer unit 0 cannot be written.<br>1: Supply the input clock, the SFR used by the universal timer unit 0 can be written. | 0 |

Note: To set the universal timer unit, the following registers must be set with the TM40EN bit at "1". When the TM40EN bit is "0", the values of the Timer Array Unit's control registers are initialized, and write operations are ignored (timer input/output select register 0 (TIOS0), port mode control register (PMCx), port mode register (PMx), and port alternate function configuration register (PxxCFG) are excluded).

- Timer status register mn (TSRmn)
- Timer channel enable status register m (TEm)
- Timer channel start register m (TSm)
- Timer channel stop register m (TTm)
- Timer output enable register m (TOEm)
- Timer output register m (TOm)
- Timer output level register m (TOLm)
- Timer output mode register m (TOMm)

## 5.2.5 Timer Clock Selection Register m (TPSm)

The TPSm register is a 16-bit register that selects the two or four common operating clocks (CKm0, CKm1, CKm2, CKm3) provided to each channel. CKm0 is selected via bits 3~0 of the TPSm register, and CKm1 is selected via bits 7~4 of the TPSm register. In addition, only channel 1 and channel 3 can select CKm2 and CKm3, and CKm2 is selected via bits 9~8 of the TPSm register, and CKm3 is selected via bits 13 and 12 of the TPSm register.

The TPSm register in timer operation can only be rewritten in the following cases.

If the PRSm00 to PRSm03 bits can be rewritten (n = 0 to 3):

All channels for which CKm0 is selected as the operation clock (CKSmn1, CKSmn0 = 0, 0) are stopped (TEmn = 0).

If the PRSm10 to PRSm13 bits can be rewritten (n = 0 to 3):

All channels for which CKm2 is selected as the operation clock (CKSmn1, CKSmn0 = 0, 1) are stopped (TEmn = 0).

If the PRSm20 and PRSm21 bits can be rewritten (n = 1, 3):

All channels for which CKm1 is selected as the operation clock (CKSmn1, CKSmn0 = 1, 0) are stopped (TEmn = 0).

If the PRSm30 and PRSm31 bits can be rewritten (n = 1, 3):

All channels for which CKm3 is selected as the operation clock (CKSmn1, CKSmn0 = 1, 1) are stopped (TEmn = 0).

The TPSm register can be set by a 16-bit memory manipulation instruction. After a reset signal is generated, the value of the TPSm register changes to “0000H”.

Table 5-9 Table of timer clock select register m (TPSm)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:14 | - | Set to 0 | 0x0 |
| 13:12 | CKm3 | Timer operation clock selection CKm3:<br>00: $f_{clk}/2^8$<br>01: $f_{clk}/2^{10}$<br>10: $f_{clk}/2^{12}$<br>11: $fclk/2^{14}$ | 0x0 |
| 11:10 | - | Set to 0 | 0x0 |
| 9:8 | CKm2 | Timer operation clock selection CKm2:<br>00: $f_{clk}/2$<br>01: $f_{clk}/2^2$<br>10: $f_{clk}/2^4$<br>11: $f_{clk}/2^6$ | 0x0 |
| 7:4 | CKm1 | Timer operation clock selection $f_{clk}/2^{CKm1}$ | 0x0 |
| 3:0 | CKm0 | Timer operation clock selection $f_{clk}/2^{CKm0}$ | 0x0 |

Note 1: $f_{CLK}$: CPU/peripheral hardware clock frequency

Note 2: The clock waveform selected by the TPSm register is high for only 1 $F_{CLK}$ cycle from the rising edge. For details, refer to “5.4.1 Counting Clock ($F_{TCLK}$)”.

Note 3: Bits 15, 14, 11 and 10 must be set to “0”.

Note 4: In case of changing the clock selected as $F_{CLK}$ (changing the value of the system clock control register (CKC)), the universal timer unit must be stopped (TTm=000FH). The universal timer unit needs to be stopped even when the operation clock ($F_{MCK}$) is selected or when the active edge of the TImn pin input signal is used.

Note 5: If $F_{CLK}$ (undivided) is selected as the operation clock (CKmk) and TDRnm is set to “0000H” (m=0, 1, n=0~3), the interrupt request of universal timer unit cannot be used.

If channels 1 and 3 are used in 8-bit timer mode and CKm2 and CKm3 are used as the operation clocks, the interval times shown in the table below can be realized with the interval timer function.

Table 5-10 Interval time that can be set by operation clocks CKSm2 and CKSm3

| Clock | | Interval time[Note] ($F_{CLK}$=32MHz) | | | |
|---|---|---|---|---|---|
| | | 10us | 100us | 1ms | 10ms |
| CKm2 | $f_{CLK}/2$ | ○ | — | — | — |
| | $f_{CLK}/2^2$ | ○ | — | — | — |
| | $f_{CLK}/2^4$ | ○ | ○ | — | — |
| | $f_{CLK}/2^6$ | ○ | ○ | — | — |
| CKm3 | $f_{CLK}/2^8$ | — | ○ | ○ | — |
| | $f_{CLK}/2^{10}$ | — | ○ | ○ | — |
| | $f_{CLK}/2^{12}$ | — | — | ○ | ○ |
| | $f_{CLK}/2^{14}$ | — | — | ○ | ○ |

Note 1: ○The margin is within 5%.

Note 2: $F_{CLK}$: CPU/peripheral hardware clock frequency

Note 3: For details about the $F_{CLK}/2^r$ waveform selected for the TPSm register, refer to “5.4.1 Count Clock ($F_{TCLK}$)”.

## 5.2.6 Timer Mode Register mn (TMRmn)

The TMRmn register is a register that sets the operation mode of channel n. It performs the selection of the operation clock ($F_{MCK}$), the selection of the count clock, the selection of master/slave, the selection of the 16-bit/8-bit timer (limited to channel 1 and channel 3), the setting of the start trigger and the capture trigger, the selection of the effective edge of the timer input, and the operation modes (interval, capture, event counter, single count, capture & single count) settings.

It is prohibited to rewrite the TMRmn register during operation (TEmn=1). However, bit7 and bit6 (CISmn1, CISmn0) can be rewritten during part of the function operation (TEmn=1) (for details, refer to "5.7 Independent Channel Operation Function of Universal Timer Unit" and "5.8 Multi-Channel Operation Function of Universal Timer Unit").

The TMRmn register is set by a 16-bit memory manipulation instruction. After a reset signal is generated, the value of the TMRmn register changes to "0000H".

Note: Bit11 of the TMRmn register varies from channel to channel.

TMRm2: MASTERmn bit (n=2)

TMRm1, TMRm3: SPLITmn bit (n=1, 3)

TMRm0: Fixed to "0".

Table 5-11 Timer channel 0 mode register TMR00

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:14 | CKS001- CKS000 | Selection of channel n operation clock ($F_{MCK}$):<br>00: The operation clock CKm0 set by the timer clock selection register m (TPSm)<br>01: The operation clock CKm2 set by the timer clock selection register m (TPSm)<br>10: The operation clock CKm1 set by the timer clock selection register m (TPSm)<br>11: The operation clock CKm30 set by the timer clock selection register m (TPSm)<br>The operation clock ($F_{MCK}$) is used for edge detection circuits. The sample clock and count clock ($F_{TCLK}$) are generated by setting the CCSmn bit. Only Channel 1 and Channel 3 can select operation clocks CKm2 and CKm3. | 0x0 |
| 13 | - | Set to 0 | 0 |
| 12 | CCS00 | Selection of channel n count clock ($F_{TCLK}$)<br>0: CKS000 bit and CKS001 bit specified operation clock ($F_{MCK}$)<br>1: The active edge of the input signal selected by TIOS0<br>Counting clocks ($F_{TCLK}$) are used in counters, output control circuits, and interrupt control circuits. | 0 |
| 11 | - | Set to 0 | 0 |
| 10:8 | STS002- STS000 | Start trigger and capture trigger settings for channel n<br>000: Only software triggering is active at the start (no other trigger source is selected).<br>001: Use the active edge of the TI00 pin input for start triggering and capture triggering.<br>010: Use the double edges of the TI00 pin input for start triggering and capture triggering respectively.<br>100: Use interrupt signals from the master channel (in the case of slave channels with multi-channel linkage operation function).<br>Other Settings are prohibited | 0x0 |
| 7:6 | CIS001- CIS000 | TI00 active edge selection<br>00: Falling edge<br>01: Rising edge | 0x0 |

| | | | |
|---|---|---|---|
| | | 10: Double edge (when measuring low level width)<br>Start trigger: falling edge, capture trigger: rising edge | |
| | | 11: Double edge (when measuring high level width)<br>Start trigger: rising edge, capture trigger: falling edge | |
| 5:4 | 0 | Reserved | 0x0 |
| 3:0 | MD003- MD000 | Setting of channel n operation mode and interrupt<br>0000: Interval timer mode, no timer interrupt is generated at the start of counting.<br>0001: Interval timer mode, a timer interrupt is generated when counting starts.<br>0100: Capture mode, no timer interrupt is generated when counting starts.<br>0101: Capture mode, a timer interrupt is generated when counting starts.<br>0110: Event counter mode, no timer interrupt is generated when counting starts.<br>1000: Single count mode, the start trigger in the count operation is invalid. No interruption at this time.<br>1001: Single count mode, the start trigger in the count operation is valid. No interruption at this time.<br>1100: Capture & single count mode, no timer interrupt is generated when counting starts, and the start trigger is invalid in couner operation.<br>Other Settings are prohibited | 0x0 |

For a detailed description of MD003- MD000, see the following table

| MD003 | MD002 | MD001 | Setting of channel n operation mode | Corresponding functions | Count operation of TCR |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Interval timer mode | Interval timer/square wave output/<br>Frequency divider function/PWM output (master) | Count down |
| 0 | 1 | 0 | Capture mode | Measurement of input pulse interval | Count up |
| 0 | 1 | 1 | Event counter mode | External event counter | Count down |
| 1 | 0 | 0 | Single count mode | Delay counter/single trigger pulse output/PWM output (slave) | Count down |
| 1 | 1 | 0 | Capture & Single count mode | Measurement of the high- and low-level width of the input signal | Count up |
| Other than the above | | | Settings are prohibited. | | |
| The operation of each mode varies depending on MD000 bit (see the table below). | | | | | |

| Operation mode<br>(Value set by the MD003 to MD001 bits (see table above)) | MD 000 | Setting of starting counting and interrupt |
|---|---|---|
| • Interval timer mode (0, 0, 0)<br>• Capture mode (0, 1, 0) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). |
| | 1 | A timer interrupt is generated when counting starts (the output of the timer also changes). |
| • Event counter mode (0, 1, 1) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). |
| • Single count mode Note 1 (1, 0, 0) | 0 | The start trigger in the count operation is invalid. No interruption at this time. |
| | 1 | The start trigger in the count operation is valid Note 2. No interruption at this time. |
| • Capture & single count mode (1, 1, 0) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). The start trigger in the count operation is invalid.<br>No interruption at this time. |

Note 1: In single count mode, the interrupt output (INTTM00) and TO00 output at the start of counting are not controlled.

Note 2: If a start trigger is generated during operation (TS00=1), the counter is initialized and counting is restarted (no interrupt request is generated).

Table 5-12 Timer channel 1 mode register TMR01

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:14 | CKS011- CKS010 | Selection of channel n operation clock ($F_{MCK}$)<br>00: The operation clock CKm0 set by the timer clock select register m (TPSm).<br>01: The operation clock CKm2 set by the timer clock select register m (TPSm).<br>10: The operation clock CKm1 set by the timer clock select register m (TPSm).<br>11: The operation clock CKm3 set by the timer clock select register m (TPSm).<br>The operation clock ($F_{MCK}$) is used for edge detection circuits. The sample clock and count clock ($F_{TCLK}$) are generated by setting the CCSmn bit. Only Channel 1 and Channel 3 can select operation clocks CKm2 and CKm3. | 0x0 |
| 13 | - | Set to 0. | 0 |
| 12 | CCS01 | Selection of channel n count clock ($F_{TCLK}$)<br>0: The operation clock specified by CKS010 bit and CKS011 bit ($F_{MCK}$)<br>1: The active edge of the input signal selected by TIOS0<br>The counting clock ($F_{TCLK}$) is used in counters, output control circuits, and interrupt control circuits. | 0 |
| 11 | SPLIT00 | Operation selection of 8-bit timer/16-bit timer for channel 1<br>0: Used as a 16-bit timer. (Used as a slave channel for independent channel operation or multi-channel linkage operation)<br>1: Used as an 8-bit timer. | 0 |
| 10:8 | STS012- STS010 | Start trigger and capture trigger settings for channel n<br>000: Only software triggering is active at the start (no other trigger source is selected).<br>001: Use the active edge of the TI01 pin input for start triggering and capture triggering.<br>010: Use the double edges of the TI01 pin input for start triggering and capture triggering respectively.<br>100: Use interrupt signals from the master channel (in the case of slave channels with multi-channel linkage operation function).<br>Other Settings are prohibited. | 0x0 |
| 7:6 | CIS011- CIS010 | TI01 pin active edge selection<br>00: Falling edge<br>01: Rising edge<br>10: Double edges (when measuring low-level width)<br>Start trigger: falling edge, capture trigger: rising edge<br>11: Double edges (when measuring high-level width)<br>Start trigger: rising edge, capture trigger: falling edge | 0x0 |
| 5:4 | - | Reserved | 0x0 |
| 3:0 | MD013- MD010 | Setting of channel n operation mode and interrupt<br>0000: Interval timer mode, no timer interrupt is generated at the start of counting.<br>0001: Interval timer mode, a timer interrupt is generated when counting starts.<br>0100: Capture mode, no timer interrupt is generated when counting starts.<br>0101: Capture mode, a timer interrupt is generated when counting starts.<br>0110: Event counter mode, no timer interrupt is generated when counting starts.<br>1000: Single count mode, the start trigger in the count operation is invalid. No interruption at this time.<br>1001: Single count mode, the start trigger in the count operation is valid. No interruption at this time. | 0x0 |

| | | | | |
|---|---|---|---|---|
| | | 1100: | Capture & single count mode, no timer interrupt is generated when counting starts. | |
| | | Other | Settings are prohibited. | |

For a detailed description of MD013- MD010, see the following table

| MD013 | MD012 | MD011 | Setting of channel n operation mode | Corresponding function | Count operation of TCR |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Interval timer mode | Interval timer/square wave output/ Frequency divider function/PWM output (master) | Count down |
| 0 | 1 | 0 | Capture mode | Measurement of input pulse interval | Count up |
| 0 | 1 | 1 | Event counter mode | External event counter | Count down |
| 1 | 0 | 0 | Single count mode | Delay counter/single trigger pulse output/PWM output (slave) | Count down |
| 1 | 1 | 0 | Capture & Single count mode | Measurement of the high- and low-level width of the input signal | Count up |
| Other than the above | | | Settings are prohibited. | | |
| The operation of each mode varies depending on MD010 bit (see the table below). | | | | | |

| Operation mode (Value set by the MD013 to MD011 bits (see table above)) | MD010 | Setting of starting counting and interrupt |
|---|---|---|
| • Interval timer mode (0, 0, 0)<br>• Capture mode (0, 1, 0) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). |
| | 1 | A timer interrupt is generated when counting starts (the output of the timer also changes). |
| • Event counter mode (0, 1, 1) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). |
| • Single count mode Note 1 (1, 0, 0) | 0 | The start trigger in the count operation is invalid. No interruption at this time. |
| | 1 | The start trigger in the count operation is valid Note 2. No interruption at this time. |
| • Capture & single count mode (1, 1, 0) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). The start trigger in the count operation is invalid. No interruption at this time. |

Note 1: In single count mode, the interrupt output (INTTM01) and TO01 output at the start of counting are not controlled.

Note 2: If a start trigger is generated during operation (TS01=1), the counter is initialized and counting is restarted (no interrupt request is generated).

Table 5-13 Timer channel 2 mode register TMR02

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:14 | CKS021- CKS020 | Selection of channel n operation clock ($F_{MCK}$)<br>00: The operation clock CKm0 set by the timer clock select register m (TPSm)<br>01: The operation clock CKm2 set by the timer clock select register m (TPSm)<br>10: The operation clock CKm1 set by the timer clock select register m (TPSm)<br>11: The operation clock CKm3 set by the timer clock select register m (TPSm)<br>The operation clock ($F_{MCK}$) is used for edge detection circuits. The sample clock and count clock ($F_{TCLK}$) are generated by setting the CCSmn bit. Only Channel 1 and Channel 3 can select operation clocks CKm2 and CKm3. | 0x0 |
| 13 | - | Set to 0. | 0 |
| 12 | CCS02 | Selection of channel n count clock ($F_{TCLK}$)<br>0: The operation clock specified by CKS020 bit and CKS021 bit ($f_{MCK}$)<br>1: The active edge of the input signal selected by TI02<br>The counting clock ($F_{TCLK}$) is used in counters, output control circuits, and interrupt control circuits. | |
| 11 | MASTER | Selection of independent channel operation/multi-channel operation (slave or master) for channel 2<br>0: Used as a slave channel for independent or multi-channel operation.<br>Channel 0 is fixed to “0” (since channel 0 is the highest bit channel, it is used as the master channel regardless of the setting of this bit).<br>1: Used as a master control channel for multi-channel operation.<br>Channel 0 is fixed to “0” (since channel 0 is the highest bit channel, it is used as the master channel regardless of the setting of this bit).<br>Only channel 2 can be set as the master channel (MASTERmn=1). | 0 |
| 10:8 | STS022- STS020 | Start trigger and capture trigger settings for channel n<br>000: Only software triggering is active at the start (no other trigger source is selected).<br>001: Use the active edge of the TI02 pin input for start triggering and capture triggering.<br>010: Use the double edges of the TI02 pin input for start triggering and capture triggering respectively.<br>100: Use interrupt signals from the master channel (in the case of slave channels with multi-channel linkage operation function).<br>Other Settings are prohibited. | 0x0 |
| 7:6 | CIS021- CIS020 | TI02 pin active edge selectio<br>00: Falling edge<br>01: Rising edge<br>10: Double edges (when measuring low level width)<br>Start trigger: falling edge, capture trigger: rising edge<br>11: Double edges (when measuring high level width)<br>Start trigger: rising edge, capture trigger: falling edge | 0x0 |
| 5:4 | - | Reserved | 0x0 |
| 3:0 | MD023- MD020 | Setting of channel n operation mode and interrupt<br>0000: Interval timer mode, no timer interrupt is generated at the start of counting.<br>0001: Interval timer mode, a timer interrupt is generated when counting starts.<br>0100: Capture mode, no timer interrupt is generated when | 0x0 |

| | | | | |
|---|---|---|---|---|
| | | | counting starts. | |
| | | 0101: | Capture mode, a timer interrupt is generated when counting starts. | |
| | | 0110: | Event counter mode, no timer interrupt is generated when counting starts. | |
| | | 1000: | Single count mode, the start trigger in the count operation is invalid. No interruption at this time. | |
| | | 1001: | Single count mode, the start trigger in the count operation is valid. No interruption at this time. | |
| | | 1100: | Capture & single count mode, no timer interrupt is generated when counting starts. The start trigger in the counting operation is invalid. | |
| | | Other | Settings are prohibited. | |

For a detailed description of MD023-MD020, see the following table

| MD023 | MD022 | MD021 | Setting of channel n operation mode | Corresponding function | Count operation of TCR |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Interval timer mode | Interval timer/square wave output/ Frequency divider function/PWM output (master) | Count down |
| 0 | 1 | 0 | Capture mode | Measurement of input pulse interval | Count up |
| 0 | 1 | 1 | Event counter mode | External event counter | Count down |
| 1 | 0 | 0 | Single count mode | Delay counter/single trigger pulse output/PWM output (slave) | Count down |
| 1 | 1 | 0 | Capture & Single count mode | Measurement of the high- and low-level width of the input signal | Count up |
| Other than the above | | | Settings are prohibited. | | |
| The operation of each mode varies depending on MD020 bit (see the table below). | | | | | |

| Operation mode (Value set by the MD023 to MD021 bits (see table above)) | MD020 | Setting of starting counting and interrupt |
|---|---|---|
| • Interval timer mode (0, 0, 0)<br>• Capture mode (0, 1, 0) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). |
| | 1 | A timer interrupt is generated when counting starts (the output of the timer also changes). |
| • Event counter mode (0, 1, 1) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). |
| • Single count mode Note 1 (1, 0, 0) | 0 | The start trigger in the count operation is invalid. No interruption at this time. |
| | 1 | The start trigger in the count operation is valid Note 2. No interruption at this time. |
| • Capture & single count mode (1, 1, 0) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). The start trigger in the count operation is invalid. No interruption at this time. |

Note 1: In single count mode, the interrupt output (INTTM02) and TO02 output at the start of counting are not controlled.

Note 2: If a start trigger is generated during operation (TS02=1), the counter is initialized and counting is restarted (no interrupt request is generated).

Table 5-14 Timer channel 3 mode register TMR03

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:14 | CKS031- CKS030 | Selection of channel n operation clock ($F_{MCK}$)<br>00: The operation clock CKm0 set by the timer clock select register m (TPSm)<br>01: The operation clock CKm2 set by the timer clock select register m (TPSm)<br>10: The operation clock CKm1 set by the timer clock select register m (TPSm)<br>11: The operation clock CKm3 set by the timer clock select register m (TPSm)<br>The operation clock ($F_{MCK}$) is used for edge detection circuits. The sample clock and count clock ($F_{TCLK}$) are generated by setting the CCSmn bit. Only Channel 1 and Channel 3 can select operation clocks CKm2 and CKm3. | 0x0 |
| 13 | - | Set to 0. | 0 |
| 12 | CCS03 | Selection of channel n count clock ($F_{TCLK}$)<br>0: The operation clock ($F_{MCK}$) specified by CKS030 bit and CKS031 bit<br>1: The active edge of the TI03 pin input signal<br>The counting clocks ($F_{TCLK}$) is used in counters, output control circuits, and interrupt control circuits. | 0 |
| 11 | SPLIT03 | Operation selection of 8-bit timer/16-bit timer for channel 3<br>0: Used as a 16-bit timer. (Used as a slave channel for independent channel operation or multi-channel linkage operation)<br>1: Used as an 8-bit timer. | 0 |
| 10:8 | STS032- STS030 | Start trigger and capture trigger settings for channel n<br>000: Only software triggering is active at the start (no other trigger source is selected).<br>001: Use the active edge of the TI03 pin input for start triggering and capture triggering.<br>010: Use the double edges of the TI03 pin input for start triggering and capture triggering respectively.<br>100: Use interrupt signals from the master channel (in the case of slave channels with multi-channel linkage operation function).<br>Other Settings are prohibited. | 0x0 |
| 7:6 | CIS031- CIS030 | TI03 pin active edge selection<br>00: Falling edge<br>01: Rising edge<br>10: Double edges (when measuring low level width)<br>Start trigger: falling edge, capture trigger: rising edge<br>11: Double edges (when measuring high level width)<br>Start trigger: rising edge, capture trigger: falling edge | 0x0 |
| 5:4 | - | Reserved | 0x0 |
| 3:0 | MD033- MD030 | Setting of channel n operation mode and interrupt<br>0000: Interval timer mode, no timer interrupt is generated at the start of counting.<br>0001: Interval timer mode, a timer interrupt is generated when counting starts.<br>0100: Capture mode, no timer interrupt is generated when counting starts.<br>0101: Capture mode, a timer interrupt is generated when counting starts.<br>0110: Event counter mode, no timer interrupt is generated when counting starts.<br>1000: Single count mode, the start trigger in the count operation is invalid. No interruption at this time.<br>1001: Single count mode, the start trigger in the count operation | 0x0 |

| | | | | |
|---|---|---|---|---|
| | | | is valid. No interruption at this time. | |
| | | 1100: | Capture & single count mode, no timer interrupt is generated when counting starts. The start trigger in the counting operation is invalid. | |
| | | Other | Settings are prohibited. | |

For a detailed description of MD033-MD030, see the following table

| MD033 | MD322 | MD031 | Setting of channel n operation mode | Corresponding function | Count operation of TCR |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Interval timer mode | Interval timer/square wave output/ Frequency divider function/PWM output (master) | Count down |
| 0 | 1 | 0 | Capture mode | Measurement of input pulse interval | Count up |
| 0 | 1 | 1 | Event counter mode | External event counter | Count down |
| 1 | 0 | 0 | Single count mode | Delay counter/single trigger pulse output/PWM output (slave) | Count down |
| 1 | 1 | 0 | Capture & Single count mode | Measurement of the high- and low-level width of the input signal | Count up |
| Other than the above | | | Settings are prohibited. | | |
| The operation of each mode varies depending on MD030 bit (see the table below). | | | | | |

| Operation mode (Value set by the MD023 to MD021 bits (see table above)) | MD030 000 | Setting of starting counting and interrupt |
|---|---|---|
| • Interval timer mode (0, 0, 0)<br>• Capture mode (0, 1, 0) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). |
| | 1 | A timer interrupt is generated when counting starts (the output of the timer also changes). |
| • Event counter mode (0, 1, 1) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). |
| • Single count mode [Note 1] (1, 0, 0) | 0 | The start trigger in the count operation is invalid. No interruption at this time. |
| | 1 | The start trigger in the count operation is valid[Note 2]. No interruption at this time. |
| • Capture & single count mode (1, 1, 0) | 0 | No timer interrupt is generated when counting starts (the output of the timer does not change). The start trigger in the count operation is invalid. |

Note 1: In single count mode, the interrupt output (INTTM03) and TO03 output at the start of counting are not controlled.

Note 2: If a start trigger is generated during operation (TS03=1), the counter is initialized and counting is restarted (no interrupt request is generated).

## 5.2.7 Timer Status Register mn (TSRmn)

The TSRmn register is a register that indicates the overflow status of the channel n counter.

The TSRmn register is valid only in capture mode (MDmn3~MDmn1=010B) and capture & single count mode (MDmn3~MDmn1=110B). Refer to Table 5-16 for the OVF bit changes and set/clear conditions in each operation mode.

The TSRmn register is read by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of the TSRmn register changes to “0000H”.

Table 5-15 Table of timer status register mn (TSRmn)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:1 | - | Reserved | 0x0 |
| 0 | OVF | Counter overflow status of channel n<br>0: No overflow occurred.<br>1: Overflow occurred.<br>If the OVF bit is “1”, this flag is cleared when the next count does not overflow and the count value is captured (OVF=0). | 0 |

Note: m: unit number (m=0) n: channel number (n=0~3)

Table 5-16 OVF bit change and set/clear conditions in each operation mode

| Timer operation mode | OVF bit | Set/clear conditions |
|---|---|---|
| • Capture mode<br>• Capture & single count mode | Clear | No overflow occurred at the capture. |
| | Set | Overflow occurred at the capture. |
| • Interval timer mode<br>• Event counter mode<br>• Single count mode | Clear | — (N/A) |
| | Set | |

Note: m: unit number (m=0) n: channel number (n=0~3)

## 5.2.8 Timer Channel Enable Status Register m (TEm)

The TEm register is a register that indicates the enable or stop status of each channel timer operation.

Each bit of the TEm register corresponds to each bit of the timer channel start register m (TSm) and timer channel stop register m (TTm). If each bit of the TSm register is “1”, the corresponding bit of the TEm register is “1”. If each bit of the TTm register is “1”, the corresponding bit of the TTm register is cleared to “0”.

The TEm register is read by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of the TEm register changes to “0000H”.

Table 5-17 Table of timer channel enable status register m (TEm)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:12 | - | Reserved | 0x0 |
| 11 | TEH03 | Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 3 is in the 8-bit timer mode<br>0: Operation is stopped<br>1: Operation is enabled | 0 |
| 10 | - | Reserved | 0 |
| 9 | TEH01 | Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 1 is in the 8-bit timer mode<br>0: Operation is stopped<br>1: Operation is enabled | 0 |
| 8:4 | - | Reserved | 0x0 |
| 3:0 | TE03-TE00 | Indication of operation enable/stop status of channel n<br>0: Operation is stopped<br>1: Operation is enabled | 0x0 |

## 5.2.9 Timer Channel Start Register m (TSm)

The TSm register is a trigger register that initializes the timer counter register mn (TCRmn) and sets the start of counting operation for each channel. If each bit is set to "1", the corresponding bit of the timer channel enable status register m (TEm) is set to "1". Since the TSmn bit, the TSHm1 bit and the TSHm3 bit are trigger bits, the TSmn bit, the TSHm1 bit and the TSHm3 bit are cleared immediately if the operation enbale state is changed (TEmn, TEHm1, TEHm3 = 1).

The TSm register is set by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of the TSm register changes to "0000H".

Table 5-18 Table of timer channel start register m (TSm)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:12 | - | Reserved | 0x0 |
| 11 | TSHm3 | Trigger to enable (start) operation of the higher 8-bit timer when channel 3 is in the 8-bit timer mode<br>0: No trigger operation.<br>1: Set the TEHm3 bit to "1" to enter the counting enable state. If the counting of the TCRm3 register is started in the count enable state, the interval timer mode is entered (refer to Table 5-25 of "5.4.2 Start Timing of Counter"). | 0 |
| 10 | - | Reserved | 0 |
| 9 | TSHm1 | Trigger to enable (start) operation of the higher 8-bit timer when channel 1 is in the 8-bit timer mode<br>0: No trigger operation<br>1: Set the TEHm1 bit to "1" to enter the counting enable state. If counting in the TCRm1 register is started in the count enable state, the interval timer mode is entered (refer to Table 5-25 of "5.4.2 Start Timing of Counter"). | 0 |
| 8:4 | - | Reserved | 0x0 |
| 3:0 | TSm3-TSm0 | Operation enable (start) trigger of channel n:<br>0: No trigger operation<br>1: Set the TEmn bit to "1" to enter the counting enable state. The start of counting in the TCRmn register in the count enable state varies with each operation mode (refer to Table 5-25 of "5.4.2 Start Timing of Counter"). When channel 1 and channel 3 are in 8-bit timer mode, TSm1 and TSm3 are operation enable (start) triggers for the lower 8-bit timer. | 0x0 |

Note 1: Bits 15~12, 10, 8~4 must be set to "0".

Note 2: When switching from a non-TImn pin input function to using the TImn pin input function, a wait of 2 system clock cycles ($F_{MCK}$) is required from setting the timer mode register mn (TMRmn) until the TSmn (TSHm1, TSHm3) bit is set to "1".

Note 3: The TSm register always reads "0".

Note 4: m: unit number (m=0)

## 5.2.10 Timer Channel Stop Register m (TTm)

The TTm register is a trigger register to set the count stop of each channel.

If each bit is set to “1”, the corresponding bit in the timer channel enable status register m (TEm) is cleared to “0”. Since the TTmn bit, TTHm1 bit, and TTHm3 bit are trigger bits, the TTmn bit, TTHm1 bit, and TTHm3 bit are cleared immediately if the operation stop state is changed (TEmn, TEHm1, and TEHm3 = 0).

The TTm register is set by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of the TTm register changes to “0000H”.

Table 5-19 Table of timer channel stop register m (TTm)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:12 | - | Reserved | 0x0 |
| 11 | TTHm3 | Trigger to stop operation of the higher 8-bit timer when channel 3 is in the 8-bit timer mode<br>0: No trigger operation<br>1: TEHm3 bit is cleared to 0 and the count operation is stopped. | 0 |
| 10 | - | Reserved | 0 |
| 9 | TTHm1 | Trigger to stop operation of the higher 8-bit timer when channel 1 is in the 8-bit timer mode<br>0: No trigger operation<br>1: TEHm1 bit is cleared to 0 and the count operation is stopped. | 0 |
| 8:4 | - | Reserved | 0x0 |
| 3:0 | TTm3-TTm0 | Operation enable trigger of channel n<br>0: No trigger operation<br>1: TEmn bit is cleared to 0 and the count operation is stopped.<br>This bit is the trigger to stop operation of the lower 8-bit timer for TTm1 and TTm3 when channel 1 or 3 is in the 8-bit timer mode. | 0x0 |

Note 1: Bits 15~12, 10, 8~4 must be set to “0”.

Note 2: The TTm register always reads “0”.

Note 3: m: unit number (m=0)

## 5.2.11 Timer Input/Output Output Select Register (TIOS0)

The TIOS0 register is used to make selections for the inputs and outputs of unit 0. The timer inputs for channel 0 and channel 1 and the timer output for channel 2 of unit 0 are selected. The TIOS0 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the TIOS0 register changes to “00H”.

Table 5-20 Table of timer input/output select register 0 (TIOS0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:5 | TIS07- TIS05 | Selection of timer input used for channel 0<br>0: Input signals for timer input pin (TI00)<br>1: Settings are prohibited. | 0x0 |
| 4 | TIS04 | Selection of timer input used for channel 0<br>0: Input signals selected by TIS07~TIS05<br>1: Settings are prohibited. | 0 |
| 3 | TOS03 | Channel 2 timer output enable<br>0: Output enable<br>1: Output disable (output fixed to 0) | 0 |
| 2:0 | TIS02- TIS00 | Channel 1 timer input selection<br>000: Input signals for timer input pin (TI01)<br>010: Input signals for timer input pin (TI01)<br>011: Input signals for timer input pin (TI01)<br>100: Low-speed on-chip oscillator clock ($F_{IL}$)<br>Other: Settings are prohibited. | 0x0 |

Note: The high-/low-level width of the selected timer inputs needs to be greater than or equal to $1/F_{MCK}$+10ns. Therefore, when $F_{IL}$ is selected as the $F_{CLK}$ (CSS bit of the CKC register =1), the TIS02 bit cannot be set to “1”.

## 5.2.12 Timer Output Enable Register m (TOEm)

The TOEm register is a register that sets to enable or disable the timer output of each channel.

Channel n for which timer output has been enabled becomes unable to rewrite the value of the TOmn bit of timer output register m (TOm) described later by software, and the value reflecting the setting of the timer output function through the count operation is output from the timer output pin (TOmn).

The TOEm register is set by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of the TOEm register changes to “0000H”.

Table 5-21 Table of timer output enable register m (TOEm)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:4 | - | Set to 0. | 0x0 |
| 3:0 | TOEmn | Enable/disbale the timer output of channel n:<br>0: Disable timer output.<br>The operation of the timer is not reflected to the TOmn bit, fixed output. The TOmn bit can be written and the level set by the TOmn bit is output from the TOmn pin.<br>1: Enable timer output<br>The operation of the timer is reflected to the TOmn bit, producing an output waveform. The write of the TOmn bit is ignored. | 0x0 |

Note 1: Bits 15~4 must be set to “0”.

Note 2: m: unit number (m=0) n: channel number (n=0~3)

## 5.2.13 Timer Output Register m (TOm)

The TOm register is a buffer register for each channel timer output.

The bit value of this register is output from the output pin (TOmn) of each channel timer.

The TOmn bit of this register can be rewritten by software only when timer output is disabled (TOEmn=0). When enabling the timer output (TOEmn=1), rewrite operations via software are ignored and its value is changed only by the operation of the timer.

To use the TI00/TO00, TI01/TO01, TI02/TO02, and TI03/TO03 pins as port functions, the corresponding TOmn bit must be set to “0”.

The TOm register is set by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of the TOm register changes to “0000H”.

Table 5-22 Table of timer output register m (TOm)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:4 | - | Set to 0. | 0x0 |
| 3:0 | TOmn | Timer output of channel n:<br>0: The output value of the timer is “0”.<br>1: The output value of the timer is “1”. | 0x0 |

Note 1: m: unit number (m=0) n: channel number (n=0~3)

## 5.2.14 Timer Output Level Register m (TOLm)

The TOLm register is a register that controls the output level of each channel timer.

When timer output (TOEmn=1) is enabled and the multi-channel linkage operation function (TOMmn=1) is used, the set and reset timing of the timer output signal reflects the inverse setting of each channel n performed by this register. In the master channel output mode (TOMmn=0), this register setting is invalid.

The TOLm register is set by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of the TOLm register changes to “0000H”.

Table 5-23 Table of timer output level register m (TOLm)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:4 | - | Set to 0. | 0x0 |
| 3:1 | TOL03- TOL01 | Control of timer output level of channel n:<br>0: Positive logic output (active-high)<br>1: Inverted output (active-low) | 0x0 |
| 0 | - | Reserved, set to 0. | 0 |

Note 1: If the value of this register is rewritten while the timer is operating, the timer output logic is inverted at the next time the timer output signal changes, rather than immediately after the rewrite.

Note 2: m: unit number (m=0) n: channel number (n=0~3)

## 5.2.15 Timer Output Mode Register m (TOMm)

The TOMm register is a register that controls the output mode of each channel timer. When used as an independent channel operation function, the corresponding bit of the using channel should be set to “0”.

When used as a multi-channel linkage operation function (PWM output, single trigger pulse output and multiple PWM output), the corresponding bit of the master channel is “0” and the corresponding bit of the slave channel is “1”.

When the timer output (TOEmn=1) is enabled, the setting of each channel n is reflected in this register during the setting and reseting timing of the timer output signal.

The TOMm register is set by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of the TOMm register changes to “0000H”.

Table 5-24 Table of timer output mode register m (TOMm)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:4 | - | Set to 0. | 0x0 |
| 3:1 | TOL03- TOL01 | Control of channel n timer output mode:<br>0: Master channel output mode (toggle output via timer interrupt request signal (INTTMmn))<br>1: Slave channel output mode (output is set via timer interrupt request signal (INTTMmn) of master channel and output is reset via timer interrupt request signal (INTTMmp) of slave channel) | 0x0 |
| 0 | - | Reserved, set to 0. | 0 |

Note: m: unit number (m=0) n: channel number n=0~3 (master channel: n=0, 2)

p: slave channel

n=0: p=1, 2, 3

n=2: p=3

(For details on the relationship between the master channel and the slave channel, refer to “5.3.1 Basic Rules for Multi-Channel Linkage Operation Function”).

### 5.2.16 Registers Controlling Port Functions of Timer Input/Output Pins

When using the Universal Timer Unit, the output pins of Timer0 are alternated to a fixed port, and the input pins of Timer0 can be configured to any port. For details, refer to “Chapter 2 Pin Functions”.

When multiplexing the timer 0 output pin to a certain port, the corresponding bit in the port mode control register (PMCxx) must be set to “0”, and the port mode register (PMxx) must also be set to “0”. Additionally, the port alternate function configuration register (PxxCFG) should be configured. At this point, the bits in the port register (Pxx) can be either “0” or “1”.

(Example)

When P20 is configured as TO00 and used as a timer output

Set the PMC21 bit of port mode control register 2 to “0”.

Set bit PM21 of port mode register 2 to “0”.

Set port output alternate function configuration register P21CFG to “0x01”.

When using the alternated port of the Timer0 input pin as the timer input, the corresponding bit of the Port Mode Register (PMx) is set to “1”, the bit of the Port Mode Control Register (PMCxx) is set to “0” and set the Port Mode Configuration Register (PStau0tin0_CFG). In this case, the bit of the port register (Pxx) can be “0” or “1”.

(Example)

Using P20/TI00 as a timer input.

Set the bit PMC20 of the Port Mode Control Register 2 to “0”.

Set the bit PM20 of the Port Mode Register 2 to “1”.

Set port input alternate function configuration register PStau0tin0_CFG to “0x20”.

## 5.3 Basic Rules of Universal Timer Unit

### 5.3.1 Basic Rules of Multi-Channel Linkage Operation Function

The multi-channel linkage function is a function that combines a master channel (a reference timer that counts cycles) and a slave channel (a timer that operates in compliance with the master channel), and several rules need to be observed when using it.

The basic rules of the multi-channel linkage operation function are shown below.

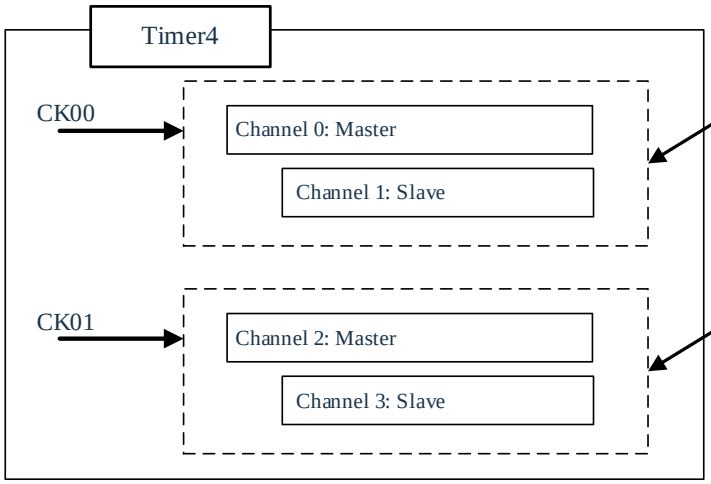
1) Only the even-number channel (channel 0, channel 2) can be set as a master channel.
2) Any channel other than channel 0 can be set as a slave channel.
3) Only the lower channel of the master channel can be set as a slave channel.
   For example, when setting channel 0 as the master channel, it is possible to set the channels starting from channel 1 (channels 1 to 3) as slave channels.
4) Multiple slave channels can be set for 1 master channel.
5) When multiple master channels are used, slave channels that span the master channel cannot be set.
   For example, when setting channel 0 and channel 2 as the master channel, channel 1 can be set as the slave channel of master channel 0, but channel 3 cannot be set as the slave channel of master channel 0.
6) The slave channels linked to the master channel need to be set to the same operating clock. The CKSmn0 bit and CKSmn1 bit (bit15 and bit14 of Timer Mode Register mn (TMRmn)) of the slave channel linked to the master channel need to be the same setting value.
7) The master channel can pass the INTTMmn (interrupt), start software trigger and count clock to the lower channel.
8) The slave channel can use the master channel's INTTMmn (interrupt), start software trigger, and count clocks as source clocks, but cannot pass its own INTTMmn (interrupt), start software trigger, and count clocks to the lower channel.
9) The master channel cannot use the INTTMmn (interrupt), start software trigger and count clocks of other high master channels as source clocks.
10) In order to start the channels to be linked at the same time, the channel start trigger bit (TSmn) of the linked channel needs to be set at the same time.
11) Only all linked channels or the master channel can use the setting of the TSmn bit in the counting operation. It is not possible to use the setting of the TSmn bit of the slave channel only.
12) In order to stop the linked channels at the same time, the channel stop trigger bit (TTmn) of the linked channel needs to be set at the same time.
13) In linked operation, CKm2/CKm3 cannot be selected because the master and slave channels need the same operating clock.
14) The timer mode register m0 (TMRm0) has no master bit and is fixed to “0”. However, since channel 0 is the highest bit channel, it can be used as the master channel during linkage operation.

The basic rules of the multi-channel linkage operation function are the rules applicable to the group of channels (a collection of master and slave channels that form a multi-channel linkage operation function).

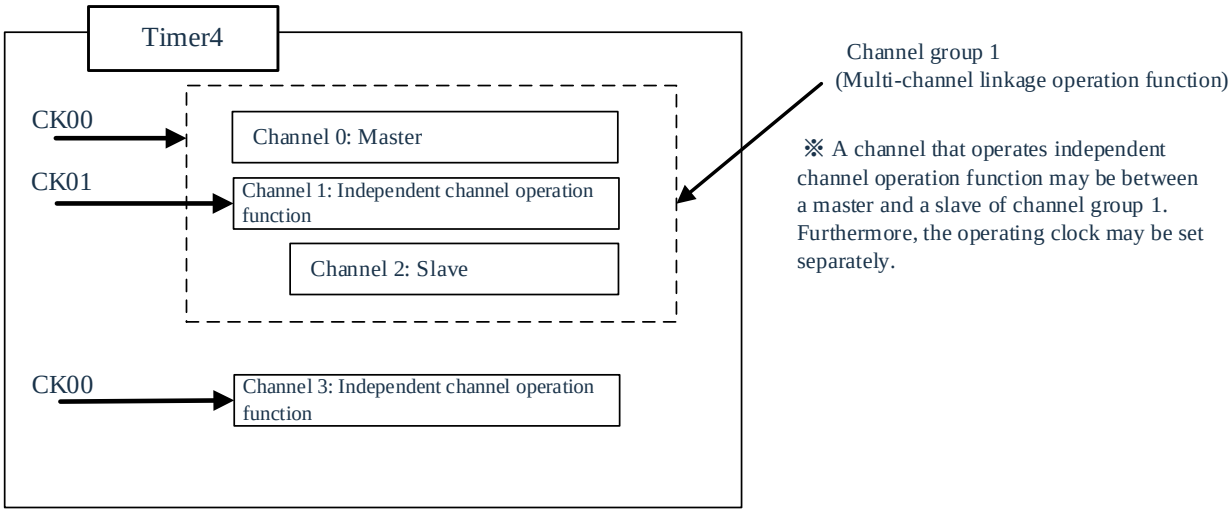
If you set 2 or more channel groups that are not linked to each other, the above basic rules do not apply to the channel groups.

Note: m: unit number (m=0) n: channel number (n=0~3)

Example 1




Example 2



## 5.3.2 Basic Rules of 8-Bit Timer Operation Function (Channels 1 and 3 Only)

The 8-bit timer operation function makes it possible to use a 16-bit timer channel in a configuration consisting of two 8-bit timer channels.

This function can only be used for channels 1 and 3, and there are several rules for using it.

The basic rules for this function are as follows:

1) The 8-bit timer operation function applies only to channels 1 and 3.
2) When using 8-bit timers, set the SPLIT bit of timer mode register mn (TMRmn) to 1.
3) The higher 8 bits can be operated as the interval timer function.
4) At the start of operation, the higher 8 bits output INTTMm1H (an interrupt) (which is the same operation performed when MDmn0 is set to 1).
5) The operation clock of the higher 8 bits is selected according to the CKSmn1 and CKSmn0 bits of the lower-bit TMRmn register.
6) For the higher 8 bits, the TSHm1/TSHm3 bit is manipulated to start channel operation and the TTHm1/TTHm3 bit is manipulated to stop channel operation. The channel status can be checked using the TEHm1/TEHm3 bit.
7) The lower 8 bits operate according to the TMRmn register settings. The following three functions support operation of the lower 8 bits:
   - Interval timer function
   - External event counter function
   - Delay count function
8) For the lower 8 bits, the TSm1/TSm3 bit is manipulated to start channel operation and the TTm1/TTm3 bit is manipulated to stop channel operation. The channel status can be checked using the TEm1/TEm3 bit.
9) During 16-bit operation, manipulating the TSHm1, TSHm3, TTHm1, and TTHm3 bits is invalid. The TSm1, TSm3, TTm1, and TTm3 bits are manipulated to operate channels 1 and 3. The TEHm3 and TEHm1 bits are not changed.
10) For the 8-bit timer function, the linkage operation functions (single pulse, PWM, and multiple PWM) cannot be used.

Note: unit number (m=0) n: channel number (n=1, 3)

# 5.4 Operation of Counter

## 5.4.1 Count Clock ($F_{TCLK}$)

The count clock of the universal timer unit ($F_{TCLK}$) can be selected by the CCSmn bit of the timer mode register mn (TMRmn) for any of the following clocks:

① The CKSmn0 bit and CKSmn1 bit specified operation clock ($F_{MCK}$)

② The active edge of the TImn pin input signal

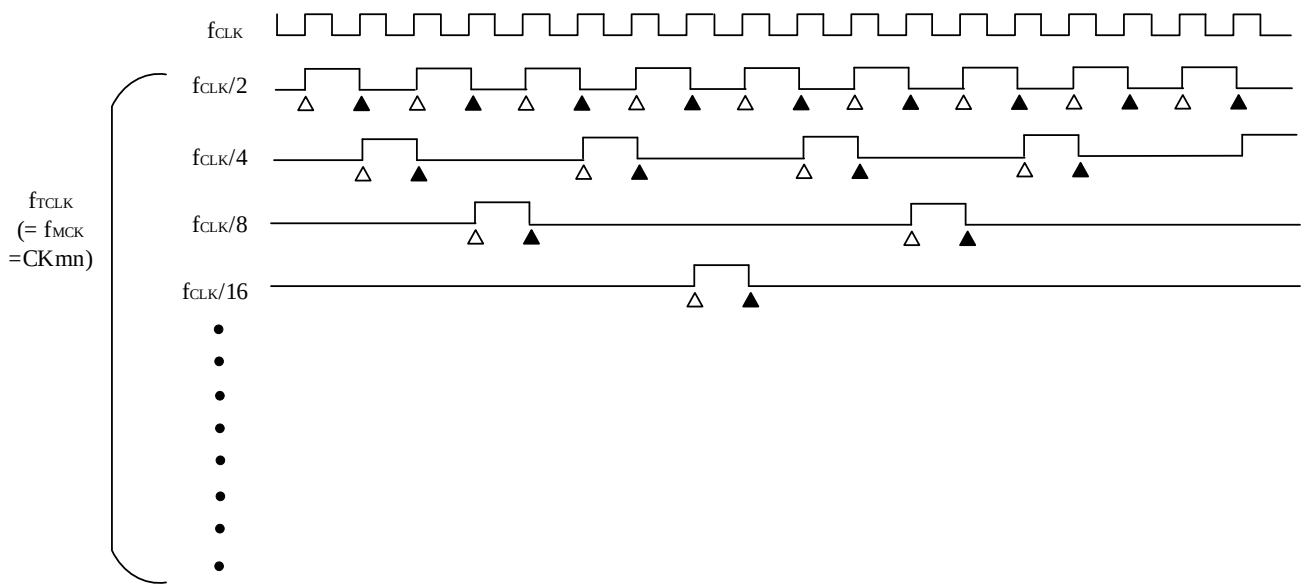
The universal timer unit is designed to operate synchronously with $F_{CLK}$, so the timing of the count clock ($F_{TCLK}$) is as follows.

(1) When operation clock ($F_{MCK}$) specified by the CKSmn0 and CKSmn1 bits is selected (CCSmn = 0)

According to the setting of timer clock selection register m (TPSm), the counting clock ($F_{TCLK}$) is $F_{CLK}$ ~$F_{CLK}$ /$2^{15}$. However, when the frequency division of $F_{CLK}$ is selected, the clock selected by TPSm register is a signal that has only 1 $F_{CLK}$ cycle of high level from the rising edge. When $F_{CLK}$ is selected, it is fixed to high level.

In order to obtain synchronization with $F_{CLK}$, timer count register mn (TCRmn) delays the counting by one $F_{CLK}$ clock from the rising edge of the counting clock, which is called “counting at the rising edge of the counting clock” for convenience.

Figure 5-2 Timing of $F_{CLK}$ and count clock ($F_{TCLK}$) (When CCSmn = 0)



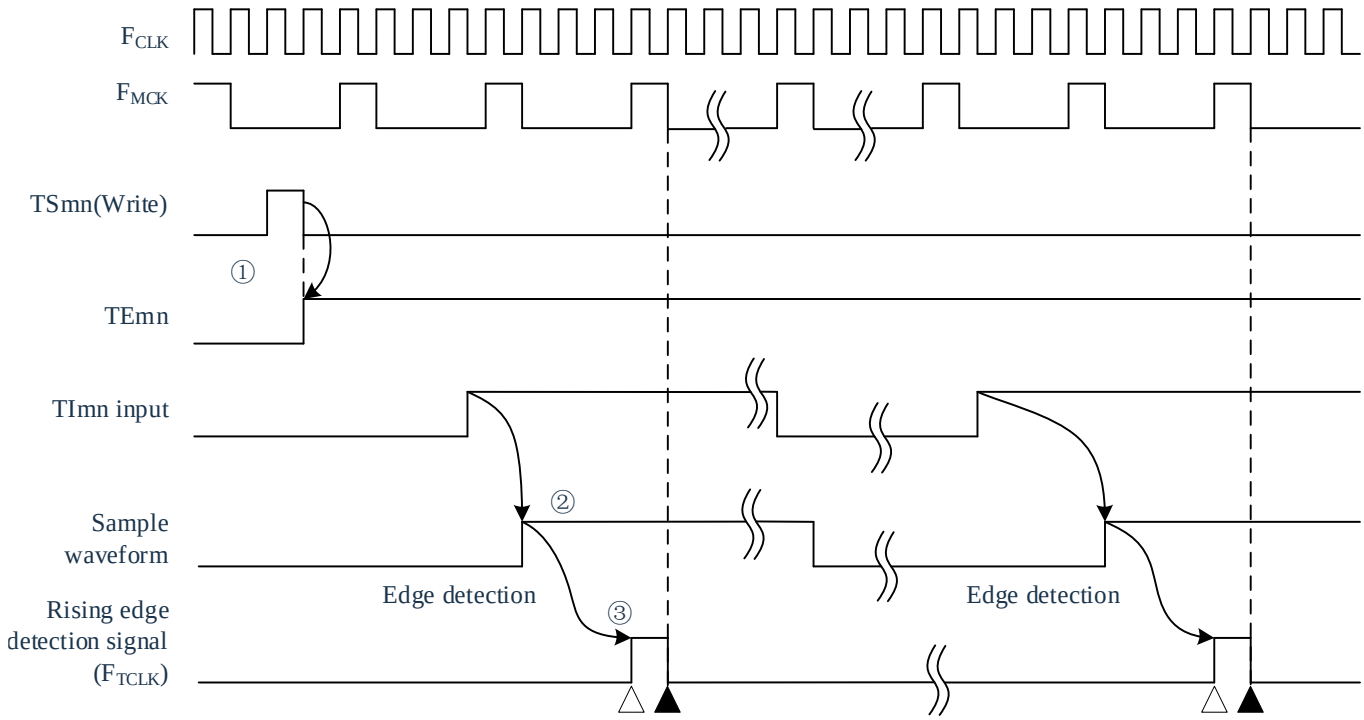

Note 1: △: Rising edge of the count clock; ▲: Synchronization, increment/decrement of counter

Note 2: $f_{CLK}$: CPU/peripheral hardware clock

(2) When valid edge of input signal via the TImn pin is selected (CCSmn = 1)

The count clock ($F_{TCLK}$) is a signal that detects an active edge of the TImn pin input signal and is synchronized with the next $F_{MCK}$ rising edge. In fact, this is a signal delayed by 1~2 $F_{MCK}$ clocks compared to the input signal of the TImn pin. In order to obtain synchronization with $F_{CLK}$, the timer count register mn (TCRmn) delays the count by one $F_{CLK}$ time from the rising edge of the count clock, which is referred to as "counting at the effective edge of the TImn pin input signal" for convenience.

Figure 5-3 Timing of the counting clock ($F_{TCLK}$) (CCSmn=1)




① Setting TSmn bit to 1 enables the timer to be started and to become wait state for valid edge of input signal via the TImn pin.

② The rise of input signal via the TImn pin is sampled by $F_{MCK}$.

③ The edge is detected by the rising of the sampled signal and the detection signal (count clock) is output.

Note 1: △: Rising edge of the count clock; ▲: Synchronization, increment/decrement of counter

Note 2: $f_{CLK}$: CPU/peripheral hardware clock

$f_{MCK}$: Operation clock of channel n

Note 3: The same waveforms are used for the measurement of the input pulse interval, the high and low measurement of the input signal, the delay counter and the TImn input for the single trigger pulse output function.

## 5.4.2 Start Timing of Counter

The timer count register mn (TCRmn) enters the operation enable state by setting TSmn bit of the timer channel start register m (TSm).

Execution from the counting enable state to the start of the timer count register mn (TCRmn) is shown in Table 5-25.

Table 5-25 Operation from the counting enable state to the start of the timer count register mn (TCRmn)

| Timer operation mode | Operation after setting TSmn bit to "1" |
|---|---|
| • Interval timer mode | No operation is performed from the detection of the start trigger (TSmn=1) until the count clock is generated.<br>The value of the TDRmn register is loaded into the TCRmn register by the first count clock and decremented by subsequent count clocks (refer to "5.4.3(1) Operation of the interval timer mode"). |
| • Event counter mode | The value of the TDRmn register is loaded into the TCRmn register by writing a "1" to the TSmn bit.<br>If the input edge of TImn is detected, the count is decremented by the subsequent count clocks. (Refer to "5.4.3(2) Operation of the event counter mode"). |
| • Capture mode | No operation is performed from the time the start trigger is detected until the count clock is generated.<br>The "0000H" is loaded into the TCRmn register by the first count clock, and incremental counting is performed by the subsequent count clocks (refer to "5.4.3(3) Operation of the capture mode (input pulse interval measurement)"). |
| • Single count mode | By writing "1" to the TSmn bit while the timer is stopped (TEmn=0), it enters the wait state for the start of the trigger. No operation is performed from the time the start trigger is detected until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first count clock, and decremental counting by subsequent count clocks (refer to "5.4.3(4) Operation of the single count mode"). |
| • Capture & single count mode | By writing "1" to the TSmn bit while the timer is stopped (TEmn=0), it enters the wait state for the start of the trigger. No operation is performed from the time the start trigger is detected until the count clock is generated. The "0000H" is loaded into the TCRmn register by the first count clock, and incremental counting is performed by the subsequent count clocks (refer to "5.4.3(5) Operation of capture & single count mode (measurement of high-level width"). |

## 5.4.3 Operation of Counter

The following describes the counter operation for each mode.

(1) Operation of interval timer mode

① The operation enable state is entered by writing "1" to the TSmn bit (TEmn=1). The timer count register mn (TCRmn) remains at its initial value until a count clock is generated.

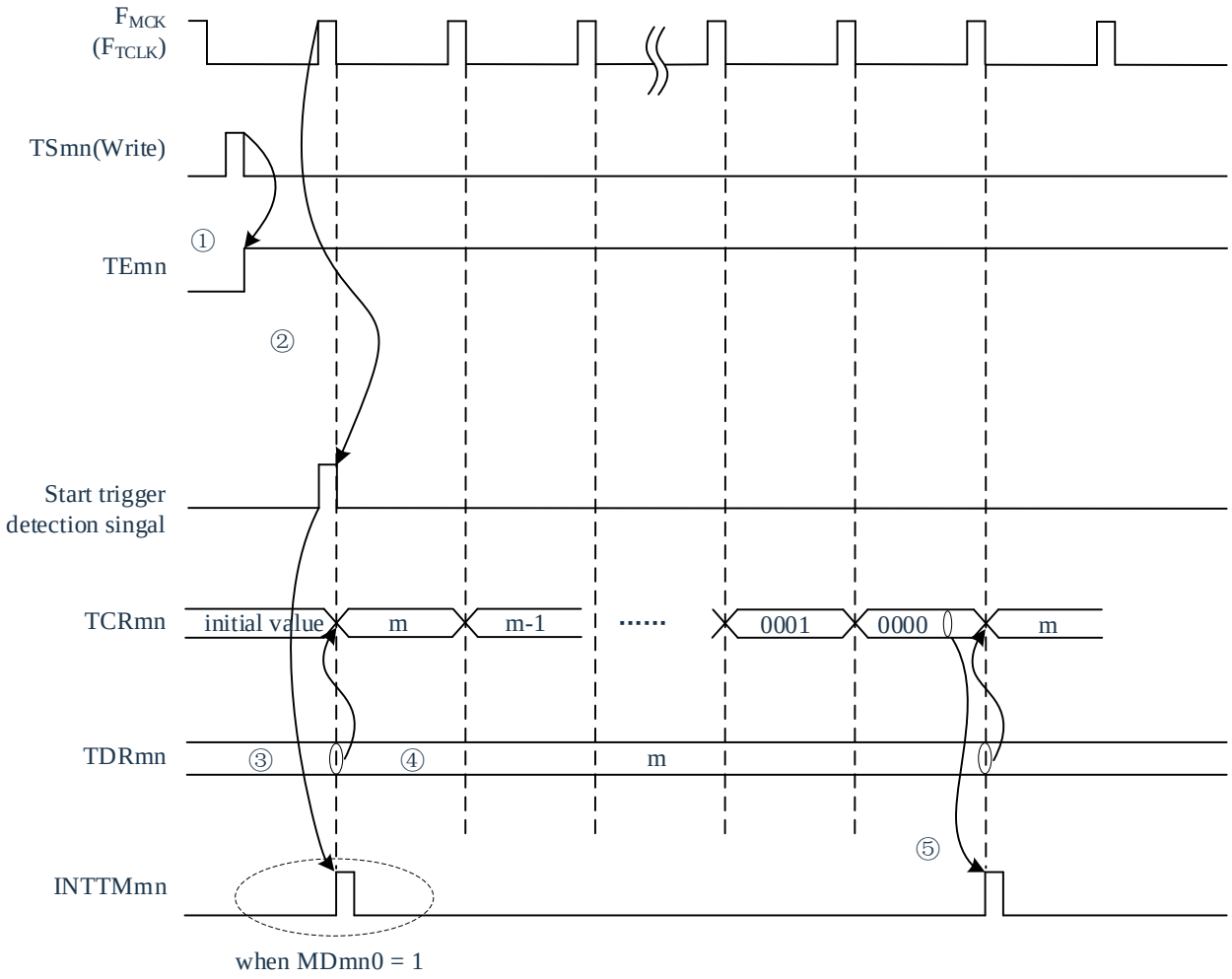
② A start trigger signal is generated by enabling the 1st count clock ($F_{MCK}$) after the operation.

③ When MDmn0 bit is "1", INTTMmn is generated by the start trigger signal.

④ The value of timer data register mn (TDRmn) is loaded into the TCRmn register by enabling the 1st count clock after the operation, and counting starts in interval timer mode.

If the TCRmn register decrements to "0000H", INTTMmn is generated by the next count clock ($F_{MCK}$) and continues counting after loading the value of timer data register mn (TDRmn) into the TCRmn register.

Figure 5-4 Operation timing (interval timer mode)




Note 1: In the first cycle operation of count clock after writing the TSmn bit, an error at a maximum of one clock is generated since count start delays until count clock has been generated. When the information on count start timing is necessary, an interrupt can be generated at count start by setting MDmn0 = 1.

Note 2: $F_{CLK}$, the start trigger detection signal, and INTTMmn become active between one clock in synchronization with $F_{CLK}$.

(2) Operation of event counter mode

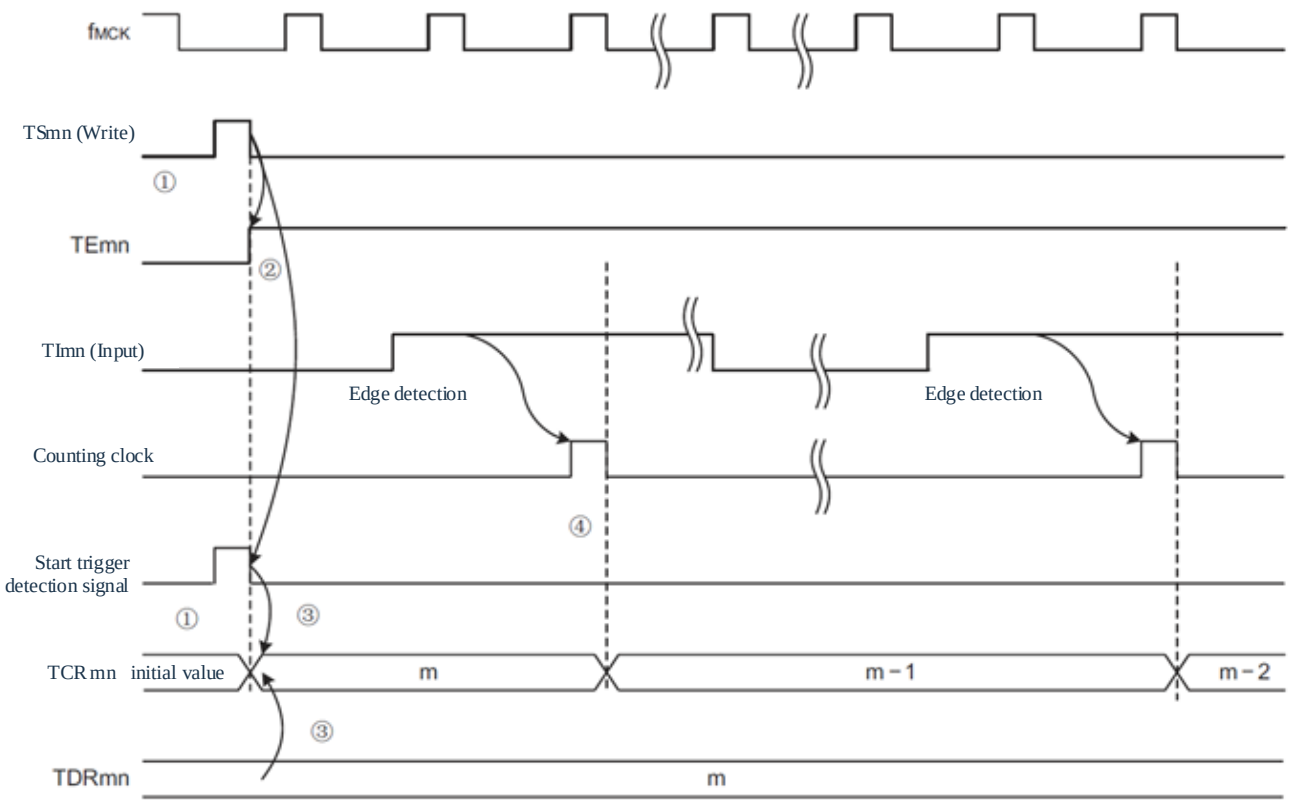
① The timer count register mn (TCRmn) holds its initial value while operation is stopped (TEmn=0).

② The operation enable state is enabled by writing "1" to the TSmn bit (TEmn=1).

③ The value of timer data register mn (TDRmn) is loaded into the TCRmn register while both the TSmn and TEmn bits are changed to "1" and counting begins.

Thereafter, the value of the TCRmn register is counted decreasingly by the count clock at the active edge of the TImn input.

Figure 5-5 Operation timing (event counter mode)




Note: The 1 cycle error is because the TImn input is not synchronized with the count clock ($F_{MCK}$).

(3) Operation of capture mode (interval measurement of input pulses)

① The operation enable state is entered by writing "1" to the TSmn bit (TEmn=1).

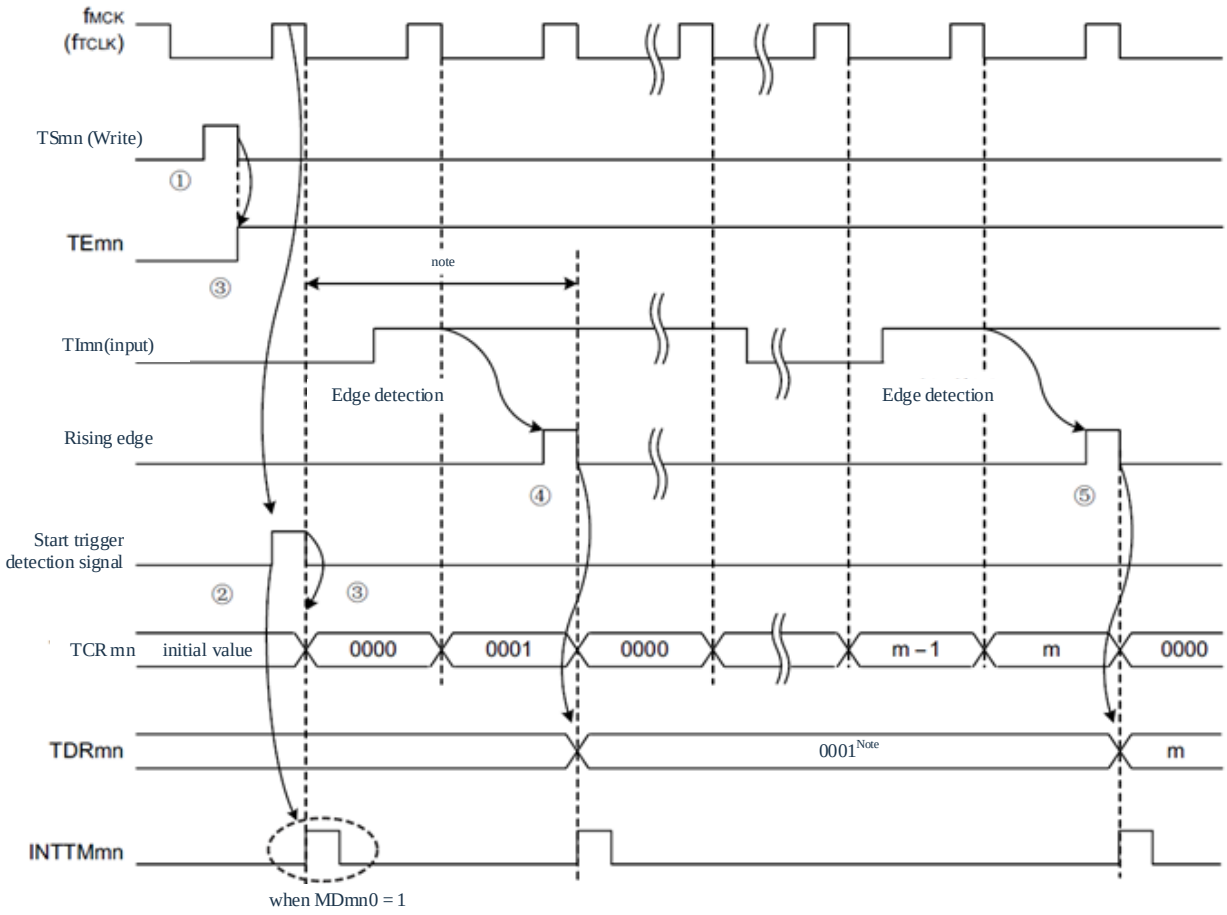
② The timer count register mn (TCRmn) remains at its initial value until a count clock is generated.

③ A start trigger signal is generated by enabling the 1st count clock ($F_{MCK}$) after the operation. Then, the "0000H" is loaded into the TCRmn register and counting starts in capture mode (INTTMmn is generated by the start trigger signal when MDmn0 bit is "1").

④ If an active edge of TImn input is detected, the value of TCRmn register is captured to TDRmn register and INTTMmn interrupt is generated. The capture value is meaningless at this point. The TCRmn register continues counting from the "0000H".

⑤ If an active edge of the next TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and the INTTMmn interrupt is generated.

Figure 5-6 Operation timing (capture mode: interval measurement of input pulses)




Note 1: When the clock is input to TImn (with trigger) before the start, the count is started by detecting the trigger even if no edge is detected, so the capture value at the 1st capture (④) is not a pulse interval (in this example, 0001: 2 clock intervals) and must be ignored.

Note 2: The 1st count clock cycle runs after the TSmn bit is written and delays the start of counting before generating the count clock, an error of up to 1 clock cycle is generated. Also, if you need information about the start of the count timing, set MDmn0 to "1" so that an interrupt can be generated at the start of the count.

Note 3: The 1 cycle error is because the TImn input is not synchronized with the count clock ($F_{MCK}$).

(4) Operation of single count mode

① The operation enable state is entered by writing "1" to the TSmn bit (TEmn=1).

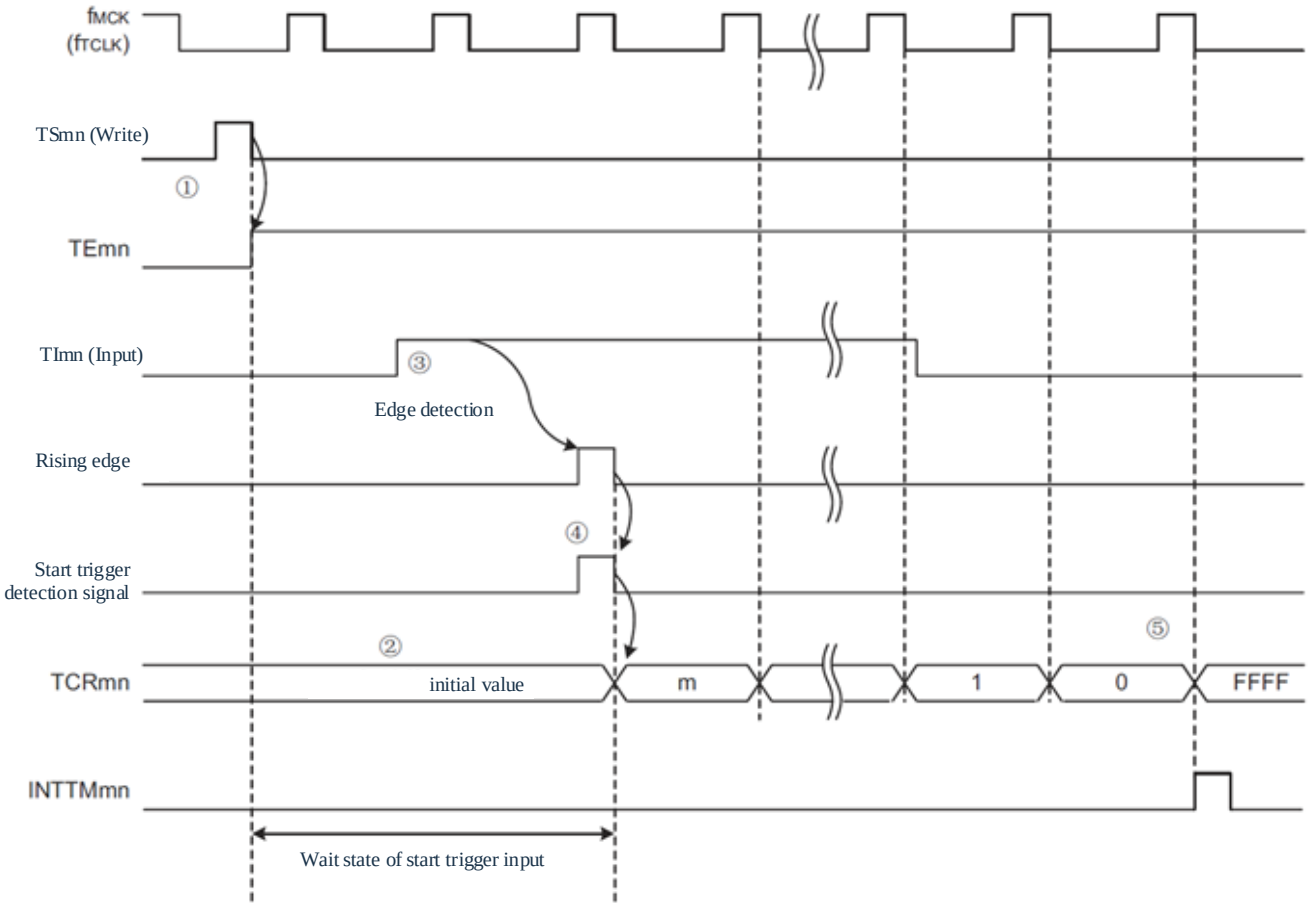
② The timer count register mn (TCRmn) remains the initial value until a start trigger signal is generated.

③ Detects the rising edge of the TImn input.

④ The value (m) of the TDRmn register is loaded into the TCRmn register after a start trigger signal is generated, and counting begins.

⑤ When the TCRmn register decrements to "0000H", the INTTMmn interrupt is generated and the value of TCRmn register changes to "FFFFH" and stop counting.

Figure 5-7 Operation timing (single count mode)




Note 1: The 1 cycle error is because the TImn input is not synchronized with the count clock ($F_{MCK}$).

(5) Operation of capture & single count mode (measurement of high-level width)

① The operation enable state is entered by writing "1" to the TSmn bit of the timer channel start register m (TSm) (TEmn=1).

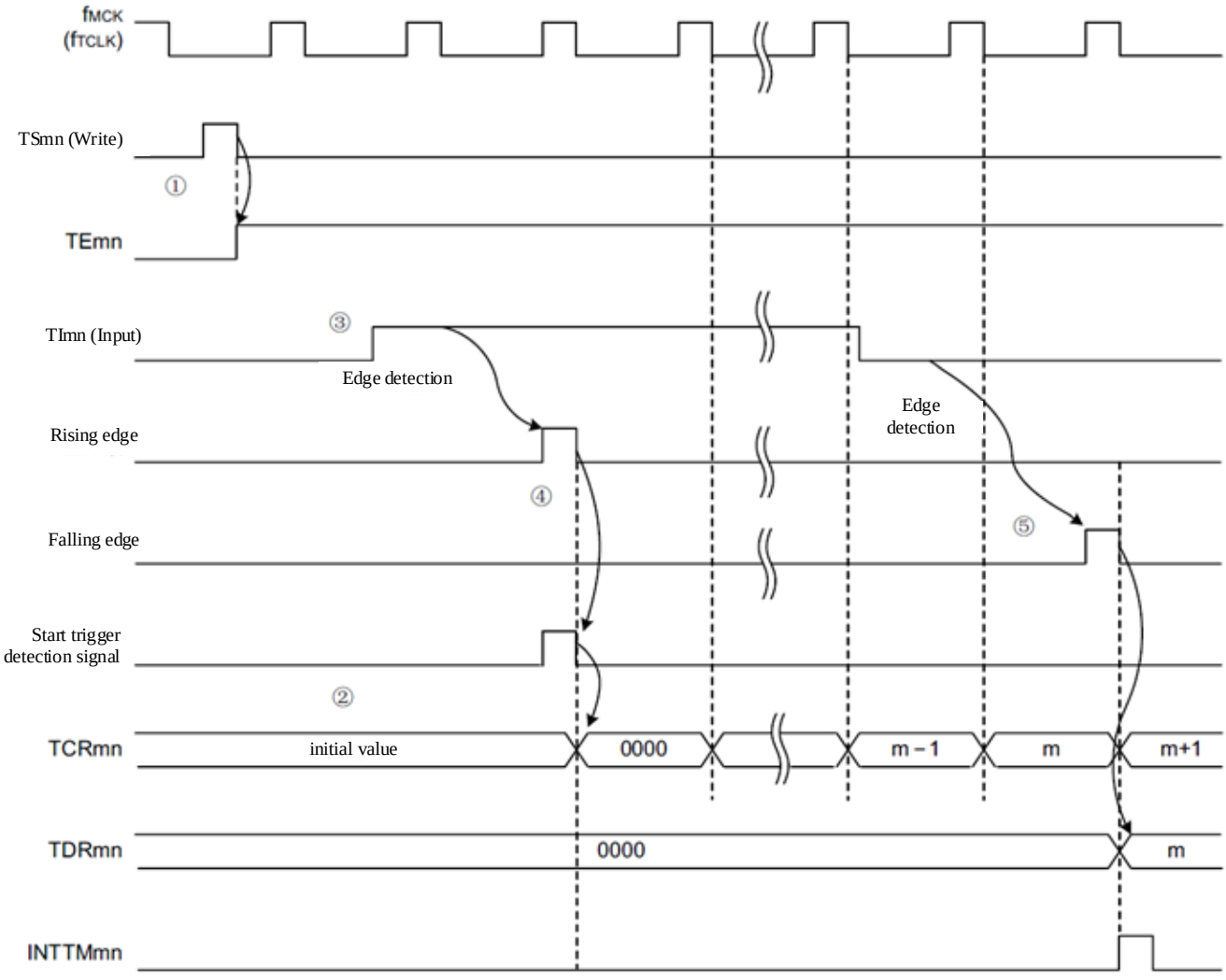
② The timer count register mn (TCRmn) remains the initial value until a start trigger signal is generated.

③ Detects the rising edge of the TImn input.

④ After the start trigger signal is generated, "0000H" is loaded into the TCRmn register and counting starts.

⑤ If the falling edge of TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and an INTTMmn interrupt is generated.

Figure 5-8 Operation tiing (capture & single count mode: measurement of high-level width)

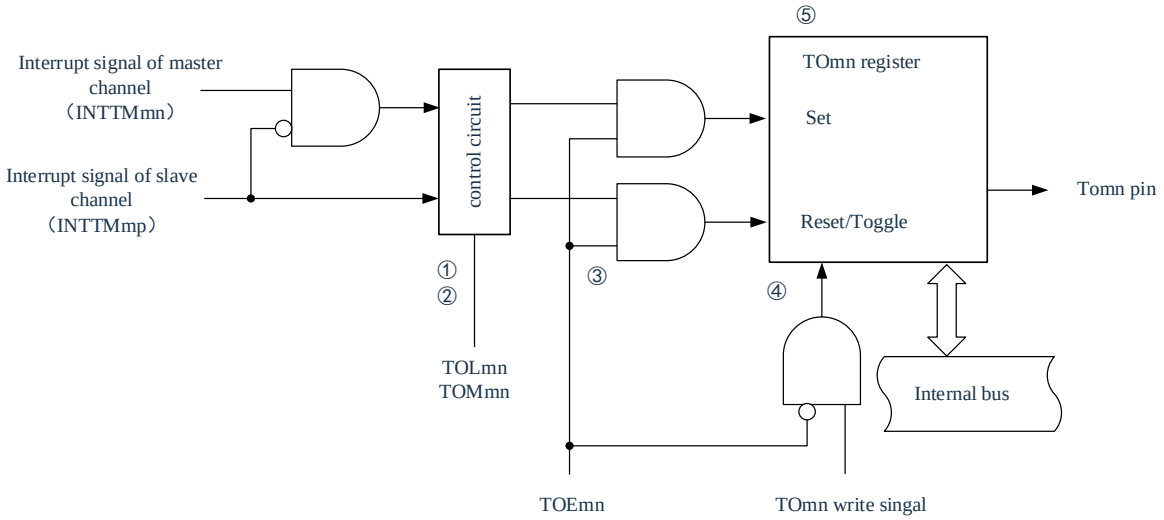



Note: The 1 cycle error is because the TImn input is not synchronized with the count clock ($F_{MCK}$).

# 5.5 Channel Output (TOmn Pin) Control

## 5.5.1 Tomn Pin Output Circuit Configuration

Figure 5-9 Output circuit configuration




① When the TOMmn bit is “0” (master channel output mode), the setting value of timer output level register m (TOLm) is ignored and only INTTMmp (slave channel timer interrupt) is passed to timer output register m (TOm).

② When the TOMmn bit is “1” (slave channel output mode), INTTMmn (master channel timer interrupt) and INTTMmp (slave channel timer interrupt) are passed to the TOm register.

At this time, the TOLm register becomes valid and the signals are controlled as follows:

When TOLmn = 0: Positive logic output (INTTMmn → set, INTTMmp → reset)

When TOLmn = 1: Negative logic output (INTTMmn → reset, INTTMmp → set)

When INTTMmn and INTTMmp are simultaneously generated, (0% output of PWM), INTTMmp (reset signal) takes priority, and INTTMmn (set signal) is masked.

③ In the state of enabling timer output (TOEmn=1), INTTMmn (master channel timer interrupt) and INTTMmp (slave channel timer interrupt) are passed to TOm register. Writing to the TOm register (TOmn write signal) is invalid.

When the TOEmn bit is “1”, the output of the TOmn pin is not changed except for the interrupt signal.

To initialize the output level of the TOmn pin, you need to write a value to the TOm register after setting it to disable the timer output (TOEmn=0).

④ Writing to the TOmn bit for the object channel (TOmn write signal) is valid when the timer output is disabled (TOEmn=0). When the timer output is disabled (TOEmn=0), INTTMmn (master channel timer interrupt) and INTTMmp (slave channel timer interrupt) are not passed to the TOm register.

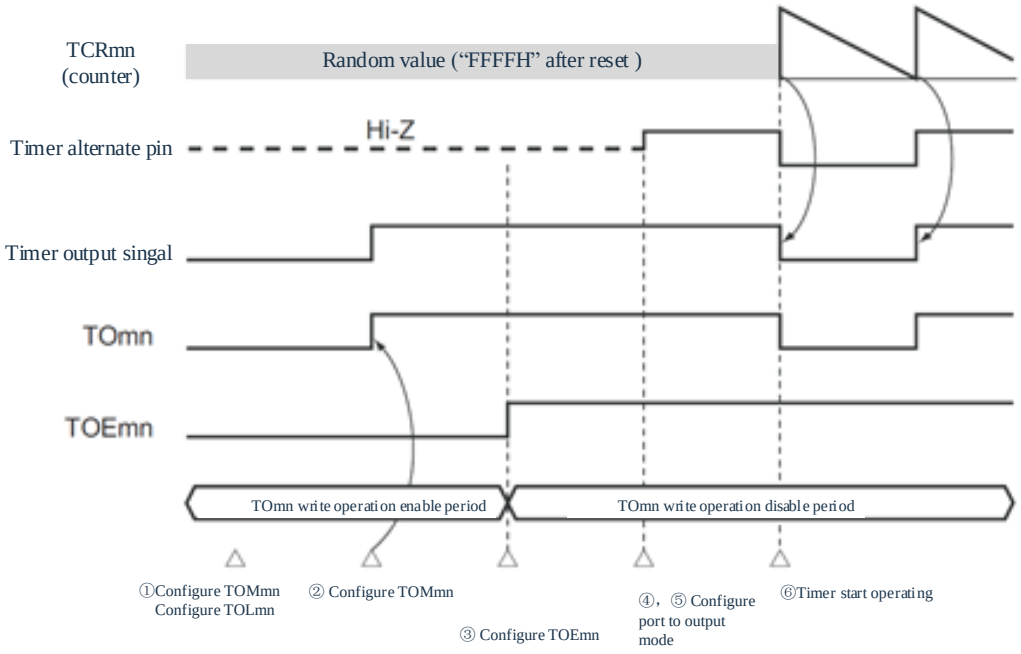
⑤ The TOm register can be read at any time and the output level of the TOmn pin can be confirmed.

Note: m: unit number (m= 0) n: channel number n=0~3 (master channel: n=0, 2) p: slave channel number (n=0: p=1, 2, 3 n=2: p=3)

## 5.5.2 TOmn Pin Output Setting

The following figure shows the procedure and status transition of the TOmn output pin from initial setting to timer operation start.

Figure 5-10 State change from setting timer output to start of operation




①Set the operation mode of the timer output.

TOMmn bit (0: master channel output mode, 1: slave channel output mode)

TOLmn bit (0: positive logic output, 1: negative logic output)

②The timer output signal is set to the initial state by setting the timer output register m (TOm).

③Writing “1” to TOEmn bit enables timer output (writing to TOm register is disabled).

④The port is set to digital input/output via the port mode control register (PMCxx)

⑤Set the input/output of the port to output

⑥Enable timer operation (TSmn=1).

Note: m: unit number (m=0) n: channel number (n=0~3)

## 5.5.3 Cautions on Channel Output Operation

(1) Change of setting values for TOm, TOEm, TOLm, TOMm registers in timer operation

The operation of the timer (timer count register mn (TCRmn) and timer data register mn (TDRmn)) and the Tomn output circuit are independent. Therefore, changes in the setting values of timer output register m (TOm), timer output enable register m (TOEm), and timer output level register m (TOLm) do not affect the operation of the timer, and the setting values can be changed during timer operation. However, in order to output the expected waveform from the TOmn pin during the operation of each timer, the value must be set to the example of the register setting contents for each operation shown in Sections 5.7 and 5.8.

If the setting values of TOEm register and TOLm register other than TOm register are changed before and after generating the timer interrupt (INTTMmn) signal for each channel, the waveform output from TOmn pin may be different depending on whether it is changed before or after generating the timer interrupt (INTTMmn) signal.

Note: m: unit number (m=0) n: channel number (n=0~3)

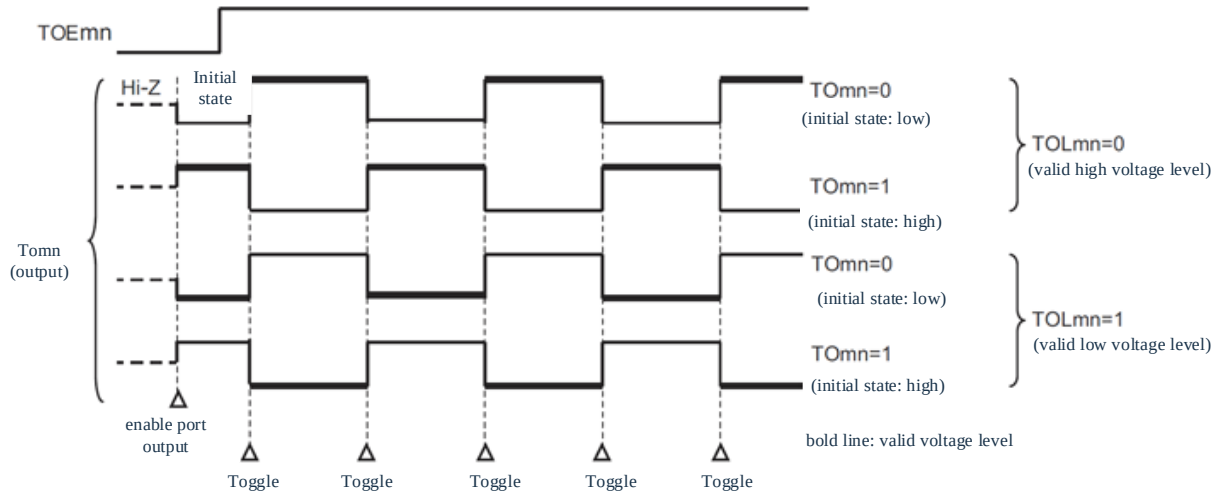
(2) Default level of TOmn pin and output level after timer operation start

The change in the output level of the TOmn pin when timer output register m (TOm) is written while timer output is disabled (TOEmn = 0), the initial level is changed, and then timer output is enabled (TOEmn = 1) before port output is enabled, is shown below.

(a) Operation starts in master channel output mode (TOMmn=0)

In the master channel output mode (TOMmn=0), the setting of the timer output level register m (TOLm) is invalid. If the timer operation is started after the initial level is set, the output level of the TOmn pin is inverted by generating a toggle signal.

Figure 5-11 Output state of TOmn pin at toggle output (TOMmn=0)



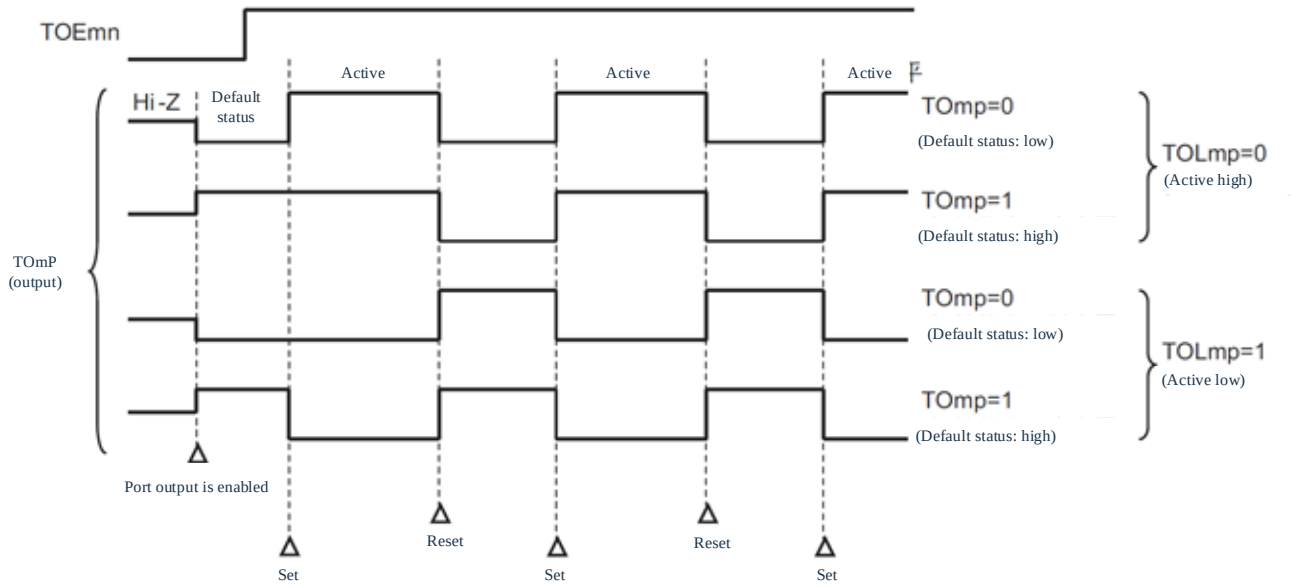

Note 1: Toggle: Reverse TOmn pin output status.

Note 2: m: unit number (m=0) n: channel number (n=0~3)

(b) When operation starts with slave channel output mode (TOMmn = 1) setting (PWM output))

In slave channel output mode (TOMmn=1), the active level depends on the setting of timer output level register m (TOLmn).

Figure 5-12 Output state of TOmn pin at PWM output (TOMmn=1)




Note 1: Set: The output signal from the TOmp pin changes from an invalid level to a valid level.

Note 2: Reset: The output signal from the TOmp pin changes from a valid level to an invalid level.

Note 3: m: unit number (m=0) n: channel number (p=1~3)

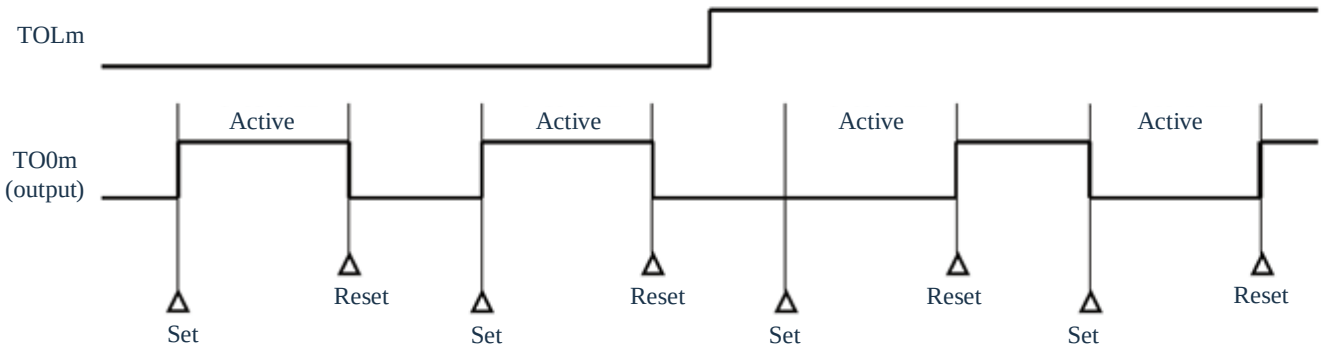
(3) Operation of TOmn pin in slave channel output mode (TOMmn = 1)

(a) When timer output level register m (TOLm) setting has been changed during timer operation

When the TOLm register setting has been changed during timer operation, the setting becomes valid at the generation timing of the TOmn pin change condition. Rewriting the TOLm register does not change the output level of the TOmn pin.

The operation when TOMmn is set to 1 and the value of the TOLm register is changed while the timer is operating (TEmn = 1) is shown below.

Figure 5-13 Operation when the contents of the TOLm register are changed during timer operation




Note 1: The output signal from the TOmp pin changes from an invalid level to a valid level.

Note 2: Reset: The output signal from the TOmp pin changes from a valid level to an invalid level.

Note 3: m: unit number (m=0) n: channel number (n=1~3)

(b) Set/reset timing

(c) In order to achieve 0% and 100% output at PWM output, the set timing of the TOmn pin/TOmn bit when generating the master channel timer interrupt (INTTMmn) is delayed by 1 count clock via the slave channel.

When the set condition and reset condition are generated at the same time, the reset condition is given priority.

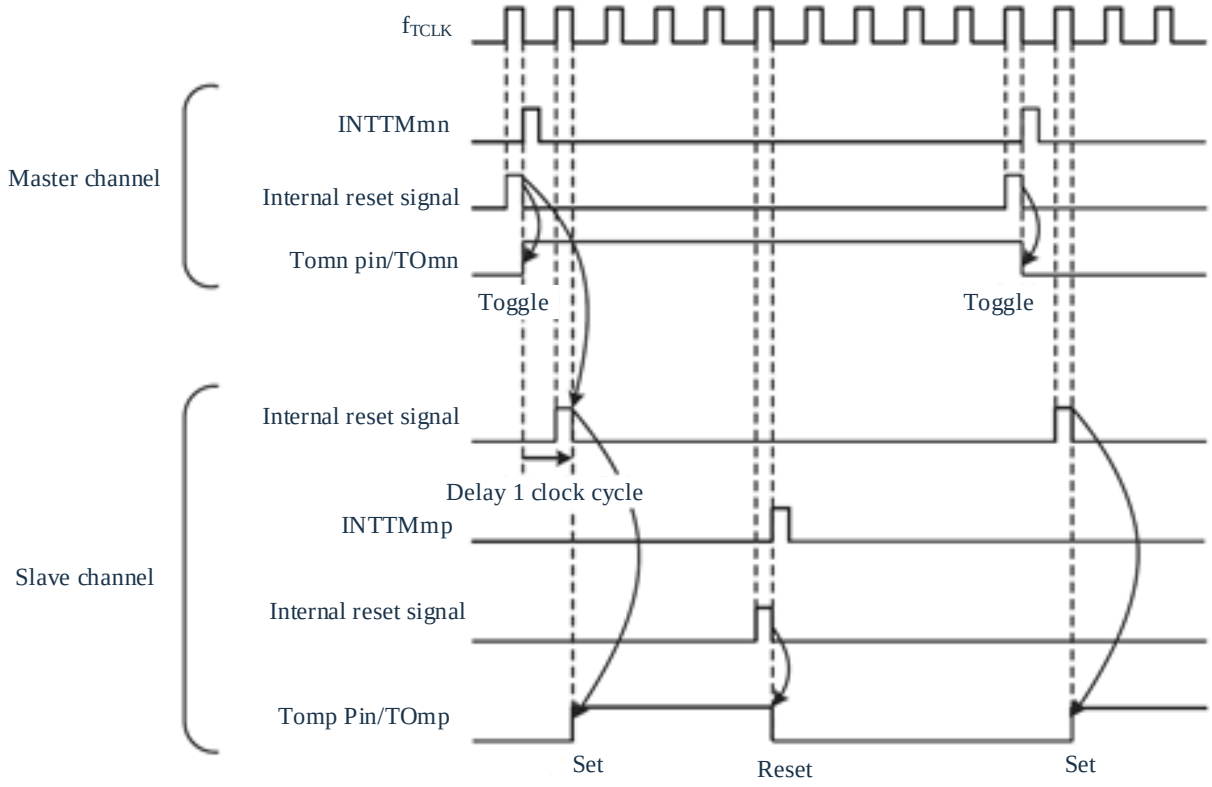
The set/reset operation status when setting the master/slave channel according to the following method is shown in Figure 5-14.

Master channel: TOEmn=1, TOMmn=0, TOLmn=0

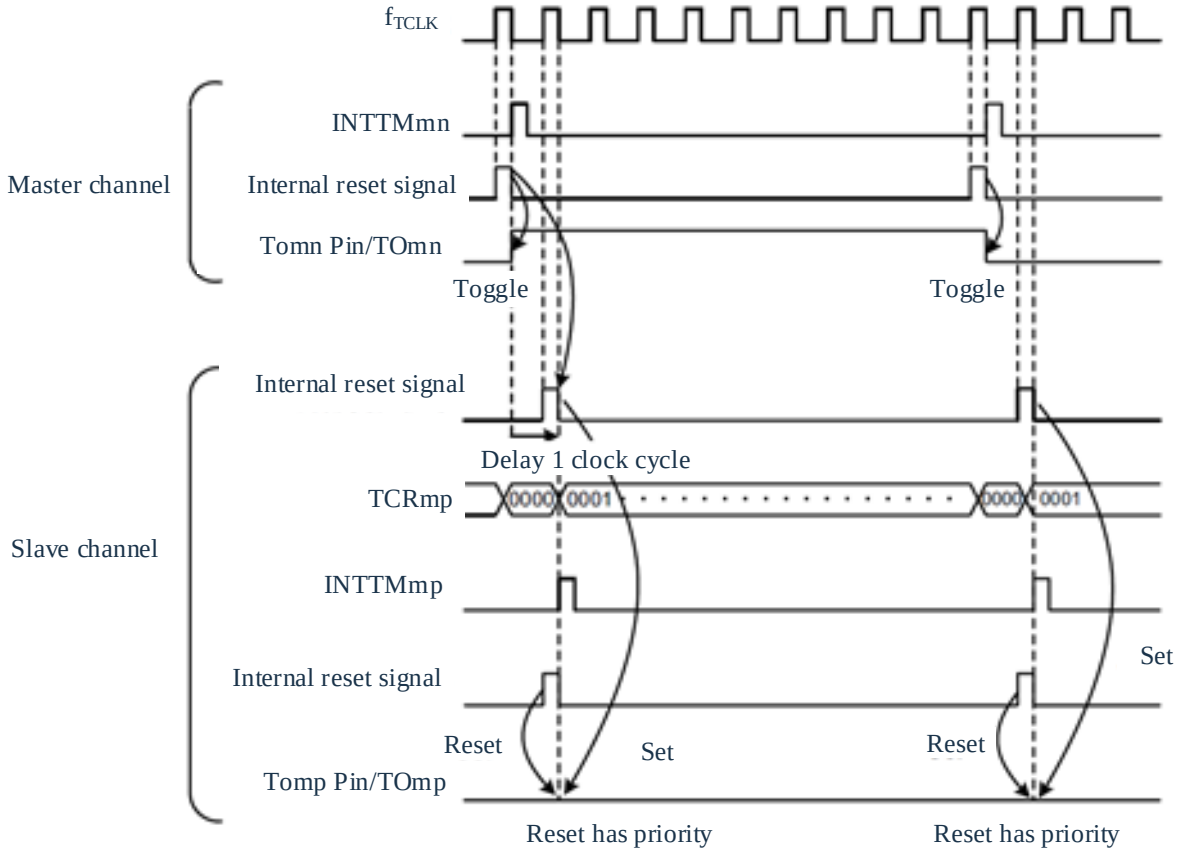
Slave channel: TOEmp=1, TOMmp=1, TOLmp=0

Figure 5-14 Set/reset timing operation status

(1) Basic operation timing




(2) Operation timing when 0% duty cycle




Note 1: Internal reset signal: TOmn pin reset/toggle signal

Note 2: Internal set signal: TOmn pin set signal

Note 3: m: unit number (m=0) n: channel number n=0~3 (master channel: n=0, 2)

p: slave channel number n=0: p=1, 2, 3 n=2: p=3

## 5.5.4 One-Shot Operation of TOmn Bit

Like the timer channel start register m (TSm), the timer output register m (TOm) has the set bits (TOmn) for all channels and can therefore operate the TOmn bits for all channels at once.

Table 5-26 One-shot operation example of TO0n bit

Before writing

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TO0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TO03<br>1 | TO02<br>0 | TO01<br>1 | TO00<br>0 |
| TOE0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TOE03<br>0 | TOE02<br>0 | TOE01<br>0 | TOE00<br>1 |

Data to be written

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

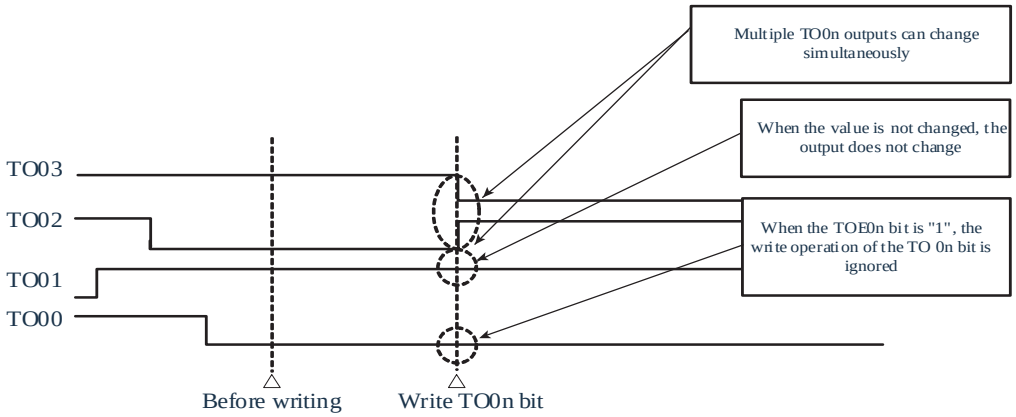
After writing

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TO0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TO03<br>0 | TO02<br>1 | TO01<br>1 | TO00<br>0 |

Only TOmn bits with TOEmn bit “0” can be written, and write is ignored when the TOmn bit is “1”.

TOmn (channel output) to which TOEmn = 1 is set is not affected by the write operation. Even if the write operation is done to the TOmn bit, it is ignored and the output change by timer operation is normally done.

Figure 5-15 TO0n pin state when the TO0n bit is operated at one time




Note: m: unit number (m=0) n: channel number (n=0 to 3)

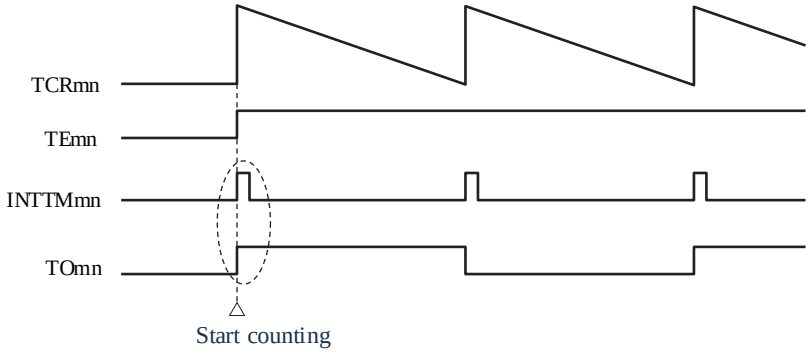
## 5.5.5 Timer Interrupt and TOmn Pin Output When Counting Starts

In interval timer mode or capture mode, the MDmn0 bit of timer mode register mn (TMRmn) is the bit that sets whether to generate a timer interrupt when counting starts.

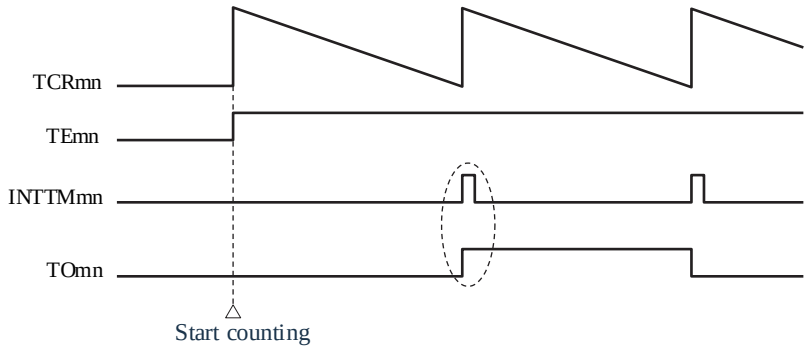
When the MDmn0 bit is “1”, the start timing of the count can be known by generating a timer interrupt (INTTMmn). In other modes, the timer interrupt and TOmn output at the start of counting are not controlled. An example of operation when set to interval timer mode (TOEmn=1, TOMmn=0) is shown below.

Figure 5-16 An operation example of timer interrupt and TOmn output at start count

(a) When MDmn0 = “1”




(b)When MDmn0 = “0”




When MDmn0 bit is “1”, the timer interrupt (INTTMmn) is output at the start of counting and TOmn is output alternately.

When MDmn0 bit is “0”, no timer interrupt (INTTMmn) is output at the start of counting and TOmn is not changed, while INTTMmn is output and TOmn is alternately output after 1 cycle of counting.

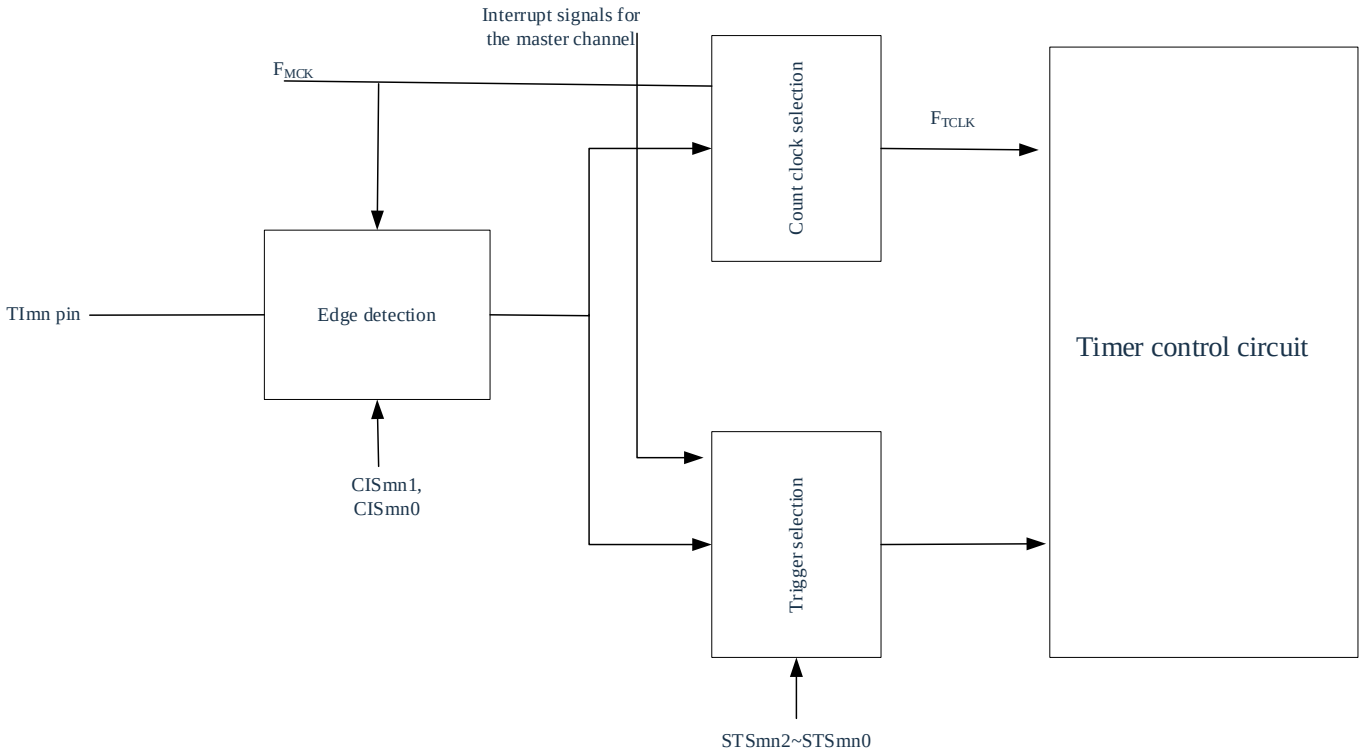
Note: m: unit number (m=0) n: channel number (n=0 to 3)

# 5.6 Control of Timer Input (TImn)

## 5.6.1 Structure of TImn Pin Input Circuit

The signals from the timer input pin are input to the timer control circuit through an edge detection circuit. The structure of the input circuit is as follows.

Figure 5-17 Structure of input circuit




## 5.6.2 Cautions on Channel Input Operation

When the timer input pin is set for use, the corresponding channel for the timer input pin is allowed to trigger. If all bits 12 (CCSmn), 9 (STSmn1), and 8 (STSmn0) in the timer mode register mn (TMRmn) are “0” and any of these bits is set, the trigger for enabling the timer channel to start (TSm) must be set at least 2 operation clock ($f_{MCK}$) cycles after the trigger.

# 5.7 Independent Channel Operation Function of Universal Timer Unit

## 5.7.1 Operation as Interval Timer/Square Wave Output

(1) Interval timer

It can be used as a reference timer to generate INTTMmn (timer interrupt) at fixed intervals. The interrupt generation period can be calculated using the following equation:

| INTTMmn (timer interrupt) generation period= count clock period × (TDRmn set value+1) |
|---|

(2) Operation as square wave output

The TOmn alternates outputs while generating the INTTMmn, outputting a square wave with a 50% duty cycle. The period and frequency of the square wave can be calculated using the following equation:

| Period of square wave output from TOmn = Period of count clock × (TDRmn set value+1)*2 |
|---|

| Frequency of square wave output from TOmn= Frequency of count clock/ {(TDRmn set value+1)*2} |
|---|

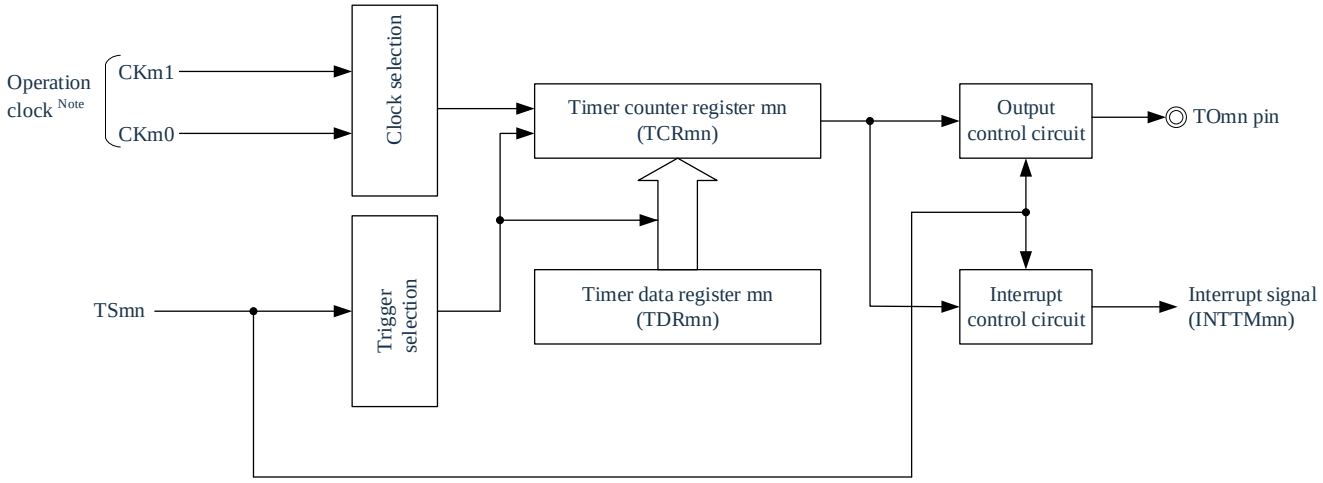
In the interval timer mode, the timer count register mn (TCRmn) is used as a decrement counter.

After setting the channel start trigger bit (TSmn, TSHm1, TSHm3) of the timer channel start register m (TSm) to "1", the value of timer data register mn (TDRmn) is loaded into the TCRmn register by the first count clock. At this time, if the MDmn0 bit of the timer mode register n (TMRmn) is "0", INTTMmn is not output and TOmn is not alternately output. If the MDmn0 bit of TMRmn register is "1", INTTMmn is output and TOmn is alternately output. Then, the TCRmn register is decremented by the count clock.

If the TCRmn becomes "0000H", the INTTMmn and TOmn are output alternately by the next count clock. At the same time, the value of TDRmn register is loaded into TCRmn register again. After that, continue the same operation.

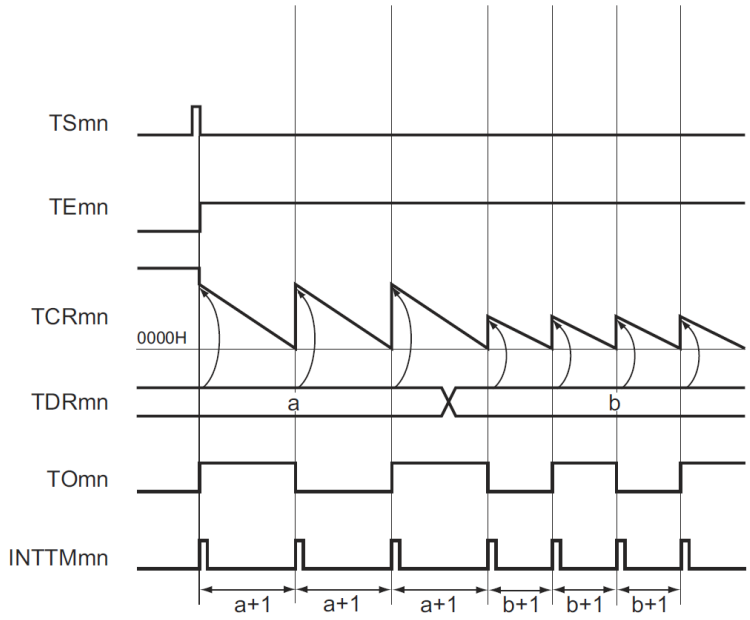
The TDRmn register can be rewritten at any time, and the rewritten TDRmn register value is valid from the next cycle.

Figure 5-18 Example of basic timing operating as an interval timer/square wave output (MDmn0=1)




Note: At channel 1 and channel 3, it is possible to select the clock from CKm0, CKm1, CKm2 and CKm3.

Figure 5-19 Example of basic timing operating as an interval timer/square wave output (MDmn0=1)




Note 1: In channel 1 and channel 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

Note 2: TSmn: Bit n of timer channel start register m (TSm)

TEmn: Bit n of timer channel enable status register m (TEm)

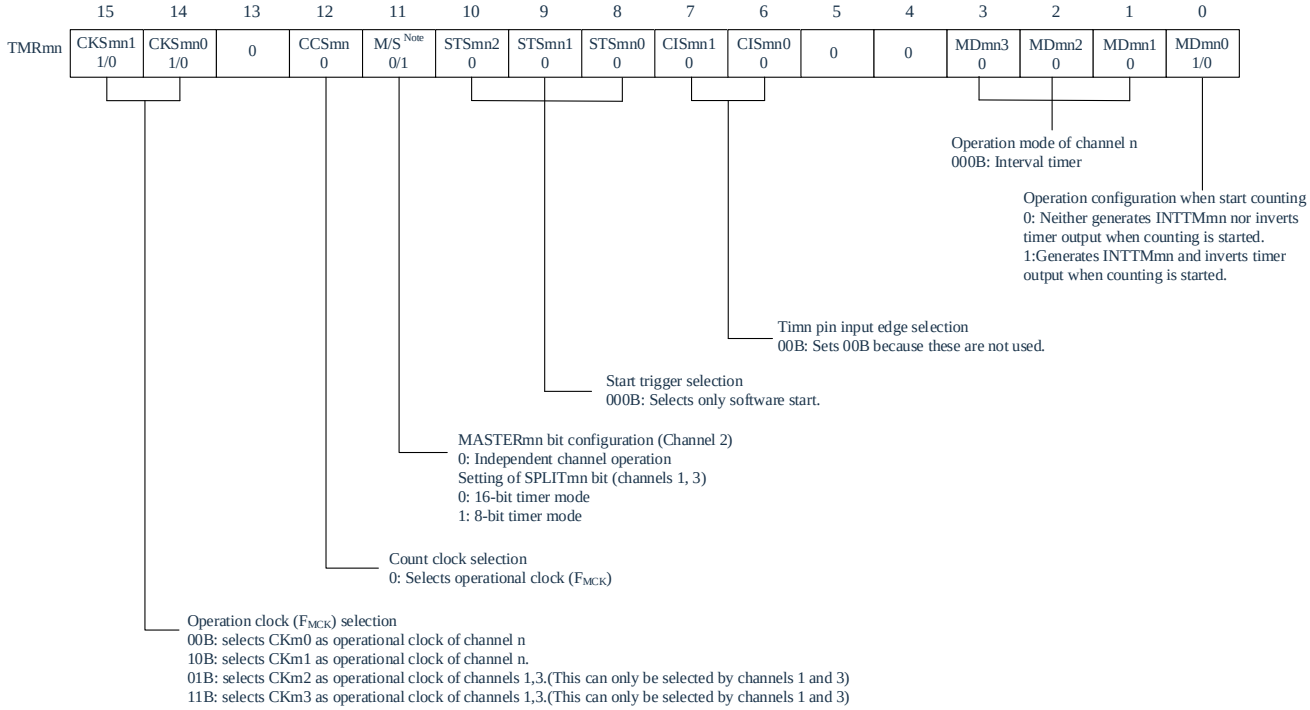
TCRmn: Timer count register mn (TCRmn)

TDRmn: Timer data register mn (TDRmn)

TOmn: TOmn pin output signal

Figure 5-20 Example of register setting contents for interval timer/square wave output

(a) Timer mode register mn (TMRmn)




(b) Timer output enable register m (TOEm)

| | bitn | |
|---|---|---|
| TOm | TOmn<br>1/0 | 0: Outputs 0 from TOmn.<br>1: Outputs 1 from TOmn. |

(c) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOEm | TOEmn<br>1/0 | 0: Stops the TOmn output operation by counting operation.<br>1: Enables the TOmn output operation by counting operation. |

(d) Timer output level register m (TOLm)

| | bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode) |

(e) Timer output mode register m (TOMm)

| | bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

Note 1: TMRm2: MASTERmn bit

Note 2: TMRm1, TMRm3: SPLITmn bit

Note 3: TMRm0: Fixed to “0”.

Note 4: m: unit number (m=0) n: channel number (n=0~3)

Table 5-27 Procedure for interval timer/square wave output function

| | Software operation | Hardware status |
|---|---|---|
| TIMER4 initial settings | | The input clock of timer unit m is in the stop-providing state.<br>(Stop providing clock, cannot write to each register) |
| | Set the TM4mEN bit of peripheral enable register 0(PER0) to "1". → | The input clock of timer unit m is in the providing state.<br>(Start providing clock, can write to each register) |
| | Set the timer clock selection register m (TPSm).<br>Determine the clock frequency of CKm0 ~ CKm3. | |
| Initial setting of channels | Set the timer mode register mn (TMRmn) (to determine the channel's operation mode).<br>Set the interval (period) value for the timer data register mn (TDRmn). | The channel is in the stop state.<br>(Provides clock, and consumes some power) |
| | Using TOmn output:<br>Set the TOMmn bit of Timer Output Mode Register m (TOMm) to "0" (master channel output mode).<br>Set the TOLmn bit to "0".<br>Set the TOmn bit to determine the initial level of the TOmn output. →<br>Set the TOEmn bit to "1" and enable TOmn output. →<br>Set the Port Register and Port Mode Register to "0". → | The TOmn pin is in Hi-Z output state.<br><br><br>When the port mode register is in output mode and the port register is "0", the TOmn initial set level is output.<br>The TOmn remains unchanged because the channel is in the stop state.<br>The TOmn pin outputs the level set by the TOmn. |
| Start operate | (The TOEmn bit set to "1" only when the TOmn output is used and restarted) Set the TSmn bit to "1".<br>Since the TSmn bit is a trigger bit, it automatically returns to "0". | The TEmn bit becomes "1" and starts counting. Load the value of the TDRmn register into the Timer Count Register mn (TCRmn).<br>When the MDmn0 bit of TMRmn register is "1", INTTMmn is generated and TOmn is output alternately. |
| In operation | The setting of the TDRmn register can be changed at will.<br>The TCRmn register can be read at any time.<br>The TSRmn register is not used.<br>The TOm register and TOEm register settings can be changed.<br>The setting of the TMRmn register, the TOMmn bit and the TOLmn bit cannot be changed. | The counter (TCRmn) performs decremental counting. If the count reaches "0000H", the value of the TDRmn register is loaded into the TCRmn register again and the count continues. When TCRmn is detected as "0000H", INTTMmn is generated and TOmn is alternately output. Thereafter, repeat this operation. |
| Stop operation | Set the TTmn bit to "1". →<br>The operation automatically returns to "0" because the TTmn bit is a trigger bit. | The TEmn bit becomes "0" and stops counting.<br>The TCRmn register holds the count value and stops counting.<br>The TOmn output is not initialized but remains its state. |
| | Set the TOEmn bit to "0" and set the value for the TOmn bit. → | The TOmn pin outputs the level set by the TOmn bit. |
| TIMER4 stop | To maintain the output level of the TOmn pin:<br>Set TOmn bit to "0" after setting the value to be held for the port register.<br>No need to maintain the output level of the TOmn pin:<br>No need to set. → | The output level of the TOmn pin is maintained by the port function. |
| | Set the TM4mEN bit of the PER0 register to "0".。 → | The input clock of timer unit m is in the stop-providing state.<br>Initialize all circuits and the SFR for each channel.<br>(TOmn bit becomes "0" and TOmn pin becomes port function) |

Restart operation →

Note: m: unit number (m=0) n: channel number (n=0~3)

## 5.7.2 Operation as External Event Counter

It can be used as an event counter to count the active edges (external events) detected on the TImn pin input and generate an interrupt if the specified count value is reached. The specified count value can be calculated using the following equation:

| Specified count value= TDRmn set value + 1 |
|---|

In the event counter mode, the timer count register mn (TCRmn) is used as a decrement counter.

The value of timer data register mn (TDRmn) is loaded into the TCRmn register by setting any channel start trigger bit (TSmn, TSHm1, TSHm3) of timer channel start register m (TSm) to “1”.

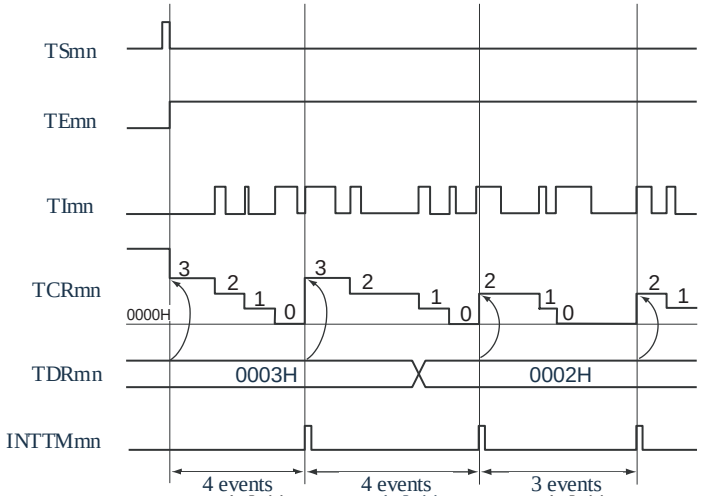
The TCRmn register decrements the count while detecting the active edge of the TImn pin input. If TCRmn becomes “0000H”, the value of TDRmn register is loaded again and INTTMmn is output.

After that, continue the same operation.

The output must be stopped by setting the TOEmn bit of the timer output enable register m (TOEm) to “0” because the TOmn pin outputs irregular waveforms based on external events.

The TDRmn register can be rewritten at any time, and the rewritten TDRmn register value is valid for the next cycle.

Figure 5-21 Example of basic timing operating as external event counter




Note: TSmn: Bit n of timer channel start register m (TSm)

TEmn: Bit n of timer channel enable status register m (TEm)

TImn: TImn pin input signal

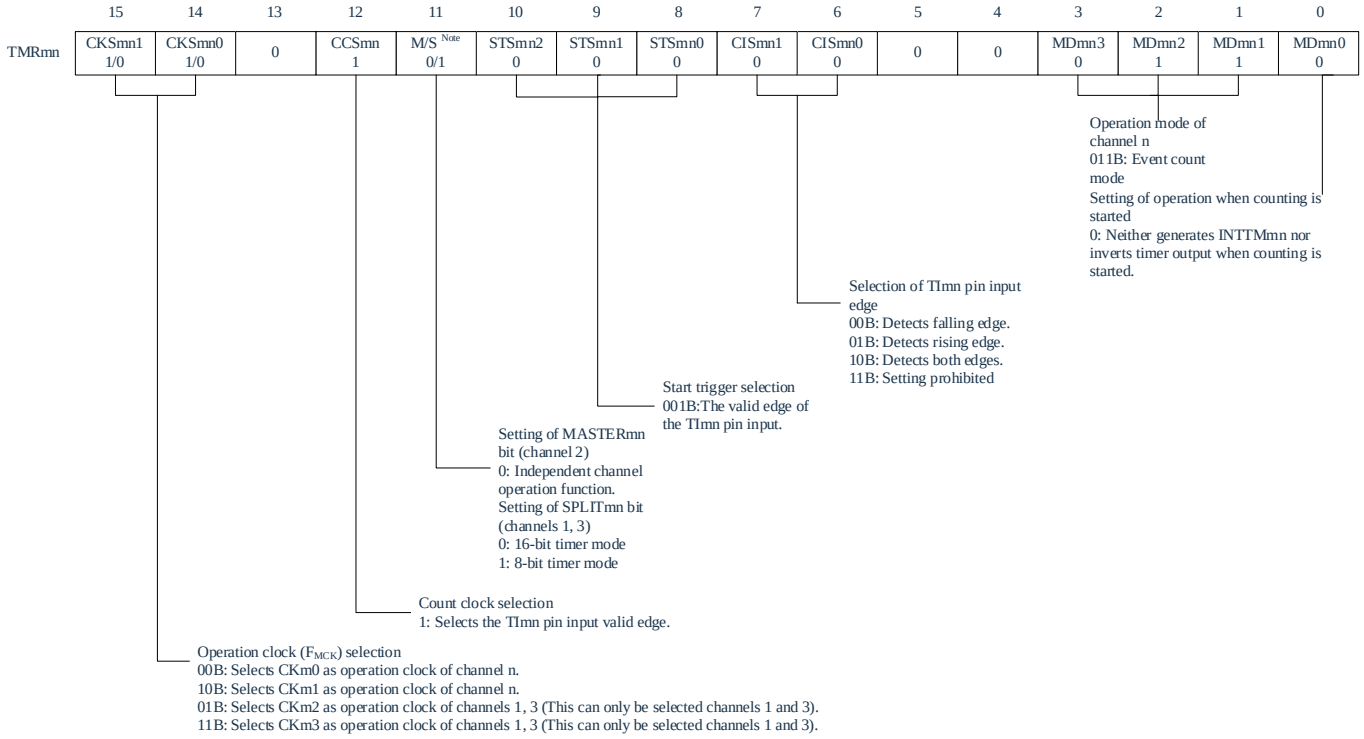
TCRmn: Timer count register mn (TCRmn)

TDRmn: Timer data register mn (TDRmn)

m: unit number (m=0) n: channel number (n=0~3)

Figure 5-22 Example of register contents setting in external event counter mode

(a) Timer mode register mn (TMRmn)




(b) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOm | TOmn<br>1/0 | 0: Outputs 0 from TOmn. |

(c) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOEm | TOEmn<br>1/0 | 0: Stops the TOmn output operation by counting operation.<br>1: Enables the TOmn output operation by counting operation. |

(d) Timer output level register m (TOLm)

| | bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode) |

(e) Timer output mode register m (TOMm)

| | bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

Note 1: TMRm2: MASTERmn bit

Note 2: TMRm1, TMRm3: SPLITmn bit

Note 3: TMRm0: Fixed to “0”.

Note 4: m: unit number (m= 0) n: channel number (n=0~3)

Table 5-28 Procedure for external event counter function

| | Software operation | Hardware status |
|---|---|---|
| Timer4 initial settings | | The input clock of timer unit m is in the stop-providing state.<br>(Stop providing clock, cannot write to each register) |
| | Set the TM4mEN bit of peripheral enable register 0(PER0) to "1". | The input clock of timer unit m is in the providing state and the channels are in the stop state.<br>(Start providing clock, can write to each register) |
| | Set the timer clock selection register m (TPSm).<br>Determine the clock frequency of CKm0 ~ CKm3. | |
| Initial setting of channels | Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON).<br>Set the timer mode register mn (TMRmn)(to determines the operating mode of the channel).<br>Set the count value for the timer data register mn (TDRmn).<br>Set the TOEmn bit of the timer output enable register m (TOEm) to "0". | The channel is in the stop state.<br>(Provides clock, and consumes some power) |
| Start operating | Set the TSmn bit to "1".<br>Since the TSmn bit is a trigger bit, it automatically returns to "0". | The TEmn bit becomes "1" and starts counting.<br>The value of the TDRmn register is loaded into the timer count register mn (TCRmn) and enter the detection wait state of the input edge of the TImn pin. |
| In operation | The setting of the TDRmn register can be changed at will.<br>The TCRmn register can be read at any time.<br>The TSRmn register is not used.<br>The setting of the TMRmn register, the TOMmn bit, the TOLmn bit, the Tomn bit and the TOEmn bit cannot be changed. | Whenever the input edge of the TImn pin is detected, the counter (TCRmn) is decremented. If the count reaches "0000H", the value of the TDRmn register is loaded into the TCRmn register again and the count continues. When TCRmn is detected as "0000H", INTTMmn is generated.<br>Thereafter, repeat this operation. |
| Stop operating | Set the TTmn bit to "1".<br>The operation automatically returns to "0" because the TTmn bit is a trigger bit. | The TEmn bit becomes "0" and stops counting.<br>The TCRmn register holds the count value and stops counting. |
| Timer4 stop | Set the TA4mEN bit of the PER0 register to "0". | The input clock of timer unit m is in the stop-providing state.<br>Initialize all circuits and the SFR for each channel. |

Restart operation

Note: m: unit number (m= 0) n: channel number (n=0~3)

## 5.7.3 Operation as Frequency Divider

The clock input from the TI00 pin can be divided and used as a divider for the output of the TO00 pin.

The divided clock frequency of the TO00 output can be calculated using the following equation:

The divided clock frequency of the TO00 output can be calculated using the following equation:

- Select rising or falling edge:

  Divider clock frequency = input clock frequency/ {(TDR00 set value+1)*2}

- Select both edges:

  Divider clock frequency≈input clock frequency/ (TDR00 set value+1)

In the interval timer mode, the timer count register 00 (TCR00) is used as a decrement counter.

After setting the channel start trigger bit (TS00) of timer channel start register 0 (TS0) to "1", the value of timer data register 00 (TDR00) is loaded into the TCR00 register by detecting an active edge of TI00. At this time, if the MD000 bit of Timer Mode Register 00 (TMR00) is "0", INTTM00 is not output and TO00 is not output alternately; if the MD000 bit of TMR00 register is "1", INTTM00 is output and TO00 is not output alternately. If the MD000 bit of TMR00 register is "1", INTTM00 is output and TO00 is output alternately.

The TCR00 register then counts down through the active edge of the TI00 pin input. If TCR00 changes to "0000H", TO00 performs an alternate output. At the same time, the value of the TDR00 register is loaded into the TCR00 register and counting continues.

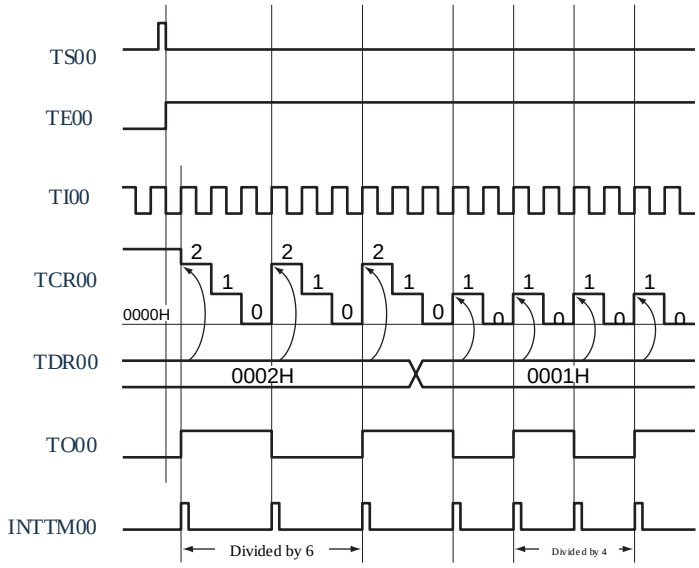
If double edge detection is selected for the TI00 pin input, the duty cycle error of the input clock affects the clock period of the TO00 output's division.

The clock period of the TO00 output contains the sampling error of 1 run clock cycle.

Clock period of the TOmn output= supposed TOmn output clock period± operating clock period (error)

The TDRmn register can be rewritten at any time, and the rewritten TDRmn register value is valid for the next cycle.

Figure 5-23 Example of basic timing operating as a frequency divider (MD000=1)




Note: TS00: Bit 0 of timer channel start register 0 (TS0)

TE00: Bit 0 of timer channel enable status register (TE0)

TI00: TI00 pin input signal

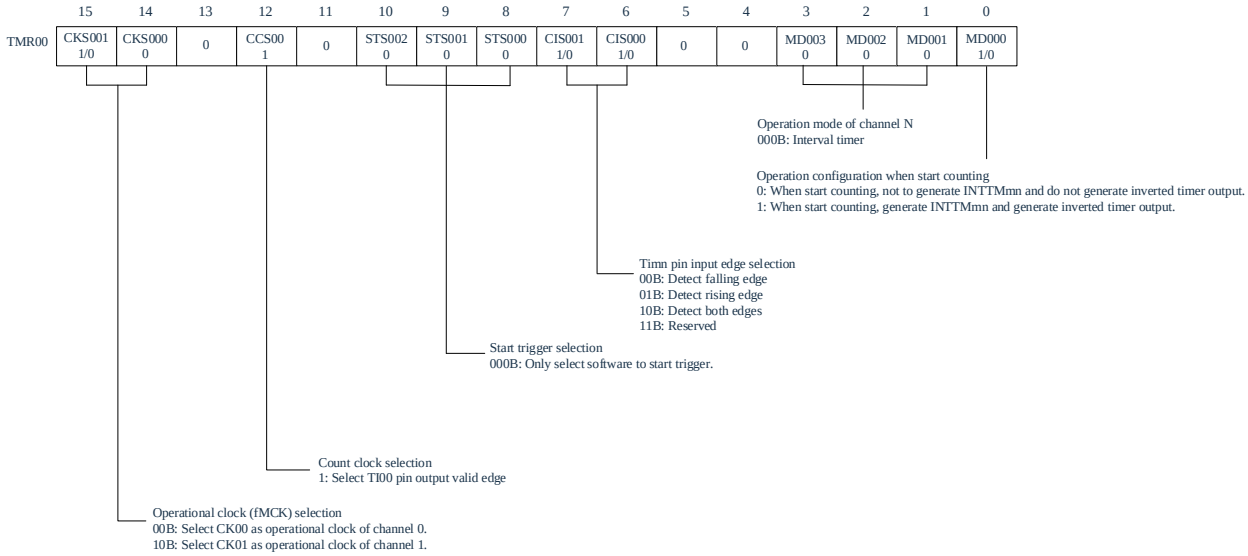
TCR00: Timer count register 00 (TCR00)

TDR00: Timer data register 00 (TDR00)

TO00: TO00 pin output signal

Figure 5-24 Example of register contents setting when operating as a frequency divider

(a) Timer mode register 00 (TMR00)




(b) Timer output register 0 (TO0)

bit 0

| TO0 | TO00<br>1/0 | 0: Outputs “0” by TO00.<br>1: Outputs “1” by TO00. |
|---|---|---|

(c) Timer output enable register 0 (TOE0)

bit 0

| TOE0 | TOE00<br>1/0 | 0: Stops the TO00 output performed by the counting operation.<br>1: Enables TO00 output performed by the counting operation. |
|---|---|---|

(d) Timer output level register 0 (TOL0)

bit 0

| TOL0 | TOL00<br>0 | 0: Sets “0” in the master channel output mode (TOM00=0). |
|---|---|---|

(e) Timer output mode register 0 (TOM0)

bit n

| TOM0 | TOM00<br>0 | 0: Sets master channel output mode. |
|---|---|---|

Table 5-29 Procedure for frequency divider function

| | Software operation | Hardware status |
|---|---|---|
| Timer4 initial settings | | The input clock of timer unit 0 is in the stop-providing state.<br>(Stop providing clock, cannot write to each register) |
| | Set the TM4mEN bit of peripheral enable register 0(PER0) to "1". | The input clock of timer unit 0 is in the providing state and the channels are in the stop state.<br>(Start providing clock, can write to each register) |
| | Set the timer clock selection register 0 (TPS0).<br>Determine the clock frequency of CK00 ~ CK03. | |
| Initial setting of channels | Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON).<br>Set the timer mode register 00 (TMR00)(to determines the operating mode of the channel, select edge detection).<br>Set the interval (cycles) value for the timer data register 00 (TDR00). | The channel is in the stop state.<br>(Provides clock, and consumes some power) |
| | Set TOM00 bit of timer output mode register 0 (TOM0) to "0" (master control channel output mode).<br>The TOL00 bit must be set to "0".<br>Set the TO00 bit and determine the initial level of TO00 output<br>Set TOE00 bit to "1" and enable TO00 output.<br>Set the Port Register and Port Mode Register to "0" | The TO00 pin is in Hi-Z output state.<br>When the port mode register is in output mode and the port register is "0", the TO00 initial set level is output.<br>Since the channel is in stop state, TO00 does not change.<br>The TO00 pin outputs the level set by the TO00. |
| Start operation | Set TOE00 bit to "1" (only limited to restart operation). The TS00 bit must be set to "1".<br>The operation automatically returns to "0" because the TS00 bit is a trigger bit. | The TE00 bit becomes "1" and starts counting.<br>Load the value of the TDR00 register into the Timer Count Register 00 (TCR00). When the MD000 bit TMR00 register is "1", INTTM00 is generated and TO00 is output alternately. |
| | The setting of the TDR00 register can be changed at will. The TCR00 register can be read at any time.<br>The TSR00 register is not used.<br>The TO0 register and TOE0 register settings can be changed.<br>The setting of the TMR00 register, the TOM00 bit and the TOL00 bit cannot be changed. | The counter (TCR00) performs decremental counting. If the count reaches "0000H", the value of the TDR00 register is loaded into the TCR00 register again and the count continues. When the TCR00 bit is "0000H", INTTM00 is generated and TO00 is output alternately. Thereafter, repeat this operation. |
| Stop operating | The TT00 bit must be set to "1".<br>The operation automatically returns to "0" because the TT00 bit is a trigger bit. | The TE00 bit becomes "0" and starts counting.<br>The TCR00 register holds the count value and stops counting.<br>The TO00 output is not initialized but remains its state. |
| | Set the TOE00 bit to "0" and set the value for the TO00 bit. | The TO00 pin outputs the level set by the TO00. |
| Timer4 stop | To maintain the output level of the TO00 pin:<br>Set TO00 bit to "0" after setting the value to be held for the port register.<br>No need to maintain the output level of the TO00 pin: No need to set. | The output level of the TO00 pin is maintained by the port function. |
| | Set the TM4mEN bit of the PER0 register to "0". | The input clock of timer unit 0 is in the stop-providing state.<br>Initialize all circuits and the SFR for each channel. (TO00 bit becomes "0" and TO00 pin becomes port function) |

Restart operation

Note: m: unit number (m= 0) n: channel number (n=0~3)

## 5.7.4 Operation as Input Pulse Interval Measurement

The count value can be captured at the active edge of TImn and the interval between TImn input pulses can be measured. The software operation (TSmn=1) can also be set to capture the count value during the period when the TEmn bit is “1”.

The pulse interval can be calculated using the following equation:

| TImn input pulse interval = period of counting clock× ((10000H × TSRmn: OVF) + (TDRmn captured value +1)) |
|---|

Note: The 1 operation clock error is generated because the TImn pin input is sampled by the operation clock selected by the CKSmn bit of the Timer Mode Register mn (TMRmn).

In capture mode, the timer count register mn (TCRmn) is used as an increment counter.

If the channel start trigger bit (TSmn) of the timer channel start register m (TSm) is set to “1”, the TCRmn register is incrementally counted from “0000H” by the count clock.

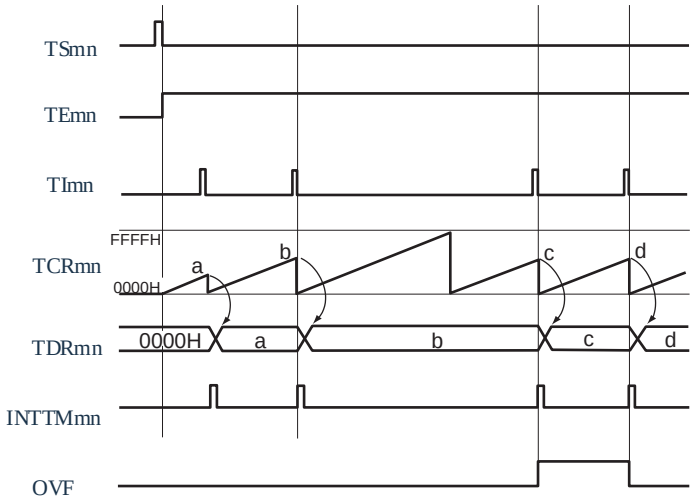
If the active edge of the TImn pin input is detected, the count value of TCRmn register is transferred (captured) to Timer Data Register mn (TDRmn), and the TCRmn register is cleared to "0000H", and then INTTMmn is output. If the counter overflows, the OVF bit of Timer Status Register mn (TSRmn) is set to "1". If the counter does not overflow, the OVF bit is cleared. After that, continue the same operation.

While capturing the count value to the TDRmn register, the OVF bit of the TSRmn register is updated according to whether or not overflow occurs during the measurement, and the overflow status of the captured value can be confirmed.

Even if the counter counts 2 or more complete cycles, the overflow is considered to have occurred and the OVF bit of the TSRmn register is set to “1”. However, when two or more overflows occur, the interval value cannot be measured normally by the OVF bit.

Set the STSmn2~STSmn0 bit of the TMRmn register to "001B", and use the valid edge of TImn for start trigger and capture trigger.

Figure 5-25 Example of basic timing operating as an input pulse interval measurement (MDmn0=0)




Note 1: m: unit number (m= 0) n: channel number (n=0~3)

Note 2: TSmn: Bit n of timer channel start register m (TSm)

TEmn: Bit n of timer channel enable status register m (TEm)

TImn: TImn pin input signal

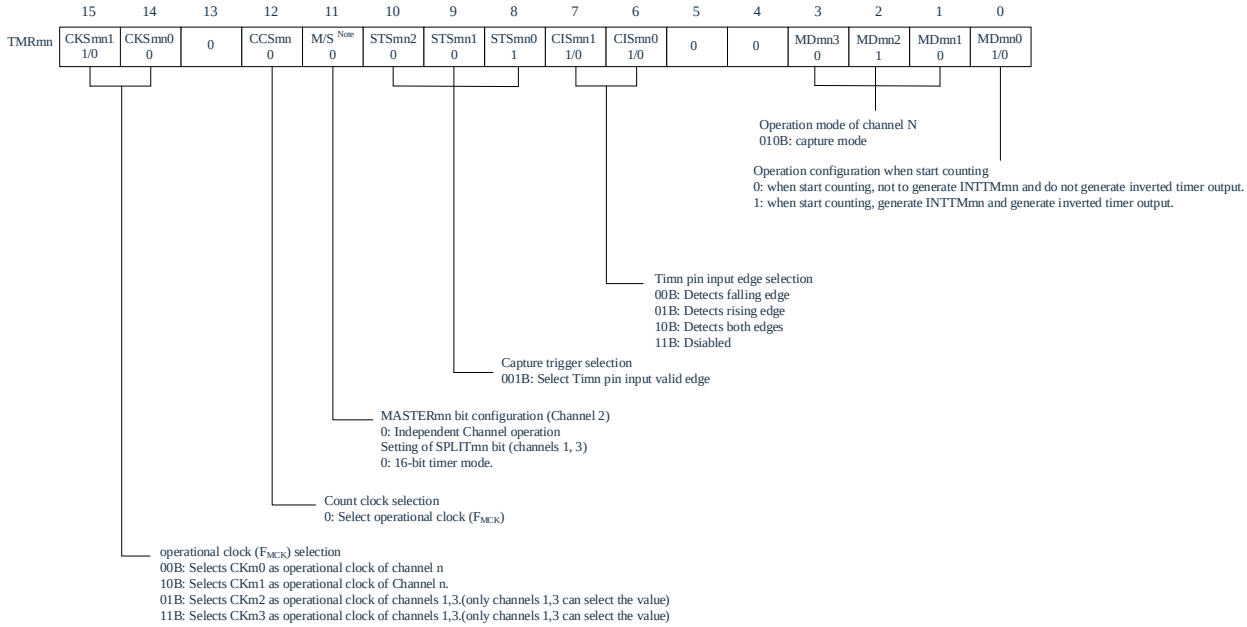
TCRmn: Timer count register mn (TCRmn)

TDRmn: Timer data register mn (TDRmn)

OVF: Bit0 of timer status register mn (TSRmn)

Figure 5-26 Example of register contents setting in measuring input pulse interval

(a) Timer mode register mn (TMRmn)




(b) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

(c) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops the TOmn output operation by counting operation. |

(d) Timer output level register m (TOLm)

| | bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Sets “0” in master channel output mode (TOMmn=0). |

(e) Timer output mode register m (TOMm)

| | bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

Note 1: TMRm2: MASTERmn bit

TMRm1, TMRm3: SPLITmn bit

TMRm0: Fixed to “0”.

Note 2: m: unit number (m=0) n: channel number (n=0~3)

Table 5-30 Procedure for input pulse interval measurement function

| | | Software operation | Hardware status |
|---|---|---|---|
| | Timer4 initial settings | | The input clock of timer unit 0 is in the stop-providing state.<br>(Stop providing clock, cannot write to each register) |
| | | Set the TM40EN bit of the peripheral enable register 0(PER0) to "1". → | The input clock of timer unit 0 is in the providing state and the channels are in the stop state.<br>(Start providing clock, can write to each register) |
| | | Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3. | |
| | Initial setting of channels | Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON).<br>Set the timer mode register mn (TMRmn)(to determines the operating mode of the channel). | The channel is in the stop state.<br>(Provides clock, and consumes some power) |
| → | Start operation | Set the TSmn bit to "1". →<br>Since the TSmn bit is a trigger bit, it automatically returns to "0". | The TEmn bit becomes "1" and starts counting.<br>Clear the timer count register mn (TCRmn) to "0000H".<br>When the MDmn0 bit of TMRmn register is "1", INTTMmn is generated. |
| Restart operation | In operation | The setting values of the CISmn1 bit and the CISmn0 bit of the TMRmn register can be changed.<br>The TDRmn register can be read at any time.<br>The TCRmn register can be read at any time.<br>The TSRmn register can be read at any time.<br>The setting of the the TOMmn bit, the TOLmn bit, the TOmn bit and the TOEmn bit cannot be changed. | The counter (TCRmn) starts incremental counting from "0000H" and transfers (captures) the count value to the timer data register mn (TDRmn) if it detects an active edge on the TImn pin input or sets TSmn bit to "1".<br>If the counter overflows, set the OVF bit of Timer Status Register mn (TSRmn). If the counter does not overflow, the OVF bit is cleared. Thereafter, repeat this operation. |
| | Stop operation | Set the TTmn bit to "1". →<br>The operation automatically returns to "0" because the TTmn bit is a trigger bit. | The TEmn bit becomes "0" and stops counting.<br>The TCRmn register holds the count value and stops counting.<br>The OVF bit of the TSRmn register remains unchanged. |
| | Timer4 stop | Set the TM40EN bit of the PER0 register to "0". → | The input clock of timer unit 0 is in the stop-providing state.<br>Initialize all circuits and the SFR for each channel. |

Note: m: unit number (m= 0) n: channel number (n=0~3)

## 5.7.5 Operation as Input Signal High-/Low-Level Width Measurement

The signal width (high-/low-level width) of TImn can be measured by starting counting at one edge of the input to the TImn pin and capturing the count value at the other edge. The TImn signal width of the TImn output can be calculated using the following equation:

| Signal width of TImn input= period of count clock × ((10000H * TSRmn: OVF) + (TDRmn captured value+1)) |
|---|

Note: Because the TImn pin inputs are sampled by the operation clock selected by the CKSmn bit of the Timer Mode Registermn (TMRmn), an error of 1 operation clock is generated.

In the Capture & Single Count mode, the timer count register mn (TCRmn) is used as an increment counter. If the channel start trigger bit (TSmn) of the timer channel start register m(TSm) is set to “1”, the TEmn bit becomes “1”, and the start edge detection wait state of the TImn pin is entered.

If the start edge of the TImn pin input (rising edge of the TImn pin input at the time of high-level width measurement) is detected, it is synchronized with the count clock and counts incrementally from “0000H”. Then, if an active capture edge is detected (falling edge of TImn pin input at the time of high-level width measurement), the count value is transferred to the Timer Data Register mn (TDRmn) and INTTMmn is output at the same time. If the counter overflows, the OVF bit of the Timer Status Register mn (TSRmn) is set to “1”. If the counter does not overflow, the OVF bit is cleared. The value of the TCRmn register changes to “Value passed to TDRmn register +1”, and the start edge detection wait state of the TImn pin is entered. After that, continue the same operation.

While capturing the count value to the TDRmn register, the OVF bit of the TSRmn register is updated according to whether or not overflow occurs during the measurement, and the overflow status of the captured value can be confirmed.

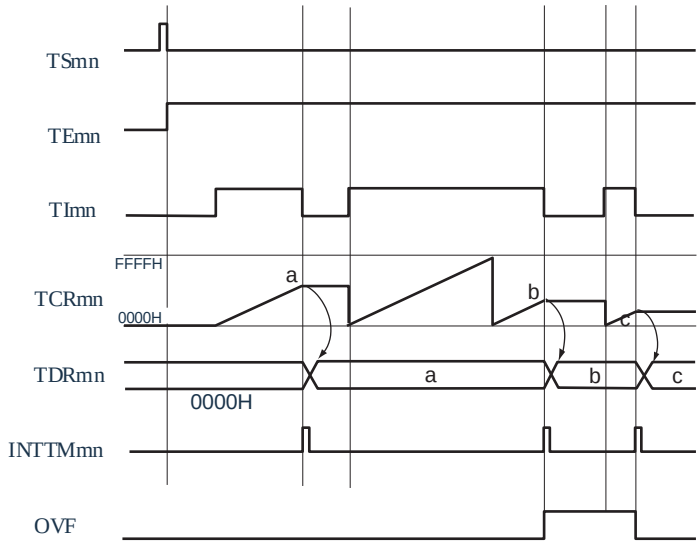
Even if the counter counts 2 or more complete cycles, the overflow is considered to have occurred and the OVF bit of the TSRmn register is set to “1”. However, when two or more overflows occur, the interval value cannot be measured normally by the OVF bit.

The CISmn1 and CISmn0 bits of the TMRmn register can be used to set whether the high-level width or low-level width of the TImn pin is to be measured. This function is designed to measure the input signal width of the TImn pin, so the TSmn bit cannot be set to “1” during the period when the TEmn bit is “1”.

CISmn1, CISmn0=10B of the TMRmn register: Measures the low-level width.

CISmn1, CISmn0=11B of the TMRmn register: Measures the high-level width.

Figure 5-27 Example of basic timing operating as high-/low-level width measurement of input signal




Note 1: m: unit number (m= 0) n: channel number (n=0~3)

Note 2: TSmn: Bit n of timer channel start register m (TSm)

TEmn: Bit n of timer channel enable status register m (TEm)

TImn: TImn pin input signal

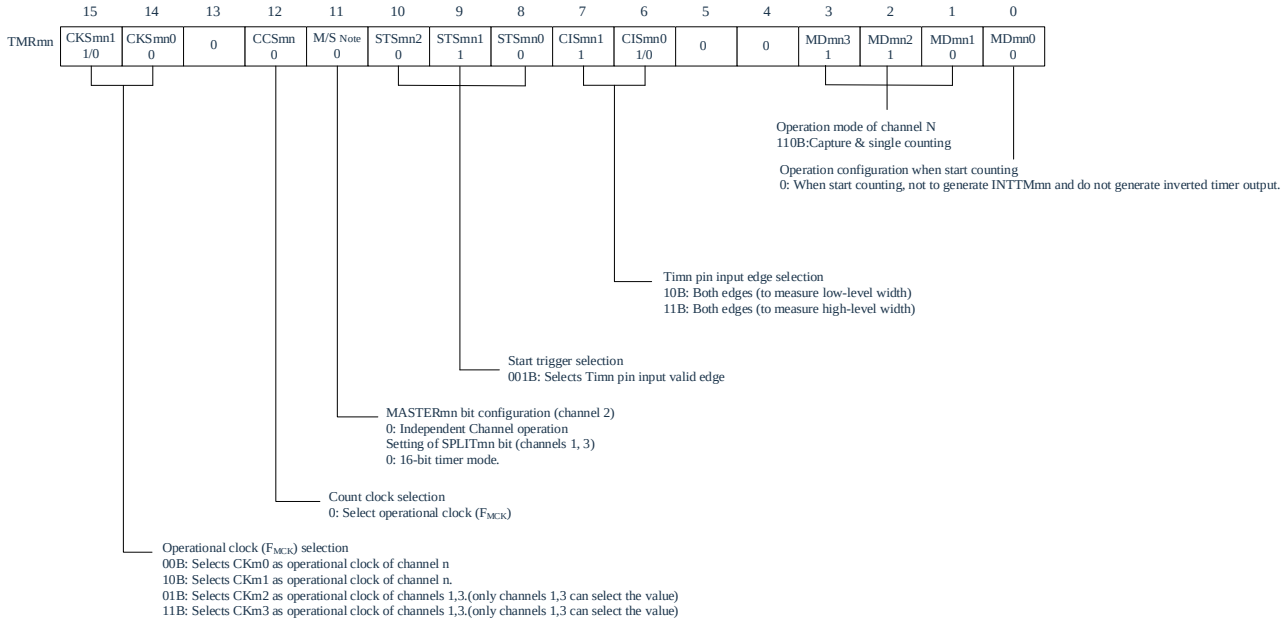
TCRmn: Timer count register mn (TCRmn)

TDRmn: Timer data register mn (TDRmn)

OVF: Bit 0 of timer status register mn (TSRmn)

Figure 5-28 Example of register contents setting in measuring high-/low-level width of input signal

(a) Timer mode register mn (TMRmn)




(b) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

(c) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops the TOmn output operation by counting operation. |

(d) Timer output level register m (TOLm)

| | bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Sets “0” in master channel output mode (TOMmn=0). |

(e) Timer output mode register m (TOMm)

| | bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

Note 1: TMRm2: MASTERmn bit

TMRm1, TMRm3: SPLITmn bit

TMRm0: Fixed to “0”.

Note 2: m: unit number (m= 0) n: channel number (n=0~3)

Table 5-31 Procedure for high-/low-level width measurement function of input signal

| | Software operation | Hardware status |
|---|---|---|
| | | |
| Timer4 initial settings | | The input clock of timer unit 0 is in the stop-providing state.<br>(Stop providing clock, cannot write to each register) |
| | Set the TM40EN bit of the peripheral enable register 0(PER0) to "1". → | The input clock of timer unit 0 is in the providing state and the channels are in the stop state.<br>(Start providing clock, can write to each register) |
| | Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3. | |
| Initial setting of channels | Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON).<br>Set the timer mode register mn (TMRmn)(to determines the operating mode of the channel). | The channel is in the stop state.<br>(Provides clock, and consumes some power) |
| Start Operation | Set the TSmn bit to "1".<br>Since the TSmn bit is a trigger bit, it automatically returns to "0". → | The TEmn bit changes to "1" and enters the detection wait state for start triggering (detecting the active edge of the TImn pin input or setting the TSmn bit to "1"). |
| | Detect TImn pin input counting start edge. → | Clear timer count register mn (TCRmn) to "0000H" and start incremental counting. |
| In operation | The setting of the TDRmn register can be changed at will.<br>The TCRmn register can be read at any time.<br>The TSRmn register is not used.<br>The setting of the TMRmn register, the TOMmn bit, the TOLmn bit, the Tomn bit and the TOEmn bit cannot be changed. | After the start edge of the TImn pin is detected, the counter (TCRmn) starts counting incrementally from "0000H". If the capture edge of the TImn pin is detected, the count value is transferred to the timer data register mn (TDRmn), and INTTMmn is generated.<br>If the counter overflows, set the OVF bit of Timer Status Register mn (TSRmn). If the counter does not overflow, the OVF bit is cleared. The TCRmn register stops counting before the start edge of the next TImn pin is detected.<br>Thereafter, repeat this operation. |
| Stop operation | Set the TTmn bit to "1".<br>The operation automatically returns to "0" because the TTmn bit is a trigger bit. → | The TEmn bit becomes "0" and stops counting.<br>The TCRmn register holds the count value and stops counting.<br>The OVF bit of the TSRmn register remains unchanged. |
| Timer4 stop | Set the TM40EN bit of the PER0 register to "0". → | The input clock of timer unit 0 is in the stop providing state.<br>Initialize all circuits and the SFR for each channel. |

Restart operation →

Note: m: unit number (m= 0) n: channel number (n=0~3)

## 5.7.6 Operation as Delay Counter

The count can be decremented by the active edge detection (external event) of the TImn pin input and INTTTMmn (timer interrupt) is generated at any set interval.

During the period when the TEmn bit is “1”, the TSmn bit can be set to “1” by software to start decreasing counting and generate INTTMmn (timer interrupt) at any set interval.

The interrupt generation period can be calculated using the following equation:

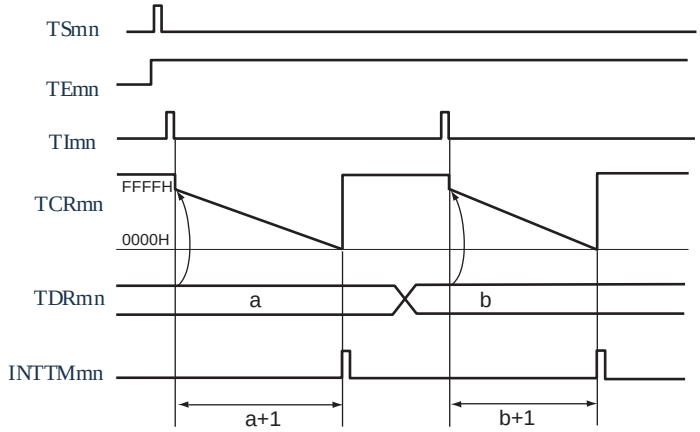
| INTTMmn (timer interrupt) generation period = counting clock period × (TDRmn set value+1) |
|---|

In the single count mode, the timer count register mn (TCRmn) is used as a decrement counter.

If the channel start trigger bit (TSmn, TSHm1, TSHm3) of the timer channel start register m(TSm) is set to “1”, the TEmn bit, TEHm1 bit, TEHm3 bit become “1”, and the active edge detection wait state of the TImn pin is entered. An active edge detection via the TImn pin input starts the TCRmn register and loads the value of the Timer Data Register mn (TDRmn). The TCRmn register counts decreasingly from the value of the loaded TDRmn register by counting the clock. If TCRmn becomes “0000H”, INTTMmn is output and counting is stopped until the next active edge of the TImn pin input is detected.

The TDRmn register can be rewritten at any time, and the rewritten TDRmn register value is valid from the next cycle.

Figure 5-29 Example of basic timing operating as a delay counter




Note 1: m: unit number (m= 0) n: channel number (n=0~3)

Note 2: TSmn: Bit n of timer channel start register m (TSm)

TEmn: Bit n of timer channel enable status register m (TEm)

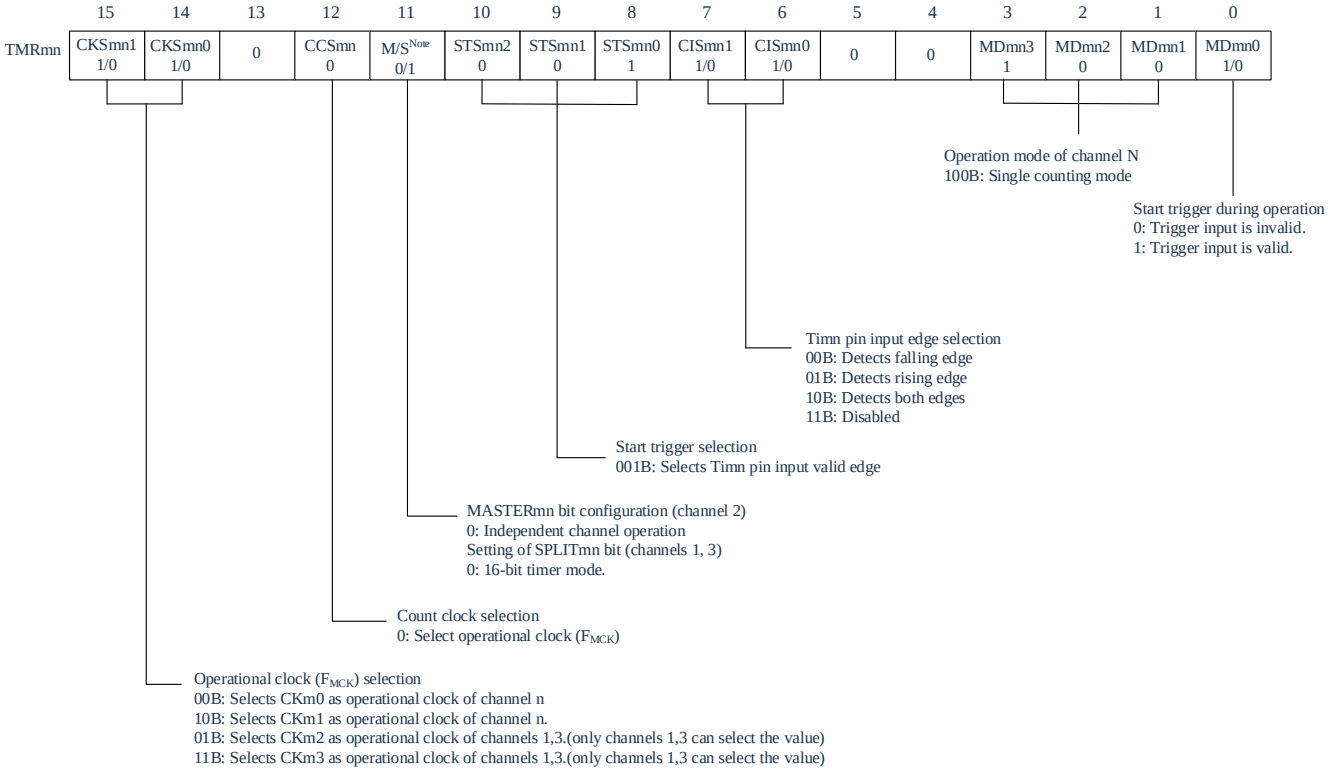
TImn: TImn pin input signal

TCRmn: Timer count register mn (TCRmn)

TDRmn: Timer data register mn (TDRmn)

Figure 5-30 Example of register contents setting for delay counter function

(a) Timer mode register mn (TMRmn)




(b) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

(c) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops the TOmn output operation by counting operation. |

(d) Timer output level register m (TOLm)

| | bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Sets “0” in master channel output mode (TOMmn=0). |

(e) Timer output mode register m (TOMm)

| | bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

Note 1: TMRm2: MASTERmn bit

TMRm1, TMRm3: SPLITmn bit

TMRm0: Fixed to “0”.

Note 2: m: unit number (m= 0) n: channel number (n=0~3)

Table 5-32 Procedure for delay counter function

| | Software operation | Hardware status |
|---|---|---|
| Timer4 initial settings | | The input clock of timer unit 0 is in the stop-providing state.<br>(Stop providing clock, cannot write to each register) |
| | Set the TM40EN bit of the peripheral enable register 0(PER0) to "1". → | The input clock of timer unit 0 is in the providing state and the channels are in the stop state.<br>(Start providing clock, can write to each register) |
| | Set the timer clock selection register m (TPSm).<br>Determine the clock frequency of CKm0 ~ CKm3. | |
| Initial setting of channels | Set the timer mode register 00 (TMRmn)(to determine the operating mode of the channel n).<br>Set the output delay time for the timer data register mn (TDRmn).<br>Set the TOEmn bit to "0" and stop TOmn operation. | The channel is in a running stop state.<br>(Provides clock, consumes some power) |
| Start Operation | Set the TSmn bit to "1". →<br>Since the TSmn bit is a trigger bit, it automatically returns to "0". | The TEmn bit turns into '1' and enter into start trigger (detect Timn pin input active edge or set TSmn bit to '1') detection waiting state. |
| | Start decreasing the count by detecting the next start trigger.<br>• The active edge of the TImn pin input.<br>• Set the TSmn bit to "1" by software. → | Load the value of the TDRmn register into the Timer Count Register mn (TCRmn). |
| In operation | The setting of the TDRmn register can be changed at will.<br>The TCRmn register can be read at any time.<br>The TSRmn register is not used. | The counter (TCRmn) performs decremental counting. If TCRmn counts to "0000H", INTTMmn is generated and TCRmn is "1" until the next start trigger is detected (detecting an active edge on the TImn pin input or setting TSmn to "1"). The count is stopped when "0000H" is detected. |
| Stop operation | Set the TTmn bit to "1". →<br>The operation automatically returns to "0" because the TTmn bit is a trigger bit. | The TEmn bit becomes "0" and stops counting.<br>The TCRmn register holds the count value and stops counting. |
| Timer4 stop | Set the TM40EN bit of the PER0 register to "0". → | The input clock of timer unit 0 is in the stop-providing state.<br>Initialize all circuits and the SFR for each channel. |

Restart operation →

Note: m: unit number (m= 0) n: channel number (n=0~3)

# 5.8 Multi-Channel Linkage Operation Function for Universal Timer Unit

## 5.8.1 Operation as Single Trigger Pulse Output Function

Using the 2 channels in pairs, a single trigger pulse with any delay pulse width can be generated from the input of the TImn pin. The delay and pulse width can be calculated using the following equation:

| |
|---|
| Delay = {TDRmn (master) set value+2} × counting clock period<br>Pulse width= {TDRmp (slave) set value} × counting clock period |

In single count mode, the master channel operates and counts the delay. By detecting a start trigger, the timer count register mn (TCRmn) of the master channel starts to operate and loads the value of timer data register mn (TDRmn). The TCRmn register counts decreasingly from the value of the loaded TDRmn register by counting the clock. If TCRmn becomes “0000H”, INTTMmn is output and counting stops before the next start trigger is detected.

In single count mode, the slave channel operates and counts the pulse width. The INTTMmn of the master channel is used as the start trigger and the TCRmp register of the slave channel is started and loaded with the value of the TDRmp register. The TCRmp register counts decreasingly from the value of the loaded TDRmp register by counting the clock. If the count value becomes “0000H”, INTTMmp is output and counting is stopped until the next start trigger (INTTMmn of the master channel) is detected. The output level of TOmp becomes valid after INTTMmn has been generated from the master channel and after 1 count clock, if TCRmp becomes “0000H”, it becomes invalid.

The software operation (TSmn=1) can also be used as a start trigger to output a single trigger pulse without using the TImn pin input.

Caution: Because the TDRmn register of the master channel and the TDRmp register of the slave channel have different loading timings, if the TDRmn register and the TDRmp register are rewritten during counting, they may compete with the loading timings and output an abnormal waveform. The TDRmn register must be rewritten after generating INTTMmn and the TDRmp register must be rewritten after generating INTTMmp.

Note: m: unit number (m= 0) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Figure 5-31 Block diagram of operation as single trigger pulse output function

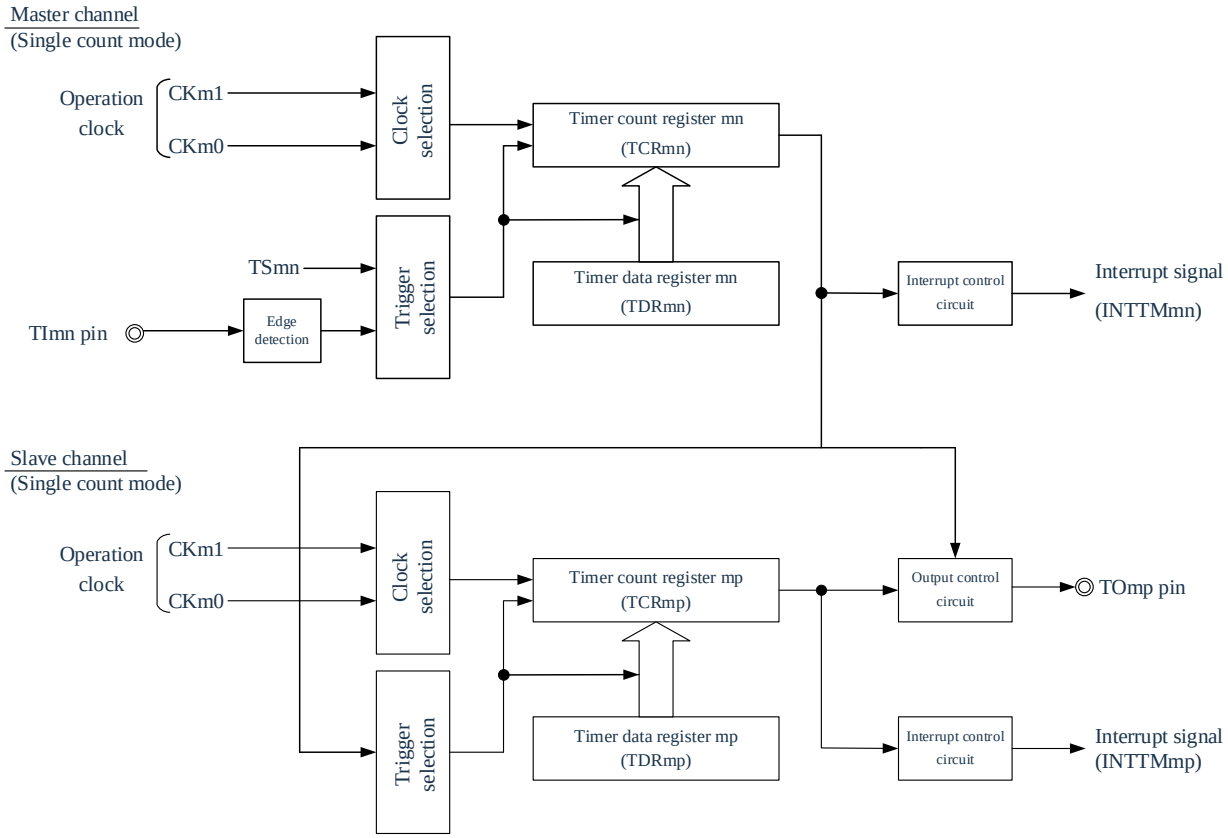
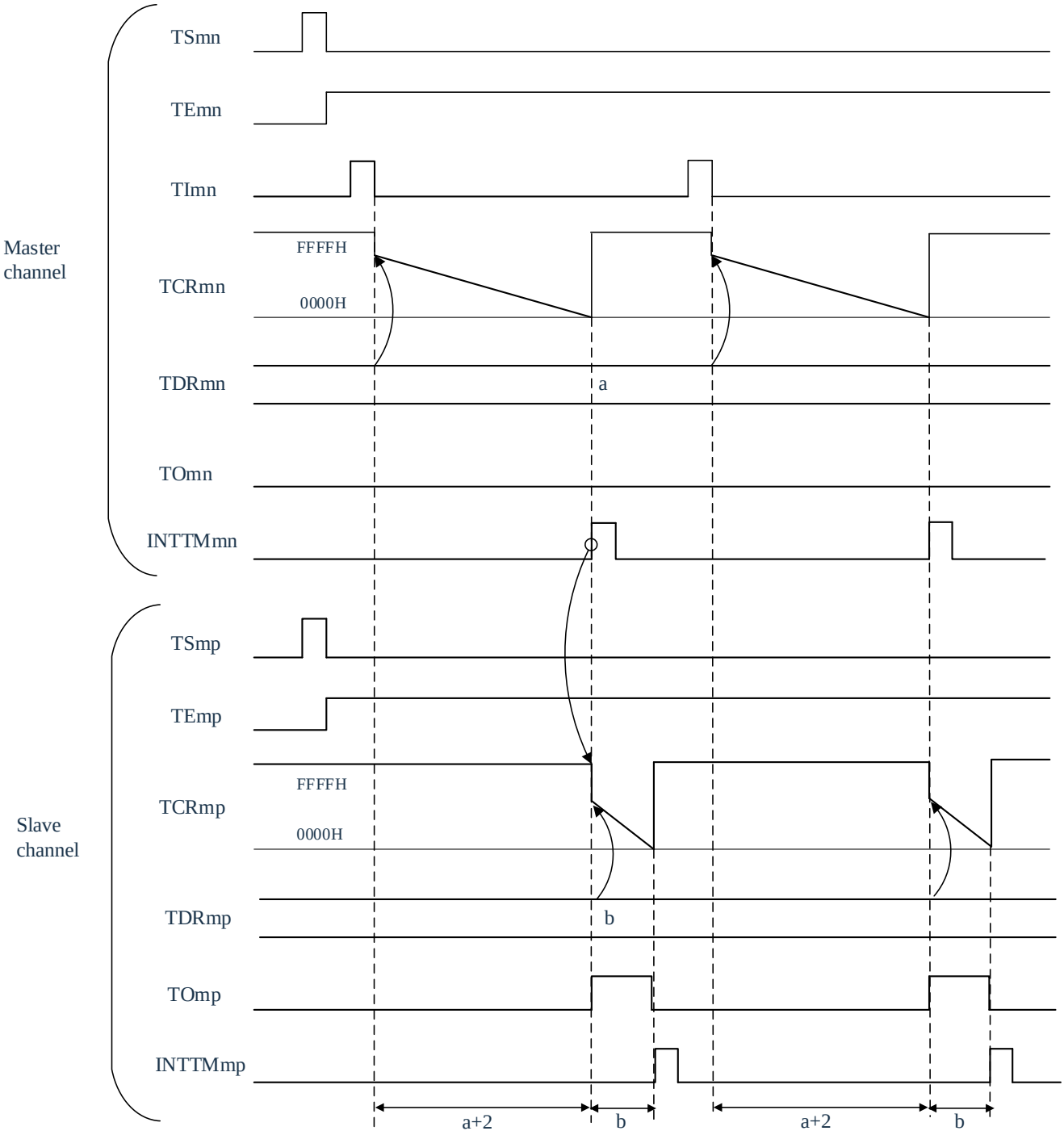


Figure 5-32 Example of basic timing operating as a single trigger pulse output function




Note 1: m: unit number (m= 0) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Note 2: TSmn, TSmp: Bits n and p of timer channel start register m (TSm)

TEmn, TEmp: Bits n and p of timer channel enable status register m (TEm)

TImn, TImp: Input signals of TImn pin and TImp pin

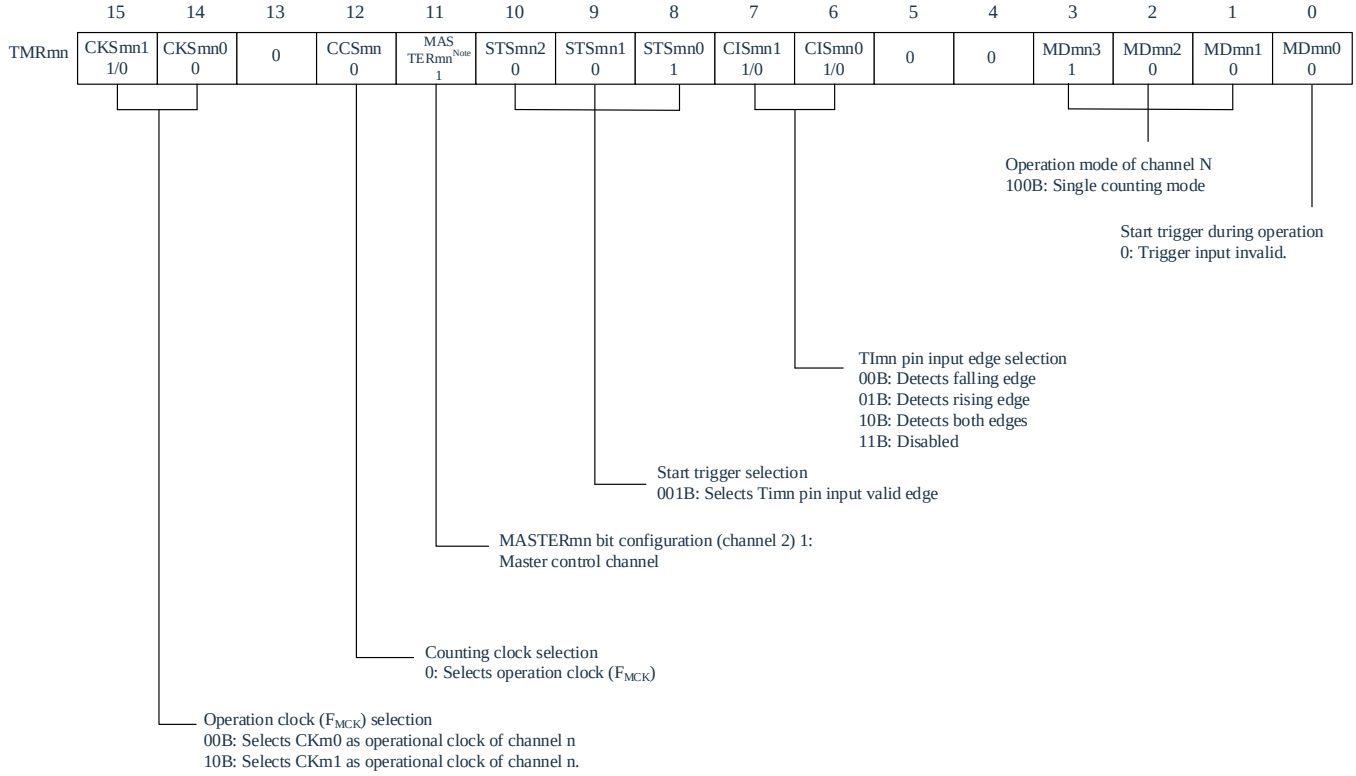
TCRmn, TCRmp: Timer count registers mn, mp (TCRmn, TCRmp)

TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)

TOmn, TOmp: Output signals of TOmn pin and TOmp pin

Figure 5-33 Example of register contents setting for single trigger pulse output function (master channel)

(a) Timer mode register mn (TMRmn)




(b) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

(c) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops the TOmn output operation by counting operation. |

(d) Timer output level register m (TOLm)

| | bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode) |

(e) Timer output mode register m (TOMm)

| | bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

Note 1: TMRm2: MASTERmn bit

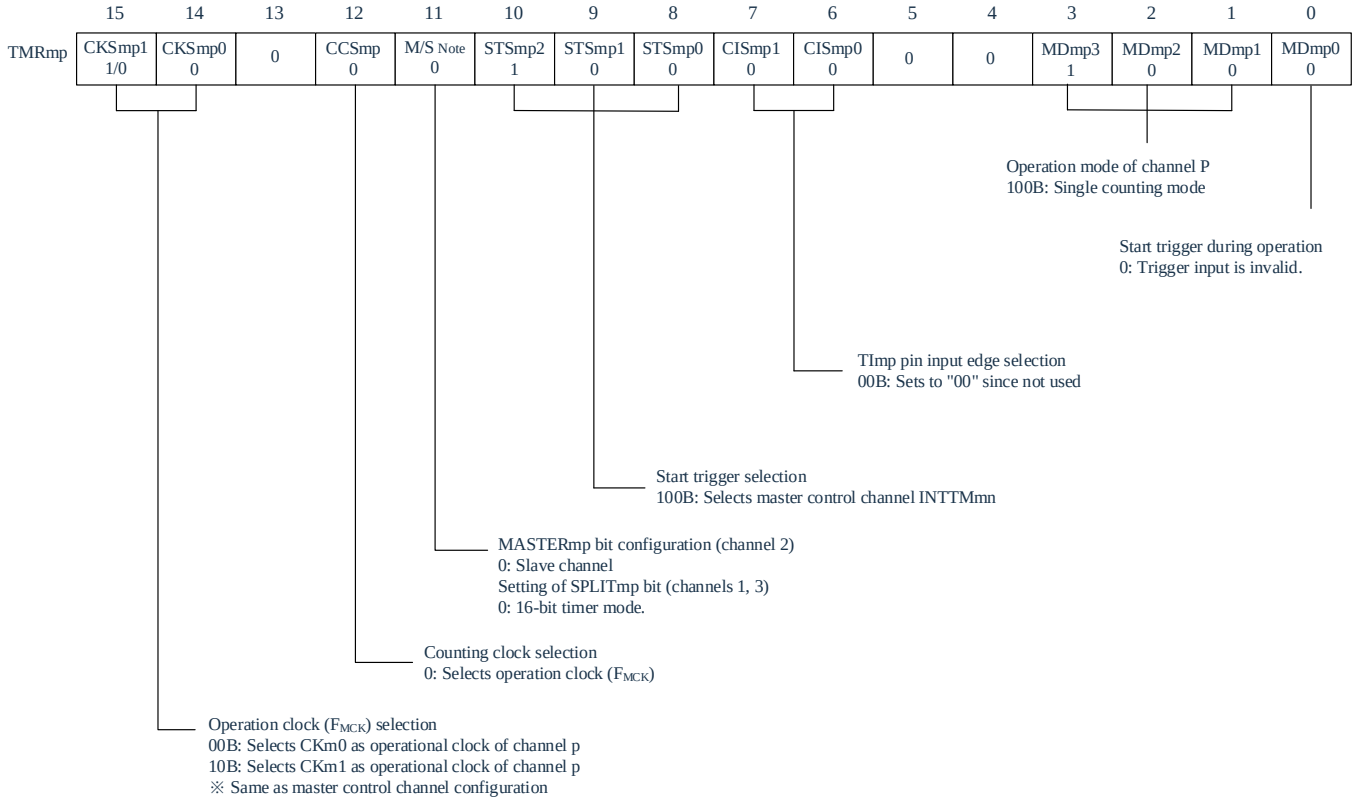
TMRm1, TMRm3: SPLITmn bit

TMRm0: Fixed to “0”.

Note 2: m: unit number (m= 0) n: channel number (n=0~2)

Figure 5-34 Example of register contents setting for single trigger pulse output function (slave channel)

(a) Timer mode register mp (TMRmp)




(b) Timer output enable register m (TOEm)

| | bit p | |
|---|---|---|
| TOm | TOmp 1/0 | 0: Outputs "0" by TOmp.<br>1: Outputs "1" by TOmp. |

(c) Timer output enable register m (TOEm)

| | bit p | |
|---|---|---|
| TOEm | TOEmp 1/0 | 0: Stops TOmp output performed by the counting operation.<br>1: Enables TOmp output performed by the counting operation. |

(d) Timer output level register m (TOLm)

| | bit p | |
|---|---|---|
| TOLm | TOLmp 1/0 | 0: Positive logic output (active high level)<br>1: Negative logic output (active low level) |

(e) Timer output mode register m (TOMm)

| | bit p | |
|---|---|---|
| TOMm | TOMmp 1 | 1: Sets slave channel output mode. |

Note 1: TMRm2: MASTERmp bit

TMRm1, TMRm3: SPLITmp bit

TMRm0: Fixed to "0".

Note 2: m: unit number (m= 0) n: channel number (n=0~2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Table 5-33 Procedure for single trigger pulse output function(1/2)

| | Software operation | Hardware status |
|---|---|---|
| Timer4 initial settings | | The input clock of timer unit 0 is in the stop-providing state.<br>(Stop providing clock, cannot write to each register) |
| | Set the TM40EN bit of the peripheral enable register 0(PER0) to "1". | The input clock of timer unit 0 is in the providing state and the channels are in the stop state.<br>(Start providing clock, can write to each register) |
| | Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3. | |
| Initial setting of channels | Set the timer mode registers mn and mp (TMRmn, TMRmp) for the 2 channels used (to determine the operation mode of the channel).<br>Set the output delay time for the timer data register mn (TDRmn) of the master channel, and set the pulse width for the TDRmp register of the slave channel. | The channel is in the stop state.<br>(Provides clock, and consumes some power) |
| | Slave channel setting<br>Set TOMmp bit of the timer output mode register m (TOMm) to "1" (slave channel output mode).<br>Set the TOLmp bit.<br>Set the TOmp bit to determine the initial level of the TOmp output.<br>Set the TOEmp bit to "1" and enable TOmp output.<br>Set the Port Register and Port Mode Register to "0". | The TOmp pin is in Hi-Z output state.<br>When the port mode register is in output mode and the port register is "0",<br>the TOmp initial set level is output.<br>The TOmp remains unchanged because the channel is in the stop state.<br>The TOmp pin outputs the level set by the TOmp. |

Table 5-33 Procedure for single trigger pulse output function(2/2)

| | | Software operation | Hardware status |
|---|---|---|---|
| Restart operation | Start operation | Set the TOEmp bit (slave) to "1" (restart operation only).<br>Set the TSmn (master) and TSmp (slave) bits of the Timer Channel Start Register m (TSm) to "1" at the same time. →<br>Since the TSmn bit and the TSmp bit are trigger bits, they automatically return to "0". | The TEmn and TEmp bits are set to 1 and the master channel enters the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit of the master channel is set to 1) wait status.<br>Counter stops operating. |
| | | Count operation of the master channel is started by start trigger detection of the master channel<br>Detects the TImn pin input valid edge<br>Sets the TSmn bit of the master channel to 1 by software. | Master channel starts counting. |
| | In operation | Set values of the CISmn1 and CISmn0 bits of the TMRmn register can be changed only .<br>Set values of the TMRmp, TDRmn, TDRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed.<br>The TCRmn and TCRmp registers can always be read.<br>The TSRmn and TSRmp registers are not used.<br>Set values of the TOm and TOEm registers by slave channel can be changed. | Master channel loads the value of the TDRmn register to timer count register mn (TCRmn) by the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit of the master channel is set to 1), and the counter starts counting down. When the count value reaches TCRmn = 0000H, the INTTMmn output is generated, and stops counting until the next TImn pin input.<br>The slave channel, triggered by INTTMmn of the master channel, loads the value of the TDRmp register to the TCRmp register, and the counter starts counting down.<br>The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped.<br>After that, the above operation is repeated. |
| | Stop operation | The TTmn (master) and TTmp (slave) bits are set to 1 at the same time. →<br>The TTmn and TTmp bits automatically return to 0 because they are trigger bits. | TEmn, TEmp = 0, and count operation stops.<br>The TCRmn and TCRmp registers hold count value and stop.<br>The TOmp output is not initialized but holds current status. |
| | | The TOEmp bit of slave channel is cleared to 0 and value is set to the TOmp bit. → | The TOmp pin outputs the TOmp set level. |
| | Timer4 stop | To hold the TOmp pin output level<br>Clears the TOmp bit to 0 after the value to be held is set to the port register. →<br>When holding the TOmp pin output level is not necessary:<br>Settings are not required. | The TOmp pin output level is held by port function. |
| | | The TM40EN bit of the PER0 register is cleared to 0. → | The input clock of timer unit 0 is in the stop-providing state.<br>Initialize all circuits and the SFR of each channel. |

Note 1: m: unit number (m= 0) n: master channel number (n=0)

p: slave channel number

q: slave channel number $n<p<q\leq 3$ (p and q are integers greater than n)

Note 2: The TSmn bit of the slave channel cannot be set to “1”.

## 5.8.2 Operation as PWM Function

By using the 2 channels in pairs, pulses of any period and duty cycle can be generated. The period and duty cycle of the output pulses can be calculated using the following equations:

| |
|---|
| Pulse period = {TDRmn (master) set value+1} * counting clock period<br>Duty cycle [%] = {TDRmp (slave) set value} / {TDRmn (master) set value+1} *100<br>0% output: TDRmp (slave) set value = 0000H<br>100% output: TDRmp (slave) set value≥{TDRmn (master) set value +1} |

Remark When the set value of TDRmp (slave) > {Set value of TDRmn (master) +1}, the duty cycle exceeds 100% but is 100% output.

The master channel is used as the interval timer mode. If the channel start trigger bit (TSmn) of the timer channel start register m (TSm) is set to “1”, an interrupt (INTTMmn) is output, and then the set value of the timer data register mn (TDRmn) is loaded into the timer count register mn (TCRmn), and the count is decremented by the count clock. When the count reaches “0000H”, the value of the TDRmn register is loaded into the TCRmn register again after the INTTMmn is output, and the count is decremented. Thereafter, this operation is repeated before setting the channel stop trigger bit (TTmn) of the timer channel stop register m (TTm) to “1”.

When used as PWM function, the master channel decrements the count and the period until "0000H" is counted as the PWM output (TOmp) period. The slave channel is used in single count mode. The value of TDRmp register is loaded into TCRmp register with INTTMmn of the master channel as the start trigger, and the count is decremented until “0000H”. When the count reaches “0000H”, INTTMmp is output and the next start trigger (INTTMmn of the master channel) is waited.

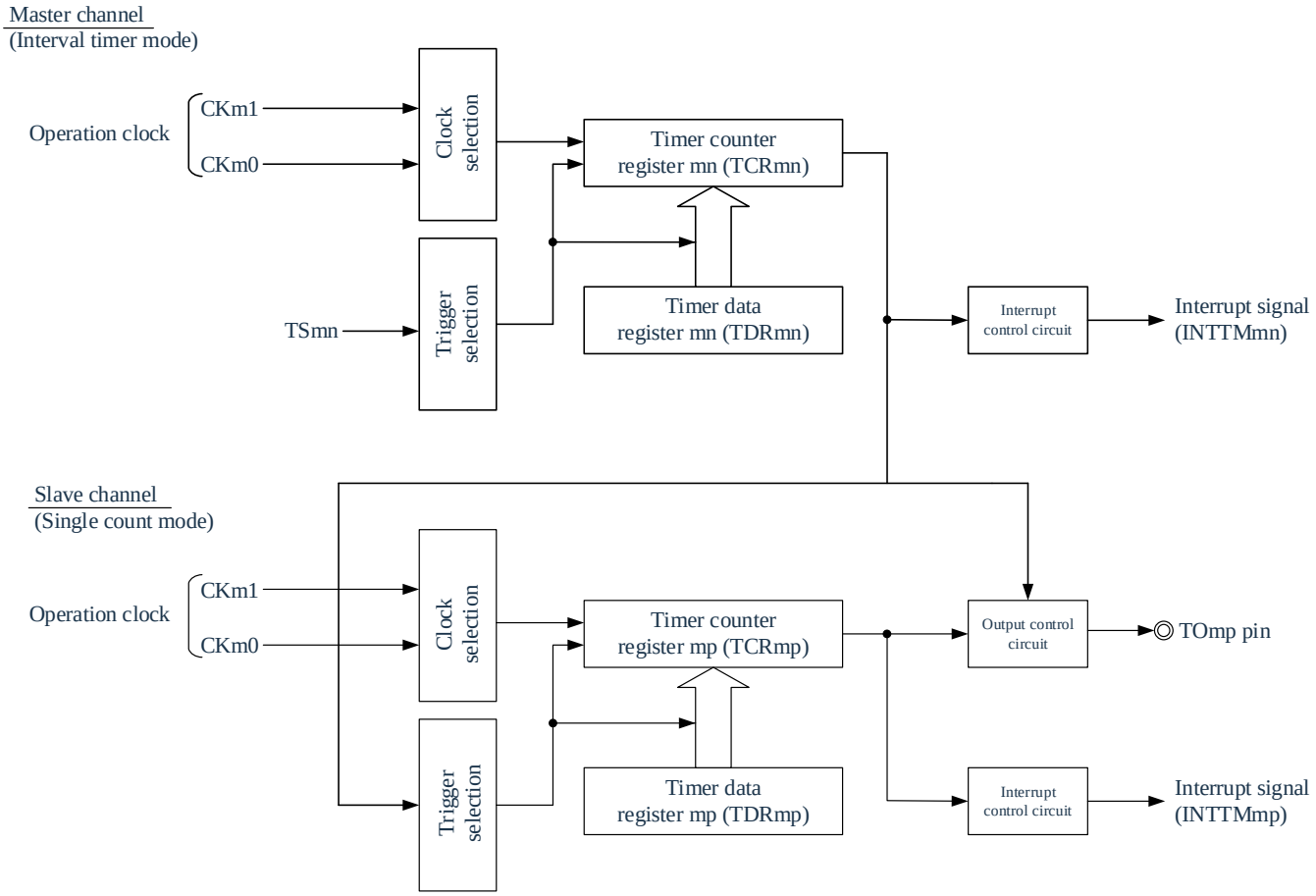
When used as PWM function, the slave channel decrements the count and the duty cycle of the PWM output (TOmp) for the period until “0000H” is counted.

After INTTMmn is generated from the master channel and 1 clock has elapsed, the PWM output (TOmp) becomes active and it becomes invalid when the value of TCRmp register of the slave channel is “0000H”.

Caution: To rewrite both timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel, a write access is necessary two times. The timing at which the values of the TDRmn and TDRmp registers are loaded to the TCRmn and TCRmp registers is upon occurrence of INTTMmn of the master channel. Thus, when rewriting is performed split before and after occurrence of INTTMmn of the master channel, the TOmp pin cannot output the expected waveform. To rewrite both the TDRmn register of the master and the TDRmp register of the slave, therefore, be sure to rewrite both the registers immediately after INTTMmn is generated from the master channel.

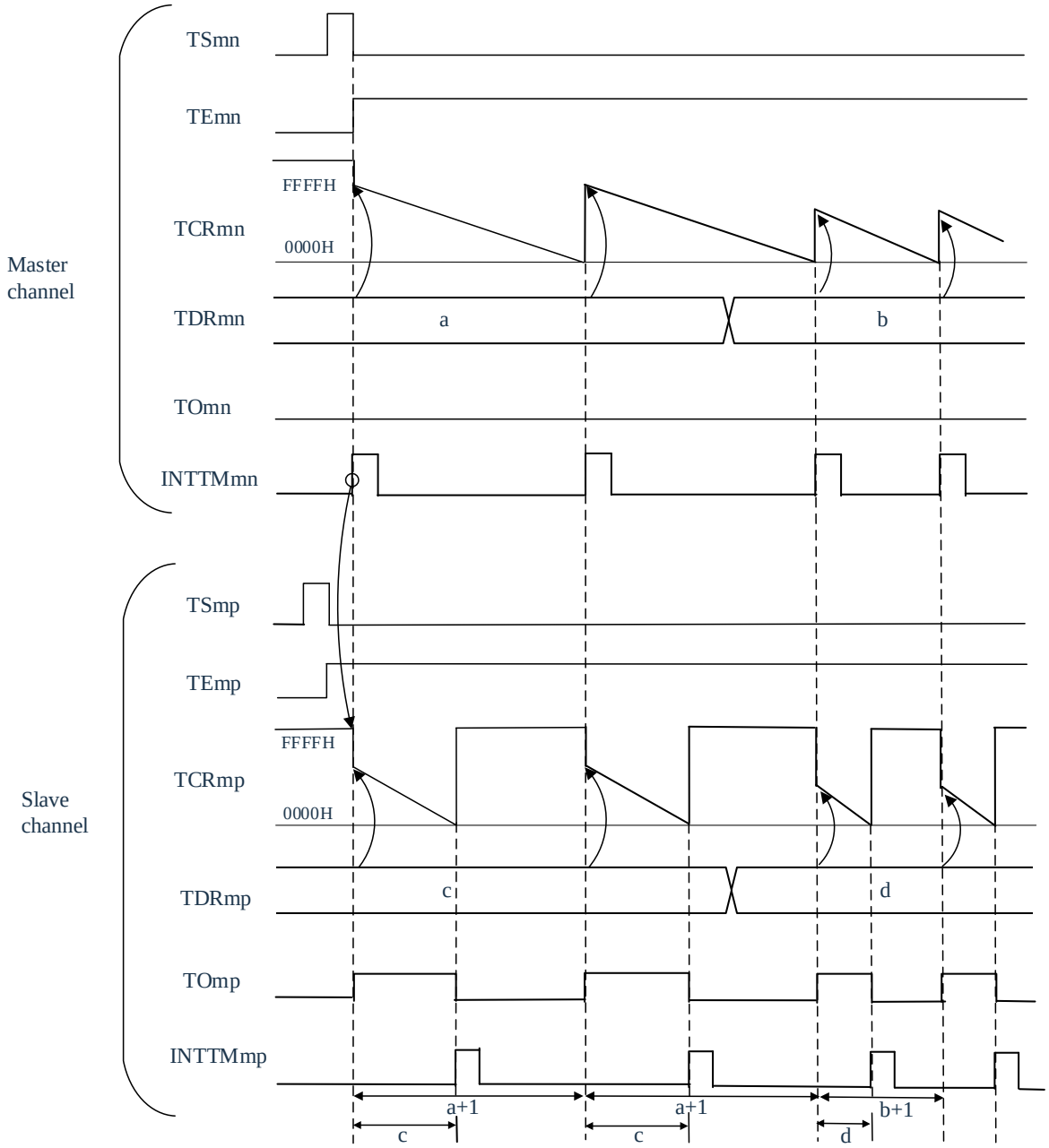
Note: m: unit number (m= 0) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Figure 5-35 Block diagram of operation as PWM function




Note: m: unit number (m= 0) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Figure 5-36 Example of basic timing operating as PWM function




Note 1: m: unit number (m= 0) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Note 2: TSmn,TSmp: Bits n and p of timer channel start register m (TSm)

TEmn, TEmp: Bits n and p of timer channel enable status register m (TEm)

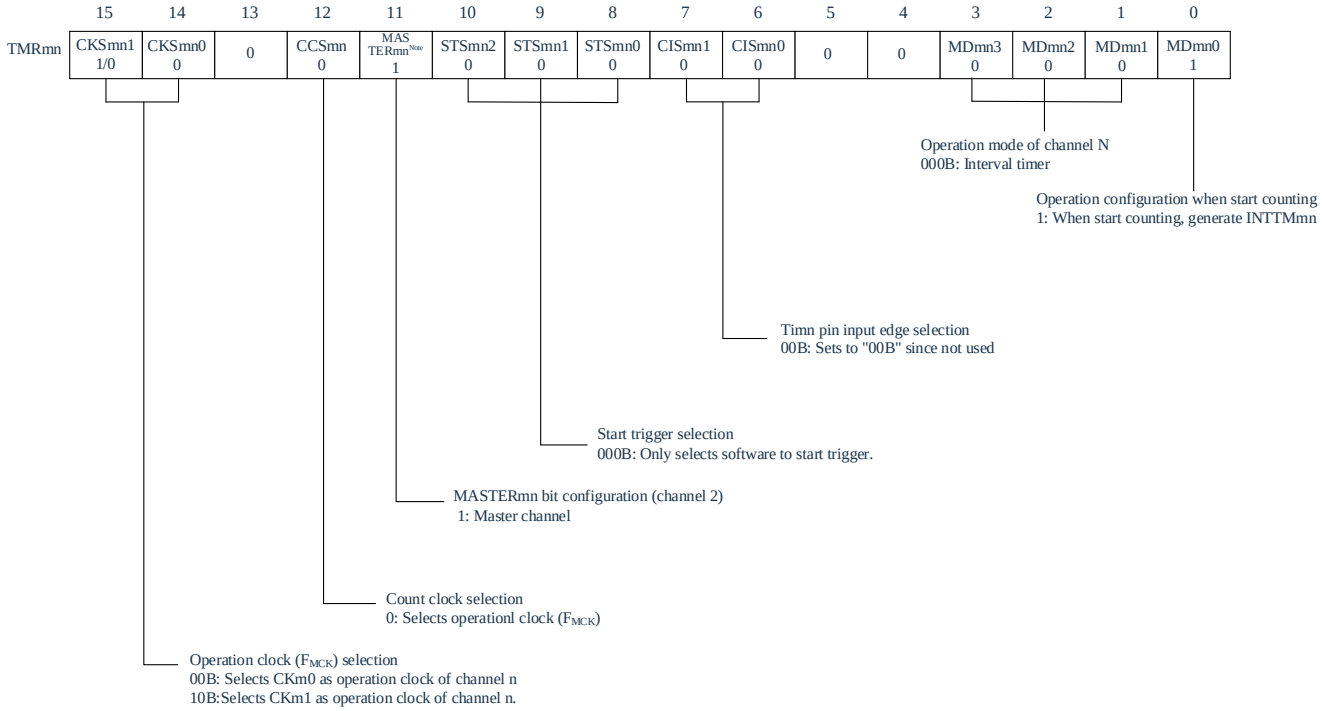
TCRmn, TCRmp: Timer count registers mn, mp (TCRmn, TCRmp)

TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)

TOmn, TOmp: Output signals of TOmn pin and TOmp pin

Figure 5-37 Example of basic timing operating as PWM function (master channel)

(a) Timer mode register mn (TMRmn)




(b) Timer output register m (TOm)

| | bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Output 0 from TOmn. |

(c) Timer output enable register m (TOEm).

| | bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stop the TOmn output operation by counting operation. |

(d) Timer output level register m(TOLm).

| | bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Set “0” when TOMmn=0 (master channel output mode). |

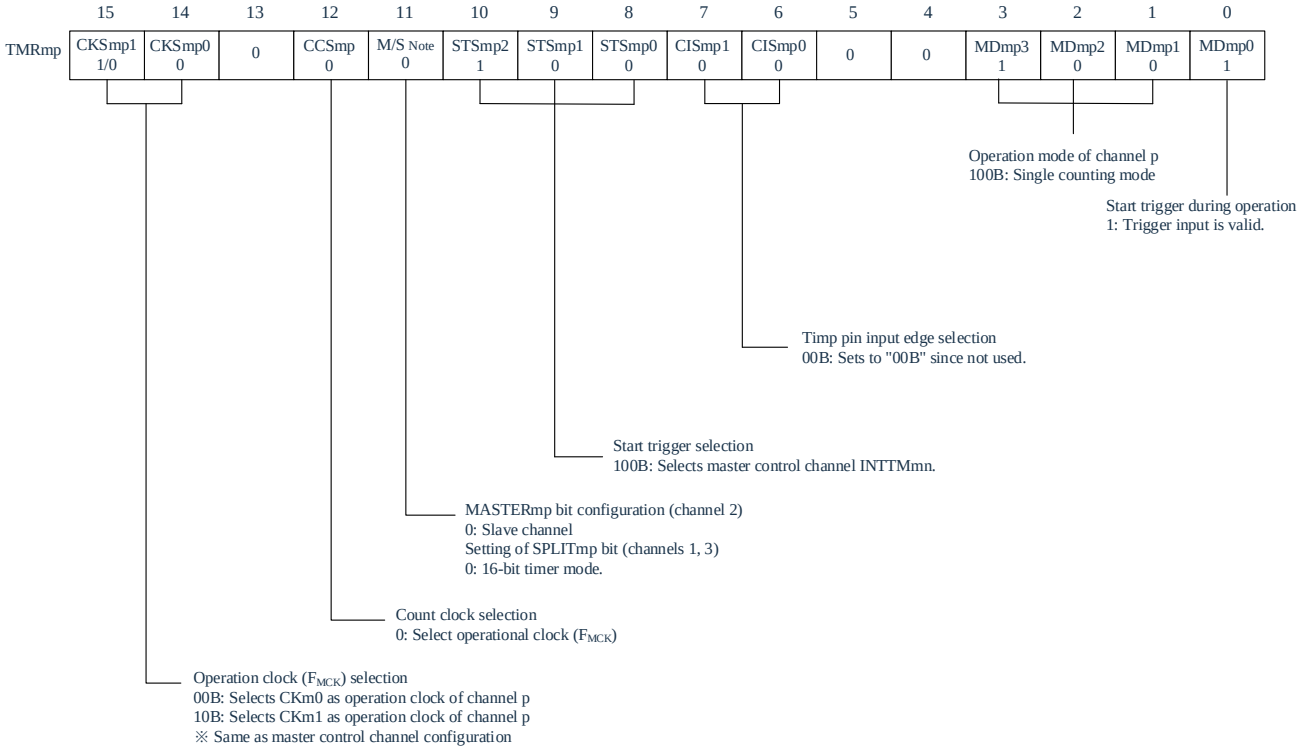
(e) Timer output mode register m (TOMm).

| | bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Set master channel output mode. |

Note: m: unit number (m= 0) n: master channel number (n=0, 2)

Figure 5-38 Example of basic timing operating as PWM function (slave channel)

(a) Timer mode register mp (TMRmp)




(b) Timer output enable register m (TOEm)

| | bit p | |
|---|---|---|
| TOm | TOmp 1/0 | 0: Outputs 0 from TOmp.<br>1: Outputs 1 from TOmp. |

(c) Timer output enable register m (TOEm)

| | bit p | |
|---|---|---|
| TOEm | TOEmp 1/0 | 0: Stops TOmp output performed by the counting operation.<br>1: Enables TOmp output performed by the counting operation. |

(d) Timer output level register m (TOLm)

| | bit p | |
|---|---|---|
| TOLm | TOLmp 1/0 | 0: Positive logic output (active high level)<br>1: Negative logic output (active low level) |

(e) Timer output mode register m (TOMm)

| | bit p | |
|---|---|---|
| TOMm | TOMmp 1 | 1: Sets slave channel output mode. |

Note 1: TMRm2: MASTERmp bit

TMRm1, TMRm3: SPLITmp bit

TMRm0: Fixed to “0”.

Note 2: m: unit number (m= 0) n: master channel number (n=0~2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Table 5-34 Procedure for the PWM function (1/2)

| | Software operation | Hardware status |
|---|---|---|
| Timer4 initial settings | | The input clock of timer unit m is in the stop-providing state.<br>(Stop providing clock, cannot write to each register) |
| | Set the TM4mEN bit of the peripheral enable register 0(PER0) to "1". | The input clock of timer unit m is in the providing state and the channels are in the stop state.<br>(Start providing clock, can write to each register) |
| | Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3. | |
| Initial setting of channels | Set the timer mode registers mn and mp (TMRmn, TMRmp) for the 2 channels used (to determine the operation mode of the channel). Set the interval (period) value for the timer data register mn (TDRmn) for the master channel and the duty cycle value for the TDRmp register for the slave channel. | The channel is in the stop state.<br>(Provides clock, and consumes some power) |
| | Slave channel setting<br>Set TOMmp bit of the timer output mode register m (TOMm)to"1" (slave channel output mode).<br>Set the TOLmp bit.<br>Set the TOmp bit to determine the initial level of the TOmp output.<br>Set the TOEmp bit to "1" and enable TOmp output.<br>Set the Port Register and Port Mode Register to "0". | The TOmp pin is in Hi-Z output state.<br>When the port mode register is in output mode and the port register is "0", the initially set level of TOmp is output.<br>The TOmp remains unchanged because the channel is in the stop state.<br>The TOmp pin outputs the level set by the TOmp. |

Table 5-34 Procedure for the PWM function (2/2)

| | Software operation | Hardware status |
|---|---|---|
| Start operation | Set the TOEmp bit to “1” (only limited to restart operation).<br>Set both the TSmn bit (master) and TSmp bit (slave) of the timer channel start register m (TSm) to "1". →<br>The operation automatically returns to "0" because the TSmn and TSmp bits are trigger bits. | The TEmn and TEmp bits become "1". The master channel starts counting and generates INTTMmn. With this as a trigger, the slave channel also starts counting. |
| In operation | The setting values of the TMRmn and TMRmp registers and the TOMmn bit, TOMmp bit, TOLmn bit, and TOLmp bit cannot be changed.<br>Able to change the setting value of the TDRmn register and the TDRmp register after the master channel has generated INTTMmn.<br>The TCRmn and TCRmp registers can be read at any time.<br>The TSRmn and TSRmp registers are not used. | The master channel loads the value of the TDRmn register into the timer count register mn (TCRmn) and perform decremental counting. If TCRmn counts till "0000H", then generating INTTMmn. At the same time, load the TDRmn register value into the TCRmn register and restart decremental counting.<br>The slave channel use INTTMmn of master channel as a trigger, load the TDRmp register value into the TCRmp register and counter start decremental counting. After INTTMmn is output from the master channel and one count clock has elapsed, the output level of TOmp is set to an active level. Then, if TCRmp counts to "0000H", it stops counting after setting the output level of TOmp to an invalid level.<br>Thereafter, repeat this operation. |
| Stop operation | Set the TTmn bit (master) and TTmp bit (slave) to "1" at the same time. →<br>The operation automatically returns to "0" because the TTmn and TTmp bits are trigger bits. | TEmn, TEmp = 0, and count operation stops.<br>The TCRmn and TCRmp registers hold count value and stop.<br>The TOmp output is not initialized but holds current |
| | Set the TOEmp bit of slave channel to "0" and set the value for the TOmp bit. → | The TOmp pin outputs the TOmp set level. |
| Timer4 stop | To maintain the output level of the TOmp pin:<br>Set TOmp bit to "0" after setting the value to be held for the port register. →<br>When holding the TOmp pin output level is not necessary:<br>No need to set. | The TOmp pin output level is held by port function. |
| | Set the TM4mEN bit of the PER0 register to "0". → | The input clock of timer unit m is in the stop-providing state.<br>Initialize all circuits and the SFR for each channel.<br>(TOmn bit becomes "0" and TOmp pin becomes port function) |

Restart operation

Note 1: m: unit number (m= 0) n: master channel number (n=0)

p: slave channel number q: slave channel number n<p<q≤3 (p and q are integers greater than n)

Note 2: The TSmn bit of the slave channel cannot be set to “1”.

## 5.8.3 Operation as Multiple PWM Output Function

This is a function that extends the PWM function and uses multiple slave channels for multiple PWM outputs with different duty cycles. For example, when using 2 slave channels in pairs, the period and duty cycle of the output pulse can be calculated by using the following equation:

| |
|---|
| Pulse period= {TDRmn(master) set value +1} × count clock period<br>Duty cycle1[%] = {TDRmp (slave 1) set value} / {TDRmn (master) set value+1}*100<br>Duty cycle2[%] = {TDRmq (slave 2) set value} / {TDRmn(master) set value+1}*100 |

Note: When the set value of TDRmp (slave 1) > {the set value of TDRmn (master) + 1} or {the set value of TDRmq (slave 2)} > {the set value of TDRmn (master) + 1}, the duty cycle exceeds 100%, but is 100% output.

In interval timer mode, the timer count register mn (TCRmn) of the master channel operates and counts the period. In single count mode, the TCRmp register of slave channel 1 operates and counts the duty cycle and outputs the PWM waveform from the TOmp pin. The TCRmp register loads the value of timer data register mp (TDRmp), using INTTMmn of the master channel as a start trigger, and starts counting down. When TCRmp = “0000H”, the TCRmp outputs INTTMmp and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of TOmp becomes valid after INTTMmn has been generated from the master channel and after 1 count clock, if TCRmp becomes "0000H", it becomes invalid.

In the same way as the TCRmp register of the slave channel 1, the TCRmq register of the slave channel 2 operates in single count mode, counts the duty cycle, and outputs a PWM waveform from the TOmq pin. The TCRmq register loads the value of the TDRmq register, using INTTMmn of the master channel as a start trigger, and starts counting down. When TCRmq = “0000H”, the TCRmq register outputs INTTMmq and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of the TOmq becomes active one count clock after generation of INTTMmn from the master channel, and inactive when TCRmq = 0000H.

When channel 0 is used as the master channel as above, up to 3 types of PWM signals can be output at the same time.

Caution: To rewrite the timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel 1 at the same time, at least 2 write accesses are required. Because the values of TDRmn register and TDRmp register are loaded into the TCRmn register and TCRmp register when the master channel generates INTTMmn, the TOmp pin cannot output the expected waveform if rewriting is performed before and after the master channel generates INTTMmn respectively. Therefore, to rewrite both the master TDRmn register and the slave TDRmp register, these two registers must be rewritten immediately after the master channel generates INTTMmn (the same applies to the TDRmq register of slave channel 2).

Note: m: unit number (m= 0) n: master channel number (n=0)

p: slave channel number

q: slave channel number $n < p < q \leq 3$ (p and q are integers greater than n)

Figure 5-39 Block diagram of operation as multiple PWM output function (output two types of PWMs)

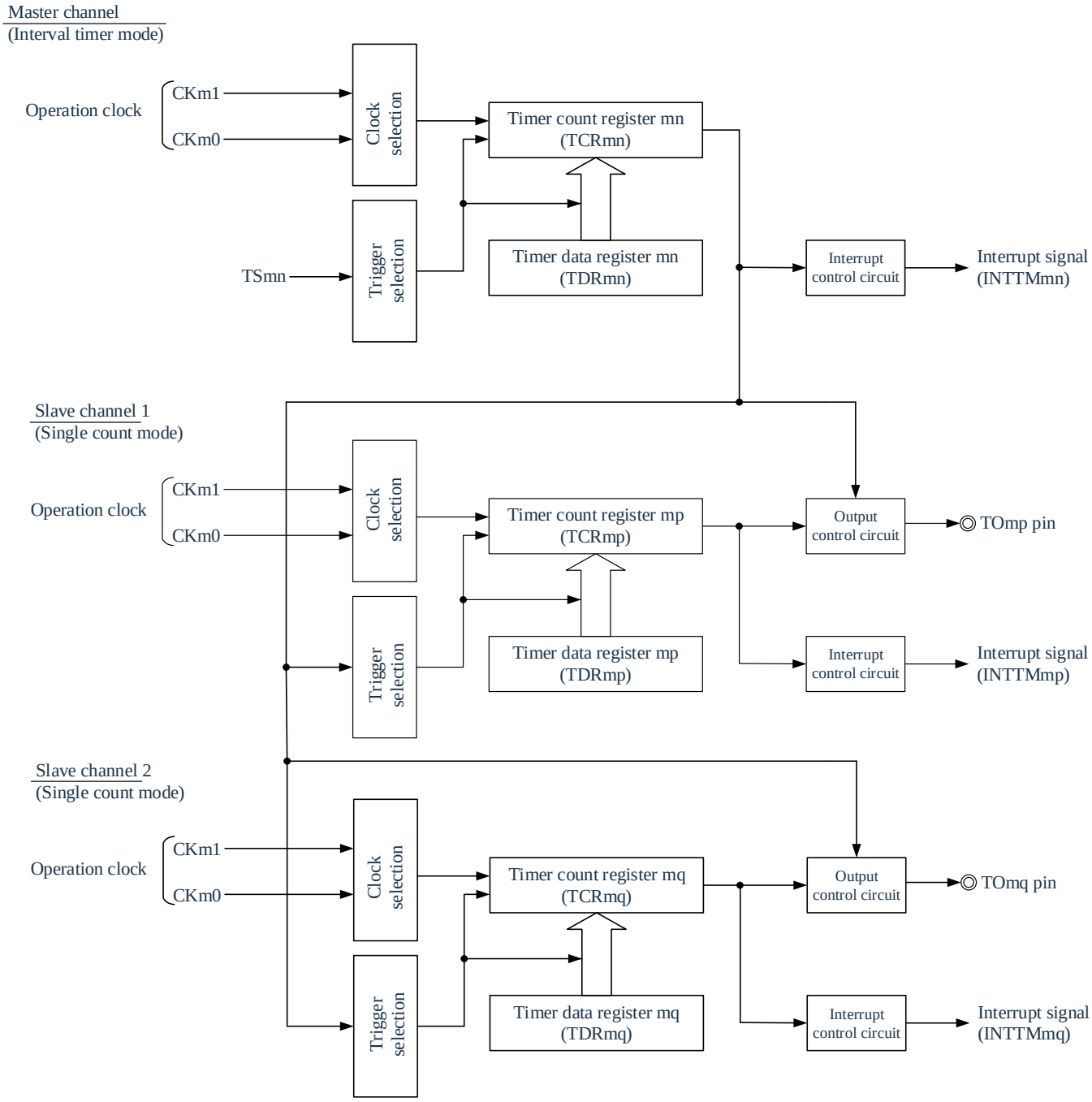
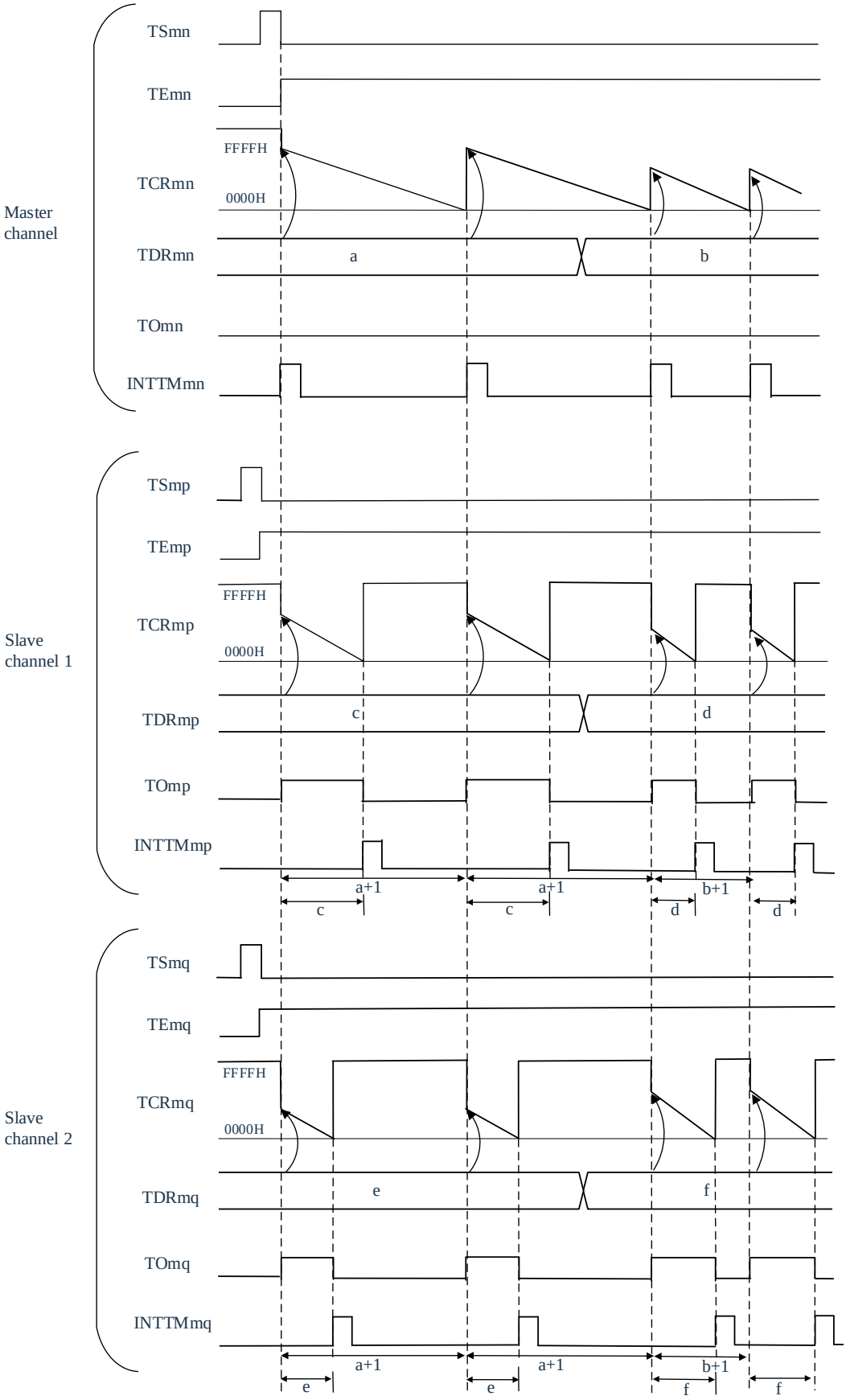


Figure 5-40 Example of basic timing operating as multiple PWM output function (output two types of PWMs)




Note 1: m: unit number (m= 0) n: master channel number (n=0)

p: slave channel number q: slave channel number n<p<q≤3 (p and q are integers greater than n)

Note 2: TSmn, TSmp, TSmq: Bits n, p and q of timer channel start register m (TSm)

TEmn, TEmp, TEmq: Bits n, p and q of timer channel enable status register m (TEm)

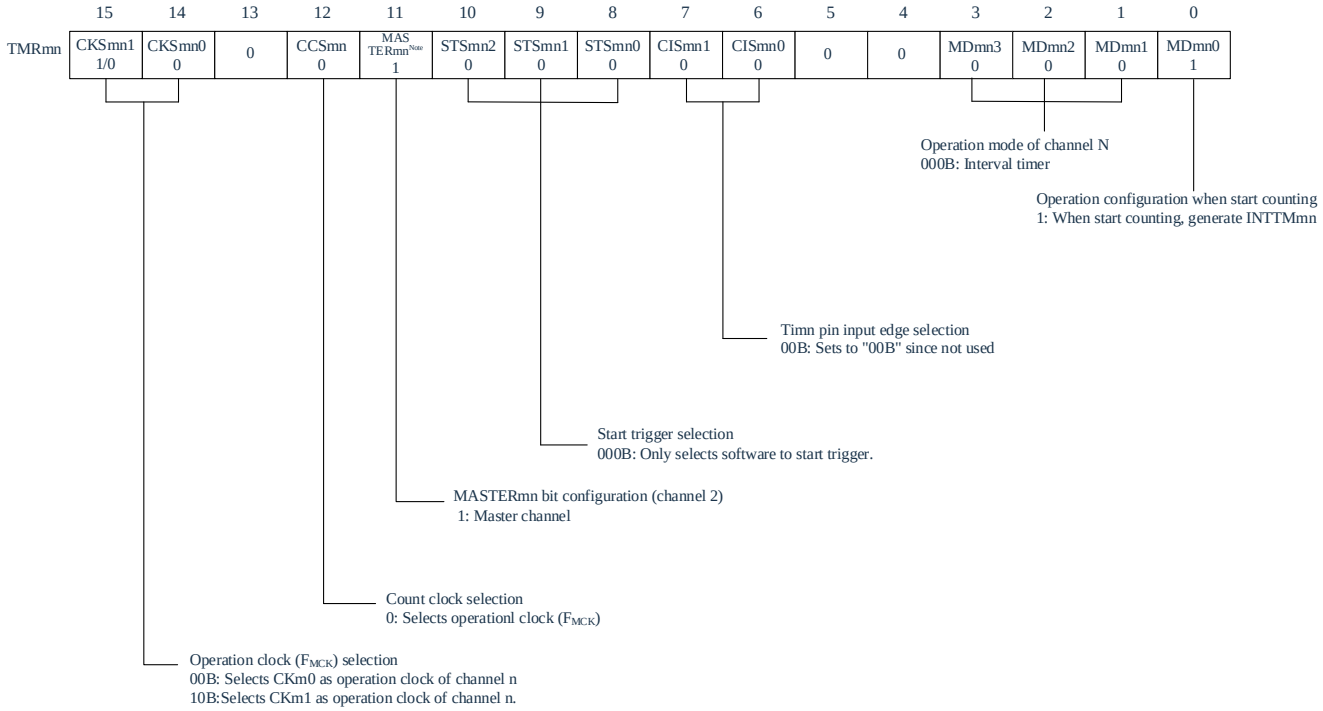
TCRmn, TCRmp, TCRmq: Timer count registers mn, mp, mq (TCRmn, TCRmp, TCRmq)

TDRmn, TDRmp, TDRmq: Timer data registers mn, mp, mq (TDRmn, TDRmp, TDRmq)

TOmn, TOmp, TOmq: TOmn, TOmp, TOmq pin output signals

Figure 5-41 Example of register contents setting for multiple PWM output function (master channel)

(a) Timer mode register mn (TMRmn)




(b) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

(c) Timer output enable register m (TOEm)

| | bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops the TOmn output operation by counting operation. |

(d) Timer output level register m (TOLm)

| | bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode) |

(e) Timer output mode register m (TOMm)

| | bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

Note 1: TMRm2: MASTERmn bit

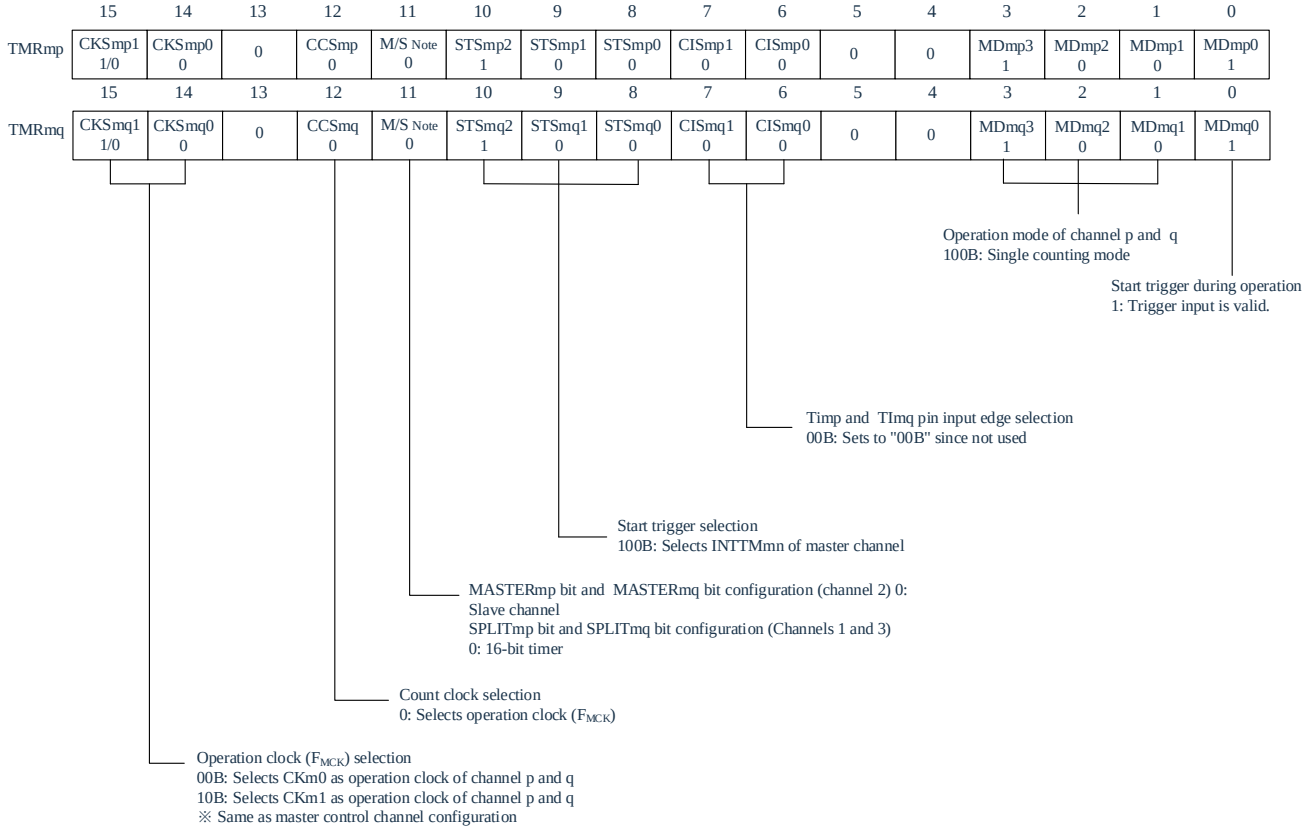
TMRm1, TMRm3: SPLITmn bit

Note 2: m: unit number (m= 0) n: channel number (n=0)

Figure 5-42 Example of register contents setting for multiple PWM output function (slave channel)

(output two types of PWMs)

(a) Timer mode registers mp, mq (TMRmp, TMRmq)




(b) Timer output enable register m (TOEm)

| TOm | bit q | bit p | |
|---|---|---|---|
| | TOmq 1/0 | TOmp 1/0 | 0: utputs 0 from TOmp or TOmq.<br>1: Outputs 0 from TOmp or TOmq. |

(c) Timer output enable register m (TOEm)

| TOEm | bit q | bit p | |
|---|---|---|---|
| | TOEmq 1/0 | TOEmp 1/0 | 0: Stops the TOmp or TOmq output operation by counting operation.<br>1: Enables the TOmp or TOmq output operation by counting operation. |

(d) Timer output level register m (TOLm)

| TOLm | bit q | bit p | |
|---|---|---|---|
| | TOLmq 1/0 | TOLmp 1/0 | 0: Positive logic output (active high level)<br>1: Negative logic output (active low level) |

(e) Timer output mode register m (TOMm)

| TOMm | bit q | bit p | |
|---|---|---|---|
| | TOMmq 1 | TOMmp 1 | 1: Sets slave channel output mode. |

Note: m: unit number (m= 0), n: masterh channel number (n=0)

p: slave channel number q: slave channel number, n＜p＜q≤3 (p and q are integers greater than n)

Table 5-35 Procedure for the multiple PWM output function (output two types of PWMs) (1/2)

| | Software operation | Hardware status |
|---|---|---|
| Timer4 initial settings | | The input clock of timer unit m is in the stop-providing state.<br>(Stop providing clock, cannot write to each register) |
| | Set the TM4mEN bit of the peripheral enable register 0(PER0) to "1". → | The input clock of timer unit m is in the providing state and the channels are in the stop state.<br>(Start providing clock, can write to each register) |
| | Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 and CKm1. | |
| Initial setting of channels | Set the timer mode registers mn, mp, (TMRmn, TMRmp,) for each channel used (to determine the channel operation mode).<br>Set the interval (period) value for the master channel's timer data register mn (TDRmn), and set the duty cycle value for the slave channel's TDRmp register and TDRmq register. | The channel is in the stop state.<br>(Provides clock, and consumes some power) |
| | Slave channel setting<br>Set TOMmp and TOMmq bits of the timer output mode register mn (TOMmn) to"1" (slave channel output mode).<br>Set the TOLmp and TOLmq bits to "0".<br>Set the TOmp and TOmq bits and determine the initial output level of the TOmp and TOmq bits. →<br>Set the TOEmp amd TOEmq bits to "1" and enable TOmp and TOmq output. →<br>Set the Port Register and Port Mode Register to "0". → | The TOmp pin is in Hi-Z output state.<br>When the port mode register is in output mode and the port register is "0", the TOmp and TOmq initial set levels are output.<br>The TOmp and TOmq remains unchanged because the channel is in the stop state.<br>The TOmp pin and TOmq pin output the levels set by the TOmp and TOmq. |

Table 5-35 Procedure for the multiple PWM output function (output two types of PWMs) (2/2)

| | | Software operation | Hardware status |
|---|---|---|---|
| Restart operation | Start operation | (Sets the TOEmp and TOEmq (slave) bits to 1 only when resuming operation.)<br>The TSmn bit (master), and TSmp and TSmq (slave) bits of timer channel start register m (TSm) are set to 1 at the same time.<br>The TSmn, TSmp, and TSmq bits automatically return to 0 because they are trigger bits. | TEmn = 1, TEmp, TEmq = 1<br>When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting. |
| | In operation | Set values of the TMRmn, TMRmp, TMRmq registers, TOMmn, TOMmp, TOMmq, TOLmn, TOLmp, and TOLmq bits cannot be changed.<br>Set values of the TDRmn, TDRmp, and TDRmq registers can be changed after INTTMmn of the master channel is generated.<br>The TCRmn, TCRmp, and TCRmq registers can always be read.<br>The TSRmn, TSRmp, and TSR0q registers are not used. | The counter of the master channel loads the TDRmn register value to timer count register mn (TCRmn) and counts down. When the count value reaches TCRmn = 0000H, INTTMmn output is generated. At the same time, the value of the TDRmn register is loaded to the TCRmn register, and the counter starts counting down again.<br>At the slave channel 1, the values of the TDRmp register are transferred to the TCRmp register, triggered by INTTMmn of the master channel, and the counter starts counting down. The output levels of TOmp become active one count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped.<br>At the slave channel 2, the values of the TDRmq register are transferred to TCRmq register, triggered by INTTMmn of the master channel, and the counter starts counting down. The output levels of TOmq become active one count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmq = 0000H, and the counting operation is stopped.<br>After that, the above operation is repeated. |
| — | Stop operation | The TTmn bit (master), TTmp, and TTmq (slave) bits are set to 1 at the same time.<br>The TTmn, TTmp, and TTmq bits automatically return to 0 because they are trigger bits. | TEmn, TEmp, TEmq = 0, and count operation stops.<br>The TCRmn, TCRmp, and TCRmq registers hold count value and stop.<br>The TOmp and TOmq output are not initialized but hold current status. |
| | | The TOEmp and TOEmq bits of slave channels are cleared to 0 and value is set to the TOmp and TOmq bits. | The TOmp and TOmq pins output the TOmp and TOmq set levels. |
| | Timer4 stop | To hold the TOmp and TOmq pin output levels<br>Clears the TOmp and TOmq bits to 0 after the value to be held is set to the port register.<br>When holding the TOmp and TOmq pin output levels are not necessary:<br>Setting not required | The TOmp and TOmq pin output levels are held by port function. |
| | | The TM4mEN bit of the PER0 register is cleared to 0. | The input clock of timer unit m is in the stop-providing state.<br>Initialize all circuits and the SFR for each channel.<br>(TOmp bit and TOmq bit become "0" and TOmp pin and TOmq pin become port function) |

Note: m: unit number (m= 0) n: master channel number (n=0)

p: slave channel number q: slave channel number $n < p < q \leq 3$ (p and q are integers greater than n)

# Chapter 6 LSITIMER 12-Bit Interval Timer

## 6.1 Functions of 12-Bit Interval Timer

An interrupt (INTIT) is generated at any previously specified time interval. It can be utilized for wakeup from sleep mode, deep sleep mode and partial power-down mode.

## 6.2 Configuration of 12-Bit Interval Timer

The 12-bit interval timer includes the following hardware.

Table 6-1 Configuration of 12-bit interval timer

| Item | Configuration |
|---|---|
| Counter | 12-bit counter |
| Control register | 12-bit interval timer control register (CON0) |

## 6.3 Register Mapping

(CON0 base address = 0x4004_4B50) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CON0 | 0x000 | R/W | 12-bit interval timer control register | 0xFFF |

## 6.4 12-Bit Interval Timer Control Register (CON0)

This register is used to set up the starting and stopping of the 12-bit interval timer operation and to specify the timer compare value.

The CON0 register can be set by a 12-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to “FFFH”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15 | RINTE | 12-bit Interval timer operation control<br>0: Count operation stopped (count clear)<br>1: Count operation started | 0 |
| 14:12 | -- | Reserved | -- |
| 11:0 | ITCMP | Specification of the 12-bit interval timer compare value | 0xFFF |

The setting of the ITCMP compare value and the calculation of the related interrupt period are shown in the following table (for reference only):

| ITCMP[11]~ITCMP[0] | Specification of the 12-bit interval timer compare value |
|---|---|
| 001H | These bits generate a fixed-cycle interrupt of “Count Clock Cycle (ITCMP Set value + 1)” |
| •<br>•<br>• | |
| FFFH | |
| 000H | Settings are prohibited. |
| Example interrupt cycles when 001H or FFFH is specified for ITCMP[11] ~ ITCMP[0]<br>ITCMP[11]~ITCMP[0]=001H, count clock: $F_{clk}$=15kHz 1/15[kHz]*(1+1)=0.13333 [ms]<br>ITCMP[11]~ITCMP[0]=FFFH, count clock: $F_{clk}$=15kHz 1/15[kHz]*(4095+1)=273. 06667[ms] | |

Note 1: Before changing the RINTE bit from 1 to 0, use the interrupt mask flag register to disable the INTIT interrupt servicing. When the operation starts (from 0 to 1) again, clear the ITIF flag, and then enable the interrupt servicing.

Note 2: The value read from the RINTE bit is applied one count clock cycle after setting the RINTE bit.

Note 3: When setting the CON0 register after returned from normal operation mode and entering sleep mode again, confirm that the written value of the CON0 register is reflected, or wait that more than one clock of the count clock has elapsed. Then enter sleep mode.

Note 4: Only change the setting of the ITCMP11 to ITCMP0 bits when RINTE = 0.

Note 5: However, it is possible to change the settings of the ITCMP11 to ITCMP0 bits at the same time as when changing RINTE from 0 to 1 or 1 to 0.

# 6.5 12-Bit Interval Timer Operation

## 6.5.1 12-Bit Interval Timer Operation Timing

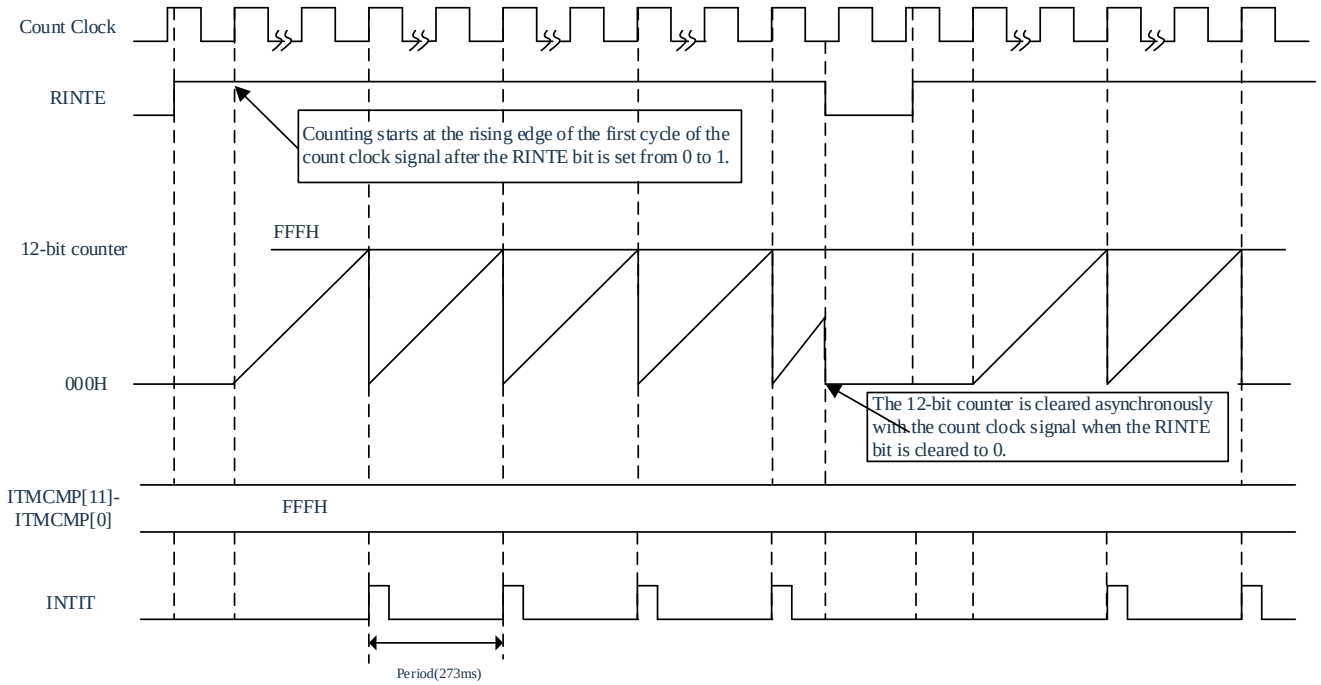
The count value specified for the ITCMP11 to ITCMP0 bits is used as an interval to operate a 12-bit interval timer that repeatedly generates interrupt requests (INTIT). When the RINTE bit is set to 1, the 12-bit counter starts counting.

When the 12-bit counter value matches the value specified for the ITCMP11 to ITCMP0 bits, the 12-bit counter value is cleared to 0, counting continues, and an interrupt request signal (INTIT) is generated at the same time.

The basic operation of the 12-bit interval timer is shown in Figure 6-1.

Figure 6-1 12-bit interval timer operation timing

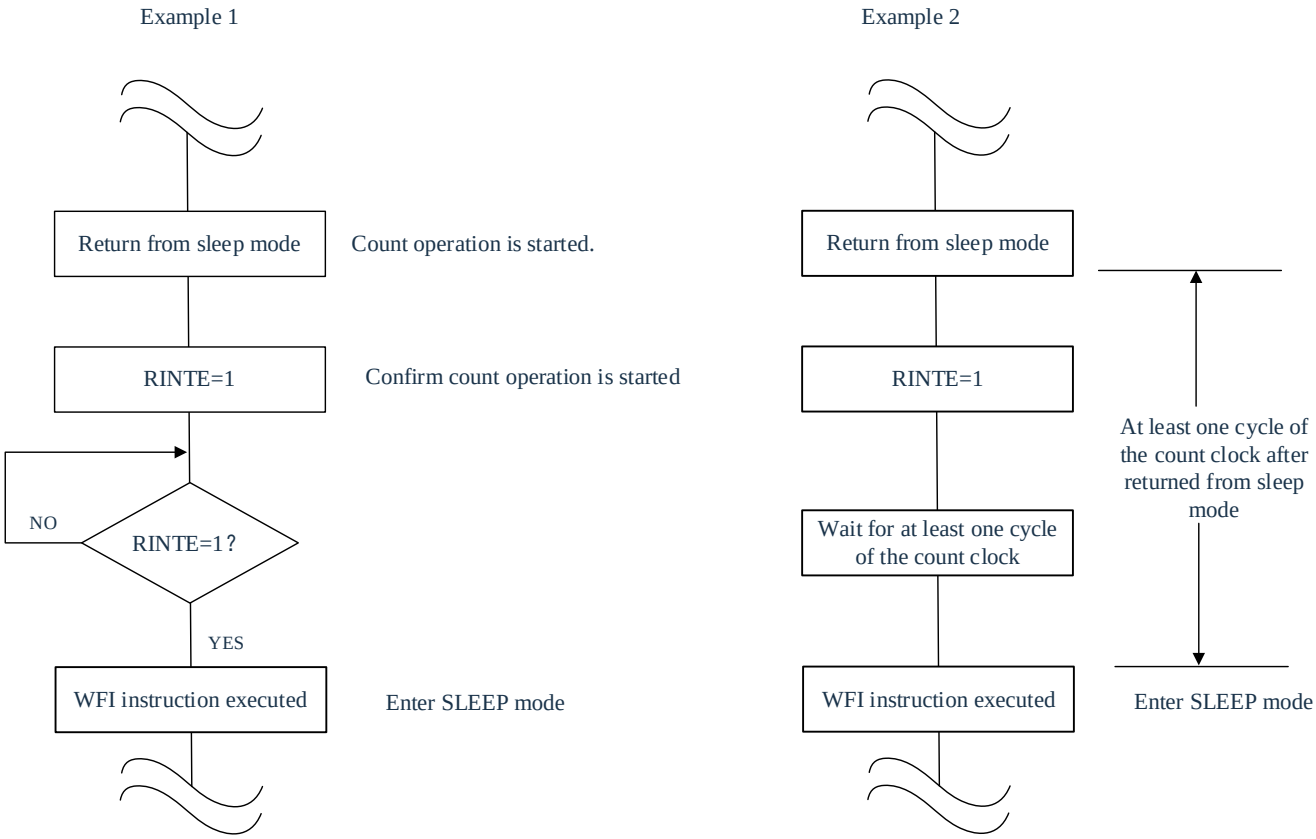
(ITCMP[11]~ITCMP[0]=FFFH, count clock: $f_{clk}$=15kHz)



## 6.5.2 Start of Count Operation and Re-Enter to Sleep Mode After Returned From Sleep Mode

When setting the RINTE bit after returned from sleep mode and entering sleep mode again, write 1 to the RINTE bit, and confirm the written value of the RINTE bit is reflected or wait for at least one cycle of the count clock. Then, enter sleep mode.

- After setting RINTE to 1, confirm by polling that the RINTE bit has become 1, and then enter sleep mode (see Example 1 in the Figure below).
- After setting RINTE to 1, wait for at least one cycle of the count clock and then enter sleep mode (see Example 2 in the Figure blow).



# Chapter 7 Clock Output Controller

## 7.1 Functions of Clock Output Controller

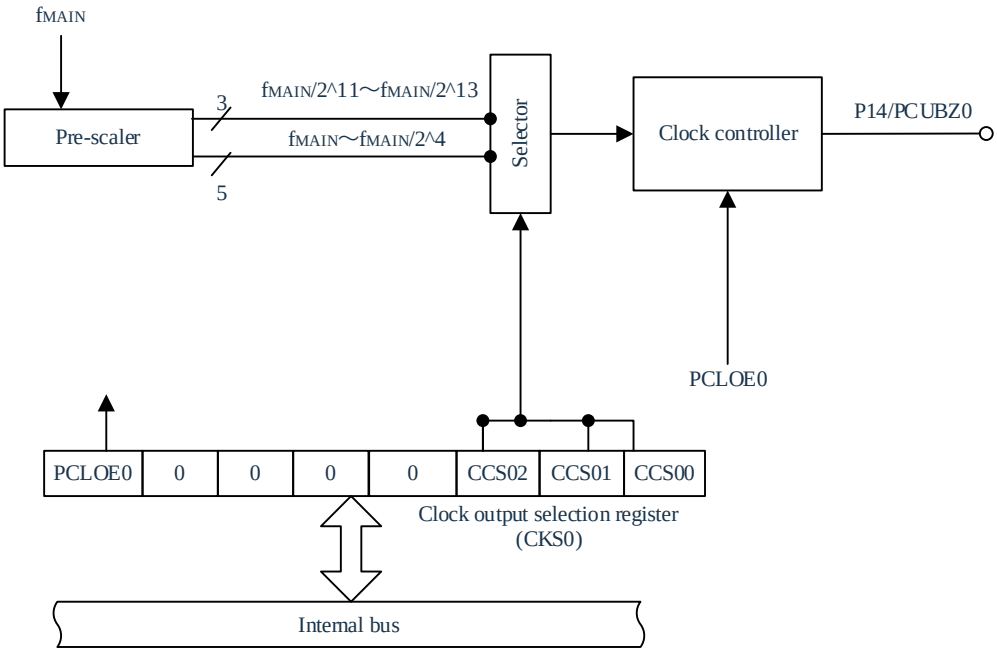
The clock output provides the clock signal to peripheral ICs.

This product features a clock output pin, PCUBZ0, which, after alternating, can function as a clock output.

The clock signal output from the PCUBZ0 pin is selected by the Clock Output Select Register (CKS0).

The block diagram of the clock output control circuit is shown in Figure 7-1.

Figure 7-1 Block diagram of clock output controller




Note 1: For the frequencies that can be output from PCUBZ0 pin, please refer to “AC Characteristics” in the data sheet.

## 7.2 Configuration of Clock Output Controller

The clock output controller consists of the following hardware.

Table 7-1 Structure of clock output controller

| Item | Structure |
|---|---|
| Control register | Clock output selection register (CKS0)<br>Port mode control register (PMCxx),<br>Port mode register (PMxx),<br>Port alternate control register (PxxCFG) |

## 7.3 Register Mapping

(CKS0 base address = 0x4004_0FA5)

RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CKS0 | 0x000 | R/W | Clock output selection register | 0x0 |

## 7.3.1 Clock Output Selection Register (CKS0)

This register enables or disables the clock output from the PCUBZ0 pin and sets the output clock. The clock output from the PCUBZ0 pin is selected through the CKS0 register. The CKS0 register is set by 8-bit memory manipulation instructions. After a reset signal is generated, the value of this register is set to “00H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | PCLOE0 | PCUBZ0 pin output enable/disable specification<br>0: Output disable (default)<br>1: Output enable | 0x0 |
| 6:3 | -- | Reserved | -- |
| 2:0 | CCS0 | PCUBZ0 pin output clock selection | 0x0 |

The specific clock selection for the PCUBZ0 pin output is as follows:

| CCS0[2] | CCS0[1] | CCS0[0] | PCUBZ0 pin output clock selection |
|---|---|---|---|
| 0 | 0 | 0 | $F_{MAIN}$ |
| 0 | 0 | 1 | $F_{MAIN}/2$ |
| 0 | 1 | 0 | $F_{MAIN}/2^2$ |
| 0 | 1 | 1 | $F_{MAIN}/2^3$ |
| 1 | 0 | 0 | $F_{MAIN}/2^4$ |
| 1 | 0 | 1 | $F_{MAIN}/2^{11}$ |
| 1 | 1 | 0 | $F_{MAIN}/2^{12}$ |
| 1 | 1 | 1 | $F_{MAIN}/2^{13}$ |

Note 1: Use the output clock within a range of 16 MHz. For details, please refer to “AC Characteristics” in the data sheet.

Note 2: Change the output clock after disabling clock output (PCLOE0=0).

Note 3: To shift to deep sleep mode, set PCLOE0=0 before executing the WFI instruction.

Note 4: $F_{MAIN}$: Main system clock frequency

## 7.4 Registers for Configuring Clock Output Port Functions

This product features a clock output pin, PCUBZ0, which, after alternating, can serve as a clock output.

When using the clock output function, it is necessary to configure the Port Alternate Function Configuration Register (PxxCFG), Port Register (Pxx), Port Mode Register (PMxx), and Port Mode Control Register (PMCxx). For detailed information, please refer to Chapter 2: Pin Functions.

For the port configured as a clock output pin, the corresponding bits in the Port Register (Pxx), Port Mode Register (PMxx), and Port Mode Control Register (PMCxx) must be set to 0.

(Example) Configuring P14 as a clock output (PCUBZ0):

Set the bit P14 in the Port Register 1 (P14) to 0.

Set the bit PM14 in the Port Mode Register 1 to 0.

Set the bit PMC14 in the Port Mode Control Register 1 to 0.

Set the P14CFG of the Port Alternating Function Configuration Register to 0x02.

# 7.5 Operation of Clock Output Controller

The corresponding port can serve as a clock output after alternating.

The clock output from the PCUBZ0 pin is selected by the Clock Output Selection Register (CKS0).

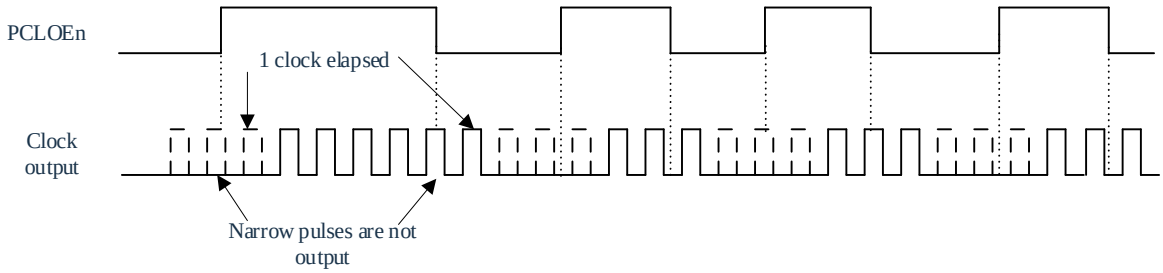
## 7.5.1 Operation as Output Pin

The corresponding port can be used as a clock output after alternating.

1) Configure the Port Alternating Function Configuration Register (PmnCFG), and set the corresponding bits in the Port Register (Pxx), Port Mode Register (PMxx), and Port Mode Control Register (PMCxx) for the port used as the PCUBZ0 pin to 0.
2) Select the output frequency (output disabled by default) via bits 0 to 2 (CCS0 to CCS2) in the Clock Output Selection Register (CKS0) of the PCUBZ0 pin.
3) Set the bit 7 (PCLOE0) of the CKS0 register to 1 to enable the clock output.

Note 1: When used as a clock output, the control circuit begins or stops the clock output one clock cycle after enabling or disabling the clock output (PCLOE0 bit). At this point, no narrow pulse is output. The timing of enabling or disabling the output and the clock output is shown in Figure 7-2.

Figure 7-2 PCUBZ0 pin output clock timing




# 7.6 Cautions of Clock Output/Buzzer Output Controller

When the main system clock is selected for the PCUBZ0 output (PCLOE0=0), if deep sleep mode is entered within 1.5 clock cycles output from the PCUBZ0 pin after the output is disabled (PCLOE0=0), the PCUBZ0 output width becomes shorter.

# Chapter 8 Watchdog Timer (WDT)

## 8.1 Functions of Watchdog Timer

The counting operation of the watchdog timer is set by the option byte (000C0H). The watchdog timer operates on the low-speed on-chip oscillator clock ($F_{IL}$).

The watchdog timer is used to detect an inadvertent program loop. If a program loop is detected, an internal reset signal is generated.

Program loop is detected in the following cases.

(1) If the watchdog timer counter overflows

(2) If data other than “ACH” is written to the WDTE register

(3) If data is written to the WDTE register during a window close period

When a reset occurs due to the watchdog timer, bit 4 (WDTRF) of the reset control flag register (RESF) is set to 1. For details of the RESF register, see Chapter 26 Reset Function. When 75% of the overflow time + 1/2 $F_{IL}$ is reached, an interval interrupt can be generated.

## 8.2 Configuration of Watchdog Timer

The watchdog timer includes the following hardware.

Table 8-1 Configuration of watchdog timer

| Item | Configuration |
|---|---|
| Counter | Internal counter (17 bits) |
| Control register | Watchdog timer enable register (WDTE) |

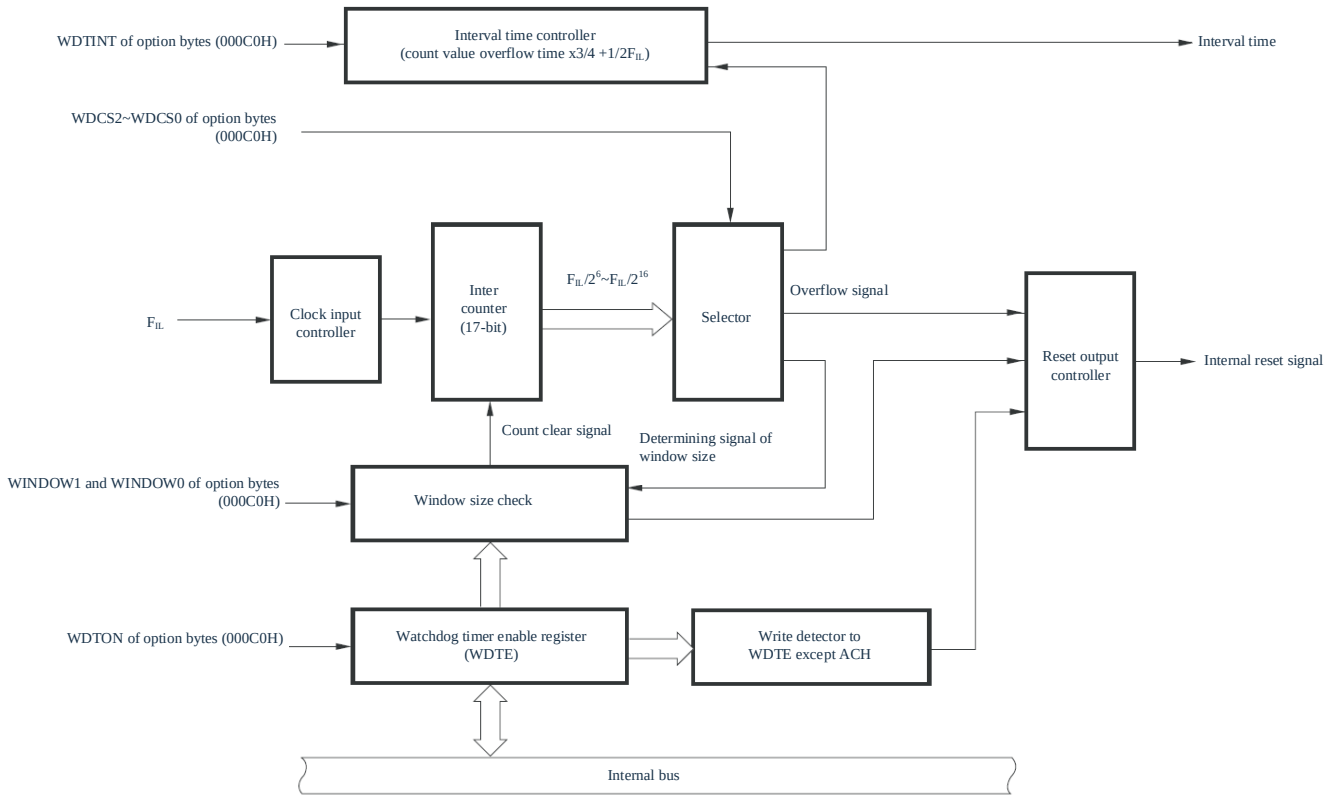
The operation of the counter is controlled by the option byte as well as the setting of the overflow time, the window opening period and the interval interrupt.

Table 8-2 Setting of option bytes and watchdog timer

| Setting of Watchdog Timer | Option Byte (000C0H) |
|---|---|
| Watchdog timer interval interrupt | bit7 (WDTINT) |
| Window open period | bit6 and bit5 (WINDOW1, WINDOW0) |
| Controlling counter operation of watchdog timer | bit4 (WDTON) |
| Overflow time of watchdog timer | bit3~1 (WDCS2~WDCS0) |
| Controlling counter operation of watchdog timer (in sleep mode) | bit0 (WDSTBYON) |

Note: For option byte, see Chapter 31 Option Byte.

Figure 8-1 Block diagram of watchdog timer




Note: $F_{IL}$: Low-speed on-chip oscillator clock frequency

## 8.3 Register Mapping

(WDTE base address = 0x4002_1001) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| WDTE | 0x000 | R/W | Watchdog timer enable register | 0x1A/0x9A |

(LOCKCTL base address = 0x4002_0405) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| LOCKCTL | 0x000 | R/W | Control register | 0x1 |

(PRCR base address = 0x4002_0406) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| PRCR | 0x000 | R/W | Protection register | 0x0 |

### 8.3.1 Watchdog Timer Enable Register (WDTE)

Writing “ACH” to the WDTE register clears the watchdog timer counter and starts counting again. The WDTE register is set by an 8-bit memory manipulation instruction. Reset signal generation sets this register to “9AH” or “1AH” Note.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | WDTE | Write 0xAC to clear the watchdog timer counter and restart counting. | 0x1A/0x9A |

Note: The WDTE register reset value differs depending on the WDTON bit setting value of the option byte (000C0H). To operate watchdog timer, set the WDTON bit to 1.

| WDTON bit setting value | WDTE register reset value |
|---|---|
| 0 (watchdog timer count operation disabled) | 1AH |
| 1 (watchdog timer count operation enabled) | 9AH |

Note 1: If a value other than “ACH” is written to the WDTE register, an internal reset signal is generated.

Note 2: The value read from the WDTE register is 9AH/1AH (this differs from the written value (ACH)).

## 8.3.2 LOCKUP Control Register (LOCKCTL)

The LOCKCTL register is a configuration register for controlling the Cortex-M0+ LockUp function to operate the watchdog timer, and PRCR is its write-protect register.

The LOCKCTL register is set by an 8-bit memory manipulation instruction.

After generating a reset signal, the value of the LOCKCTL register changes to “01H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:1 | - | Reserved | - |
| 0 | lockup_rst | Configuration of LOCKUP function<br>0: LOCKUP does not cause a WDT reset<br>1: LOCKUP causes the WDT to reset | 1 |

## 8.3.3 Protection Register (PRCR)

The LOCKCTL register is a configuration register for controlling the Cortex-M0+ LockUp function to operate the watchdog timer, and PRCR is its write-protect register.

The PRCR register is set by an 8-bit memory manipulation instruction.

After generating a reset signal, the value of the PRCR register changes to “00H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:1 | PRTKEY | Write protection of PRCR<br>78H: PRCR is writable<br>Other: PRCR is not writable | 0x0 |
| 0 | PRCR | Write protection of LOCKUP control register<br>0: LOCKCTL is not writable<br>1: LOCKCTL is writable | 0 |

## 8.3.4 Watchdog Configuration Register WDTCFGx (x=0~3)

The WDTCFGx configuration register is a register that forces the watchdog timer to operate or not.

The WDTCFGx register is set by an 8-bit register manipulation instruction.

After a reset signal is generated, the value of the WDTCFGx register changes to “00H”.

WDTCFGx configuration register

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | WDTCFGx | The Watchdog Configuration Register can be set to a specific value to force the watchdog timer to run. See the table below. | 0x0 |

| WDTCFG0 | WDTCFG1 | WDTCFG2 | WDTCFG3 | Configuration of the watchdog timer function |
|---|---|---|---|---|
| 0x1A | 0x2B | 0x3C | 0x4D | The operation of the watchdog timer after reset is determined by the option byte Note1 |
| Other | | | | Forces the watchdog timer to run after reset |

Note 1: For detailed configuration refer to section 31.4 User Option Byte

# 8.4 Operation of Watchdog Timer

## 8.4.1 Operational Control of Watchdog Timer

1. When using the watchdog timer, set the following items by option byte (000C0H):

• The bit 4 (WDTON) of the option byte (000C0H) must be set to 1 to enable the watchdog timer count to operate (the counter starts operating after the reset is released) (refer to Chapter 31 Option Byte for details).

| WDTON | Counter of watchdog timer |
|---|---|
| 0 | Disables counting operation (stop counting after reset released) |
| 1 | Enables counting operation (start counting after release reset) |

• The overflow time must be set by bit3~1 (WDCS2~WDCS0) of the option byte (000C0H) (refer to 8.4.2 and Chapter 31 Option Byte for details).

• The window opening period must be set by bit6 and bit5 (WINDOW1, WINDOW0) of the option byte (000C0H) (refer to 8.4.3 and Chapter 31 Option Byte for details).

2. After the reset is released, the watchdog timer starts counting.

3. After starting counting and before the overflow time set by the option byte, writing “ACH” to the watchdog timer enable register (WDTE) clears the watchdog timer and starts counting again.

4. Thereafter, writes to WDTE registers after the second time after the reset must be performed while the window is open. If you write the WDTE register while the window is closed, an internal reset signal is generated.

5. If you do not write “ACH” to the WDTE register and exceed the overflow time, an internal reset signal is generated. An internal reset signal is generated if:

• If data other than “ACH” is written to the WDTE register

Note 1: When data is written to the watchdog timer enable register (WDTE) for the first time after reset release, the watchdog timer is cleared in any timing regardless of the window open time, as long as the register is written before the overflow time, and the watchdog timer starts counting again.

Note 2: After “ACH” is written to the WDTE register, an error of up to 2 $F_{IL}$ clocks may occur before the watchdog timer is cleared.

Note 3: The watchdog timer can be cleared immediately before the count value overflows.

Note 4: As shown below, the watchdog timer operates in sleep or deep sleep mode depending on the set value of bit0 (WDSTBYON) of the option byte (000C0H).

| | WDSTBYON=0 | WDSTBYON=1 |
|---|---|---|
| Sleep mode | Stop operation of watchdog timer. | Continue operation of watchdog timer. |
| Deep sleep mode | | |

When the WDSTBYON bit is “0”, restart the watchdog timer count after the sleep or deep sleep mode released. At this point, the counter is cleared to “0” and the count begins.

If the period between the deep sleep mode release and the watchdog timer overflow is short, an overflow occurs during the oscillation stabilization time, causing a reset.

## 8.4.2 Setting Overflow Time of Watchdog Timer

Set the overflow time of the watchdog timer by using bits 3 to 1 (WDCS2 to WDCS0) of the option byte (000C0H).

If an overflow occurs, an internal reset signal is generated. The present count is cleared and the watchdog timer starts counting again by writing "ACH" to the watchdog timer enable register (WDTE) during the window open period before the overflow time. The following overflow times can be set.

Table 8-3 Setting of overflow time of watchdog timer

| WDCS2 | WDCS1 | WDCS0 | Overflow time of watchdog timer (When $F_{IL}$=15kHz) |
|---|---|---|---|
| 0 | 0 | 0 | $2^6/F_{IL}$ (4.3ms) |
| 0 | 0 | 1 | $2^7/F_{IL}$ (8.5ms) |
| 0 | 1 | 0 | $2^8/F_{IL}$ (17.0ms) |
| 0 | 1 | 1 | $2^9/F_{IL}$ (34.0ms) |
| 1 | 0 | 0 | $2^{11}/F_{IL}$ (135.9ms) |
| 1 | 0 | 1 | $2^{13}/F_{IL}$ (543.5ms) |
| 1 | 1 | 0 | $2^{14}/F_{IL}$ (1086.9ms) |
| 1 | 1 | 1 | $2^{16}/F_{IL}$ (4347.8ms) |

Remark: $F_{IL}$: Low-speed on-chip oscillator clock frequency

## 8.4.3 Setting Window Open Period of Watchdog Timer

Set the window open period of the watchdog timer by using bits 6 and 5 (WINDOW1, WINDOW0) of the option byte (000C0H). The outline of the window is as follows:

- If "ACH" is written to the watchdog timer enable register (WDTE) during the window open period, the watchdog timer is cleared and starts counting again.
- Even if "ACH" is written to the WDTE register during the window close period, an abnormality is detected and an internal reset signal is generated.

Caution: When data is written to the WDTE register for the first time after reset release, the watchdog timer is cleared in any timing regardless of the window open time, as long as the register is written before the overflow time, and the watchdog timer starts counting again.

The window open period can be set is as follows.

Table 8-4 Setting window open period of watchdog timer

| WINDOW1 | WINDOW0 | Window open period of watchdog timer |
|---|---|---|
| 0 | - | Settings are disabled. |
| 1 | 0 | 75% |
| 1 | 1 | 100% |

Caution: When bit 0 (WDSTBYON) of the option byte (000C0H) = 0, the window open period is 100% regardless of the values of the WINDOW1 and WINDOW0 bits.

If the overflow time is set to $2^9/F_{IL}$, the window close time and open time are as follows.

| | Setting of window open period | |
|---|---|---|
| | 75% | 100% |
| Window close time | 0~12.8ms | None |
| Window open time | 12.8~25.6ms | 0~25.6ms |

<When window open period is 75%>

(1) Overflow time:

$2^9/F_{IL}$(MAX.)=$2^9$/20kHz(MAX.)=25.6ms

(2) Window close time:

0~$2^9/F_{IL}$(MIN.)×(1–0.75)=0~$2^9$/10kHz×0.25=0~12.8ms

(3) Window open time:

$2^9/F_{IL}$(MIN.)×(1–0.75)~$2^9/F_{IL}$(MAX.)=12.8~25.6ms

## 8.4.4 Setting Watchdog Timer Interval Interrupt

Depending on the setting of bit 7 (WDTINT) of an option byte (000C0H), an interval interrupt (INTWDTI) can be generated when 75%+1/2$F_{IL}$ of the overflow time is reached.

Table 8-5 Setting of watchdog timer interval interrupt

| WDTINT | Use of watchdog timer interval interrupt |
|---|---|
| 0 | Interval interrupt is not used. |
| 1 | Interval interrupt is generated when 75%+1/2$F_{IL}$ of the overflow time is reached. |

Caution: When operating with the X1 oscillation clock after releasing the deep sleep mode, the CPU starts operating after the oscillation stabilization time has elapsed. Therefore, if the period between the deep sleep mode release and the watchdog timer overflow is short, an overflow occurs during the oscillation stabilization time, causing a reset.

Remark: The watchdog timer continues counting even after a watchdog interval interrupt is generated (until "ACH" is written to the watchdog timer enable register (WDTE)). If "ACH" is not written to the WDTE register before the overflow time, an internal reset signal is generated.

## 8.4.5 Operation of Watchdog Timer During LOCKUP

When lockup_rst bit of the lockup control register lockcTL is set to 1, once the kernel enters the LOCKUP state, the low-speed internal oscillator begins to oscillate, the watchdog timer automatically starts operating, and the overflow time control bit (WDCS2~WDCS0) is set to 3'b010, that is, the overflow time is set to 17ms.

# Chapter 9 Division and Square Root Operation Unit (DIVSQRT)

## 9.1 Overview

The chip contains a 32-bit/32-bit hardware divider and a 32-bit hardware square root extractor.

## 9.2 Features

- Support signed/unsigned division and square root operations.
- Both quotient and remainder are 32 bits wide.
- Divider has a clear flag indicator bit.
- Operations are completed within 22 APB clocks.
- Start the operation by writing to the ALUB register.

## 9.3 Function Description

The operation unit can choose between division mode and square root mode through the register DIVSQRT->CON[4]. In division mode, the quotient is saved in DIVSQRT->RES0 and the remainder is saved in DIVSQRT->RES1. The DIVSQRT->CON[2] register can be used to detect whether the divisor is zero, which is a read-only bit. In square root mode, DIVSQRT->RES0 saves the square root result, and DIVSQRT->RES1 is unused.

Note that in square root mode, if the highest bit of the number being square rooted is 1, it is treated as a signed number. The absolute value is taken first before performing the square root operation:

$$RSE0 = \sqrt[2]{absval(ALUB)}$$

The register DIVSQRT->CON[3] can be used to detect whether the operation has been completed. This is a read-only bit, with a value of 0 indicating that the operation is still ongoing and a value of 1 indicating that the operation has been completed. When the division unit is idle, this bit is also set to 1.

The register DIVSQRT->CON[1] can be used to select between signed and unsigned division modes.

Please note that the clock enable bit for the arithmetic unit is set in the peripheral enable register PER12.

Caution: Do not write to the ALUA or ALUB registers or read from the RES0 or RES1 registers during calculation, otherwise the results are unpredictable.

Register definitions in different modes:

| Operation unit mode | ALUA | ALUB | RES0 | RES1 |
|---|---|---|---|---|
| Division mode | Dividend | Divisor | Quotient | Remainder |
| Square root mode | - | Radicand | Square root result (low 16 bits valid) | - |

## 9.4 Register Mapping

(DIVSQRT base address = 0x4006_4480) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CON | 0x000 | R/W | Operation unit control register | 0x8 |
| ALUA | 0x004 | R/W | Operation unit data A register | 0x0 |
| ALUB | 0x008 | R/W | Operation unit data B register | 0x0 |
| RES0 | 0x00C | RO | Operation unit result 0 register | 0x0 |
| RES1 | 0x010 | RO | Operation unit result 1 register | 0x0 |

## 9.5 Register Description

### 9.5.1 DIVSQRT Control Register (CON)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:5 | - | Reserved | - |
| 4 | MODE | Operation mode select bit<br>0: Division mode<br>1: Square root mode | 0 |
| 3 | READY | Operation completion indicator bit<br>0: Operation is ongoing<br>1: Operation is completed or is in idle state | 1 |
| 2 | DIVBY0 | Division mode clear indicator bit (this bit is updated automatically when the divisor is written)<br>0: Divisor is not 0<br>1: Divisor is 0 | 1 |
| 1 | SIGN | Division mode symbol select bit<br>0: Unsigned mode<br>1: Signed mode | 0 |
| 0 | - | Reserved | 0 |

### 9.5.2 DIVSQRT Data A Register (ALUA)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | ALUA | 32-bit data A | 0x0 |

### 9.5.3 DIVSQRT Data B Register (ALUB)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | ALUB | 32-bit data B | 0x0 |

### 9.5.4 DIVSQRT Result 0 Register (RES0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | RES0 | 32-bit result 0 | 0x0 |

### 9.5.5 DIVSQRT Result 1 Register (RES1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | RES1 | 32-bit result 1 | 0x0 |

# Chapter 10 Division Operation Unit (DIV)

## 10.1Overview

The chip contains a 32-bit/32-bit hardware divider.

## 10.2Features

- Support signed/unsigned division operations.
- Both quotient and remainder are 32 bits wide.
- Divider has a clear flag indicator bit.
- Operations are completed within 22 APB clocks.
- Start the operation by writing to the ALUB register.

## 10.3Function Description

The quotient is saved in DIV->RES0 and the remainder is saved in DIV->RES1. The DIV->CON[2] register can be used to detect whether the divisor is zero, which is a read-only bit.

The register DIV->CON[3] can be used to detect whether the operation has been completed. This is a read-only bit, with a value of 0 indicating that the operation is still ongoing and a value of 1 indicating that the operation has been completed. When the division unit is idle, this bit is also set to 1.

The register DIV->CON[1] can be used to select between signed and unsigned division modes.

Please note that the clock enable bit for the arithmetic unit is set in the peripheral enable register PER12.

Caution: Do not write to the ALUA or ALUB registers or read from the RES0 or RES1 registers during calculation, otherwise the results are unpredictable.

Register definitions in different modes:

| Operation unit mode | ALUA | ALUB | RES0 | RES1 |
|---|---|---|---|---|
| Division mode | Dividend | Divisor | Quotient | Remainder |

## 10.4Register Mapping

(DIV base address = 0x4006_44C0) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CON | 0x000 | R/W | Operation unit control register | 0x8 |
| ALUA | 0x004 | R/W | Operation unit data A register | 0x0 |
| ALUB | 0x008 | R/W | Operation unit data B register | 0x0 |
| RES0 | 0x00C | RO | Operation unit result 0 register | 0x0 |
| RES1 | 0x010 | RO | Operation unit result 1 register | 0x0 |

# 10.5Register Description

## 10.5.1 Division Control Register (CON)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3 | READY | Operation completion indicator bit<br>0: Operation is ongoing<br>1: Operation is completed or is in idle state | 1 |
| 2 | DIVBY0 | Division mode clear indicator bit (this bit is updated automatically when the divisor is written)<br>0: Divisor is not 0<br>1: Divisor is 0 | 0 |
| 1 | SIGN | Division mode symbol select bit<br>0: Unsigned mode<br>1: Signed mode | 0 |
| 0 | - | Reserved | 0 |

## 10.5.2 Division Data A Register (ALUA)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | ALUA | 32-bit data A | 0x0 |

## 10.5.3 Division Data B Register (ALUB)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | ALUB | 32-bit data B | 0x0 |

## 10.5.4 Division Result 0 Register (RES0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | RES0 | 32-bit result 0 | 0x0 |

## 10.5.5 Division Result 1 Register (RES1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | RES1 | 32-bit result 1 | 0x0 |

# Chapter 11 Timer (TIMER0/1)

## 11.1Overview

It contains two programmable 32-bit/16-bit counters, TIMER0/TIMER1, providing users with convenient timer counting functions.

## 11.2Features

- Configurable 32-bit/16-bit count down counter.
- Each timer has an independent prescaler.
- Support one-shot trigger, periodic count, and continuous count modes.
- Support waking up the chip from sleep mode.

## 11.3Function Description

### 11.3.1 One-Shot Trigger Mode

In one-shot trigger mode, when the timer is enabled, the counter loads the initial value from the load register, counts down, and stops working when the counter reaches 0. An interrupt is generated at this time. To start the one-shot trigger mode again, the TMROS bit needs to be cleared and then set again.
(When starting the one-shot trigger mode again, it should be noted that the time during which the TMROS bit is cleared and remains 0 should be greater than one timer counting cycle.)

### 11.3.2 Periodic Count Mode

In periodic count mode, when the timer is enabled, the counter loads the initial value from the load register, counts down, and when the counter reaches 0, it loads the initial value again from the load register and continues counting. An interrupt is generated at this time.

### 11.3.3 Continuous Count Mode

In continuous count mode, when the timer is enabled, the counter loads the initial value from the load register, counts down, and when the counter reaches 0, it loads the maximum value as the initial value and continues counting. An interrupt is generated at this time.

### 11.3.4 Delayed Load Function

When data is written to the delayed load register, the data is written into the load register on the next TIMER_CLK rising edge. If the counter has already started counting, it will wait until the current period count reaches 0 before loading the initial value from the load register.

# 11.4Register Mapping

(Timer0 base address= 0x4006_1000, Timer1 base address= 0x4006_1100)

RO: Read only, WO: Write Only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CON | 0x000 | R/W | Timer control register | 0x0 |
| LOAD | 0x004 | R/W | Timer load register | 0x0 |
| VAL | 0x008 | RO | Timer current value register | 0xFFFFFFFF |
| RIS | 0x00C | RO | Timer interrupt source status register | 0x0 |
| MIS | 0x010 | RO | Timer enabled interrupt status register | 0x0 |
| ICLR | 0x014 | WO | Timer interrupt clear register | - |
| BGLOAD | 0x018 | R/W | Timer delayed load register | 0x0 |

# 11.5Register Description

## 11.5.1 Timer Control Register (CON0/1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7 | TMREN | Timer enable bit<br>0: Disable<br>1: Enable | 0 |
| 6 | TMRMS | Timer mode select bit<br>0: Continuous counting mode<br>1: Periodic counting mode | 0 |
| 5 | TMRIE | Timer interrupt enable bit<br>0: Disable interrupts<br>1: Enable interrupts | 0 |
| 4 | - | Reserved | - |
| 3:2 | TMRPRE | Timer prescaler<br>00: Divided by 1<br>01: Divided by 16<br>10: Divided by 256<br>11: Reserved | 0x0 |
| 1 | TMRSZ | Timer count bit selection<br>0: 16-bit counter<br>1: 32-bit counter | 0 |
| 0 | TMROS | One-shot trigger mode select bit<br>0: The mode is determined by the TMRMS bit<br>1: Single trigger mode<br>(Re-triggered in one-shot mode, and the initial value of the counting is determined by the TMRMS bit) | 0 |

### 11.5.2 Timer Load Register (LOAD0/1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | LOAD | Timer load register | 0x0 |

### 11.5.3 Timer Current Value Register (VAL0/1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | VAL | Timer current count value | 0xFFFFFFFF |

### 11.5.4 Timer Interrupt Source Status Register (RIS0/1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:1 | - | Reserved | - |
| 0 | RIS | Timer interrupt source status<br>1: An interrupt is generated<br>0: No interrupt is generated | 0 |

### 11.5.5 Timer Enabled Interrupt Status Register (MIS0/1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:1 | - | Reserved | - |
| 0 | MIS | Timer enabled interrupt status bit<br>1: An interrupt is enabled and an interrupt is generated<br>0: No interrupt is generated | 0 |

### 11.5.6 Timer Interrupt Clear Register (ICLR0/1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | ICLR | Write any number, clear timer interrupt | - |

### 11.5.7 Timer Delayed Load Register (BGLOAD0/1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | BGLOAD | Timer delayed load register (reads the value of the most recent write to LOAD or BGLOAD) | 0x0 |

# Chapter 12 Capture/Compare/PWM Module (CCP0/1)

## 12.1Overview

It includes two sets of CCP modules, CCP0/CCP1, with each set corresponding to two channels, A and B. CCP0 corresponds to CCP0A/CCP0B, and CCP1 corresponds to CCP1A/CCP1B.

## 12.2Features

- There are two sets of CCP modules, each supporting two PWM outputs.
- Each set of CCP can be configured with an independent period.
- CCP0 has a 16-bit counter that can generate compare/overflow interrupts.
- CCP1 has a 32-bit counter that can generate compare/overflow interrupts.
- CCPn has an independent capture function, and can select input signals on either the A or B channel pins.
- CCP1 has a 4-channel capture function, capable of simultaneously capturing CCP0A, CCP0B, CCP1A, and CCP1B input signals.
- In Capture Mode 1, it supports the capture operation to reload the CCP0 counter.
- The internal channel CAP3 supports the analog comparator output capture function.
- The internal channels CAP0-CAP3 support software capture functionality.

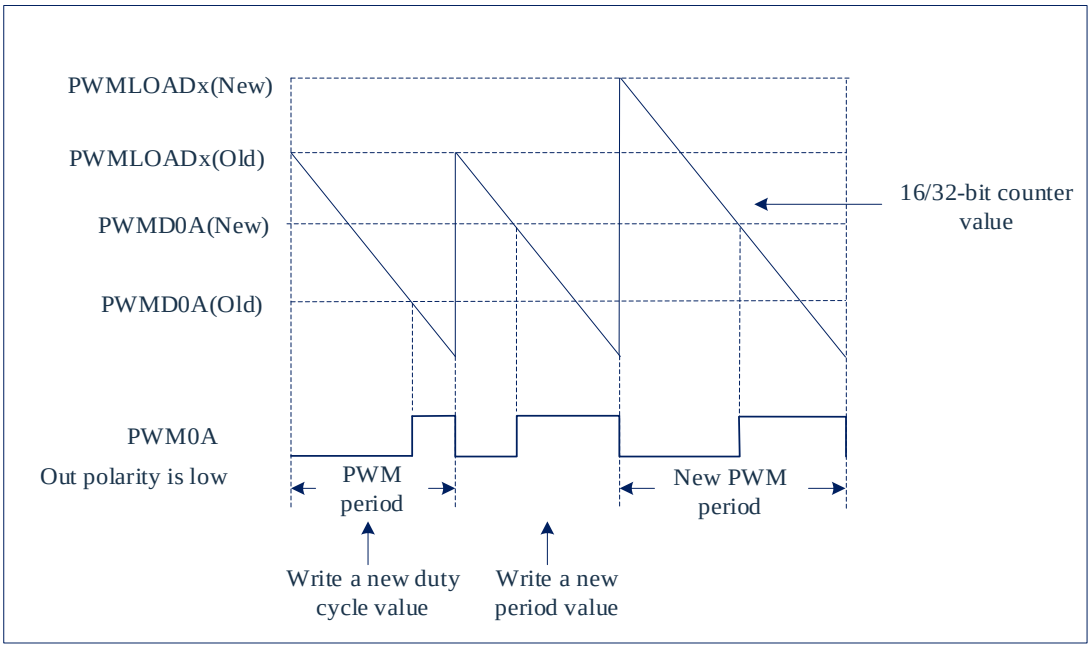
# 12.3Function Description

## 12.3.1 Pulse Width Modulation (PWM)

Each CCP module can output two PWM signals, A and B: PWMxA and PWMxB. These two signals share a common period, and the output duty cycle can be independently set via CCPxDA and CCPxDB. The output polarity of PWMxA/PWMxB can be configured using the PWMxAOP/PWMxBOP bits, which correspond to the CCPxA/CCPxB channel outputs.

When CCP0RUN is set to 1, the 16-bit counter loads the initial count value for CCP0 and counts down. When CCP1RUN is set to 1, the 32-bit counter loads the initial count value for CCP1 and counts down. When the counter value equals the CCPxDA/B value or when a counter overflow occurs, the PWMxA/PWMxB output level changes.

Figure 12-1: PWM timing diagram




The calculation methods for period and duty cycle are as follows:

Period=CCPxLOAD×CCP clock period.

PWMxA duty cycle=CCPxDA/CCPxLOAD (supports 0% to 100%).

PWMxB duty cycle =CCPxDB/CCPxLOAD (supports 0% to 100%).

When CCPxLOAD=0, the duty cycle for PWMxA and PWMxB is 0%.

When CCPxDA/CCPxDA≥CCPxLOAD, the duty cycle is 100%.

### 12.3.2 Square Wave Output Mode

The square wave output mode is a type of pulse-width modulation (PWM) mode, where the period is freely adjustable, and the duty cycle is fixed at 50%.

Each CCP module can be configured to output a square wave on either the A or B channel. To set the square wave output mode, the PWM mode must be configured, and the corresponding CCPxCON.ZAEN or CCPxCON.ZBEN bit must be set. The period is determined by the initial count value of the CCPx, and the duty cycle is defaulted to half of the CCPxLOAD value (with the least significant bit ignored).

### 12.3.3 Capture Mode 0

This is the external capture mode.

CCP0 can be configured for either the A or B channel as the external capture signal pin. After CCP0RUN is set, the 16-bit counter starts counting down from 0xFFFF or CCP0LOAD[15:0]. When a capture condition is triggered, the counter stops counting, and CCP0A or CCP0B returns the current counter value. To perform the next capture, CCP0RUN must be cleared and then set again.

The capture time is calculated as follows:

CCP0LOAD[15:0]=0 or 1, the capture time= (0xFFFF -CCP0DA/B)× CCPx clock cycle

CCP0LOAD[15:0]=other, the capture time= (CCP0LOAD[15:0]-CCPxDA/B)× CCPx clock cycle

CCP1 can be configured for either the A or B channel as the external capture signal pin. After CCP1RUN is set, the 32-bit counter starts counting down from 0xFFFFFFFF or CCP1LOAD[31:0]. When a capture condition is triggered, the counter stops counting, and CCP1A or CCP1B returns the current counter value. To perform the next capture, CCP1RUN must be cleared and then set again.

The capture time is calculated as follows:

CCP1LOAD[31:0]=0 or 1, the capture time= (0xFFFFFFFF -CCP1DA/B)× CCPx clock cycle

CCP1LOAD[31:0]=other, the capture time= (CCP1LOAD[31:0]-CCP1DA/B)× CCPx clock cycle

Note: If the reload enable is set, it is recommended that the reload value should not be set to 1.

### 12.3.4 Capture Mode 1

CCP1 includes four internal channels: CAP0, CAP1, CAP2, and CAP3. Any of these channels can be selected to correspond to any external channel from ECAP00-03 or ECAP10-13 as the capture channel. Alternatively, CCP0A/CCP0B/CCP1A/CCP1B can also be selected as the capture channels.

ECAP00-03 correspond to the positive inputs of Analog Comparator 0: C0P0 to C0P3.

ECAP10-13 correspond to the positive inputs of Analog Comparator 1: C1P0 to C1P3.

When using ECAP for external capture, the corresponding ports need to be configured as GPIO functions.

When using CCP0A/CCP0B/CCP1A/CCP1B for capture, the respective ports should be configured as CCP ports.

The correspondence between CAPn and external channels is as follows:

| Internal channel | External channel |
|---|---|
| CAP0 | CAP0CHS=n: Select ECAP0n (n=0-3)@ECAPS=0<br>CAP0CHS=n: Select ECAP1n (n=0-3)@ECAPS=1<br>CAP0CHS=F: Select CAP0A<br>CAP0CHS=other value: Reserved |
| CAP1 | CAP1CHS=n: Select ECAP0n (n=0-3)@ECAPS=0<br>CAP1CHS=n: Select ECAP1n (n=0-3)@ECAPS=1<br>CAP1CHS=F: Select CAP0B<br>CAP1CHS=other value: Reserved |
| CAP2 | CAP2CHS=n: Select ECAP0n (n=0-3)@ECAPS=0<br>CAP2CHS=n: Select ECAP1n (n=0-3)@ECAPS=1<br>CAP2CHS=F: Select CAP1A<br>CAP2CHS=other value: Reserved |
| CAP3 | CAP3CHS=n: Select ECAP0n (n=0-3)@ECAPS=0<br>CAP3CHS=n: Select ECAP1n (n=0-3)@ECAPS=1<br>CAP3CHS=8: Select after ACMP0 filter selection<br>CAP3CHS=9: Select after ACMP1 filter selection<br>CAP3CHS=F: Select CAP1B<br>CAP3CHS=other value: Reserved |

In Capture Mode 1, the PWM output modes of CCP0 and CCP1 are disabled when operating in external capture mode 0.

In this mode, CCP1 needs to operate in count mode, and the capture operation will load the CCP1 count value into the corresponding register.

Additionally, CCP0 can be selected to operate in count mode, and the CAP0-CAP3 channels can be configured with a capture trigger load function. That is, when a capture operation is triggered on the selected channel, the initial value of the CCP0 counter will be reloaded. Multiple channels can be set with this feature, and a software trigger capture will not reload the CCP0 initial value.

In Capture Mode 1, the compare/overflow interrupt functions of CCP0 and CCP1 can still be used.

This capture method has two types: one triggered by an external signal and the other triggered by software.

1) External Signal Triggered Capture:

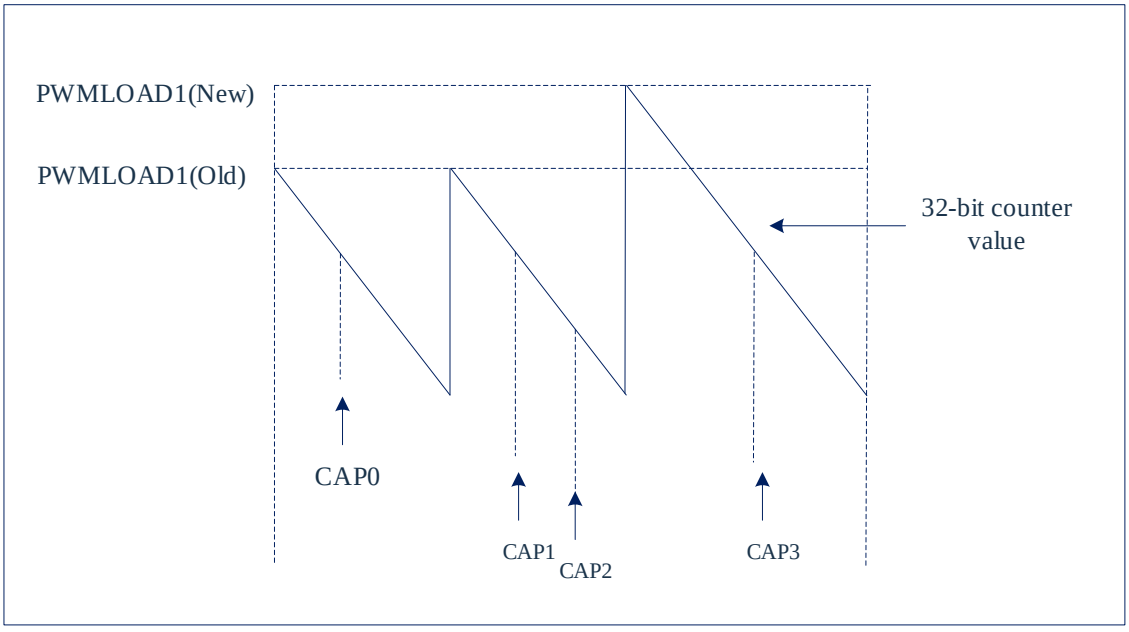
CAP0-CAP3 can be configured to capture on the rising edge, falling edge, or both edges. When a signal is generated, the value of the CCP1 counter will be captured into the corresponding register, and an interrupt flag will be set. The correspondence between the four channels and the capture registers is as follows:

CAP0/CAP1/CAP2/CAP3 corresponds to CAP0DATA/CAP1DATA/CAP2DATA/CAP3DATA

2) Software Triggered Capture:

Writing 0x55AA to the following registers: CAPCON2[31:16], CAPCON2[15:0], CAPCON3[31:16], CAPCON3[15:0] will trigger capture operations on channels CAP0 to CAP3, capturing the value of the CCP1 counter into the corresponding registers. A software-triggered capture does not generate an interrupt flag.

Figure 12-2: CAP0-CAP3 channel capture operation



### 12.3.5 Capture Mode 2

This capture mode is external capture mode, primarily used for capturing external PWM waveform information.

In capture mode 2, the channels CAP2 and CAP3 are mapped to CAP1, meaning that CAP1-3 share the same capture channel, and CAP0 is disabled.

CCP0 can be freely configured and is not affected.

CCP1 operates in count mode. After CCP1RUN is set, the 32-bit counter begins counting down from its initial value. If the counter overflows without triggering a capture operation, the counter reloads its initial value and continues counting down.

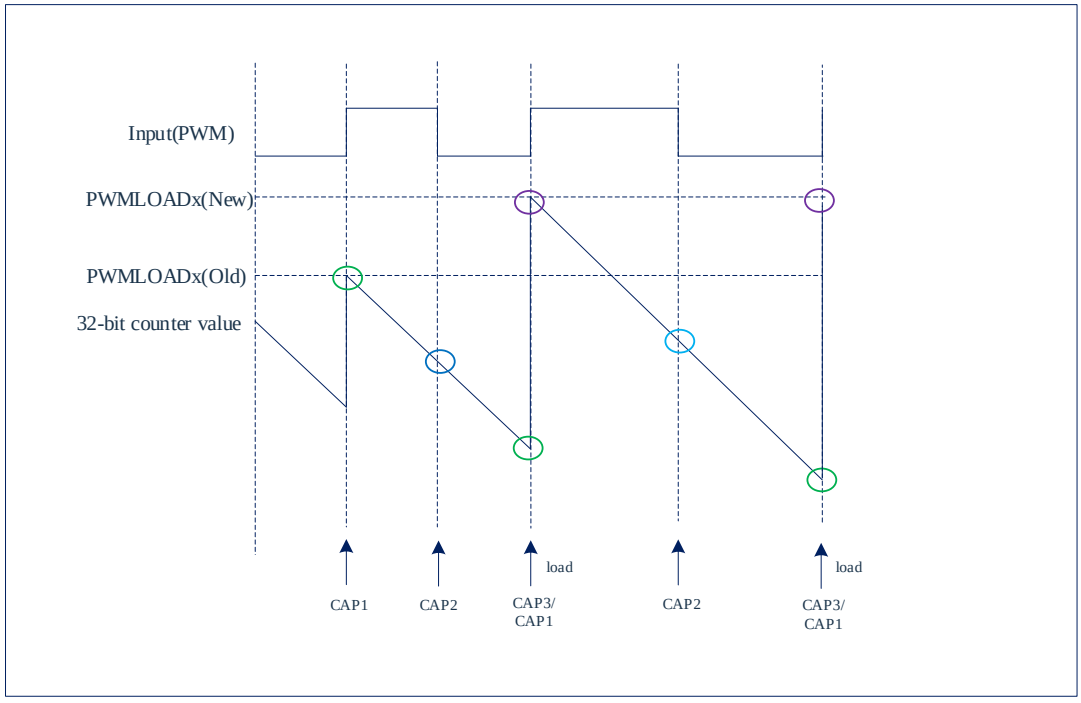
The operation to start capture mode 2 is as follows:

(1) After a capture occurs on CAP1, the CCP1 counter reloads its initial value and loads this value into CAP1DATA, allowing CAP2 to trigger a capture.

(2) After a capture occurs on CAP2, the captured counter value is loaded into CAP2DATA, allowing CAP3 to trigger a capture.

(3) After a capture occurs on CAP3, the captured counter value is loaded into CAP3DATA, completing the entire capture process. At this point, the values of CAP1DATA-CAP3DATA are loaded into CAP0DATA[31:0], and the values of CAP1DATA-CAP2DATA are loaded into CAPDUTY[31:0]. Then CAP1 triggers a capture action, returning to step (1).

Figure 12-3: Capture Mode 2 capture operation




Note 1: If the time taken for the capture on CAP3 exceeds one cycle of the CCP1 counter, an overflow will occur, and the calculated values of CAP0DATA and CAPDUTY will be inaccurate. It is recommended to set the counter period much larger than the PWM period to be captured.

Note 2: The interval time between the capture on CAP1 and CAP2, as well as between CAP2 and CAP3, must be greater than 8 CCP1 counter values.

Note 3: Capture mode 2 supports software capture actions for CAP1-3.

Note 4: CAP1 to CAP3 share the same capture channel, and the capture operation will simultaneously generate corresponding capture flags on CAP1-CAP3.

### 12.3.6 PWM Configuration Process

- Configure PWM control registers by setting the prescaler, selecting the PWM mode, and enabling PWM.
- Configure the PWM period by writing to the CCPxLOAD register.
- Configure the PWM duty cycle by writing to the CCPxDA/CCPxDB registers.
- If interrupts are required, enable the relevant interrupt bits and clear the interrupt status register.
- Set the corresponding I/O port as a PWM output.
- Set the PWM operation register to start the output.

### 12.3.7 Interrupt

In PWM mode, CCPx can generate two types of interrupts:

- Overflow interrupt: Generated when the counter decreases to 0.
- Compare interrupt: Generated when the counter value is equal to the value of CCPxDA or CCPxDB.

In capture mode 0/1/2, two types of interrupts can be generated:

- Overflow interrupt: Generated when the counter decreases to 0.
- Capture interrupt: Generated when the capture condition is triggered.

# 12.4Register Mapping

(CCP base address = 0x4006_4300) RO: read only; WO: write only; R/W: read/write.

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CCP0CON(P1B) | 0x000 | R/W | CCP0 Control Register | 0x0 |
| CCP0LOAD(P1A) | 0x004 | R/W | CCP0 Reload Register | 0x0 |
| CCP0DA(P1A) | 0x008 | R/W | CCP0 Channel A Data Register | 0x0 |
| CCP0DB(P1A) | 0x00c | R/W | CCP0 Channel B Data Register | 0x0 |
| CCP1CON(P1B) | 0x010 | R/W | CCP1 Control Register | 0x0 |
| CCP1LOAD(P1A) | 0x014 | R/W | CCP1 Reload Register | 0x0 |
| CCP1DA(P1A) | 0x018 | R/W | CCP1 Channel A Data Register | 0x0 |
| CCP1DB(P1A) | 0x01C | R/W | CCP1 Channel B Data Register | 0x0 |
| CCPIMSC(P1B) | 0x040 | R/W | CCP Interrupt Enable Register | 0x0 |
| CCPRIS | 0x044 | RO | CCP Interrupt Source Status Register | 0x0 |
| CCPMIS | 0x048 | RO | CCP Enabled Interrupt Status Register | 0x0 |
| CCPICLR | 0x04C | WO | CCP Interrupt Clear Register | 0x0 |
| CCPRUN(P1B) | 0x050 | R/W | CCP Operation Register | 0x0 |
| CCPLOCK | 0x054 | R/W | CPP0/1 Write Enable Register | 0x0 |
| CAPCON(P1B) | 0x058 | R/W | Capture Control Register | 0x0 |
| CAPCON2(P1B) | 0x05C | R/W | Capture Control Register 2 | 0x0 |
| CAPCON3(P1B) | 0x060 | R/W | Capture Control Register 3 | 0x0 |
| CAPCHS(P1B) | 0x064 | R/W | Capture Channel Select Register | 0x0 |
| CAP0DATA(P1A) | 0x068 | RO | Capture Channel 0 Data Register | 0x0 |
| CAP1DATA(P1A) | 0x06C | RO | Capture Channel 1 Data Register | 0x0 |
| CAP2DATA(P1A) | 0x070 | RO | Capture Channel 2 Data Register | 0x0 |
| CAP3DATA(P1A) | 0x074 | RO | Capture Channel 3 Data Register | 0x0 |
| CAPDUTY(P1A) | 0x078 | RO | Capture Mode 2 Capture Duty Cycle Register | 0x0 |

Note 1: The registers labeled with (P1A/P1B) are protected registers.

Note 2: When (P1A): LOCK=55H or AAH, the labeled register is allowed to be written; = other values, writing is forbidden.

Note 3: When (P1B): LOCK=55H, the labeled register is allowed to be written; = other values, writing is forbidden.

# 12.5Register Description

## 12.5.1 CCP0 Control Register (CCP0CON)

| Bit | Symbol | Decription | Reset value |
|---|---|---|---|
| 31:10 | - | Reserved | - |
| 9 | CCP0ZBEN | Square wave mode enable bit for channel B (valid for PWM mode)<br>0: Disable<br>1: Enable, and duty cycle loaded value is LOAD0/2 | 0 |
| 8 | CCP0ZAEN | Square wave mode enable bit for channel A (valid for PWM mode)<br>0: Disable<br>1: Enable, and duty cycle loaded value is LOAD0/2 | 0 |
| 7 | - | Reserved | - |
| 6 | CCP0EN | CCP0 enable bit<br>0: Disable<br>1: Enable | 0 |
| 5:4 | CCP0PS | CCP0 prescale selection<br>0x0: PCLK<br>0x1: PCLK/4<br>0x2: PCLK/16<br>0x3: PCLK/64 | 0x0 |
| 3 | CCP0MS | CCP0 mode selection<br>0: Capture mode 0 (valid when CAPEN=0)<br>1: PWM mode (valid when CAPEN=0) | 0 |
| 2 | CCP0CM0CS | CCP0 capture mode 0 capture channel selection<br>0: Channel CCP0A<br>1: Channel CCP0B | 0 |
| 1:0 | CCP0CM0ES | CCP0 capture mode 0 capture method selection<br>0x0: Start counting at CCP0RUN=1, capture on the rising edge and generate an interrupt.<br>0x1: Start counting at CCP0RUN=1, capture on the falling edge and generate an interrupt.<br>0x2: Start counting on the rising edge, capture on the falling edge and generate an interrupt.<br>0x3: Start counting on the falling edge, capture on the rising edge and generate an interrupt. | 0x0 |

### 12.5.2 CCP0 Reload Register (CCP0LOAD)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:17 | - | Reserved | - |
| 16 | RELOAD | PWM module: Reload enable bit<br>0: Counter reload value is 0xFFFF<br>1: Counter reload value is CCP0LOAD<br>Capture mode:<br>0: Counter reload value is 0xFFFF<br>1: Counter reload value is CCP0LOAD | 0 |
| 15:0 | CCP0LOAD | CCP0 counter reload value (It is recommended that the loaded value is not 0) | 0x0 |

### 12.5.3 CCP0A Data Register (CCP0DA)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:17 | - | Reserved | - |
| 16 | PWM0AOP | PWM0A output polarity selection<br>0: Normal output<br>1: Inverted output | 0 |
| 15:0 | CCP0ADATA | PWM mode: PWM0A duty cycle<br>Capture mode 0: Capture result | 0x0 |

### 12.5.4 CCP0B Data Register (CCP0DB)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:17 | - | Reserved | - |
| 16 | PWM0BOP | PWM0B output polarity selection<br>0: Normal output<br>1: Inverted output | 0 |
| 15:0 | CCP0BDATA | PWM mode: PWM0B duty cycle<br>Capture mode 0: Capture result | 0x0 |

## 12.5.5 CCP1 Control Register (CCP1CON)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:15 | - | Reserved | - |
| 14 | PWM1AOP | PWM1A output polarity selection<br>0: Normal output<br>1: Inverted output | 0 |
| 13 | PWM1BOP | PWM1B output polarity selection<br>0: Normal output<br>1: Inverted output | 0 |
| 12 | RELOAD | PWM mode: Reload selection bit<br>0: The counter reload value is 0xFFFFFFFF<br>1: The counter reload value is CCP1LOAD<br>In capture mode 0:<br>0: The counter reload value is 0xFFFFFFFF<br>1: The counter reload value is CCP1LOAD | 0 |
| 11:10 | - | Reserved | - |
| 9 | CCP1ZBEN | Square wave mode enable bit for channel B (valid for PWM mode)<br>0: Disable<br>1: Enable, duty cycle loaded to LOAD1/2 | 0 |
| 8 | CCP1ZAEN | Square wave mode enable bit for channel A (valid for PWM mode)<br>0: Disable<br>1: Enable, duty cycle loaded to LOAD1/2 | 0 |
| 7 | - | Reserved | - |
| 6 | CCP1EN | CCP1 enable bit<br>0: Disable<br>1: Enable | 0 |
| 5:4 | CCP1PS | CCP1 pre-scaler selection<br>0x0: PCLK<br>0x1: PCLK/4<br>0x2: PCLK/16<br>0x3: PCLK/64 | 0x0 |
| 3 | CCP1MS | CCP1 mode selection<br>0: Capture mode 0 (valid when CAPEN=0)<br>1: PWM mode (valid when CAPEN=0) | 0 |
| 2 | CCP1CM0CS | CCP1 capture mode 0 capture channel selection<br>0: Channel CCP1A<br>1: Channel CCP1B | 0 |
| 1:0 | CCP1CM0ES | CCP1 capture mode 0 capture method selection<br>0x0: Start counting at CCP1RUN=1, capture on the rising edge and generate an interrupt.<br>0x1: Start counting at CCP1RUN=1, capture on the falling edge and generate an interrupt.<br>0x2: Start counting on the rising edge, capture on the falling edge and generate an interrupt.<br>0x3: Start counting on the falling edge, capture on the rising edge and generate an interrupt. | 0x0 |

## 12.5.6 CCP1 Reload Register (CCP1LOAD)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | CCP1LOAD | CCP1 counter reload value (It is recommended that the loaded value is not 0) | 0x0 |

## 12.5.7 CCP1A Data Register (CCP1DA)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | CCP1ADATA | PWM mode: PWM1A duty cycle<br>Capture mode 0: Capture result | 0x0 |

## 12.5.8 CCP1B Data Register (CCP1DB)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | CCP1BDATA | PWM mode: PWM1B duty cycle<br>Capture mode 0: Capture result | 0x0 |

## 12.5.9 CCP Interrupt Enable Register (CCPIMSC)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:12 | - | Reserved | - |
| 11 | CAP3IMSC | CAP3 capture interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 10 | CAP2IMSC | CAP2 capture interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 9 | CAP1IMSC | CAP1 capture interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 8 | CAP0IMSC | CAP0 capture interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 7:6 | - | Reserved | 0x0 |
| 5 | CCPIMSC5 | CCP1 overflow interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 4 | CCPIMSC4 | CCP0 overflow interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 3:2 | - | Reserved | - |
| 1 | CCPIMSC1 | CCP1 compare/capture interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 0 | CCPIMSC0 | CCP0 compare/capture interrupt enable bit<br>0: Disable<br>1: Enable | 0 |

## 12.5.10 CCP Interrupt Source Status Register (CCPRIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:12 | - | Reserved | - |
| 11 | CAP3RIS | CAP3 capture interrupt status bit<br>1: An interrupt is generated<br>0: No interrupt is generated | 0 |
| 10 | CAP2RIS | CAP2 capture interrupt status bit<br>1: An interrupt is generated<br>0: No interrupt is generated | 0 |
| 9 | CAP1RIS | CAP1 capture interrupt status bit<br>1: An interrupt is generated<br>0: No interrupt is generated | 0 |
| 8 | CAP0RIS | CAP0 capture interrupt status bit<br>1: An interrupt is generated<br>0: No interrupt is generated | 0 |
| 7:6 | - | Reserved | 0x0 |
| 5 | CCPRIS5 | CCP1 overflow interrupt status bit<br>1: An interrupt is generated<br>0: No interrupt is generated | 0 |
| 4 | CCPMRIS4 | CCP0 overflow interrupt status bit<br>1: An interrupt is generated<br>0: No interrupt is generated | 0 |
| 3:2 | - | Reserved | - |
| 1 | CCPRIS1 | CCP1 compare/capture interrupt status bit<br>1: An interrupt is generated<br>0: No interrupt is generated | 0 |
| 0 | CCPRIS0 | CCP0 compare/capture interrupt status bit<br>1: An interrupt is generated<br>0: No interrupt is generated | 0 |

## 12.5.11 CCP Enabled Interrupt Status Register (CCPMIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:12 | - | Reserved | - |
| 11 | CAP3MIS | CAP3 enabled capture interrupt status bit<br>1: An interrupt is enabled and generated<br>0: No interrupt is generated | 0 |
| 10 | CAP2MIS | CAP2 enabled capture interrupt status bit<br>1: An interrupt is enabled and generated<br>0: No interrupt is generated | 0 |
| 9 | CAP1MIS | CAP1 enabled capture interrupt status bit<br>1: An interrupt is enabled and generated<br>0: No interrupt is generated | 0 |
| 8 | CAP0MIS | CAP0 enabled capture interrupt status bit<br>1: An interrupt is enabled and generated<br>0: No interrupt is generated | 0 |
| 7:6 | - | Reserved | - |
| 5 | CCPMIS5 | CCP1 enabled overflow interrupt status bit<br>1: An interrupt is enabled and generated<br>0: No interrupt is generated | 0 |
| 4 | CCPMIS4 | CCP0 enabled overflow interrupt status bit<br>1: An interrupt is enabled and generated<br>0: No interrupt is generated | 0 |
| 3:2 | - | Reserved | - |
| 1 | CCPMIS1 | CCP1 enabled compare/capture interrupt status bit<br>1: An interrupt is enabled and generated<br>0: No interrupt is generated | 0 |
| 0 | CCPMIS0 | CCP0 enabled compare/capture interrupt status bit<br>1: An interrupt is enabled and generated<br>0: No interrupt is generated | 0 |

## 12.5.12 CCP Interrupt Clear Register (CCPICLR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:12 | - | Reserved | - |
| 11 | CAP3ICLR | Write 1 to clear the CAP3 capture interrupt status bit | 0 |
| 10 | CAP2ICLR | Write 1 to clear the CAP2 capture interrupt status bit | 0 |
| 9 | CAP1ICLR | Write 1 to clear the CAP1 capture interrupt status bit | 0 |
| 8 | CAP0ICLR | Write 1 to clear the CAP0 capture interrupt status bit | 0 |
| 7:6 | - | Reserved | - |
| 5 | CCPICLR5 | Write 1 to clear the CCP1 overflow interrupt status bit | 0 |
| 4 | CCPICLR4 | Write 1 to clear the CCP0 overflow interrupt status bit | 0 |
| 3:2 | - | Reserved | - |
| 1 | CCPICLR1 | Write 1 to clear the CCP1 compare/capture interrupt status bit | 0 |
| 0 | CCPICLR0 | Write 1 to clear the CCP0 compare/capture interrupt status bit | 0 |

## 12.5.13 CCP Operation Register (CCPRUN)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:2 | - | Reserved | - |
| 1 | CCP1RUN | CCP1 operation control bit<br>0: Stop<br>1: Operate | 0 |
| 0 | CCP0RUN | CCP0 operation control bit<br>0: Stop<br>1: Operate | 0 |

## 12.5.14 CCP Write Enable Control Register (LOCK)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | LOCK | When LOCK=0xaa, enable the registers with protection level P1A.<br>When LOCK=0x55, enable the operation of registers with protection level P1B and P1A;<br>when LOCK=other values, disable the operation of registers with protection levels. | 0x0 |

## 12.5.15 CCP CAP Control Register (CAPCON)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:14 | - | Reserved | - |
| 13 | CAPEN2 | Capture mode 2 enable bit (only valid for CCP1)<br>0: --<br>1: Capture Mode 2 enable bit, and disable Capture Mode 1 | 0 |
| 12 | CAPEN | Capture mode 1 enable bit<br>0: CCP0/CCP1 in PWM mode or Capture Mode 0 enabled<br>1: Capture Mode 1 enabled, i.e., full channel capture mode<br>CCP0 can be set to Continuous Counting Mode<br>CCP1 can be set to Continuous Counting Mode | 0 |
| 11 | CAP3RLEN | CAP3 capture in capture mode 1 triggers the CCP0 counter load enable bit<br>0: Disable<br>1: Enable, (Valid in Capture Mode 1 and CCP0 running state)<br>When CAP3 captures a trigger signal, CCP0 will reload the initial counting value in the CCP0 register during the counter's operation. | 0 |
| 10 | CAP2RLEN | CAP2 capture in capture mode 1 triggers the CCP0 counter load enable bit<br>0: Disable<br>1: Enable, (Valid in Capture Mode 1 and CCP0 running state)<br>When CAP2 captures a trigger signal, CCP0 will reload the initial counting value in the CCP0 register during the counter's operation. | 0 |
| 9 | CAP1RLEN | CAP1 capture in capture mode 1 triggers the CCP0 counter load enable bit<br>0: Disable<br>1: Enable, (Valid in Capture Mode 1 and CCP0 running state)<br>When CAP1 captures a trigger signal, CCP0 will reload the initial counting value in the CCP0 register during the counter's operation. | 0 |
| 8 | CAP0RLEN | CAP0 capture in capture mode 1 triggers the CCP0 counter load enable bit<br>0: Disable<br>1: Enable, (Valid in Capture Mode 1 and CCP0 running state)<br>When CAP0 captures a trigger signal, CCP0 will reload the initial counting value in the CCP0 register during the counter's operation. | 0 |
| 7:6 | CAP3ES | CAP3 capture mode selection<br>0x0: Disable<br>0x1: Rising edge capture<br>0x2: Falling edge capture<br>0x3: Both edges | 0x0 |
| 5:4 | CAP2ES | CAP2 capture mode selection<br>0x0: Disable<br>0x1: Rising edge capture<br>0x2: Falling edge capture<br>0x3: Both edges | 0x0 |
| 3:2 | CAP1ES | CAP1 capture mode selection<br>0x0: Disable<br>0x1: Rising edge capture<br>0x2: Falling edge capture | 0x0 |

| | | | |
|---|---|---|---|
| | | 0x3: Both Edges | |
| 1:0 | CAP0ES | CAP0 capture mode selection<br>0x0: Disable<br>0x1: Rising edge capture<br>0x2: Falling edge capture<br>0x3: Both edges | 0x0 |

## 12.5.16 CCP CAP Channel Select Register (CAPCHS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:17 | - | Reserved | - |
| 16 | ECAPS | ECAP capture channel group selection<br>0: Select ECAP00-ECAP03<br>1: Select ECAP10-ECAP13 | 0 |
| 15:12 | CAP3CHS | CAP3 capture channel selection (x=0 or 1, determined by ECAPS)<br>0x0: ECAPx0<br>0x1: ECAPx1<br>0x2: ECAPx2<br>0x3: ECAPx3<br>0x4: Disable<br>0x5: Disable<br>0x8: ACMP0 output (non-event output)<br>0x9: ACMP1 output (non-event output)<br>0xF: CCP1B<br>Other: Reserved | 0x0 |
| 11:8 | CAP2CHS | CAP2 capture channel selection (x=0 or 1, determined by ECAPS)<br>0x0: ECAPx0<br>0x1: ECAPx1<br>0x2: ECAPx2<br>0x3: ECAPx3<br>0x4: Disable<br>0x5: Disable<br>0xF: CCP1A<br>Other: Reserved | 0x0 |
| 7:4 | CAP1CHS | CAP1 capture channel selection (x=0 or 1, determined by ECAPS)<br>0x0: ECAPx0<br>0x1: ECAPx1<br>0x2: ECAPx2<br>0x3: ECAPx3<br>0x4: Disable<br>0x5: Disable<br>0xF: CCP0B<br>Other: Reserved | 0x0 |
| 3:0 | CAP0CHS | CAP0 capture channel selection (x=0 or 1, determined by ECAPS)<br>0x0: ECAPx0<br>0x1: ECAPx1<br>0x2: ECAPx2<br>0x3: ECAPx3<br>0x4: Disable<br>0x5: Disable<br>0xF: CCP0A<br>Other: Reserved | 0x0 |

## 12.5.17 CCP1 CAP Control Register 2 (CAPCON2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Read: Written value<br>Write: 0x55aa, generates a CAP0 capture<br>Write: Other values are invalid | 0x0 |
| 15:0 | - | Read: Written value<br>Write: 0x55aa, generates a CAP1 capture<br>Write: Other values are invalid | 0x0 |

## 12.5.18 CCP1 CAP Control Register 3 (CAPCON3)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Read: Written value<br>Write: 0x55aa, generates a CAP2 capture<br>Write: Other values are invalid | 0x0 |
| 15:0 | - | Read: Written value<br>Write: 0x55aa, generates a CAP3 capture<br>Write: Other values are invalid | 0x0 |

## 12.5.19 CCP1 CAP0 Data Register (CAP0DATA)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | CAP0DATA/<br>CAPYDATA | Read: Capture mode 2: After the capture is completed, store CAPYDATA = Capture value of CAP1 - Capture value of CAP3 (cycle)<br>Other: Capture the 32bit value of the CCP1 counter for CAP0.<br>Write: Invalid | 0x0 |

## 12.5.20 CCP1 CAPn Data Register (CAPnDATA) (n=1-3)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | CAPnDATA | Read: Capture the 32bit value of the CCP1 counter for CAPn<br>Write: Invalid | 0x0 |

## 12.5.21 CCP1 Capture Mode 2 Capture Duty Cycle Register (CAPDUTY)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | CAPDUTY | Read: Duty cycle in CCP1 capture mode 2<br>Write: Invalid | 0x0 |

# Chapter 13 HALL Signal Processing Module

## 13.1Overview

The HALL signal processing module is primarily used to process the input signals from sensors, supporting the processing of 3-channel HALL sensor signals.

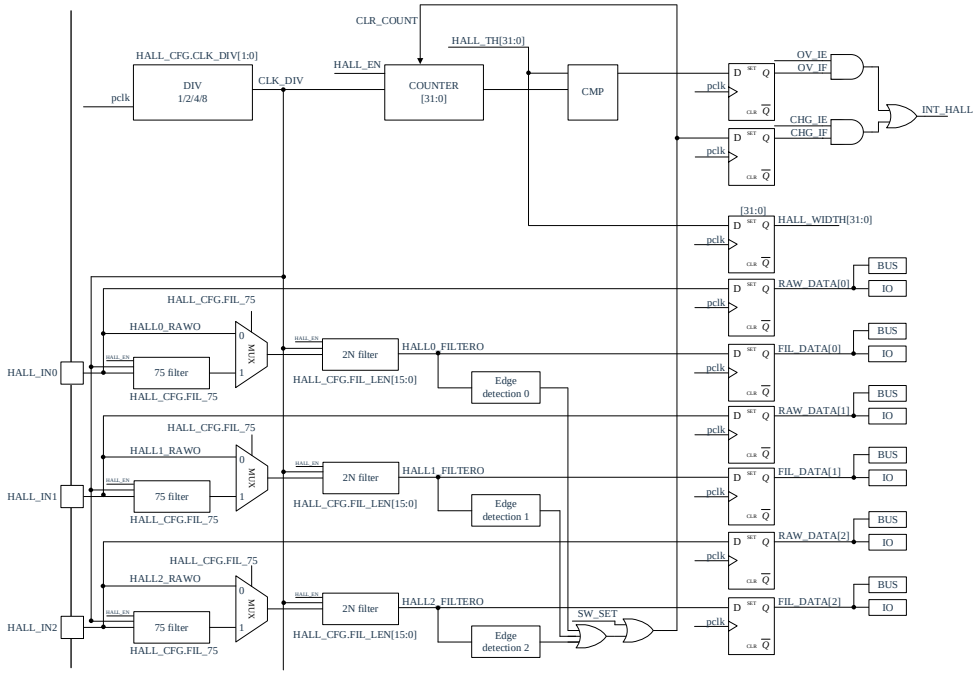
## 13.2Features

- Supports 3-channel HALL sensor input signals
- The 3 signals are independent of each other
- Each signal includes two stages of filters
- Supports HALL signal capture, recording signal change times and generating interrupts
- Supports software-triggered signal change interrupts
- 32-bit independent counter, supports overflow interrupts
- Supports real-time input signal and filtering result output.

# 13.3Function Description

## 13.3.1 Block Diagram

The data flow of the HALL module is shown in the diagram below, with PCLK being consistent with the system's main clock.




## 13.3.2 Clock Division

The operating frequency of the HALL module can be selected as a division of the system's main clock by 1/2/4/8, with both the filter and the 32-bit independent counter operating at this frequency.

## 13.3.3 Signal Input

The output signal from the HALL sensor is input through GPIO to the HALL signal processing module. The chip provides 2 GPIO options for each HALL signal input channel, and the user can select one of these as the input channel for the HALL signal.

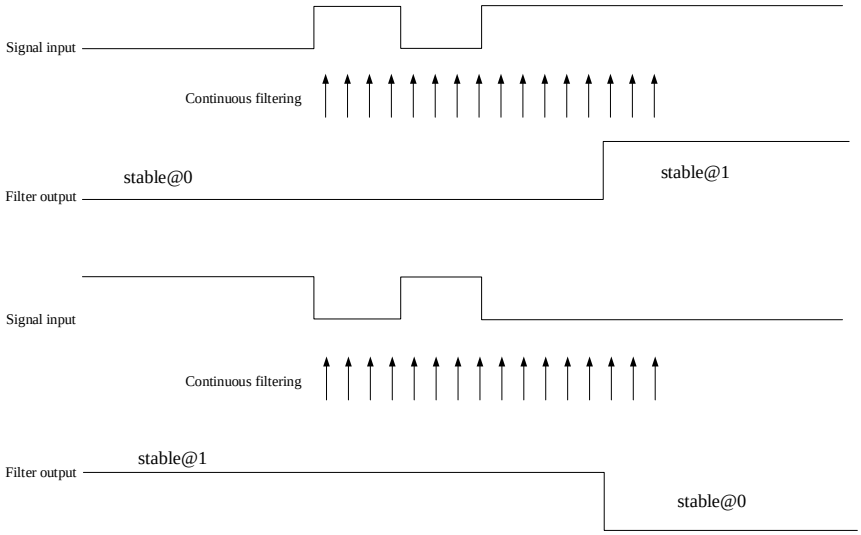
For detailed pin positions, refer to the pin function section.

### 13.3.4 Filter

The filter module contains two stages of filtering, with the main function being to remove spikes from the HALL signal.

The first stage filter (7-sample, 5-filter) works as follows:

If in a continuous set of 7 sampling points, more than 5 of them are 1, the output will be 1. If 5 or more are 0, the output will be 0. Otherwise, the output will hold the previous filtered result. The specific details are shown in the diagram below:




The second-level filter (continuous filtering) works as follows:

In a continuous sequence of N sampling points, if all are 0, the output will be 0. If all are 1, the output will be 1. Otherwise, the output will retain the previous filtering result. The filter width can be set via HALL_CFG[15:0].

### 13.3.5 Capture

An independent 32-bit counter can be used to measure the time width between two consecutive changes of the HALL signal.

The 32-bit counter (HALL_CNT) starts counting from 0. When a change in the HALL signal occurs, the current value of the counter is saved to the HALL_WIDTH register, and the current HALL signal is stored in HALL_INFO.FIT_DATA. A HALL signal change interrupt is triggered, and the counter is reset to 0.

Writing any value to the HALL_CNT can clear the counter value, causing the counter to start counting from 0 again.

When the counter reaches the set count threshold (HALL_TH), a counter overflow interrupt is generated, and the counter resets to 0.

## 13.4Register Mapping

(HALL base address = 0x4006_4500) RO: read only; WO: write only; R/W: read/write.

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CFG(P1B) | 0x000 | R/W | HALL Module Configuration Register | 0x0 |
| INFO(P1B) | 0x004 | RO | HALL Module Information Register | 0x0 |
| CLRF(P1B) | 0x008 | WO | HALL Interrupt Flag Clear Register | 0x0 |
| WIDTH(P1B) | 0x00C | RO | HALL Width Counter Value Register | 0x0 |
| TH(P1B) | 0x010 | R/W | HALL Module Counter Threshold Value Register | 0x0 |
| CNT(P1B) | 0x014 | R/W | HALL Count Register | 0x0 |
| LOCK | 0x018 | R/W | HALL Write Enable Register | 0x0 |

Note 1: Registers marked with (P1B) are protected registers.

Note 2: When LOCK == 55H, the registers marked with (P1B) are writable; when LOCK == other values, writing is prohibited.

# 13.5Register Description

## 13.5.1 HALL Module Configuration Register (CFG)

| Bit | Symbol | Decription | Reset value |
|---|---|---|---|
| 31 | - | Reserved | - |
| 30 | SW_IE | Software-triggered HALL signal change interrupt enable<br>0: Disable<br>1: Enable, CLRF[16]=1, and a HALL signal change interrupt will be generated manually. | 0 |
| 29 | OV_IE | HALL counter overflow interrupt enable<br>0: Disable<br>1: Enable | 0 |
| 28 | CHG_IE | HALL signal change interrupt enable<br>0: Disable<br>1: Enable | 0 |
| 27:25 | - | Reserved | - |
| 24 | HALL_EN | HALL module enable switch<br>0: Disable<br>1: Enable | 0 |
| 23:21 | - | Reserved | - |
| 20 | FIT_75 | 7/5 filter enable (7 consecutive samples, 5 times the same value)<br>0: Disable<br>1: Enable | 0 |
| 19:18 | - | Reserved | - |
| 17:16 | CLK_DIV | HALL clock division factor<br>00: No division<br>01: Divided by 2<br>10: Divided by 4<br>11: Divided by 8 | 0x0 |
| 15:0 | FIT_LEN | Filter width, signals lower than the corresponding pulse width will be automatically filtered out by the hardware, the formula of the filter width is [15:0]+1 | 0x0 |

## 13.5.2 HALL Module Information Register (INFO)

| Bit | Symbol | Decription | Reset value |
|---|---|---|---|
| 31:29 | - | Reserved | - |
| 18:16 | RAW_DATA | HALL[2:0] real time value | 0x0 |
| 15:11 | - | Reserved | - |
| 10:8 | FIT_DATA | HALL[2:0] filter value | 0x0 |
| 7:2 | - | Reserved | - |
| 1 | OV_IF | HALL counter overflow event flag<br>0: No overflow<br>1: Overflow | 0 |
| 0 | CHG_IF | HALL event change flag<br>0: Without event generation<br>1: With event generation | 0 |

### 13.5.3 HALL Message Clear Register (CLRF)

| Bit | Symbol | Decription | Reset value |
|---|---|---|---|
| 31:17 | - | Reserved | - |
| 16 | SET_SWF | Software trigger HALL signal change flag trigger bit<br>0: No trigger action<br>1: Trigger HALL to generate an event, CHG_IF is set to 1, and this bit is automatically cleared to 0. | 0 |
| 15:2 | - | Reserved | - |
| 1 | CLR_OVF | HALL counter overflow event flag clear bit<br>0: No clear action<br>1: Clear HALL counter overflow flag OV_IF | 0 |
| 0 | CLR_CHGF | HALL signal change event flag clear bit<br>0: No clear action<br>1: Clear HALL event change flag CHG_IF | 0 |

### 13.5.4 HALL Module Width Counter Value Register (WIDTH)

| Bit | Symbol | Decription | Reset value |
|---|---|---|---|
| 31:0 | CAP_CNT | HALL width counter value | 0x0 |

### 13.5.5 HALL Module Counter Threshold Value Register (TH)

| Bit | Symbol | Decription | Reset value |
|---|---|---|---|
| 31:0 | TH | HALL counter threshold value | 0x0 |

### 13.5.6 HALL Counter Register (CNT)

| Bit | Symbol | Decription | Reset value |
|---|---|---|---|
| 31:0 | CNT | HALL count value, write any value to clear it. | 0x0 |

### 13.5.7 HALL Write Enable Register (LOCK)

| Bit | Symbol | Decription | Reset value |
|---|---|---|---|
| 31:0 | LOCK | When LOCK=0X55, enable the operation of HALL related registers. | 0x0 |

# Chapter 14 Enhanced PWM(EPWM)

## 14.1Overview

The enhanced PWM module supports 8 PWM generators, which can be configured as 8 independent PWM outputs (EPWM0-EPWM7), or configured as 4 pairs of complementary or synchronized PWM outputs, each with a programmable dead-time generator (EPWM0-EPWM1, EPWM2-EPWM3, EPWM4-EPWM5, EPWM6-EPWM7).

Each pair of PWM outputs shares an 8-bit prescaler, with 8 sets of clock dividers providing 5 different division factors (1, 1/2, 1/4, 1/8, 1/16). Each PWM output has an independent 16-bit counter for control, and a 16-bit comparator is used to adjust the duty cycle. The 8 PWM generators provide 36 interrupt flags; when the period or duty cycle of a related PWM channel matches the counter, an interrupt flag is generated. Each PWM channel has a separate enable bit.

Each PWM channel can be configured in one-shot mode (to generate one PWM signal cycle) or continuous mode (to continuously output the PWM waveform).

## 14.2Features

The enhanced PWM module has the following features:

- 8 independent 16-bit PWM control modes:
  - 8 independent outputs: EPWM0, EPWM1, EPWM2, EPWM3, EPWM4, EPWM5, EPWM6, EPWM7.
  - 4 pairs of complementary PWM: (EPWM0-EPWM1), (EPWM2-EPWM3), (EPWM4-EPWM5), (EPWM6-EPWM7), with programmable dead-time insertion.
  - 4 pairs of synchronized PWM: (EPWM0-EPWM1), (EPWM2-EPWM3), (EPWM4-EPWM5), (EPWM6-EPWM7), with synchronized pins for each pair.
- Supports group control, where EPWM0, EPWM2, EPWM4, and EPWM6 outputs are synchronized, and EPWM1, EPWM3, EPWM5, and EPWM7 outputs are synchronized.
- One-shot mode (only supports edge-aligned) or auto-load mode.
- Supports both edge-aligned and center-aligned modes.
- Center-aligned mode supports both symmetric and asymmetric counting.
- In complementary PWM mode, supports a programmable dead-time generator.
- Each PWM channel has independent polarity control.
- Fault brake protection and recovery functionality (both software/hardware trigger and software/hardware recovery).
- ACMP0/1 output and ADC comparator output can trigger hardware brake protection.
- PWM edge, period point, zero point, and comparison point 0/1 can trigger ADC conversion start.

## 14.3Function Description

The enhanced PWM module has the following features:

1) Period Point: When the counter CNTn counts to be equal to the period PERIODn, it is called the period point. The interrupt generated is PIFn.

2) Zero Point: When the counter CNTn counts to 0, it is called the zero point. The interrupt generated is ZIFn.

3) Up-Count Compare Point: When the counter CNTn incrementally counts to be equal to CMPDATn, it is called the up-count compare point. The interrupt generated is UIFn. There is no up-count compare point in edge-aligned counting mode.

4) Down-Count Compare Point: When the counter CNTn decrementally counts to be equal to CMPDATn or CMPDDATn, it is called the down-count compare point. The interrupt generated is DIFn.

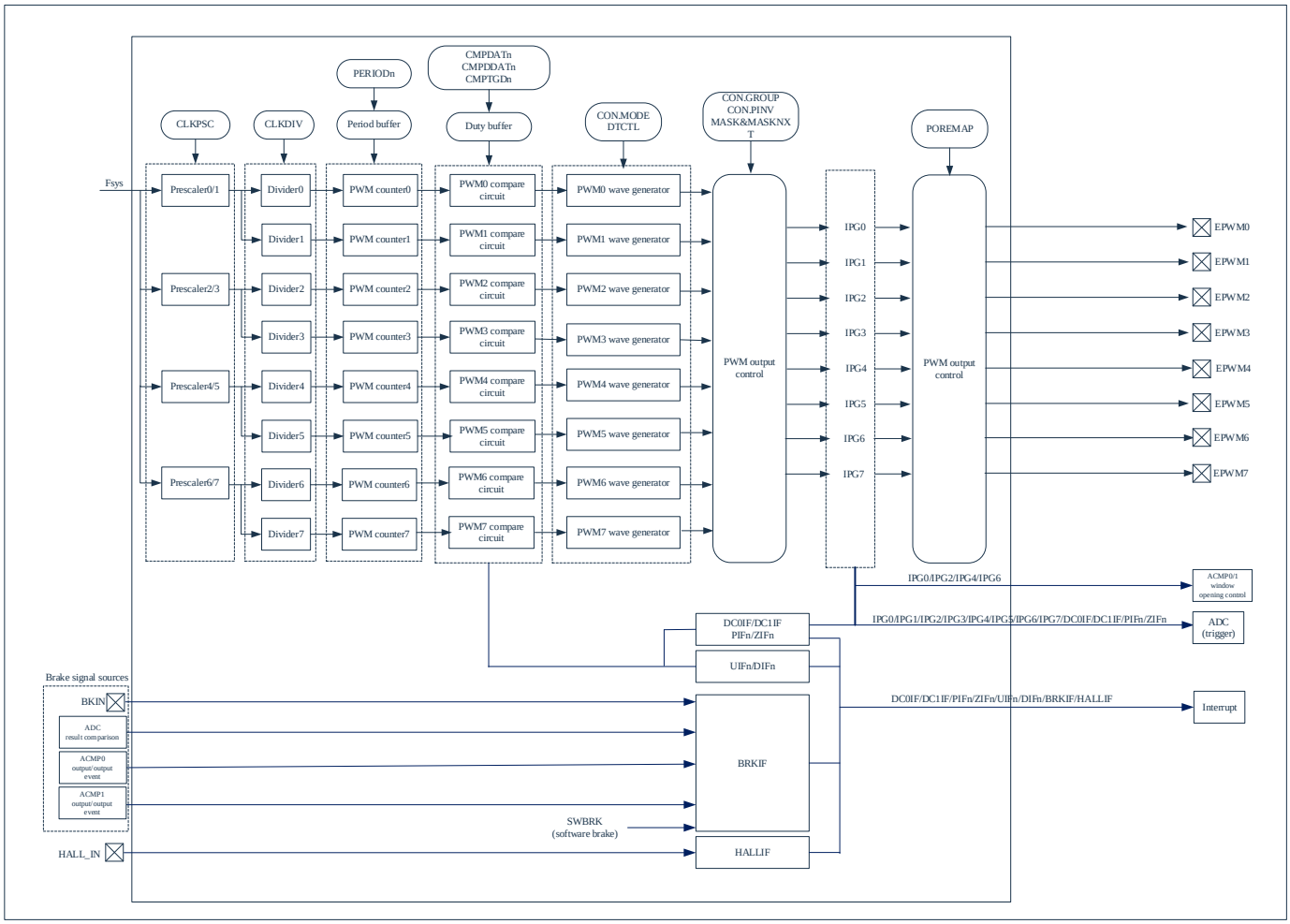
5) Center Point: The center point refers to the moment when CNTn counts to be equal to PERIODn in center-aligned counting mode. It is called the center point because CNTn will count down to 0 afterward, making it also a period point. There is no center point in edge-aligned counting mode, but there is a period point.

Caution 1: In edge-aligned mode, the period data is loaded at the start of the first count, which generates the period point. Since the counter counts to 0 afterward, the positions of subsequent period points coincide with zero points. This alignment mode has down-count compare points but no up-count compare points.

Caution 2: In center-aligned mode, the count starts from 0 and increments upward, generating the zero point. When it counts to the period data, it generates the period point (center point). The zero point alternates with the center point. This alignment mode has both up-count compare points and down-count compare points. In symmetric counting, both up-count compare points and down-count compare points are determined by CMPDATn. In asymmetric counting, the up-count compare point is determined by CMPDATn, and the down-count compare point is determined by CMDDATn.

## 14.3.1 Block Diagram

Figure 14-1: Block diagram of EPWM




## 14.3.2 Clock Division

Each PWM shares the same 8-bit prescaler, after the prescaler, each PWM can select 5 kinds (1, 1/2,1/4,1/8,1/16) of prescaler ratios.

PWM_CLK = PCLK/(CLKPSCxx + 1)/CLKDIVn, here xx can be 01, 23, 45, 67, n=0-7.

## 14.3.3 Independent Output Mode

The 8 EPWM channel outputs do not affect each other and operate according to their own period/duty cycle data.

### 14.3.4 Complementary Output Mode

In complementary output mode, the 8 PWM channels are divided into 4 pairs: EPWM0 and EPWM1 form one pair, EPWM2 and EPWM3 form another pair, EPWM4 and EPWM5 form another pair, and EPWM6 and EPWM7 form the final pair. A total of 4 pairs of PWM are present.

EPWM0-EPWM1 operates according to the period/duty cycle data of EPWM0, with the waveforms of EPWM0 and EPWM1 being inverted.

EPWM2-EPWM3 operates according to the period/duty cycle data of EPWM2, with the waveforms of EPWM2 and EPWM3 being inverted.

EPWM4-EPWM5 operates according to the period/duty cycle data of EPWM4, with the waveforms of EPWM4 and EPWM5 being inverted.

EPWM6-EPWM7 operates according to the period/duty cycle data of EPWM6, with the waveforms of EPWM6 and EPWM7 being inverted.

In this mode, the outputs of EPWM1, EPWM3, EPWM5, and EPWM7 are not directly controlled by their corresponding data registers, but the output control is still effective, such as enabling, masking, brake control, etc.

The complementary mode supports dead-time delay control.

### 14.3.5 Synchronous Output Mode

In the synchronous output mode, the 8 PWM channels are divided into 4 pairs: EPWM0 with EPWM1, EPWM2 with EPWM3, EPWM4 with EPWM5, and EPWM6 with EPWM7. There are a total of 4 pairs of PWM.

EPWM0 and EPWM1 operate based on the cycle and duty cycle data of EPWM0, with the waveforms of EPWM0 and EPWM1 being in phase.

EPWM2 and EPWM3 operate based on the cycle and duty cycle data of EPWM2, with the waveforms of EPWM2 and EPWM3 being in phase.

EPWM4 and EPWM5 operate based on the cycle and duty cycle data of EPWM4, with the waveforms of EPWM4 and EPWM5 being in phase.

EPWM6 and EPWM7 operate based on the cycle and duty cycle data of EPWM6, with the waveforms of EPWM6 and EPWM7 being in phase.

In this mode, the outputs of EPWM1, EPWM3, EPWM5, and EPWM7 are independent of their corresponding data registers, but output control (such as enable, mask, braking, etc.) is still effective.

### 14.3.6 Group Output Mode

When GROUPEN = 1, the grouping function is enabled. The 8 PWM channels are divided into two groups: EPWM0, EPWM2, EPWM4, EPWM6 form one group, and EPWM1, EPWM3, EPWM5, EPWM7 form another group.

EPWM0, EPWM2, EPWM4, and EPWM6 operate based on the cycle and duty cycle data of EPWM0, with all 4 channels' waveforms being in phase.

EPWM1, EPWM3, EPWM5, and EPWM7 operate based on the cycle and duty cycle data of EPWM1, with all 4 channels' waveforms being in phase.

When the grouping function is enabled, the outputs of EPWM2, EPWM4, EPWM6, EPWM3, EPWM5, and EPWM7 are independent of their corresponding data registers, but output control (such as enable, mask, braking, etc.) is still effective.

### 14.3.7 Load Update Mode

There are two types of counter loading modes: One-shot and Continuous (auto-loading mode).

**One-shot mode:**

The period and duty cycle-related data are loaded into the counter once it begins counting.

**Continuous mode:**

In this mode, the duty cycle data is automatically loaded at zero point and center point within the PWM period. The center point loading only exists in center-aligned counting mode.

In edge-aligned counting mode, a zero point is generated along with a period point, and the counting comparison circuit reloads the values of CMPDATn/PERIODn/CMPTGD0/CMPTGD1.

In the center-aligned counting mode, both the center point and the zero point are automatically loaded with the values of the associated registers. This structure allows the first half of the waveform cycle duty cycle to be set differently from the second half of the waveform cycle duty cycle, and then remain the same when the period duty cycle related registers are not changed.

Due to the double-buffered structure of EPWM, when changing the values of the running registers such as CMPDATn/CMPDDATn/CMPDATn/CMPDDATn/PERIODn/CMPTGD0/CMPTGD1 during EPWM operation, the PWM output waveform will not change immediately. Only at the zero-point or period point, these register values will be loaded into the corresponding buffer.

With this structure, after changing the duty cycle data, the current PWM period or half-period output waveform will not immediately change. The PWM waveform will only change in the next period or half-period. That means any changes to PWM-related data will not affect a current complete PWM period or half-period.

In high-speed applications, it is possible that the loading point has arrived, but the write operation to the running registers has not been completed. In this case, it is not expected to have partial running data loaded while the other part is not.

To address this high-speed application scenario, the EPWM module provides a loading enable bit. After changing the relevant running registers, the loading enable bit LOADENn needs to be set to 1. After the loading is completed, the LOADENn bit will automatically be cleared. Additionally, the state of this bit can be read to determine whether the values of the relevant registers have been loaded into the actual circuit. If LOADENn=0, it means that the values have been loaded and will affect the output PWM waveform. If LOADENn=1, it means that the values have not been loaded yet and the current PWM waveform has not changed. The values of the previously changed registers will be loaded at the next loading point. If the relevant running registers are changed again, the LOADENn bit also needs to be set to 1 again.

By default, PWM will load the running data of the relevant registers at both zero point and period point and generate zero point and period point interrupts. To adapt to more flexible application requirements, PWM supports different loading methods and zero point/period point interrupt generation methods.

In the register EPWMCON3, LOADTYPn (0-7) can be set to determine the loading method and the interrupt generation method for the zero point/period point:

| LOADTYEn | Center-aligned loading | Edge-aligned loading |
|---|---|---|
| 00 | Load and generate zero point and period point interrupt flags at each zero point or period point | Load and generate zero point and period point interrupt flags at each zero point or period point |
| 01 | Load and generate zero point interrupt flags at each zero point | Load and generate zero point interrupt flags at every 2 zero points |
| 10 | Alternate load and generate zero point and period point interrupt flags between the first zero point and the next period point | Load and generate zero-point and period point interrupt flags at every 3 zero points or period points |
| 11 | Load and generate related zero point interrupt flags every two zero points | Load and generate zero point interrupt flags at every 4 zero points |

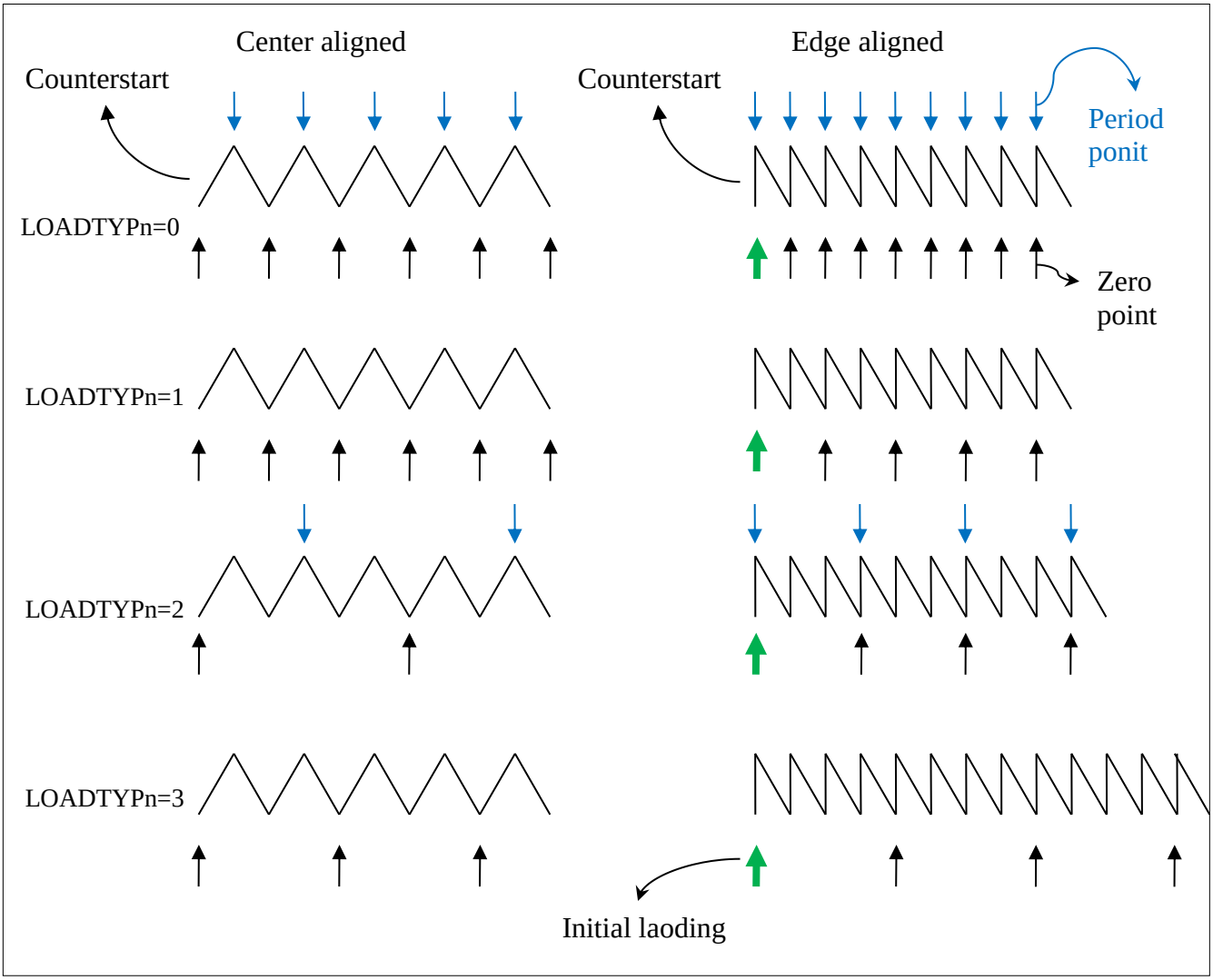
When LOADTYPn = 0, the data is loaded once every 1 cycle for edge alignment and once every 0.5 cycles for center alignment.

When LOADTYPn = 1, the data is loaded once every 2 cycles for edge alignment and once every 1 cycle for center alignment.

When LOADTYPn = 2, the data is loaded once every 3 cycles for edge alignment and once every 1.5 cycles for center alignment.

When LOADTYPn = 3, the data is loaded once every 4 cycles for edge alignment and once every 2 cycles for center alignment.

Figure 14-2: Updated block diagram of PWM period/duty cycle loading



### 14.3.8 Edge-Aligned Counting Mode

In edge-aligned mode, with counting down method, the 16-bit PWM counter CNTn starts counting down at the beginning of each cycle. It compares with the latched value CMPDATn, and when CNTn=CMPDATn, EPWMn outputs a high-level signal and sets CMPnDIF to 1. The CNTn continues counting down until it reaches 0, at which point EPWMn outputs a low-level signal. When PWMnCNTM=1, the current CMPDATn and PERIODn will be reloaded, and PIF (period interrupt flag) will be set.

Edge-aligned related parameters:

High level time= (CMPDATn+1)x Tpwm

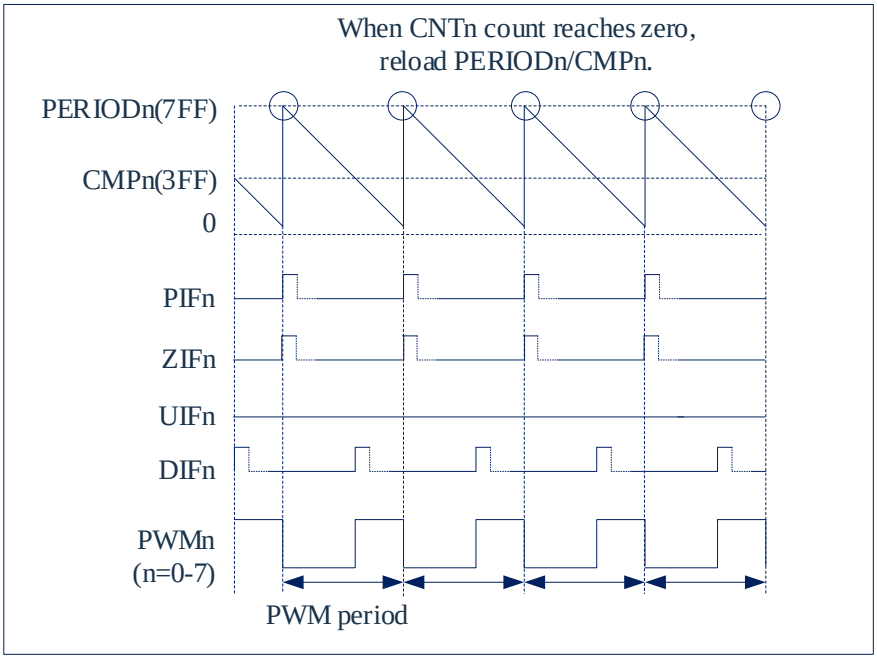
Period= (PERIODn+1)x Tpwm

$$Duty\ cycle = \frac{CMPDATn + 1}{PERIODn + 1}$$

If CMPDATn > PERIODn, the duty cycle is 100%, and the EPWMn channel remains high. It will not generate a down-compare interrupt.

If CMPDATn = 0, the duty cycle is 0%.

Figure 14-3: Edge alignment mode waveform



### 14.3.9 Center-Aligned Counting Mode

In center-aligned mode, the counting process starts by counting up and then counts down.

Center-aligned mode can be further divided into two types: symmetric counting mode and asymmetric counting mode.

In symmetric counting mode (ASYMEN=0), the duty cycle is determined by CMPDATAn.

In asymmetric counting mode (ASYMEN=1), the duty cycle is determined by both CMPDATAn and CMPDDATn.

In center-aligned symmetric counting mode, the 16-bit PWM counter CNTn starts counting up from 0. When CNTn reaches CMPDATn, EPWMn outputs a high level. Then, CNTn continues counting up until it reaches the value of PERIODn. After that, CNTn starts counting down. During the counting down process, when CNTn = CMPDATn, EPWMn outputs a low level. Then, it continues counting down until it reaches 0.

$$High\ level\ time = (PERIODn \times 2 - CMPDATn \times 2 - 1) \times Tpwm$$

$$Period = PERIODn \times 2 \times Tpwm$$

$$Duty\ cycle = \frac{PERIODn \times 2 - CMPDATn \times 2 - 1}{PERIODn \times 2}$$

If CMPDATn is greater than or equal to PERIODn, the duty cycle is 0%, and EPWMn channel remains low, without generating any up-compare or down-compare interrupts.

If PERIODn is 0, the duty cycle is 0%, and EPWMn channel remains low, with zero interrupts and period-point interrupts present as long as CNTn is enabled.

If CMPDATn is 0, the duty cycle is 100%.

Figure 14-4: Center-aligned mode symmetric counting waveform

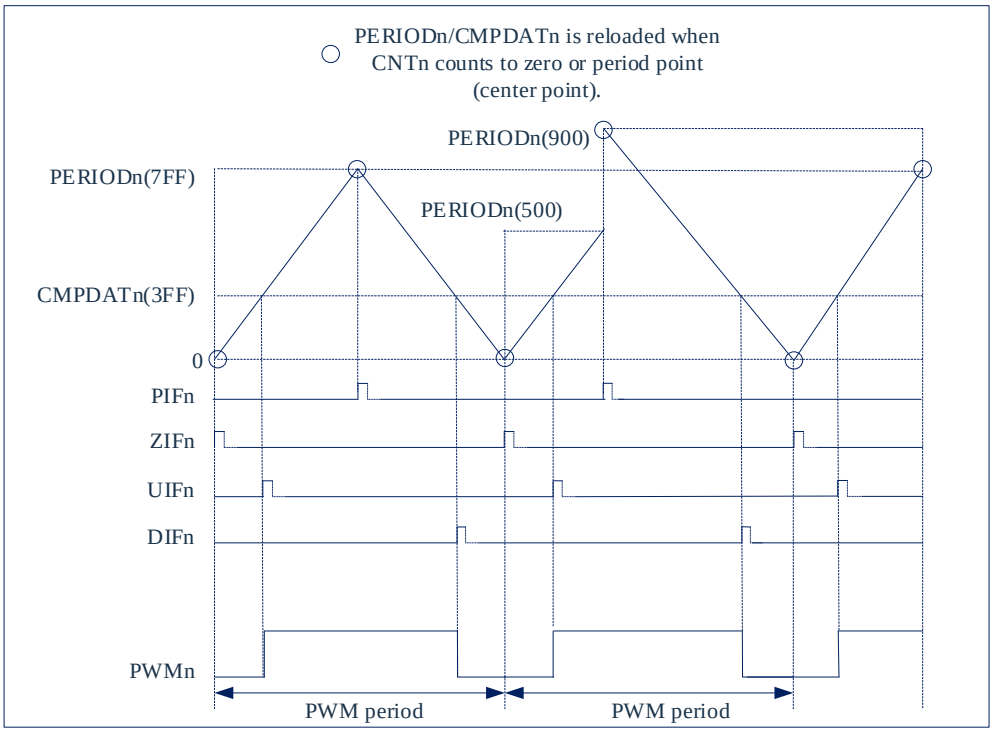
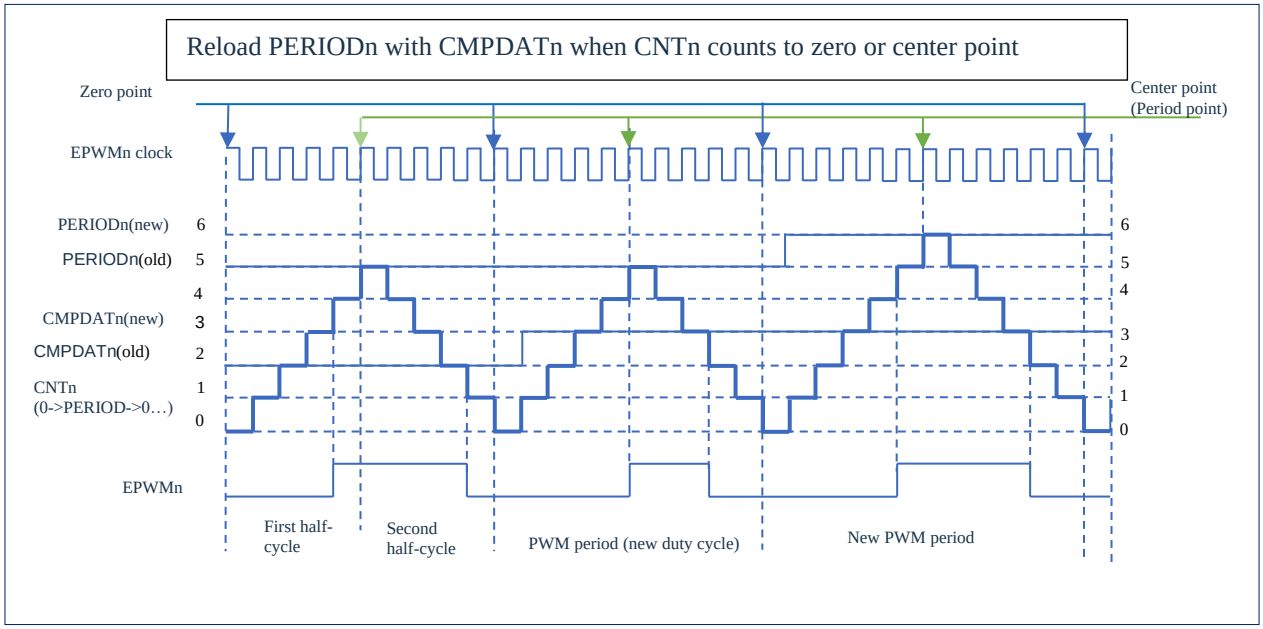



Figure 14-5: Center-aligned counter waveform (symmetric counting)




In center-aligned asymmetric counting mode, the 16-bit PWM counter CNTn starts counting from 0 and increments upward. When CNTn = CMPDATn, EPWMn outputs a high level. After that, CNTn continues counting upward until it reaches PERIODn. Then, CNTn starts counting downward. During the downward counting process, when CNTn = CMPDDATn, EPWMn outputs a low level. Afterward, it continues counting downward until it reaches 0. To enable center-aligned asymmetric counting mode, ASYMEN needs to be set to 1. The asymmetric counting mode can realize the precise center-aligned waveform.

The parameters related to center-aligned asymmetric counting mode are as follows:

$$High\ level\ time = (PERIODn \times 2 - CMPDDATn - CMPDATn - 1) \times Tpwm$$

$$Period = PERIODn \times 2 \times Tpwm$$

$$Duty\ cycle = \frac{PERIODn \times 2 - CMPDDATn - CMPDATn - 1}{PERIODn \times 2}, (CMPDATn < PERIODn, CMPDDATn < PERIODn)$$

$$Duty\ cycle = \frac{PERIODn - CMPDDATn - 1}{PERIODn \times 2}, (CMPDATn \geq PERIODn, CMPDDATn < PERIODn)$$

$$Duty\ cycle = \frac{PERIODn - CMPDATn}{PERIODn \times 2}, (CMPDATn < PERIODn, CMPDDATn \geq PERIODn)$$

$$Duty\ cycle = 0\%, (CMPDATn \geq PERIODn, CMPDDATn \geq PERIODn)$$

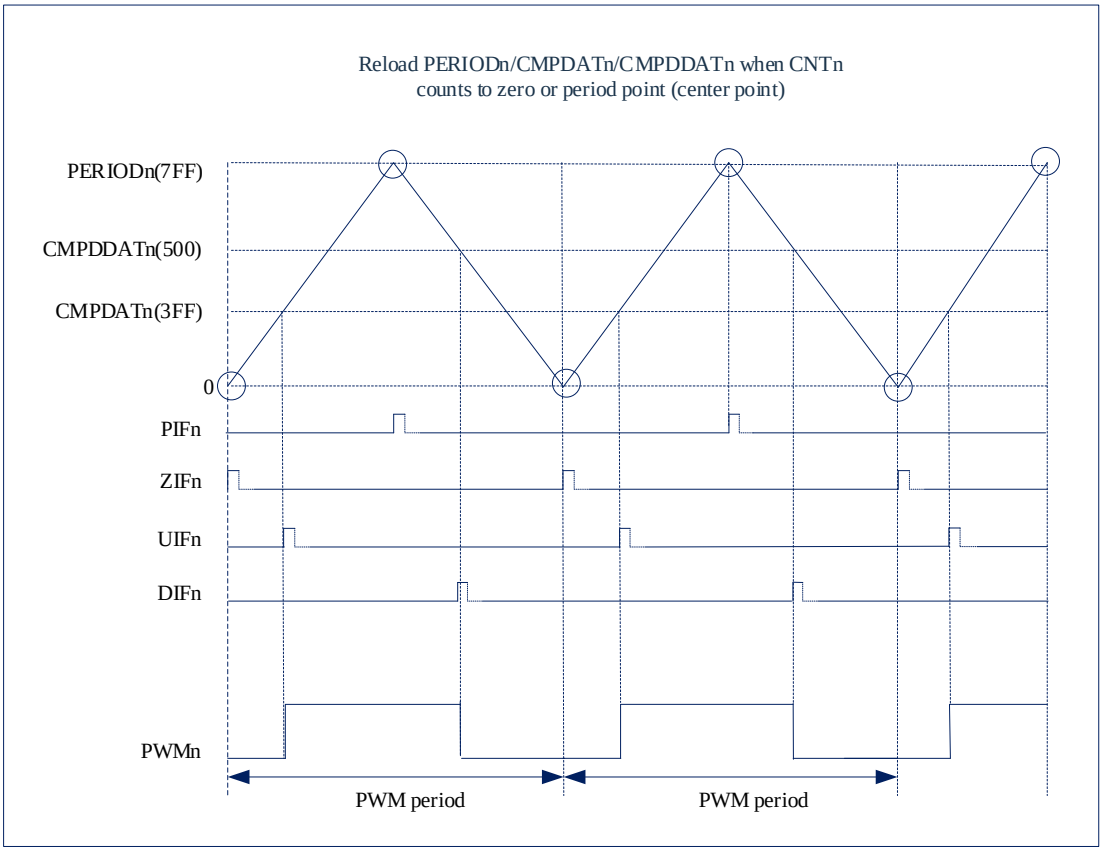
If CMPDATAn≥PERIODn, an upward comparison interrupt will not be triggered.

If CMPDDATAn≥PERIODn, a downward comparison interrupt will not be triggered.

If PERIODn = 0, the duty cycle is 0%, and the EPWMn channel remains low. When CNTn is enabled, both the zero-crossing interrupt and the period point interrupt will always be present.

If CMPDATn = 0 and CMDATDn = 0, the duty cycle is 100%.

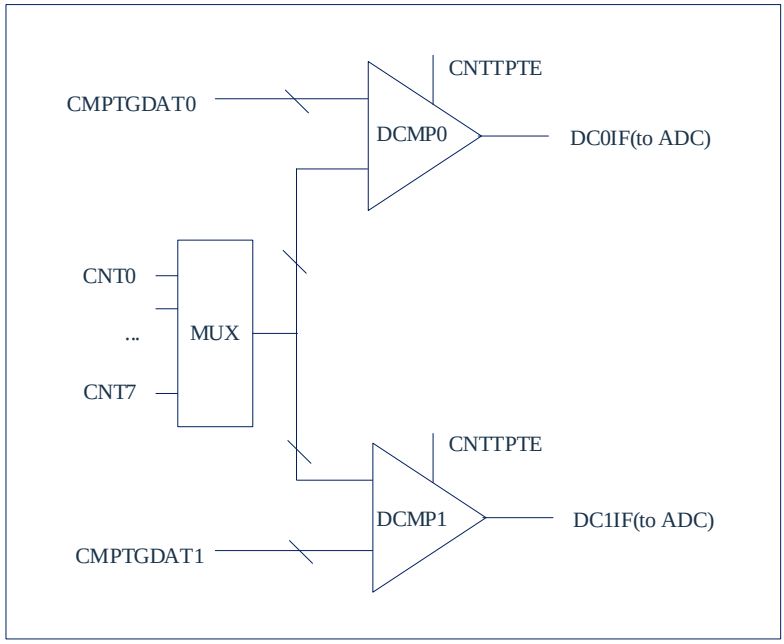
Figure 14-6: Center-aligned mode asymmetric counting waveform



## 14.3.10 Independent Counter Compare Function

During the counting of the PWMn channel counter (CNTn), two digital comparators are provided to compare the counter value with pre-set values. If the counter value equals the pre-set value, an interrupt signal or ADC trigger can be generated. This function does not affect the PWM output.

Figure 14-7: Independent counter compare function

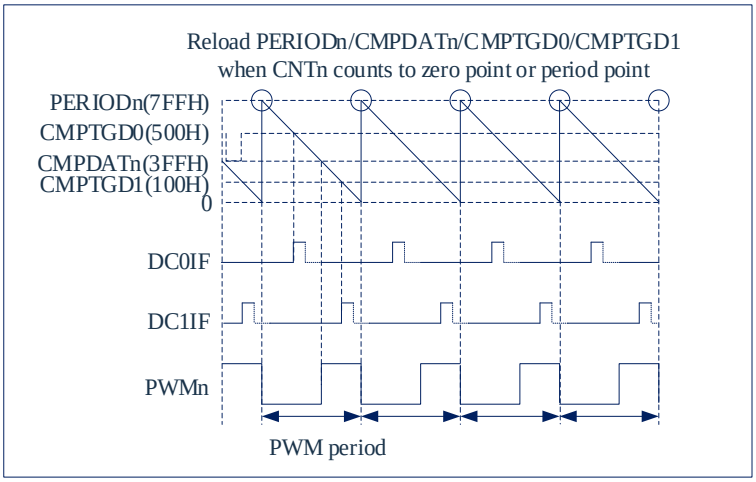



Digital Comparator 0 compares the value of CNTn with CMPTGDAT0. If they are equal, the interrupt flag DC0IF is generated. CMPTGD0[18:16] selects one of the PWM0-7 channel counters to compare with CMPTGDAT0.

Digital Comparator 1 compares the value of CNTn with CMPTGDAT1. If they are equal, the interrupt flag DC1IF is generated. CMPTGD1[18:16] selects one of the PWM0-7 channel counters to compare with CMPTGDAT1.

1) In edge-aligned mode, digital comparator operation method:

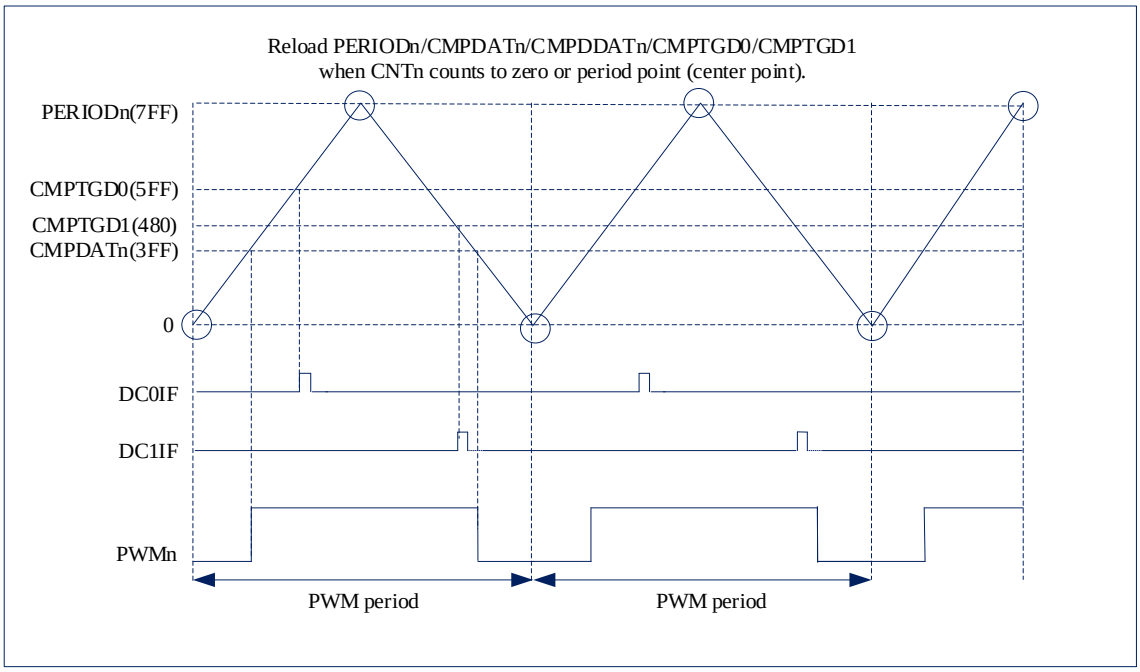
Figure 14-8: Edge-aligned mode, digital comparator operation method




In edge counting mode, the digital comparator 0/1 can be set to generate a compare interrupt at any counting moment.

2) In center-aligned mode, digital comparator operation method:

Figure 14-9: Center-aligned mode, digital comparator operation method




In center-aligned counting mode, the digital comparators 0/1 can each be set to trigger in either upward or downward counting mode. That is, both can be triggered in the first half-cycle or the second half-cycle, or one can be triggered in the first half-cycle and the other in the second half-cycle. This is determined by the CMPTGD0[19] bit CMPTGDSn.

### 14.3.11 Programmable Dead-Time Generator

The 8-channel PWM can be configured into 4 pairs of complementary outputs. In complementary output mode, the period and duty cycle of PWM1, PWM3, PWM5, and PWM7 are determined by the related registers of PWM0, PWM2, PWM4, and PWM6, respectively. Additionally, the dead-time delay register also affects the duty cycle of the complementary PWM pairs. In this mode, except for the corresponding output enable control bit (PWMnOE), channel remapping control, masking control, braking control, group control, and polarity control, the output waveforms of PWM1/PWM3/PWM5/PWM7 are no longer controlled by their own registers.

In complementary mode, each complementary PWM pair supports the insertion of dead-time delay, as described below:

PWM0/1 dead-time: Left dead-time = (PWM01LDT[15:0] + 1) * TPWM0; Right dead-time = (PWM01RDT[15:0] + 1) * TPWM0

PWM2/3 dead-time: Left dead-time = (PWM23LDT[15:0] + 1) * TPWM2; Right dead-time = (PWM23RDT[15:0] + 1) * TPWM2

PWM4/5 dead-time: Left dead-time = (PWM45LDT[15:0] + 1) * TPWM4; Right dead-time = (PWM45RDT[15:0] + 1) * TPWM4

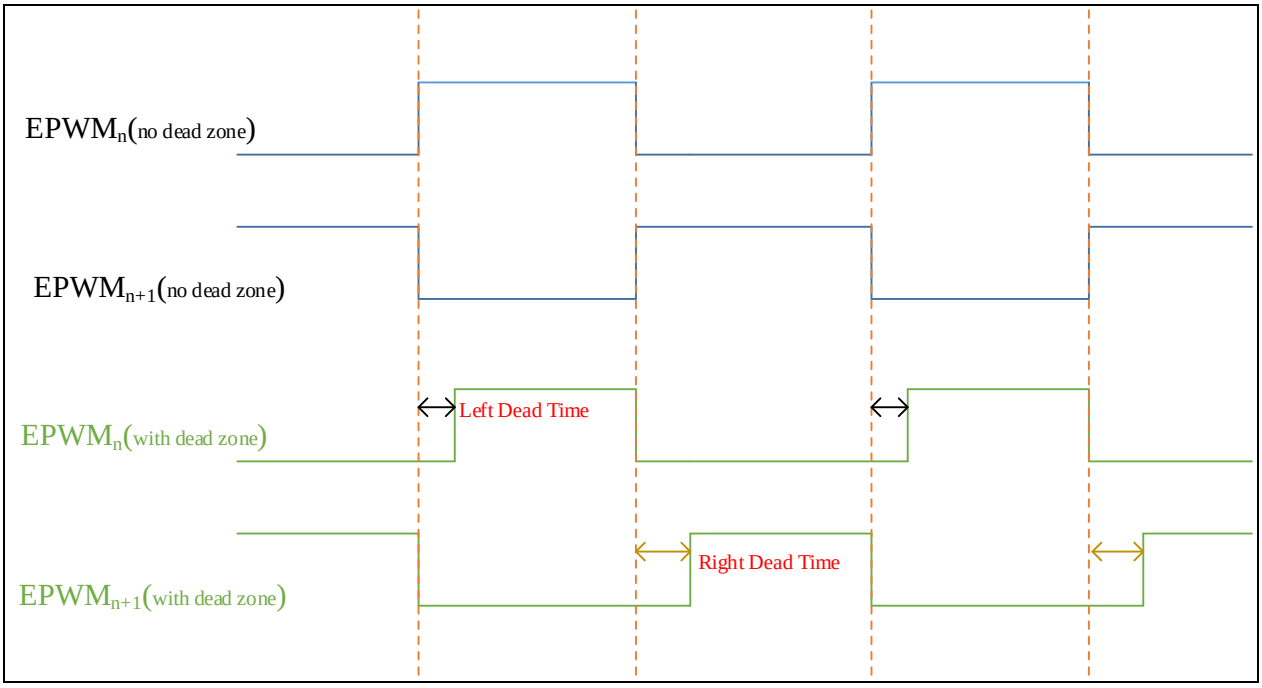
PWM6/7 dead-time: Left dead-time = (PWM67LDT[15:0] + 1) * TPWM6; Right dead-time = (PWM67RDT[15:0] + 1) * TPWM6

TPWM0/TPWM2/TPWM4/TPWM6 refer to the clock source period for PWM0/PWM2/PWM4/PWM6.

The dead-time delay is adjustable in the range of 0.014 μs to 904.40 μs (with Fpwmn = 72 MHz).

The output mode does not affect the counter mode, so both center-aligned and edge-aligned configurations support complementary output mode.

Figure 14-10: Complementary output mode supported by center-aligned and edge-aligned modes



## 14.3.12 Mask and Mask Preset Function

EPWM supports the mask function. Each channel of EPWM0-EPWM7 has individual control, and the corresponding control bits for EPWMn are MASKENn and MASKDn (in the MASK register).

When MASKENn=0, the EPWMn channel outputs the normal PWM waveform.

When MASKENn=1, the EPWMn channel outputs the data from MASKDn.

The control register MASK for the mask function also supports the automatic loading of preset values. To enable this feature, set the MASKLE bit in the output control register POEN to 1, allowing MASK to automatically load the value from the MASKNXT register, while disabling writing to the MASK register.

The loading time is determined by the MASKLS<2:0> bits in POEN, which can be set to match the load cycle/duty (loading point) of one of EPWM0-EPWM7.

## 14.3.13 Hall Sensor Interface Function

EPWM considers the interface with a Hall sensor. It includes an internal HALL position detection circuit that detects the levels of the filtered CCP0/1 module's internal capture channels CAP0, CAP1, and CAP2.

After internal processing, the detection circuit produces a state called HALLST:

HALLST has eight states, corresponding to the HALL position states as follows:

| HALLST | Corresponding state |
|---|---|
| 000 | HALL detection circuit is not started or initial state |
| 001 | {CAP2-CAP0}=001 |
| 010 | {CAP2-CAP0}=010 |
| 011 | {CAP2-CAP0}=011 |
| 100 | {CAP2-CAP0}=100 |
| 101 | {CAP2-CAP0}=101 |
| 110 | {CAP2-CAP0}=110 |
| 111 | Error state during {CAP2-CAP0} change process or incorrect sequence. |

The value of HALLST can be read from the MASKNXT register, allowing the HALL position or sequence state to be determined at any time.

The HALL state detection sequence supports the following two orders ({CAP2, CAP1, CAP0} appearing in a sequence):

- ……-6-2-3-1-5-4-6-……
- ……-6-4-5-1-3-2-6-……

If any other sequence occurs, it is considered an error, and HALLST will enter the 111 state and stop the detection. It will also generate an interrupt flag HALLIF. To restart the HALL detection circuit, set the HALLCLR bit in the MASKNXT register to 1, and HALLST will shift from the 111 state to the initial 000 state to restart the detection circuit.

The HALL detection circuit provides a feature that can automatically load related functions with the mask. This feature allows controlling the output channel waveform of EPWM without software intervention.

Each valid state of HALLST corresponds to a mask preset cache, and there are totally seven mask preset caches:

| HALLST(HALLEN=1) | Corresponding mask preset cache: |
|---|---|
| 000 | Mask preset cache 7 |
| 001 | Mask preset cache 1 |
| 010 | Mask preset cache 2 |
| 011 | Mask preset cache 3 |
| 100 | Mask preset cache 4 |
| 101 | Mask preset cache 5 |
| 110 | Mask preset cache 6 |
| 111 | Mask preset cache 7 |
| HALLEN=0 | Mask preset cache 0 |

If the feature of automatic loading of mask preset values is enabled, then at the corresponding state and at the selected loading point, the data in the corresponding mask preset cache will be loaded into the MASK register. For example:

When the position state in HALLST changes from 000 to 001 and enters the first loading point of state 001, the data in Mask Preset Cache 1 will be loaded into the MASK register.

Later, when the position state in HALLST changes from 001 to 101 and enters the first loading point of state 101, the data in Mask Preset Cache 5 will be loaded into the MASK register.

If an incorrect sequence occurs, such as the CAP2-CAP0 input changing from 101 to 010, which is not the correct sequence, the position state in HALLST will change from 101 to 111, and the interrupt flag HALLIF will be set to 1. At the first loading point of state 111, the data in Mask Preset Cache 7 will be loaded into the MASK register.

In the initial state, the data in Mask Preset Cache 7 is loaded into the MASK register at the loading point.

The following diagram is a timing example of the HALL detection and mask preset functionality (for reference only, not representing actual waveforms), where the HALL detection function is enabled, and the EPWM mask control preset data loading is enabled. The mask preset caches are set as follows:

Mask preset cache 1: 0xFF55

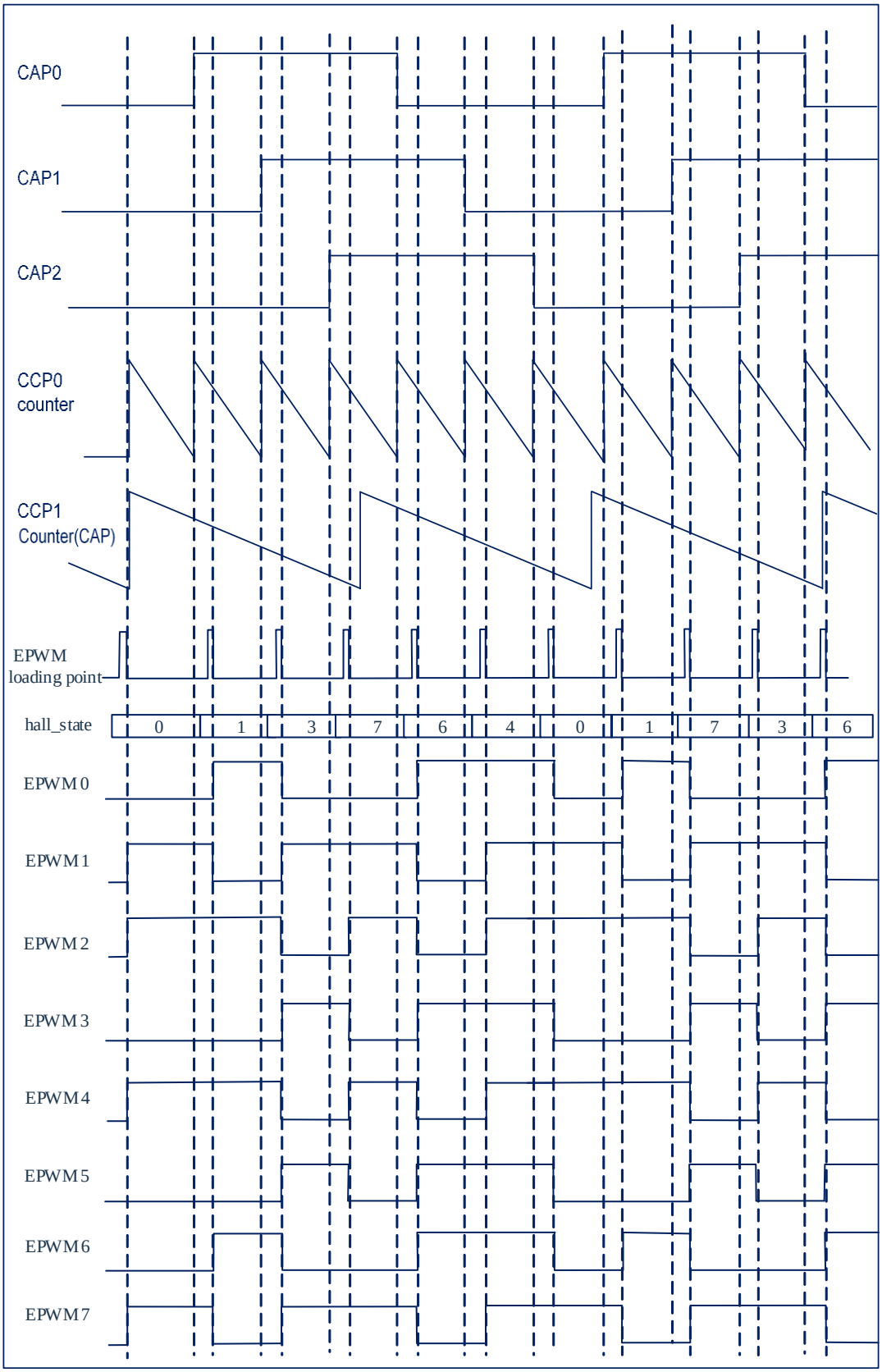
Mask preset cache 3: 0xFFAA

Mask preset cache 4: 0xFFFF

Mask preset cache 6: 0xFF96

Mask preset cache 7: 0xFF96

Figure 14-11: Example of HALL detection and mask preset timing



### 14.3.14 Fault Protection Function (Brake and Recovery Function)

EPWM supports fault protection functionality, with BRKODn controlling the brake output level of 8 channels. The fault protection function is controlled by the BRKCTL register.

The fault protection triggers for EPWM include:

Level trigger sources:

1) External BKIN level signal (high or low level)
2) Software brake signal (SWBRK set to 1)
3) Analog comparator 0 output (high or low output)
4) Analog comparator 1 output (high or low output)

Pulse trigger sources:

1) Edge signal of external BKIN (rising edge or falling edge)
2) Analog comparator 0 output event (rising edge, falling edge, or both edges)
3) Analog comparator 1 output event (rising edge, falling edge, or both edges)
4) ADC result comparator 0 event (result compare event)

Fault interrupt flag bit BRKIF (Cleared to 0 by software):

After detecting a valid brake trigger source signal, the fault interrupt flag BRKIF is set to 1 and needs to be cleared to 0 by software.

Fault signal flag bit BRKAF (read-only):

The fault signal flag BRKAF is set to 1, and it automatically clears to 0 when the brake signal is revoked. BRKAF is a read-only bit.

Fault protection output status flag bit BRKOSF (read-only):

When BRKOSF is 1, it indicates that EPWMn channel outputs the BRKODn data state;

When BRKOSF is 0, it indicates that EPWMn is in normal output state.

It indicates whether the EPWM output is in brake state or normal state. BRKOSF will be set to 1 when a valid brake signal is detected. In software recovery mode, a brake clear operation (BRKCLR=1) will affect the state of this bit.

Fault protection modes can be divided into 4 types to meet different requirements in fault protection scenarios.

| BRKMS | Fault protection mode |
|---|---|
| 00 | Stop Mode (Software Recovery) |
| 01 | Pause Mode (Software Recovery) |
| 10 | Recovery Mode (Hardware Recovery) |
| 11 | Delayed Recovery Mode (Hardware Recovery) |

Note: The fault interrupt flag (BRKIF) is unrelated to the recovery function and only represents the occurrence of a brake signal. The fault interrupt flag also supports accumulation function.

**Stop Mode:**

Generate fault protection and fault interrupt flags, clear the CNTENn bit to 0, and stop the counter operation. To recover the output, the brake signal needs to be revoked, and the fault state clearing operation (BRKCLR=1) needs to be executed, then set CNTENn to 1 again.

**Pause Mode:**

Generate fault protection and fault interrupt flags, but the counter continues to operate. To recover the output, revoke the brake signal, execute the fault state clearing operation (BRKCLR=1), and restore normal output at the most recent load update point.

**Recovery mode:**

Generate fault protection and fault interrupt flags, but the counter continues to operate. After revoking the brake signal, the normal output automatically restores at the most recent load update point. There is no need to execute the fault state clearing operation.

Pay attention to distinguish whether the brake signal is a pulse signal or a level signal: If the brake source is a level signal, the output can only be restored after the brake is revoked; if it is a pulse signal, the EPWM output restores at the most recent load update point after triggering the brake, unless another brake pulse signal is generated during this period.

**Delayed recovery mode:**

Generate fault protection and fault interrupt flags, but the counter continues to operate. After revoking the brake signal, the EPWM restores normal output after a delay time at the most recent load update point. There is no need to execute the fault state clearing operation.

The delay time can be freely set, and the low 16-bit RDT of BRKRDT control the delay time. The delay time is as follows:

$$T_{delay} = (RDT+1)*T_{PCLK}$$

Pay attention to distinguish whether the brake signal is a pulse signal or a level signal: If the brake source is a level signal, the output can only be restored after the brake is revoked; if it is a pulse signal, the EPWM output waits for the completion of the delay time and then restores at the most recent load update point, unless another brake pulse signal is generated during this period.

After generating the brake protection, EPWMn channel outputs the data in BRKODn. Each channel can independently set the output to high or low level.

### 14.3.15 Output Status in Debug Mode

In debug mode, the CPU has two states: operation state and pause state. The operation state is the normal execution state, while the pause state occurs after executing a STOP instruction, reaching a breakpoint, or stepping.

In the pause state, the output status of EPWMn (with POEn=1) can be configured using the HALTMS bit in the CON register.

When HALTMS=0, the output status of EPWMn remains normal during the pause.

When HALTMS=1, the output status of EPWMn during the pause becomes the brake data, but no fault-related flags are generated. The EPWMn counter continues to run, and the EPWMn output is restored to the nearest load update point when the running state is resumed.

It should be noted that in debug mode, the values of the relevant operational data registers of EPWMn do not automatically change and will retain their previous states.

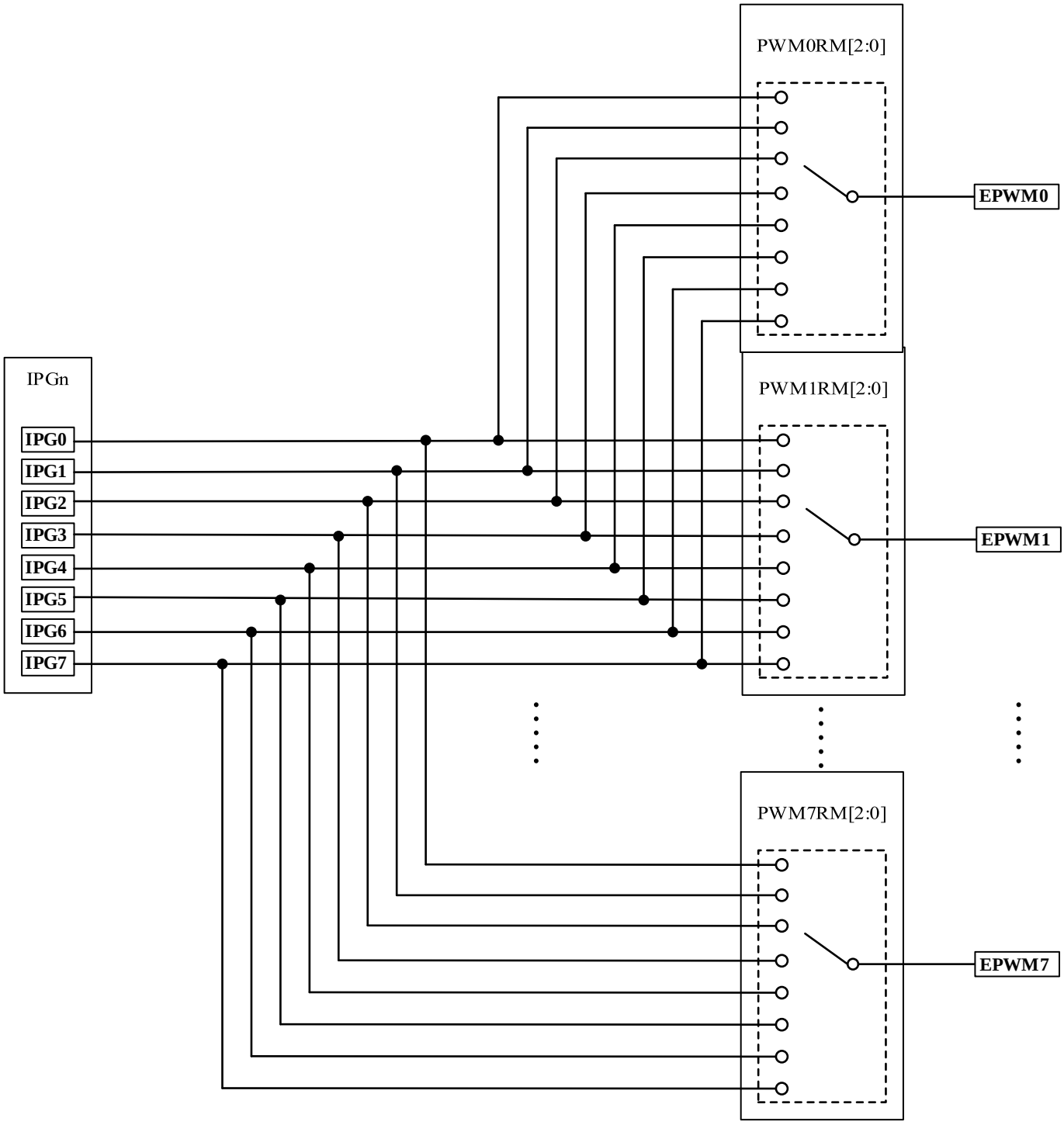
## 14.3.16 Output Channel Remapping Function

The output channel remapping function allows for the reconfiguration of the required channels, providing more flexibility to meet the layout needs of the application.

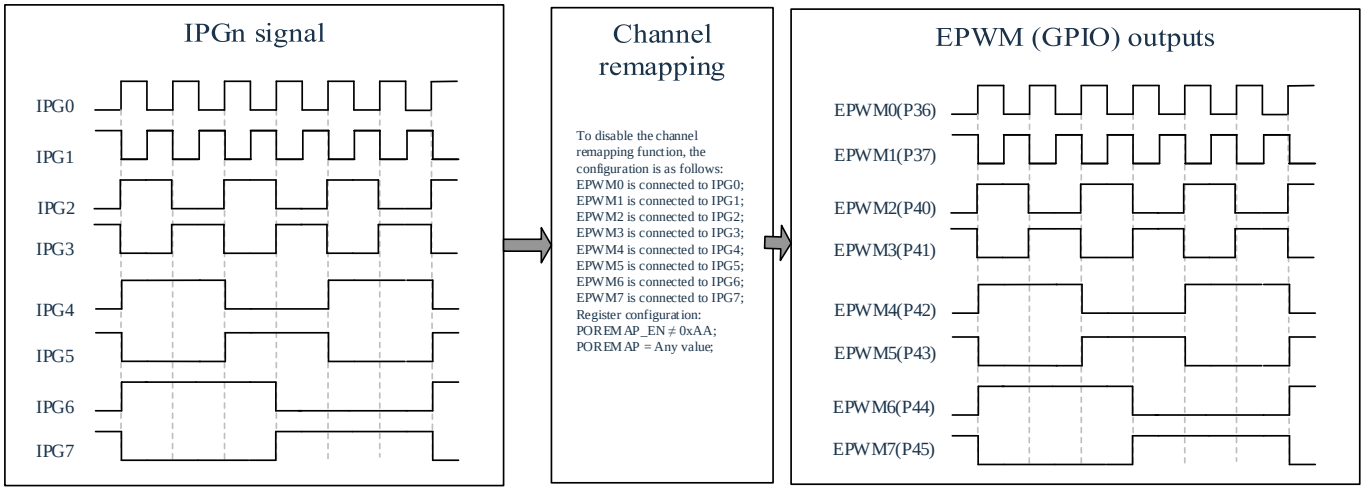
The block diagram of the output channel remapping function is shown below:

Notes: 1. IPGn (Internal Passage number) corresponds to the internal PWMn channels, where n=0~7.

2. PWMnRM refers to the control bit in the POREMAP register.

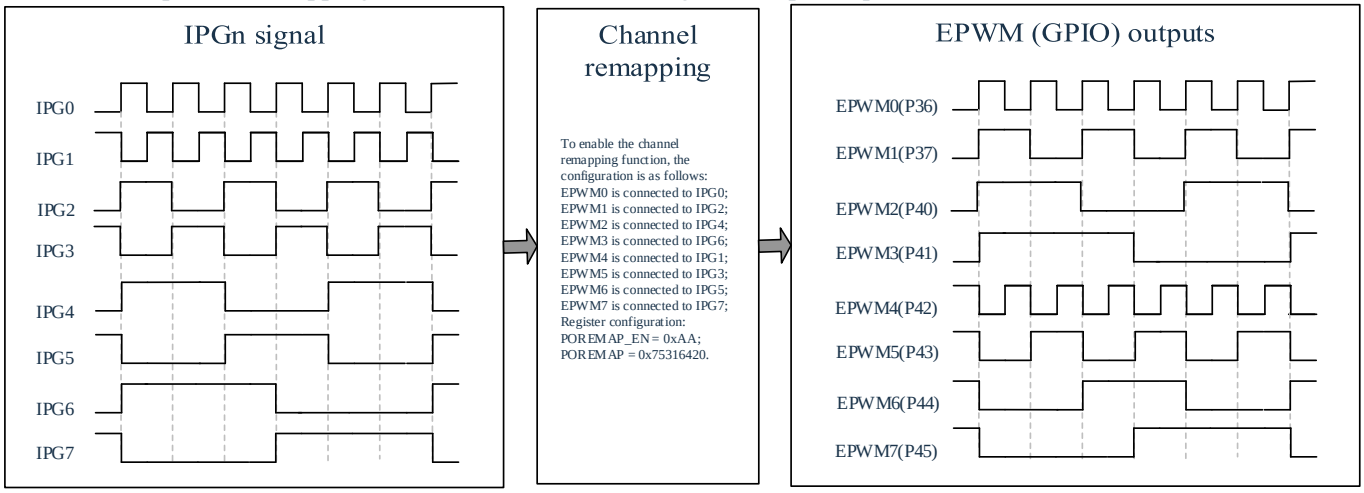


1. When the remapping function is not enabled (POREMAP_EN ≠ 0xAA), the output of EPWMn is not controlled by the PPREMAP register. EPWM0 is connected to IPG0, EPWM1 is connected to IPG1, EPWM2 is connected to IPG2, EPWM3 is connected to IPG3, EPWM4 is connected to IPG4, EPWM5 is connected to IPG5, EPWM6 is connected to IPG6, and EPWM7 is connected to IPG7.




2. When the remapping function is enabled (POREMAP_EN = 0xAA), the output of EPWMn is controlled by the POREMAP register. By setting the value of the PPREMAP register, EPWMn (n=0-7) can choose any one of IPG0/IPG1/IPG2/IPG3/IPG4/IPG5/IPG6/IPG7 for output.

For example, the remapping function can be set to change the output sequence.




3. The output channel remapping function only reallocates the port output channels; its internal control and interrupts are not remapped.

### 14.3.17 EPWM Configuration Process

- Enable EPWM register operation by writing 0x55 to the LOCK register.
- Configure EPWM clock division and set the pre-divider ratio and independent divider ratio.
- Select mode, independent mode or complementary mode.
- Set the EPWM period and duty cycle.
- Set the EPWM output polarity.
- Enable the EPWM counter.
- Configure the relevant IO ports as EPWM functionality.
- Enable the output of the corresponding EPWM channels.
- Write 0x00 to the LOCK register to prevent accidental operation on EPWM-related registers until the next operation on EPWM-related registers is needed, at which point re-enable it.

### 14.3.18 Interrupt

The EPWM unit has eight interrupt sources:

- ZIFn – An interrupt flag generated when the EPWM counter counts to zero.
- UIFn - An interrupt flag generated when the EPWM counter counts up to CMPDATn.
- PIFn - An interrupt flag generated during edge-aligned counting period point or center-aligned counting center point of the EPWM counter.
- DIFn - An interrupt flag generated when the EPWM counter counts down to CMPDATn/CMPDDATn.
- DC0IF - An interrupt flag generated when the EPWM counter counts to the value equal to CMPTGD0.
- DC1IF - An interrupt flag generated when the EPWM counter counts to the value equal to CMPTGD1.
- HALLIF - Hall state error interrupt flag.
- BRKIF - Fault interrupt flag.

All interrupt flags are set by hardware and must be cleared by software.

# 14.4Register Mapping

(EPWM base address = 0x4006_4200) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Descripton | Reset value |
|---|---|---|---|---|
| CLKPSC(P1B) | 0x000 | R/W | EPWM Prescaler Register | 0x00000000 |
| CLKDIV(P1B) | 0x004 | R/W | EPWM Clock Select Register | 0x00000000 |
| CON(P1B) | 0x008 | R/W | EPWM Control Register | 0x00000000 |
| CON2(P1B) | 0x00C | R/W | EPWM Control Register 2 | 0x00000000 |
| CON3(P1B) | 0x010 | R/W | EPWM Control Register 3 | 0x00000000 |
| CON4(P1B) | 0x014 | R/W | EPWM Control Register 4 | 0x00000000 |
| PERIOD0(P1A) | 0x018 | R/W | EPWM Period Register 0 | 0x00000000 |
| PERIOD1(P1A) | 0x01C | R/W | EPWM Period Register 1 | 0x00000000 |
| PERIOD2(P1A) | 0x020 | R/W | EPWM Period Register 2 | 0x00000000 |
| PERIOD3(P1A) | 0x024 | R/W | EPWM Period Register 3 | 0x00000000 |
| PERIOD4(P1A) | 0x028 | R/W | EPWM Period Register 4 | 0x00000000 |
| PERIOD5(P1A) | 0x02C | R/W | EPWM Period Register 5 | 0x00000000 |
| PERIOD6(P1A) | 0x030 | R/W | EPWM Period Register 6 | 0x00000000 |
| PERIOD7(P1A) | 0x034 | R/W | EPWM Period Register 7 | 0x00000000 |
| CMPDAT0(P1A) | 0x038 | R/W | EPWM Compare Register 0 | 0x00000000 |
| CMPDAT1(P1A) | 0x03C | R/W | EPWM Compare Register 1 | 0x00000000 |
| CMPDAT2(P1A) | 0x040 | R/W | EPWM Compare Register 2 | 0x00000000 |
| CMPDAT3(P1A) | 0x044 | R/W | EPWM Compare Register 3 | 0x00000000 |
| CMPDAT4(P1A) | 0x048 | R/W | EPWM Compare Register 4 | 0x00000000 |
| CMPDAT5(P1A) | 0x04C | R/W | EPWM Compare Register 5 | 0x00000000 |
| CMPDAT6(P1A) | 0x050 | R/W | EPWM Compare Register 6 | 0x00000000 |
| CMPDAT7(P1A) | 0x054 | R/W | EPWM Compare Register 7 | 0x00000000 |
| POREMAP_EN(P1B) | 0x058 | R/W | EPWM Output Channel Remapping Enable Register | 0x0000 |
| POREMAP(P1B) | 0x05C | R/W | EPWM Output Channel Remapping Register | 0x76543210 |
| POEN(P1B) | 0x060 | R/W | EPWM Output Control Register | 0x00000000 |
| BRKCTL(P1B) | 0x064 | R/W | EPWM Fault Protection Control Register | 0x00000000 |
| MASK(P1B) | 0x068 | R/W | EPWM Output Mask Register | 0x00000000 |
| MASKNXT(P1B) | 0x06C | R/W | EPWM Output Mask Preset Register | 0x00000000 |
| CMPTGD0(P1B) | 0x070 | R/W | EPWM Counter Compare Register 0 | 0x00000000 |
| CMPTGD1(P1B) | 0x074 | R/W | EPWM Counter Compare Register 1 | 0x00000000 |
| IMSC (P1B) | 0x078 | R/W | EPWM Interrupt Enable Register | 0x00000000 |
| IMSC2(P1B) | 0x07C | R/W | EPWM Interrupt Enable Register 2 | 0x00000000 |
| RIS | 0x080 | RO | EPWM Interrupt Source Status Register | 0x00000000 |
| RIS2 | 0x084 | RO | EPWM Interrupt Source Status Register 2 | 0x00000000 |
| MIS | 0x088 | RO | EPWM Enabled Interrupt Status Register | 0x00000000 |
| MIS2 | 0x08C | RO | EPWM Enabled Interrupt Status Register 2 | 0x00000000 |
| ICLR | 0x090 | WO | EPWM Interrupt Clear Register | 0x00000000 |
| ICLR2 | 0x094 | WO | EPWM Interrupt Clear Register 2 | 0x00000000 |
| IFA(P1B) | 0x098 | R/W | EPWM Interrupt Accumulation Control Register | 0x00000000 |
| LOCK | 0x09C | R/W | EPWM Write Enable Control Register | 0x00000000 |
| BRKRDT(P1B) | 0x0A0 | R/W | EPWM Fault Protection Recovery Delay Register | 0x00000000 |
| DTCTL01(P1B) | 0x0A4 | R/W | EPWM01 Deadzone Length Register | 0x00000000 |
| DTCTL23(P1B) | 0x0A8 | R/W | EPWM23 Deadzone Length Register | 0x00000000 |

| | | | | |
|---|---|---|---|---|
| DTCTL45(P1B) | 0x0AC | R/W | EPWM45 Deadzone Length Register | 0x00000000 |
| DTCTL67(P1B) | 0x0B0 | R/W | EPWM67 Deadzone Length Register | 0x00000000 |

Note 1: The registers marked with (P1A/P1B) are protected registers.

Note 2: (P1A): When LOCK==55H or AAH, the marked registers are writable; when it equals any other value, writing is prohibited.

Note 3: (P1B): When LOCK==55H, the marked registers are writable; when it equals any other value, writing is prohibited.

## 14.5Register Description

### 14.5.1 EPWM Prescaler Register (CLKPSC)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:24 | CLKPSC67 | EPWM counter 6 and 7 clock prescaler<br>CLK_PSC67 = PCLK/(CLKPSC67+1)<br>If CLKPSC67=0, the pre-scaler has no clock output. If the CLKDIVn bit selects a clock related to PSC, the counter does not operate. | 0x0 |
| 23:16 | CLKPSC45 | EPWM counter 4 and 5 clock prescaler<br>CLK_PSC45 = PCLK/(CLKPSC45+1)<br>If CLKPSC45=0, the pre-scaler has no clock output. If the CLKDIVn bit selects a clock related to PSC, the counter does not operate. | 0x0 |
| 15:8 | CLKPSC23 | EPWM counter 2 and 3 clock prescaler<br>CLK_PSC23 = PCLK/(CLKPSC23+1)<br>If CLKPSC23=0, the pre-scaler has no clock output. If the CLKDIVn bit selects a clock related to PSC, the counter does not operate. | 0x0 |
| 7:0 | CLKPSC01 | EPWM counter 0 and 1 clock prescaler<br>CLK_PSC01 = PCLK/(CLKPSC01+1)<br>If CLKPSC01=0, the pre-scaler has no clock output. If the CLKDIVn bit selects a clock related to PSC, the counter does not operate. | 0x0 |

## 14.5.2 EPWM Clock Selection Register (CLKDIV)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | - | Reserved | - |
| 30:28 | CLKDIV7 | Counter 7 clock division frequency selection<br>000: CLK_PSC67/2<br>001: CLK_PSC67/4<br>010: CLK_PSC67/8<br>011: CLK_PSC67/16<br>100: CLK_PSC67/1<br>Other value: PCLK | 0x0 |
| 27 | - | Reserved | - |
| 26: 24 | CLKDIV6 | Counter 6 clock division frequency selection<br>000: CLK_PSC67/2<br>001: CLK_PSC67/4<br>010: CLK_PSC67/8<br>011: CLK_PSC67/16<br>100: CLK_PSC67/1<br>Other value: PCLK | 0x0 |
| 23 | - | Reserved | - |
| 22:20 | CLKDIV5 | Counter 5 clock division frequency selection<br>000: CLK_PSC45/2<br>001: CLK_PSC45/4<br>010: CLK_PSC45/8<br>011: CLK_PSC45/16<br>100: CLK_PSC45/1<br>Other value: PCLK | 0x0 |
| 19 | - | Reserved | - |
| 18:16 | CLKDIV4 | Counter 4 clock division frequency selection<br>000: CLK_PSC45/2<br>001: CLK_PSC45/4<br>010: CLK_PSC45/8<br>011: CLK_PSC45/16<br>100: CLK_PSC45/1<br>Other value: PCLK | 0x0 |
| 15 | - | Reserved | - |
| 14:12 | CLKDIV3 | Counter 3 clock division frequency selection<br>000: CLK_PSC23/2<br>001: CLK_PSC23/4<br>010: CLK_PSC23/8<br>011: CLK_PSC23/16<br>100: CLK_PSC23/1<br>Other value: PCLK | 0x0 |
| 11 | - | Reserved | - |
| 10:8 | CLKDIV2 | Counter 2 clock division frequency selection<br>000: CLK_PSC23/2 | 0x0 |

| | | | |
|---|---|---|---|
| | | 001: CLK_PSC23/4<br>010: CLK_PSC23/8<br>011: CLK_PSC23/16<br>100: CLK_PSC23/1<br>Other value: PCLK | |
| 7 | - | Reserved | - |
| 6:4 | CLKDIV1 | Counter 1 clock division frequency selection<br>000: CLK_PSC01/2<br>001: CLK_PSC01/4<br>010: CLK_PSC01/8<br>011: CLK_PSC01/16<br>100: CLK_PSC01/1<br>Other value: PCLK | 0x0 |
| 3 | - | Reserved | - |
| 2:0 | CLKDIV0 | Counter 0 clock division frequency selection<br>000: CLK_PSC01/2<br>001: CLK_PSC01/4<br>010: CLK_PSC01/8<br>011: CLK_PSC01/16<br>100: CLK_PSC01/1<br>Other value: PCLK | 0x0 |

### 14.5.3 EPWM Control Register (CON)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:27 | - | Reserved | - |
| 26 | HALTMS | EPWMn channel status control bit during HALT (debug pause)<br>(If POENn=0, the output of EPWMn is in high resistance state)<br>0: All channels output normally (POENn=1)<br>1: All channels output brake data (POENn=1).<br>(In the debug state, the output of EPWMn is the brake data after running to a breakpoint/single step or pausing after operating the STOP button. ) | 0 |
| 25:24 | MODE | EPWM operating mode selection<br>00: Independent mode<br>01: Complementary mode<br>10: Synchronous mode<br>11: Reserved | 0x0 |
| 23 | GROUNPEN | EPWM group function enable bit<br>0: All PWM channels are independent of each other<br>1: EPWM0 controls EPWM2,EPWM4, and EPWM6.<br>EPWM1 controls EPWM3,EPWM5, and EPWM7 | 0 |
| 22 | ASYMEN | Asymmetric count enable in EPWM center alignment mode<br>0: Symmetric count enable<br>1: Asymmetric count enable | 0 |
| 21 | CNTTYPE | EPWM count alignment selection<br>0: Edge alignment<br>1: Center alignment | 0 |
| 20 | - | Reserved | - |
| 19 | EN_DT67 | EPWM counter 6 and 7 deadzone enable bit<br>0: Disable counter 6 and 7 deadzone<br>1: Enable counter 6 and 7 deadzone | 0 |
| 18 | EN_DT45 | EPWM counter 4 and 5 deadzone enable bit<br>0: Disable counter 4 and 5 deadzone<br>1: Enable counter 4 and 5 deadzone | 0 |
| 17 | EN_DT23 | EPWM counter 2 and 3 deadzone enable bit<br>0: Disable counter 2 and 3 deadzone<br>1: Enable counter 2 and 3 deadzone | 0 |
| 16 | EN_DT01 | EPWM counter 0 and 1 deadzone enable bit<br>0: Disable counter 0 and 1 deadzone<br>1: Enable counter 0 and 1 deadzone | 0 |
| 15 | PINV7 | EPWM7 output polarity control bit<br>0: Normal output<br>1: Inverted output | 0 |
| 14 | PINV6 | EPWM6 output polarity control bit<br>0: Normal output<br>1: Inverted output | 0 |
| 13 | PINV5 | EPWM5 output polarity control bit<br>0: Normal output<br>1: Inverted output | 0 |
| 12 | PINV4 | EPWM4 output polarity control bit<br>0: Normal output | 0 |

| | | | |
|---|---|---|---|
| | | 1: Inverted output | |
| 11 | PINV3 | EPWM3 output polarity control bit<br>0: Normal output<br>1: Inverted output | 0 |
| 10 | PINV2 | EPWM2 output polarity control bit<br>0: Normal output<br>1: Inverted output | 0 |
| 9 | PINV1 | EPWM1 output polarity control bit<br>0: Normal output<br>1: Inverted output | 0 |
| 8 | PINV0 | EPWM0 output polarity control bit<br>0: Normal output<br>1: Inverted output | 0 |
| 7 | CNTMODE7 | EPWM7 auto-load/one-shot mode<br>0: One-shot mode<br>1: Auto-load mode | 0 |
| 6 | CNTMODE6 | EPWM6 auto-load/one-shot mode<br>0: One-shot mode<br>1: Auto-load mode | 0 |
| 5 | CNTMODE5 | EPWM5 auto-load/one-shot mode<br>0: One-shot mode<br>1: Auto-load mode | 0 |
| 4 | CNTMODE4 | EPWM4 auto-load/one-shot mode<br>0: One-shot mode<br>1: Auto-load mode | 0 |
| 3 | CNTMODE3 | EPWM3 auto-load/one-shot mode<br>0: One-shot mode<br>1: Auto-load mode | 0 |
| 2 | CNTMODE2 | EPWM2 auto-load/one-shot mode<br>0: One-shot mode<br>1: Auto-load mode | 0 |
| 1 | CNTMODE1 | EPWM1 auto-load/one-shot mode<br>0: One-shot mode<br>1: Auto-load mode | 0 |
| 0 | CNTMODE0 | EPWM0 auto-load/one-shot mode<br>0: One-shot mode<br>1: Auto-load mode | 0 |

### 14.5.4 EPWM Control Register (CON2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7 | CNTEN7 | EPWM7 counter enable bit<br>0: Disable<br>1: Enable<br>(The bit is cleared automatically after one-shot mode completion) | 0 |
| 6 | CNTEN6 | EPWM6 counter enable bit<br>0: Disable<br>1: Enable<br>(The bit is cleared automatically after one-shot mode completion) | 0 |
| 5 | CNTEN5 | EPWM5 counter enable bit<br>0: Disable<br>1: Enable<br>(The bit is cleared automatically after one-shot mode completion) | 0 |
| 4 | CNTEN4 | EPWM4 counter enable bit<br>0: Disable<br>1: Enable<br>(The bit is cleared automatically after one-shot mode completion) | 0 |
| 3 | CNTEN3 | EPWM3 counter enable bit<br>0: Disable<br>1: Enable<br>(The bit is cleared automatically after one-shot mode completion) | 0 |
| 2 | CNTEN2 | EPWM2 counter enable bit<br>0: Disable<br>1: Enable<br>(The bit is cleared automatically after one-shot mode completion) | 0 |
| 1 | CNTEN1 | EPWM1 counter enable bit<br>0: Disable<br>1: Enable<br>(The bit is cleared automatically after one-shot mode completion) | 0 |
| 0 | CNTEN0 | EPWM0 counter enable bit<br>0: Disable<br>1: Enable<br>(The bit is cleared automatically after one-shot mode completion) | 0 |

### 14.5.5 EPWM Control Register (CON3)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:30 | LOADTYP7 | EPWM7 load/interrupt mode selection bit<br>00: Interrupt flags are loaded and generated at each zero and period point.<br>01: Interrupt flags are loaded and generated at each zero point.<br>10: Interrupt flags are loaded and generated alternately at the first zero and the next period point.<br>11: Interrupt flags are generated every two zero points | 0x0 |
| 29:28 | LOADTYP6 | EPWM6 load/interrupt mode selection bit<br>00: Interrupt flags are loaded and generated at each zero and period point.<br>01: Interrupt flags are loaded and generated at each zero point.<br>10: Interrupt flags are loaded and generated alternately at the first zero and the next period point.<br>11: Interrupt flags are generated every two zero points | 0x0 |
| 27:26 | LOADTYP5 | EPWM5 load/interrupt mode selection bit<br>00: Interrupt flags are loaded and generated at each zero and period point.<br>01: Interrupt flags are loaded and generated at each zero point.<br>10: Interrupt flags are loaded and generated alternately at the first zero and the next period point.<br>11: Interrupt flags are generated every two zero points | 0x0 |
| 25:24 | LOADTYP4 | EPWM4 load/interrupt mode selection bit<br>00: Interrupt flags are loaded and generated at each zero and period point.<br>01: Interrupt flags are loaded and generated at each zero point.<br>10: Interrupt flags are loaded and generated alternately at the first zero and the next period point.<br>11: Interrupt flags are generated every two zero points | 0x0 |
| 23:22 | LOADTYP3 | EPWM3 load/interrupt mode selection bit<br>00: Interrupt flags are loaded and generated at each zero and period point.<br>01: Interrupt flags are loaded and generated at each zero point.<br>10: Interrupt flags are loaded and generated alternately at the first zero and the next period point.<br>11: Interrupt flags are generated every two zero points | 0x0 |
| 21:20 | LOADTYP2 | EPWM2 load/interrupt mode selection bit<br>00: Interrupt flags are loaded and generated at each zero and period point.<br>01: Interrupt flags are loaded and generated at each zero point.<br>10: Interrupt flags are loaded and generated alternately at the first zero and the next period point.<br>11: Interrupt flags are generated every two zero points | 0x0 |
| 19:18 | LOADTYP1 | EPWM1 load/interrupt mode selection bit<br>00: Interrupt flags are loaded and generated at each zero and period point.<br>01: Interrupt flags are loaded and generated at each zero point.<br>10: Interrupt flags are loaded and generated alternately at the first zero and the next period point.<br>11: Interrupt flags are generated every two zero points | 0x0 |
| 17:16 | LOADTYP0 | EPWM0 load/interrupt mode selection bit | 0x0 |

| | | | |
|---|---|---|---|
| | | 00: Interrupt flags are loaded and generated at each zero and period point.<br>01: Interrupt flags are loaded and generated at each zero point.<br>10: Interrupt flags are loaded and generated alternately at the first zero and the next period point.<br>11: Interrupt flags are generated every two zero points | |
| 15 | LOADEN7 | EPWM7 period/comparator load enable<br>0: Disable<br>1: Enable (Automatically cleared by hardware after loading) | 0 |
| 14 | LOADEN6 | EPWM6 period/comparator load enable<br>0: Disable<br>1: Enable (Automatically cleared by hardware after loading) | 0 |
| 13 | LOADEN5 | EPWM5 period/comparator load enable<br>0: Disable<br>1: Enable (Automatically cleared by hardware after loading) | 0 |
| 12 | LOADEN4 | EPWM4 period/comparator load enable<br>0: Disable<br>1: Enable (Automatically cleared by hardware after loading) | 0 |
| 11 | LOADEN3 | EPWM3 period/comparator load enable<br>0: Disable<br>1: Enable (Automatically cleared by hardware after loading) | 0 |
| 10 | LOADEN2 | EPWM2 period/comparator load enable<br>0: Disable<br>1: Enable (Automatically cleared by hardware after loading) | 0 |
| 9 | LOADEN1 | EPWM1 period/comparator load enable<br>0: Disable<br>1: Enable (Automatically cleared by hardware after loading) | 0 |
| 8 | LOADEN0 | EPWM0 period/comparator load enable<br>0: Disable<br>1: Enable (Automatically cleared by hardware after loading) | 0 |
| 7 | CNTCLR7 | EPWM7 counter clear bit<br>0: Disable<br>1: Enable (Automatically cleared by hardware) | 0 |
| 6 | CNTCLR6 | EPWM6 counter clear bit<br>0: Disable<br>1: Enable (Automatically cleared by hardware) | 0 |
| 5 | CNTCLR5 | EPWM5 counter clear bit<br>0: Disable<br>1: Enable (Automatically cleared by hardware) | 0 |
| 4 | CNTCLR4 | EPWM4 counter clear bit<br>0: Disable<br>1: Enable (Automatically cleared by hardware) | 0 |
| 3 | CNTCLR3 | EPWM3 counter clear bit<br>0: Disable<br>1: Enable (Automatically cleared by hardware) | 0 |
| 2 | CNTCLR2 | EPWM2 counter clear bit<br>0: Disable<br>1: Enable (Automatically cleared by hardware) | 0 |
| 1 | CNTCLR1 | EPWM1 counter clear bit | 0 |

| | | | |
|---|---|---|---|
| | | 0: Disable<br>1: Enable (Automatically cleared by hardware) | |
| 0 | CNTCLR0 | EPWM0 counter clear bit<br>0: Disable<br>1: Enable (Automatically cleared by hardware) | 0 |

Note 1: ADCCON3[31:16] describes the EPWMn load/interrupt mode selection in center-aligned mode. The EPWMn load/interrupt mode selection in edge-aligned mode can be found in Section 14.3.7.

## 14.5.6 EPWM Control Register (CON4)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3 | LOADNWINT | EPWM load and interrupt flag related control bit<br>1: The load control and interrupt flags are unrelated<br>0: The control and interrupt flags are related<br>When loading control and interrupt flags are related, the interrupt flag will be generated after the loading action, depending on whether the loading produces an interrupt flag. If they are unrelated, the interrupt flag will be generated every cycle and at zero points. | 0 |
| 2 | LETGHALL | HALL state triggers the LOADENn enable bit<br>0: Disable<br>1: Enable when HALL state changes, triggering LOADENn = 1.<br>Note: If the HALL detection state changes, it will set the load enable bits for EPWM0-EPWM7 to 1. | 0 |
| 1 | LETGACMP1 | ACMP1 triggers the LOADENn enable bit<br>0: Disable<br>1: Enable when ACMP1 triggers, setting LOADENn = 1<br>Note: If an ACMP1 event occurs, it will set the load enable bits for EPWM0-EPWM7 to 1. | 0 |
| 0 | LETGACMP0 | ACMP0 triggers the LOADENn enable bit<br>0: Disable<br>1: Enable when ACMP0 triggers, setting LOADENn = 1.<br>Note: If an ACMP0 event occurs, it will set the load enable bits for EPWM0-EPWM7 to 1. | 0 |

## 14.5.7 EPWM Period Register 0-7 (PERIOD0-7)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Reserved | - |
| 15:0 | PERIODn | EPWMn counter period value | 0x0 |

## 14.5.8 EPWM Compare Register 0-7 (CMPDAT0-7)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | CMPDDATn | EPWMn counter down compare value | 0x0 |
| 15:0 | CMPDATn | EPWMn counter compare value | 0x0 |

## 14.5.9 EPWM Output Control Register (POEN)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:12 | - | Reserved | - |
| 11 | MASKLE | EPWM mask control preset data load enable bit<br>0: Disable<br>1: Enable<br>(Enable MASKNXT register to load data into MASK register, and disable writing to MASK register. In addition, when this bit is set to 1, the mask data is not loaded immediately, but only when the corresponding load point is reached.) | 0 |
| 10:8 | MASKLS | EPWM mask control data load time select bit<br>000: Load at the EPWM0 load point<br>001: Load at the EPWM1 load point<br>010: Load at the EPWM2 load point<br>011: Load at the EPWM3 load point<br>100: Load at the EPWM4 load point<br>101: Load at the EPWM5 load point<br>110: Load at the EPWM6 load point<br>111: Load at the EPWM7 load point | 0x0 |
| 7:0 | POENn | EPWMn output enable bit<br>0: EPWM channel n output disable<br>1: EPWM channel n output enable | 0x0 |

## 14.5.10 EPWM Output Channel Remap Enable Register 1 (POREMAP_EN)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 16:8 | - | Reserved | - |
| 7:0 | PWMRMEN | EPWM channel remap function enable control<br>AAH: Remap function enable<br>EPWMn channel output is selected by PWMnRM<br>Other: Remap function disable<br>EPWMn fix channel outputs:<br>EPWM0<- IPG0<br>EPWM1<- IPG1<br>EPWM2<- IPG2<br>EPWM3<- IPG3<br>EPWM4<- IPG4<br>EPWM5<- IPG5<br>EPWM6<- IPG6<br>EPWM7<- IPG7 | 0x0 |

## 14.5.11 EPWM Output Channel Remap Register (POREMAP)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | - | Reserved | - |
| 30:28 | PWM7RM | EPWM channel 7 remap selection bit<br>000: Map the output of IPG0<br>001: Map the output of IPG1<br>010: Map the output of IPG2<br>011: Map the output of IPG3<br>100: Map the output of IPG4<br>101: Map the output of IPG5<br>110: Map the output of IPG6<br>111: Map the output of IPG7 | 0x7 |
| 27 | - | Reserved | - |
| 26:24 | PWM6RM | EPWM channel 6 remap selection bit<br>000: Map the output of IPG0<br>001: Map the output of IPG1<br>010: Map the output of IPG2<br>011: Map the output of IPG3<br>100: Map the output of IPG4<br>101: Map the output of IPG5<br>110: Map the output of IPG6<br>111: Map the output of IPG7 | 0x6 |
| 23 | - | Reserved | - |
| 22:20 | PWM5RM | EPWM channel 5 remap selection bit<br>000: Map the output of IPG0<br>001: Map the output of IPG1<br>010: Map the output of IPG2<br>011: Map the output of IPG3<br>100: Map the output of IPG4<br>101: Map the output of IPG5<br>110: Map the output of IPG6<br>111: Map the output of IPG7 | 0x5 |
| 19 | - | Reserved | - |
| 18:16 | PWM4RM | EPWM channel 4 remap selection bit<br>000: Map the output of IPG0<br>001: Map the output of IPG1<br>010: Map the output of IPG2<br>011: Map the output of IPG3<br>100: Map the output of IPG4<br>101: Map the output of IPG5<br>110: Map the output of IPG6<br>111: Map the output of IPG7 | 0x4 |
| 15 | - | Reserved | - |
| 14:12 | PWM3RM | EPWM channel 3 remap selection bit<br>000: Map the output of IPG0<br>001: Map the output of IPG1<br>010: Map the output of IPG2 | 0x3 |

| | | | |
|---|---|---|---|
| | | 011: Map the output of IPG3<br>100: Map the output of IPG4<br>101: Map the output of IPG5<br>110: Map the output of IPG6<br>111: Map the output of IPG7 | |
| 11 | - | Reserved | - |
| 10:8 | PWM2RM | EPWM channel 2 remap selection bit<br>000: Map the output of IPG0<br>001: Map the output of IPG1<br>010: Map the output of IPG2<br>011: Map the output of IPG3<br>100: Map the output of IPG4<br>101: Map the output of IPG5<br>110: Map the output of IPG6<br>111: Map the output of IPG7 | 0x2 |
| 7 | - | Reserved | - |
| 6:4 | PWM1RM | EPWM channel 1 remap selection bit<br>000: Map the output of IPG0<br>001: Map the output of IPG1<br>010: Map the output of IPG2<br>011: Map the output of IPG3<br>100: Map the output of IPG4<br>101: Map the output of IPG5<br>110: Map the output of IPG6<br>111: Map the output of IPG7 | 0x1 |
| 3 | - | Reserved | - |
| 2:0 | PWM0RM | EPWM channel 0 remap selection bit<br>000: Map the output of IPG0<br>001: Map the output of IPG1<br>010: Map the output of IPG2<br>011: Map the output of IPG3<br>100: Map the output of IPG4<br>101: Map the output of IPG5<br>110: Map the output of IPG6<br>111: Map the output of IPG7 | 0x0 |

## 14.5.12 EPWM Fault Protection Control Register (BRKCTL)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | BRKEN | EPWM fault protection function general enable bit<br>0: Disable (reset fault protection circuit)<br>1: Enable | 0 |
| 30 | BRKAF | EPWM fault signal flag bit (read-only)<br>0: No fault is generated<br>1: A fault signal is generated or the brake signal remains valid | 0 |
| 29:28 | BRKMS | EPWM fault protection mode selection bit<br>00: Stop mode<br>01: Pause mode<br>10: Recovery mode<br>11: Delayed recovery mode<br>Note: When switching to the fault protection mode, the fault protection enable must be disabled first, then switch to the fault protection mode, and finally, enable the fault protection general enable bit. | 0x0 |
| 27 | BRKCLR | EPWM fault protection clear bit (write-only)<br>0: --<br>1: Clear the fault protection status<br>Note: only when BRKAF=1 can write 1 to perform fault clear operation, otherwise the operation is invalid. | 0 |
| 26:24 | BRKRCS | EPWM fault recovery load point selection bit<br>000: EPWM0 load point recovery<br>001: EPWM1 load point recovery<br>010: EPWM2 load point recovery<br>011: EPWM3 load point recovery<br>100: EPWM4 load point recovery<br>101: EPWM5 load point recovery<br>110: EPWM6 load point recovery<br>111: EPWM7 load point recovery | 0x0 |
| 23 | ACMP1BKLE | Analog comparator 1 output level control brake enable bit<br>0: Disable<br>1: Enable | 0 |
| 22 | ACMP1BKLS | Analog comparator 1 output level control brake selection bit<br>0: Low level generates brake<br>1: High level generates brake | 0 |
| 21 | ACMP0BKLE | Analog comparator 0 output level control brake enable bit<br>0: Disable<br>1: Enable | 0 |
| 20 | ACMP0BKLS | Analog comparator 0 output level control brake selection bit<br>0: Low level generates brake<br>1: High level generates brake | 0 |
| 19 | ACMP1BKEN | Analog comparator 1 output event control brake enable bit<br>0: Disable<br>1: Enable<br>(Comparator output event refers to generating rising edge/falling edge/double edges, which can be selected in ACMP->CEVCON) | 0 |
| 18 | ACMP0BKEN | Analog comparator 0 output event control brake enable bit | 0 |

| | | | |
|---|---|---|---|
| | | 0: Disable<br>1: Enable<br>(Comparator output event refers to generating rising edge/falling edge/double edges, which can be selected in ACMP->CEVCON) | |
| 17 | ADCMP1BKEN | ADC comparator 1 output brake enable bit<br>0: Disable<br>1: Enable | 0 |
| 16 | ADCMP0BKEN | ADC comparator 0 output brake enable bit<br>0: Disable<br>1: Enable | 0 |
| 15:14 | - | Reserved | - |
| 13 | BRKOSF | EPWM fault protection output status flag bit (read-only)<br>0: EPWMn channel is in normal output state<br>1: EPWMn channel is in output BRKODn data state | 0 |
| 12 | SWBRK | Software brake enable bit<br>0: Disable software brake<br>1: Immediately generate software brake | 0 |
| 11 | EXTBRKEE | External hardware brake edge detection enable bit<br>0: Disable<br>1: Enable | 0 |
| 10 | EXTBRKES | External hardware brake edge detection selection bit<br>0: Falling edge triggers brake<br>1: Rising edge triggers brake | 0 |
| 9 | EXTBRKLE | External hardware brake level detection enable bit<br>0: Disable<br>1: Enable | 0 |
| 8 | EXTBRKLS | External hardware brake level detection selection bit<br>0: Low level generates brake<br>1: High level generates brake | 0 |
| 7:0 | BRKODn | EPWMn brake output level selection bit<br>0: After fault brake, channel n outputs low level<br>1: After fault brake, channel n outputs high level | 0x0 |

## 14.5.13 EPWM Dead Zone Length Register (DTCTL01)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | DTI01L | Channel 0 and Channel 1 left deadzone length register<br>Dead time = PWM_CLK01 × DTI01L | 0x0 |
| 15:0 | DTI01R | Channel 0 and Channel 1 right deadzone length register<br>Dead time = PWM_CLK01 × DTI01R | 0x0 |

## 14.5.14 EPWM Dead Zone Length Register (DTCTL23)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | DTI23L | Channel 2 and Channel 3 left deadzone length register<br>Dead time = PWM_CLK23 × DTI23L | 0x0 |
| 15:0 | DTI23R | Channel 2 and Channel 3 right deadzone length register<br>Dead time = PWM_CLK23 × DTI23R | 0x0 |

## 14.5.15 EPWM Dead Zone Length Register (DTCTL45)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | DTI45L | Channel 4 and Channel 5 left deadzone length register<br>Dead time = PWM_CLK45 × DTI45L | 0x0 |
| 15:0 | DTI45R | Channel 4 and Channel 5 right deadzone length register<br>Dead time = PWM_CLK45 × DTI45R | 0x0 |

## 14.5.16 EPWM Dead Zone Length Register (DTCTL67)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | DTI67L | Channel 6 and Channel 7 left deadzone length register<br>Dead time = PWM_CLK67 × DTI67L | 0x0 |
| 15:0 | DTI67R | Channel 6 and Channel 7 right deadzone length register<br>Dead time = PWM_CLK67 × DTI67R | 0x0 |

## 14.5.17 EPWM Mask Output Control Register (MASK)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Reserved | - |
| 15 | MASKEN7 | EPWM7 mask output enable bit<br>0: Disable<br>1: Enable | 0 |
| 14 | MASKEN6 | EPWM6 mask output enable bit<br>0: Disable<br>1: Enable | 0 |
| 13 | MASKEN5 | EPWM5 mask output enable bit<br>0: Disable<br>1: Enable | 0 |
| 12 | MASKEN4 | EPWM4 mask output enable bit<br>0: Disable<br>1: Enable | 0 |
| 11 | MASKEN3 | EPWM3 mask output enable bit<br>0: Disable<br>1: Enable | 0 |
| 10 | MASKEN2 | EPWM2 mask output enable bit<br>0: Disable<br>1: Enable | 0 |
| 9 | MASKEN1 | EPWM1 mask output enable bit<br>0: Disable<br>1: Enable | 0 |
| 8 | MASKEN0 | EPWM0 mask output enable bit<br>0: Disable<br>1: Enable | 0 |
| 7 | MASKD7 | EPWM7 mask data<br>0: Output 0<br>1: Output 1 | 0 |
| 6 | MASKD6 | EPWM6 mask data<br>0: Output 0<br>1: Output 1 | 0 |
| 5 | MASKD5 | EPWM5 mask data<br>0: Output 0<br>1: Output 1 | 0 |
| 4 | MASKD4 | EPWM4 mask data<br>0: Output 0<br>1: Output 1 | 0 |
| 3 | MASKD3 | EPWM3 mask data<br>0: Output 0<br>1: Output 1 | 0 |
| 2 | MASKD2 | EPWM2 mask data<br>0: Output 0<br>1: Output 1 | 0 |
| 1 | MASKD1 | EPWM1 mask data<br>0: Output 0 | 0 |

| | | | |
|---|---|---|---|
| | | 1: Output 1 | |
| 0 | MASKD0 | EPWM0 mask data<br>0: Output 0<br>1: Output 1 | 0 |

## 14.5.18 EPWM Mask Output Control Preset Register (MASKNXT)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:25 | - | Reserved | - |
| 24 | HALLEN | HALL detection mode enable bit<br>0: Disable<br>1: Enable | 0 |
| 23 | HALLCLR | HALL error status clear bit<br>0: Writing 0 is invalid<br>1: Writing 1 clears the HALLST error status and resets it to the initial state 000.<br>Reading as 0.<br>Note 1: If an error state or sequence occurs when HALLST = 111, the HALL detection function stops. To enable HALL status again, write 1 to clear the 111 state. | 0 |
| 22:20 | HALLST | HALL interface status bit (read-only)<br>Detect the state corresponding to {CAP2,CAP1,CAP0}<br>000: State 0 (initial state)<br>001: State 1<br>010: State 2<br>011: State 3<br>100: State 4<br>101: State 5<br>110: State 6<br>111: Error state<br>Note 1: This state indicates the internal detection of the HALL interface in the chip, which can be used to determine if a valid state has been entered. If there are errors in the states of the three HALL sensors or errors in the order of the states, this status bit will be set to 111.<br>Valid sequence 1: …6-2-3-1-5-4-6-…<br>Valid sequence 2: …6-4-5-1-3-2-6-…<br>Note 2: Under a valid state bit, if the enable mask preset data loading function is enabled, the corresponding mask preset cache data is loaded into the MASK register at the loading point. For example, when the HALL detection changes to state 3, the data of mask preset cache 3 is loaded into the MASK register at the first loading point after entering state 3.<br>Note 3: Output the data of mask preset cache 7 in the initial state 000 or error state 111. | 0x0 |
| 19 | - | Reserved | - |
| 18:16 | PMASKSEL | Mask preset cache selection bit;<br>000: Select mask preset cache 0<br>001: Select mask preset cache 1<br>010: Select mask preset cache 2<br>011: Select mask preset cache 3<br>100: Select mask preset cache 4<br>101: Select mask preset cache 5<br>110: Select mask preset cache 6<br>111: Select mask preset cache 7<br>Note 1: This selection bit affects the read and write of the lower 16 bits of data. There are 8 mask preset caches inside the EPWM.<br>000: The lower 16 bits of this register are read and written as the data in mask cache 0 | 0x0 |

| | | | |
|---|---|---|---|
| | | 001: The lower 16 bits of this register are read and written as the data in mask cache 1<br>110: The lower 16 bits of this register are read and written as the data in mask cache 6.<br>Note 2: When HALLEN = 0, the data in mask preset cache 0 is loaded by default. | |
| 15 | PMASKEN7 | EPWM7 mask output enable preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 14 | PMASKEN6 | EPWM6 mask output enable preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 13 | PMASKEN5 | EPWM5 mask output enable preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 12 | PMASKEN4 | EPWM4 mask output enable preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 11 | PMASKEN3 | EPWM3 mask output enable preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 10 | PMASKEN2 | EPWM2 mask output enable preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 9 | PMASKEN1 | EPWM1 mask output enable preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 8 | PMASKEN0 | EPWM0 mask output enable preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 7 | PMASKD7 | EPWM7 mask data preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 6 | PMASKD6 | EPWM6 mask data preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 5 | PMASKD5 | EPWM5 mask data preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 4 | PMASKD4 | EPWM4 mask data preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 3 | PMASKD3 | EPWM3 mask data preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 2 | PMASKD2 | EPWM2 mask data preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 1 | PMASKD1 | EPWM1 mask data preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |
| 0 | PMASKD0 | EPWM0 mask data preset bit<br>(This bit can be set to load into the MASK register at the loading point of EPWMn) | 0 |

## 14.5.19 EPWM Trigger Compare Register (CMPTGD0-1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:20 | - | Reserved | - |
| 19 | CMPTGDSn | EPWM count comparator n trigger mode<br>(Valid in center-aligned counting mode)<br>0: Triggered on count down<br>1: Triggered on count up | 0 |
| 18:16 | CMPPCHSn | EPWM digital comparator n compare channel selection<br>000: PWM0 counter<br>001: PWM1 counter<br>010: PWM2 counter<br>011: PWM3 counter<br>100: PWM4 counter<br>101: PWM5 counter<br>110: PWM6 counter<br>111: PWM7 counter | 0x0 |
| 15:0 | CMPTGDn | EPWM count comparator n trigger compare value | 0x0 |

## 14.5.20 EPWM Interrupt Enable Register (IMSC)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:24 | EN_DIFn<br>n=7-0 | EPWMn downward compare interrupt enable bit<br>0: Disable<br>1: Enable | 0x0 |
| 23:16 | EN_UIFn<br>n=7-0 | EPWMn upward compare interrupt enable bit<br>0: Disable<br>1: Enable | 0x0 |
| 15:8 | EN_PIFn<br>n=7-0 | EPWMn period interrupt enable bit<br>0: Disable<br>1: Enable | 0x0 |
| 7:0 | EN_ZIFn<br>n=7-0 | EPWMn zero interrupt enable bit<br>0: Disable<br>1: Enable | 0x0 |

## 14.5.21 EPWM Interrupt Enable Register 2(IMSC2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3 | EN_BRKIF | EPWM fault interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 2 | EN_HALLIF | HALL status error interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 1 | EN_DC1IF | Count comparator 1 interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 0 | EN_DC0IF | Count comparator 0 interrupt enable bit<br>0: Disable<br>1: Enable | 0 |

## 14.5.22 EPWM Interrupt Source Status Register (RIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:24 | RIS_DIFn<br>n=7-0 | EPWMn downward compare interrupt source status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0x0 |
| 23:16 | RIS_UIFn<br>n=7-0 | EPWMn upward compare interrupt source status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0x0 |
| 15:8 | RIS_PIFn<br>n=7-0 | EPWMn period interrupt source status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0x0 |
| 7:0 | RIS_ZIFn<br>n=7-0 | EPWMn zero point interrupt source status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0x0 |

## 14.5.23 EPWM Interrupt Source Status Register 2 (RIS2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3 | RIS_BRKIF | EPWM fault interrupt source status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0 |
| 2 | RIS_HALLIF | HALL state error interrupt source status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0 |
| 1 | RIS_DC1IF | Counting comparator 1 interrupt status bit<br>0: Disable<br>1: Enable | 0 |
| 0 | RIS_DC0IF | Counting comparator 0 interrupt status bit<br>0: Disable<br>1: Enable | 0 |

## 14.5.24 EPWM Enabled Interrupt Status Register (MIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:24 | MIS_DIFn<br>n=7-0 | EPWMn downward compare enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is enabled and generated | 0x0 |
| 23:16 | MIS_UIFn<br>n=7-0 | EPWMn upward compare enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is enabled and generated | 0x0 |
| 15:8 | MIS_PIFn<br>n=7-0 | EPWMn period enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is enabled and generated | 0x0 |
| 7:0 | MIS_ZIFn<br>n=7-0 | EPWMn zero point enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is enabled and generated | 0x0 |

## 14.5.25 EPWM Enabled Interrupt Status Register 2(MIS2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3 | MIS_BRKIF | EPWM fault enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is enabled and generated | 0 |
| 2 | MIS_HALLIF | HALL state error enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is enabled and generated | 0 |
| 1 | RIS_DC1IF | Counting comparator 1 enabled interrupt status bit<br>0: Disable<br>1: Enable | 0 |
| 0 | RIS_DC0IF | Counting comparator 0 enabled interrupt status bit<br>0: Disable<br>1: Enable | 0 |

## 14.5.26 EPWM Interrupt Clear Control Register (ICLR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:24 | ICLR_DIFn<br>n=7-0 | EPWMn downward compare interrupt clear control bit<br>0: No effect<br>1: Clear RIS_DIFn flag bit | 0x0 |
| 23:16 | ICLR_UIFn<br>n=7-0 | EPWMn upward compare interrupt clear control bit<br>0: No effect<br>1: Clear RIS_UIFn flag bit | 0x0 |
| 15:8 | ICLR_PIFn<br>n=7-0 | EPWMn period interrupt clear control bit<br>0: No effect<br>1: Clear RIS_PIFn flag bit | 0x0 |
| 7:0 | ICLR_ZIFn<br>n=7-0 | EPWMn zero point interrupt clear control bit<br>0: No effect<br>1: Clear RIS_ZIFn flag bit | 0x0 |

## 14.5.27 EPWM Interrupt Clear Control Register 2(ICLR2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3 | ICLR_BRKIF | EPWM fault interrupt clear control bit<br>0: No effect<br>1: Clear RIS_BRKIF flag bit | 0 |
| 2 | ICLR_HALLIF | HALL state error interrupt clear control bit<br>0: No effect<br>1: Clear RIS_HALLIF flag bit<br>Note: RIS_HALLIF flag cannot be cleared if HALLST=111 | 0 |
| 1 | ICLR_DC1IF | Counting comparator 1 interrupt clear control bit<br>0: No effect<br>1: Clear RIS_DC1IF flag bit | 0 |
| 0 | ICLR_DC0IF | Counting comparator 0 interrupt clear control bit<br>0: No effect<br>1: Clear RIS_DC0IF flag bit | 0 |

## 14.5.28 EPWM Interrupt Accumulation Control Register (IFA)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Reserved | - |
| 15:12 | BRKIFCMP | Accumulated compare value for fault protection interrupt<br>When the fault interrupt accumulates to (BRKIFCMP+1), set the BRKIF interrupt flag bit to 1. | 0x0 |
| 11:9 | - | Reserved | - |
| 8 | BRKIFAEN | Fault protection interrupt accumulation enable bit<br>0: Disable<br>1: Enable | 0 |
| 7:4 | ZIFCMP | Zero point interrupt accumulation compare value<br>When the zero point interrupt of the corresponding channel accumulates to (ZIFCMP+1), set the ZIFn interrupt flag bit to 1(all channels se to the same compare value) | 0x0 |
| 3:1 | - | Reserved | - |
| 0 | ZIFAEN | Zero point interrupt accumulation enable bit<br>0: Disable<br>1: Enable | 0 |

## 14.5.29 EPWM Write Enable Control Register (LOCK)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | LOCK | When LOCK=0x55, enable the operation of other EPWM registers;<br>When LOCK=0xaa, only enable the operation of the EPWM period register and the compare register.<br>When LOCK=other values, disable the operation of EPWM related registers. | 0x0 |

## 14.5.30 EPWM Fault Protection Recovery Delay Register (BRKRDT)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:20 | - | Set to 0. | 0x0 |
| 19:16 | FILS | Fault protection (brake) signal filter time selection bit<br>0000: (0~1)* TPCLK<br>0001: (1~2)* TPCLK<br>0010: (2~3)* TPCLK<br>0011: (4~5)* TPCLK<br>0100: (8~9)* TPCLK<br>0101: (16~17)* TPCLK<br>0110: (24~25)* TPCLK<br>0111: (32~33)* TPCLK<br>1000: (48~49)* TPCLK<br>1001: (64~65)* TPCLK<br>1010: (80~81)* TPCLK<br>1011: (96~97)* TPCLK<br>1100: (112~113)* TPCLK<br>Other: (0~1)* TPCLK | 0x0 |

| | | | |
|---|---|---|---|
| 15:0 | RDT | Fault protection recovery delay (only available in delayed recovery mode)<br>Delay time = (RDT+1)*$T_{PCLK}$ | 0x0 |

# Chapter 15 Universal Asynchronous Receiver Transmitter (UARTn n=0/1)

It contains 2 universal asynchronous serial interfaces.

## 15.1Features

- Full duplex, asynchronous communication.
- Programmable serial interface features.
  - Data bit length can be set to 5-8 bits.
  - Parity bit can be set to odd, even, no parity, or fixed parity generation and detection.
  - Stop bit length can be set to 1, 1.5, or 2 bits.

## 15.2Function Description

### 15.2.1 UARTn Function Mode

UART is a full-duplex asynchronous communication interface. The UART transceiver contains a buffer for both transmitting and receiving, and the byte length and stop bit length can be flexibly set. Communication parameters for the full-duplex serial interface can be configured.

### 15.2.2 UARTn Interrupt and Status

UART supports three types of interrupts, including:

- Line status interrupts (parity check error, frame error, break interrupt).
- Receive data valid interrupt.
- Transmit holding register null interrupt.

# 15.3Register Mapping

(UART0 base address = 0x4006_4000; UART1 base address = 0x4006_4100)

RO: read only; WO: write only; R/W: read/write.

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| RBR | 0x000 | RO | Receive Buffer Register | - |
| THR | 0x004 | WO | Transmit Buffer Register | - |
| DLR | 0x008 | R/W | Baud Rate Divider Register | 0x00000001 |
| IER | 0x00c | R/W | Interrupt Enable Register | 0x00000000 |
| IIR | 0x010 | RO | Interrupt Status Register | 0x00000001 |
| LCR | 0x018 | R/W | Line Control Register | 0x00000000 |
| MCR | 0x01C | R/W | Modem Control Register | 0x00000000 |
| LSR | 0x020 | RO | Line Status Register | 0x00000060 |

# 15.4Register Description

## 15.4.1 Receive Buffer Register (RBR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | RBR | Read operation, return the data received from the receive buffer. | - |

## 15.4.2 Transmit Buffer Register (THR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | THR | Write data to the transmit buffer, the UART module will subsequently send the data out from the buffer. | - |

## 15.4.3 Baud Rate Divider Register (DLR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Reserved | - |
| 15:0 | DLR | Baud rate = PCLK/(16×DLR) | 0x1 |

### 15.4.4 Interrupt Enable Register (IER)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:3 | - | Reserved | - |
| 2 | RLSIE | Receive line status interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 1 | THREIE | Transmit hold register empty interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 0 | RBRIE | Receive data valid interrupt/receive timer overflow interrupt enable bit<br>0: Disable<br>1: Enable | 0 |

### 15.4.5 Interrupt Status Register (IIR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3:1 | INTID | Interrupt status indication<br>0x0: Modem status has been changed.<br>0x1: Transmit holding register is empty.<br>0x2: Receive data is valid.<br>0x3: Receive line status | 0x0 |
| 0 | INT STATUS | Interrupt status<br>0: At least one interrupt in the queue.<br>1: No interrupt in the queue. | 1 |

## 15.4.6 Line Control Register (LCR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:7 | - | Reserved | - |
| 6 | BCON | Break control bit<br>When this bit is written as 1, it enables Break transfer, and TXD port is forced to output logic 0. | 0 |
| 5:4 | PSEL | Parity bit selection<br>0x0: Odd parity, where odd numbers of logic 1 are transmitted and checked in each byte.<br>0x1: Even parity, where even numbers of logic 1 are transmitted and checked in each byte.<br>0x2: Parity bit is forced to 1.<br>0x3: Parity bit is forced to 0. | 0x0 |
| 3 | PEN | Parity check bit enable<br>0: Disable parity check bit generation and detection<br>1: Enable parity check bit generation and detection | 0 |
| 2 | SBS | Stop bit selection<br>0: 1-bit stop bit<br>1: When the transmit word length is 5 bits, the stop bit is 1.5 bits; when the transmit word length is other, the stop bit is 2 bits. | 0 |
| 1:0 | WLS | Word length select bit<br>0x0: 5-bit<br>0x1: 6-bit<br>0x2: 7-bit<br>0x3: 8-bit | 0x0 |

## 15.4.7 Modem Control Register (MCR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:5 | - | Reserved | - |
| 4 | MLBM | Modem loopback mode<br>0: Disable Modem loopback mode<br>1: Enable Modem loopback mode | 0 |
| 3:0 | - | Reserved | - |

### 15.4.8 Line Status Register (LSR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:7 | - | Reserved | - |
| 6 | TEMT | Transmit buffer empty flag bit (read-only)<br>0: Transmit buffer has unsent data<br>1: Transmit buffer is empty | 0 |
| 5 | THRE | Transmit register empty flag bit (read-only)<br>0: Transmit register has unsent data<br>1: Transmit register is empty | 0 |
| 4 | BI | Break interrupt flag bit (read-only)<br>0: No break interrupt detected.<br>1: Break interrupt detected.<br>A break interrupt is generated when the UART data input is held low during a transmission (start bit, data, parity bit, and stop bit). The UART remains idle until the data input goes high.<br>The bit can be cleared by reading LSR. | 0 |
| 3 | FE | Frame error flag bit (read-only)<br>0: No frame error detected.<br>1: Frame error detected.<br>The bit can be cleared by reading LSR. | 0 |
| 2 | PE | Parity check error flag bit (read-only)<br>0: No parity check error detected.<br>1: Parity check error detected.<br>The bit can be cleared by reading LSR. | 0 |
| 1 | - | Reserved | 0 |
| 0 | RDR | Receive data valid flag bit (read-only)<br>0: No unread data in receive buffer.<br>1: Unread data in receive buffer. | 0 |

# Chapter 16 Serial Interface IICA (IICA)

## 16.1Functions of Serial Interface IICA

This Serial Interface IICA has the following three modes.

1) Run stop mode

This is a mode for non-serial transfer, which reduces power consumption.

2) $I^2C$ bus mode (multi-master supported)

This mode transfers 8-bit data to and from multiple devices via two lines on the serial clock (SCLAn) and serial data bus (SDAAn). Conforming to the $I^2C$ bus format, the master device can generate “start condition”, “address”, “transfer direction”, “data” and “stop condition” for slave devices on the serial data bus. The slave device automatically detects the received status and data by hardware. This function simplifies the $I^2C$ bus control of the application program.

Because the SCLAn pin and SDAAn pin of the serial interface IICA are used as open-drain outputs, the serial clock line and serial data bus require pull-up resistors.

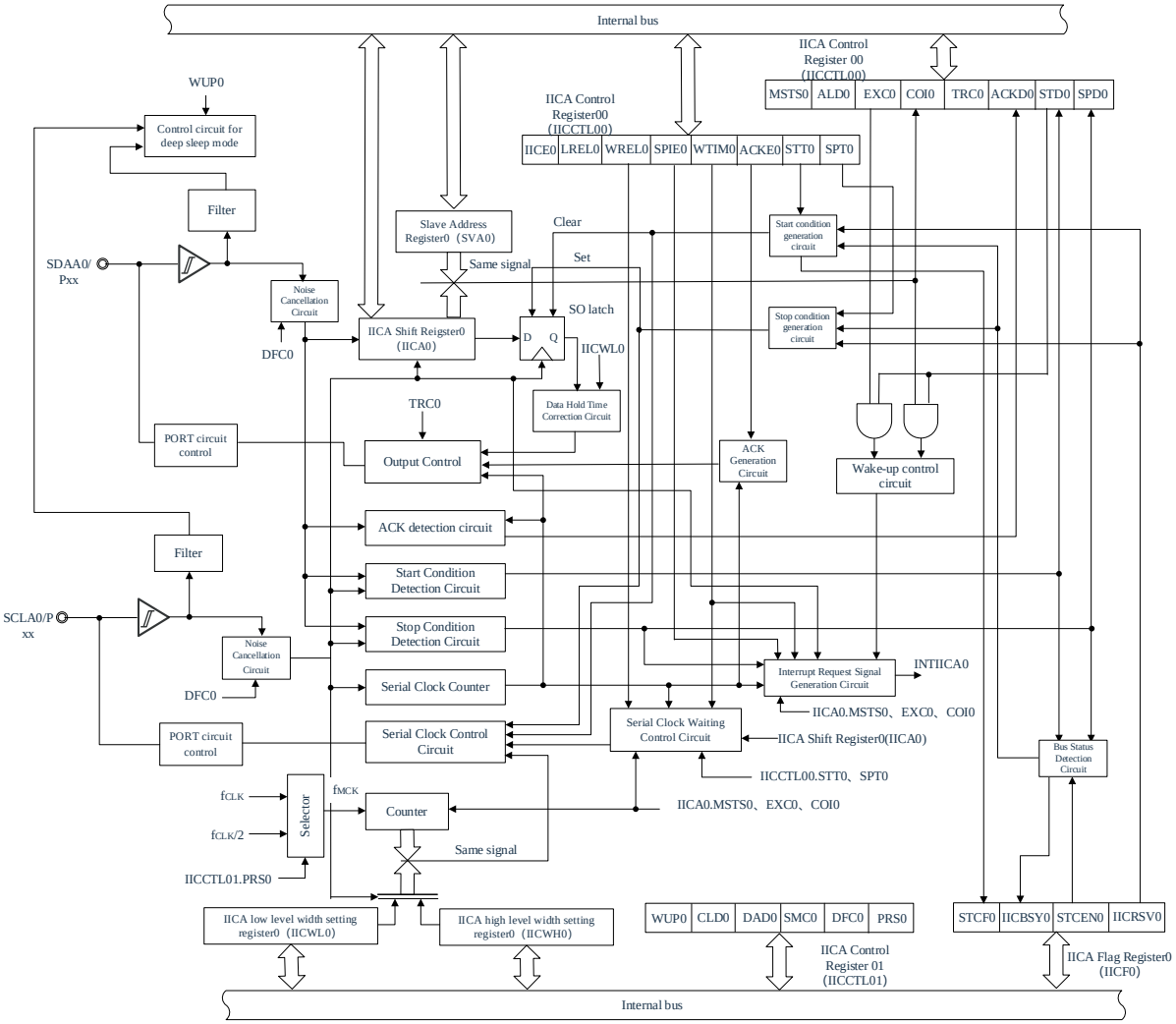
3) Wake-up mode

In a deep sleep mode, when an extension code or a local station address from a master device is received, the deep sleep mode can be released by generating an interrupt request signal (INTIICAn). It is set by the WUPn bit of the IICA control register n1 (IICTLn1).

The block diagram of the serial interface IICA is shown in Figure 16-1.

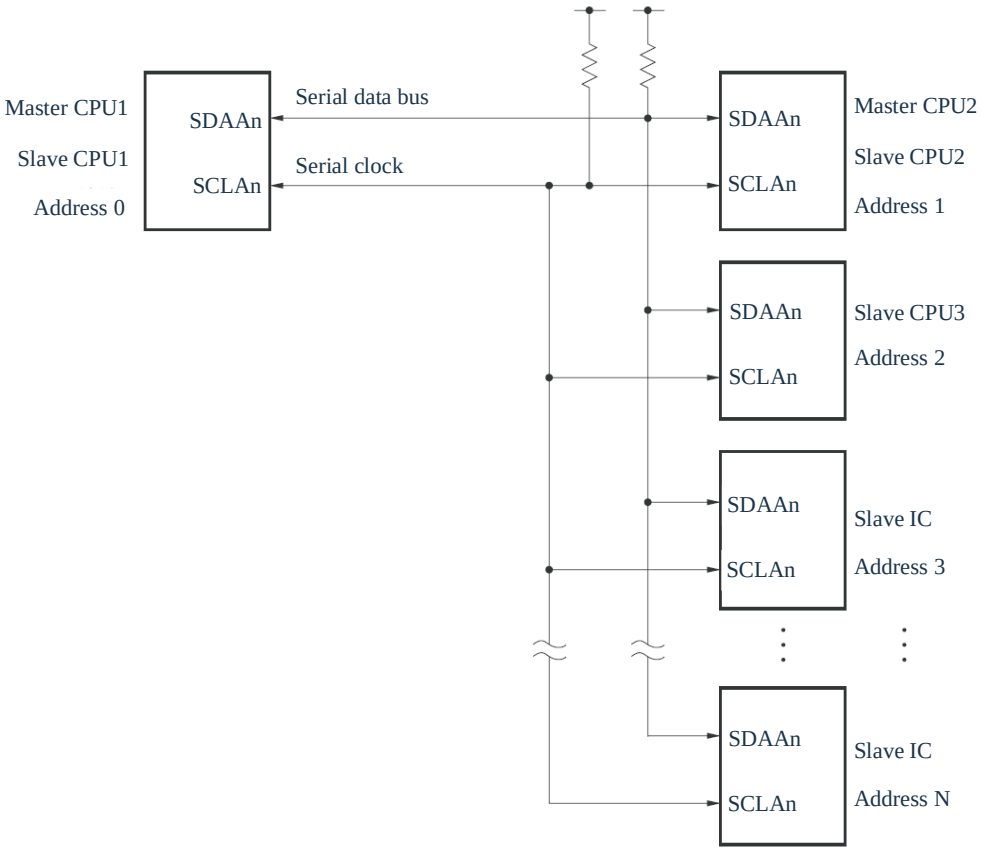
Note: n=0.

Figure 16-1 Block diagram of serial interface IICA



An example of serial bus structure is shown in Figure 16-2.

Figure 16-2 Example of a serial bus structure for I²C bus




Note: n=0.

# 16.2Structure of Serial Interface IICA

The serial interface IICA consists of the following hardware.

Table 16-1 Structure of serial interface IICA

| Item | Structure |
|---|---|
| Register | IICA shift register n (IICAn)<br>Slave address register n (SVAn) |
| Control register | Peripheral enable register 0 (PER0)<br>IICA control register n0 (IICCTLn0)<br>IICA status register n (IICSn)<br>IICA flag register n (IICFn)<br>IICA control register n1 (IICCTLn1)<br>IICA low-level width setting register n (IICWLn)<br>IICA high-level width setting register n (IICWHn)<br>Port mode register (PMxx)<br>Port mode control register (PMCxx)<br>Port alternate function configuration register (PxxCFG) |

Note 1: n=0.

Note 2: This product can alternate the IICA input/output pin functions to multiple ports. When a port is configured to IICA pin alternate function, the N-channel open drain output ($V_{DD}$/$EV_{DD}$ withstand) mode of that port is guaranteed by design to be turned on automatically, i.e., the POMxx registers do not need to be set by the user.

(1) IICA shift register n (IICAn)

The IICAn registers are registers for transmitting and receiving 8-bit serial data and 8-bit parallel data interconversion synchronously with the serial clock. The actual transmission and reception can be controlled by reading and writing the IICAn register.

During the waiting period, the waiting is released by writing the IICAn register to start transferring data. The IICAn register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register changes to “00H”.

Table 16-2 Format of IICAn shift register n (IICAn)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | IICAn | IICAn shift register n | 0x0 |

Note 1: During data transfer, no data can be written to IICAn registers.

Note 2: IICAn registers can only be read and written during the wait period. Access to the IICAn register is prohibited in the communication state except for the waiting period. However, in the case of a master device, the IICAn register can be written once after setting the communication trigger bit (STTn) to 1.

Note 3: When making a reservation for communication, data must be written to the IICAn register after detecting an interrupt caused by a stop condition.

Note 4: n=0.

(2) Slave address register n (SVAn)

This is a register that holds the 7-bit local station address {A6, A5, A4, A3, A2, A1, A0} when used as a slave.

The SVAn register is set by an 8-bit memory manipulation instruction. However, rewriting this register is prohibited when the STDn bit is 1 (start condition detected).

After a reset signal is generated, the value of this register changes to “00H”.

Table 16-3 Format of the slave address register n (SVAn)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | SVAn | Slave address register n | 0x0 |

Note: bit0 is fixed to “0”.

(3) SO latch

The SO latch holds the output level of the SDAAn pin.

(4) Wake-up control circuit

The circuit generates an interrupt request (INTIICAn) when the address value set in the slave address register n (SVAn) is the same as the received address.

(5) Serial clock counter

During transmission or reception, this counter counts the output or input serial clocks and checks whether 8-bit data transmission and reception are performed.

(6) Interrupt request signal generation circuit

This circuit controls the generation of an interrupt request signal (INTIICAn).

An $I^2C$ interrupt request is generated by the following 2 triggers.

- Falling on the 8th or 9th serial clock (set by the WTIMn bit)
- An interrupt request is generated due to the detection of a stop condition (set by the SPIEn bit).

Note 1: WTIMn bit: bit3 of IICA control register n0 (IICCTLn0)

Note 2: SPIEn bit: bit4 of IICA control register n0 (IICCTLn0)

(7) Serial clock control circuit

In the master mode, the circuit generates a clock output to the SCLAn pin from the sampling clock.

(8) Serial clock wait control circuit

This circuit controls the wait timing.

(9) Ack generation circuit, stop condition detection circuit, start condition detection circuit, Ack detection circuit

These circuits generate and detect various states.

(10) Data hold time correction circuit

This circuit generates a data hold time against a falling serial clock.

(11) Start condition generation circuit

If the STTn bit is set to 1, this circuit generates a start condition.

However, in the state where reservation communication is prohibited (IICRSVn bit = 1) and the bus is not released (IICBSYn bit = 1), the start condition request is ignored and the STCFn bit is set to 1.

(12) Stop condition generation circuit

If the SPTn bit is set to 1, this circuit generates a stop condition.

(13) Bus status detection circuit

This circuit detects whether the bus is released by detecting the start and stop conditions. However, the bus status cannot be detected immediately when it is just operated, so the initial state of the bus status detection circuit must be set by the STCENn bit.

Note: TTn bit: bit1 of IICA control register n0 (IICCTLn0)

SPTn bit: bit0 of IICA control register n0 (IICCTLn0)

IICRSVn bit: bit0 of IICA flag register n (IICFn)

IICBSYn bit: bit6 of IICA flag register n (IICFn)

STCFn bit: bit7 of IICA flag register n (IICFn)

STCENn bit: bit1 of IICA flag register n (IICFn)

n=0.

# 16.3Register Mapping

(Base address = 0x4004_1A30) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| IICCTLn0 | 0x000 | R/W | IICA control register n0 | 0x00 |
| IICCTLn1 | 0x001 | R/W | IICA control register n1 | 0x00 |
| IICWLn | 0x002 | R/W | IICA low level width setting register n | 0xFF |
| IICWHn | 0x003 | R/W | IICA high level width setting register n | 0xFF |
| SVA0 | 0x004 | R/W | IICA slave address register | 0x00 |
| IICA0 | 0x120 | R/W | IICA shift register | 0x00 |
| IICSn | 0x121 | RO | IICA status register n | 0x00 |
| IICFn | 0x122 | R/W | IICA flag register n | 0x00 |

Note: n=0.

(Base address = 0x4002_0420) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| PER0 | 0x020 | R/W | Peipheral register 0 | 0x00 |

# 16.4Registers for Controlling Serial Interface IICA

The serial interface IICA is controlled by the following registers.

Peripheral enable register 0 (PER0)

IICA control register n0 (IICCTLn0)

IICA flag register n (IICFn)

IICA status register n (IICSn)

IICA control register n1 (IICCTLn1)

IICA low level width setting register n (IICWLn)

IICA high level width setting register n (IICWHn)

Port mode register (PMxx)

Port mode control register (PMCxx)

Port alternate function configuration register (PxxCFG)

Note: n=0.

# 16.5Peripheral Enable Register 0 (PER0)

The PER0 register is used to enable or disable the clock supplied to various peripheral hardware. By stopping the clock to unused hardware, it helps reduce power consumption and noise.

To use the serial interface IICAn, set the bit4 (IICAEN) to 1.

The PER0 register is set by the 8-bit memory manipulation instruction.

After generating a reset signal, the value of this register changes to “00H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | LSITIMEREN | Control of LSITIMER input clock supply (Power-down sleep is possible)<br>0: Stop to supply the input clock, the SFR used by the LSITIMER cannot be written.<br>1: Supply the input clock, the SFR used by the LSITIMER can be written. | 0 |
| 6:5 | -- | Reserved | 0x0 |
| 4 | IICAEN | Control of IICA input clock supply<br>0: Stop to supply the input clock, the SFR used by the IICA cannot be written.<br>1: Supply the input clock, the SFR used by the IICA can be written. | 0 |
| 3:1 | -- | Reserved | 0x0 |
| 0 | TM40EN | Control of universal timer unit0 input clock supply<br>0: Stop to supply the input clock, the SFR used by the universal timer unit0 cannot be written.<br>1: Supply the input clock, the SFR used by the universal timer unit0 can be written. | 0 |

Note 1: To set the serial interface IICA, the following registers must be set when the IICAnEN bit is 1. When the IICnEN bit is 0, the control registers of the serial interface IICA are initialized and write operations are ignored (except for the port alternate function configuration register (PxxCFG), the port mode register (PMxx) and the port mode control register (PMCxx)).

(1) IICA control register n0 (IICCTLn0)

(2) IICA flag register n (IICFn)

(3) IICA status register n (IICSn)

(4) IICA control register n1 (IICCTLn1)

(5) IICA low-level width setting register n (IICWLn)

(6) IICA high-level width setting register n (IICWHn)

Note 2: n=0.

## 16.5.1 IICA Control Register n0 (IICCTLn0)

This is the register that enables or stops $I^2C$ operation, sets wait timings, and sets other $I^2C$ operations.

The IICCTLn0 register is set by an 8-bit memory manipulation instruction. However, the SPIEn bit, the WTIMn bit, and the ACKEn bit must be set when the ICEn bit is 0 or during the wait period, and these bits can be set simultaneously when the ICEn bit is set from 0 to 1.

After a reset signal is generated, the value of this register changes to “00H”.

Note: n=0.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | IICEn | $I^2C$ operation enable<br>0: Stop operation. Reset the IICA status register n (IICSn) Note 1 and stop internal operation.<br>1: Enable operation<br>Caution: This bit must be set to “1” when the SCLAn and SDAAn lines are high.<br><br>Set condition (IICEn=1):<br>• Set by intructions<br><br>Clear condition (IICEn=0):<br>• Cleared by intructions.<br>• When resetting | 0 |
| 6 | LRELn Note 2, 3 | Exit of communication<br>0: Normal operation<br>1: Exit from the current communication and enter standby status. Automatically cleared to 0 after execution.<br>Used in cases such as when an extension code is received that is not related to the local station.<br>The SCLAn line and the SDAAn line become high impedance.<br>The following flags in the IICA control register n0 (IICCTLn0) and IICA status register n (IICSn) are cleared to 0.<br>•STTn•SPTn•MSTSn•EXCn•COIn•TRCn•ACKDn•STDn<br><br>Changes to the standby state for exiting communication and remains until the following communication participation conditions are met.<br>• Starts as a master device after a stop condition is detected.<br>• The address matches or receives an extension code after a start condition is detected.<br><br>Set condition (LRELn=1):<br>•Set by intructions.<br><br>Clear condition (LRELn=0):<br>•Automatically cleared after execution.<br>•When resetting | 0 |
| 5 | WRELn Note2, 3 | Release waiting<br>0: The wait is not released.<br>1: Release the wait. Automatically clears after the wait is released.<br>If the WRELn bit (release from wait) is set during the 9th clock wait in the transmit state (TRCn=1), the SDAAn line changes to a high impedance state (TRCn=0).<br><br>Set condition (WRELn=1):<br>•Set by intructions. | 0 |

| | | | |
|---|---|---|---|
| | | Clear condition (WRELn=0):<br>•Automatically cleared after execution.<br>•When resetting | |
| 4 | SPIEn[Note4] | Enable or disable interrupt requests generated by stop condition detection<br>0: Disable<br>1: Enable<br>When the WUPn bit of IICA control register n1 (IICCTLn1) is 1, even if the SPIEN is 1, there is also no stop condition interrupt.<br><br>Set condition (SPIEn=1):<br>•Set by intructions.<br><br>Clear condition (SPIEn=0):<br>•Automatically cleared after execution.<br>•When resetting | 0 |
| 3 | WTIMn[Note4] | Control of wait and interrupt requests<br>0: An interrupt request signal is generated on the falling edge of the 8th clock.<br>Master: After outputting 8 clocks, set the clock output low to wait.<br>Slave: After inputting 8 clocks, set the clock low and wait for the master.<br>1: An interrupt request signal is generated on the falling edge of the 9th clock.<br>Master: After outputting 9 clocks, set the clock output low to wait.<br>Slave: After inputting 9 clocks, set the clock low and wait for the master.<br><br>An interrupt is generated on the falling edge of the 9th clock during the address transfer, regardless of the setting of this bit; the setting of this bit is valid at the end of the address transfer. The master device enters the wait state on the falling edge of the 9th clock during address transfer. A slave device that receives a local station address enters the wait state on the falling edge of the 9th clock after an acknowledge (ACK) is generated, but a slave device that receives an extension code enters the wait state on the falling edge of the 8th clock.<br><br>Set condition (WTIMn=1):<br>•Set by intructions.<br><br>Clear condition (WTIMn=0):<br>•Cleared by intructions.<br>•When resetting | 0 |
| 2 | ACKEn[Note4, 5] | ACK control<br>0: No ACK.<br>1: Enable ACK. Set the SDAAn line low during the 9th clock.<br>Set condition (ACKEn=1):<br>•Set by intructions.<br><br>Clear condition (ACKEn=0):<br>•Cleared by intructions.<br>•When resetting | 0 |
| 1 | STTn[Note2, 6] | Triggering of the start condition<br>0: No start conditions are generated.<br>1: When the bus is released (standby state, IICBSYn bit is 0): If this bit is 1, a start condition (boot as the master device) is generated.<br>When a third party is communicating: | 0 |

| | | | |
|---|---|---|---|
| | | • Enable communication reservation function (IICRSVn=0).<br>Used as a start condition reservation flag. If this bit set to 1, a start condition is automatically generated just after the bus is released.<br><br>• Disable communication reservation function (IICRSVn=1).<br>Even if this bit is 1, the STTn bit is cleared and the STTn clear flag (STCFn) is set to 1 without generating a start condition.<br>• Wait status (master device):<br>Generates a restart condition after the wait is released.<br><br>Cautions on setting timing:<br>• Master reception: Disable setting this bit to 1 during transfer. This bit can only be set to 1 during the wait period after the ACKEn bit is set to 0 and the slave device is notified that reception has been completed.<br>• Master transmission: During the Ack, a start conditionmay not be generated properly. This bit must be set to 1 during the wait period after the 9th clock is output.<br>• It is prohibited to set 1 at the same time as the stop condition trigger (SPTn).<br>• After setting the STTn bit to 1, it is prohibited to set this bit to 1 again before the clear condition is satisfied.<br>Set condition (STTn=1):<br>•Set by intructions.<br><br>Clear condition (STTn=0):<br>· Set the STTn bit to 1 when the communication reservation is disabled.<br>· When arbitration fails<br>· Master device generates start conditions.<br>· Cleared due to the LRELn bit is 1 (exit communication).<br>· When the IICEn bit is 0 (stop operation).<br>· When resetting | |
| 0 | SPTn[Note7] | Trigger of stop condition<br>0: No stop condition is generated.<br>1: Generate a stop condition (end of transfer as master device).<br>Cautions on setting timing:<br>• Master receive: Disable setting this bit to 1 during transfer. This bit can only be set to 1 during the waiting period when ACKEn is at 0 and notifying the slave reception has completed.<br>• Master transmit: During the Ack, the stop conditions may not be generated properly. This bit must be set to 1 during the wait period after the 9th clock is output.<br>• Prohibit setting 1 at the same time as the start condition trigger (STTn).<br>• When the WTIMn bit is 0, it must be noted that if the SPTn bit is set to 1 during the wait period after 8 clocks of output, a stop condition is generated during the high level of the 9th clock after the wait is released. The WTIMn bit must be set from 0 to 1 during the wait period after 8 clocks of output and the SPTn bit must be set to 1during the wait period after the 9th clock of output.<br>• After setting the SPTn bit to 1, it is prohibited to set this bit to 1 again until the clear condition is satisfied.<br>Set condition (SPTn=1):<br>•Set by intructions.<br><br>Clear condition (SPTn=0):<br>· When arbitration fails | 0 |

| | | | |
|---|---|---|---|
| | | · Automatically cleared when a stop condition is detected.<br>· Cleared due to LRELn bit is 1 (exit communication)<br>· When the IICEn bit is 0 (stop operation)<br>· When resetting | |

Note 1: Reset the STCFn bit and IICBSYn bit of the IICA shift register n (IICAn), the IICA flag register n (IICFn), and the CLDn bit and DADn bit of the IICA control register n1 (IICCTLn1).

Note 2: If the ICEn bit is 0, the signal of this bit is invalid.

Note 3: The LRELn bit and the WRELn bit are always read as 0.

Note 4: The signal of this bit is invalid when the ICEn bit is 0. This bit must be set during this period.

Note 5: The set value is invalid when it is not an extension code during address transfer. When it is a slave device and the address matches, an ACK is generated regardless of the set value.

Note 6: The STTn bit is always read as 0.

Note 7: The SPTn bit is always read as 0.

Note 8: If $I^2C$ operation is enabled (IICCTLn=1) when the SCLAn line is high, the SDAAn line is low and the digital filter is ON (DFCn=1 in the IICCTLn1 register), the start condition is detected immediately. In this case, the LRELn bit must be set to 1 consecutively by a bit memory manipulation instruction after $I^2C$ operation is enabled (ICEn=1).

Note 9: n=0. If bit3 (TRCn) of the IICA status register n (IICSn) is 1 (transmit state), the wait is released by setting bit5 (WRELn) of the IICCTLn0 register to 1 on the 9th clock, the SDAAn line is set to high impedance after clearing the TRCn bit (receive state). The wait release must be performed by writing to the IICA shift register n when the TRCn bit is 1 (transmit state).

Note 10: n=0

## 16.5.2 IICA Status Register n (IICSn)

This is a register that represents the $I^2C$ status.

The IICSn register can be read by an 8-bit memory manipulation instruction only when the STTn bit is 1 and waiting. After the reset signal is generated, the value of this register changes to "00H".

Note 1: In the enabled address matching wake function (WUPn=1) state in deep sleep mode, reading the IICSn register is prohibited. In the state where the WUPn bit is 1, it has nothing to do with the INTIICAn interrupt request if you change the WUPn bit from 1 to 0 (stop wake-up operation) reflects a change in state until the next start condition or stop condition is detected. Therefore, to use the wake-up function, it is necessary to enable (SPIEn=1) an interrupt due to the detection of a stop condition, and to read the IICSn register after an interrupt is detected.

Note 2: STTn: bit1 of IICA control register n0 (IICCTLn0), n=0.

Note 3: WUPn: bit7 of IICA control register n1 (IICCTLn1), n=0.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | MSTSn | Acknowledgement flag of master status<br>0: Slave or communication standby status<br>1: Master communication status<br>Set condition (MSTSn=1):<br>• When generating a start condition<br><br>Clear condition (MSTSn=0):<br>· When a stop condition is detected<br>· When the ALDn bit is 1<br>· Cleared due to LRELn bit is 1 (exit communication)<br>· When the IICEn bit changes from 1 to 0 (stop operation)<br>· When resetting | 0 |
| 6 | ALDn | Detection of arbitration failures<br>0: It indicates that arbitration did not occur or was won.<br>1: It indicates an arbitration failure. Clear the MSTSn bit.<br>Set condition (ALDn=1):<br>• When arbitration fails<br><br>Clear condition (ALDn=0)<br>· Automatically cleared after reading the IICSn register Note1.<br>· When the IICEn bit changes from 1 to 0 (stop operation)<br>· When resetting | 0 |
| 5 | EXCn | Reception detection of extended codes<br>0: No extension code is received.<br>1: An extension code is received.<br>Set condition (EXCn=1):<br>· When the high 4 bits of the received address data are "0000" or "1111".<br>(Set on the rising edge of the 8th clock).<br><br>Clear condition (EXCn=0):<br>· When a start condition is detected<br>· When a stop condition is detected<br>· Cleared due to the LRELn bit is 1(exits communication).<br>· When the IICEn bit changes from 1 to 0 (stops running).<br>· When resetting | 0 |
| 4 | COIn | Detection of address matching<br>0: The addresses are different.<br>1: The addresses are the same.<br>Set condition (COIn=1):<br>· When the receive address and the local station address (Slave Address Register n (SVAn)) are the same (set on the rising edge of the 8th clock) | 0 |

| | | | |
|---|---|---|---|
| | | Clear condition (COIn=0):<br>· When a start condition is detected<br>· When a stop condition is detected<br>· Cleared due to the LRELn bit is 1(exit communication).<br>· When the IICEn bit changes from 1 to 0 (stop operation).<br>· When resetting | |
| 3 | TRCn | Transmit/receive status detection<br>0: In the receive state (except in the transmit state). Set the SDAAn line to high impedance.<br>1: In the transmitting state. Set to output the value of the SOn latch to the SDAAn line (valid after the falling edge of the 9th clock byte of the 1st byte).<br>Set condition (TRCn=1):<br>＜Master＞<br>· When generating a start condition<br>· When the LSB (transmission direction indicator bit) of byte 1 (address transmission) outputs 0 (master transmit).<br>＜Slave＞<br>· When the LSB (transmission direction indication bit) of byte 1 (address transmission) of the master device inputs 1 (slave transmit)<br><br>Clear condition (TRCn=0):<br>＜Master and slave＞<br>· When a stop condition is detected<br>· Cleared due to the LRELn bit is 1 (exit communication).<br>· When the IICEn bit changes from 1 to 0 (stop operation).<br>· Cleared because the WRELn bit is 1 (released from wait)Note 2<br>· When the ALDn bit changes from 0 to 1 (arbitration fails)<br>· When resetting<br>＜Master＞<br>· When the LSB (transmission direction flag bit) of the 1st byte outputs 1<br>＜Slave＞<br>· When a start condition is detected<br>· When the LSB (transfer direction flag bit) of the 1st byte inputs 0 | 0 |
| 2 | ACKDn | Detection of acknowledge<br>0: No ACK is detected.<br>1: An ACK is detected.<br>Set condition (ACKDn=1):<br>• When the SCLAn line is set to a low level by the 9th clock rising edge of the SDAAn line<br><br>Clear condition (ACKDn=0):<br>· When a stop condition is detected<br>· When the first clock of the next byte goes up<br>· Cleared due to LRELn bit is 1 (exit communication)<br>· When the IICEn bit changes from 1 to 0 (stop operation)<br>· When resetting | 0 |
| 1 | STDn | Detection of starting conditions<br>0: No start condition detected.<br>1: A start condition is detected, indicating that it is in address transfer.<br>Set condition (STDn=1):<br>• When a start condition is detected<br><br>Clear condition (STDn=0):<br>· When a stop condition is detected<br>· When the 1st clock rises after the next byte of the address is transmitted<br>· Cleared due to LRELn bit is 1 (exit communication)<br>· When the IICEn bit changes from 1 to 0 (stop operation)<br>· When resetting | 0 |
| 0 | SPDn | Detection of stop conditions | 0 |

| | | | |
|---|---|---|---|
| | | 0: No stop condition detected.<br>1: A stop condition is detected, the master device ends communication and the bus is released.<br>Set condition (SPDn=1):<br>• When a stop condition is detected<br><br>Clear condition (SPDn=0):<br>· After setting this bit, the address transmits the byte after the start condition is detected when the clock rises<br>· When the WUPn bit changes from 1 to 0<br>· When the IICEn bit changes from 1 to 0 (stop operation)<br>· When resetting | |

Note 1: This bit is cleared even if the bit memory manipulation instruction is executed on a bit other than the IICSn register. Therefore, when using ALDn bits, the data for the ALDn bits must be read before reading other bits.

Note 2: If bit3 (TRCn) of the IICA status register n (IICSn) is 1 (transmit state) and bit5 (WRELn) of the IICA control register n0 (IICCTLn0) is set to 1 on the 9th clock to release the wait, the SDAAn line is set to high impedance after clearing the TRCn bit (receive state). The wait release must be performed by writing the IICA shift register n when the TRCn bit is 1 (transmit state).

Note 3: LRELn: bit6 of IICA control register n0 (IICCTLn0)

Note 4: IICEn: bit7 of IICA control register n0 (IICCTLn0)

Note 5: n=0.

### 16.5.3 IICA Flag Register n (IICFn)

This is a register that sets the $I^2C$ operating mode and indicates the status of the $I^2C$-bus.

The IICFn register is set by an 8-bit memory manipulation instruction. However, only the STTn clear flag (STCFn) and $I^2C$ bus status flag (IICBSYn) can be read.

The communication reservation function is enabled or disabled by the IICRSVn bit setting, and the initial value of the IICBSYn bit is set by the STCENn bit. The IICRSVn bit and the STCENn bit can be written only when $I^2C$ operation is disabled (bit7(IICENn)=0 of the IICA control register n0 (IICCTLn0)). Only the IICFn register can be read after operation is allowed. The value of this register changes to "00H" after the reset signal is generated.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | STCFn | STTn clear flag<br>0: Release start conditions.<br>1: The STTn flag cannot be cleared while the start condition cannot be issued.<br>Set condition (STCFn=1):<br>· If the STTn bit is cleared to 0 because the start condition cannot be issued when the communication reservation is disabled (IICRSVn=1).<br><br>Clear condition (STCFn=0):<br>· Cleared due to STTn bit is 1<br>· When the IICEn bit is 0 (stop operation)<br>· When resetting | 0 |
| 6 | IICBSYn | $I^2C$ bus status flag<br>0: Bus release state (initial state of communication when STCENn=1)<br>1: Bus communication state (initial state of communication when STCENn=0)<br>Set condition (IICBSYn=1):<br>· When a start condition is detected<br>· Set the IICEn bit when the STCENn bit is 0<br><br>Clear condition (IICBSYn=0):<br>· When a stop condition is detected<br>· When the IICEn bit is 0 (stop operation)<br>· When resetting | 0 |
| 5:2 | - | Reserved | - |
| 1 | STCENn | Initial start enable trigger<br>0: After enabling operation (IICEn=1), a start condition is allowed to be generated by detecting a stop condition.<br>1: After enabling operation (IICEn=1), a start condition is allowed to be generated without detecting a stop condition.<br>Set condition (STCENn=1):<br>•Set by intructions.<br><br>Clear condition (STCENn=0):<br>· Cleared by intructions.<br>· When a start condition is detected<br>· When resetting | 0 |
| 0 | IICRSVn | Communication reservation function disable bit<br>0: Communication reservations are enabled.<br>1: Communication reservations are disabled.<br>Set condition (IICRSVn=1):<br>•Set by intructions.<br>Clear condition (IICRSVn=0):<br>•Cleared by intructions. | 0 |

Note 1: bit6 and bit7 are read-only bits.

Note 2: The STCENn bit can only be written when the operation is stopped (IICEn=0).

Note 3: If the STCENn bit is 1, the bus is considered to be released (IICBSYn=0) regardless of the actual bus state, so it is necessary to confirm that there is no third party in communication in order to avoid disrupting other communications when the first start condition (STTn=1) is issued.

Note 4: The IICRSVn can be written only when it is stopped (IICEn=0).

Note 5: STTn: bit1 of IICA control register n0 (IICCTLn0).

Note 6: IICEn: bit7 of IICA control register n0 (IICCTLn0).

## 16.5.4 IICA Control Register n1 (IICCTLn1)

This is a register used to set the $I^2C$ operation mode and detect the status of the SCLAn and SDAAn pins.

The IICCTLn1 register is set by an 8-bit memory manipulation instruction. However, only CLDn and DADn bits can be read.

In addition to the WUPn bit, the IICCTLn1 register must be set when $I^2C$ operation is disabled (bit7 (IICEn) = 0 of the IIC control register n0 (IICCTLn0)).

After a reset signal is generated, the value of this register changes to “00H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | WUPn | Control of address matching wakeup<br>0: Stop the operation of the Address Match Wakeup function in deep sleep mode.<br>1: Enable the operation of the Address Match Wakeup function in deep sleep mode.<br>To transfer to deep sleep mode by setting the WUPn bit to 1, at least three $F_{MCK}$ clocks must be elapsed after setting the WUPn bit to 1, and then the deep sleep instruction must be executed (refer to Figure 16-3).<br>The WUPn bit must be cleared to 0 after the address is matched or the extension code is received. It is possible to participate in subsequent communications by clearing the WUPn bit to 0 (it is necessary to release the wait and write transmit data after clearing the WUPn bit to 0).<br>When the WUPn bit is 1, the interrupt timing when the address is matched or the extension code is received is the same as the interrupt timing when WUPn bit is 0.<br>(The delay difference of sampling error is generated according to the clock). In addition, when the WUPn bit is 1, even if the SPIEn bit is set to 1, no stop condition interrupt is generated.<br><br>Set condition (WUPn=1):<br>· Set by instructions (MSTSn=0, EXCn=0, COIn=0 and STDn=0. (does not participate in communications)) Note 2.<br><br>Clear condition (WUPn=0):<br>· Cleared by instructions (after an address match or reception of an extension code). | 0 |
| 6 | - | Reserved | - |
| 5 | CLDn | Level detection of the SCLAn pin (valid only when the IICEn bit is 1)<br>0: SCLAn pin is detected low.<br>1: SCLAn pin is detected high.<br>Set condition (CLDn=1):<br>· When the SCLAn pin is high level<br><br>Clear condition (CLDn=0):<br>· When the SCLAn pin is low<br>· When the IICEn bit is 0 (stop operation)<br>· When resetting | 0 |
| 4 | DADn | Level detection on the SDAAn pin (valid only when the IICEn bit is 1)<br>0: SDAAn pin is detected as low.<br>1: SDAAn pin is detected as high.<br>Set condition (DADn=1):<br>· When the SDAAn pin is high level<br><br>Clear condition (DADn=0):<br>· When the SDAAn pin is low<br>· When the IICEn bit is 0 (stop operation) | 0 |

| | | | |
|---|---|---|---|
| | | · When resetting | |
| 3 | SMCn | Switching of operating modes<br>0: Operates in standard mode (maximum transfer rate: 100kbps).<br>1: Operates in fast mode (maximum transfer rate: 400kbps) or enhanced fast mode (maximum transfer rate: 1Mbps). | 0 |
| 2 | DFCn | Operation control of digital filters<br>0: Digital filter OFF<br>1: Digital filter ON<br>Digital filters must be used in fast mode or enhanced fast mode.<br>Digital filters are used to eliminate noise.<br>Whether the DFCn is 1 or 0, the transfer clock is unchanged. | 0 |
| 1 | - | Reserved | - |
| 0 | PRSn | Control of the operation clock ($F_{MCK}$)<br>0: Selects $F_{CLK}$ (1MHz≤$F_{CLK}$≤20MHz).<br>1: Selects $F_{CLK}$/2 (20MHz＜$F_{CLK}$). | 0 |

Note 1: bit4 and bit5 are read-only bits.

Note 2: During the period shown below, it is necessary to check the status of the IIC status register n (IISn) and set it.

Note 3: The maximum operating frequency of the IICA running clock ($f_{MCK}$) is 20MHz (Max.). Bit0 (PRSn) of the IICA control register n1 (IICCTLn1) must be set to "1" only if $f_{CLK}$ exceeds 20MHz.

Note 4: When setting the transfer clock, attention must be paid to the minimum operating frequency of $F_{CLK}$. The minimum operating frequency of the $F_{CLK}$ of the serial interface IICA depends on the mode of operation.

Fast mode: $f_{CLK}$=3.5MHz(Min.)

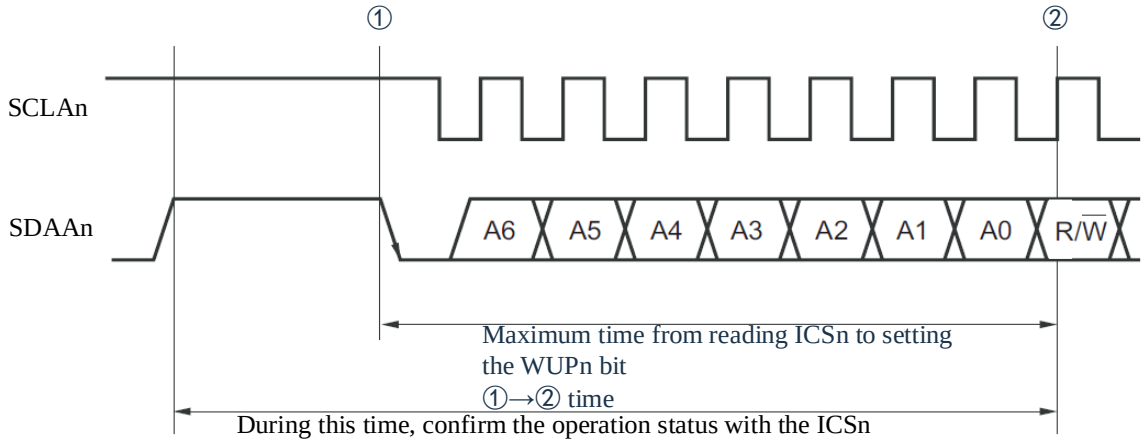
Enhanced fast mode: $f_{CLK}$=10MHz(Min.)

Standard mode: $f_{CLK}$=1MHz(Min.)

Note 5: IICEn: bit7 of IICA control register n0 (IICCTLn0)

Note 6: n=0.

Figure 16-3 Setting the WUPn bit to "1"



## 16.5.5 IICA Low Level Width Setting Register n (IICWLn)

This register controls the SCLAn pin signal low level width ($t_{LOW}$) and the SDAAn pin signal output by the serial interface IICA.

The IICWLn register is set by an 8-bit memory manipulation instruction.

The IICWHn register must be set when $I^2C$ operation is disabled (bit7(IICWHn)=0 of IICA control register n0 (IICCTLn0)).

After a reset signal is generated, the value of this register changes to FFH.

For IICWLn register set-up methods, refer t0 16.6.2 Setting transfer clock via IICWLn and IICWHn registers. The data retention time is 1/4 of the time set by IICWLn.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | IICWLn | IICA low level width setting register | 0xFF |

## 16.5.6 IICA High Level Width Setting Register n (IICWHn)

This register controls the width of the SCLAn pin signal high and the SDAAn pin signal for the serial interface IICA output. The IICWHn register is set by an 8-bit memory manipulation instruction.

The IICWHn register must be set when $I^2C$ operation is disabled (bit7(IICWHn)=0 of IICA control register n0 (IICCTLn0)). After a reset signal is generated, the value of this register changes to “FFH”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | IICWHn | IICA high level width setting register | 0xFF |

Note 1: For the method of setting the master transfer clock, please refer to 16.6.2(1); for how to set the slave IICWLn register and the IICWHn register, refer to 16.6.2(2).

Note 2: n=0.

## 16.5.7 Registers for Controlling Pin Port Functions of IICA

This product allows the IICA pins to be alternated to multiple ports.

Set the corresponding bits in the Port Mode Control Register (PMCxx) and the Port Mode Register (PMxx) to 0.

By configuring the Port Alternate Function Configuration Register (PmnCFG), the SCALn pin and SDAAn pin can be mapped to two different ports.

After these two ports are configured for IICA pin alternating, the N-channel open-drain output mode (VDD/EVDD withstand voltage) is automatically enabled by design, meaning the POMxx register does not need to be configured by the user.

For detailed configuration methods, refer to Chapter 2: Pin Functions.

# 16.6Functions of I²C Bus Mode

## 16.6.1 Pin Structure

The serial clock pin (SCLAn) and the serial data bus pin (SDAAn) are structured as follows.

(1) SCLAn: Input/output pins of serial clock

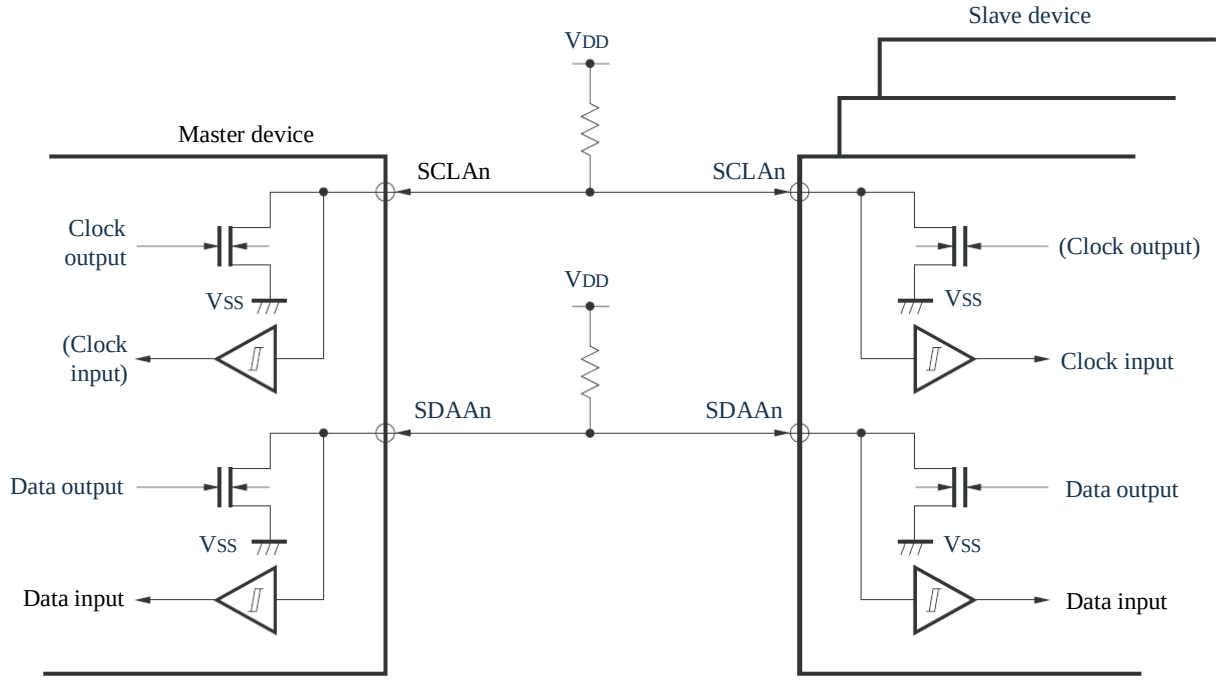
The outputs of both the master and slave devices are N-channel open-drain outputs, and the inputs are Schmitt inputs

(2) SDAAn: Input/output pins of serial data

The outputs of both the master and slave devices are N-channel open-drain outputs, and the inputs are Schmitt inputs.

Because the outputs of the serial clock line and serial data bus are N-channel open-drain outputs, an external pull-up resistor is required.

Figure 16-4 Pin structure diagram




Note: n=0.

## 16.6.2 Setting Transfer Clock via IICWLn and IICWHn Registers

(1) Setting transfer clock on master side

$$\text{Transfer clock}=\frac{F_{MCK}}{IICWL+IICWH+F_{MCK}(T_R+T_F)}$$

At this point, the optimal setpoints for the IICWLn register and the IICWHn register are as follows:
(The fractional parts of all setting values are rounded up.)

① Fast mode

$$IICWLn=\frac{0.52}{\text{transfer clock}}\times F_{MCK}$$

$$IICWHn=(\frac{0.48}{\text{transfer clock}}-t_R-t_F)\times F_{MCK}$$

② Standard mode

$$IICWLn=\frac{0.47}{\text{transfer clock}}\times F_{MCK}$$

$$IICWHn=(\frac{0.53}{\text{transfer clock}}-t_R-t_F)\times F_{MCK}$$

③ Enhanced fast mode

$$IICWLn=\frac{0.50}{\text{transfer clock}}\times F_{MCK}$$

$$IICWHn=(\frac{0.50}{\text{transfer clock}}-t_R-t_F)\times F_{MCK}$$

(2) Setting IICWLn and IICWHn registers on slave side
(The fractional parts of all setting values are rounded up.)

① Fast mode

$IICWLn=1.3\mu s\times F_{MCK}$

$IICWHn=(1.2\mu s-t_R-t_F)\times F_{MCK}$

② Standard mode

$IICWLn=4.7\mu s\times F_{MCK}$

$IICWHn=(5.3\mu s-t_R-t_F)\times F_{MCK}$

③ Enhanced fast mode

$IICWLn=0.50\mu s\times F_{MCK}$

$IICWHn=(0.50\mu s-t_R-t_F)\times F_{MCK}$

Notes:

1. The maximum operating frequency of the IICA operating clock ($F_{MCK}$) is 20MHz (Max.). The bit0 (PRSn) of the IICA control register n1 (IICCTLn1) must be set to 1 only when the $F_{CLK}$ exceeds 20MHz.
2. Note the minimum $F_{CLK}$ operation frequency when setting the transfer clock. The minimum $F_{CLK}$ operation frequency for serial interface IICA is determined according to the mode.

   Fast mode: $F_{CLK}$=3.5MHz(Min.)

   Enhanced fast mode: $F_{CLK}$=10MHz(Min.)

   Standard mode: $F_{CLK}$=1MHz(Min.)
3. Calculate the rise time ($T_R$) and fall time ($T_F$) of the SDAAn and SCLAn signals separately, because they differ depending on the pull-up resistance and wire load.

4. IICWLn: IICA low-level width setting register n

   IICWHn: IICA high-level width setting register n

   $T_F$: SDAAn and SCLAn signal falling times

   $T_R$: SDAAn and SCLAn signal rising times

   $F_{MCK}$: IICA operation clock frequency

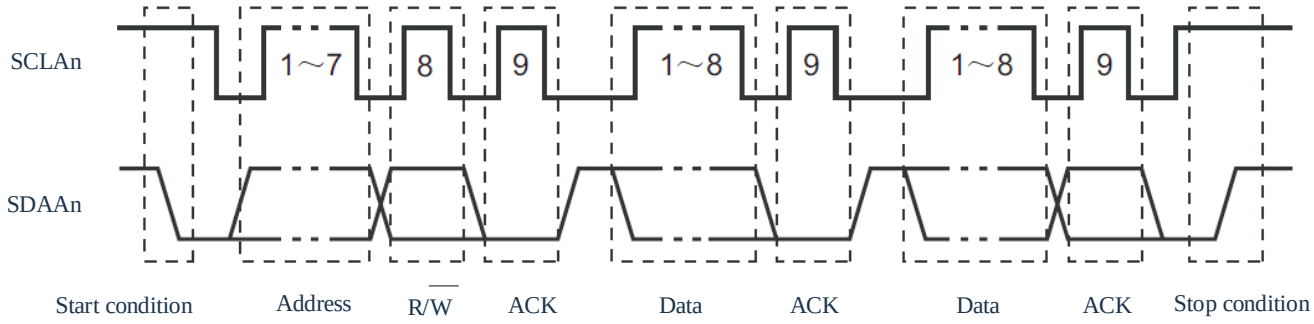
5. n=0

# 16.7Definition and control method of I²C bus

The following section describes the I²C bus's serial data communication format and the signals used by the I²C bus.

The figure below shows the transfer timing for the "start condition", "address", "data", and "stop condition" output via the I²C bus's serial data bus.

Figure 16-5 I²C bus serial data transfer timing



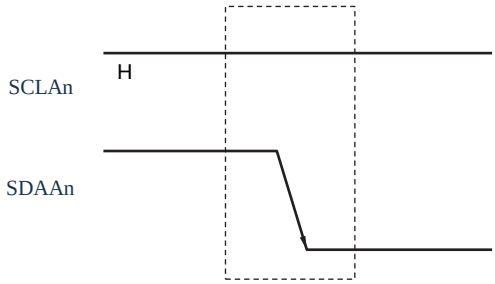

The master generates start conditions, slave addresses, and stop conditions.

Both the master and slave devices can generate an acknowledgement (ACK) (in general, the receiver outputs 8 bits of data). The master device continuously outputs a serial clock (SCLAn). However, the slave can extend the low level period of the SCLAn pin and insert a wait.

### 16.7.1 Start Condition

When the SCLAn pin is high, if the SDAAn pin changes from high to low, a start condition is generated. The SCLAn pin and the SDAAn pin start conditions are the signals generated when the master device starts serial transfer to the slave device. When used as a slave, a start condition is detected.

Figure 16-6 Start condition




If bit1 (STTn) of the IICA control register n0 (IICCTLn0) is set to “1” when a stop condition (SPDn: bit0=1 of the IICA status register n (IICSn)) is detected, the start condition is output. If a start condition is detected, set bit1 (STDn) of the IICsn register to “1”.

Note: n=0.

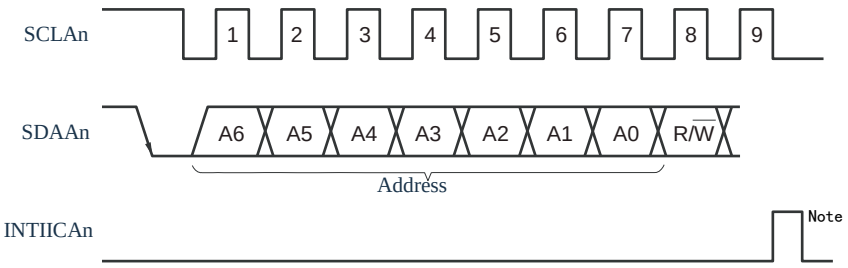
## 16.7.2 Address

The subsequent 7-bit data for the start condition is defined as an address.

The address is 7 bits of data output by the master device for selecting a specific slave device from among a plurality of slave devices connected to the bus. Therefore, the slave device on the bus needs to set a completely different address.

The slave device detects the start condition by hardware and checks whether the 7-bit data is the same as the slave address register n (SVAn). At this time, if the 7-bit data and the SVAn register have the same value, the slave device is selected to communicate with the master device before generating a start condition or a stop condition.

Figure 16-7 Address




Note: If data other than the local station address or extension code is received while the slave is running, INTIICAn is not generated.

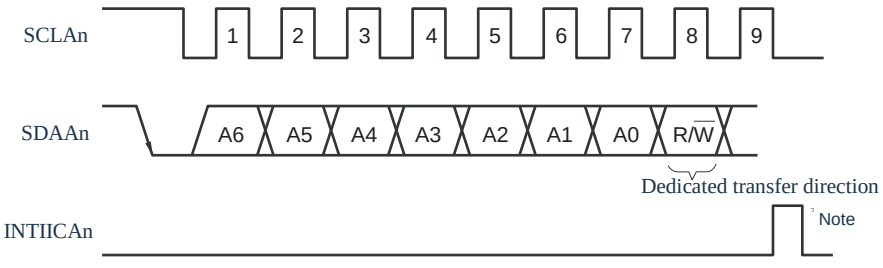
If the 8-bit data consisting of the slave address and the transfer direction described in “16.7.3 Transfer direction specification” is written to the IICA shift register n (IICAn), the address is output. The received address is written to the IICAn register. The slave address is assigned to the higher 7 bits of the IICAn register.

## 16.7.3 Transfer Direction Specification

In addition to the 7-bit address data, the master device sends 1 bit that specifies the transfer direction.

When this transfer direction specification bit is “0”, it indicates that the master device is transmitting data to a slave device. When the transfer direction specification bit is “1”, it indicates that the master device is receiving data from a slave device.

Figure 16-8 Transfer direction specification




Note 1: The INTIICAn is not generated if data other than a local address or extension code is received during slave device operation.

Note 2: n=0.

### 16.7.4 Acknowledge (ACK)

The serial data status of the sender and receiver can be confirmed by an acknowledgement (ACK). The receiver returns an ACK each time it receives an 8-bit data.

Typically, the sender receives an ACK after sending an 8-bit data. When the receiver returns the ACK, it is deemed to have been received normally and continues processing. The bit2 (ACKDn) of the IICA status register n (IICSn) can confirm the detection of the ACK. When the master receives the last data for the received state, a stop condition is generated without returning an ACK. When the slave does not return an ACK after receiving the data, the master device outputs a stop condition or a restart condition to stop the transmission. The reasons why an ACK is not returned are as follows:

① Reception is not performed normally.

② The reception of the last data has been terminated.

③ The receiver specified by the address does not exist.

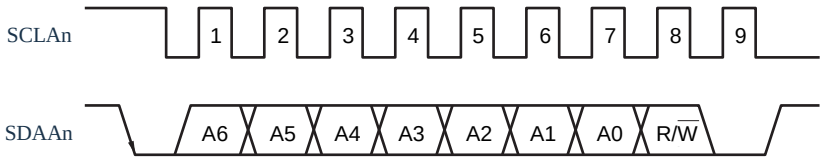
The receiver generates an ACK (normal reception) by setting the SDAAn line low on the 9th clock.

Set bit2 (ACKEn) of IICA control register n0 (IICCTLn0) to 1 to enable automatic generation of an ACK. Set bit3 (TRCn) of the IICsn register with the 8th bit of data following the 7-bit address information. In the case of reception (TRCn=0), the ACKEn bit must be set to 1 in general.

When data cannot be received during slave receive operation (TRCn=0) or the next data is not required, the ACKEn bit must be cleared to 0 to notify the master that data cannot be received.

When the master does not need the next data during reception operation (TRCn=0), the ACKEn bit must be cleared to 0 to notify the slave sender of the end of data (stop sending) in order not to generate an ACK.

Figure 16-9 ACK




When the local address is received, ACK is automatically generated, regardless of the value of the ACKEn bit. When an address other than that of the local address is received, an ACK is not generated (NACK).

When an extension code is received, an ACK is generated if the ACKEn bit is set to 1 in advance. How ACK is generated when data is received differs as follows depending on the setting of the clock stretch timing.

(1) When an 8-clock wait is selected (bit3(WTIMn)=0 in the IICCTLn0 register): an ACK is generated synchronously with the falling edge of the 8th clock on the SCLAn pin by setting the ACKEn bit to 1 before releasing the wait.

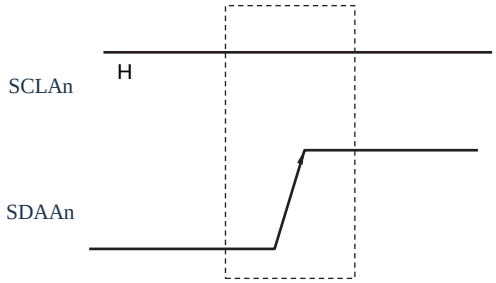
(2) When a 9-clock wait is selected (bit3(WTIMn)=1 in the IICCTLn0 register): an ACK is generated by setting the ACKEn bit to 1 in advance.

Note: n=0.

### 16.7.5 Stop Condition

A stop condition is generated if the SDAAn pin changes from low to high while the SCLAn pin is high. The stop condition is a signal generated when the master device ends serial transfer to the slave device. The stop condition is detected when the device is used as a slave.

Figure 16-10 Stop condition




If the bit0 (SPTn) of the IICA control register n0 (IICTLn0) is set to 1, a stop condition is generated. If a stop condition is detected, the bit0 (SPDn) of the IICA status register n (IICSn) is set to 1 and the INTIICAn is generated when the bit4 (SPIEn) of the IICCTLn0 register is 1.

Note: n=0.

## 16.7.6 Wait

Notify the other party that the master or slave device is preparing data for transmitting/receiving (wait state) by waiting.

Notify the other party that it is in a waiting state by setting the SCLAn pin low. If both the master and slave wait states are released, the next transfer can begin.

Figure 16-11 Wait (1/2)

(1) When master device has a nine-clock wait and slave device has an eight-clock wait

(master transmits, slave receives, and ACKEn = 1)

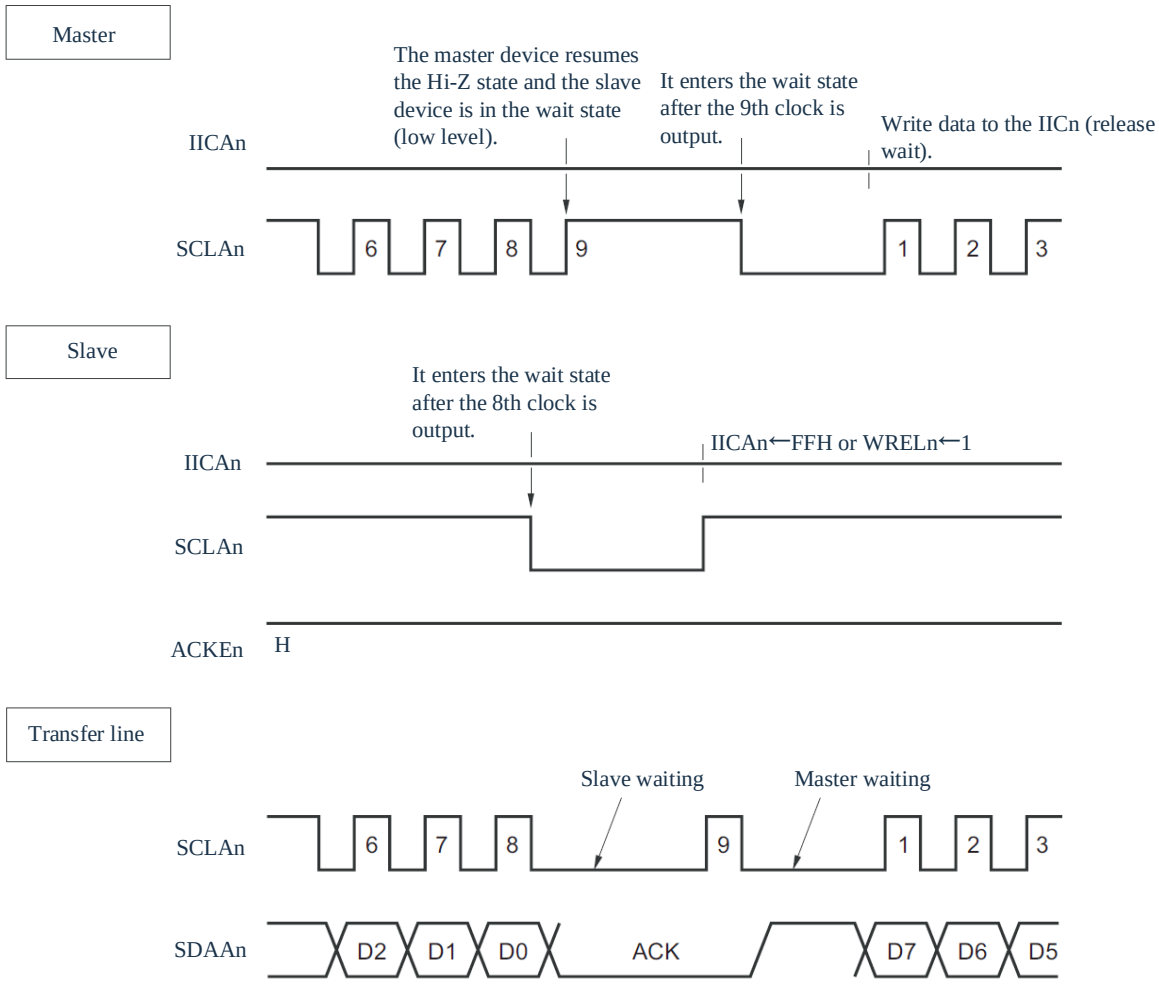
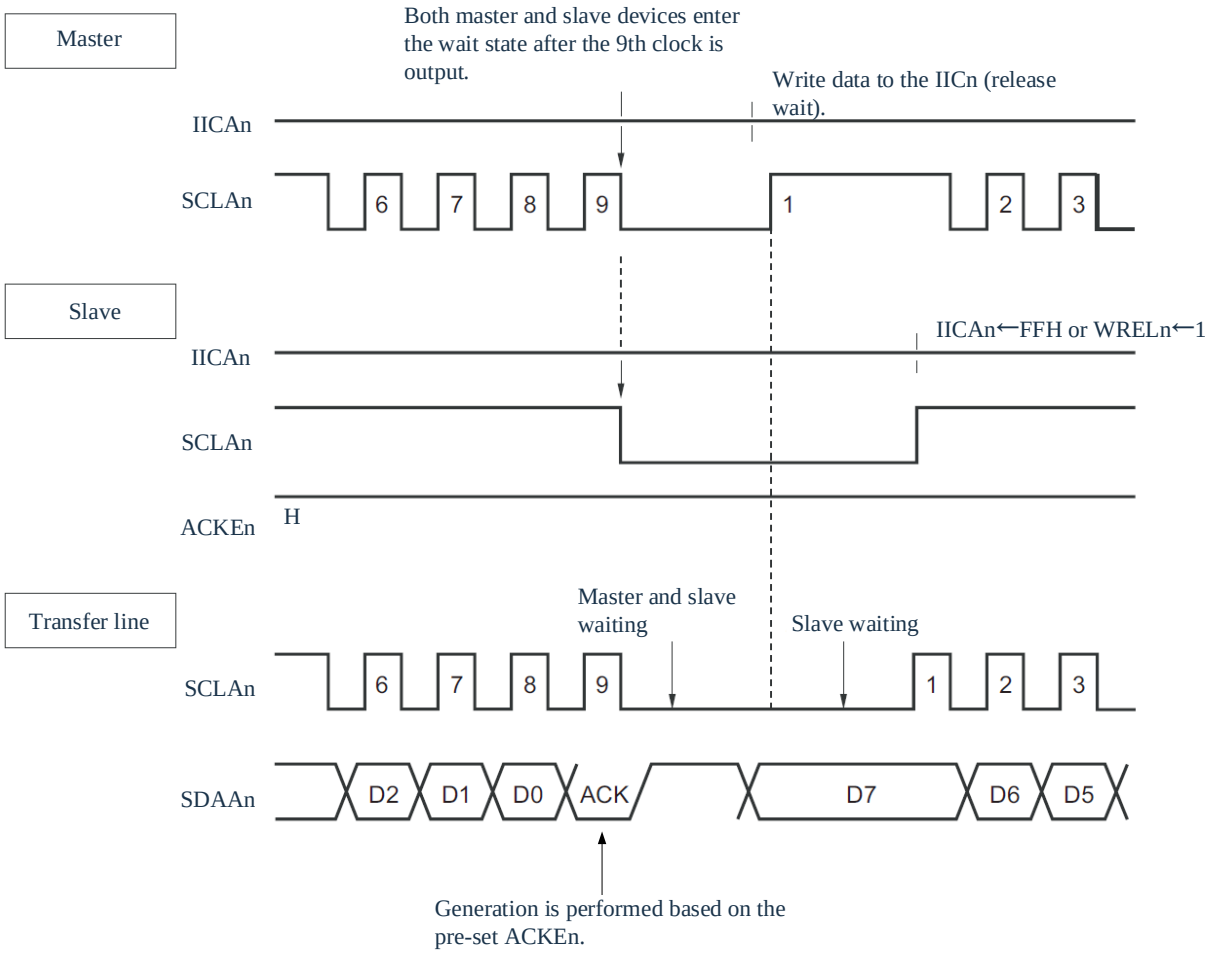


Figure 16-11 Wait (2/2)

(2) When both the master and slave devices are waiting for 9 clocks

(master transmits, slave receives, and ACKEn=1)




Note 1: ACKEn: bit2 of IICA control register n0 (IICCTLn0)

Note 2: WRELn: bit5 of IICA control register n0 (IICCTLn0)

A wait state is automatically generated by setting the bit3 (WTIMn) of the IICA control register n0 (IICCTLn0). In general, at the receiver, if the bit5 (WRELn) of the IICCTLn0 register is 1 or writes “FFH” to the IICA shift register n (IICAn), the wait is released. At the transmitter, if data is written to the IICAn register, the wait is released. The master device can release the wait by:

Setting the bit1 (STTn) of the IICCTLn0 register to 1.

Setting the bit0 (SPTn) of the IICCTLn0 register to 1.

Note: n=0.

### 16.7.7 Wait Release Methods

In general, the I²C can release the wait by the following methods.

(1) Write data to the IICA shift register n(IICAn).

(2) Set the bit5(WRELn) of the IICA control register n0(IICCTLn0) (release wait).

(3) Set the bit1 (STTn) of the IICCTLn0 register (a start condition is generated) Note.

(4) Set the bit0 (SPTn) of the IICCTLn0 register (a stop condition is generated) Note.

Note: Limited to master devices.

If the release processing of these waits is performed, the I²C releases the waits and restarts communication. To send data (including the address) after releasing the wait, you must write data to the IICn register.

To receive data after release from waiting or to end sending data, the bit 5 (WRELn) of the IICCTLn0 register must be set to 1. To generate a restart condition after releasing the wait, the bit 1 (STTn) of the IICCTLn0 register must be set to 1. To generate a stop condition after releasing a wait, the bit 0 (SPTn) of the IICCTLn0 register must be set to 1. Only one release process can be performed for one wait.

For example, if data is written to the IICAn register after the wait is released by setting the WRELn bit to 1, the change timing of the SDAAn line may conflict with the write timing of the IICAn register, resulting in the wrong value being output to the SDAAn line. In addition to these processes, if the IICEn bit is cleared to 0 when stopping communications in the middle of the communication, communication is stopped, so that waiting can be released. If the I²C bus state is deadlocked due to noise, if the bit 6 (LRELn) of the IICCTLn0 register is set to 1, communication is exited, and thus waiting is released.

Note 1: If the wait release process is performed when the WUPn bit is 1, the wait will not be released.

Note 2: n=0.

## 16.7.8 Generation Timing and Waiting Control of Interrupt Requests (INTIICAn)

By setting the bit3 (WTIMn) of the IICA control register n0 (IICCTLn0), the INTIICAn is generated at the timing shown in Table 16-3 and wait control is performed.

Table 16-3 Generation timing and waiting control of INTIICAn

| WTIMn | Slave operation | | | Master operation | | |
|---|---|---|---|---|---|---|
| | Address | Data reception | Data transmission | Address | Data reception | Data transmission |
| 0 | 9Note1, 2 | 8Note2 | 8Note2 | 9 | 8 | 8 |
| 1 | 9Note1, 2 | 9Note2 | 9Note2 | 9 | 9 | 9 |

Note 1: Only when the received address and the set address of the slave address register n(SVAn) are the same, the slave generates an INTIICAn signal on the falling edge of the 9th clock and enters a waiting state. At this point, regardless of the bit2 (ACKEn) setting of the IICCTLn0 register, an ACK is generated. The slave that receives the extension code generates INTIICAn on the falling edge of the 8th clock. If the addresses are different after restarting, the INTIICAn is generated on the falling edge of the 9th clock, but does not enter the waiting state.

Note 2: If the contents of the received address and the slave address register n(SVAn) are different and the extension code is not received, the INTIICAn is not generated and does not enter the waiting state.

Note 3: The numbers in the table represent the number of clocks for a serial clock. Both interrupt request and wait control are synchronized with the falling edge of the serial clock.

(1) Address transmission and reception

① Slave operation: Regardless of the WTIMn bit, the timing of interrupts and waits is determined according to the conditions in Note 1 and Note 2 above.

② Master operation: Regardless of the WTIMn bit, the timing of interrupts and waits is generated on the falling edge of the 9th clock.

(2) Data reception

Master/slave operation: Determines the timing of interrupts and waits via the WTIMn bit.

(3) Data transmission

Master/slave operation: Determines the timing of interrupts and waits via the WTIMn bit.

Note: n=0.

(4) Release waiting

There are 4 ways to release from waiting:

① Writes data to IICA shift register n (IICAn).

② Sets the bit5 (WRELn) of the IICA control register n0 (IICCTLn0) (wait released).

③ Sets the bit1 (STTn) of the IICCTLn0 register (a start condition is generated) Note.

④ Sets the bit0 (SPTn) of the IICCTLn0 register (a stop condition is generated) Note.

Note 1: Limited to master devices.

Note 2: When you select a wait for 8 clocks (WTIMn=0), you need to decide whether to generate an ACK before you release the wait.

(5) Detection of stop condition

If a stop condition is detected, the INTIICAn is generated (only when SPIEn=1).

### 16.7.9 Detection for Address Matching

In I²C-bus mode, the master device can select a specific slave by sending a slave address. Address matching can be automatically detected by hardware. When the slave address sent by the master device and the set address of the slave address register n(SVAn) are the same or only the extension code is received, an INTIICAn interrupt request is generated.

### 16.7.10 Error Detection

In I²C bus mode, because the state of the serial data bus (SDAAn) is captured into the transmitter's IICA shift register n (IICAn) during the transmission process, it is possible to detect transmission errors by comparing the data in the IICA before the transmission starts and after it ends. If the two sets of data are different, it is considered that a transmission error has occurred.

Note: n=0.

## 16.7.11 Extension Code

(1) When the high 4 bits of the receiving address are “0000” or “1111”, as the received extension code, the extended code receive flag (EXCn) is set to 1, and in the 8th The falling edge of the clock generates an interrupt request (INTIICAn).

Does not affect local station addresses stored in slave address register n (SVAn).

(2) When the SVAn register is set to “11110xx0”, if “11110xx0” is sent from the master device via a 10-bit address, the following assertion occurs. However, an interrupt request (INTIICAn) is generated on the falling edge of the 8th clock.

① High 4 bits data are the same: EXCn=1

② 7 bits of data are the same: COIn=1

Remark:

1. EXCn: Bit5 of the IICA status register n
2. COIn: Bit4 of IICA status register n (IICSn)

(3) Since the processing after the interrupt request occurs differs according to the data that follows the extension code, such processing is performed by software. If the extension code is received while a slave device is operating, then the slave device is participating in communication even if its address does not match. For example, after the extension code is received, if you do not wish to operate the target device as a slave device, set bit 6 (LRELn) of IICA control register n0 (IICCTLn0) to 1 to set the standby mode for the next communication operation.

Table 16-4 Bit definitions of major extension codes

| Slave address | R/W bit | Definition |
|---|---|---|
| 0000000 | 0 | Full call address |
| 11110xx | 0 | 10-bit slave address specification (during address authentication) |
| 11110xx | 1 | 10-bit slave address specification (after address match, when read command is issued) |

Note 1: See the $I^2C$ bus specifications issued by NXP Semiconductors for details of extension codes other than those described above.

Note 2: n=0.

## 16.7.12 Arbitration

When multiple master devices generate start conditions at the same time (Set STTn bit to 1 before the STDn bit becomes 1), the communication of the master device is carried out while adjusting the clock until the data is different. This operation is called arbitration.

When arbitration fails, the master device that has failed arbitration sets the arbitration failure flag (ALDn) in the IICA status register n (IICSn) to 1 and sets both the SCLAn line and the SDAAn line to a high impedance state, releasing the bus.

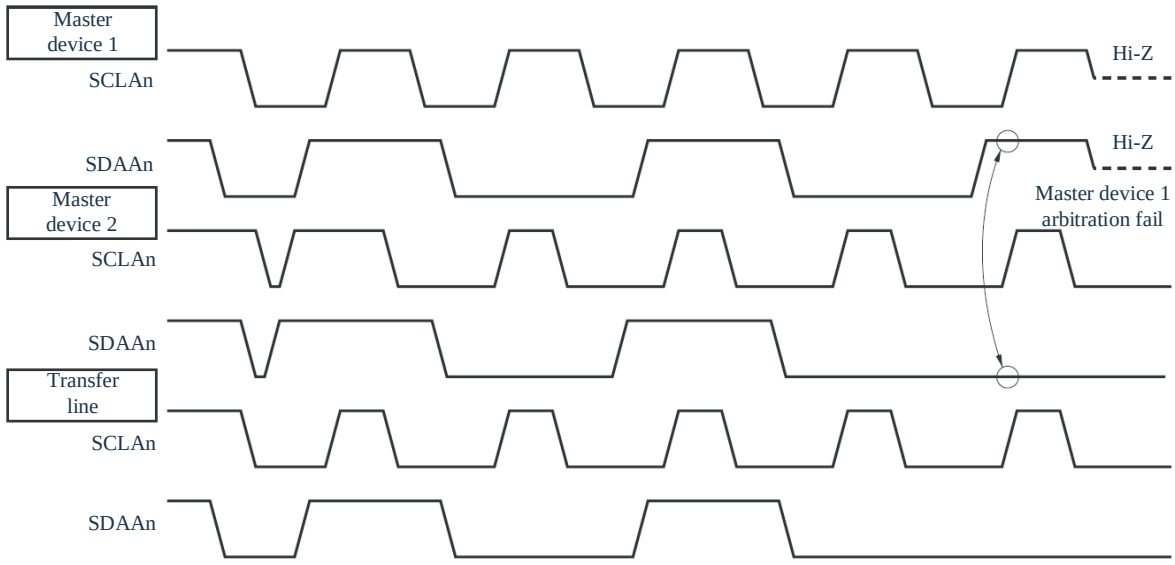
When the next interrupt request occurs (e.g., a stop condition is detected on the 8th or 9th clock), the failure of arbitration is detected by setting the ALDn bit to 1 in software.

For the timing of interrupt requests, please refer to "16.7.8 Generation timing and waiting control of interrupt requests (INTIICAn)".

Note 1: STDn: bit1 of IICA status register n(IICSn)

Note 2: STTn: bit1 of IICA control register n0 (IICCTLn0)

Figure 16-12 Timing example of arbitration




Note: n=0.

Table 16-5 Status at the time of arbitration and timing of generation of interrupt requests

| The state in which the arbitration occurred | Timing of the generation of interrupt requests |
|---|---|
| During address transmission | On the falling edge of the 8th or 9th clock after the byte is transmitted Note1 |
| Reads and writes information after address transmission | |
| During address extension codes transmission | |
| Reads and writes information after extension codes transmission | |
| During data transmission | |
| During ACK transmission after transmitted data | |
| Restart condition detected during data transmission | |
| Stop condition detected during data transmission | When generating a stop condition (SPIEn=1) Note2 |
| Trying to generate a restart condition, but the data is low. | On the falling edge of the 8th or 9th clock after the byte is transmitted Note1 |
| Trying to generate a restart condition, but a stop condition was detected. | When generating a stop condition (SPIEn=1) Note2 |
| Trying to generate a stop condition, but the data is low. | On the falling edge of the 8th or 9th clock after the byte is transmitted Note1 |
| Trying to generate a restart condition, but SCLAn is low. | |

Note 1: The interrupt request is generated on the falling edge of the 9th clock when the WTIMn bit (bit3 of the IICA control register n0 (IICCTLn0)) is 1, and on the falling edge of the 8th clock when the WTIMn bit is "0" and the slave address of the extended code is received.

Note 2: When there is a possibility of arbitration, the SPIEn bit must be 1 when the master is operating.

Note 3: SPIEn: bit4 of IICA control register n0 (IICTLn0)

Note 4: n=0.

## 16.7.13 Wake-Up Function

This is a slave function of $I^2C$, which is the function of generating an interrupt request signal (INTIICAn) when the local station address and extension code are received. The processing efficiency is improved by not generating unwanted INTIICAn signals under different addresses. If a start condition is detected, it enters wake-up standby. Because the master device (where a start condition has already been generated) may also become a slave due to an arbitration failure, it enters wake-up standby at the same time as the address is sent.

To use the wake function in deep sleep mode, you must place the WUPn at 1. The address can be received independent of the operating clock. Even in this case, an interrupt request signal (INTIICAn) is generated when the local station address and extension code are received. After this interrupt is generated, the WUPn bit is cleared to "0" by the instruction and returned to the normal operation.

The flow when the WUPn bit is set to 1 is shown in Figure 16-13, and the flow when the WUPn bit is set to 0 by address matching is shown in Figure 16-14.

Figure 16-13 Setting the WUPn bit to 1

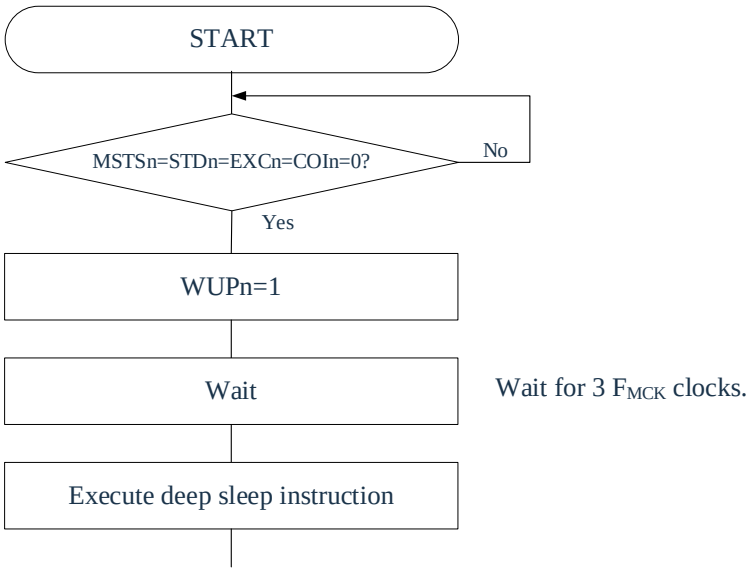
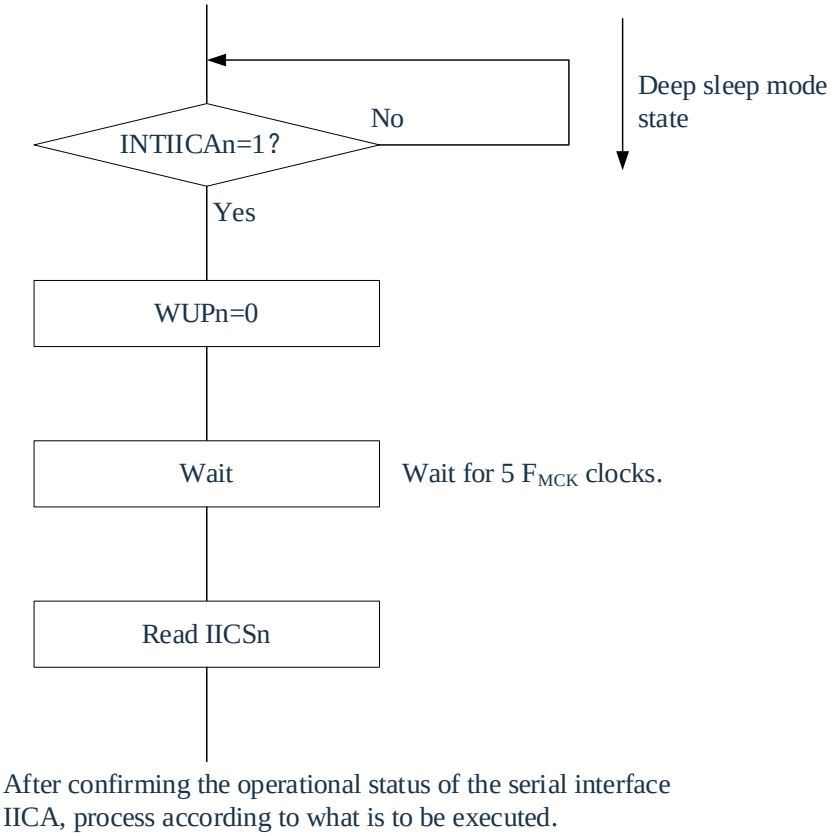


Figure 16-14 Flow when the WUPn bit is set to 0 by address matching (including receiving extension codes)




In addition to the interrupt request (INTIICAn) generated by the serial interface IICA, the deep sleep mode must be removed through the following procedure.

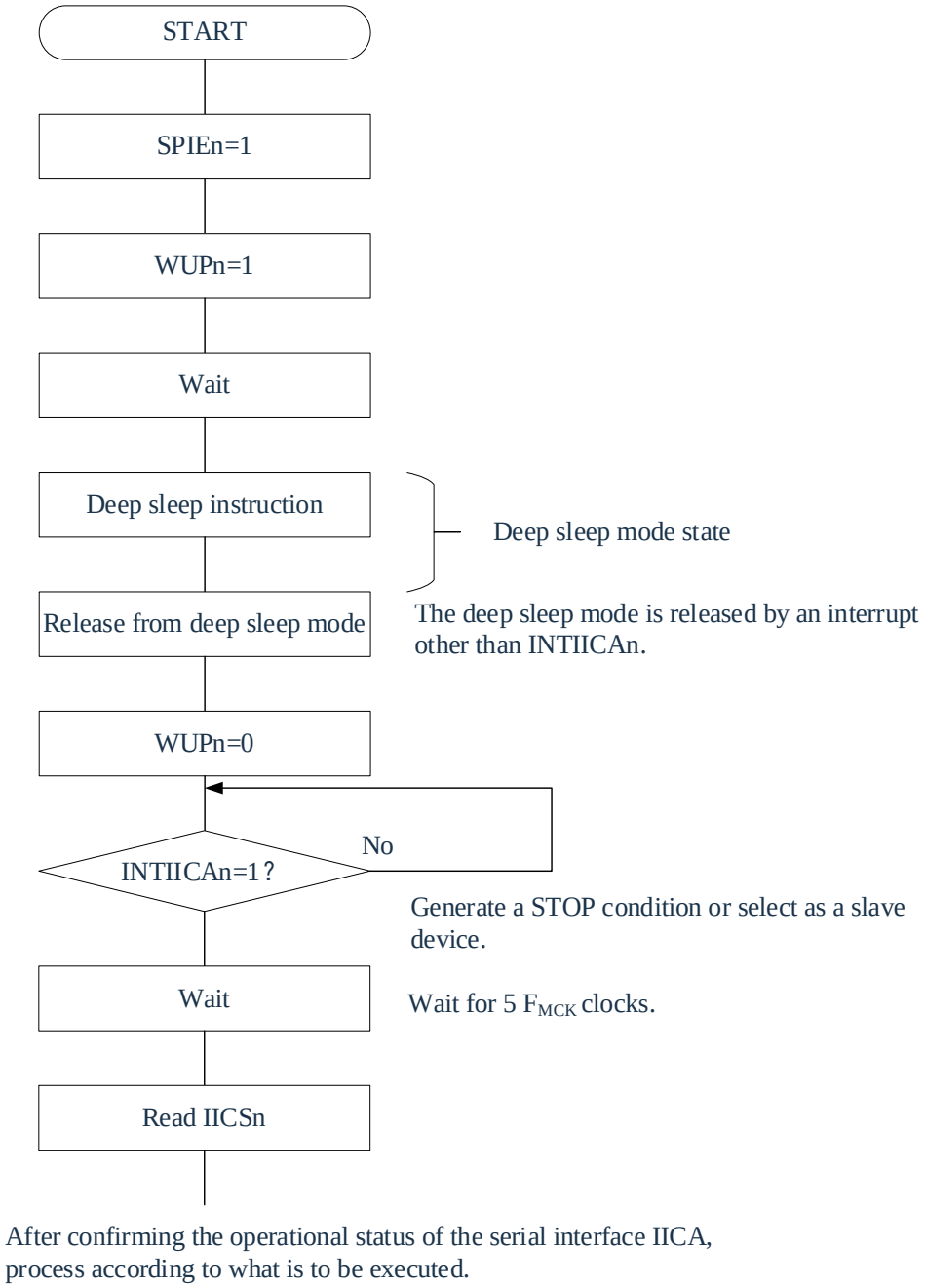
- Next IIC communication for the operation of the master device: the flow of Figure 16-14
- Next IIC communication for slave device operation:

Return via INTIICAn interrupt: same process as Figure 16-15.

Return from an interrupt other than the INTIICAn interrupt: operation must be continued with the WUPn bit set to 1 before the INTIICAn interrupt is generated.

Note: n=0.

Figure 16-15 Operation as master device after being released from deep sleep mode by an interrupt other than INTIICAn




Note: n=0.

## 16.7.14 Communication Reservation

(1) When communication reservation function is enabled (bit0 (IICRSVn)=0 of the IIC flag register n (IICFn))

To perform the next master communication without using the bus, you can send a start condition when the bus is released through a communication reservation. There are two modes under which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled (ACK is not returned and the bus was released by setting bit 6 (LRELn) of IICA control register n0 (IICCTLn0) to 1 and saving communication).

If bit 1 (STTn) of the IICCTLn0 register is set to 1 while the bus is not used (after a stop condition is detected), a start condition is automatically generated and wait state is set.

If an address is written to the IICA shift register n (IICAn) after bit 4 (SPIEn) of the IICCTLn0 register was set to 1, and it was detected by generation of an interrupt request signal (INTIICAn) that the bus was released (detection of the stop condition), then the device automatically starts communication as the master. Data written to the IICAn register before the stop condition is detected is invalid.

When the STTn bit has been set to 1, the operation mode (as start condition or as communication reservation) is determined according to the bus status.

- Generate a start condition when the bus is released.
- Communication reservation when the bus is not released (standby state)

After the STTn bit is set to “1” and a wait time has elapsed, operation as a communication reservation is confirmed by the MSTSn bit (bit 7 of the IICA status register n (IICSn)).

Waiting times must be ensured by software for the following formula calculations.

| |
|---|
| Wait time from setting STTn bit to 1 until the MSTSn flag is confirmed:<br>(IICWLn set value+ IICWHn set value+ 4) / $F_{MCK}$ + $T_F \times 2$ |

Note 1: IICWLn : IICA low-level width setting register n

Note 2: IICWHn: IICA high-level width setting register n

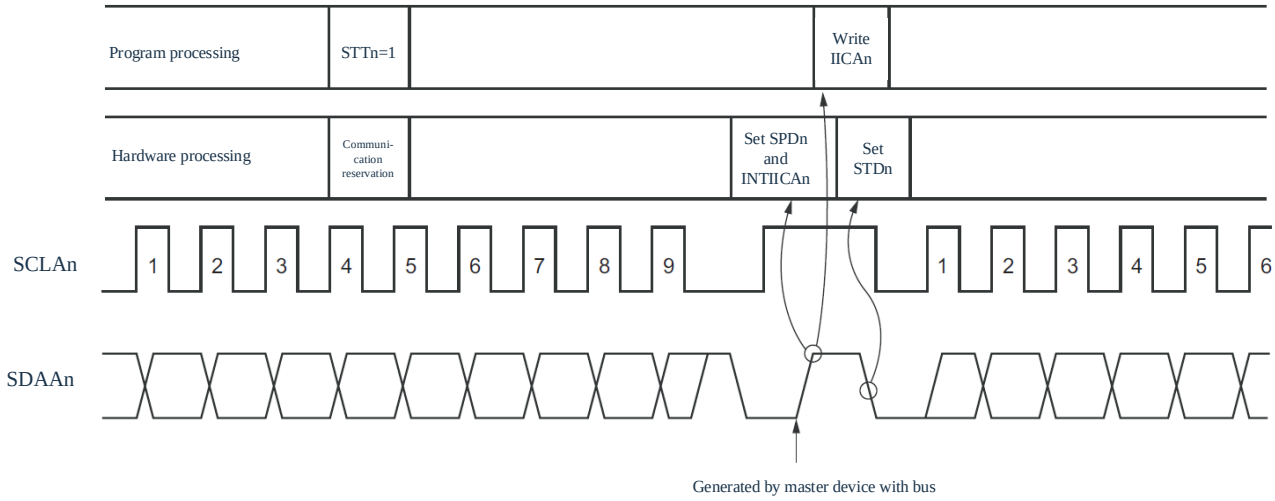
Note 3: $t_F$: SDAAn and SCLAn signal falling times

Note 4: $f_{MCK}$: IICA operation clock frequency

Note 5: n=0

The timing of the communication reservation is shown in the figure below.

Figure 16-16 Timing of communication reservation




Note 1: IICAn: IICA shift register n.

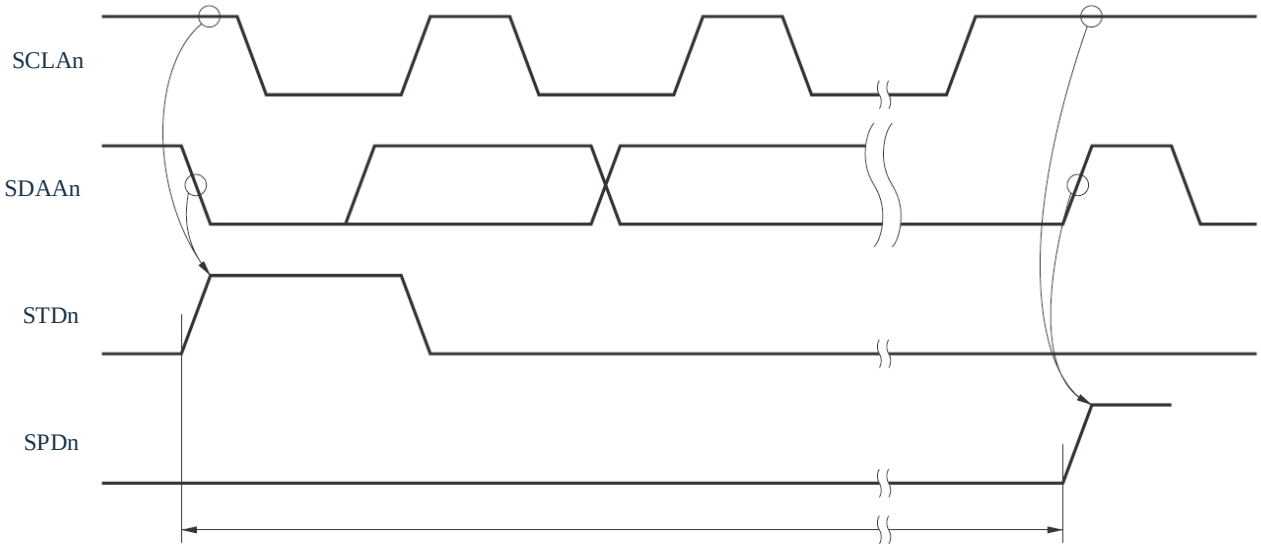
Note 2: STTn: bit1 of IICA control register n0 (IICCTLn0)

Note 3: STDn: bit1 of IICA status register n (IICSn)

Note 4: SPDn: bit0 of IICA status register n (IICSn)

The communication reservation is accepted by the timing sequence shown in Figure 16-17. After the bit1 (STDn) of the IICA status register n (IICSn) becomes 1 and before a stop condition is detected, the bit1 (STTn) of the IICA control register n0 (IICCTLn0) is set to 1 to make a communication reservation.

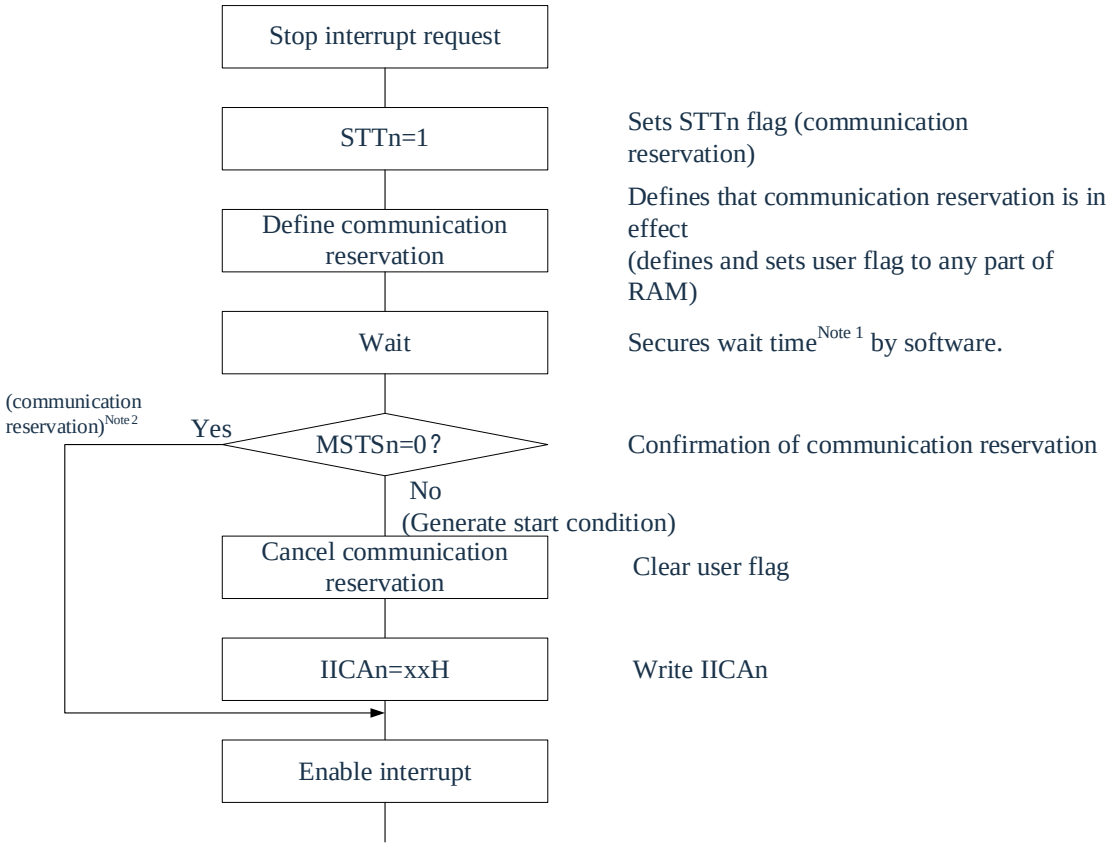
Figure 16-17 Timing of accepting communication reservations




Note: n=0

The steps for communication reservations are shown in Figure 16-18.

Figure 16-18 Steps for communication reservation




Note 1: The wait time is calculated as follows. (IICWLn set value+IICWHn set value +4)/$F_{MCK}$+$T_F$.

Note 2: The communication reservation operation executes a write to the IICA shift register n (IICAn) when a stop condition interrupt request occurs.

Note 3: STTn: Bit1 of IICA control register n0 (IICCTLn0).

Note 4: MSTSn: Bit 7 of IICA status register n (IICSn).

Note 5: IICAn: IICA shift register n.

Note 6: IICWLn: IICA low-level width setting register n.

Note 7: IICWHn: IICA high-level width setting register n.

Note 8: $t_F$: SDAAn and SCLAn signal falling times.

Note 9: $f_{MCK}$: IICA operation clock frequency.

Note 10: n=0.

(2) When communication reservation function is disabled (bit 0 (IICRSVn) of IICA flag register n (IICFn) = 1)

When bit 1 (STTn) of IICA control register n0 (IICCTLn0) is set to 1 when the bus is not used in a communication during bus communication, this request is rejected and a start condition is not generated. The following two statuses are included in the status where bus is not used.

① When arbitration results in neither master nor slave operation

② When an extension code is received and slave operation is disabled (ACK is not returned and the bus was released by setting bit 6 (LRELn) of the IICCTLn0 register to 1 and saving communication)

To confirm whether the start condition was generated or request was rejected, check STCFn (bit 7 of the IICFn register). It takes up to 5 cycles of $F_{MCK}$ until the STCFn bit is set to 1 after setting STTn = 1. Therefore, secure the time by software.

Note: n=0.

## 16.7.15 Cautions

(1) When STCENn = 0

Immediately after $I^2C$ operation is enabled (IICEn = 1), the bus communication status (IICBSYn = 1) is recognized regardless of the actual bus status. When changing from a mode in which no stop condition has been detected to a master device communication mode, first generate a stop condition to release the bus, then perform master device communication. When using multiple masters, it is not possible to perform master device communication when the bus has not been released (when a stop condition has not been detected). Use the following sequence for generating a stop condition.

① Set IICA control register n1 (IICCTLn1).

② Set bit 7 (IICEn) of IICA control register n0 (IICCTLn0) to 1.

③ Set bit 0 (SPTn) of the IICCTLn0 register to 1.

(2) When STCENn = 1

Immediately after $I^2C$ operation is enabled (IICEn = 1), the bus released status (IICBSYn = 0) is recognized regardless of the actual bus status. To generate the first start condition (STTn = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

(3) If other $I^2C$ communications are already in progress

If $I^2C$ operation is enabled and the device participates in communication already in progress when the SDAAn pin is low and the SCLAn pin is high, the macro of $I^2C$ recognizes that the SDAAn pin has gone low (detects a start condition). If the value on the bus at this time can be recognized as an extension code, ACK is returned, but this interferes with other $I^2C$ communications. To avoid this, start $I^2C$ in the following sequence.

① Clear bit 4 (SPIEn) of the IICCTLn0 register to 0 to disable generation of an interrupt request signal (INTIICAn) when the stop condition is detected.

② Set bit 7 (IICEn) of the IICCTLn0 register to 1 to enable the operation of $I^2C$.

③ Wait for detection of the start condition.

④ Set bit 6 (LRELn) of the IICCTLn0 register to 1 before ACK is returned (4 to 72 cycles of $F_{MCK}$ after setting the IICEn bit to 1), to forcibly disable detection.

(4) Setting the STTn and SPTn bits (bits 1 and 0 of the IICCTLn0 register) again after they are set and before they are cleared to 0 is prohibited.

(5) When transmission is reserved, set the SPIEn bit (bit 4 of the IICCTLn0 register) to 1 so that an interrupt request is generated when the stop condition is detected. Transfer is started when communication data is written to the IICA shift register n (IICAn) after the interrupt request is generated. Unless the interrupt is generated when the stop condition is detected, the device stops in the wait state because the interrupt request is not generated when communication is started. However, it is not necessary to set the SPIEn bit to 1 when the MSTSn bit (bit 7 of the IICA status register n (IICSn)) is detected by software.

Note: n=0.

## 16.7.16 Communication Operation

The following shows three operation procedures with flowcharts.

(1) Master operation in single master system

The flowchart when using as the master in a single master system is shown below.

This flowchart is broadly divided into the initial settings and communication processing. Execute the initial settings at startup. If communication with the slave is required, prepare the communication and then execute communication processing.

(2) Master operation in multimaster system

In a multi-master system of the $I^2C$ bus, it is not possible to judge whether the bus is in the release state or in use during the stage of participating in the communication based on the specifications of the $I^2C$ bus. Here, if the data and clock are high for a certain amount of time (1 frame), the bus participates in communication as a release state. This process is roughly divided into “initial settings”, “communication waiting” and “communication processing”. The processing designated as a slave due to the failure of the arbitration is omitted here, and only the processing used as the master device is omitted. Join the bus after performing the Initial Setup section at startup, and then wait for a communication request from the master device or a designation of the slave device via Communication Wait. The actual communication is the “Communication Processing” section, which supports arbitration with other master devices in addition to data sending and receiving with the slave.

(3) Slave operation

An example of an $I^2C$ bus slave is shown below.

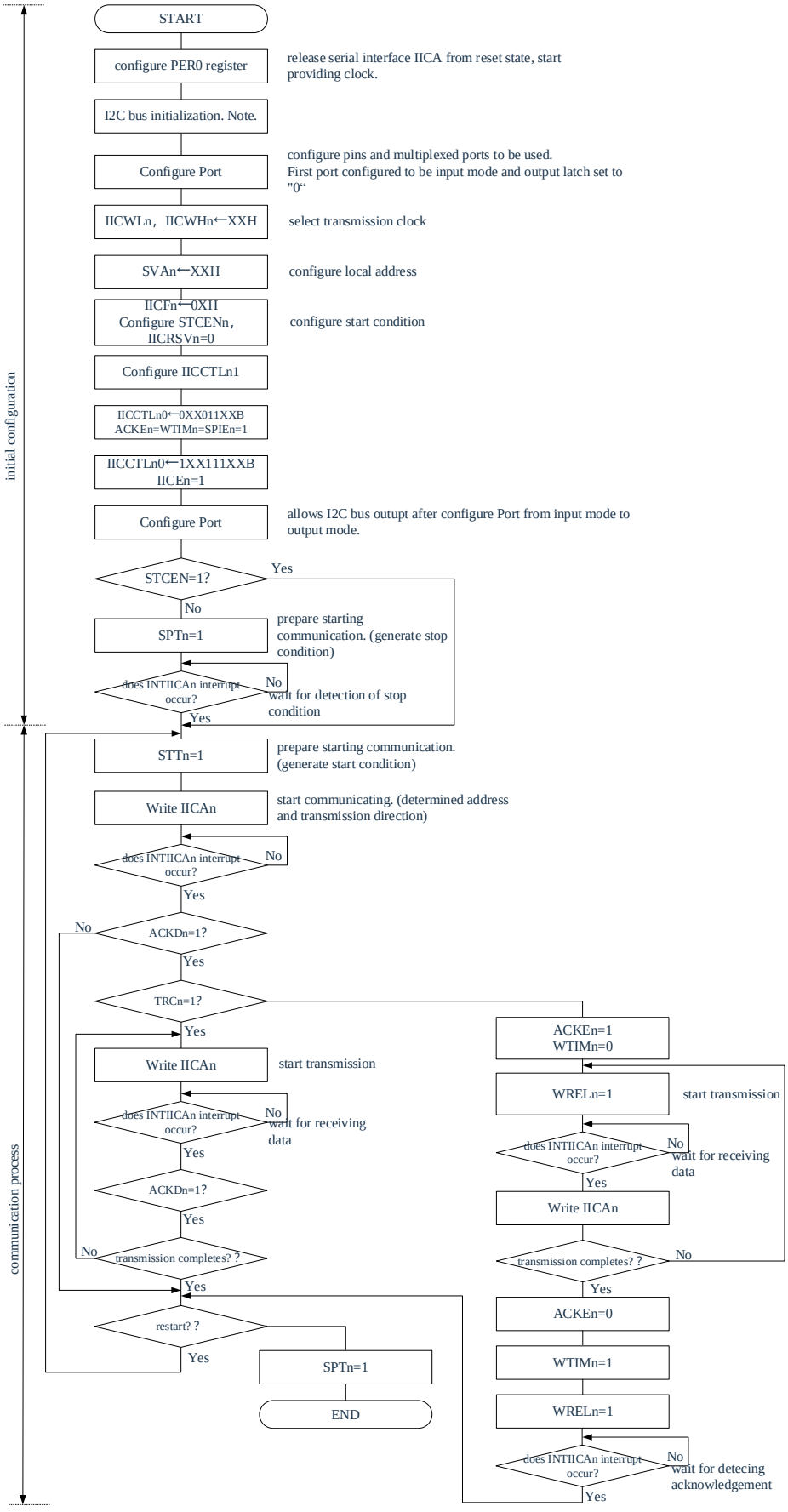
When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIICAn interrupt occurrence (communication waiting). When an INTIICAn interrupt occurs, the communication status is judged and its result is passed as a flag over to the main processing.

By checking the flags, necessary communication processing is performed.

Note: n=0.

(1) Master operation in single-master system

Figure 16-19 Master operation of single-master system



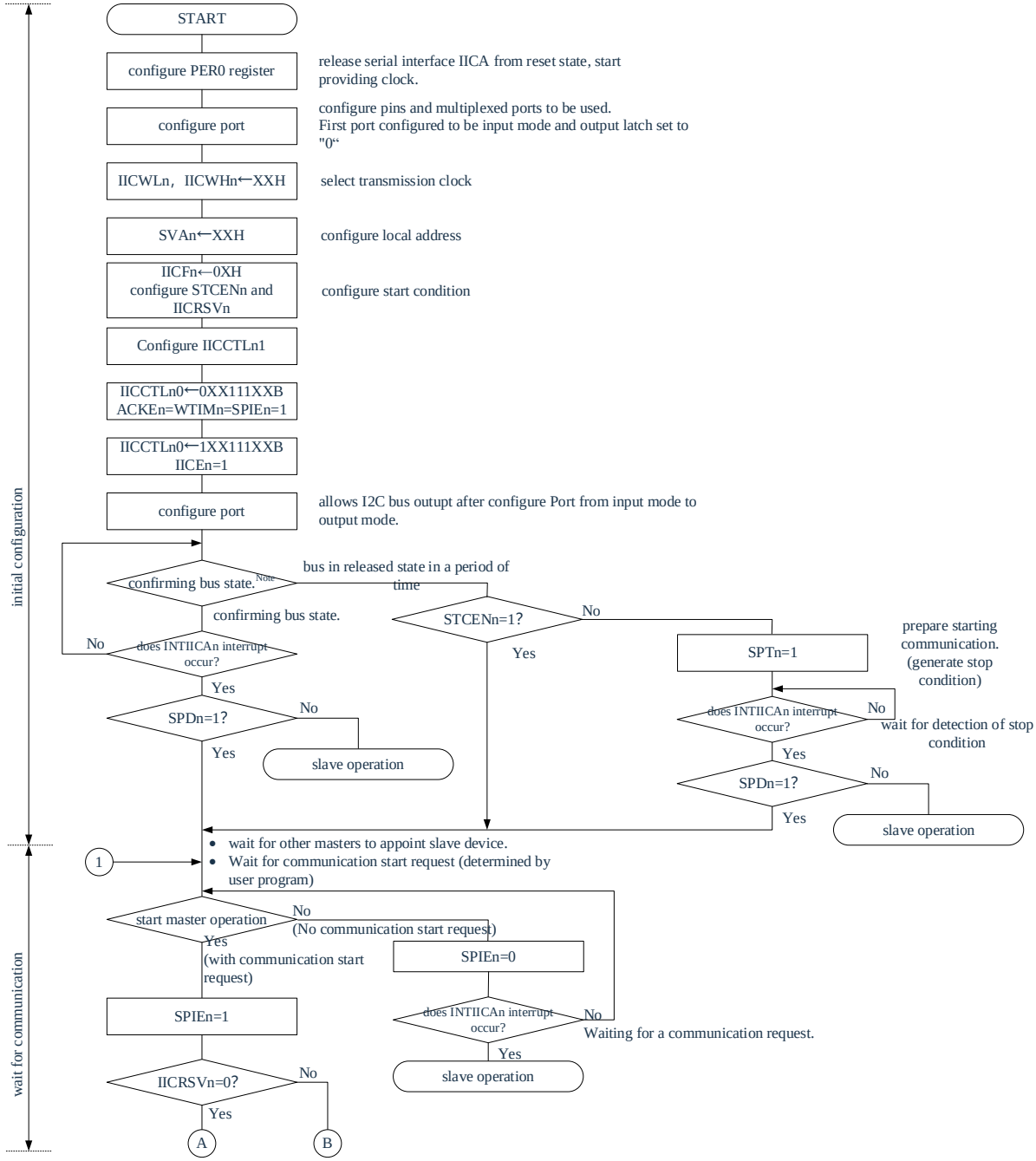
Note 1: I²C-bus must be released (SCLAn pins and SDAAn pins are high) depending on the specifications of the product in communication. For example, if the EEPROM is in a state that outputs a low level to the SDAAn pin, the SCLAn pin must be set to the output port and a clock pulse must be output from the output port before the SDAAn pin is fixed high.

Note 2: The format of transmission and reception must conform to the specifications of the product in communication.

Note 3: n=0.

(2) Master operation in multi-master system

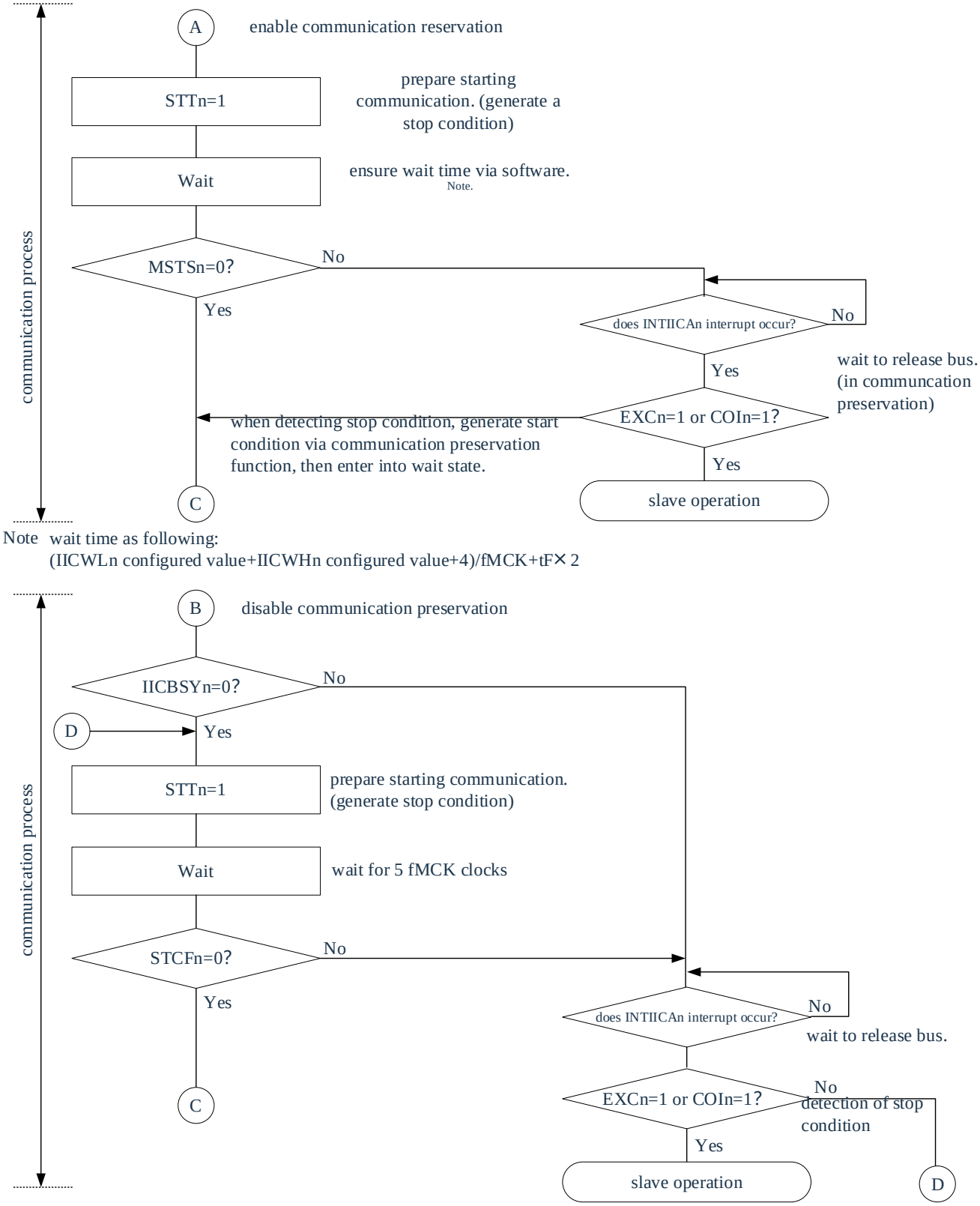
Figure 16-20 Master operation in multi-master system (1/3)




Note 1: Confirm that the bus is released (CLDn bit = 1, DADn bit = 1) for a specific period (for example, for a period of one frame). If the SDAAn pin is constantly at low level, decide whether to release the I2C bus (SCLAn and SDAAn pins = high level) in conformance with the specifications of the product that is communicating.

Note 2: n=0.

Figure 16-20 Master operation in multi-master system (2/3)




Note wait time as following:
(IICWLn configured value+IICWHn configured value+4)/fMCK+tF× 2

Note 1: IICWLn : IICA low-level width setting register n

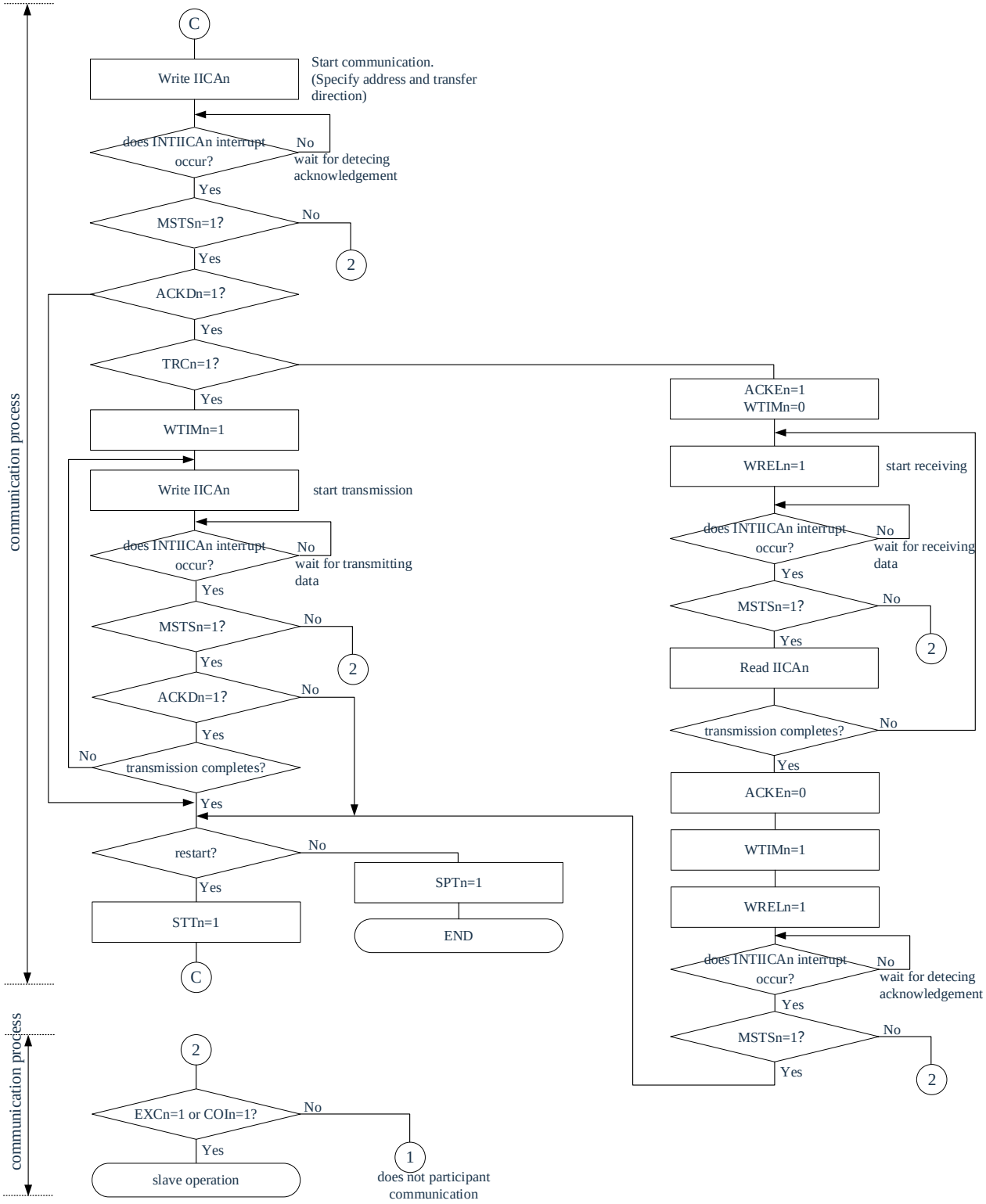
Note 2: IICWHn: IICA high-level width setting register n

Note 3: $t_F$: SDAAn and SCLAn signal falling times

Note 4: $f_{MCK}$: IICA operation clock frequency

Note 5: n=0.

Figure 16-20 Master operation in multi-master system (3/3)




Note 1: The format of transmission and reception must conform to the specifications of the product in communication.

Note 2: When used as a master device in a multi-master system, the MSTSn bit must be read each time an INTIICAn interrupt occurs to acknowledge the arbitration result.

Note 3: When used as a slave device in a multi-master system, the status must be confirmed each time an INTIICAn interrupt occurs by means of the IICA status register n (IICSn) and the IICA flag register n (IICFn) to determine subsequent processing.

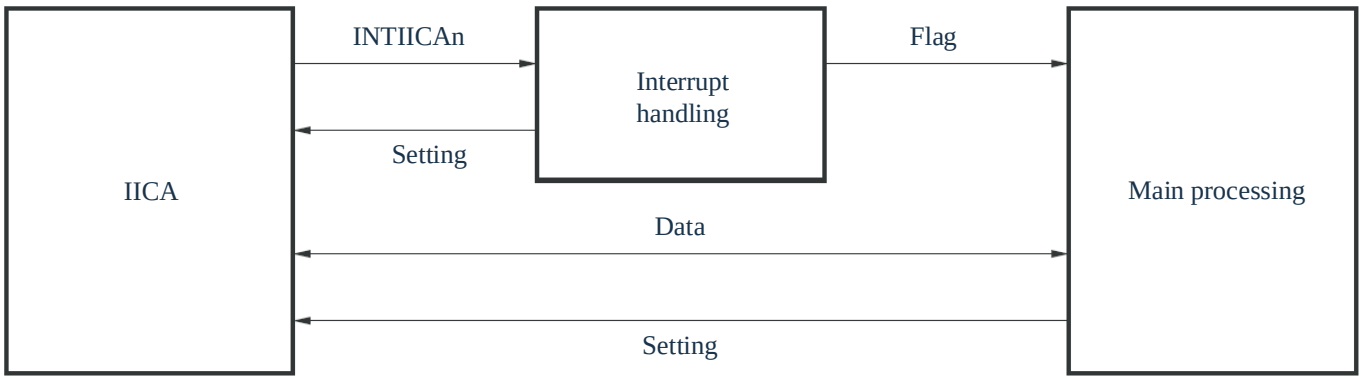
Note 4: n=0.

(3) Slave operation

The processing procedure of the slave operation is as follows.

Basically, the slave operation is event-driven. Therefore, processing by the INTIICAn interrupt (processing that must substantially change the operation status such as detection of a stop condition during communication) is necessary.

In the following explanation, it is assumed that the extension code is not supported for data communication. It is also assumed that the INTIICAn interrupt servicing only performs status transition processing, and that actual data communication is performed by the main processing.




Therefore, the following three flags are prepared and pass the flags to the main processing department instead of INTRAICAn for data communication processing.

① Communication mode flag

This flag indicates the following 2 communication states:

- Clear mode: Status in which data communication is not performed
- Communication mode: Status in which data communication is performed (from valid address detection to stop condition detection, no detection of ACK from master, address mismatch)

② Ready flag

This flag indicates that data communication is enabled. Its function is the same as the INTIICAn interrupt for ordinary data communication. This flag is set by interrupt handling and cleared by the main processing. Clear this flag by interrupt servicing when communication is started. However, the ready flag is not set by interrupt servicing when the first data is transmitted. Therefore, the first data is transmitted without the flag being cleared (an address match is interpreted as a request for the next data).

③ Communication direction flag

This flag indicates the direction of communication. Its value is the same as the TRCn bit.

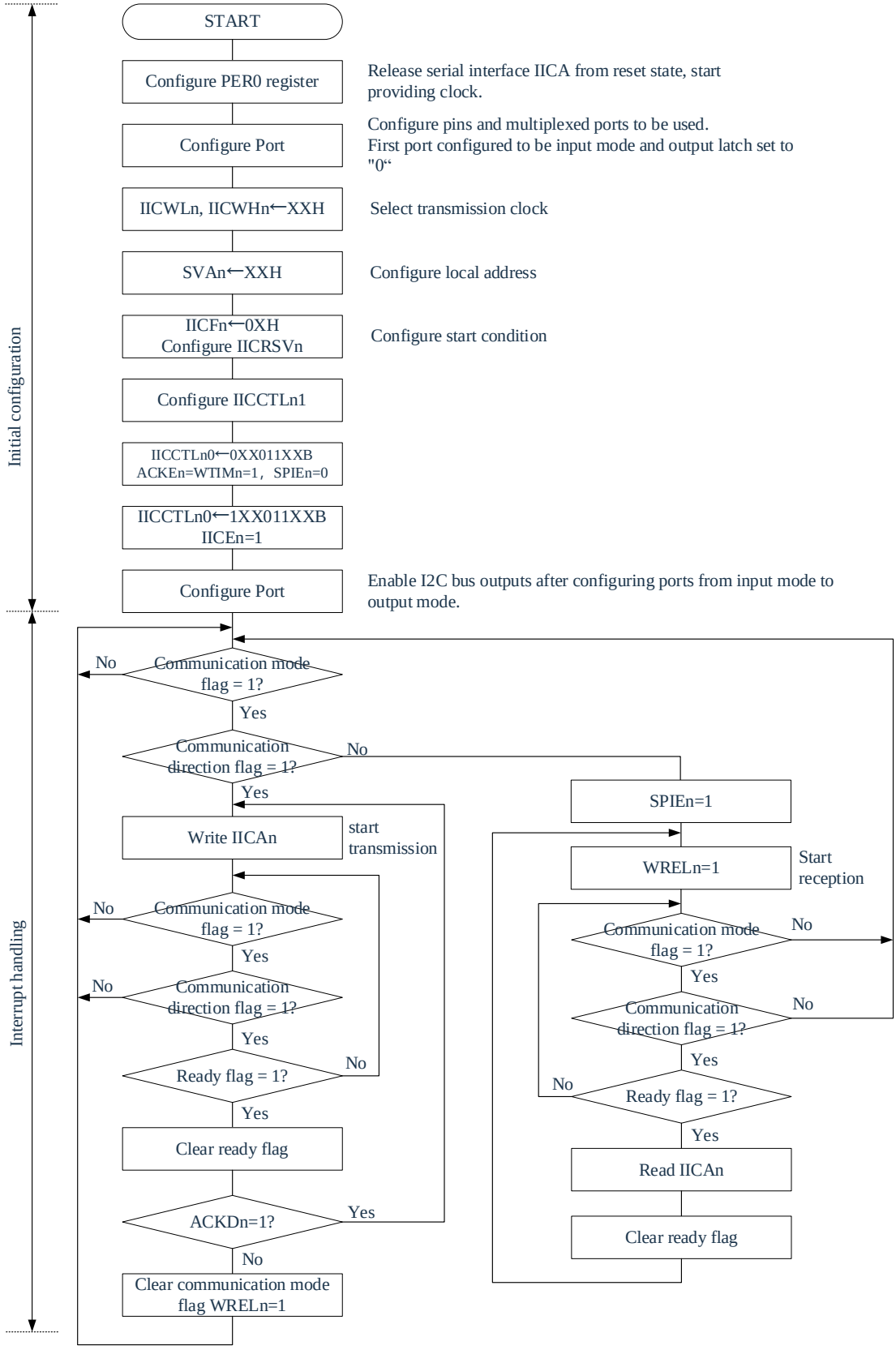
Note: n=0.

The main processing of the slave operation is explained next.

Start serial interface IICA and wait until communication is enabled. When communication is enabled, execute communication by using the communication mode flag and ready flag (processing of the stop condition and start condition is performed by an interrupt. Here, checks the status by using the flags).

The transmission operation is repeated until the master no longer returns ACK. If ACK is not returned from the master, communication is completed. For reception, the necessary amount of data is received. When communication is completed, ACK is not returned as the next data. After that, the master generates a stop condition or restart condition. Exit from the communication status occurs in this way.

Figure 16-21 Slave operation flowchart (1)




Note 1: The formats of transmission and reception must conform to the specifications of the product in communication.

Note 2: n=0.

An example of the steps for slave device processing via the INTIICAn interrupt is shown below (it is assumed that there is no processing with the extension code here). Confirm the status via the INTIICAn interrupt and perform the following processing.

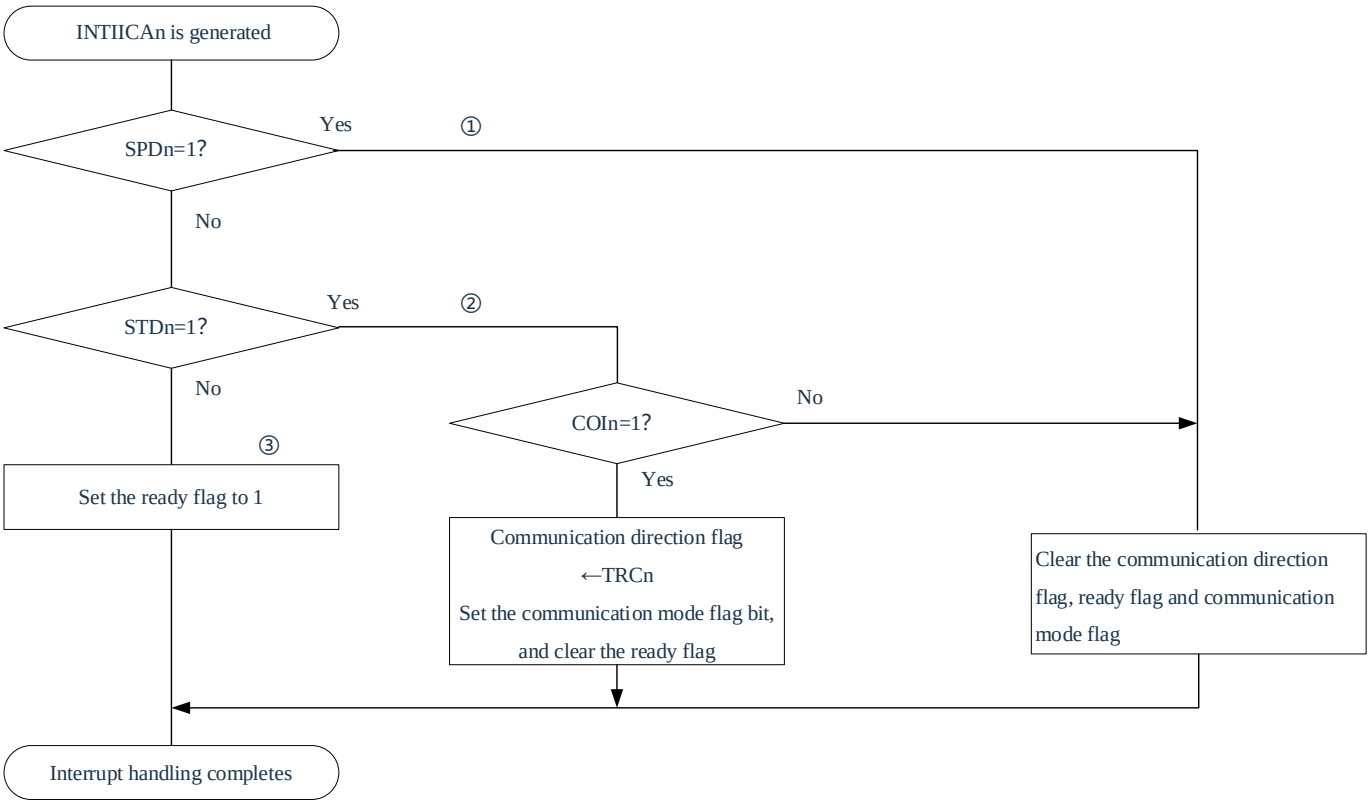
① Communication is stopped if the stop condition is issued.

② If the start condition is issued, the address is checked and communication is completed if the address does not match. If the address matches, the communication mode is set, wait is cancelled, and processing returns from the interrupt (the ready flag is cleared).

③ For data transmit/receive, only the ready flag is set. Processing returns from the interrupt with the $I^2C$ bus remaining in the wait state.

Note: ① to ③ above correspond to ① to ③ in Figure 16-21.

Figure 16-21 Slave operation flowchart (2)




Note: n=0.

## 16.7.17 Timing of I²C Interrupt Request (INTIICAn) Generation

The transmission and reception timing of data, the timing for the generation of the INTIICAn interrupt request signal, and the value of the IICA status register n (IICSn) when the INTIICAn signal is generated are as follows.

Note 1: ST: Start condition

Note 2: AD6~AD0: Address

Note 3: R/W: Transfer direction specification

Note 4: ACK: Acknowledge

Note 5: D7~D0: Data

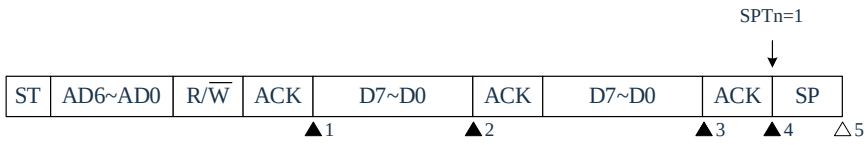
Note 6: SP: Stop condition

Note 7: n=0.

(1) Master operation

① Start~Address~Data~Data~Stop (transmit and receive)

a) When WTIMn=0




▲1:IICSn=1000X110B

▲2:IICSn=1000X000B

▲3:IICSn=1000X000B(set the WTIMn bit to 1) Note

▲4:IICSn=1000XX00B(set the SPTn bit to 1)

△5:IICSn=00000001B

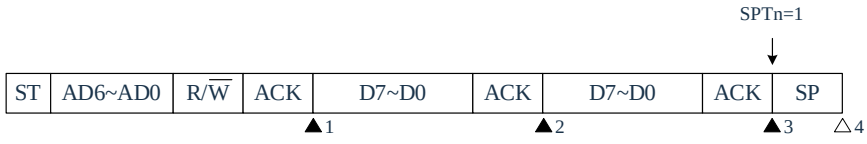
Note: To generate a stop condition, set the WTIMn bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

b) When WTIM0=1




▲1:IICSn=1000X110B

▲2:IICSn=1000X100B

▲3:IICSn=1000XX00B(set the SPTn bit to 1)

△4:IICSn=00000001B

Remark ▲ Always generated

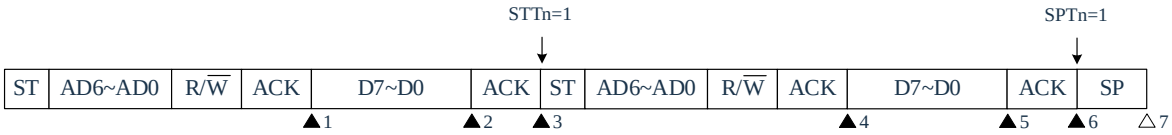
△ Generated only when SPIEn = 1

X Any

Note: n=0.

② Start~Address~Data~Start~Address~Data~Stop (restart)

a) When WTIMn=0

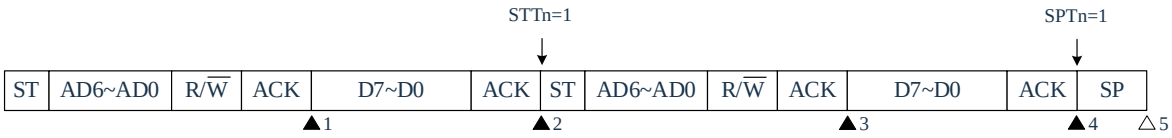



▲1:IICSn=1000X110B
▲2:IICSn=1000X000B(set the WTIMn bit to 1) Note1
▲3:IICSn=1000XX00B(set the WTIMn bit to 0 Note 2 and set the STTn bit to 1)
▲4:IICSn=1000X110B
▲5:IICSn=1000X000B(set the WTIMn bit to 1) Note3
▲6:IICSn=1000XX00B(set the SPTn bit to 1)
△7:IICSn=00000001B

Note1. To generate a start condition, set the WTIMn bit to 1 and change the timing for generating theINTIICAn interrupt request signal.
2. Clear the WTIMn bit to 0 to restore the original setting.
3. To generate a stop condition, set the WTIMn bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

Remark ▲ Always generated
△ Generated when SPIE =1
X Any

b) When WTIMn=1



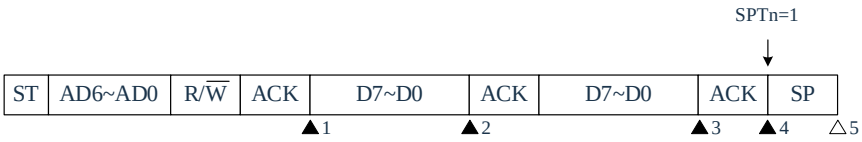

▲1:IICSn=1000X110B
▲2:IICSn=1000XX00B(set STTn bit to "1")
▲3:IICSn=1000X110B
▲4:IICSn=1000XX00B(set SPTn bit to "1")
△5:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

Note: n=0.

③ Start~Code~Data~Data~Stop (extension code transmission)

a) When WTIMn=0

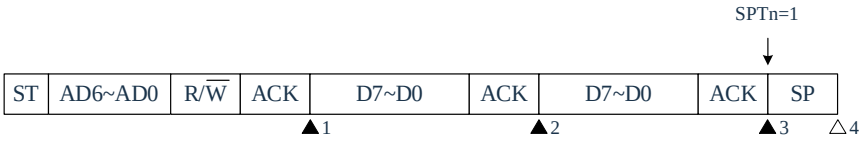



▲1:IICSn=1010X110B
▲2:IICSn=1010X000B
▲3:IICSn=1010X000B(Set the WTIMn bit to 1)Note
▲4:IICSn=1010XX00B(Set the SPTn bit to 1)
△5:IICSn=00000001B

Note To generate a stop condition, set the WTIMn bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

b) When WTIMn=1




▲1:IICSn=1010X110B
▲2:IICSn=1010X100B
▲3:IICSn=1010XX00B(Set SPTn bit to 1)
△4:IICSn=00000001B

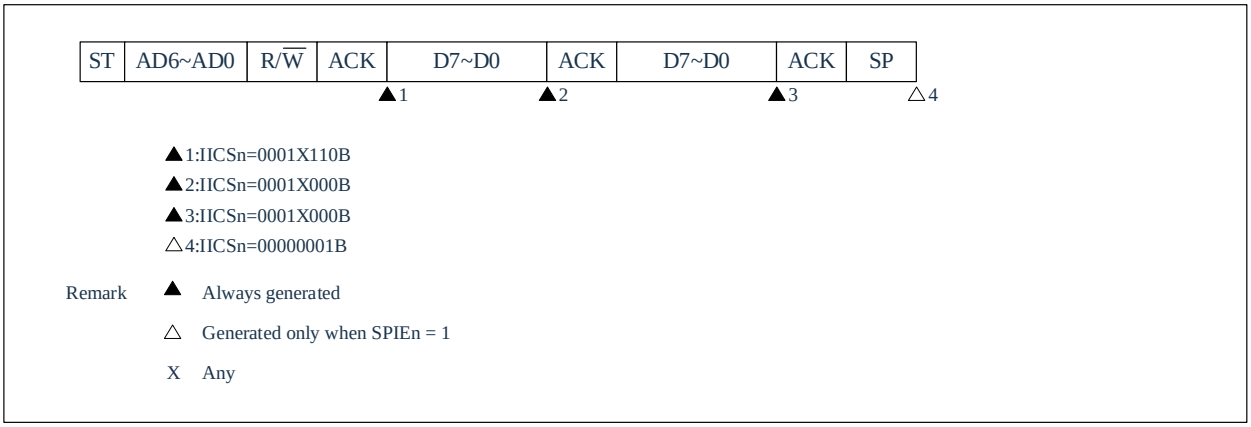
Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

Note: n=0.

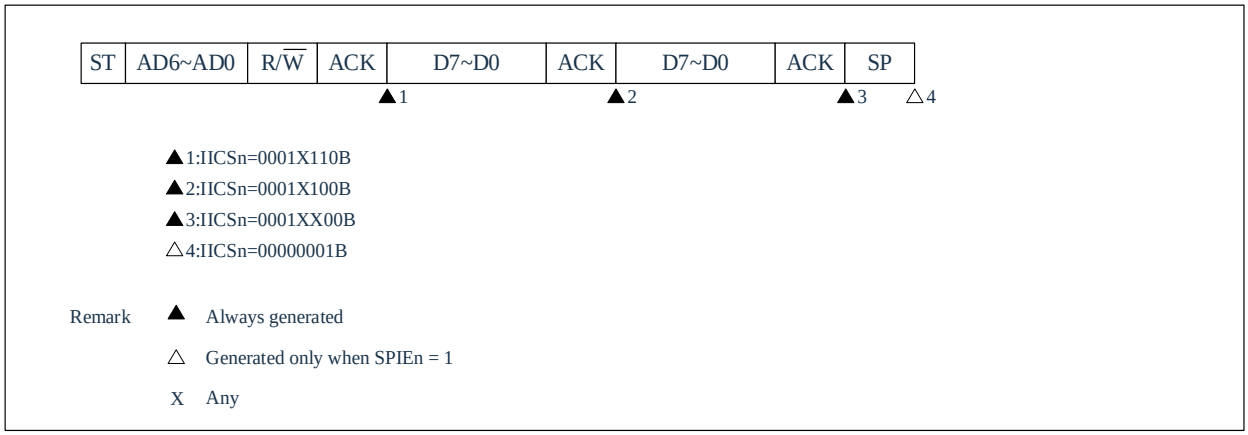
(2) Slave operation (when receiving a slave address)

① Start~Address~Data~Data~Stop

a) When WTIMn=0




b) When WTIMn=1




Note: n=0.

② Start~Address~Data~Start~Address~Data~Stop

a) When WTIMn=0 (after restart, matches with SVAn)


▲1:IICSn=0001X110B
▲2:IICSn=0001X000B
▲3:IICSn=0001X110B
▲4:IICSn=0001X000B
△5:IICSn=00000001B

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

b) When WTIMn=1 (after restart, matches with SVAn)


▲1:IICSn=0001X110B
▲2:IICSn=0001XX00B
▲3:IICSn=0001X110B
▲4:IICSn=0001XX00B
△5:IICSn=00000001B

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

Note: n=0.

③ Start~Address~Data~Start~Code~Data~Stop

a) When WTIMn=0 (after restart, does not match address (= extension code))

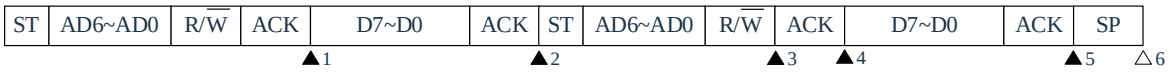

▲1:IICSn=0001X110B
▲2:IICSn=0001X000B
▲3:IICSn=0010X010B
▲4:IICSn=0010X000B
△5:IICSn=00000001B

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

b) When WTIMn=1 (after restart, does not match address (= extension code))




▲1:IICSn=0001X110B
▲2:IICSn=0001XX00B
▲3:IICSn=0010X010B
▲4:IICSn=0010X110B
▲5:IICSn=0010XX00B
△6:IICSn=00000001B

Remark ▲ Always generated

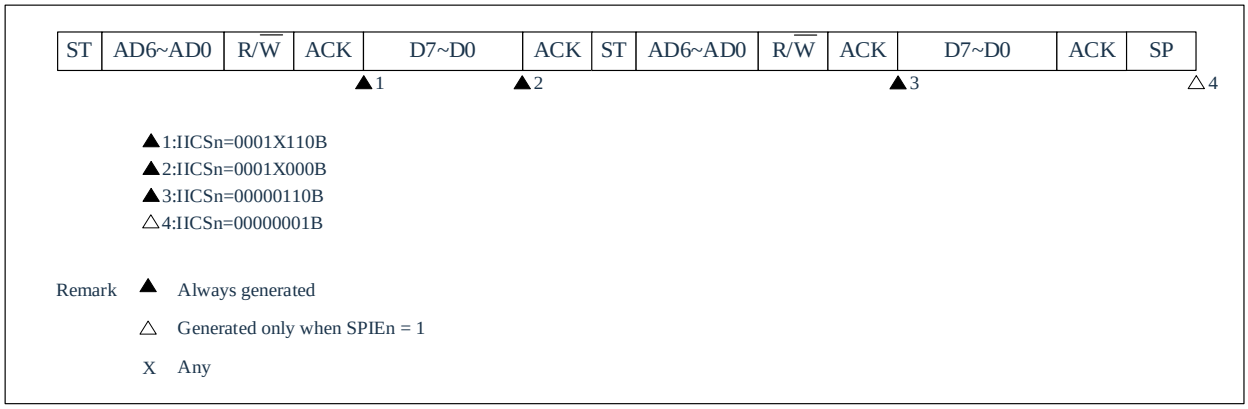
△ Generated only when SPIEn = 1

X Any

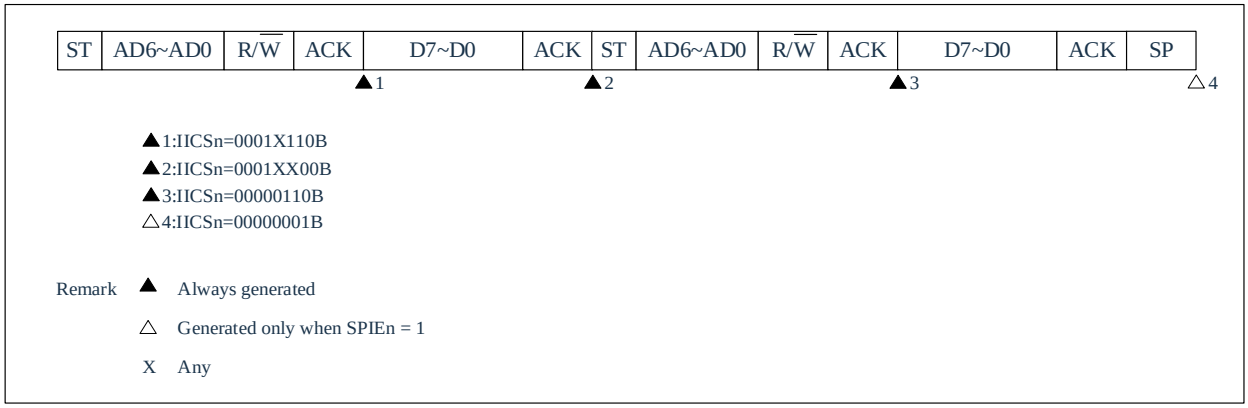
Note: n=0.

④ Start~Address~Data~Start~Address~Data~Stop

a) When WTIMn=0 (after restart, does not match address (= not extension code))




b) When WTIMn=1 (after restart, does not match address (= not extension code))




Note: n=0.

(3) Slave operation (when receiving extension code)

The device is always participating in communication when it receives an extension code.

① Start~Code~Data~Data~Stop

a) When WTIMn=250


▲1:IICSn=0010X010B
▲2:IICSn=0010X000B
▲3:IICSn=0010X000B
△4:IICSn=00000001B

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

b) When WTIMn=1


▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010X100B
▲4:IICSn=0010XX00B
△5:IICSn=00000001B

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

Note: n=0.

② Start~Code~Data~Start~Address~Data~Stop

a) When WTIMn=0 (after restart, matches SVAn)


▲1:IICSn=0010X010B
▲2:IICSn=0010X000B
▲3:IICSn=0001X110B
▲4:IICSn=0001X000B
△5:IICSn=00000001B

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

b) When WTIMn=1 (after restart, matches SVAn)


▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010XX00B
▲4:IICSn=0001X110B
▲5:IICSn=0001XX00B
△6:IICSn=00000001B

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

Note: n=0.

③ Start~Code~Data~Start~Code~Data~Stop

a) When WTIMn=0 (after restart, extension code reception)


▲1:IICSn=0010X010B
▲2:IICSn=0010X000B
▲3:IICSn=0010X010B
▲4:IICSn=0010X000B
△5:IICSn=00000001B

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

b) When WTIMn=1 (after restart, extension code reception)


▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010XX00B
▲4:IICSn=0010X010B
▲5:IICSn=0010X110B
▲6:IICSn=0010XX00B
△7:IICSn=00000001B

Remark ▲ Always generated

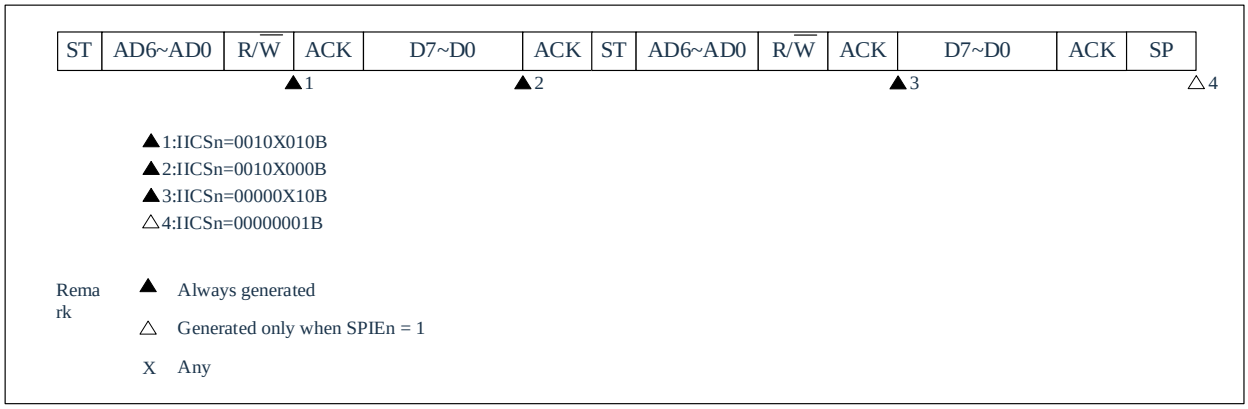
△ Generated only when SPIEn = 1

X Any

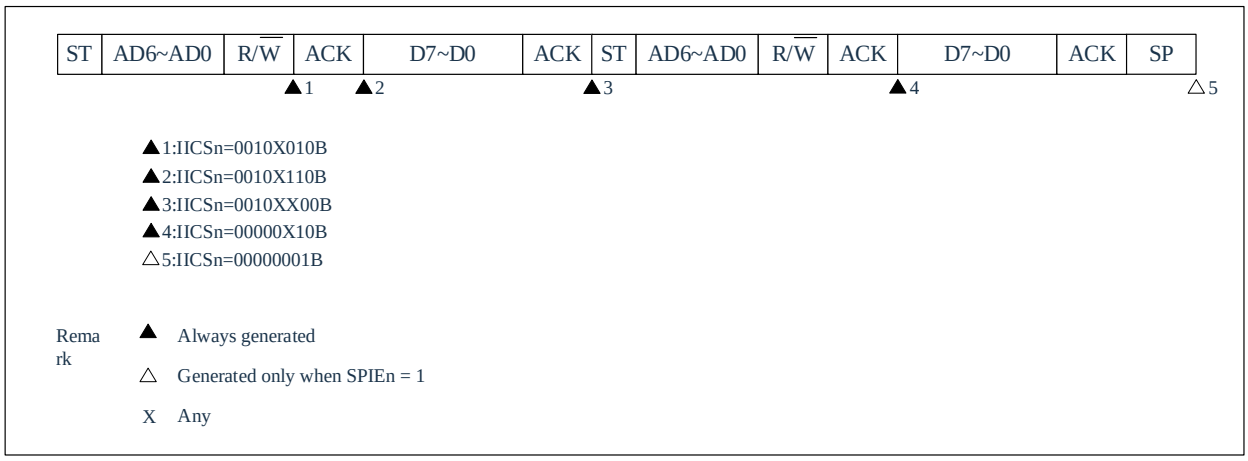
Note: n=0.

④ Start~Code~Data~Start~Address~Data~Stop

a) When WTIMn=0 (after restart, does not match address (= not extension code))



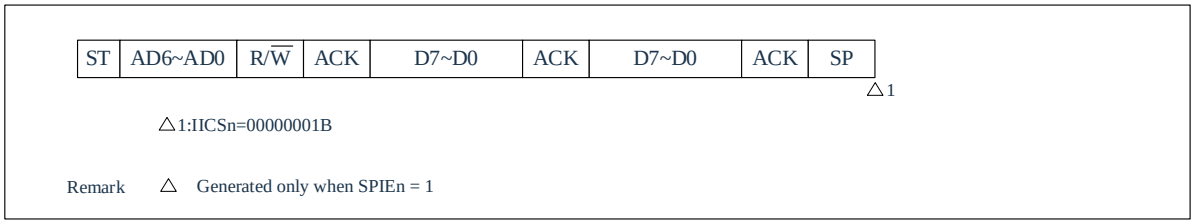

b) When WTIMn=1 (after restart, does not match address (= not extension code))




Note: n=0.

(4) Operation without communication

Start~Code~Data~Data~Stop

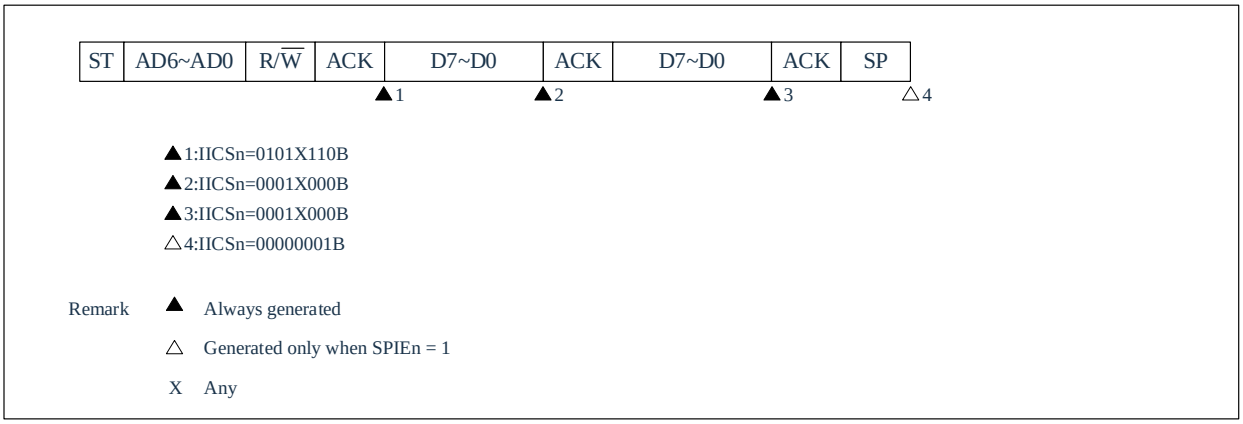



(5) Arbitration loss operation (operation as slave after arbitration loss)

When the device is used as a master in a multi-master system, read the MSTSn bit each time interrupt request signal INTIICAn has occurred to check the arbitration result.

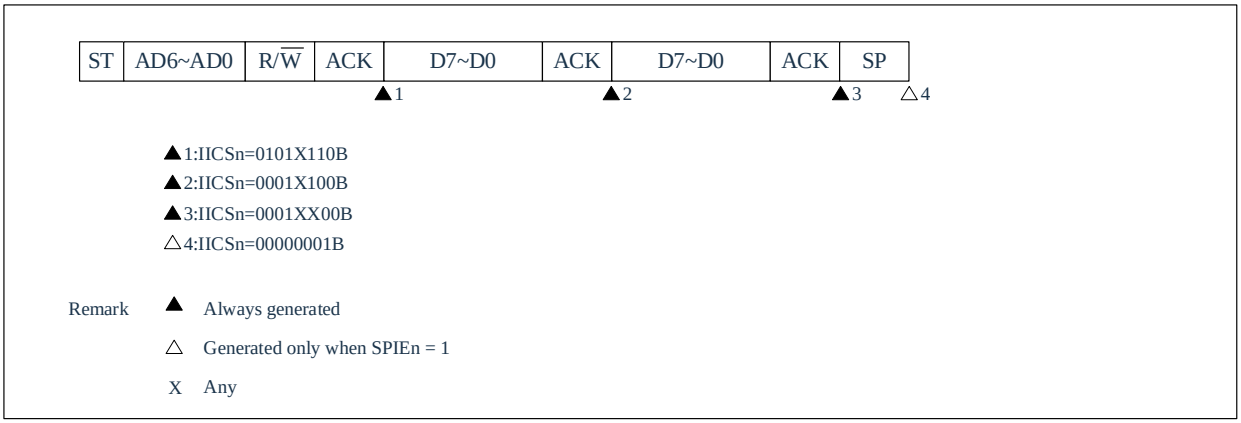
① When arbitration loss occurs during transmission of slave address data

a) When WTIMn=0



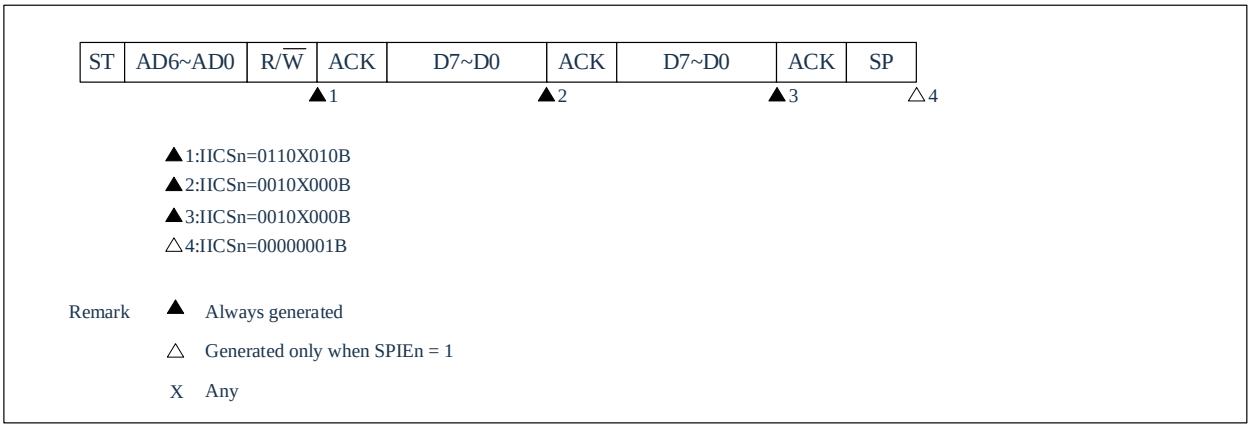

Note: n=0.

b) When WTIMn=1



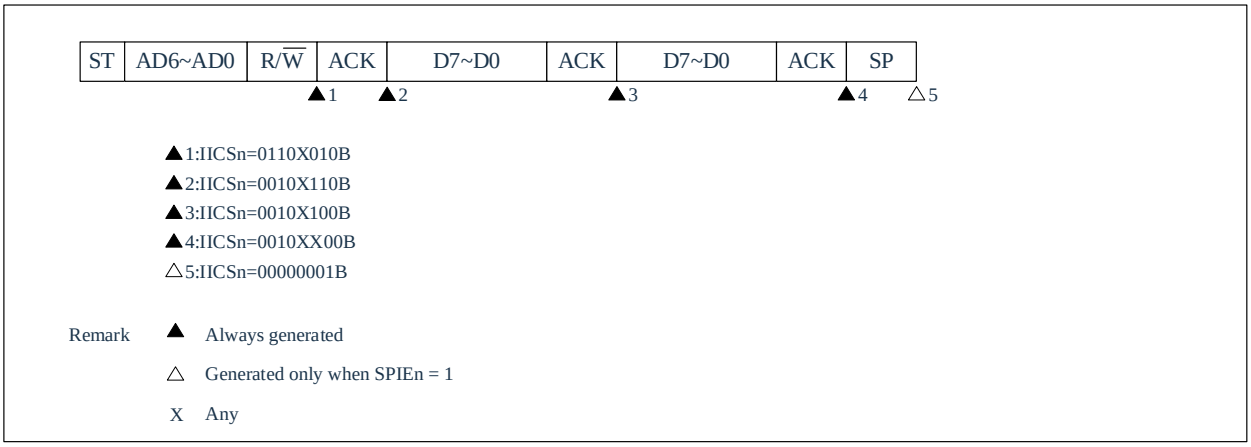
② When arbitration loss occurs during transmission of extension code

a) When WTIMn=0




Note: n=0.

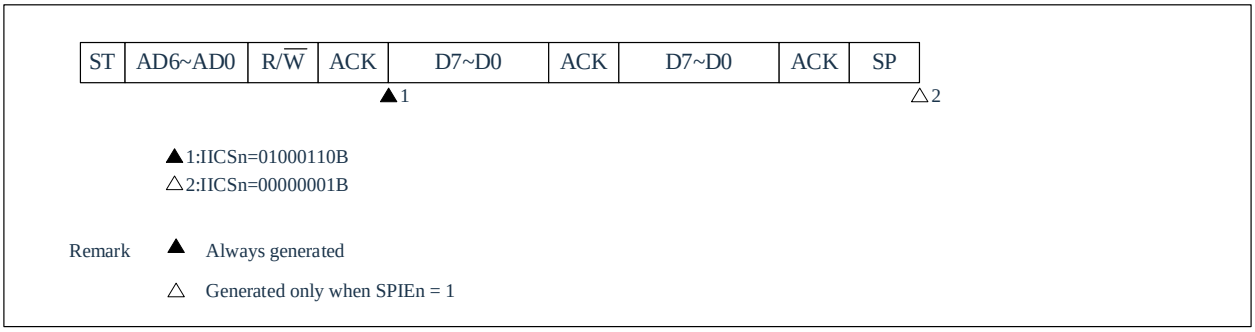
b) When WTIMn=1



(6) Operation when arbitration loss occurs (no communication after arbitration loss)

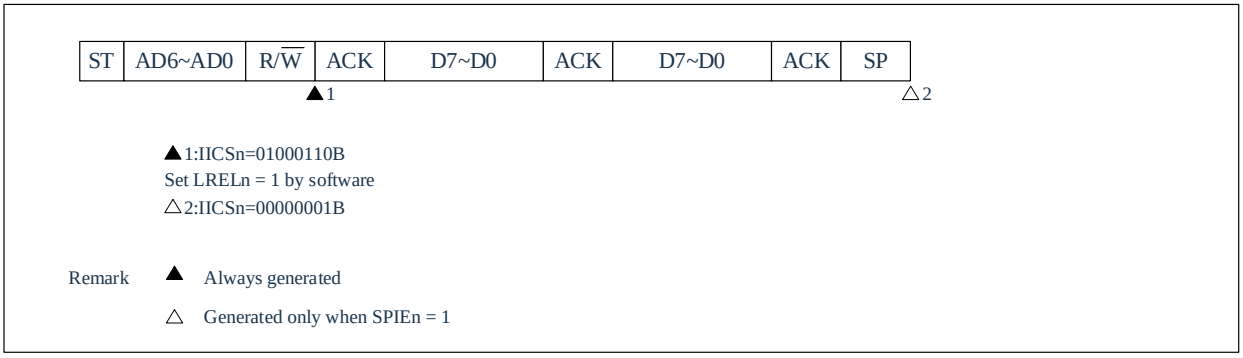
When the device is used as a master in a multi-master system, read the MSTSn bit each time interrupt request signal INTIICAn has occurred to check the arbitration result.

① When arbitration loss occurs during transmission of slave address data (when WTIMn = 1)



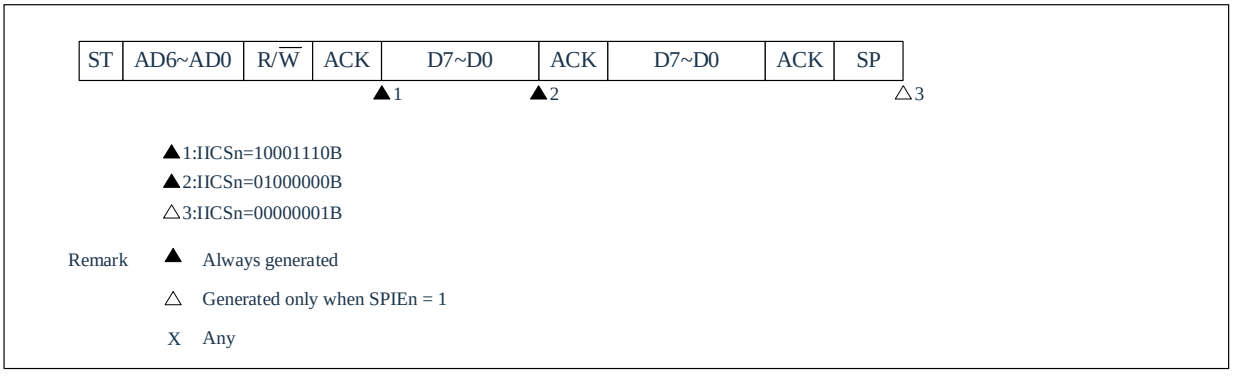

Note: n=0.

② When arbitration loss occurs during transmission of extension code



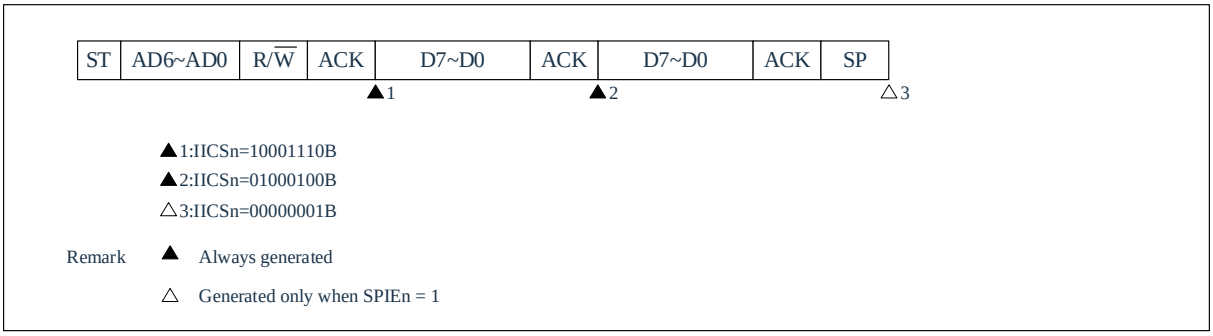

③ When arbitration loss occurs during transmission of data

a) When WTIMn=0




Note: n=0.

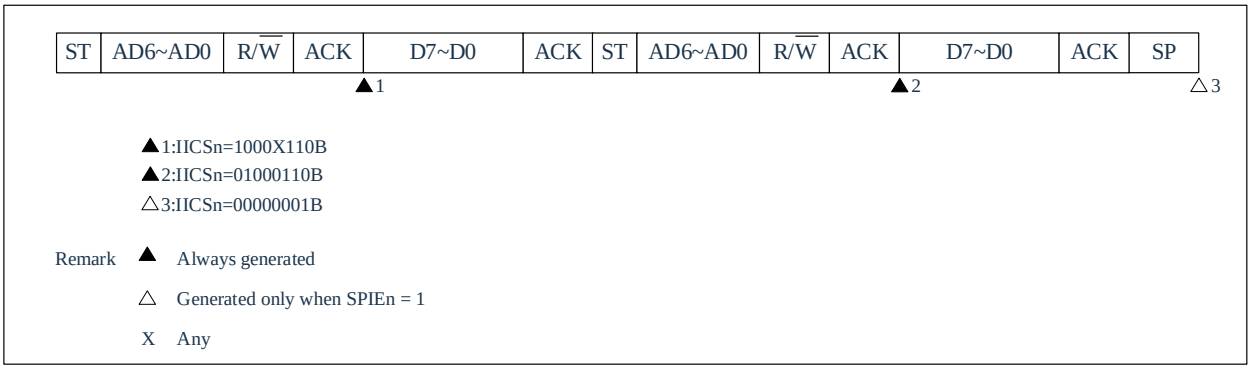
b) When WTIMn=1




Note: n=0.

④ When loss occurs due to restart condition during data transfer

a) Not extension code (Example: unmatches with SVAn)




▲1:IICSn=1000X110B
▲2:IICSn=01000110B
△3:IICSn=00000001B

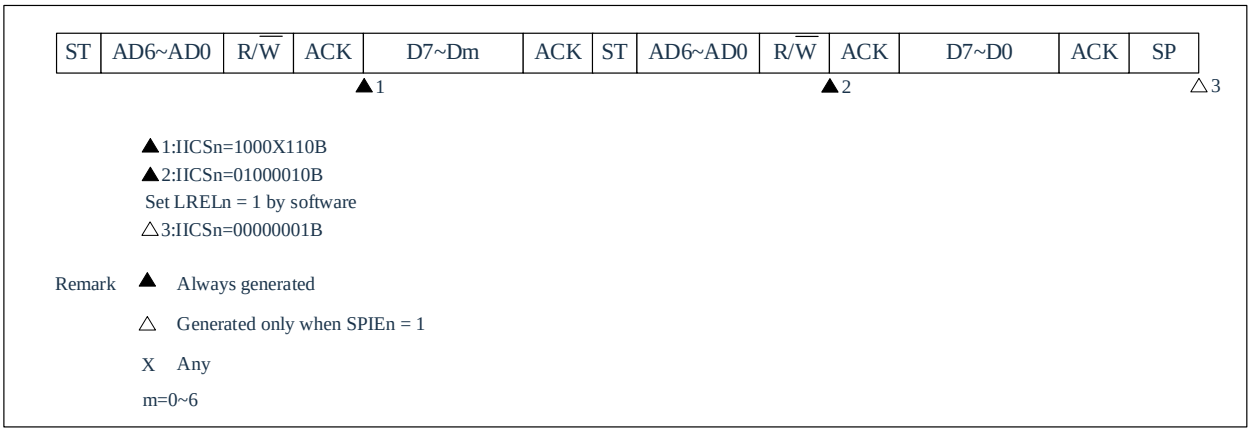
Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

Note: n=0.

b) Extension code




▲1:IICSn=1000X110B
▲2:IICSn=01000010B
Set LRELn = 1 by software
△3:IICSn=00000001B

Remark ▲ Always generated

△ Generated only when SPIEn = 1

X Any

m=0~6

⑤ When loss occurs due to stop condition during data transfer


▲1:IICSn=10000110B
△2:IICSn=01000001B

Remark ▲ Always generated

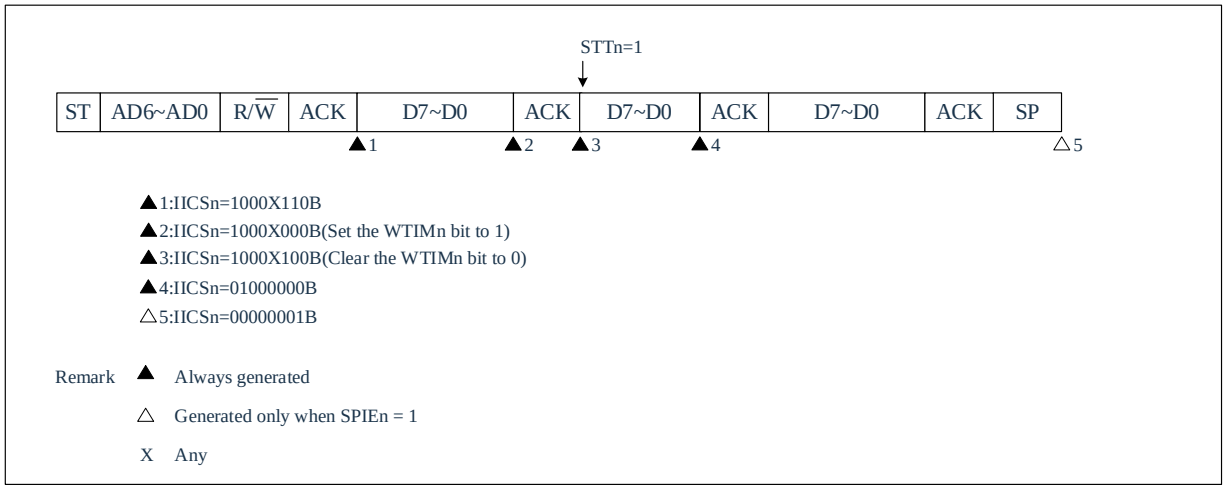
△ Generated only when SPIEn = 1

m=0~6

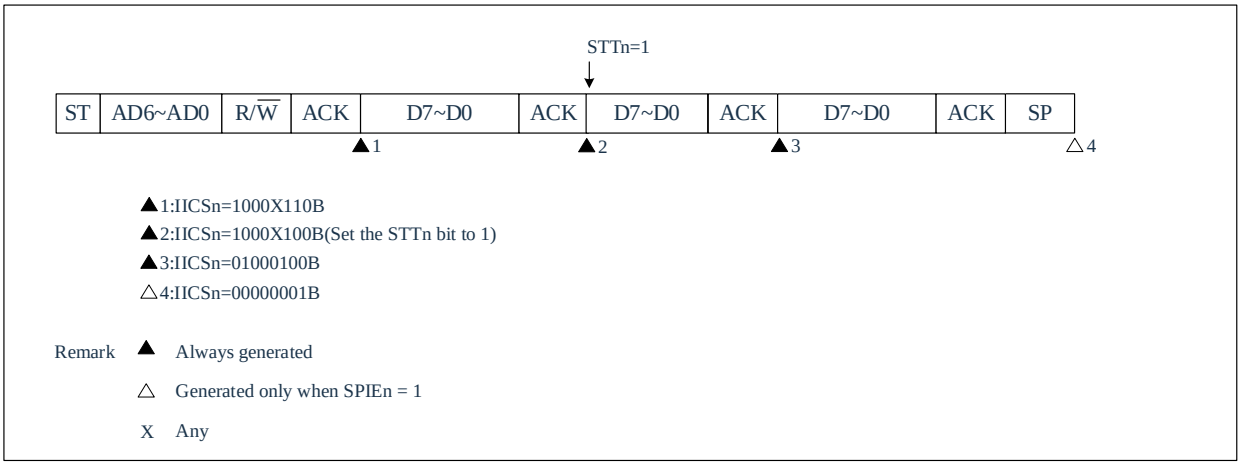
Note: n=0.

⑥ When arbitration loss occurs due to low-level data when attempting to generate a restart condition

a) When WTIMn=0



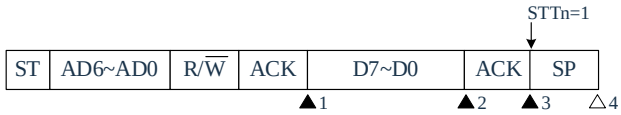

b) When WTIMn=1




Note: n=0.

⑦ When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

a) When WTIMn=0




▲1:IICSn=1000X110B
▲2:IICSn=1000X000B(Set the WTIMn bit to 1)
▲3:IICSn=1000XX00B(Set the STTn bit to 1)
△4:IICSn=01000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

b) When WTIMn=1

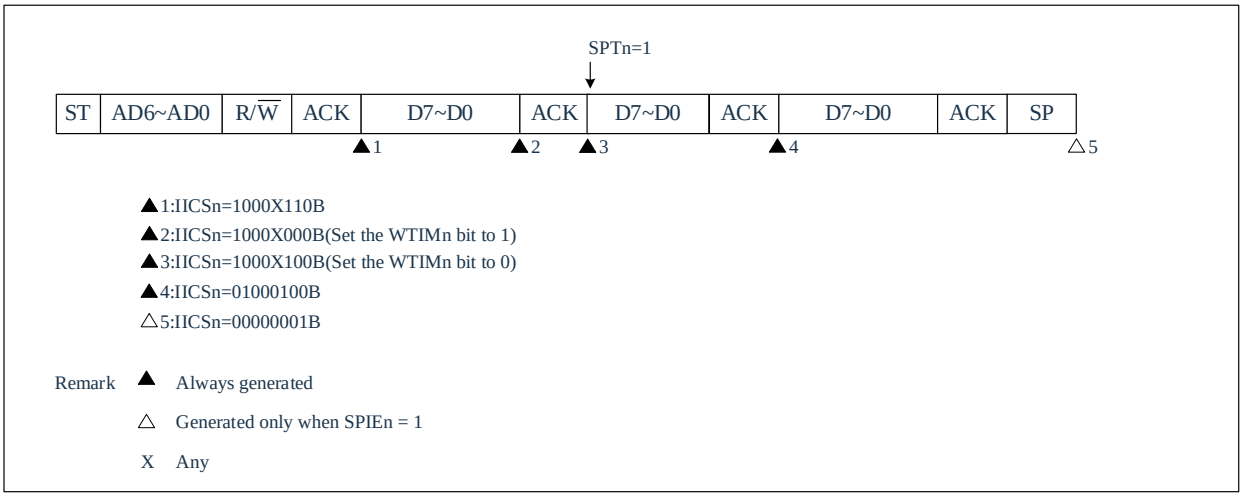

▲1:IICSn=1000X110B
▲2:IICSn=1000XX00B(Set the STTn bit to 1)
△3:IICSn=01000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

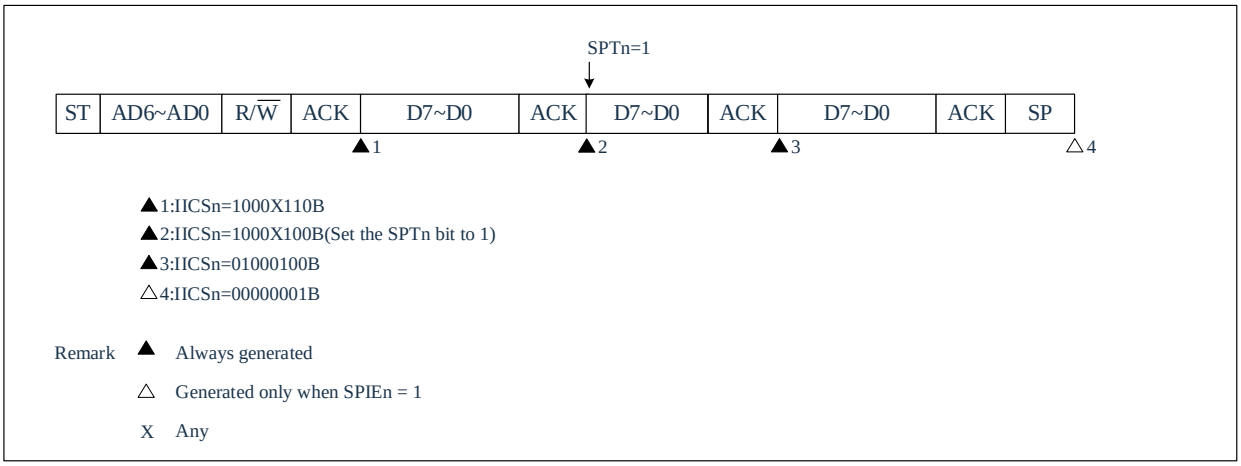
Note: n=0.

⑧ When arbitration loss occurs due to low-level data when attempting to generate a stop condition

a) When WTIMn=0




b) When WTIMn=1




Note: n=0.

## 16.8Timing Diagrams

When using the $I^2C$ bus mode, the master device outputs an address via the serial bus to select one of several slave devices as its communication partner. After outputting the slave address, the master device transmits the TRCn bit (the bit 3 of the IICA status register n (IICSn)), which specifies the data transfer direction, and then starts serial communication with the slave device. Timing diagrams of the data communication are shown in Figure 16-22 and Figure 16-23.

The IICA shift register n (IICAn)'s shift operation is synchronized with the falling edge of the serial clock (SCLAn). The transmit data is transferred to the SO latch and is output (MSB first) via the SDAAn pin.

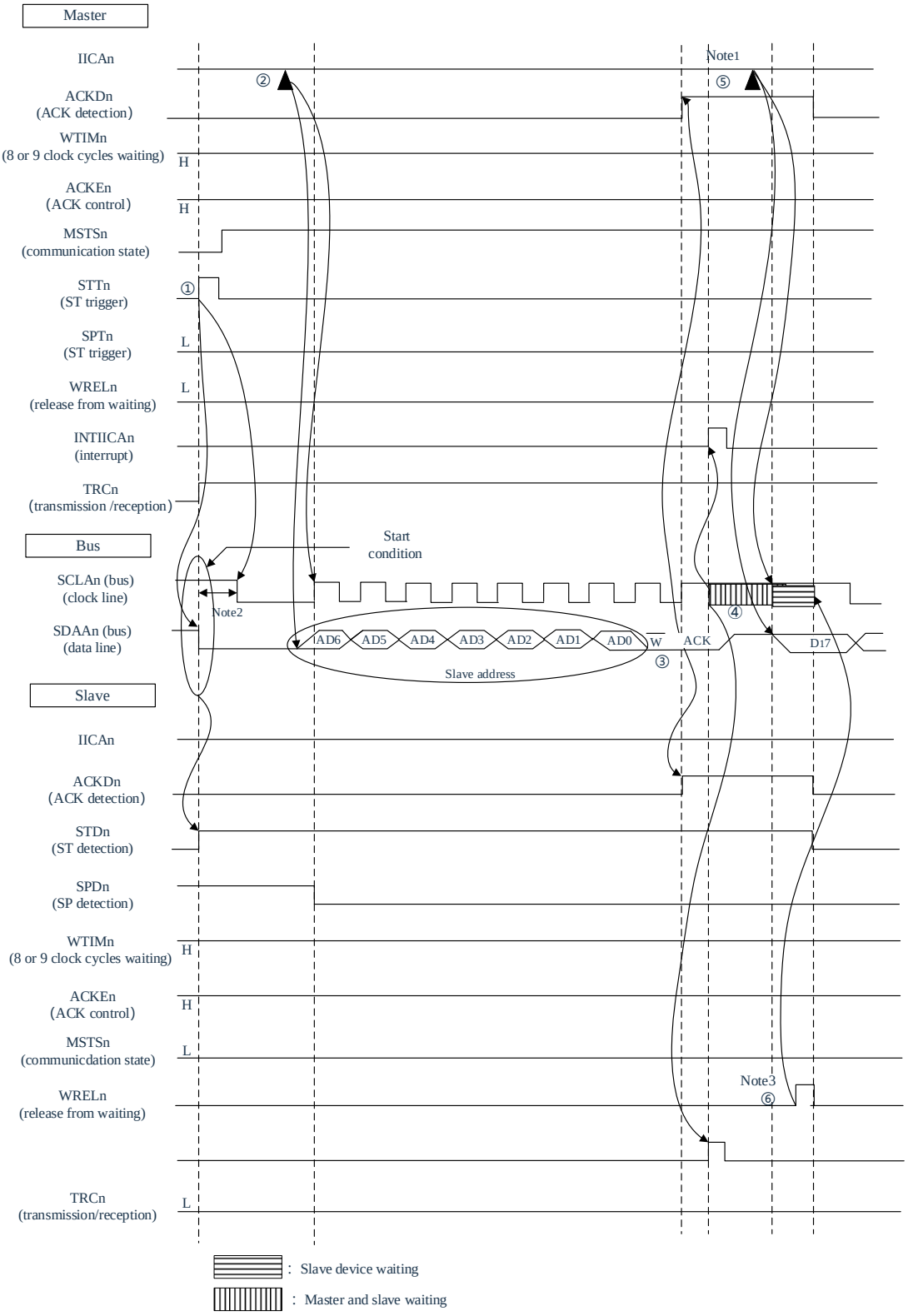
Data input via the SDAAn pin is captured into IICAn at the rising edge of SCLAn.

Note: n=0.

Figure 16-22 Example of master to slave communication

(Master: selects 9 clocks to wait, slave: selects 9 clocks to wait) (1/4)

(1) Start Condition ~ Address ~ Data




Note 1: Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.

Note 2: Make sure that the time between the fall of the SDAAn pin signal and the fall of the SCLAn pin signal is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 3: For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

Figure 16-22 shows the descriptions of ① to ⑥ of "(1) Start condition ~ Address ~ Data"

① If the master sets the start condition trigger set (STTn=1), the bus data line (SDAAn) drops and the start condition is generated (SDAAn is changed from "1" to "0" by SCLAn=1). Thereafter, if a start condition is detected, the master enters the master communication state (MSTSn=1) and after the hold time elapses the bus clock line drops (SCLAn=0), ending the communication preparation.

② If the master writes address +W (transmit) to IICA shift register n (IICAn), the slave address is sent.

③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same [Note], an ACK is sent to the master through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) [Note].

⑤ The master writes and transmits data to the IICAn registers, releasing the master of waiting.

⑥ If the slave releases the wait (WRELn=1), the master begins to transmit data to the slave.

Note 1: If the transmitted address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master, and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address transmit end interrupts) for both ACK and NAK.

Note 2: ①~⑮ of Figure 16-22 shows a series of operational steps for data communication via the $I^2C$ bus.

Note 3: "(1) Address~Data~Data" of Figure 16-22 illustrates steps ①~⑥.

Note 4: "(2) Address~Data~Data" of Figure 16-22 illustrates steps ③~⑩.

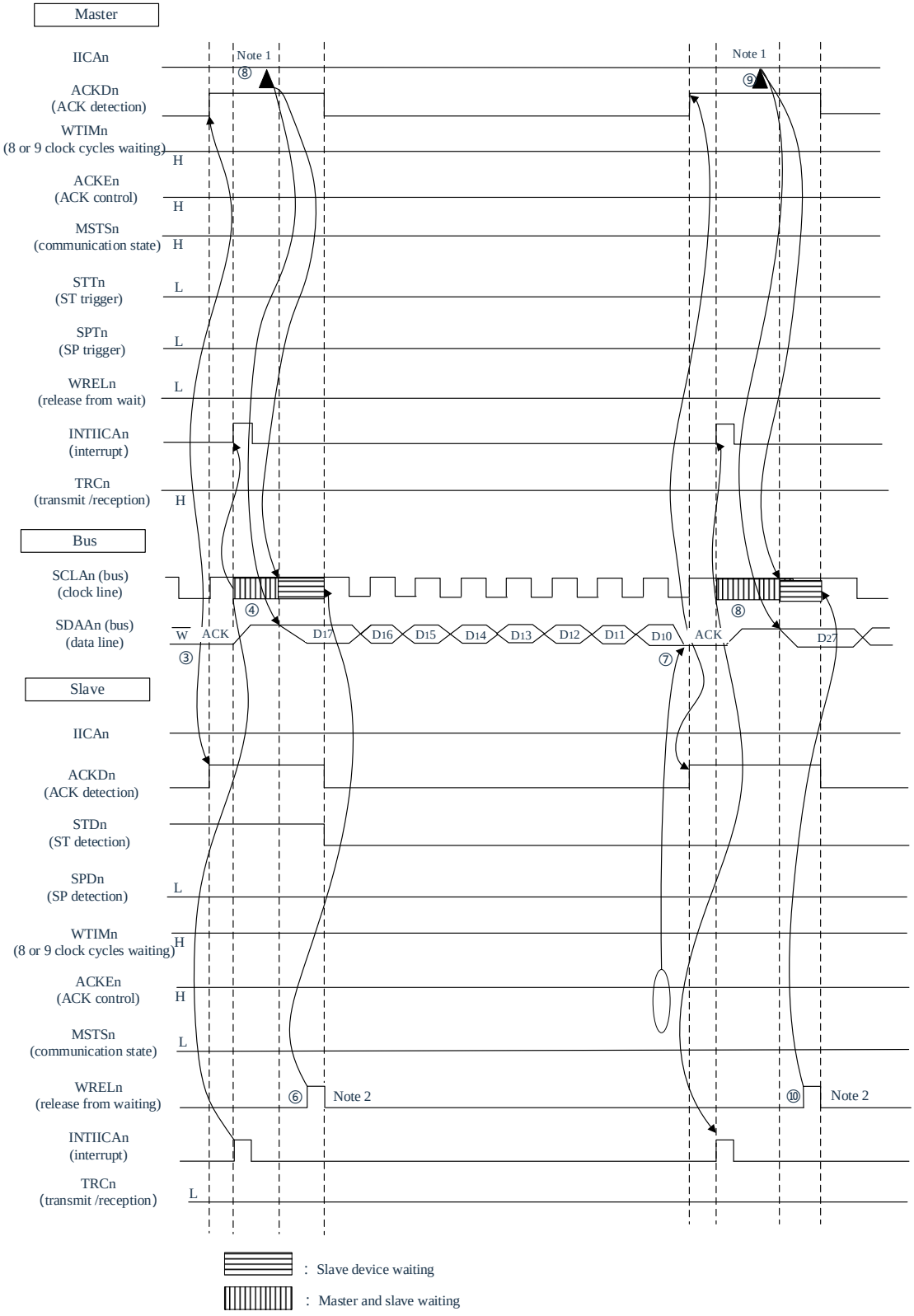
Note 5: "(3) Data~Data~Stop Condition" of Figure 16-22 illustrates steps ⑦~⑮.

Note 6: n=0.

Figure 16-22 Example of master to slave communication

(Master: selects 9 clocks to wait, slave: selects 9 clocks to wait) (2/4)

(2) Address ~ Data ~ Data




Note 1: Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.

Note 2: For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

Note 3: n=0

Figure 16-22 shows the descriptions of ③ to ⑩ of "(2) Address~Data~Data":

③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same [Note], the ACK is sent to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) [Note].

⑤ The master writes and sends data to the IICA shift register n (IICAn) to remove the wait of the main controller.

⑥ If the slave releases the wait (WRELn=1), the master party begins to transmit data to the slave.

⑦ After the data transfer is completed, because the ACKEn bit of the slave is 1, the ACK is sent to the master control through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

⑧ Both the master and the slave enter a waiting state (SCLAn= 0) on the falling edge of the 9th clock, and both produce interrupts (INTIICAn: Transmit End Interrupt).

⑨ The master controller writes and sends data to the IICAn register to remove the waiting of the main controller.

⑩ If the slave reads and receives the data and cancels the wait (WRELn=1), the master party begins to transmit data to the slave.

Note 1: If the transmitted address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

Note 2: ①~⑮ of Figure 16-22 shows a series of operational steps for data communication via the $I^2C$ bus.

Note 3: "(1) Address~Data~Data" of Figure 16-22 illustrates steps ①~⑥.

Note 4: "(2) Address~Data~Data" of Figure 16-22 illustrates steps③~⑩.

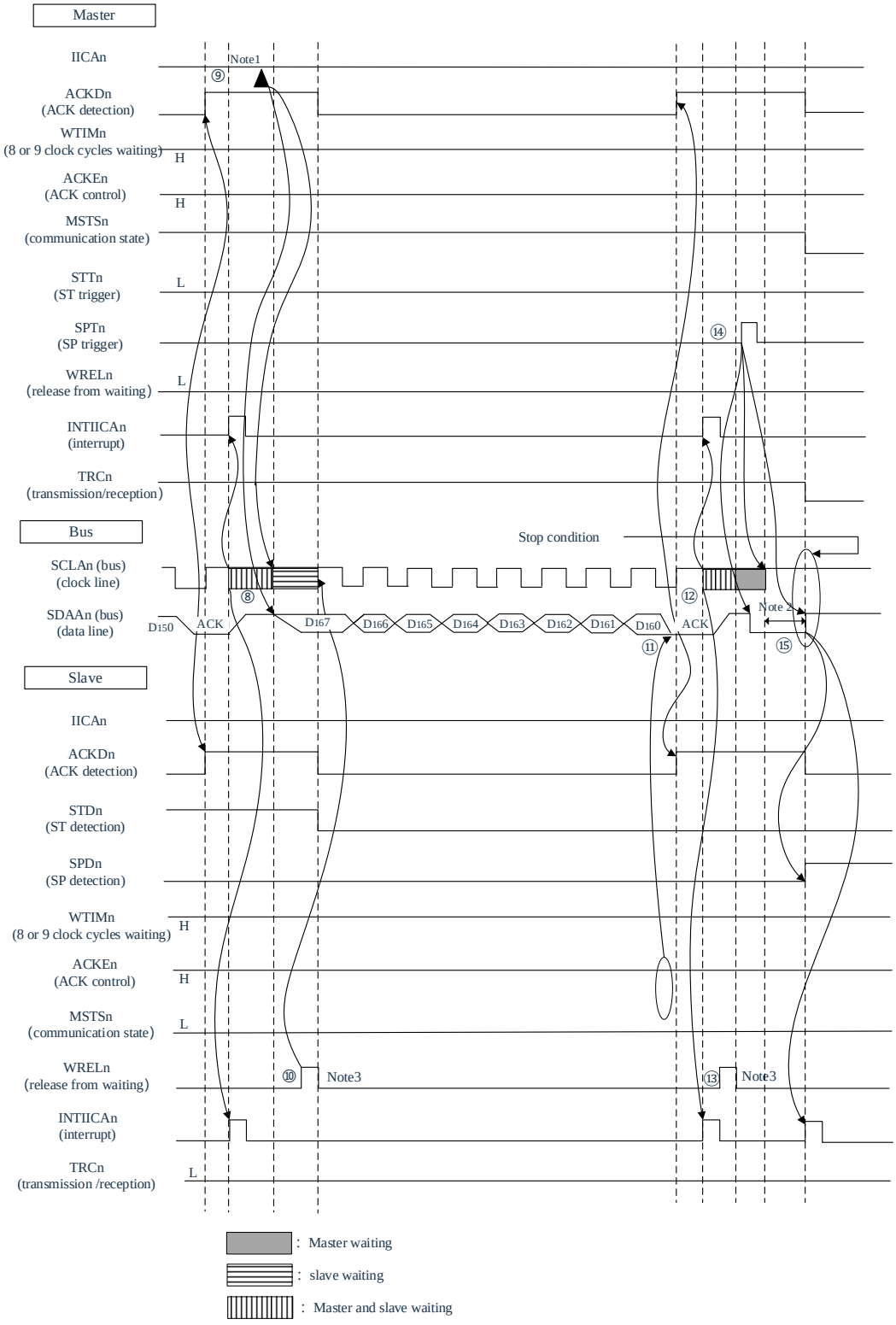
Note 5: "(3) Data~Data~Stop Condition" of Figure 16-22 illustrates steps ⑦~⑮.

Note 6: n=0.

Figure 16-22 Example of master to slave communication

(Master: selects 9 clocks to wait, slave: selects 9 clocks to wait) (3/4)

(3) Data~Data~Stop Condition




Note 1: Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.

Note 2: After the stop condition is issued, the time from the SCLAn pin signal to generate the stop condition is at least 4.0μs when set to standard mode and at least 0.6μs when set to fast mode.

Note 3: For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

Figure 16-22 shows the descriptions of ⑦~⑮ of "(3) Data ~ Data ~ Stop condition":

⑦ At the end of the data transfer, because the ACKEn bit of the slave is 1, the ACK is sent to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1)

⑧ Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock, and both produce an interrupt (INTIICAn: end-of-transmit interrupt).

⑨ The master writes and transmits data to the IICA shift register n (IICAn), relieving the master of waiting.

⑩ If the slave reads the received data and dismisses the wait (WRELn=1), the master starts transmitting data to the slave

⑪ At the end of the data transfer, the slave (ACKEn=1) sends an ACK to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

⑫ Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock, and both produce an interrupt (INTIICAn: end-of-transmit interrupt).

⑬ The slave reads the received data and dismisses the wait (WRELn=1).

⑭ If the master sets the stop condition trigger set (SPTn=1), the bus data line (SDAAn=0) is cleared and the bus clock line is set (SCLAn=1), and the bus data line is set after the preparation time for the stop condition is passed (SDAAn=1), Generate a stop condition (SDAAn from "0" to "1" by SCLAn=1).

⑮ If a stop condition is generated, the slave detects the stop condition and generates an interrupt (INTIICAn: Stop condition interrupt).

Note 1:①~⑮ of Figure 16-22 shows a series of operational steps for data communication via the $I^2C$ bus.

Note 2: "(1) Address~Data~Data" of Figure 16-22 illustrates steps ①~⑥.

Note 3: "(2) Address~Data~Data" of Figure 16-22 illustrates steps③~⑩.

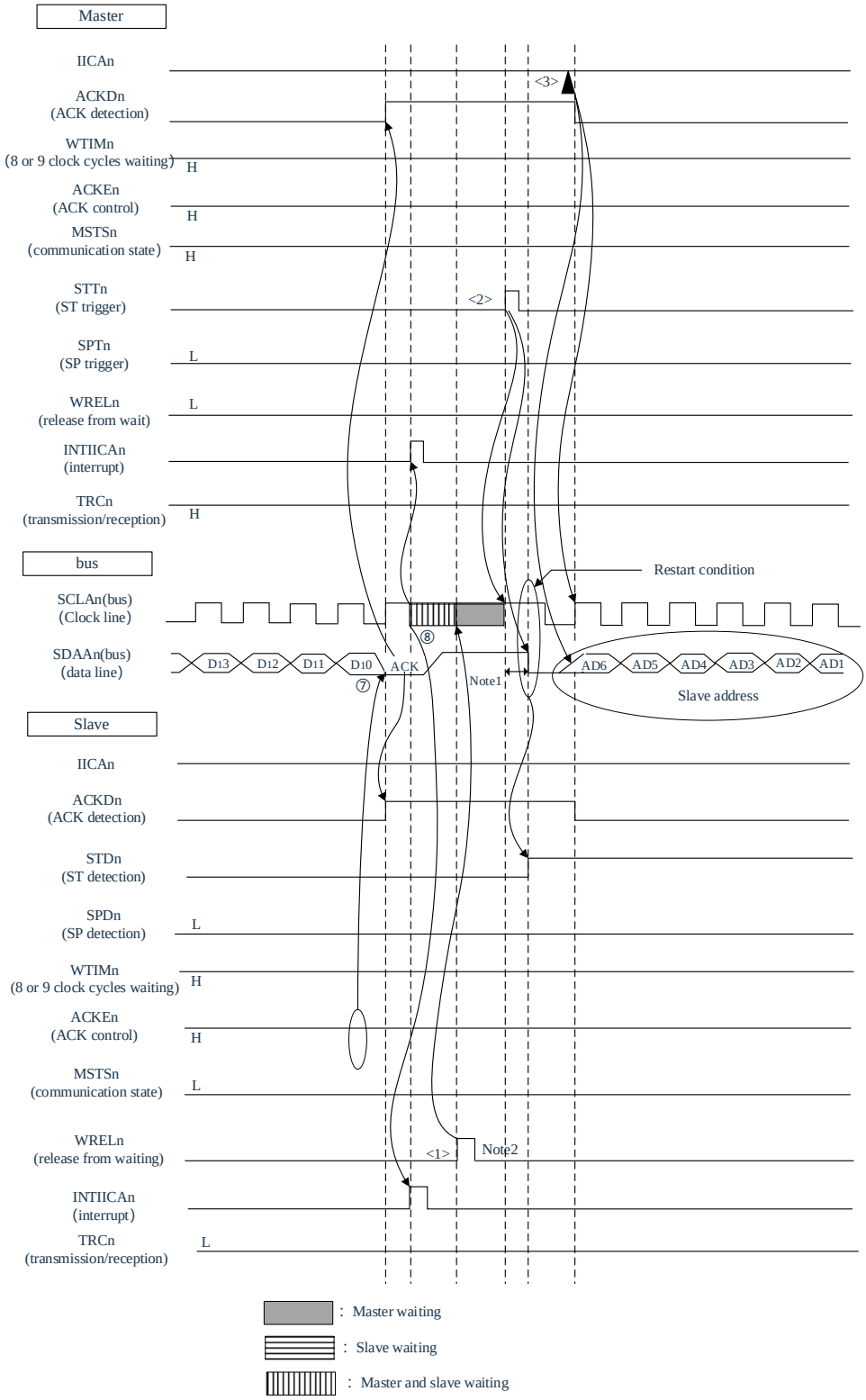
Note 4: "(3) Data~Data~Stop Condition" of Figure 16-22 illustrates steps ⑦~⑮.

Note 5: n=0.

Figure 16-22 Example of master to slave communication

(Master: selects 9 clocks to wait, slave: selects 9 clocks to wait) (4/4)

(4) Data ~ Restart Condition ~ Address




Note 1: Make sure that the time between the rise of the SCLAn pin signal and the generation of the start condition after a restart condition has been issued is at least 4.7 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 2: For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

Note 3: n=0

Figure 16-22 shows the operation of "(4) Data ~ Restart Condition ~ Address" as follows. After executing steps ⑦ and ⑧, execute <1> to <3> to return to the data sending step of step ③.

⑦ After the data transfer is completed, because the ACKEn bit of the slave is 1, the ACK is sent to the master control through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

⑧ Both the master and the slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock, and both produce interrupts (INTIICAn: Transmit End Interrupt).

<1> The slave reads and receives the data, and the wait is released (WRELn=1).

<2> The start condition trigger is set again by the master device (STTn = 1) and a start condition (i.e. SCLAn =1 changes SDAAn from 1 to 0) is generated once the bus clock line goes high (SCLAn = 1) and the bus data line goes low (SDAAn=0) after the restart condition setup time has elapsed. When the start condition is subsequently detected, the master device is ready to communicate once the bus clock line goes low (SCLAn=0) after the hold time has elapsed.

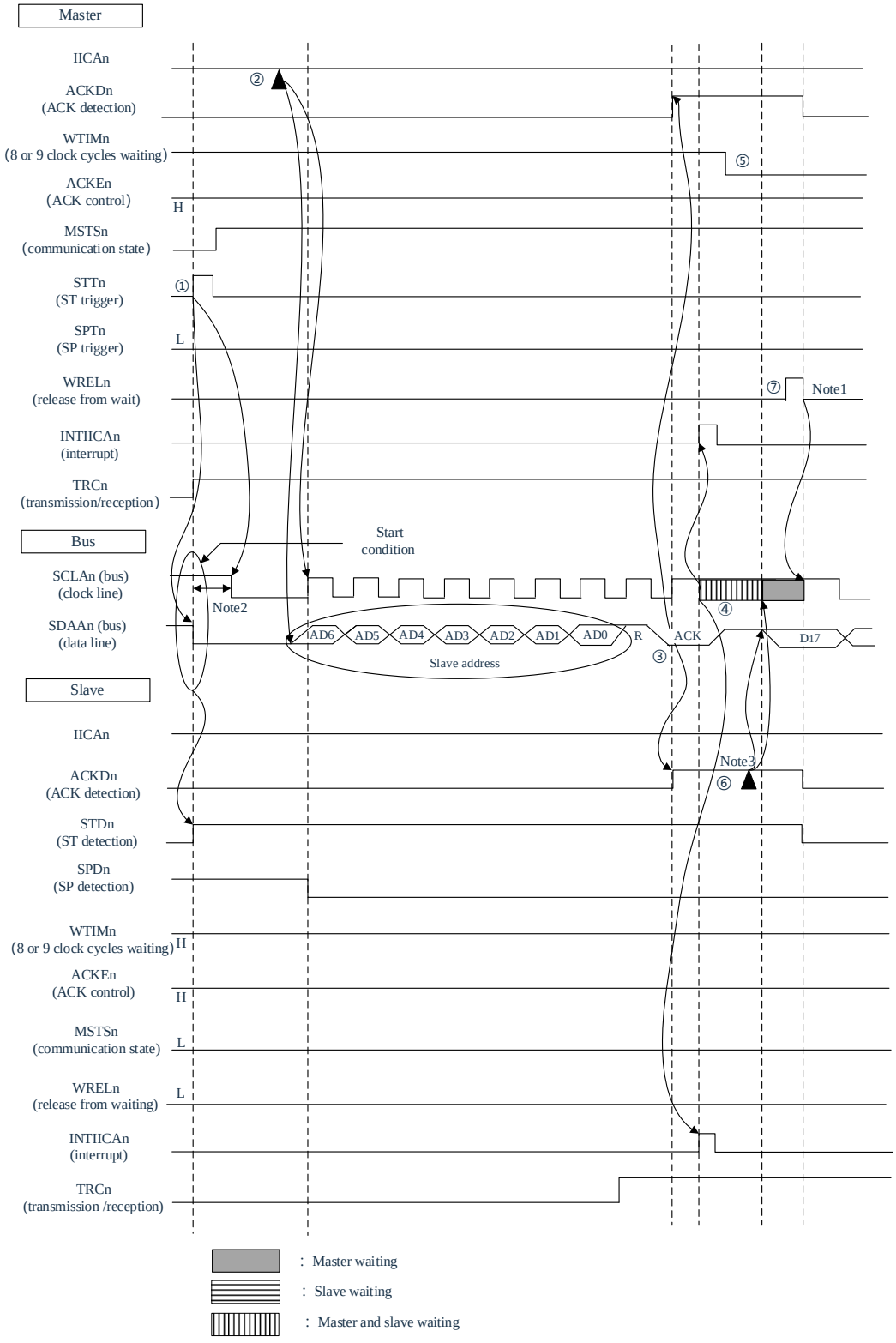
<3> The master device writing the address + R/W (transmission) to the IICA shift register (IICAn) enables the slave address to be transmitted.

Note: n=0

Figure 16-23 Example of slave to master communication

(Master: selects 8 clocks to wait, slave: selects 9 clocks to wait) (1/3)

(1) Start Condition ~ Address ~ Data




Note 1: To release the master from waiting during transmission, write data to the IICAn or set the WRELn bit.

Note 2: Make sure that the time between the fall of the SDAAn pin signal and the fall of the SCLAn pin signal is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 3: To release the slave from waiting during transmission write data to the IICn instead of setting the WRELn bit.

Figure 16-23 shows ① to ⑦ of "(1) Start condition~Address~Data" as follows.

① If the master sets the start condition trigger set (STTn=1), the bus data line (SDAAn) drops and the start condition is generated (SDAAn is changed from "1" to "0" by SCLAn=1). Thereafter, if a start condition is detected, the master enters the master communication state (MSTSn=1) and after the hold time elapses the bus clock line drops (SCLAn=0), ending the communication preparation.

② If the master writes address +R (receive) to the IICA shift register n (IICAn), the slave address is sent.

③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same [Note], an ACK is sent to the master through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) [Note].

⑤ The master changes the wait sequence to the 8th clock (WTIMn=0).

⑥ The slave writes and sends data to the IICAn register, relieving the slave of the wait.

⑦ The master relieshes the wait (WRELn=1) and begins data transfer from the slave.

Note 1: If the transmitted address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address transmit end interrupt) for both ACK and NAK.

Note 2: ①~⑲ of Figure 16-23 shows a series of operational steps for data communication via the $I^2C$ bus.

Note 3: "(1) Start Condition ~ Address ~ Data" of Figure 16-23 illustrates steps ①~⑦.

Note 4: "(2) Address~Data~Data" of Figure 16-23 illustrates steps ③~⑫.

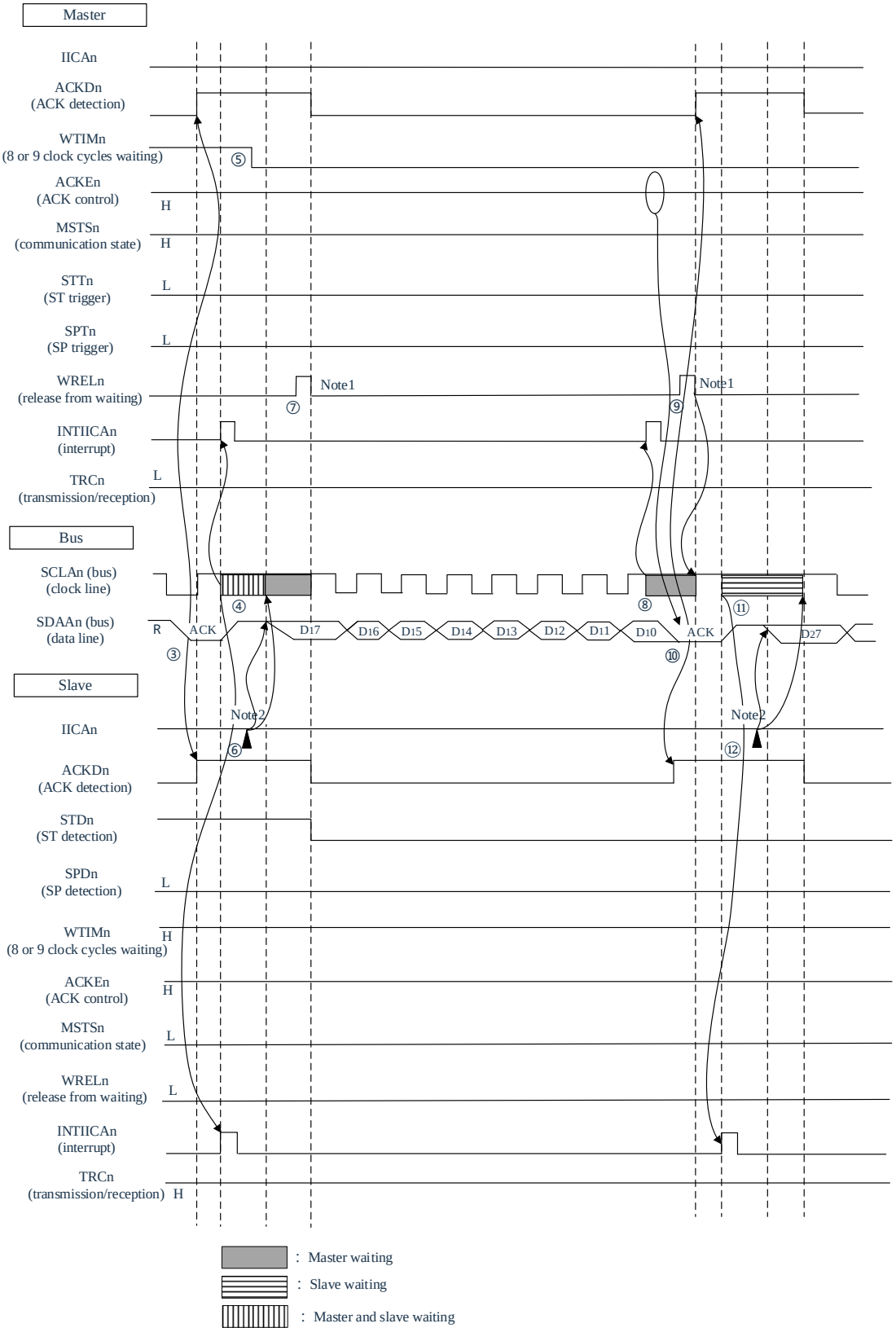
Note 5: "(3) Data~Data~Stop Condition" of Figure16-23 illustrates steps ⑧~⑲.

Note 6: n=0.

Figure 16-23 Example of slave to master communication

(Master: selects 8 clocks to wait, slave: selects 9 clocks to wait) (2/3)

(2) Address ~ Data ~ Data




Note 1: To release the master from waiting during reception, the IICn must be set to “FFH” or the WRELn bit must be set.

Note 2: To release the slave from waiting during transmission, write data to the IICn instead of setting the WRELn bit.

Figure 16-23 shows ③~⑫ of "(2) Address~Data~Data" as follows.

③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same [Note], the ACK is sent to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address transmit end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) [Note].

⑤ The master changes the wait sequence to the 8th clock (WTIMn= 0).

⑥ The slave writes and sends data to the IICA shift register n (IICAn) to release the slave's wait.

⑦ The master releases the wait (WRELn=1) and begins data transfer from the slave.

⑧ The master enters a waiting state (SCLAn= 0) on the falling edge of the 8th clock and produces an interrupt (INTIICAn: Transmit End Interrupt). Because the ACKEn bit of the master is 1, the ACK is sent to the slave through the hardware.

⑨ The master reads the received data and releases the wait (WRELn=1).

⑩ The slave detects ACK (ACKDn=1) on the rising edge of the 9th clock.

⑪ The slave enters a waiting state on the falling edge of the 9th clock (SCLAn = 0) and produces an interrupt (INTIICAn: Transmit End Interrupt)

⑫ If the slave writes and transmits data to the IICAn register, the slave's wait is released and the data transfer from the slave to the master is started.

Note 1: If the transmitted address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address transmit end interrupt) for both ACK and NAK.

Note 2: ①~⑲ of Figure 16-23 shows a series of operational steps for data communication via the $I^2C$ bus.

Note 3: "(1) Start Condition ~ Address ~ Data" of Figure 16-23 illustrates steps ①~⑦.

Note 4: "(2) Address~Data~Data" of Figure 16-23 illustrates steps ③~⑫.

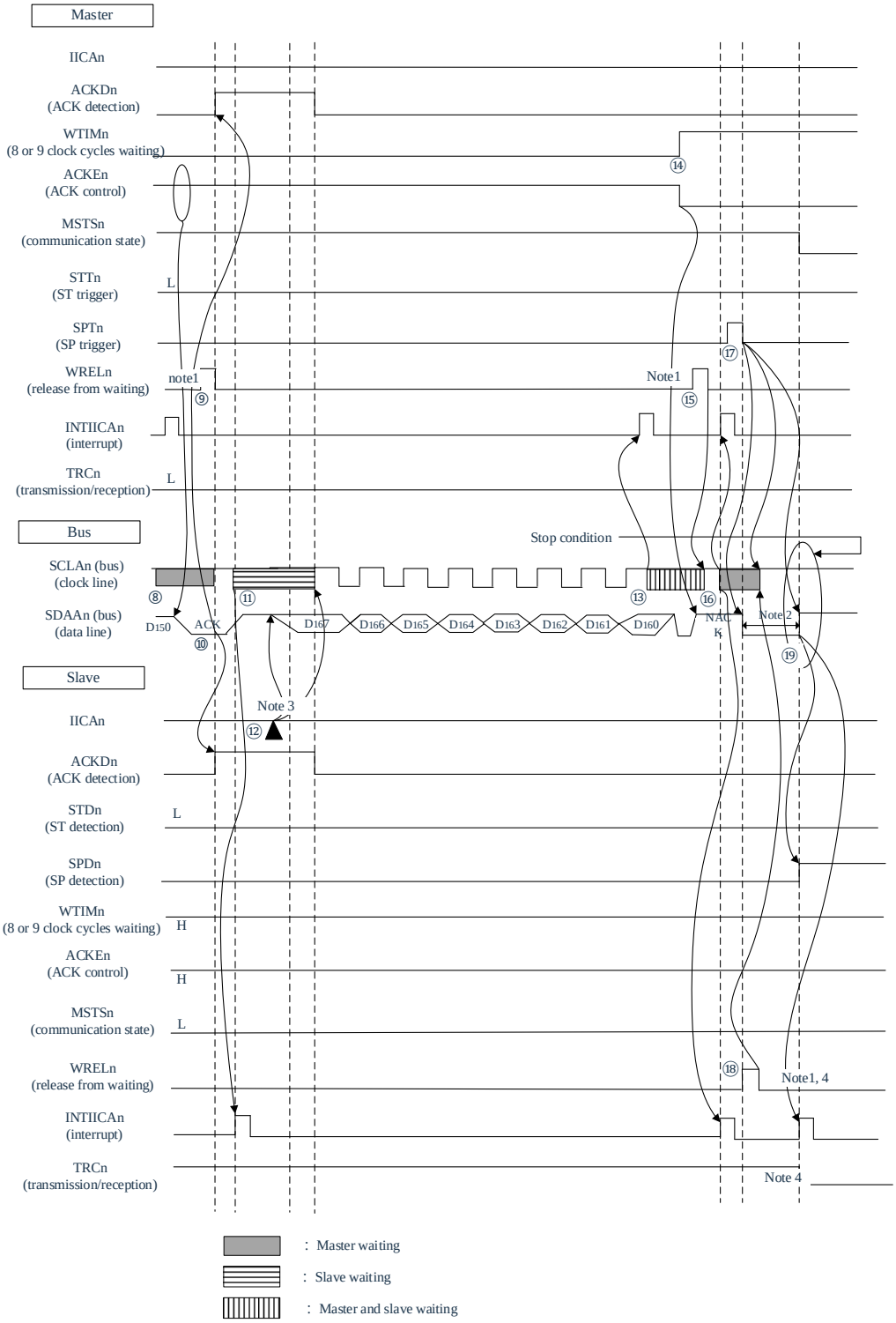
Note 5: "(3) Data~Data~Stop Condition" of Figure 16-23 illustrates steps ⑧~⑲.

Note 6: n=0.

Figure 16-23 Example of slave to master communication

(Master: selects 8 to 9 clocks to wait, slave: selects 9 clocks to wait) (3/3)

(3) Data ~ Data ~ Stop Condition




Note 1: To release the wait, the IICAn must be set to “FFH” or the WRELn bit must be set.

Note 2: After the stop condition is issued, the time from the SCLAn pin signal to generate the stop condition is at least 4.0μs when set to standard mode and at least 0.6μs when set to fast mode.

Note 3: To release the slave from waiting during transmission, you must write data to the IICn instead of setting the WRELn bit.

Note 4: The TRCn bit is cleared if the wait is released by setting the WRELn bit during the slave transmission.

Figure 16-23 shows ⑧~⑲ of "(3) Data~Data~Stop Condition" as follows.

⑧ The master enters a waiting state (SCLAn = 0) on the falling edge of the 8th clock and generates an interrupt (INTIICAn: Transmit End Interrupt). Because the ACKEn bit of the master is 0, the ACK is sent to the slave through the hardware

⑨ The master reads the received data and releases the wait (WRELn=1).

⑩ The slave detects ACK (ACKDn=1) on the rising edge of the 9th clock.

⑪ The slave enters a waiting state on the falling edge of the 9th clock (SCLAn= 0) and generates an interrupt (INTIICAn: transmit end interrupt).

⑫ If the slave writes and transmits data to the IICA shift register n (IICAn), the slave's wait is released and the transfer of data from the slave to the master begins.

⑬ The master generates an interrupt (INTIICAn: transmit end interrupt) on the falling edge of the 8th clock and enters a waiting state (SCLAn=0). Because ACK control (ACKEn=1) occurs, the bus data line at this stage becomes low (SDAAn=0)

⑭The master sets the NACK Acknolwdge (ACKEn=0) and changes the wait sequence to the 9th clock (WTIMn=1). If the master releases the wait (WRELn=1), the slave detects NACK (ACKDn=0) on the rising edge of the 9th clock.

⑮ Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock, and both produce an interrupt (INTIICAn: transmit end interrupt).

⑯ If the master issues a stop condition (SPTn=1), the bus data cable (SDAAn=0) is cleared and the master's wait is released. After that, the master is on standby until the bus clock line is set (SCLAn=1).

⑰ The slave stops sending after confirming the NACK, in order to end the communication, the wait is released (WRELn=1). If the slave wait is released, the bus clock line is set (SCLAn=1).

⑱ If the master confirms that the bus clock line is set (SCLAn=1), the bus data line is set after the stop condition preparation time has elapsed.

⑲ (SDAAn=1), and then issue a stop condition (SDAAn is changed from 0 to 1 by SCLAn=1). If a stop condition is generated, the slave detects the stop condition and generates an interrupt (INTIICAn: Stop Condition Interrupt).

# Chapter 17 Serial Peripheral Interface Controller (SSP/SPI)

## 17.1Overview

The Serial Peripheral Interface (SPI) is a synchronous serial data communication protocol that operates in full-duplex mode. Devices can operate in master/slave mode and communicate with each other using a 4-wire bidirectional interface. SPI performs serial-to-parallel conversion when receiving data from a peripheral device and parallel-to-serial conversion when sending data to a peripheral device. The SPI controller can be configured as either a master or a slave device.

## 17.2Features

- Support master or slave mode.
- Full-duplex.
- Configurable bit length for transmission (4-bit to 16-bit).
- MSB first for transmission/reception.
- Built-in a receive buffer and a transmit buffer.

## 17.3Register Mapping

(SSP0 base address= 0x4006_3000) RO: read only; WO: write only; R/W: read/write.

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CON | 0x000 | R/W | SSP Control Register | 0x0 |
| STAT | 0x004 | RO | SSP Status Register | 0x3 |
| DAT | 0x008 | R/W | SSP Data Register | 0x0 |
| CLK | 0x00C | R/W | SSP Clock Control Register | 0x0 |
| IMSC | 0x010 | R/W | SSP Interrupt Enable Register | 0x0 |
| RIS | 0x014 | RO | SSP Interrupt Source Status Register | 0x8 |
| MIS | 0x018 | RO | SSP Enabled Interrupt Status Register | 0x0 |
| ICLR | 0x01C | WO | SSP Interrupt Clear Register | 0x0 |
| CSCR | 0x028 | R/W | SSP Software Chip Select Signal Register | 0x0 |

# 17.4Register Description

## 17.4.1 SSP Control Register (CON)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:12 | - | Reserved | - |
| 11 | LBM | Loopback mode enable bit<br>0: Normal operaton mode<br>1: Loopback mode, connect serial inputs to serial outputs | 0 |
| 10 | SSPEN | SSP enable bit<br>0: Disable<br>1: Enable | 0 |
| 9 | MS | Master/slave mode select bit<br>0: Master mode<br>1: Slave mode | 0 |
| 8 | - | Reserved | 0 |
| 7 | CPH | Clock phase control bit<br>0: SSP samples data on the first clock edge<br>1: SSP samples data on the second clock edge | 0 |
| 6 | CPO | Clock output polarity select bit<br>0: SPI_CLK is low when idle<br>1: SPI_CLK is high when idle | 0 |
| 5:4 | FRF | Frame format<br>0x0: SPI-compatible frame format<br>0x1: TISS-compatible frame format<br>0x2: Microwire-compatible frame format<br>0x3: Reserved | 0x0 |
| 3:0 | DSS | Data transfer length select bit<br>0x0: Reserved<br>0x1: Reserved<br>0x2: Reserved<br>0x3: 4-bit<br>0x4: 5-bit<br>0x5: 6-bit<br>0x6: 7-bit<br>0x7: 8-bit<br>0x8: 9-bit<br>0x9: 10-bit<br>0xA: 11-bit<br>0xB: 12-bit<br>0xC: 13-bit<br>0xD: 14-bit<br>0xE: 15-bit<br>0xF: 16-bit | 0x0 |

## 17.4.2 SSP Status Register (STAT)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:5 | - | Reserved | - |
| 4 | BSY | Busy flag bit, read-only<br>0: SSP is idle<br>1: SSP is transmitting/receiving data or the Transmit Buffer has been written data | 0 |
| 3:0 | - | Reserved | 0x3 |

## 17.4.3 SSP Data Register (DAT)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Reserved | - |
| 15:0 | DATA | When writing data to this register, the data will be written into the transmit register and sent out when there is no data being transmitted on the bus. If there is data being transmitted on the bus, the data will be stored in the buffer and sent after the previous transmission is completed. The minimum interval between two transmissions is 3 SSPCLK clocks.<br>When the data length is less than 16 bits, it needs to be right-aligned.<br>When reading this register, the most recently received data is read, and when the length of the data is less than 16 bits, it should be right-aligned. | 0x0 |

## 17.4.4 SSP Clock Controller (CLK)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Reserved | - |
| 15:8 | M | SSPCLK = PCLK / ( (M+1)×N)<br>N is an even number from 2 to 254 | 0x0 |
| 7:0 | N | | 0x0 |

## 17.4.5 SSP Interrupt Enable Register (IMSC)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3 | TXIM | Transmit Buffer interrupt enable bit<br>0: Disable empty data interrupt in transmit Buffer<br>1: Enable empty data interrupt in transmit Buffer | 0 |
| 2 | RXIM | Receive Buffer interrupt enable bit<br>0: Disable received data interrupt in receive Buffer<br>1: Enable received data interrupt in receive Buffer | 0 |
| 1 | RTIM | Receive Buffer timer overflow interrupt enable bit<br>0: Disable timer overflow interrupt in receive buffer.<br>1: Enable timer overflow interrupt in receive buffer. (Overflow time: 32×SSPCLK) | 0 |
| 0 | RORIM | Receive Buffer overflow interrupt enable bit<br>0: Disable overflow interrupt in receive buffer.<br>1: Enable overflow interrupt in receive buffer. | 0 |

### 17.4.6 SSP Interrupt Source Status Register (RIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3 | TXRIS | Set when the transmit buffer is empty or data in the transmit buffer has been sent. (Automatically cleared when there is data in the transmit buffer) | 1 |
| 2 | RXRIS | Set when the receive buffer receives data. (Automatically cleared when there is no data in the receive buffer or data in the receive buffer has been read) | 0 |
| 1 | RTRIS | Set when the receive buffer receives data and remains unread for a timeout period. (Cleared by reading the data register or writing to the ICLR register) | 0 |
| 0 | RORRIS | When the receive buffer receives data and remains unread, and another frame of data is received, this bit is set, and the new data will be lost. (Cleared by writing to the ICLR register) | 0 |

### 17.4.7 SSP Enabled Interrupt Status Register (MIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:4 | - | Reserved | - |
| 3 | TXMIS | = TXIM & TXRIS | 0 |
| 2 | RXMIS | = RXIM & RXRIS | 0 |
| 1 | RTMIS | = RTIM & RTRIS | 0 |
| 0 | RORMIS | = RORIM & RORRIS | 0 |

### 17.4.8 SSP Interrupt Clear Register (ICLR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:2 | - | Reserved | - |
| 1 | RTIC | 1: Clear the RTRIS flag bit | 0 |
| 0 | RORIC | 1: Clear the RORRIS flag bit | 0 |

### 17.4.9 SSP Software Chip Select Signal Register (CSCR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:5 | - | Reserved | - |
| 4 | SPH | Slave chip select signal (SPI as slave)<br>0: After the data transfer of each frame is completed, the chip select signal is pulled high.<br>1: After the data transfer of the last frame is completed, the chip select signal is pulled high. | 0 |
| 3 | SWCS | Software chip select signal control bit in master mode<br>0: Output low level<br>1: Output high level | 0 |
| 2 | SWSEL | Chip select signal selection in master mode<br>0: The chip select signal is automatically controlled by the SPI module<br>1: The chip select signal is controlled by the SWCS bit | 0 |
| 1:0 | - | Reserved | - |

# Chapter 18 Enhanced DMA

## 18.1Functions of DMA

The DMA is a function of transferring data between memories without using the CPU. Start the DMA for data transfer via peripheral interrupts. When the DMA and CPU access the same unit in flash, SRAM0, SRAM1, or peripheral modules at the same time, their bus usage is higher than the CPU. When the DMA and CPU access flash, SRAM0, SRAM1, or different units in the peripheral module, respectively, the two do not interfere with each other and can be executed in parallel.

The specifications for DMA are shown in Table 18-1.

Table 18-1 Specifications for DMA (1/2)

| Item | | Specification |
|---|---|---|
| Startup source | | Up to 8 startup sources |
| Assignable control data | | 8 groups |
| The address space that can be transmitted | Address space | Full address range space |
| | Source | Full address range space is optional |
| | Target | Full address range space is optional |
| Maximum number of transfers | Normal mode | 65535 times |
| | Repeat mode | 65535 times |
| The maximum transfer block size | Normal mode (8-bit transfer) | 65535 bytes |
| | Normal mode (16-bit transfer) | 131070 bytes |
| | Normal mode (32-bit transfer) | 262140 bytes |
| | Repeat mode | 65535 bytes |
| Transfer unit | | 8-bit/16-bit/32-bit |
| Transfer mode | Normal mode | Ends after a transfer of the DMACTj register from 1 to 0. |
| | Repeat mode | After the transfer of the DMCTj register from 1 to 0 is completed, the address of the repeat area is initialized, and the DMRLDj is changed.<br>The value of the register continues after reloading into the DMCTj register. |
| Address control | Normal mode | Fixed or incremented |
| | Repeat mode | Fixed or incremented the address of the non-repeating area. |
| Priority of the start-up sources | | Refer to Table 18-4. |

Table 18-1 Specifications for DMA (2/2)

| Item | | Specification |
|---|---|---|
| Interrupt request | Normal mode | When a data transfer with a DMCTj register changed from 1 to 0, an interrupt of the start-up source is requested to the CPU and interrupt processing is performed. |
| | Repeat mode | When the RPTINT bit of the DMACRj register is 1 (enable an interrupt to be generated) and the data transfer of the DMACTj register changes from 1 to 0, an interrupt from the startup source is requested to the CPU and interrupt processing is performed. |
| Transfer start | | If the DMAENi0~DMAENi7 position of the DMAENi register is 1 (boot allowed), the transmission of data begins each time the DMA boot source occurs. |
| Transfer stop | Normal mode | • Set DMAENi0 to DMAENi7 to 0(disable startup).<br>• At the end of the data transfer when the DMACTj register changes from 1 to 0. |
| | Repeat mode | • Set DMAENi0~DMAENi7 bits to 0 (disable startup).<br>• At the end of data transfer when the RPTINT bit is 1(interrupt generation is allowed) and the DMACTj register changes from 1 to 0. |

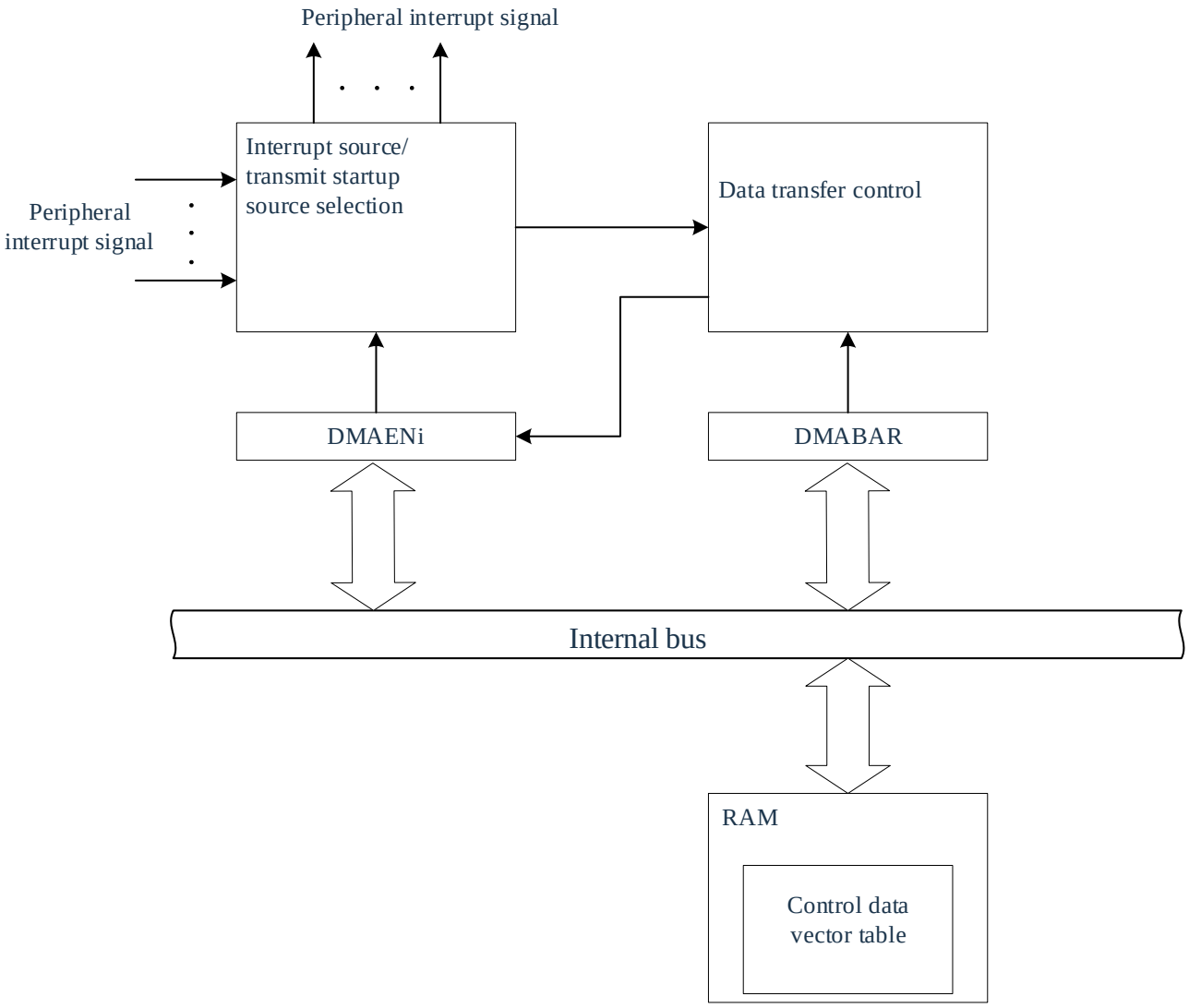
Note 1: In deep sleep mode, the flash memory cannot be used as a DMA transfer source because it is stopped.

Note 2: i=0, j=0~7.

# 18.2Structure of DMA

The block diagram for DMA is shown in Figure 18-1.

Figure 18-1 Block diagram of DMA



## 18.3Register Mapping

(Base address = 0x40020810) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| PER10 | 0x000 | R/W | Peripheral enable register 10 | 0x00 |

(Base address = 0x40005000) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| DMAEN0 | 0x000 | R/W | DMA start enable register 0 | 0x00 |
| DMABAR | 0x008 | R/W | DMA base address register | 0x00 |

(Base address = 0x40020D00) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| DMATGS | 0x000 | R/W | DMA trigger source selection register | 0x00 |
| TGSEN0 | 0x004 | R/W | Trigger source signal DMA enable register 0 | 0x00 |
| TGSEN1 | 0x008 | R/W | Trigger source signal DMA enable register 1 | 0x00 |
| TGSEN2 | 0x00C | R/W | Trigger source signal DMA enable register 2 | 0x00 |

The DMA control data is shown in Table 18-2.

The DMA control data is allocated in the DMA control data area of RAM. The DMA control data area and the 144-byte region containing the DMA vector table (which stores the starting address of the control data) are configured via the DMABAR register.

Table 18-2 DMA control data

| Register name | Symbol |
|---|---|
| DMA control register j | DMACRj |
| DMA block size register j | DMBLSj |
| DMA transfer count register j | DMACTj |
| DMA transfer count reload register j | DMRLDj |
| DMA source address register j | DMSARj |
| DMA target address register j | DMDARj |

Note: j=0~7.

# 18.4Registers for Controlling DMA

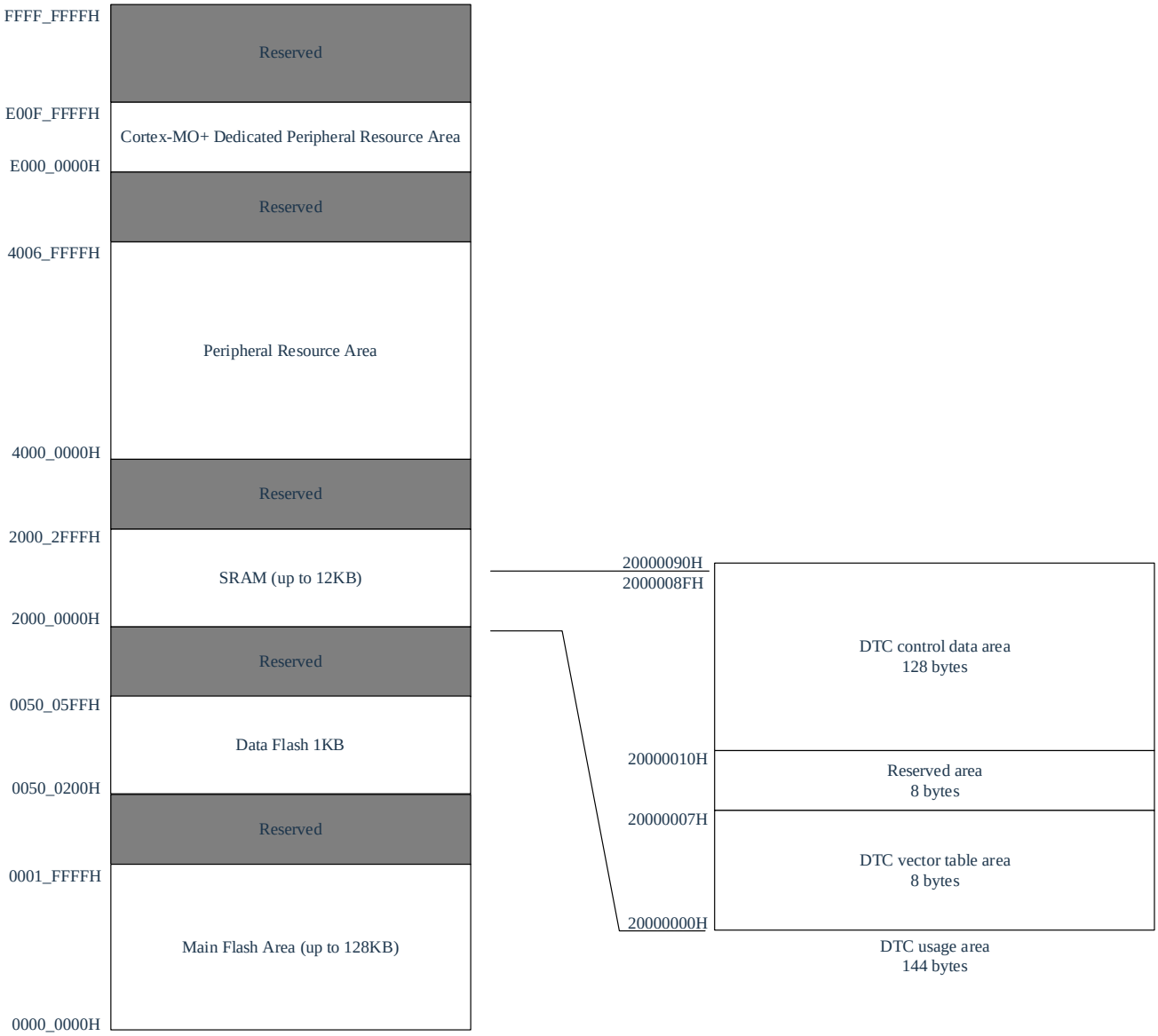
## 18.4.1 Assignment of DMA Control Data Area and DMA Vector Table Area

The DMABAR register configures the allocated DMA control data and the 144-byte region of the vector table into the RAM area.

An example of the memory mapping when the DMABAR register is set to 20000000H is shown in Figure 18-2.

The space within the 144 bytes of the DMA control data area that is not used by DMA can be used as RAM.

Figure 18-2 Example of memory mapping when the DMABAR register is set to “20000000H”



## 18.4.2 Controlling Data Assignment

Starting from the starting address, control data is allocated in the order of DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, and DMDARj registers ( j=0~7).

The starting address is set by the DMABAR register, and the lower 8 bits are set by the vector tables assigned by each starting source.

The assignment of control data is shown in Figure 18-3.

Note 1: The data in the DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, and DMDARj registers must be changed when the DMAENi0 to DMAENi7 bits of the corresponding DMAENi (i=0) are 0 (startup disabled).

Note 2: DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, and DMDARj cannot be accessed via DMA transfer.

Figure 18-3 Assignment of control data (set DMABAR to 20000000H)

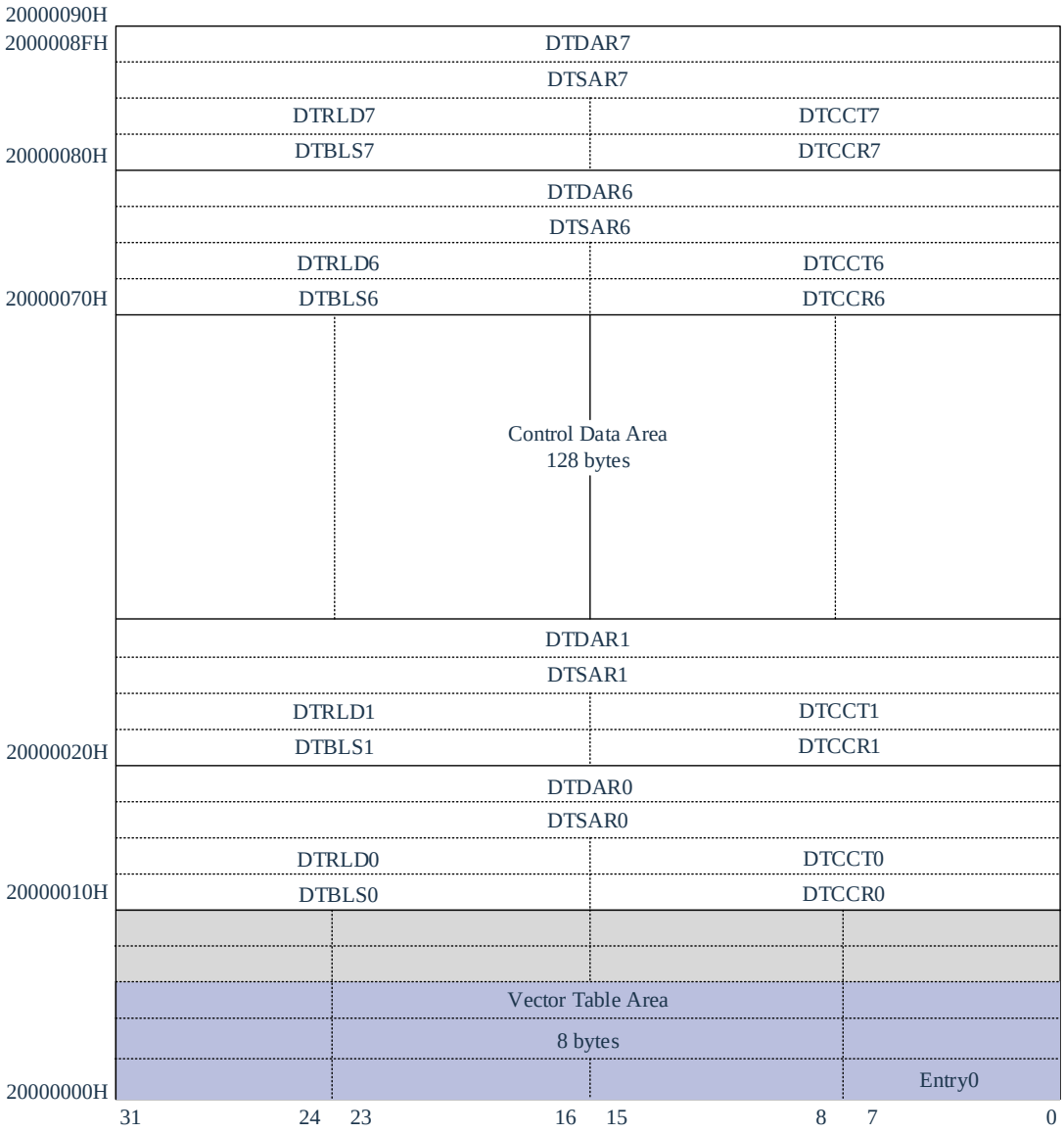


Table 18-3 Start addresses of control data

| j | Address |
|---|---|
| 7 | baseaddr+80H |
| 6 | baseaddr+70H |
| 5 | baseaddr+60H |
| 4 | baseaddr+50H |
| 3 | baseaddr+40H |
| 2 | baseaddr+30H |
| 1 | baseaddr+20H |
| 0 | baseaddr+10H |

Note: baseaddr: Setting values of the DMABAR register

## 18.4.3 Vector Table

Once the DMA is started, control data assigned to the DMA control data area is read by data read from a vector table assigned to each start source.

The DMA startup sources and vector addresses are shown in Table 18-4. The vector table of each startup source stores the data from “00H” to “07H” in 1 byte. One group of data is selected from the 8 groups of control data. The high 24 bits of the vector address are set by the DMABAR register, and the low 8 bits are assigned to “00H” to “17H” of the corresponding startup source.

Note: The start address of the DMA control data area set in the vector table must be changed when the DMAENi0 bit of the corresponding DMAENi (i=0) register is 0 (startup disabled).

Figure 18-4 Starting address and vector table of control data

When DMABAR set value is “20000000H” (example)

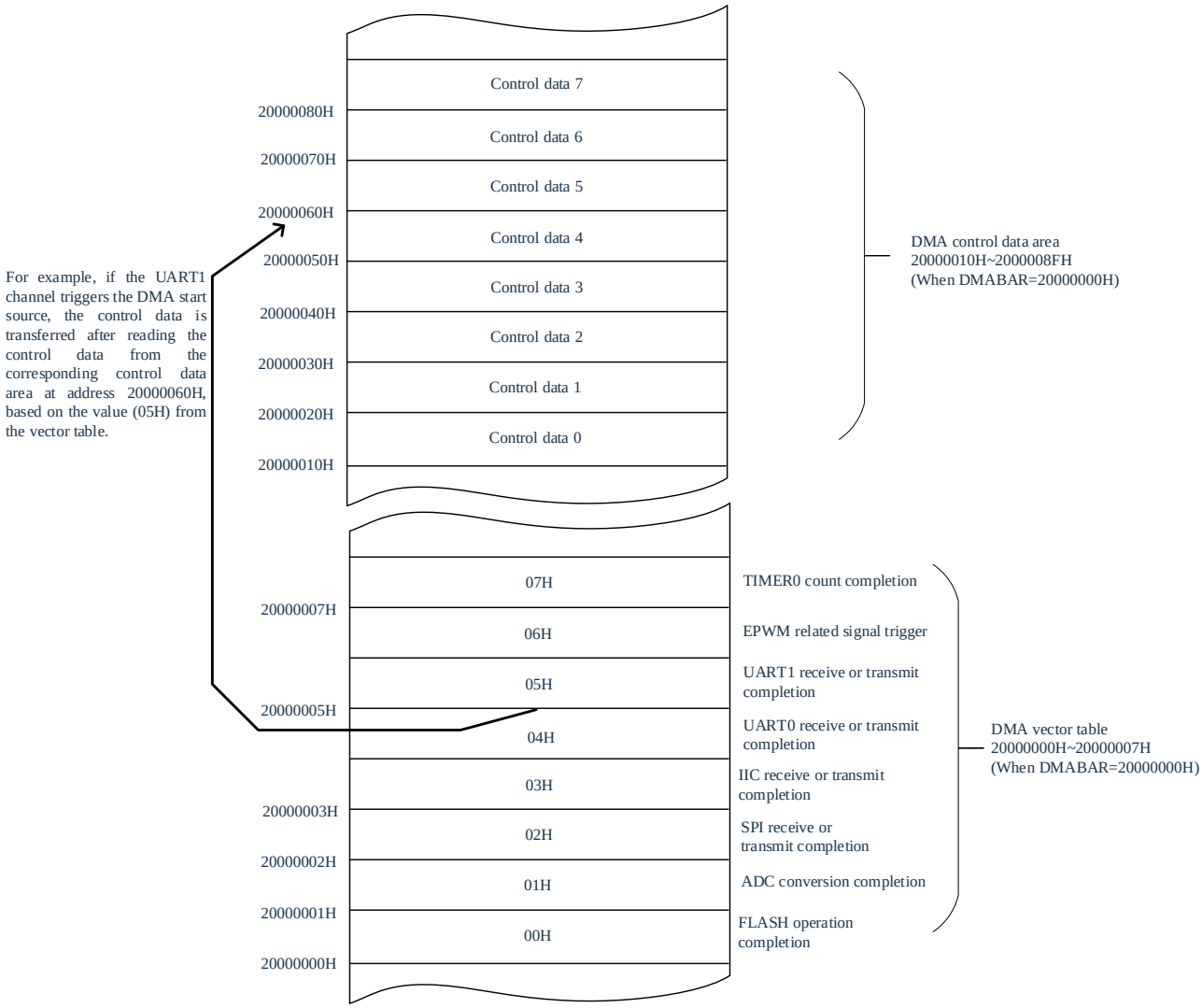


Table 18-4 DMA startup sources and vector addresses

| DMA start source (interrupt request generat source) | Source number | Vector address | Priority |
|---|---|---|---|
| Flash read/write/erase end | 0 | DMABAR register set address+00H | High ↕ Low |
| ADC transfer end | 1 | DMABAR register set address+01H | |
| SPI | 2 | DMABAR register set address+02H | |
| IIC | 3 | DMABAR register set address+03H | |
| UART0 | 4 | DMABAR register set address+04H | |
| UART1 | 5 | DMABAR register set address+05H | |
| EPWM | 6 | DMABAR register set address+06H | |
| TIMER0 | 7 | DMABAR register set address+07H | |

### 18.4.4 Peripheral Enable Register 10 (PER10)

The PER10 register is a register that sets a clock that is enabled or disabled to supply to each peripheral hardware. Reduce power consumption and noise by stopping clock supply to unused hardware. To use DMA, the bit3(DMAEN) must be set to 1.

The PER10 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register changes to “00H”.

Peripheral enable register 10 (PER10)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:4 | -- | Reserved | 0x0 |
| 3 | DMAEN | Control of DMA input clock supply<br>0: Stop to supply the input clock, DMA cannot run.<br>1: Supply the input clock, DMA can run. | 0 |
| 2:0 | -- | Reserved | 0x0 |

### 18.4.5 DMA Trigger Source Selection Register (DMATGS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7 | TIMER0_SEL | TIMER0 request DMA signal source selection<br>0: TIMER0 interrupt signal<br>1: TIMER0 trigger signal | 0 |
| 6 | EPWM_SEL | EPWM request DMA signal source selection<br>0: EPWM interrupt signal<br>1: EPWM trigger signal | 0 |
| 5 | UART1_SEL | UART1 request DMA signal source selection<br>0: UART1 interrupt signal<br>1: UART1 trigger signal | 0 |
| 4 | UART0_SEL | UART0 request DMA signal source selection<br>0: UART0 interrupt signal<br>1: UART0 trigger signal | 0 |
| 3 | - | Reserved | - |
| 2 | SPI_SEL | SPI request DMA signal source selection<br>0: SPI interrupt signal<br>1: SPI trigger signal | 0 |
| 1 | ADC_SEL | ADC request DMA signal source selection<br>0: ADC interrupt signal<br>1: ADC trigger signal | 0 |
| 0 | - | Reserved | - |

### 18.4.6 Trigger Source Signal DMA Enable Register 0 (TGSEN0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:28 | - | Reserved | - |
| 27 | ADCMPEN | ADC digital compare end request DMA enable bit<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 26:0 | ADCHENn<br>(n=0~26) | ADC channel conversion end request DMA enable bit<br>0: Disable DMA request<br>1: Enable DMA request | 0x0 |

### 18.4.7 Trigger Source Signal DMA Enable Register 1 (TGSEN1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:18 | - | Reserved | - |
| 17 | PWMPTG1EN | PWM compare point 1 request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 16 | PWMPTG0EN | PWM compare point 0 request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 15:8 | PWMZIFENn<br>(n=0~7) | PWM zero point request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0x0 |
| 7:0 | PWMPIFENn<br>(n=0~7) | PWM period point request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0x0 |

## 18.4.8 Trigger Source Signal DMA Enable Register 2 (TGSEN2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:18 | - | Reserved | - |
| 15 | UART1BUFEN | UART1 transmit data buffer empty request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 14 | UART1TXOEN | UART0 transmit end request DMA enable (Data has been transmitted)<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 13 | UART1TXDEN | UART0 transmit end request DMA enable (The last bit of data is transmitted from the buffer)<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 12 | UART1RXEN | UART1 receive end request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 11 | UART0BUFEN | UART0 transmit data buffer empty request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 10 | UART0TXOEN | UART0 transmit end request DMA enable (Data has been transmitted)<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 9 | UART0TXDEN | UART0 transmit end request DMA enable (The last bit of data is transmitted from the buffer)<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 8 | UART0RXEN | UART0 receive end request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 7:4 | - | Reserved | - |
| 3 | SPITXBUSY | SPI bus idle request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 2 | SPITXBUFEN | SPI data buffer empty request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 1 | SPITXEN | SPI transmit end request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |
| 0 | SPIRXEN | SPI receive end request DMA enable<br>0: Disable DMA request<br>1: Enable DMA request | 0 |

## 18.4.9 DMA Control Register j (DMACRj) (j=0~7)

The DMACRj register controls the operation mode of the DMA.

Address: refer to 18.4.2 Controlling Data Assignment  After reset: undefined value  R/W

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:8 | - | Reserved | Undefined value |
| 7: 6 | SZ | Transfer data length selection<br>00: 8 bits<br>01: 16 bits<br>10: 32 bits<br>11: Settings are disabled | |
| 5 | RPTINT | Enable/disable repeat mode interrupt Note1<br>0: Interrupts are disabled<br>1: Interrupts are enabled | |
| 4 | CHNE | Chain transfer enabled/disabled Note2<br>0: Chain transfer is disabled.<br>1: Chain transfer is enabled. | |
| 3 | DAMOD | Control of the transfer target address Note3<br>0: Fixed<br>1: Incremental | |
| 2 | SAMOD | Control of transfer source address Note4<br>0: Fixed<br>1: Incremental | |
| 1 | RPTSEL | Selection of repeat area Note5<br>0: The transfer target is a repeat area.<br>1: The transfer source is a repeat area. | |
| 0 | MODE | Selection of transfer modes<br>0: Normal mode<br>1: Repeat mode | |

Note 1: When the MODE bit is 0 (Normal Mode), the setting of the RPTINT bit is invalid.

Note 2: The CHNE bit in the DMACR7 register must be set to 0 (Disable chain transfer).

Note 3: When the MODE bit is 1 (Repeat Mode) and the RPTSEL bit is 0 (Transfer target is the repeat area), the setting of the DAMOD bit is invalid.

Note 4: When the MODE bit is 1(Repeat Mode) and the RPTSEL bit is 1 (Transfer source is the repeat area), the setting of the SAMOD bit is invalid.

Note 5: When the MODE bit is 0 (Normal Mode), the setting of the RPTSEL bit is invalid.

Note 6: The DMACRj registers cannot be accessed via DMA transfer.

## 18.4.10 DMA Block Size Register j (DMBLSj) (j=0~7)

This register sets the block size for data transfer to be initiated once.

Address: refer to 18.4.2 Controlling Data Assignment After reset: undefined value R/W

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:0 | DMBLSj | Set the block size for a single data transfer initiation. | Undefined |

| DMBLSj | Transfer block size | | |
|---|---|---|---|
| | 8-bit transfer | 16-bit transfer | 32-bit transfer |
| 00H | Settings are disabled | Settings are disabled | Settings are disabled |
| 01H | 1 byte | 2 bytes | 4 bytes |
| 02H | 2 bytes | 4 bytes | 8 bytes |
| 03H | 3 bytes | 6 bytes | 12 bytes |
| • • • | • • • | • • • | • • • |
| FDH | 253 bytes | 506 bytes | 1012 bytes |
| FEH | 254 bytes | 508 bytes | 1016 bytes |
| FFH | 255 bytes | 510 bytes | 1020 bytes |
| • • • | • • • | • • • | • • • |
| FFFFH | 65535 bytes | 131070 bytes | 262140 bytes |

Note 1: The DMBLSj registers cannot be accessed via DMA transfer.

## 18.4.11 DMA Transfer Count Register j (DMACTj) (j=0~7)

This register sets the number of DMA data transfers. It is decremented by 1 whenever 1 DMA transfer is initiated.

Address: refer to 18.4.2 Controlling Data Assignment After reset: undefined value R/W

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:0 | DMACTj | Set the number of DMA data transfers. It decrements by 1 each time a DMA transfer is initiated. | Undefined |

| DMACTj | Transfer count |
|---|---|
| 00H | Settings are disabled |
| 01H | 1 time |
| 02H | 2 times |
| 03H | 3 times |
| •<br>•<br>• | •<br>•<br>• |
| FDH | 253 times |
| FEH | 254 times |
| FFH | 255 times |
| •<br>•<br>• | •<br>•<br>• |
| FFFFH | 65535 times |

Note 1: The DMACTj registers cannot be accessed via DMA transfer.

## 18.4.12 DMA Transfer Count Reload Register j (DMRLDj) (j=0~7)

This register sets the initial value of the transfer number register in repeat mode. In Repeat mode, since the value of this register is reloaded into the DMACT register, the set value must be the same as the initial value of the DMACT register.

Address: refer to 18.4.2 Controlling Data Assignment After reset: undefined value R/W

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:0 | DMRLDj | Set the initial value of the transfer count register in repeat mode. | Undefined |

Note: The DMRLDj registers cannot be accessed via DMA transfer.

## 18.4.13 DMA Source Address Register j (DMSARj) (j=0~7)

This register specifies the transfer source address when the data is transferred.

When the SZ bit of the DMACRj register is 01 (16-bit transfer), the lowest bit is ignored and processed as an even address.

When the SZ bit of the DMACRj register is 10 (32-bit transfer), the low 2 bits are ignored and processed as word addresses.

Address: refer to 18.4.2 Controlling Data Assignment After reset: undefined value R/W

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | DMSARj | Set the source address for data transfer. | Undefined |

Note: The DMSARj registers cannot be accessed via DMA transfer.

## 18.4.14 DMA Target Address Register j (DMDARj) (j=0~7)

This register specifies the target address of the transfer when the data is transferred.

When the SZ bit of the DMACRj register is 01 (16-bit transfer), the lowest bit is ignored and processed as an even address.

When the SZ bit of the DMACRj register is 10 (32-bit transfer), the low 2 bits are ignored and processed as word addresses.

Address: refer to 18.4.2 Controlling Data Assignment After reset: undefined value R/W

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | DMDARj | Set the target address for data transfer. | Undefined |

Note: The DMDARj registers cannot be accessed via DMA transfer.

## 18.4.15 DMA Start Enable Register i (DMAEN0)

This is an 8-bit register that controls whether or not starting DMA through each interrupt source. The correspondence between the interrupt sources and the DMAEN0 bit is shown in Table 18-5.

The DMAEN register can be set by an 8-bit memory manipulation instruction.

Note 1: The DMAENi0 to DMAENi7 bits must be changed at a location that does not generate a startup source corresponding to this bit.

Note 2: The DMAENi registers cannot be accessed via DMA transfer.

Note 3: The assigned function varies from product to product and must be set to 0 if no function is assigned.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | DMAEN0n<br>(n=0~7) | DMA start enable Note1 (n=0~7)<br>0: Start disabled<br>1: Start enabled | 0x0 |

Note 1: Based on the conditions for the transfer end interrupt, the DMAENin bit is set to 0 (start disabled).

n=0,1,2,3,4,5,6,7.

Table 18-5 Mapping of interrupt sources to the DMAEN00~DMAEN07 bits

| Register | DMAENi7 bit | DMAENi6 bit | DMAENi5 bit | DMAENi4 bit | DMAENi3 bit | DMAENi2 bit | DMAENi1 bit | DMAENi0 bit |
|---|---|---|---|---|---|---|---|---|
| DMAEN0 | TIMER0 overflow | EPWM conversion completion | UART1 transmit /receive completion | UART0 transmit /receive completion | IIC transmit /receive completion | SPI transmit /receive completion | ADC conversion completion bit | Flash erase/write completion |

Note 1: Bits without assigned functions must be set to 0.

Note 2: i=0

## 18.4.16 DMA Base Address Register (DMABAR)

This is a 32-bit register that sets the vector address that holds the starting address of the DMA control data area and the address of the DMA control data area.

Note 1. The DMABAR register must be changed with all DMA startup sources set to disable startup.

Note 2. The DMABAR register can be rewritten only 1 time.

Note 3. Access to the DMABAR register cannot be performed via DMA transfer.

Note 4. Refer to 18.4.1 for the DMA control data area and DMA vector table area assignment.

Note 5. To set this register, keep it 256 Byte aligned, that is, the lower 8 bits are set to zero. The DMA hardware ignores the lower 8 bits.

Note 6. The register can only be accessed by WORD, BYTE and HALFWORD access are ignored.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | DMABARj | Set the higher 24 bits of the target address for data transfer. | 0x0 |
| 7:0 | DMABARj | Set the lower 8 bits of the target address for data transfer to 0, due to 256-byte alignment. | 0x0 |

Note: j=0

# 18.5Operation of DMA

Once the DMA is started, the control data is read from the DMA control data area, data transfer is performed based on the control data, and the control data transferred is written back. It is possible to store 8 sets of control data to a DMA control data area and to perform transfer of 8 sets of data. The transfer mode has a normal mode and a repeat mode, and the transfer size has 8-bit transfer, 16-bit transfer and 32-bit transfer. When the CHNE bit of the DMACRj (j=0~7) register is 1 (chain transfer enabled), the continuous data transfer (chain transfer) is read by a start source.

The transfer source address and the transfer target address are specified through the 32-bit DMSARj register and the 32-bit DMDARj register, respectively. After the data is transferred, the values of the DMSARj register and the DMDARj register are incremented or fixed according to the control data.

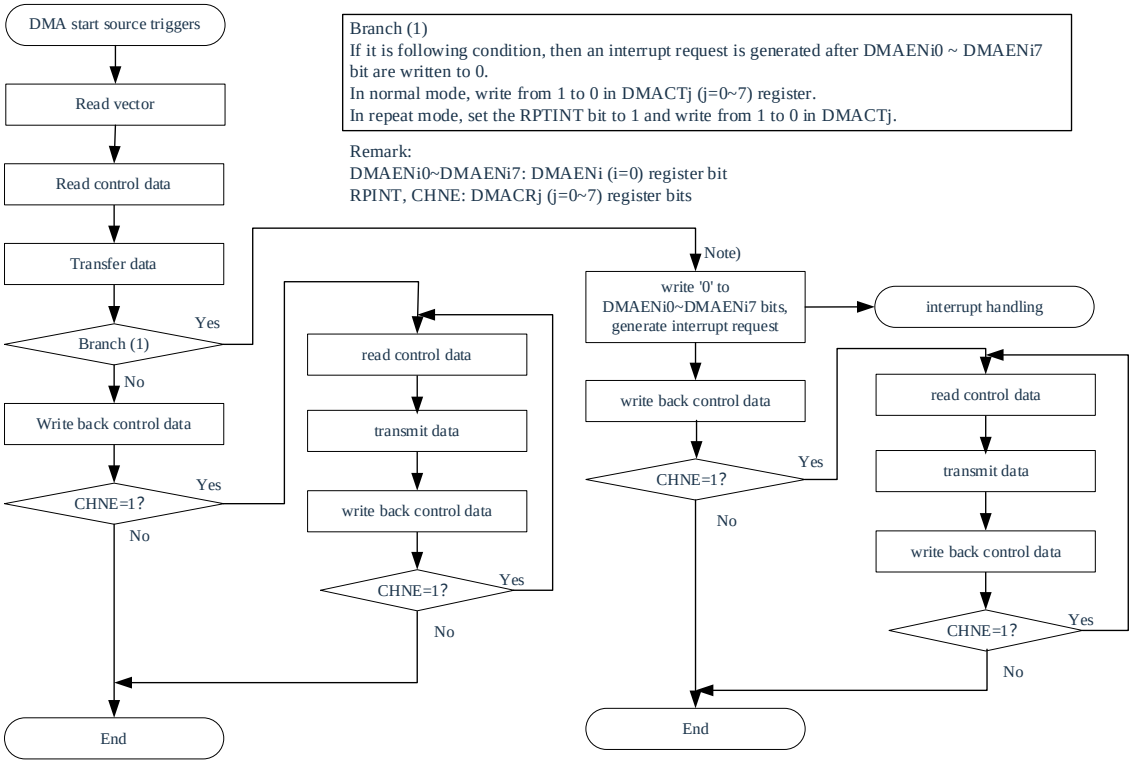
## 18.5.1 Start Sources

The DMA is started by a peripheral function interrupt signal and the interrupt signal for starting the DMA is selected by a DMAENi (i=0) register. When the data transfer (the initial transfer is continued during chain transfer) is set to the following two conditions, the corresponding DMAENi0-DMAENi7 bits of the DMAENi register are set to 0 (start disabled).

- In normal mode, the DMACTj (j=0~7) register is transferred to 0.
- In repeat mode, the RPTINT bit of the DMACRj register is 1 (interrupt enabled) and a transfer in which the DMACTj register becomes 0 is performed

The internal operation flowchart of the DMA is shown in Figure 18-5.

Figure 18-5 Internal operation flowchart of DMA




Note: In data transfers initiated with chain transfer enabled (CHNE=1), do not write 0 to the DMAENi0~DMAENi7 bits and no interrupt request will be generated.

## 18.5.2 Normal Mode

At 8-bit transfer, the transfer data of 1 start-up is 1~65535 bytes. In 16-bit transfer, the transfer data of 1 start-up is 2~131070 bytes; At 32-bit transfer, the transfer data of 1 start-up is 4 to 262140 bytes. The number of transfers is 1 to 65535. If a data transfer in which the DMACTj (j=0~7) register becomes 0 is performed, an interrupt request corresponding to the start source is generated to the interrupt controller during DMA operation, and the DMAENi0~DMAENi7 bits of the corresponding DMAENi (i=0) register are set to 0 (startup disabled).

The register functions and data transfers for normal mode are shown in Table 18-6 and Figure 18-6, respectively.

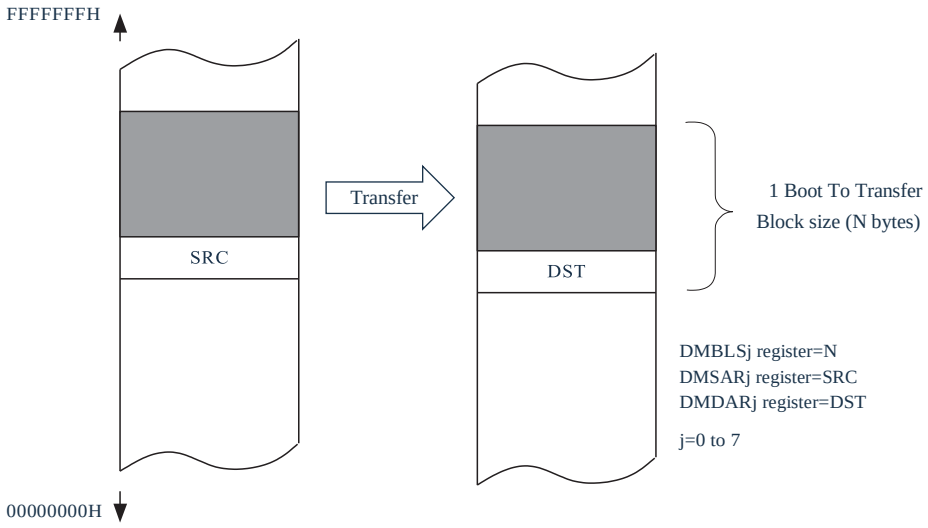
Note: j=0

Table 18-6 Register functions in normal mode

| Register name | Symbol | Function |
|---|---|---|
| DMA block size register j | DMBLSj | 1-start-up data block size to be transferred |
| DMA transfer count register j | DMACTj | Number of data transfers |
| DMA transfer count reload register j | DMRLDj | Not used Note |
| DMA source address register j | DMSARj | Data transfer source address |
| DMA target address register j | DMDARj | Data transfer target address |

Note: j=0

Figure 18-6 Normal mode data transfer




| Settings for the DMACR register | | | | Control of the source address | Control of the target address | The source address after transfer | The target address after transfer |
|---|---|---|---|---|---|---|---|
| DAMOD | SAMOD | RPTSEL | MODE | | | | |
| 0 | 0 | X | 0 | Fixed | Fixed | SRC | DST |
| 0 | 1 | X | 0 | Incremental | Fixed | SRC+N | DST |
| 1 | 0 | X | 0 | Fixed | Incremental | SRC | DST+N |
| 1 | 1 | X | 0 | Incremental | Incremental | SRC+N | DST+N |

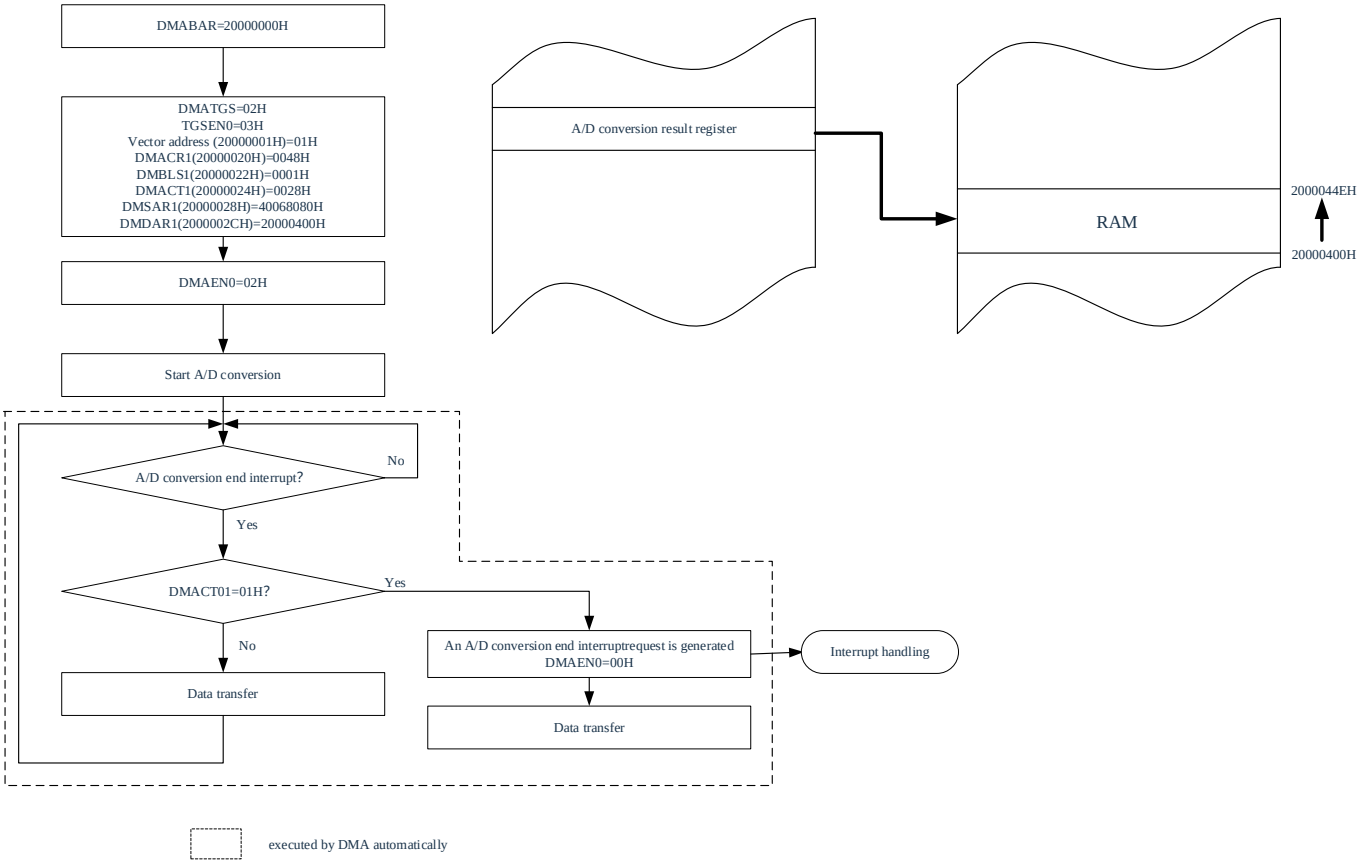
X: “0” or “1”

(1) Example 1 of normal mode usage: Continuous A/D conversion results

The DMA is started by an A/D conversion end interrupt, and the value of the A/D conversion result register is transferred to the RAM.

- The vector address is assigned at 20000001H and the control data is assigned at 20000020H~20000002FH.
- Transfer 2 bytes of data from A/D conversion result registers (40068080H, 40068081H) 40 times to 80 bytes of RAM 20000400H~2000044FH.

Figure 18-7 Example 1 of normal mode usage: Continuous A/D conversion results



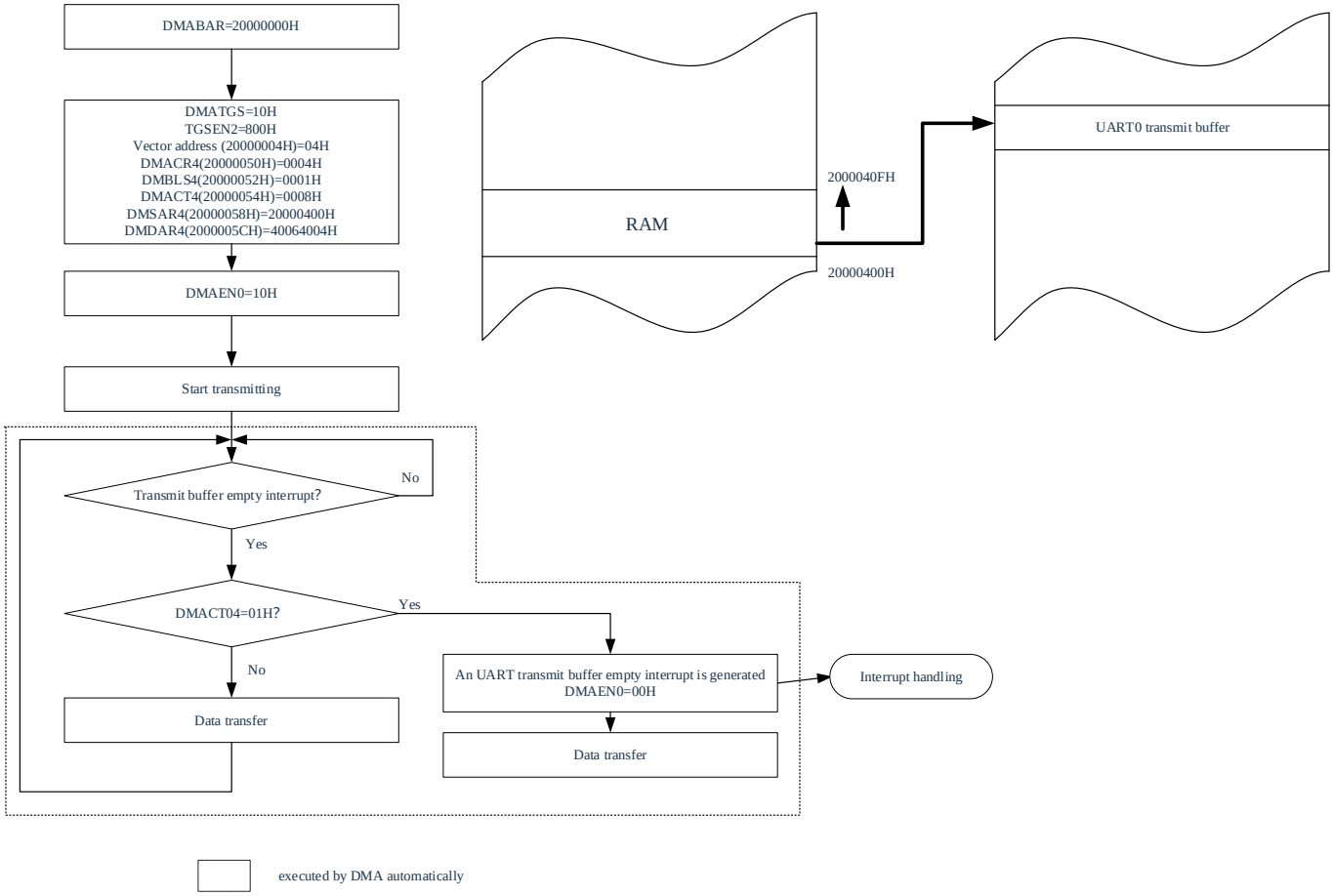

Since it is in normal mode, the value of the DMRLD01 register is not used.

(2) Example 2 of normal mode usage: UART0 continuous transmission

The DMA is initiated via the transmit buffer empty interrupt of the UART0 and the value of the RAM is transferred to the transmit buffer of the UART0.

- The vector address is assigned at 20000004H, and the control data is assigned in 20000050H~2000005FH.
- Transfer 8 bytes from 20000400H~20000407H of RAM to the UART0 transmit buffer (40064004H).

Figure 18-8 Example 2 of normal mode usage: UART0 continuous transmission




Since it is in normal mode, the value of the DMRLD12 register is not used. The first UART0 transmission must be initiated by software. DMA is triggered by the transmit buffer empty interrupt, and then subsequent transmissions are automatically performed.

### 18.5.3 Repeat Mode

The transfer data for 1 startup is 1 to 65535 bytes. The transfer source or the transfer target is specified as a repeat area, and the number of transfers from 1 to 65535. Once the specified number of transfers is completed, the DMACTj (j=0 to 7) register and the address specified as the repeat area are initialized, and the transfer is repeated. When the RPTINT bit in the DMACRj register is 1 (interrupt enabled) and a data transfer is performed with the DMACTj register turned to 0, an interrupt request corresponding to the start source is generated to the interrupt controller during DMA operation. And the DMAENi0 bit of the corresponding DMAENi (i=0) register is set to 0 (startup disabled). When the RPTINT bit of the DMACRj register is 0 (interrupt generation is prohibited), an interrupt request is not generated even if a data transfer in which the DMACTj register is changed to 0 is performed, and the DMAENi0 to DMAENi7 bits remain unchanged at 0.

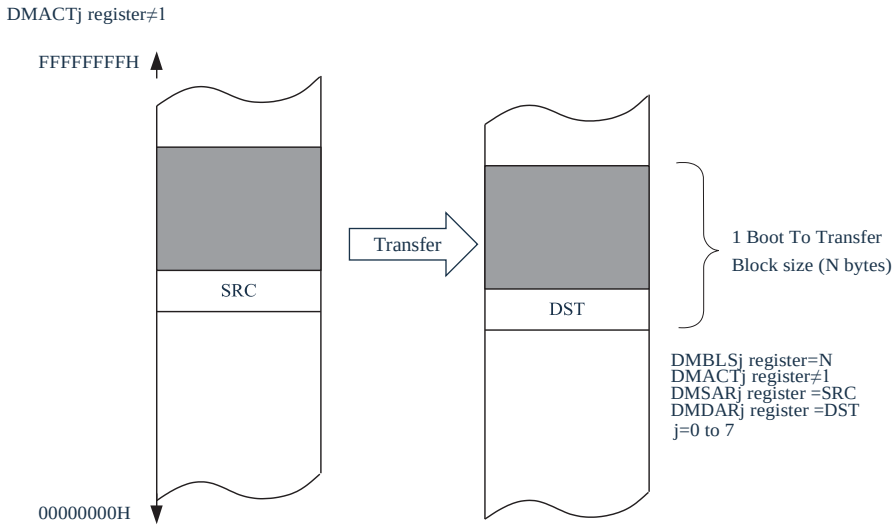
The register functions and data transfers for Repeat mode are shown in Table 18-7 and Figure 18-9, respectively.

Table 18-7 Register function for repeat mode

| Register name | Symbol | Function |
|---|---|---|
| DMA block size register j | DMBLSj | 1-start-up data block size to be transferred |
| DMA transfer count register j | DMACTj | Number of data transfers |
| DMA transfer count reload register j | DMRLDj | Reload the value of this register to the DMACT register. (Initialize the number of data transfers) |
| DMA source address register j | DMSARj | Data transfer source address |
| DMA target address register j | DMDARj | Data transfer target address |

Note: j=0~7

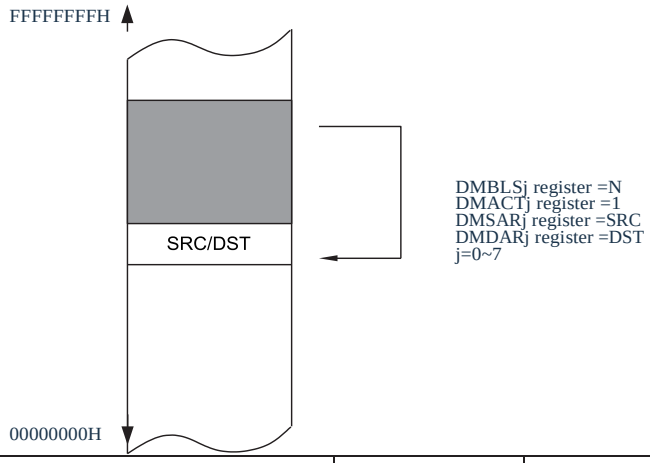
Figure 18-9 Repeat mode data transfer




| Setting for the DMACR register | | | | Control of the source address | Control of the target address | The source address after transfer | The target address after transfer |
|---|---|---|---|---|---|---|---|
| DAMOD | SAMOD | RPTSEL | MODE | | | | |
| 0 | X | 1 | 1 | Repeat area | Fixed | SRC+N | DST |
| 1 | X | 1 | 1 | Repeat area | Incremental | SRC+N | DST+N |
| X | 0 | 0 | 1 | Fixed | Repeat area | SRC | DST+N |
| X | 1 | 0 | 1 | Incremental | Repeat area | SRC+N | DST+N |

Remark: X: “0” or “1”

DMACTj register=1




| Settings for the DMACR register | | | | Control of the source address | Control of the target address | The source address after transfer | The target address after transfer |
|---|---|---|---|---|---|---|---|
| DAMOD | SAMOD | RPTSEL | MODE | | | | |
| 0 | X | 1 | 1 | Repeat area | Fixed | SRC | DST |
| 1 | X | 1 | 1 | Repeat area | Incremental | SRC | DST+N |
| X | 0 | 0 | 1 | Fixed | Repeat area | SRC | DST |
| X | 1 | 0 | 1 | Incremental | Repeat area | SRC+N | DST |

Remark: X: “0” or “1”

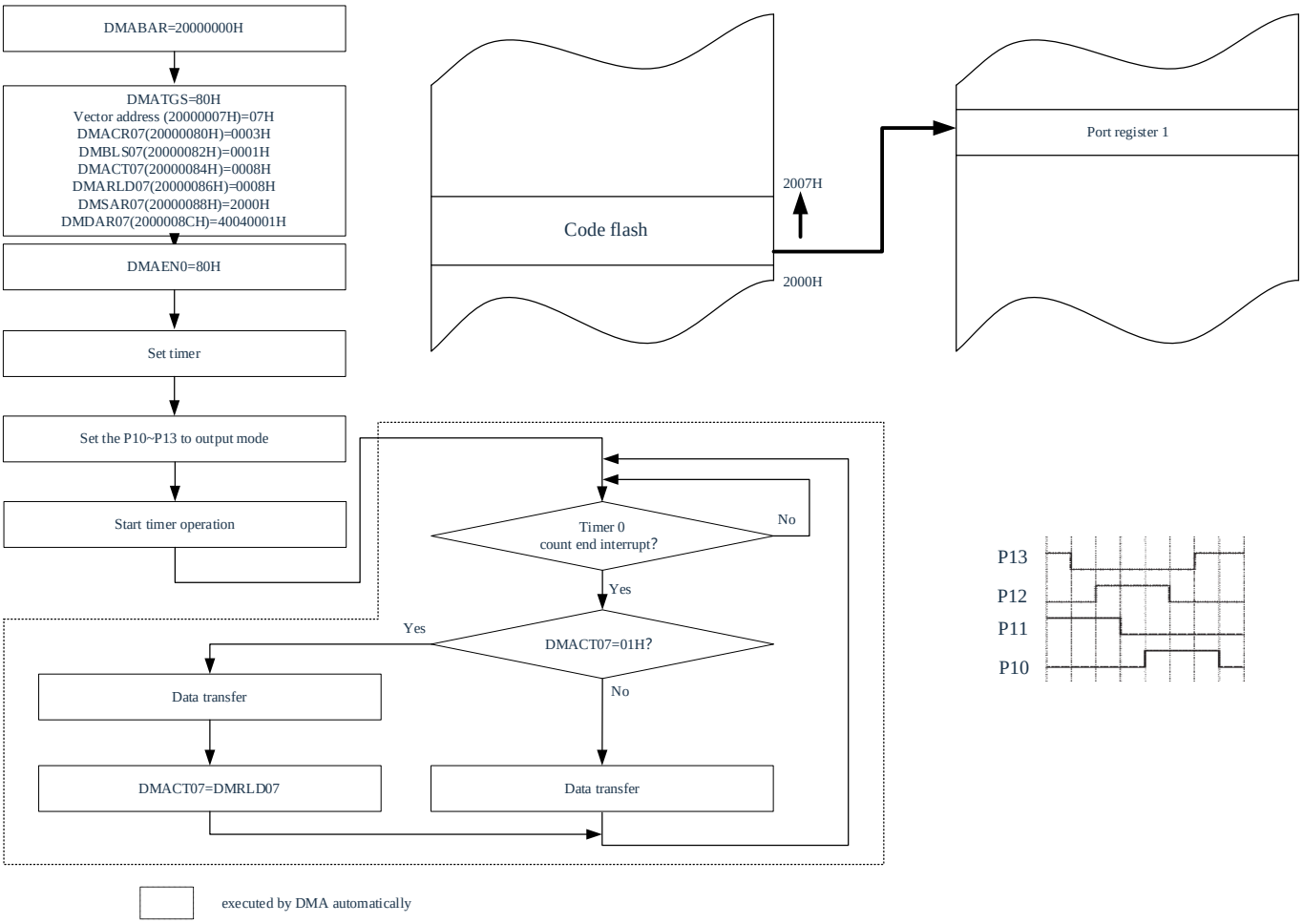
Note: In repeat mode, the data length of the repeat area must be set within 65535 bytes.

(1) Example of repeat mode usage: Use the stepper motor of the port to control the pulse output

Use the TIMER0 timer function to trigger DMA, and transfer the motor control pulse mode stored in the code flash to the universal port.

- The vector address is assigned at 20000007H, and the control data is assigned at 20000080H~2000008FH.
- Transfer 8 bytes from 02000H to 02007H of the code flash to port register 1 (40040001H).
- Disable repeat mode interrupt.

Figure 18-10 Example of repeat mode usage: use the stepper motor of the port to control the pulse output




To stop the output, the bit3 of the DMAEN2 must be cleared after stopping the timer operation.

## 18.5.4 Chain Transfer

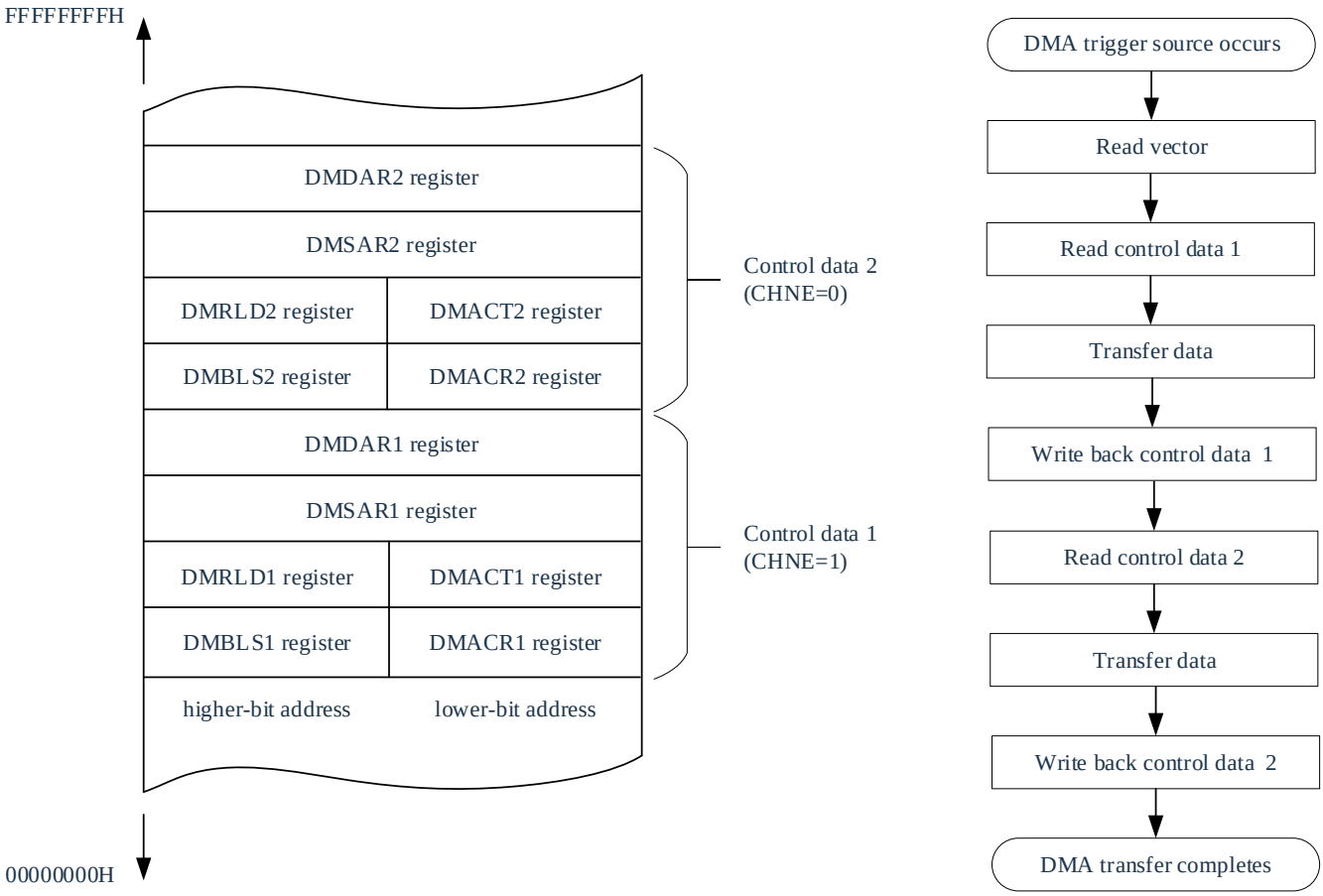
When the CHNE bit of the DMACRj register (j=0~7) is set to 1 (chain transfer enabled), multiple data transfers can be performed consecutively through a single start source.

Once DMA is started, control data is selected by reading data from the vector address corresponding to the start source. The read data is allocated in the DMA control data area. If the CHNE bit of the control data read is 1 (chain transfer enabled), the next allocated control data is read after the transfer completes, and the transfer continues. This operation is repeated until control data with the CHNE bit set to 0 (chain transfer disabled) ends the transfer.

When using multiple control data for chain transfer, the transfer count set in the first control data is valid, while the transfer counts in subsequent control data are ignored.

The flowchart of chain transfer is shown in Figure 18-11.

Figure 18-11 Flow chart for chain transfer




Note 1: The CHNE bit of the DMACR07 register must be set to 0 (chain transfer disabled).

Note 2: For data transfer after the second chain transfer, the DMAENi0 to DMAENi7 bits of the DMAENi (i=0) register are unchanged to 0 (DMA startup is disabled), and no interrupt request is generated.

(1) Example of chain transfer: Continuously taking A/D conversion result for UART0 transmission

Trigger DMA through the A/D conversion end interrupt and transfer the A/D conversion results to RAM for UART0 transmission.

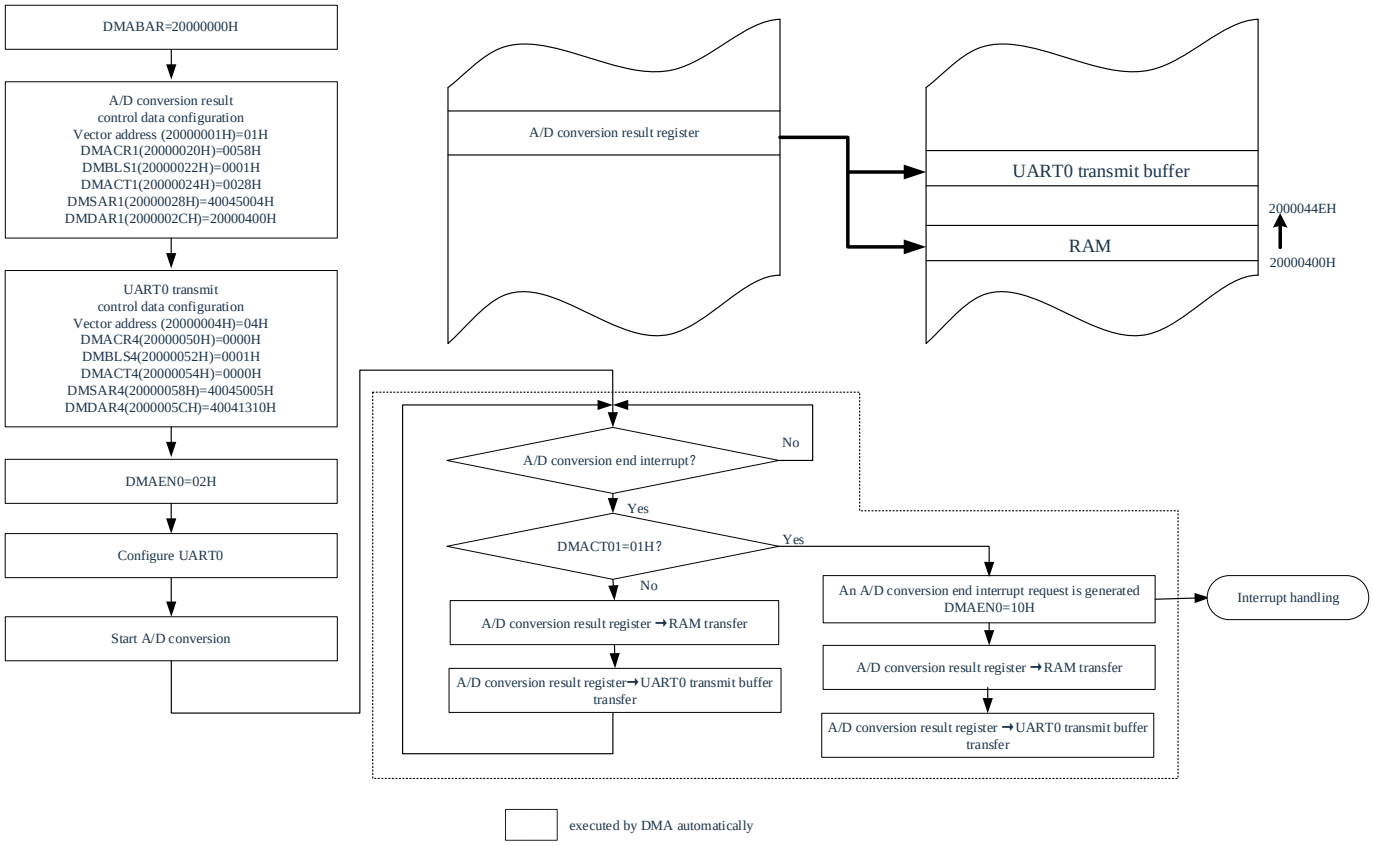
• The vector addresses are 20000001H and 20000004H, respectively.

• The control data for the A/D conversion results is allocated at 20000020H~2000002FH.

• The control data for UART0 transmission is allocated at 20000050H~2000005FH.

• Transfer the 2-byte data from the A/D conversion result registers (40068080H, 40068081H) to RAM at 20000400H~2000044FH, and transfer the high byte (40068081H) of the A/D conversion result register to the UART0 transmit buffer (40064004H).

Figure 18-12 Example of chain transfer: Continuously taking A/D conversion result for UART0 transmission



# 18.6Cautions When Using DMA

## 18.6.1 Settings of DMA Control Data and Vector Table

- The DMA base address register (DMABAR) must be modified when all DMA start sources are set to the disabled state.
- The DMA base address register (DMABAR) can only be written to once.
- The data in the DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, and DMDARj registers must be modified when the corresponding DMAENi (i=0) register's DMAENi0~DMAENi7 bits are 0 (DMA is disabled).
- The starting address of the DMA control data area in the vector table must be modified when the corresponding DMAENi (i=0) register's DMAENi0~DMAENi7 bits are 0 (DMA is disabled).

## 18.6.2 Assignment of DMA Control Data and Vector Table

The allocation of the DMA control data and vector table areas may vary depending on the product and usage conditions.

- The stack area, DMA control data area, and DMA vector table area must not overlap.

### 18.6.3 Number of Execution Clocks for DMA

The execution of the DMA at start-up and the number of clocks required are shown in Table 18-8.

Table 18-8 Execution and number of clocks required when DMA is started

| Read vector | Control data | | Read data | Write data |
|---|---|---|---|---|
| | Read | Write back | | |
| 1 | 4 | Note 1 | Note 2 | Note 2 |

Note 1: For the number of clocks required to write back control data, please refer to Table 18-9.

Note 2: For the number of clocks required to read and write data, please refer to Table 18-10.

Table 18-9 Number of clocks required to write back control data

| Settings for the DMACR register | | | | Address setting | | Control of register writeback | | | | Clock count |
|---|---|---|---|---|---|---|---|---|---|---|
| DAMOD | SAMOD | RPTSEL | MODE | Source | Target | DMACTj register | DMRLDj register | DMSARj register | DMDARj register | |
| 0 | 0 | X | 0 | Fixed | Fixed | Write back | Write back | Do not write back | Do not write back | 1 |
| 0 | 1 | X | 0 | Incremental | Fixed | Write back | Write back | Write back | Do not write back | 2 |
| 1 | 0 | X | 0 | Fixed | Incremental | Write back | Write back | Do not write back | Write back | 2 |
| 1 | 1 | X | 0 | Incremental | Incremental | Write back | Write back | Write back | Write back | 3 |
| 0 | X | 1 | 1 | Repeat area | Fixed | Write back | Write back | Write back | Do not write back | 2 |
| 1 | X | 1 | 1 | | Incremental | Write back | Write back | Write back | Write back | 3 |
| X | 0 | 0 | 1 | Fixed | Repeat area | Write back | Write back | Do not write back | Write back | 2 |
| X | 1 | 0 | 1 | Incremental | | Write back | Write back | Write back | Write back | 3 |

Note: j=0~7, X: “0” or “1”

Table 18-10 Number of clocks required to read and write data

| Execution status | RAM | Code flash | Data flash | Special function register (SFR) | Extended special function register (2ndSFR) | |
|---|---|---|---|---|---|---|
| | | | | | No wait | Wait |
| Read data | 1 | 2 | 4 | 1 | 1 | 1+ wait number |
| Write data | 1 | — | — | 1 | 1 | 1+ wait number |

### 18.6.4 Response Time for DMA

The DMA response time is shown in Table 18-11. DMA response time refers to the time from detecting the DMA start source to the beginning of the DMA transfer, excluding the number of execution clock cycles of DMA.

Table 18-11 Response time for DMA

| | Minimum time | Maximum time |
|---|---|---|
| Response time | 3 clocks | 23 clocks |

However, the DMA response may also be delayed in the following cases. The number of delayed clocks varies depending on the condition.

- Maximum response time to execute instructions from internal RAM: 20 clocks

Note: 1 clock: $1/f_{CLK}$ ($f_{CLK}$: CPU/peripheral hardware clock)

### 18.6.5 Start Source for DMA

- It is not possible to input the same start source during the period from the input of the DMA start source to the end of the DMA transfer.
- The DMA startup enable bit corresponding to the startup source cannot be manipulated at the location where the DMA startup source is generated.
- If the DMA startup source sends a contention, it judges the priority when the CPU accepts the DMA transfer and decides to start the startup source. Refer to 18.4.3 Vector Table for the priority of the startup source.

### 18.6.6 Operation in Standby Mode

| Status | DMA operation |
|---|---|
| Sleep mode | Can operate (disable operation in low-power RTC mode). |
| Deep sleep mode | Can accept DMA start source, and perform DMA transfer[Note 1] |

Note 1: In deep sleep mode, it is possible to perform a DMA transfer after detecting a DMA start source and return to deep sleep mode after the transfer is completed. However, since the code flash and data flash stop operating in the deep sleep mode, the flash cannot be set as the transfer source.

# Chapter 19 A/D Converter (ADC)

## 19.1 Overview

The chip includes a 12-bit, 27-channel fast successive approximation analog-to-digital converter (ADC).

## 19.2 Features

- Analog input voltage range: VSS to AVDD.
- Maximum sampling rate: 1.2 Msps.
- Up to 27 single-ended analog input channels.
- Supports two power modes: High-speed mode and low-current mode.
- In high-speed mode, the conversion time for one conversion is: 52*$T_{ADCK}$ (with a sample time set to 13.5*$T_{ADCK}$).
- Single conversion mode: Perform one A/D conversion on a specified channel.
- Continuous conversion mode: Perform A/D conversion on all selected channels.
- Supports external input signal triggering for ADC conversion.
- Supports interrupt generation after conversion completion.
- Internal ADC conversion result comparator.
- The conversion result of each channel is stored in its corresponding data register.

# 19.3Function Description

## 19.3.1 ADC Channels

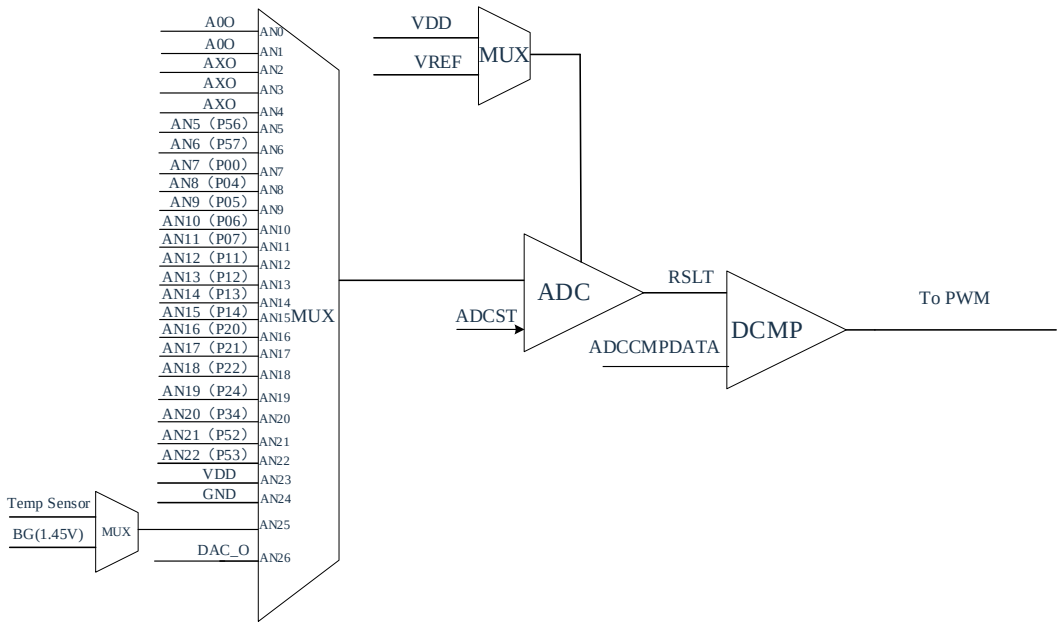
| ADC channel No. (hardware trigger supported) | ADC channel | ADC channel priority | Remark |
|---|---|---|---|
| 0 | AN0 (A0O) | Highest | PGA0 channel (see Chapter 20) |
| 1 | AN1 (A0O) | | PGA0 channel (see Chapter 20) |
| 2 | AN2 (AXO) | | PGA1 channel (see Chapter 20) |
| 3 | AN3 (AXO) | | PGA2 channel (see Chapter 20) |
| 4 | AN4 (AXO) | | PGA3 channel (see Chapter 20) |
| 5 | AN5 (P56) | | External channel 5 |
| 6 | AN6 (P57) | | External channel 6 |
| 7 | AN7 (P00) | | External channel 7 |
| 8 | AN8 (P04) | | External channel 8 |
| 9 | AN9 (P05) | | External channel 9 |
| 10 | AN10 (P06) | | External channel 10 |
| 11 | AN11 (P07) | | External channel 11 |
| 12 | AN12 (P11) | | External channel 12 |
| 13 | AN13 (P12) | | External channel 13 |
| 14 | AN14 (P13) | | External channel 14 |
| 15 | AN15 (P14) | | External channel 15 |
| 16 | AN16 (P20) | | External channel 16 |
| 17 | AN17 (P21) | | External channel 17 |
| 18 | AN18 (P22) | | External channel 18 |
| 19 | AN19 (P24) | | External channel 19 |
| 20 | AN20 (P34) | | External channel 20 |
| 21 | AN21 (P52) | | External channel 21 |
| 22 | AN22 (P53) | | External channel 22 |
| 23 | AN23 (VDD) | | |
| 24 | AN24 (GND) | | |
| 25 | AN25 (BG2AD) | Dedicated channel | BG1.45V/temperature sensor channel |
| 26 | AN26 | Dedicated channel | |

Note: Any combination of channels AN0-AN26 supports continuous mode conversion.

ADC internal channels:

| ADC internal channel number | ADC internal channel | Remark |
|---|---|---|
| 0~3 | - | Not selectable |
| 4 | IAN_4 (DAC_O) | DAC output channel (see Chapter 22) |

## 19.3.2 ADC Block Diagram




## 19.3.3 ADC Power Consumption Modes

The ADC operates in two modes: High-speed mode and low-current mode.

High-speed mode: In this mode, the conversion speed is faster.

Low-current mode: In this mode, the conversion speed is slightly slower, but the ADC's operating current significantly decreases. This mode can be used for applications where the conversion rate is not critical, helping to reduce the ADC's power consumption. The successive approximation time in this mode is 10 $T_{ADCK}$ longer than in high-speed mode.

## 19.3.4 ADC Conversion Modes

The ADC conversion mode is divided into two types: Single Conversion Mode and Continuous Conversion Mode.

Single Conversion Mode: After performing a conversion on the highest priority enabled channel, the operation ends and an interrupt flag is generated.

Continuous Conversion Mode: After converting all the enabled channels, the operation ends and an interrupt flag is generated. Channels that are not enabled are ignored and skipped.

ADCSWCHE = 0: Software channel enabling is disabled, and the ADC channel selection and enabling are automatically controlled by hardware.

ADCSWCHE = 1: Software channel enabling is enabled, and the selection and enabling of ADC channels are controlled by ADCSWCHS. After selecting a channel with ADCSWCHS, the channel is automatically enabled (ADCEN must be set to 1). In this condition, both Single Conversion Mode and Continuous Conversion Mode actually perform conversions on the channel selected by ADCSWCHS.

### 19.3.5 ADC Clock

The ADC clock source comes from the APB clock, and eight different division factors can be selected: 1/2/4/8/16/32/64/128, which can be configured through ADCCON.ADCDIV.

In High-Speed Mode: AD conversion time ($T_{ADCK}$ is the clock for the ADC when there is no prescaling applied.):

$1*T_{ADCK}$ $+2*T_{ADCK}$ (default switch stabilization time)$+13.5*T_{ADCK}$ (default sampling time)$+31.5*T_{ADCK}$ (successive approximation time)$+4*T_{ADCK}$

Continuous Conversion Mode: The total time for completing one ADC conversion is: ($T_{ADC}$ is the clock for the ADC when there is no prescaling applied):

$2*T_{ADCK}$ (default switch stabilization tim)$+13.5*T_{ADCK}$ (default sampling time)$+31.5*T_{ADCK}$ (successive approximation time)$+3*T_{ADCK}$

When ADCSWCHE = 1, the actual switch stabilization time refers to the time from selecting a channel until the conversion starts.

### 19.3.6 ADC Channel Selection and Interrupt Generation

| ADCSWCHE | ADCMS | Channel description | Result storage | Interrupt generation |
|---|---|---|---|---|
| 0 | 0 | Enable the highest priority channel that has been hardware-enabled | Store the result in the result register of the highest priority hardware-enabled channel | Interrupts are generated for the highest priority hardware-enabled channel |
| 0 | 1 | Sequentially enable hardware-enabled channels until all enabled channels complete conversion | After each channel conversion, store the result in the corresponding result register of that channel | After each conversion, an interrupt flag is generated for the corresponding channel |
| 1 | 0 | Enable the channel specified in ADCSWCHS | Store the result in the result register of the highest priority hardware-enabled channel | Interrupt flags are generated for the highest priority hardware-enabled channel |
| 1 | 1 | Enable the channel specified in ADCSWCHS. The number of hardware-enabled channels determines the number of consecutive conversions (the enabled channel is always set by ADCSWCHE) | Results are sequentially stored in the result registers of hardware-enabled channels | Interrupts are sequentially generated for the hardware-enabled channels |

Note 1: The hardware-enabled channels refer to the following: When triggered by the EPWM zero point, the channel enabled in the corresponding CHZIFTG register. When triggered by the EPWM period point, the channel enabled in the corresponding CHPIFTG register. When triggered by the EPWM compare points 0/1, the channels enabled in the corresponding CHPTG0/1 registers. When triggered by the EPWM output, the channel enabled in the corresponding CHEPWM register. For other software triggers, internal triggers, and external triggers, the channel enabled in the SCAN register.

Note 2: In hardware automatic channel selection mode, when two or more trigger signals arrive simultaneously, the hardware selects the channel based on the following priority of the channel selection registers:

CHZIFTG > CHPIFTG > CHPTG1 > CHPTG0 > CHEPWM > SCAN

Note 3: When using software to select the ADC conversion channel, do not switch channels via software when ADCCON.ADCST is 1.

### 19.3.7 ADC Software Start

Write 1 to the ADCCON2.ADCST bit to start the ADC conversion. The bit is automatically cleared by the hardware after the conversion is completed.

During the ADC conversion, any software or hardware trigger start signals will be ignored.

## 19.3.8 ADC Hardware Trigger Start

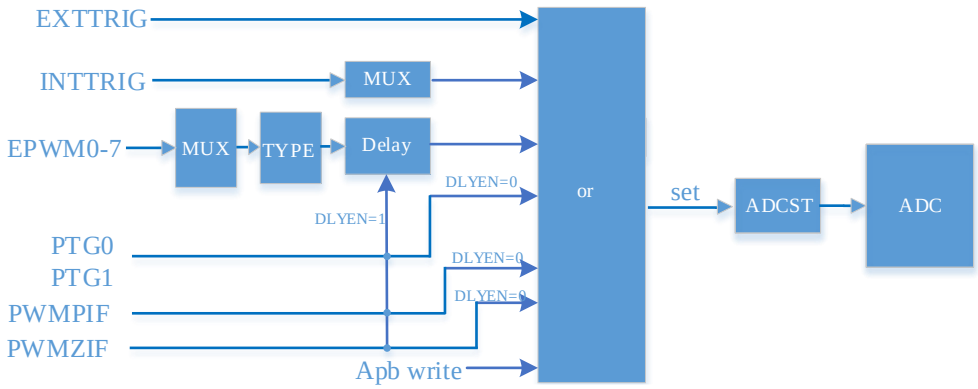
**Trigger sources:**

In addition to software-triggered conversion, the ADC can also be triggered by hardware signals. There are several types of hardware trigger sources:

1) External trigger
2) Internal trigger
3) EPWM output channel trigger
4) EPWM period point trigger
5) EPWM zero point trigger
6) EPWM count comparator 0 trigger
7) EPWM count comparator 1 trigger

Different types of trigger sources can be simultaneously active. A single type of trigger source may contain different trigger signals. For example, the EPWM output channel trigger allows you to choose one of the trigger signals from EPWM0 to EPWM7.

Figure 19-1: ADC hardware trigger start




**Internal triggers:**

Internal triggers include ADC, ACMP0, ACMP1, TIMER0/1 triggers.

ADC: ADC conversion completion

ACMP0: Event output of ACMP0

ACMP1: Event output of ACMP1

Timer0: Enabled interrupt of Timer0 (TMR0MIS)

Timer1: Enabled interrupt of Timer1 (TMR1MIS)

**EPWM output channel trigger:**

The EPWM output channel trigger can choose to trigger the ADC on the rising edge, falling edge, zero point, or period point. If the EPWM trigger signal is detected, the ADC conversion can be initiated after a certain delay (there is no separate delay trigger enable bit; if the delay data is not zero, it will automatically be delayed). If the EPWM output channel is remapped, the EPWM trigger signal will be the signal from before the remapping, which is the IPGn signal.

The EPWM output channel trigger only supports hardware channel selection and allows separate setting of the ADC conversion channel. After the EPWM output channel trigger signal is generated, the ADC will convert according to the independently configured conversion channel. The ADC conversion channel triggered by the EPWM output channel is set in the ADCCHEPWM register. Once the conversion is complete, it will revert to the channel configuration in the

ADCSCAN register.

**EPWM zero and period trigger:**

The EPWM zero and period point trigger sources can be selected from any EPWM channel's zero point and period point. The triggering method is the same as the EPWM channel trigger. You can also choose to start the ADC conversion after a certain delay. It is also possible to individually set whether the zero point and period point trigger ADC conversion with a delay, using the ADCHWTG.ADCZDELAYEN and ADCHWTG.ADCPEWMPDLYEN registers.

The zero point and period point trigger only support hardware-selected channels, and the ADC conversion channel can be set independently. After the zero point or period point trigger signal is generated, the ADC will convert based on the channel that is independently configured. The ADC conversion channel for the EPWM zero point trigger is set in the ADCCHZIFTG register, while for the EPWM period point trigger, the channel is set in the ADCCHPIFTG register. After the conversion is complete, it will revert to the channel setting in the ADCSCAN register.

**EPWM count comparator trigger:**

The EPWM counter comparators 0/1 trigger can be set to trigger ADC conversion at any time within the EPWM cycle. The method is similar to the EPWM channel trigger. You can also choose to start the ADC conversion after a certain delay, and this is controlled using ADCHWTG.ADCPTG0DLYEN and ADCHWTG.ADCPTG1DLYEN registers to set whether the trigger from comparators 0/1 involves a delay.

The EPWM counter comparator 0/1 trigger only supports hardware-selected channels, and the ADC conversion channel can be set independently. After the trigger signal is generated, the ADC will convert based on the independently configured channel. The conversion channel for the EPWM comparator 0 trigger is set in the ADCCHPTG0 register, while for comparator 1, it is set in the ADCCHPTG1 register. After the conversion is complete, it will revert to the channel configuration in the ADCSCAN register.

Note: In the delayed trigger mode, if a higher priority trigger signal arrives during the delay of a previous trigger signal, the system will automatically start conversion on the channel configured by the higher priority trigger signal after the delay is complete (priority order is described in section 19.3.6). For example, if both EPWM period point and EPWM zero point delayed triggers are enabled, and the EPWM period point trigger signal arrives, the internal delay counter starts counting. If an EPWM zero point trigger signal arrives during this delay, the system will automatically enable the channel configured in the ADCCHZIFTG register once the delay timer value reaches the set delay.

**EPWM trigger delay:**

The ADCEPWMTGDLY register determines the delay time for the EPWM trigger to start the ADC. This delay time is the time from the arrival of the valid trigger signal to the rising edge of the ADCST signal.

$$(ADCEPWMTGDLY[9:0]+3)*T_{PCLK}$$

The EPWM trigger delay ranges are as follows:

| | pclk 64MHz (delay range) | pclk 72MHz (delay range) |
|---|---|---|
| When delay is enabled and delay data is not 0 | 0.062us~16.03us | 0.056us~14.25us |
| When delay is disabled or delay data is 0 | 0.047us | 0.042us |

When ADCEPWMTGDLY = 0 or the corresponding delay trigger is disabled, the ADC conversion is started after a delay of 3 $T_{PCLK}$ clock cycles for the EPWM comparator 0, EPWM comparator 1, EPWM output, period point, and zero point.

**EPWM trigger for ADC conversion setup**

EPWM triggering ADC conversions may have specific timing requirements in some applications. To meet this need, the ADC internally supports different EPWM trigger conditions, allowing independent conversion channels to be set. For example:

EPWM output channel trigger can select conversion for AN0, AN1, or AN2 channels.

EPWM comparator 0 trigger can select conversion for the AN18 channel.

EPWM comparator 1 trigger can select conversion for the AN19 channel.

EPWM period point trigger can select conversion for the AN20 channel.

EPWM zero point trigger can select conversion for the AN21 channel.

Software trigger or other external triggers can select conversion for AN5, AN6, AN7, or AN8 channels.

When there are no EPWM trigger conditions, the default conversion channels are AN5 to AN8.

When the EPWM output channel trigger is activated, only AN0 to AN2 channels are selected for ADC conversion. After conversion is complete, the channels automatically switch to AN5 to AN8 for enabling.

When the EPWM comparator 0 trigger is activated, only the AN18 channel is selected for ADC conversion. After conversion is complete, the channels automatically switch to AN5 to AN8 for enabling.

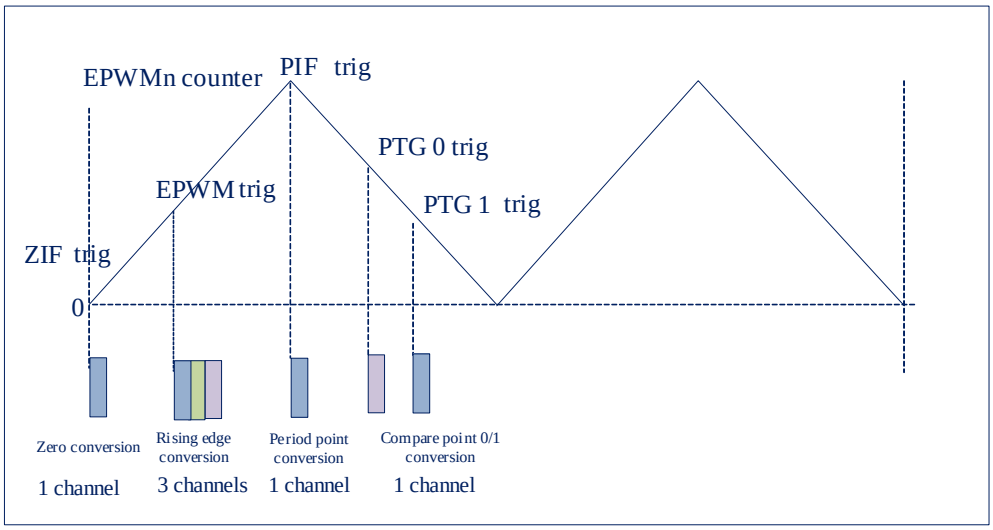
When the EPWM comparator 1 trigger is activated, only the AN19 channel is selected for ADC conversion. After conversion is complete, the channels automatically switch to AN5 to AN8 for enabling.

When the EPWM period point trigger is activated, only the AN20 channel is selected for ADC conversion. After conversion is complete, the channels automatically switch to AN5 to AN8 for enabling.

When the EPWM zero point trigger is activated, only the AN21 channel is selected for ADC conversion. After conversion is complete, the channels automatically switch to AN5 to AN8 for enabling.

It is important to note that during an ongoing ADC conversion, any other trigger signals will be ignored.

Figure 19-2: ADC configuration for EPWM trigger start




Note 1: The channels for ADC conversion enabled by triggers from the zero point, period, EPWM output rising edge, and EPWM output falling edge are determined by ADCCHPEM.

Note 2: The channels for ADC conversion enabled by the comparator 0 trigger are determined by ADCCHPTG0.

Note 3: The channels for ADC conversion enabled by the comparator 1 trigger are determined by ADCCHPTG1.

Note 4: The channels for ADC conversion enabled by the period point trigger are determined by ADCCHPIFTG.

Note 5: The channels for ADC conversion enabled by the zero point trigger are determined by ADCCHZIFTG.

Note 6: The channels for ADC conversion enabled by other methods are determined by ADCSCAN or ADCSWCHS.

# 19.4Register Mapping

(ADC base address = 0x4006_8000) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CON(P1B) | 0x000 | R/W | ADC Control Register | 0x000D0000 |
| CON2(P1B) | 0x004 | R/W | ADC Control Register 2 | 0x003F0000 |
| HWTG(P1B) | 0x008 | R/W | ADC Hardware Trigger Control Register | 0x00000000 |
| PWMTGDLY(P1B) | 0x00C | R/W | ADC EPWM Trigger Delay Data Register | 0x00000000 |
| SCAN(P1B) | 0x010 | R/W | ADC Scan Register | 0x00000000 |
| CMP0(P1B) | 0x014 | R/W | ADC Comparator 0 Control Register | 0x00000000 |
| CMP1(P1B) | 0x018 | R/W | ADC Comparator 1 Control Register | 0x00000000 |
| IMSC(P1B) | 0x01C | R/W | ADC Interrupt Enable Register | 0x00000000 |
| RIS | 0x020 | RO | ADC Interrupt Source Status Register | 0x00000000 |
| MIS | 0x024 | RO | ADC Enabled Interrupt Status Register | 0x00000000 |
| ICLR | 0x028 | WO | ADC Interrupt Clear Register | 0x00000000 |
| LOCK | 0x02C | R/W | ADC Write Enable Control Register | 0x00000000 |
| CHEPWM(P1B) | 0x030 | R/W | ADC EPWM Output Trigger Conversion Channel Register | 0x00000000 |
| CHPTG0(P1B) | 0x034 | R/W | ADC EPWM Comparator 0 Trigger Conversion Channel Register | 0x00000000 |
| CHPTG1(P1B) | 0x038 | R/W | ADC EPWM Comparator 1 Trigger Conversion Channel Register | 0x00000000 |
| CHPIFTG(P1B) | 0x03C | R/W | ADC EPWM Period Point Trigger Conversion Channel Register | 0x00000000 |
| CHZIFTG(P1B) | 0x040 | R/W | ADC EPWM Zero Point Trigger Conversion Channel Register | 0x00000000 |
| TEST(P1B) | 0x048 | R/W | ADC Test Register | 0x00000000 |
| RESULT | 0x050 | RO | ADC Conversion Result Register, ADC Conversion Completion Refresh | 0x00000000 |
| DATA0 | 0x080 | RO | ADC Channel 0 Conversion Result Register | 0x00000000 |
| DATA1 | 0x084 | RO | ADC Channel 1 Conversion Result Register | 0x00000000 |
| DATA2 | 0x088 | RO | ADC Channel 2 Conversion Result Register | 0x00000000 |
| DATA3 | 0x08C | RO | ADC Channel 3 Conversion Result Register | 0x00000000 |
| DATA4 | 0x090 | RO | ADC Channel 4 Conversion Result Register | 0x00000000 |
| DATA5 | 0x094 | RO | ADC Channel 5 Conversion Result Register | 0x00000000 |
| DATA6 | 0x098 | RO | ADC Channel 6 Conversion Result Register | 0x00000000 |
| DATA7 | 0x09C | RO | ADC Channel 7 Conversion Result Register | 0x00000000 |
| DATA8 | 0x0A0 | RO | ADC Channel 8 Conversion Result Register | 0x00000000 |
| DATA9 | 0x0A4 | RO | ADC Channel 9 Conversion Result Register | 0x00000000 |
| DATA10 | 0x0A8 | RO | ADC Channel 10 Conversion Result Register | 0x00000000 |
| DATA11 | 0x0AC | RO | ADC Channel 11 Conversion Result Register | 0x00000000 |
| DATA12 | 0x0B0 | RO | ADC Channel 12 Conversion Result Register | 0x00000000 |
| DATA13 | 0x0B4 | RO | ADC Channel 13 Conversion Result Register | 0x00000000 |
| DATA14 | 0x0B8 | RO | ADC Channel 14 Conversion Result Register | 0x00000000 |
| DATA15 | 0x0BC | RO | ADC Channel 15 Conversion Result Register | 0x00000000 |
| DATA16 | 0x0C0 | RO | ADC Channel 16 Conversion Result Register | 0x00000000 |
| DATA17 | 0x0C4 | RO | ADC Channel 17 Conversion Result Register | 0x00000000 |
| DATA18 | 0x0C8 | RO | ADC Channel 18 Conversion Result Register | 0x00000000 |
| DATA19 | 0x0CC | RO | ADC Channel 19 Conversion Result Register | 0x00000000 |
| DATA20 | 0x0D0 | RO | ADC External Channel 20 Conversion Result Register | 0x00000000 |
| DATA21 | 0x0D4 | RO | ADC External Channel 21 Conversion Result Register | 0x00000000 |

| | | | | |
|---|---|---|---|---|
| DATA22 | 0x0D8 | RO | ADC External Channel 22 Conversion Result Register | 0x00000000 |
| DATA23 | 0x0DC | RO | ADC External Channel 23 Conversion Result Register | 0x00000000 |
| DATA24 | 0x0E0 | RO | ADC External Channel 24 Conversion Result Register | 0x00000000 |
| DATA25 | 0x0E4 | RO | ADC External Channel 25 Conversion Result Register | 0x00000000 |
| DATA26 | 0x0E8 | RO | ADC External Channel 26 Conversion Result Register | 0x00000000 |

Note 1: The registers marked with (P1B) are protected registers.

Note 2: (P1B): When LOCK==55H, the marked registers are allowed to be written; LOCK== other values, writing is prohibited.

# 19.5Register Description

## 19.5.1 ADC Control Register (CON)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | ADCRST | ADC module reset control bit<br>0: ---<br>1: ADC module reset | 0 |
| 30:26 | - | Reserved, set to 0. | 0x0 |
| 25:24 | ADMODE10 | ADC power mode select bit<br>00: High-speed mode<br>01: Reserved, disable selection.<br>10: Reserved, disable selection.<br>11: Low current mode | 0x0 |
| 23:16 | ADCNSMP | ADC internal sample time select bit<br>00000000: to 00000100: Disable selection<br>00000101: 5.5 ADC clockcycles<br>00000110: 6.5 ADC clockcycles<br>00000111: 7.5 ADC clockcycles<br>00001000: 8.5 ADC clockcycles<br>00001001: 9.5 ADC clockcycles<br>00001010: 10.5 ADC clockcycles<br>00001011: 11.5 ADC clockcycles<br>00001100: 12.5 ADC clockcycles<br>00001101: 13.5 ADC clockcycles<br>00001000: …<br>11111110: 254.5 ADC clockcycles<br>11111111: 255.5 ADC clockcycles | 0x0D |
| 15:14 | - | Reserved, set to 0. | 0x0 |
| 13 | ADCSWCHE | ADC channel software enable bit<br>0: Automatically turned on by hardware<br>1: The channel activation is determined by ADCSWCHS. | 0 |
| 12 | ADCNDISEN | ADC charge/discharge function select bit<br>0: Discharge<br>1: Charge | 0 |
| 11:8 | ADCNDISTS | ADC charge/discharge time select bit<br>0000: No charging or discharging<br>0001: Disable selection<br>0010: 2 ADC clockcycles<br>0011: 3 ADC clockcycles<br>…<br>1111: 15 ADC clockcycles | 0x0 |
| 7:6 | ADCVS | ADC positive reference select bit<br>00: Select VDD<br>01: Select VREF<br>10: Reserved<br>11: Disable selection | 0x0 |

| | | | |
|---|---|---|---|
| 5 | - | Reserved, set to 0. | 0 |
| 4 | ADCEN | ADC enable control bit<br>0: Disable<br>1: Enable | 0 |
| 3 | ADCMS | ADC conversion mode select bit<br>0: Single conversion<br>1: Continuous conversion<br>(Convert all enabled ADC channels at one time, the order is channel 0 to channel 26, the hardware automatically ignores the channels that are not enabled, and no conversion operation will be generated.) | 0 |
| 2:0 | ADCDIV | ADC clock prescaler select bit<br>$F_{ADC} = PCLK/2^{ADCDIV}$ | 0x0 |

## 19.5.2 ADC Control Register 2 (CON2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:24 | - | Reserved | - |
| 23 | PGA_HOLD_EN | PGA hold time enable control bit<br>0: Disable<br>1: Enable | 0 |
| 22 | - | Reserved | |
| 21:16 | PGA_HOLD_TIME | PGA hold time selection<br>The time from enabling the PGA sample and hold to maintaining the sample and hold after the ADC is triggered by hardware or software:<br>(PGA_HOLD_TIME+1)*pclk | 0x3f |
| 15:13 | ADCICHES | ADC internal channel (AN26) select bit<br>100: Select internal channel 4<br>Other: Disable selection | 0x0 |
| 12 | ADCSF4 | ADC conversion status flag bit 4 (read-only)<br>0: -<br>1: Single conversion completed | 0 |
| 11 | ADCSF3 | ADC conversion status flag bit 3 (read-only)<br>0: -<br>1: An ADC clock cycle before conversion is completed | 0 |
| 10 | ADCSF2 | ADC conversion status flag bit 2 (read-only)<br>0: -<br>1: Two ADC clock cycles before conversion is completed | 0 |
| 9 | ADCSF1 | ADC conversion status flag bit 1 (read-only)<br>0: -<br>1: During conversion | 0 |
| 8 | ADCSF0 | ADC conversion status flag bit 0 (read-only)<br>0: -<br>1: During sampling | 0 |
| 7 | ADCST | ADC conversion starts (hardware automatically clears after conversion)<br>0: Conversion finished or ADC in idle mode (Write 0 is invalid)<br>1: Start conversion (ADCEN must be 1) | 0 |
| 6 | ADCSMPWAIT | ADC sample time extension control bit<br>0: -<br>1: Forced hold sampling state during sampling | 0 |
| 5 | BG2ADSEL | TS channel (AN25) select bit<br>0: TS temperature sensor<br>1: BG reference voltage 1.45V | 0 |
| 4:0 | ADCSWCHS | ADC channel software selection bit<br>(Valid only when ADCSWCHE=1)<br>00000: Select channel 0<br>00001: Select channel 1<br>…<br>11001: Select channel 25<br>11010: Select channel 26<br>Other: Reserved | 0x0 |

### 19.5.3 ADC Hardware Trigger Control Register (HWTG)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | ADCEPWMZEN | ADC EPWM zero point output trigger enable bit<br>0: Disable<br>1: Enable | 0 |
| 30:28 | ADCEPWMZSS | ADC EPWM zero point output trigger source channel select bit<br>000: Trigger source EPWM0<br>001: Trigger source EPWM1<br>010: Trigger source EPWM2<br>011: Trigger source EPWM3<br>100: Trigger source EPWM4<br>101: Trigger source EPWM5<br>110: Trigger source EPWM6<br>111: Trigger source EPWM7 | 0x0 |
| 27 | ADCEPWMPEN | ADC EPWM period point output trigger enable bit<br>0: Disable<br>1: Enable | 0 |
| 26:24 | ADCEPWMPSS | ADC EPWM period point output trigger source channel select bit<br>000: Trigger source EPWM0<br>001: Trigger source EPWM1<br>010: Trigger source EPWM2<br>011: Trigger source EPWM3<br>100: Trigger source EPWM4<br>101: Trigger source EPWM5<br>110: Trigger source EPWM6<br>111: Trigger source EPWM7 | 0x0 |
| 23:22 | - | Reserved | - |
| 21 | ADCEPWMZDLYEN | ADC EPWM zero point delay trigger enable bit<br>0: Enable<br>1: Disable (without delay) | 0 |
| 20 | ADCEPWMPDLYEN | ADC EPWM period point delay trigger enable bit<br>0: Enable<br>1: Disable (without delay) | 0 |
| 19:18 | - | Reserved | - |
| 17 | ADCEXTEN | ADC external trigger enable bit<br>0: Disable<br>1: Enable | 0 |
| 16 | ADCEXTES | ADC external trigger edge select bit<br>0: Falling edge<br>1: Rising edge | 0 |
| 15 | ADCINTTGEN | ADC internal function trigger enable bit<br>0: Disable<br>1: Enable | 0 |
| 14:12 | ADCINTTGSS | ADC internal function trigger source channel select bit<br>000: Reserved<br>001: ADC conversion end signal<br>010: ACMP0 event | 0x0 |

| | | | |
|---|---|---|---|
| | | 011: ACMP1 event<br>100: Timer0 interrupt signal<br>101: Timer1 interrupt signal | |
| 11 | ADCPTG1DLYEN | ADC EPWM count comparator 1 delay trigger enable bit<br>0: Enable<br>1: Disable (without delay) | 0 |
| 10 | ADCPTG0DLYEN | ADC EPWM count comparator 0 delay trigger enable bit<br>0: Enable<br>1: Disable (without delay) | 0 |
| 9 | ADCPTG1EN | ADC EPWM count comparator 1 trigger enable bit<br>0: Disable<br>1: Enable | 0 |
| 8 | ADCPTG0EN | ADC EPWM count comparator 0 trigger enable bit<br>0: Disable<br>1: Enable | 0 |
| 7 | ADCEPWMTEN | ADC EPWM output trigger enable bit<br>0: Disable<br>1: Enable | 0 |
| 6:4 | ADCEPWMTSS | ADC EPWM output trigger source channel select bit<br>000: Trigger source EPWM0<br>001: Trigger source EPWM1<br>010: Trigger source EPWM2<br>011: Trigger source EPWM3<br>100: Trigger source EPWM4<br>101: Trigger source EPWM5<br>110: Trigger source EPWM6<br>111: Trigger source EPWM7 | 0x0 |
| 3:2 | - | Reserved | - |
| 1:0 | ADCPEWMTPS | ADC EPWMn trigger mode select bit (n=0-7)<br>00: Rising edge of the EPWMn waveform<br>01: EPWMn period point (IPGn)<br>10: Falling edge of the EPWMn waveform<br>11: Zero point of EPWMn (IPGn) | 0x0 |

### 19.5.4 ADC EPWM Trigger Delay Register (EPWMTGDLY)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:10 | - | Reserved | - |
| 9:0 | ADCEPWMTGDLY | ADC EPWM trigger delay data<br>(including output channel triggering and EPWM zero point trigger, EPWM period trigger, and EPWM comparator 0/1 triggering) delay trigger ADC delay data (see section 19.3.8 EPWM trigger delay for details) | 0x0 |

### 19.5.5 ADC Scan Register (SCAN)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:27 | - | Reserved | - |
| 26:0 | ADCEn | ADC channel n enable bit (n=26-0)<br>0: Disable<br>1: Enable | 0x0 |

### 19.5.6 ADC EPWM Output Trigger Conversion Channel Enable Register (CHEPWM)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:27 | - | Reserved | - |
| 26:0 | ADCCHEPWMn | ADC EPWM output trigger conversion channel enable bit (n=26-0)<br>0: Disable<br>1: Enable | 0x0 |

### 19.5.7 ADC EPWM Period Point Trigger Conversion Channel Enable Register (CHPIFTG)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:27 | - | Reserved | - |
| 26:0 | ADCCHPIFTGn | ADC EPWM period point trigger conversion channel enable bit (n=26-0)<br>0: Disable<br>1: Enable | 0x0 |

### 19.5.8 ADC EPWM Zero Point Trigger Conversion Channel Enable Bit (CHZIFTG)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:27 | - | Reserved | - |
| 26:0 | ADCCHZIFTGn | ADC EPWM zero point trigger conversion channel enable bit (n=26-0)<br>0: Disable<br>1: Enable | 0x0 |

### 19.5.9 ADC EPWM Comparator 0 Trigger Conversion Channel Enable Register (CHPTG0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:27 | - | Reserved | - |
| 26:0 | ADCCHPTG0n | ADC EPWM comparator 0 trigger conversion channel enable bit (n=26-0)<br>0: Disable<br>1: Enable | 0x0 |

### 19.5.10 ADC EPWM Comparator 1 Trigger Conversion Channel Enable Register (CHPTG1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:27 | - | Reserved | - |
| 26:0 | ADCCHPTG1n | ADC EPWM comparator 1 trigger conversion channel enable bit (n=26-0)<br>0: Disable<br>1: Enable | 0x0 |

### 19.5.11 ADC Test Register (TEST)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:24 | ADCSWT | The delay time for switching the ADC analog switch:<br>(i.e., the time from the analog switch turning on to the start of sampling)<br>(ADCSWT+2)ADC clock cycles<br>Note: It is recommended to extend this time when performing ADC conversions on weak signals. | 0x0 |
| 23:0 | - | Reserved, set to 0. | 0x0 |

### 19.5.12 ADC Conversion Result Register (RESULT)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:12 | - | Reserved | - |
| 11:0 | RESULT | ADC conversion result, and refresh this value after each conversion is completed. | 0x0 |

### 19.5.13 ADC Channel Conversion Result Register (DATAx) x=0~26

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:12 | - | Reserved | - |
| 11:0 | RESULT | ADC conversion result | 0x0 |

## 19.5.14 ADC Compare Control Register 0 (CMPx) x=0~1

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | ADCCMPxEN | ADC comparator x enable bit<br>0: Disable<br>1: Enable | 0 |
| 30 | ADCCMPxO | ADC comparator x result bit (read-only)<br>(This bit is automatically updated after the selected channel conversion is completed)<br>0: Does not meet the comparison condition<br>1: Meets the comparison condition | 0 |
| 29 | - | Reserved | - |
| 28 | ADCCMPxCOND | ADC comparator x compare condition select bit<br>0: ADC result < preset value<br>1: ADC result >= preset value | 0 |
| 27:24 | ADCCMPxMCNT | ADC comparator x match count preset value<br>When the analog-to-digital conversion result of the specified channel matches the comparison condition, the internal counter will be incremented by 1, and when the internal counter equals to the value of ADCCMPxMCNT+1, the internal counter will be cleared to zero automatically. The internal counter will also be cleared to zero if the matching condition is not met during the accumulation process, i.e., this function has a filtering function.<br>The ADC compare event is generated at the same time as the match, which can be used as a signal to trigger the brake operation of the EPWM.<br>Caution:<br>The ADC Comparator 0 compare event will set the interrupt flag ADCCMP0IF to 1. | 0x0 |
| 23:21 | - | Reserved | - |
| 20:16 | ADCCMPxCHS | ADC comparator x compare channel select bit<br>00000: Channel 0<br>…… ……<br>11001: Channel 25<br>11010: Channel 26<br>Other: Reserved | 0x0 |
| 15:12 | - | Reserved | - |
| 11:0 | ADCCMPxDATA | ADC comparator x data preset value (12-bit) | 0x0 |

## 19.5.15 ADC Interrupt Enable Register (IMSC)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | ADCIMSC31 | ADC comparator 0 interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 30:27 | - | Reserved | - |
| 26:0 | ADCIMSCn | ADC channel n interrupt enable bit (n=26-0)<br>0: Disable<br>1: Enable | 0x0 |

## 19.5.16 ADC Interrupt Source Status Register (RIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | ADCRIS31 | ADC comparator 0 interrupt source status<br>0: No interrupt is generated by interrupt sources<br>1: An interrupt is generated by interrupt sources | 0 |
| 30:27 | - | Reserved | - |
| 26:0 | ADCRISn | ADC channel n interrupt source status (n=26-0)<br>0: No interrupt is generated by interrupt sources<br>1: An interrupt is generated by interrupt sources | 0x0 |

## 19.5.17 ADC Enabled interrupt Status Register (MIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | ADCMIS31 | ADC comparator 0 interrupt status<br>0: No interrupt is generated<br>1: Enable and generate an interrupt | 0 |
| 30:27 | - | Reserved | - |
| 26:0 | ADCMISn | ADC channel n interrupt status (n=26-0)<br>0: No interrupt is generated<br>1: Enable and generate an interrupt | 0x0 |

## 19.5.18 ADC Interrupt Clear Register (ICLR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | ADCICLR31 | Write 1 to clear the ADC comparator 0 interrupt status<br>Writing 0 has no effect | 0 |
| 30:27 | - | Reserved | - |
| 26:0 | ADCICLRn | Write 1 to clear the ADC channel n interrupt status (n=26-0)<br>Writing 0 has no effect | 0x0 |

## 19.5.19 ADC Write Enable Control Register (LOCK)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | LOCK | When LOCK=0x55, enable the operation of ADC related registers. (For details, please refer to the description of ADC register mapping.)<br>When LOCK=other values, disable the operation of ADC related registers. | 0x0 |

# Chapter 20 Programmable Gain Amplifier (PGA0/1/2/3)

## 20.1Overview

The chip includes four programmable gain amplifiers. With a few external components, it can achieve basic signal amplification and signal processing functions.

## 20.2Features

**PGA0 (Programmable Gain Amplifier 0)**

- ◆ Adjustable gain: 1X/2X/2.5X/5X/7.5X/10X/15X.
- ◆ Positive input: PGA0P; negative input: PGA0N
- ◆ Reference voltage: BGR (0.8V) or VREF/2
- ◆ Multiple output options for PGA0:
  - (1) Direct output to ADC channels 0, 1
  - (2) Direct output to comparator
  - (3) Direct output to PAD (A0O)
  - (4) Output to PAD (A0O) through a 10K resistor

**PGA1 (Programmable Gain Amplifier 1)**

- ◆ Adjustable gain: 1X/2X/2.5X/5X/7.5X/10X/15X.
- ◆ Positive input: PGA1P; negative input: PGA1N
- ◆ Reference voltage: VREF/2
- ◆ Output options for PGA1:
  - (1) Direct output to ADC channel 2
  - (2) Direct output to comparator

**PGA2 (Programmable Gain Amplifier 2)**

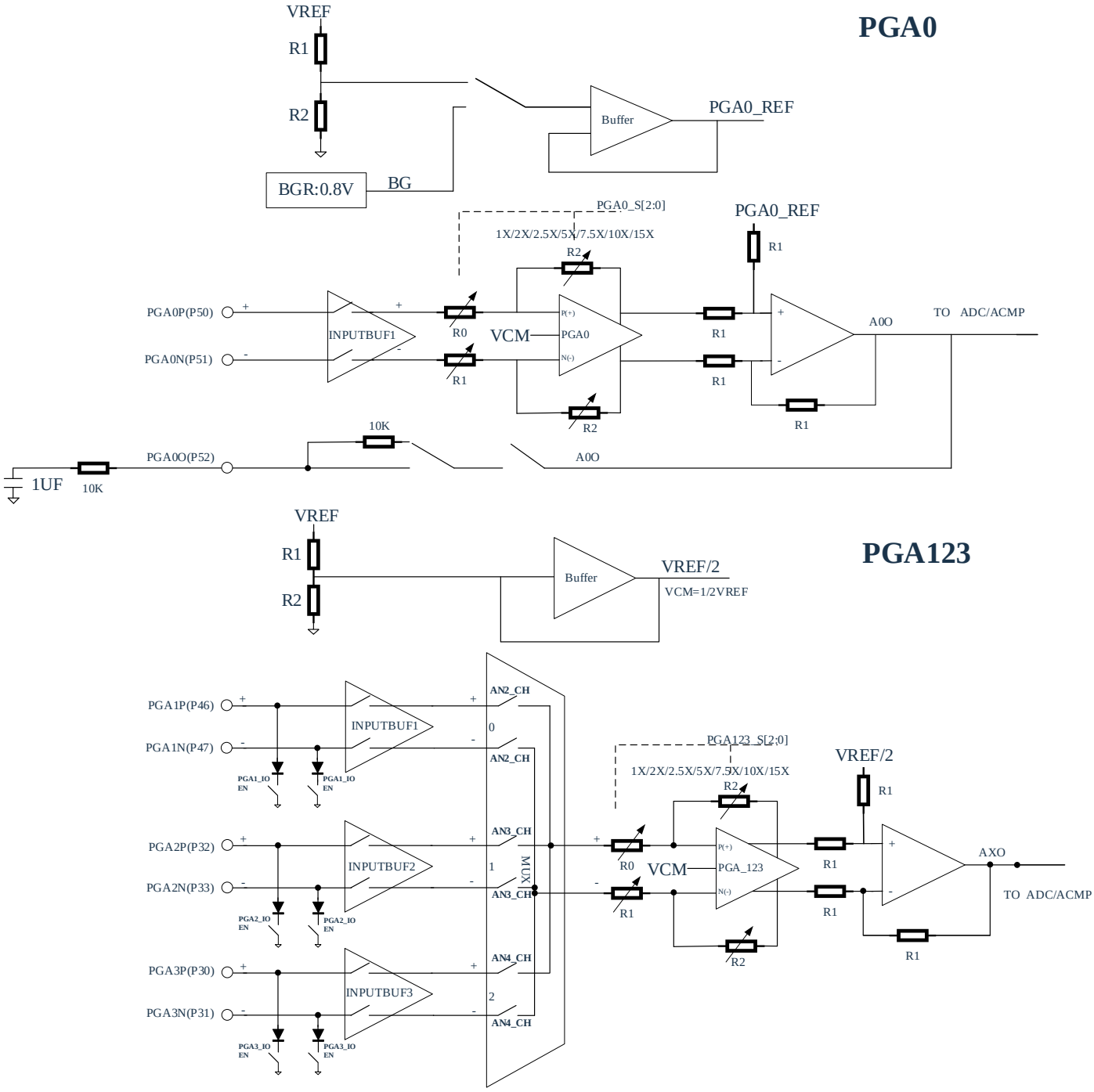
- ◆ Adjustable gain: 1X/2X/2.5X/5X/7.5X/10X/15X
- ◆ Positive input: PGA2P; negative input: PGA2N
- ◆ Reference voltage: VREF/2
- ◆ Output options for PGA2:
  - (1) Direct output to ADC channel 3
  - (2) Direct output to comparator

**PGA3 (Programmable Gain Amplifier 3)**

- ◆ Adjustable gain: 1X/2X/2.5X/5X/7.5X/10X/15X
- ◆ Positive input: PGA3P; negative input: PGA3N
- ◆ Reference voltage: VREF/2
- ◆ Output options for PGA2:
  - (1) Direct output to ADC channel 4
  - (2) Direct output to comparator

# 20.3Block Diagram of Structure

Figure 20-1 PGA structure diagram



## 20.4Register Mapping

(PGA base address = 0x4006_8300) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| PGA0CON0 | 0x000 | R/W | PGA0 Control Register 0 | 0x00000000 |
| PGANCON0 | 0x004 | R/W | PGA1/2/3 Control Register 0 | 0x00000000 |
| PGANCON1 | 0x008 | R/W | PGA1/2/3 Control Register 1 | 0x00000000 |
| PGANSW | 0x00C | R/W | PGA1/2/3 Input Channel Selection Register | 0x00000000 |
| PGALOCK | 0x010 | R/W | PGA Access Register Enable | 0x00000000 |

# 20.5Register Description

## 20.5.1 PGA0 Control Register 0 (PGA0CON0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:19 | - | Reserved | - |
| 18 | PGA0_S_DRV | PGA0 INPUTBUF drive current selection signal<br>0: Use 20μA current with external 1K/20K resistors.<br>1: Use 40μA current with an external 1K resistor to enhance driving capability. | 0 |
| 17 | PGA0_OTEN | PGA0 output to PAD channel enable<br>0: Disable<br>1: Enable | 0 |
| 16 | PGA0_OT_SELR | PGA0 output to PAD serial resistor selection<br>0: No internal series resistor.<br>1: With an internal 10K series resistor. | 0 |
| 15:13 | - | Reserved | - |
| 12 | PGA0_EN | PGA0 enable bit<br>0: Disable<br>1: Enable | 0 |
| 11 | PGA0_SWBTP | PGA0 sample and hold enable<br>0: PGA0 sample and hold (shoot-throught) is disabled<br>1: PGA0 sample and hold is enabled | 0 |
| 10 | PGA0_HOLD | PGA0 sample and hold state enable<br>0: Sample state<br>1: Sample and hold state | 0 |
| 9 | - | Set to 0 | |
| 8 | PGA0S_VREF | PGA0 reference voltage selection bit<br>0: VREF/2<br>1: BG (0.8V) | 0 |
| 7 | - | Reserved | - |
| 6:4 | PGA0_BW | PGA0 front-end filter bandwidth selection<br>000: 290ns<br>111: 30ns<br>Step Size: 37ns | 0x0 |
| 3 | - | Reserved | - |
| 2:0 | PGA0_S | PGA0 gain selection<br>000: 1X<br>001: 2X<br>010: 2.5X<br>011: 5X<br>100: 7.5X<br>101: 10X<br>110: 15X<br>111: 15X | 0x0 |

## 20.5.2 PGA1/2/3 Control Register 0 (PGANCON0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Reserved | - |
| 15 | PGA2_EN | PGA2 enable bit<br>0: Disable<br>1: Enable | 0 |
| 14:11 | - | Reserved | - |
| 10:8 | PGA2_BW | PGA2 front-end filter bandwidth selection<br>000: 290ns<br>111: 30ns<br>Step Size :37ns | 0x0 |
| 7 | PGA1_EN | PGA1 enable bit<br>0: Disable<br>1: Enable | 0 |
| 6:3 | - | Reserved | - |
| 2:0 | PGA1_BW | PGA1 front-end filter bandwidth selection<br>000: 290ns<br>111: 30ns<br>Step Size :37ns | 0x0 |

## 20.5.3 PGA1/2/3 Control Register 1 (PGANCON1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:14 | - | Reserved | - |
| 13 | PGA123_S_DRV | PGA123 INPUTBUF drive current selection signal<br>0: Use 20μA current with external 1K/20K resistors.<br>1: Use 40μA current with an external 1K resistor to enhance driving capability. | 0 |
| 12 | - | Set to 0. | 0 |
| 11 | - | Reserved | - |
| 10:8 | PGA123_S | PGA123 gain selection<br>000: 1X<br>001: 2X<br>010: 2.5X<br>011: 5X<br>100: 7.5X<br>101: 10X<br>110: 15X<br>111: 15X | 0x0 |
| 7 | PGA3_EN | PGA3 enable bit<br>0: Disable<br>1: Enable | 0 |
| 6:3 | - | Reserved | - |
| 2:0 | PGA3_BW | PGA3 front-end filter bandwidth selection<br>000: 290ns<br>111: 30ns<br>Step Size :37ns | 0x0 |

## 20.5.4 PGA1/2/3 Input Channel Switch Register 1 (PGANSW)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | PGA123SW_SEL | PGA1/2/3 input channel switch selection bit<br>0x55aa: Controlled by PGA123_SW<br>Other: Controlled by the ADC channel switch, independent of the PGA123_SW status.<br>PGA1_SW corresponds to ADC channel 2, PGA2_SW corresponds to ADC channel 3, and PGA3_SW corresponds to ADC channel 4. | 0x0 |
| 15:11 | - | Reserved | - |
| 10 | PGA3_IOEN | PGA3 IO clamping diode enable<br>0: Disable<br>1: Enable | 0 |
| 9 | PGA2_IOEN | PGA2 IO clamping diode enable<br>0: Disable<br>1: Enable | 0 |
| 8 | PGA1_IOEN | PGA1 IO clamping diode enable<br>0: Disable<br>1: Enable | 0 |
| 7:2 | - | Reserved | - |
| 1:0 | PGA123_SW | PGA1/2/3 channel switch selection<br>00: PGA1/2/3 channels are disabled.<br>01: PGA1 channel switch enabled, PGA2/3 channels disabled.<br>10: PGA2 channel switch enabled, PGA1/3 channels disabled.<br>11: PGA3 channel switch enabled, PGA1/2 channels disabled. | 0x0 |

## 20.5.5 PGA Access Register Enable (PGALOCK)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | PGA_LOCK | PGA0/1/2/3 register access enable bit<br>0x55: Access PGA0/1/2/3 related registers<br>Other: Disable access | 0x0 |

# Chapter 21 Analog Comparator (ACMP0/1)

## 21.1Overview

The chip contains two internal analog comparators, which can be configured for different application scenarios. When the positive input voltage is greater than the negative input voltage, the comparator outputs a logic 1; otherwise, it outputs a logic 0. The comparator supports the following functions:

(1) The output polarity can be changed through the polarity selection bit.

(2) It has a opening window function, and the opening window polarity can be set.

(3) The output can be filtered, with selectable filter time.

(4) The comparator output can be used as the capture input for CCP1's CAP3.

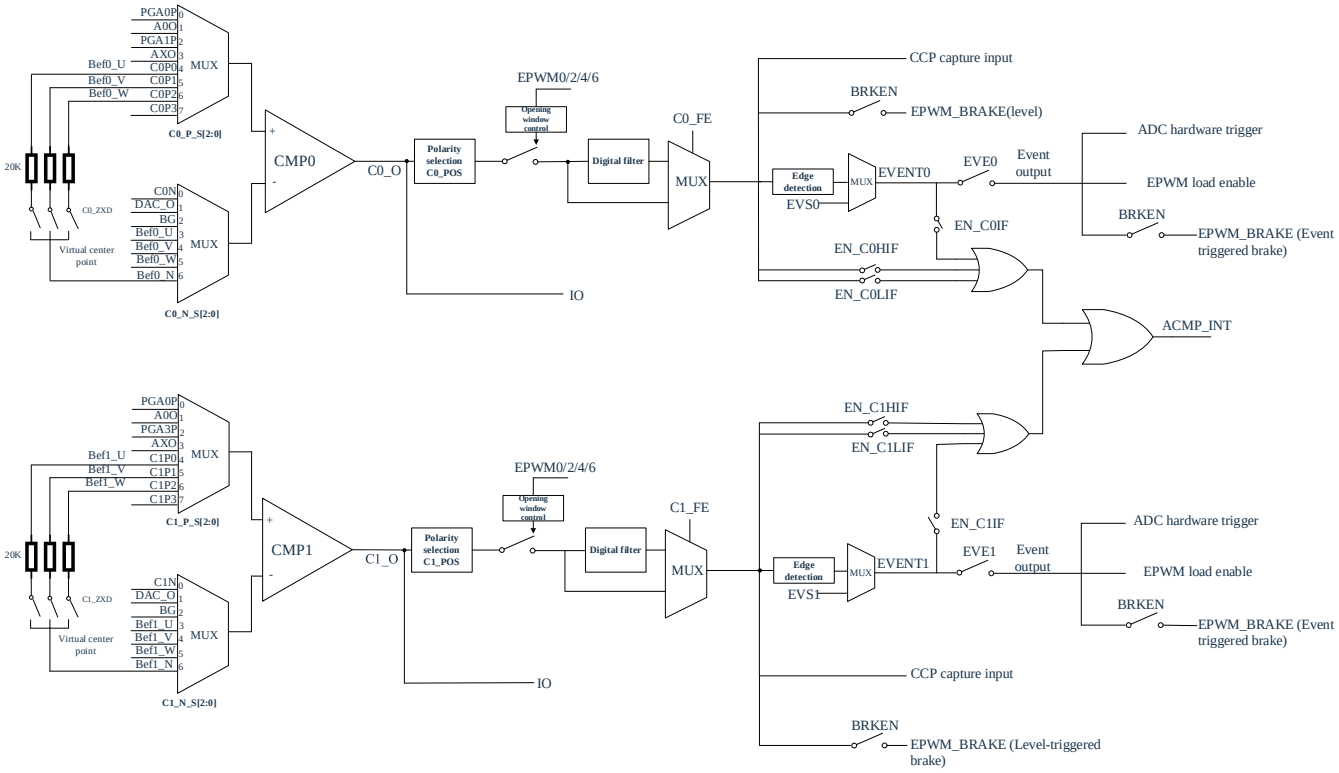
(5) Comparator-generated event outputs can serve as a hardware trigger for the ADC.

(6) The comparator output and event-generated output can also be used as brake trigger signals for enhanced PWM.

(7) When the comparator output value changes, each comparator can be configured to generate an interrupt.

## 21.2Block Diagram of Structure

Figure 21-1: Comparator block diagram



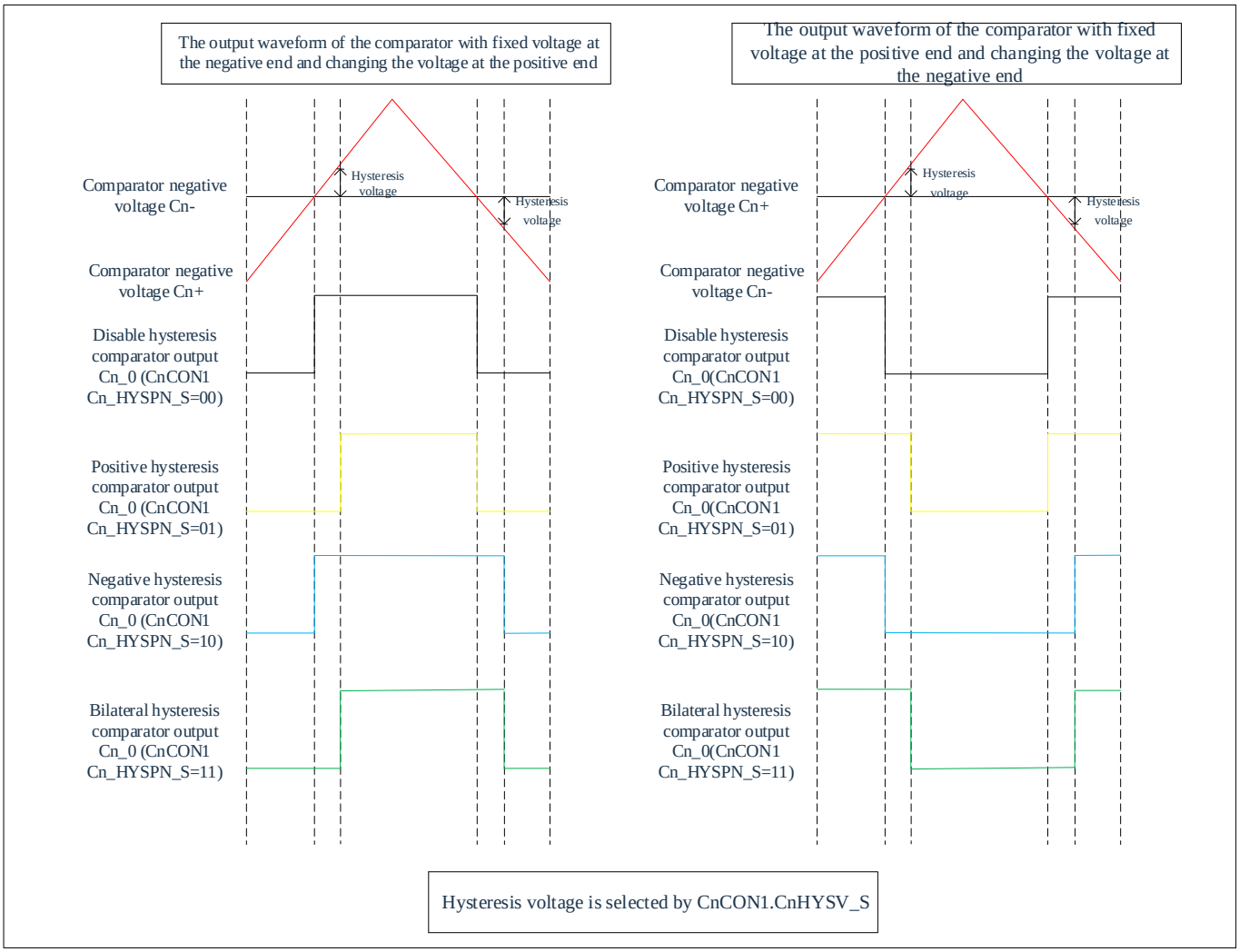
## 21.3Features

- Analog input voltage range: (0 ~ VDD) V.
- Supports single-side/double-side hysteresis function.
- Supports hysteresis voltage selection (10mV/20mV/60mV - typical values).
- Each comparator’s positive input can be selected from multiple channels.
- Each comparator’s negative input can be selected from port inputs or internal reference voltage.
- Output filter time is selectable: 0~512*Tsys.
- Comparator output polarity is selectable.
- Both comparator output and event-generated output can serve as brake trigger signals for enhanced PWM.
- Event generation and comparator output changes can both trigger interrupts.
- Supports opening window function, and the opening window polarity can be set.
- Both ACMP0 and ACMP1 have four selectable EPWM opening window channels.

# 21.4Function Description

## 21.4.1 Hysteresis Function

Figure 21-2: Block diagram of comparator hysteresis function

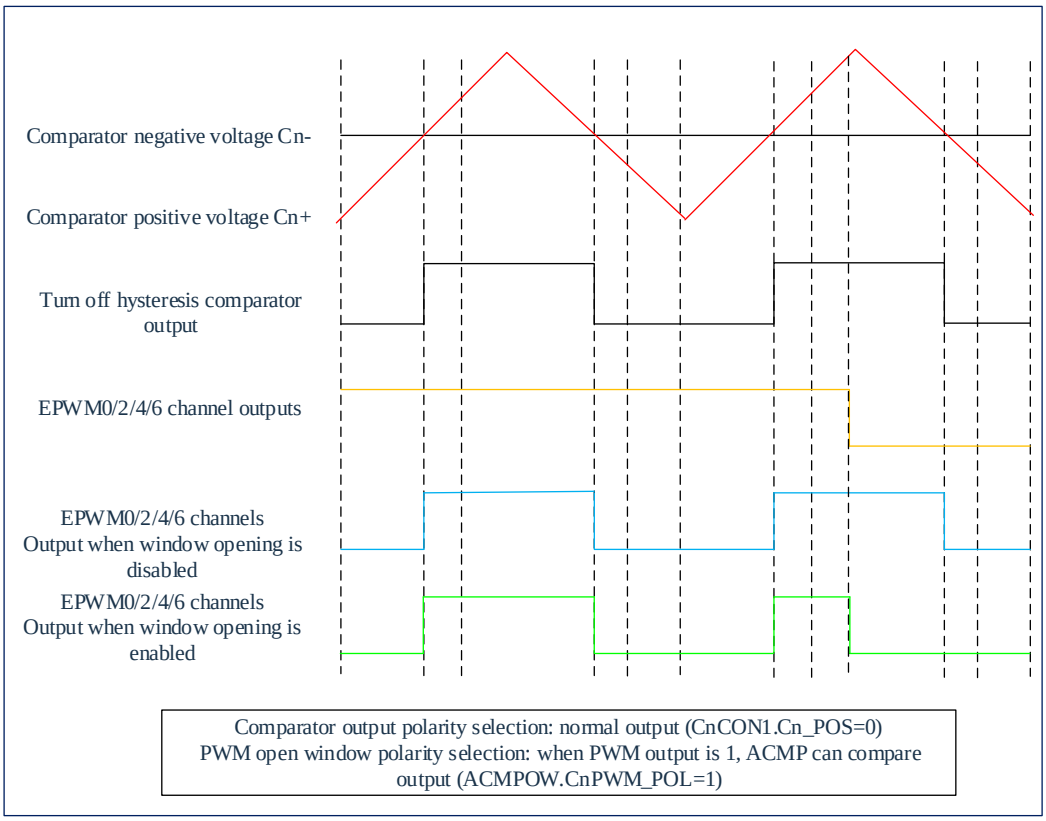



Note: n=0, 1.

## 21.4.2 Opening Window Control

The block diagram of the opening window control function is shown in the figure below.

Figure 21-3: Block diagram of opening window control function




Note: n=0, 1.

When the opening window function for the EPWM0/2/4/6 channels of the analog comparators 0/1 is disabled, the output of the opening window control is the output after the comparator polarity selection.

When the PWM opening window polarity selection bit (ACMPOW.CnPWM_POL) is set to 1, the opening window function for the EPWM0/2/4/6 channels is enabled. The output of the opening window control is the comparator polarity selected output when the EPWM0/2/4/6 channel output is 1, and the opening window control output is 0 when the EPWM0/2/4/6 channel output is 0.

When the PWM opening window polarity selection bit (ACMPOW.CnPWM_POL) is set to 0, the opening window function for the EPWM0/2/4/6 channels is enabled. The output of the opening window control is the comparator polarity selected output when the EPWM0/2/4/6 channel output is 0, and the opening window control output is 0 when the EPWM0/2/4/6 channel output is 1.

### 21.4.3 Filter Function

The main clock for the comparator module is PCLK, and the digital filter also uses this clock as the filtering clock. When the filter enable is set, it will filter the output signal after the opening window control, meaning that only if the signal stabilization time exceeds the set filter time will the signal pass through the filter. The filtered signal will change only after the stabilization time is sufficient. If the input signal's stabilization time is shorter than the set filter time and the signal changes, the filtered signal will maintain its original value.

When the filter enable is set, the following situations can occur:

(1) Filter input signal width < filter set time: No output signal.

(2) Filter input signal width = filter set time: Outputs a pulse signal of one clock cycle (cannot trigger EPWM braking).

(3) Filter input signal width > filter set time: Normal filter output.

Note: The filter time is set by the CnCON1.Cn_FS[3:0] settings, where n = 0, 1.

## 21.5Register Mapping

(ACMP base address = 0x4006_8200) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| C0CON0$_{(P1B)}$ | 0x00 | R/W | Analog Comparator 0 Control Register 0 | 0x00 |
| C0CON1$_{(P1B)}$ | 0x04 | R/W | Analog Comparator 0 Control Register 1 | 0x00 |
| C1CON0$_{(P1B)}$ | 0x08 | R/W | Analog Comparator 1 Control Register 0 | 0x00 |
| C1CON1$_{(P1B)}$ | 0x0C | R/W | Analog Comparator 1 Control Register 1 | 0x00 |
| CVECON$_{(P1B)}$ | 0x10 | R/W | Analog Comparator Event Control Register | 0x00 |
| IMSC$_{(P1B)}$ | 0x14 | R/W | Analog Comparator Interrupt Enable Register | 0x00 |
| RIS | 0x18 | RO | Analog Comparator Interrupt Source Status Register | 0x00 |
| MIS | 0x1C | RO | Analog Comparator Enabled Interrupt Status Register | 0x00 |
| ICLR | 0x20 | WO | Analog Comparator Interrupt Clear Register | 0x00 |
| ACMPOW$_{(P1B)}$ | 0x24 | R/W | Analog Comparator Opening Window Control Register | 0x00 |
| LOCK | 0x28 | R/W | Analog Comparator Write Enable Register | 0x00 |

Note 1: The registers marked with (P1B) are protected registers.

Note 2: When (P1B): LOCK=55H, the marked registers are allowed to be written; for other values, writing is prohibited.

# 21.6Register Description

## 21.6.1 Analog Comparator 0 Control Register 0 (C0CON0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:17 | - | Reserved | - |
| 16 | C0_ZXD | Analog comparator 0 center point select enable<br>0: Disable<br>1: Enable | 0 |
| 15 | C0_EN | Analog comparator 0 enable bit<br>0: Disable<br>1: Enable | 0 |
| 14 | C0_OEN | Analog comparator 0 output enable bit<br>0: Disable<br>1: Enable | 0 |
| 13:9 | - | Reserved | - |
| 8 | C0_OUT | Analog comparator 0 result bit (read-only) | 0 |
| 7 | - | Reserved | - |
| 6:4 | C0_P_S | Analog comparator 0 positive channel selection<br>000: PGA0P<br>001: A0O<br>010: PGA1P<br>011: AXO<br>100: C0P0<br>101: C0P1<br>110: C0P2<br>111: C0P3 | 0x0 |
| 3 | - | Reserved | - |
| 2:0 | C0_N_S | Analog comparator 0 negative channel selection<br>000: C0N<br>001: DAC_O<br>010: BG (0.8V)<br>011: BEF0_U (C0P0)<br>100: BEF0_V (C0P1)<br>101: BEF0_W (C0P2)<br>110: BEF0_N<br>111: Disable selection | 0x0 |

## 21.6.2 Analog Comparator 0 Control Register 1 (C0CON1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:14 | - | Reserved | - |
| 13:12 | C0_HYSPN_S | Analog comparator 0 hysteresis mode selection<br>00: No hysteresis<br>01: Positive hysteresis<br>10: Negative hysteresis<br>11: Positive and negative hysteresis | 0x0 |
| 11:10 | C0_HYSV_S | Analog comparator 0 hysteresis voltage selection<br>00: No hysteresis<br>01: 10mV<br>10: 20mV<br>11: 60mV | 0x0 |
| 9 | C0_POS | Analog comparator 0 output polarity select bit<br>0: Normal output<br>1: Inverted output | 0 |
| 8 | C0_FE | Analog comparator 0 output filter enable bit<br>0: Disable<br>1: Enable | 0 |
| 7:4 | - | Reserved | - |
| 3:0 | C0_FS | Analog comparator 0 output filter time select bit<br>0000: (0~1)*Tpclk<br>0001: (1~2)*Tpclk<br>0010: (2~3)*Tpclk<br>0011: (4~5)*Tpclk<br>0100: (8~9)*Tpclk<br>0101: (16~17)*Tpclk<br>0110: (32~33)*Tpclk<br>0111: (64~65)*Tpclk<br>1000: (128~129)*Tpclk<br>1001: (256~257)*Tpclk<br>1010: (512~513)*Tpclk<br>Other: (0~1)*Tpclk | 0x0 |

### 21.6.3 Analog Comparator 1 Control Register 0 (C1CON0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:17 | - | Reserved | - |
| 16 | C1_ZXD | Analog comparator 1 center point selection enable<br>0: Disable<br>1: Enable | 0 |
| 15 | C1_EN | Analog comparator 1 enable bit<br>0: Disable<br>1: Enable | 0 |
| 14 | C1_OEN | Analog comparator 1 output enable bit<br>0: Disable<br>1: Enable | 0 |
| 13:9 | - | Reserved | - |
| 8 | C1_OUT | Analog comparator 1 result bit (read-only) | |
| 7 | - | Reserved | - |
| 6:4 | C1_P_S | Analog comparator 1 positive channel selection<br>000: PGA0P<br>001: A0O<br>010: PGA3P<br>011: AXO<br>100: C1P0<br>101: C1P1<br>110: C1P2<br>111: C1P3 | 0x0 |
| 3 | - | Reserved | - |
| 2:0 | C1_N_S | Analog comparator 1 negative channel selection<br>000: C1N<br>001: DAC_O<br>010: BG(0.8V)<br>011: BEF1_U (C1P0)<br>100: BEF1_V (C1P1)<br>101: BEF1_W (C1P2)<br>110: BEF1_N<br>Other: Disable | 0x0 |

## 21.6.4 Analog Comparator 1 Control Register 1 (C1CON1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:14 | - | Reserved | - |
| 13:12 | C1HYSPN_S | Analog comparator 1 hysteresis mode selection<br>00: No hysteresis<br>01: Positive hysteresis<br>10: Negative hysteresis<br>11: Positive and negative hysteresis | 0x0 |
| 11:10 | C1_HYSV_S | Analog comparator 1 hysteresis voltage selection<br>00: No hysteresis<br>01: 10mV<br>10: 20mV<br>11: 60mV | 0x0 |
| 9 | C1_POS | Analog comparator 1 output polarity select bit<br>0: Normal output<br>1: Inverted output | 0 |
| 8 | C1_FE | Analog comparator 1 output filter enable bit<br>0: Disable<br>1: Enable | 0 |
| 7:4 | - | Reserved | - |
| 3:0 | C1_FS | Analog comparator 1 output filter time select bit<br>0000: (0~1)*Tpclk<br>0001: (1~2)*Tpclk<br>0010: (2~3)*Tpclk<br>0011: (4~5)*Tpclk<br>0100: (8~9)*Tpclk<br>0101: (16~17)*Tpclk<br>0110: (32~33)*Tpclk<br>0111: (64~65)*Tpclk<br>1000: (128~129)*Tpclk<br>1001: (256~257)*Tpclk<br>1010: (512~513)*Tpclk<br>Other: (0~1)*Tpclk | 0x0 |

### 21.6.5 Analog Comparator Event Control Register (CEVCON)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:6 | - | Reserved | - |
| 5 | EVE1 | Analog comparator 1 event output enable bit<br>(does not affect interrupt generation)<br>0: Disable<br>1: Enable | 0 |
| 4 | EVE0 | Analog comparator 0 event output enable bit<br>(does not affect interrupt generation)<br>0: Disable<br>1: Enable | 0 |
| 3:2 | EVS1 | Analog comparator 1 event generation condition select bit<br>00: Comparator 1 output jumps from 0->1<br>01: Comparator 1 output jumps from 1->0<br>10: Comparator 1 output jumps from 0->1 or from 1->0<br>11: Reserved | 0x0 |
| 1:0 | EVS0 | Analog comparator 0 event generation condition select bit<br>00: Comparator 0 output jumps from 0->1<br>01: Comparator 0 output jumps from 1->0<br>10: Comparator 0 output jumps from 0->1 or from 1->0<br>11: Reserved | 0x0 |

### 21.6.6 Analog Comparator Interrupt Enable Register (IMSC)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:7 | - | Reserved | - |
| 6 | EN_C1LIF | Analog comparator 1 output low level interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 5 | EN_C1HIF | Analog comparator 1 output high level interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 4 | EN_C1IF | Analog comparator 1 event interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 3 | - | Reserved | - |
| 2 | EN_C0LIF | Analog comparator 0 output low level interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 1 | EN_C0HIF | Analog comparator 0 output high level interrupt enable bit<br>0: Disable<br>1: Enable | 0 |
| 0 | EN_C0IF | Analog comparator 0 event interrupt enable bit<br>0: Disable<br>1: Enable | 0 |

### 21.6.7 Analog Comparator Interrupt Source Status Register (RIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:7 | - | Reserved | - |
| 6 | RIS_C1LIF | Analog comparator 1 low level interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated (low level generation) | 0 |
| 5 | RIS_C1HIF | Analog comparator 1 high level interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated (high level generation) | 0 |
| 4 | RIS_C1IF | Analog comparator 1 event interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated (event generation) | 0 |
| 3 | - | Reserved | - |
| 2 | RIS_C0LIF | Analog comparator 0 low level interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated (low level generation) | 0 |
| 1 | RIS_C0HIF | Analog comparator 0 high level interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated (high level generation) | 0 |
| 0 | RIS_C0IF | Analog comparator 0 event interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated (event generation) | 0 |

### 21.6.8 Analog Comparator Enabled Interrupt Source Status Register (MIS)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:7 | - | Reserved | - |
| 6 | MIS_C1LIF | Analog comparator 1 low level output interrupt flag bit, enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0 |
| 5 | MIS_C1HIF | Analog comparator 1 high level output interrupt flag bit, enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0 |
| 4 | MIS_C1IF | Analog comparator 1 event interrupt flag bit, enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0 |
| 3 | - | Reserved | - |
| 2 | MIS_C0LIF | Analog comparator 0 low level output interrupt flag bit, enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0 |
| 1 | MIS_C0HIF | Analog comparator 0 high level output interrupt flag bit, enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0 |
| 0 | MIS_C0IF | Analog comparator 0 event interrupt flag bit, enabled interrupt status bit<br>0: No interrupt is generated<br>1: An interrupt is generated | 0 |

### 21.6.9 Analog Comparator Interrupt Clear Control Register (ICLR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:7 | - | Reserved | - |
| 6 | ICLR_C1LIF | Analog comparator 1 low level interrupt clear control bit<br>0: No effect<br>1: Clear the RIS_C1LIF flag bit | 0 |
| 5 | ICLR_C1HIF | Analog comparator 1 high level interrupt clear control bit<br>0: No effect<br>1: Clear the RIS_C1HIF flag bit | 0 |
| 4 | ICLR_C1IF | Analog comparator 1 event interrupt clear control bit<br>0: No effect<br>1: Clear the RIS_C1IF flag bit | 0 |
| 3 | - | Reserved | - |
| 2 | ICLR_C0LIF | Analog comparator 0 low level interrupt clear control bit<br>0: No effect<br>1: Clear the RIS_C0LIF flag bit | 0 |
| 1 | ICLR_C0HIF | Analog comparator 0 high level interrupt clear control bit<br>0: No effect<br>1: Clear the RIS_C0HIF flag bit | 0 |
| 0 | ICLR_C0IF | Analog comparator 0 event interrupt clear control bit<br>0: No effect<br>1: Clear the RIS_C0IF flag bit | 0 |

## 21.6.10 Analog Comparator Opening Window Control Register (ACMPOW)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:16 | - | Reserved | 0 |
| 15 | C1PWM_POL | Analog comparator 1 PWM opening window polarity selection<br>0: When the PWM output is 0, the ACMP1 comparator output is available.<br>1: When the PWM output is 1, the ACMP1 comparator output is available. | 0 |
| 14:12 | - | Reserved | - |
| 11 | C1PWM6_WEN | Analog comparator 1 EPWM6 channel opening window enable<br>0: Opening window is disabled<br>1: Opening window is enabled | 0 |
| 10 | C1PWM4_WEN | Analog comparator 1 EPWM4 channel opening window enable<br>0: Opening window is disabled<br>1: Opening window is enabled | 0 |
| 9 | C1PWM2_WEN | Analog comparator 1 EPWM2 channel opening window enable<br>0: Opening window is disabled<br>1: Opening window is enabled | 0 |
| 8 | C1PWM0_WEN | Analog comparator 1 EPWM0 channel opening window enable<br>0: Opening window is disabled<br>1: Opening window is enabled | 0 |
| 7 | C0PWM_POL | Analog comparator 0 PWM opening window polarity selection<br>0: When the PWM output is 0, the ACMP0 comparator output is available.<br>1: When the PWM output is 1, the ACMP0 comparator output is available. | 0 |
| 6:4 | - | Reserved | - |
| 3 | C0PWM6_WEN | Analog comparator 0 EPWM6 channel opening window enable<br>0: Opening window is disabled<br>1: Opening window is enabled | 0 |
| 2 | C0PWM4_WEN | Analog comparator 0 EPWM4 channel opening window enable<br>0: Opening window is disabled<br>1: Opening window is enabled | 0 |
| 1 | C0PWM2_WEN | Analog comparator 0 EPWM2 channel opening window enable<br>0: Opening window is disabled<br>1: Opening window is enabled | 0 |
| 0 | C0PWM0_WEN | Analog comparator 0 EPWM0 channel opening window enable<br>0: Opening window is disabled<br>1: Opening window is enabled | 0 |

## 21.6.11 Analog Comparator Write Enable Control Register (LOCK)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | LOCK | When LOCK=0x55, enable the operation of ACMP related registers. (For details, please refer to the description of ACMP register mapping.)<br>When LOCK=other values, disable the operation of ACMP related registers. | 0x0 |

# Chapter 22 Digital-to-Analog Converter (DAC)

## 22.1Overview

The chip contains an 8-bit Digital-to-Analog Converter (DAC).

The DAC's input digital signal register is DAC_S[7:0] (the lower 8 bits of the controller CON0), with 8 valid bits.

The digital input signal range is from 0x00 to 0xFF, where 0x00 corresponds to a zero analog output of 0V, and 0xFF corresponds to the full-scale analog output of VREF.

## 22.2Block Diagram of Structure


| Name | Signal type | Remark |
|---|---|---|
| VREF | DAC reference voltage input | Source: ADCLDO (3.6V/4.2V/VDD) |
| DAC_S[7:0] | Digital signal input | Voltage value set in the digital controller (CON0) |
| DAC_O | Analog signal output | DAC channel analog output |

The formula for the analog output voltage of the DAC channel pin: $DAC\_O = \frac{VREF}{256} \times DAC\_S[7:0]$

## 22.3Features

- The analog reference voltage input is the output of ADCLDO.
- Multiple levels of output voltage are available for selection.

## 22.4Register Mapping

(DAC base address = 0x4006_8360) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CON0(P1B) | 0x000 | R/W | DAC control register 0 | 0x0 |
| LOCK | 0x004 | R/W | DAC register enable control bit | 0x0 |

Note 1: The registers marked with (P1B) are protected registers.

Note 2: (P1B): When LOCK==55H, the marked registers are allowed to be written; for other values, writing is prohibited.

# 22.5Register Description

## 22.5.1 DAC Control Register 0 (CON0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:17 | - | Reserved | - |
| 17 | DAC_OEN2 | DAC output to port enable<br>1: Enable output to port P53<br>0: Disable output to port | 0 |
| 16 | DAC_OEN1 | DAC output to port enable<br>1: Enable output to port P12<br>0: Disable output to port | 0 |
| 15:9 | - | Reserved | - |
| 8 | DAC_EN | DAC module enable<br>1: Enable<br>0: Disable | 0 |
| 7:0 | DAC_S | DAC_O = (VREF/256) × DAC_S[7:0]<br>DAC_S: The data input range for DAC conversion is from 0x00 to 0xFF. | 0x0 |

## 22.5.2 DAC Write Enable Control Register (LOCK)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | LOCK | When LOCK=0x55, enable the operation of DAC related registers. (For details, please refer to the description of ACMP register mapping.)<br>When LOCK=other values, disable the operation of DAC related registers. | 0x0 |

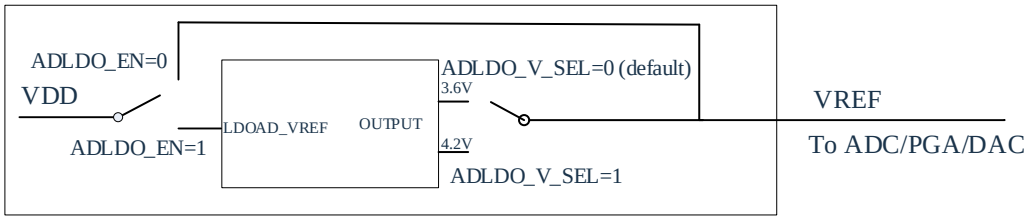
# Chapter 23 ADCLDO

It contains an internal LDO to provide reference voltage to some modules.

## 23.1Features

- Analog input voltage range: VDD.
- Output voltage: can be selected from VDD, 4.2V, and 3.6V.

## 23.2Block Diagram of Structure




ADCLDO

## 23.3Register Mapping

(ADCLDO base address = 0x4006_8340) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CON0$_{(P1B)}$ | 0x000 | R/W | ADCLDO control register 0 | 0x100000 |
| LOCK | 0x00C | R/W | ADCLDO register enable control bit | 0x0 |

Note 1: The registers marked with (P1B) are protected registers.

Note 2: (P1B): When LOCK==55H, the marked registers are allowed to be written; for other values, writing is prohibited.

# 23.4Register Description

## 23.4.1 ADCLDO Control Register 0 (CON0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:9 | - | Reserved | - |
| 8 | ADCLDO_EN | ADCLDO module enable<br>0: Disable, ADLDO outputs VDD<br>1: Enable, ADLDO outputs LDO voltage | 0 |
| 7:0 | ADCLDO_V_SEL | ADCLDO output voltage selection<br>0x55: LDO voltage outputs 4.2V<br>Other: LDO voltage outputs 3.6V | 0x0 |

## 23.4.2 ADCLDO Write Enable Control Register (LOCK)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | LOCK | When LOCK=0x55, enable the operation of ADCLDO related registers.<br>(For details, please refer to the description of ACMP register mapping.)<br>When LOCK=other values, disable the operation of ADCLDO related registers. | 0x0 |

# Chapter 24 Nested Vector Interrupt Controller (NIVC)

The Cortex®-M0+ CPU provides a Nested Vector Interrupt Controller (NVIC) for interrupt handling.

## 24.1Features

- Supports nested vector interrupts.
- Automatically saves and restores processor state.
- Supports dynamic priority changes.
- Simplified and defined interrupt times.

The NVIC processes all supported exceptions based on their priority. All exceptions are handled in “Handler mode”. The NVIC supports 23 discrete interrupts (IRQ[31:0]), with each interrupt supporting 4 levels of interrupt priority. All interrupts and most system exceptions can be configured to have different priorities. When an interrupt occurs, the NVIC compares the priority of the new interrupt with the current interrupt. If the new interrupt has a higher priority, it is immediately processed.

After accepting an interrupt, the start address of the Interrupt Service Routine (ISR) can be obtained from the vector table in memory. Software does not need to determine which interrupt is being responded to or allocate the start address of the relevant ISR. Once the start address is obtained, the NVIC automatically saves the values of processor state registers (PC, PSR, LR, R0~R3, R12) to the stack. After the ISR ends, the NVIC restores the values of the relevant registers from the stack and resumes normal operation. This allows for minimal and identified interrupt handling time.

The NVIC supports “Tail-Chaining”, which efficiently handles back-to-back interrupts without saving and restoring the current state, reducing the latency for switching from the current ISR to a pending ISR. The NVIC also supports “Late Arrival”, improving the efficiency of concurrent interrupts. When a higher-priority interrupt request occurs before the current ISR begins executing (during the stage of saving processor state and obtaining the start address), the NVIC immediately processes the higher-priority interrupt, thus enhancing real-time performance.

For more detailed information, please refer to the “ARM® Cortex®-M0+ Technical Reference Manual” and the “ARM®v6-M Architecture Reference Manual”.

## 24.2Exception Mode and System Interrupt Mapping

The table below lists the exception modes supported by this product. Like all interrupts, software can set 4 levels of priority for some of these exceptions. Users can configure the highest priority as 0 and the lowest priority as 3. The default priority for all user-configurable interrupts is 0.

| Exception name | Exception number | Priority |
|---|---|---|
| Reset | 1 | -3 |
| NMI | 2 | -2 |
| Hard Fault | 3 | -1 |
| Reserved | 4~10 | Reserved |
| SVCall | 11 | Configurable |
| Reserved | 12~13 | Reserved |
| PendSV | 14 | Configurable |
| SysTick | 15 | Configurable |
| Interrupt (IRQ0~IRQ31) | 16~47 | Configurable |

Note: Priority 0 is the 4th priority in the system, after “Reset”, “NMI” and “Hard Fault”.

## 24.3Vector Table

| Exception number | Interrupt number | Vector address | Exception type | DMA | Description |
|---|---|---|---|---|---|
| 1-15 | - | 0x00-0x3c | System exception | | |
| 16 | 0 | 0x40 | INTLVI | | Voltage detection |
| 17 | 1 | 0x44 | INTP0 | | Edge detection of pin input |
| 18 | 2 | 0x48 | INTP1 | | Edge detection of pin input |
| 19 | 3 | 0x4c | INTP2 | | Edge detection of pin input |
| 20 | 4 | 0x50 | INTP3 | | Edge detection of pin input |
| 21 | 5 | 0x54 | INTTM01H | | Timer channel 01 count or capture end (high 8 bits of timer operation) |
| 22 | 6 | 0x58 | INTCCP | | CCP interrupt |
| 23 | 7 | 0x5c | INTACMP | | Comparator completion interrupt |
| 24 | 8 | 0x60 | INTADC | DTC1 | ADC interrupt |
| 25 | 9 | 0x64 | INTSPI | DTC2 | SPI interrupt |
| 26 | 10 | 0x68 | INTIICA0 | DTC3 | IIC interrupt |
| 27 | 11 | 0x6c | INTUART0 | DTC4 | UART0 interrupt |
| 28 | 12 | 0x70 | INTUART1 | DTC5 | UART1 interrupt |
| 29 | 13 | 0x74 | INTEPWM | DTC6 | EPWM interrupt |
| 30 | 14 | 0x78 | INTTIMER0 | DTC7 | TIMER0 interrupt |
| 31 | 15 | 0x7c | INTTIMER1 | | TIMER1 interrupt |
| 32 | 16 | 0x80 | INTHALL | | HALL interrupt |
| 33 | 17 | 0x84 | INTLSITIMER | | LSI timer interrupt |
| 34 | 18 | 0x88 | Reserved | | |
| 35 | 19 | 0x8c | INTTM00 | | Timer channel 00 count end or capture end |
| 36 | 20 | 0x90 | INTTM01 | | Timer channel 01 count end or capture end |
| 37 | 21 | 0x94 | INTTM02 | | Timer channel 02 count end or capture end |
| 38 | 22 | 0x98 | INTTM03 | | Timer channel 03 count end or capture end |
| 39 | 23 | 0x9c | Reserved | | |
| 40 | 24 | 0xa0 | Reserved | | |
| 41 | 25 | 0xa4 | Reserved | | |
| 42 | 26 | 0xa8 | Reserved | | |
| 43 | 27 | 0xac | Reserved | | |
| 44 | 28 | 0xb0 | Reserved | | |
| 45 | 29 | 0xb4 | Reserved | | |
| 46 | 30 | 0xb8 | Reserved | | |
| 47 | 31 | 0xbc | INTFL | DTC0 | FLASH programming end |

## 24.4Register Mapping

(NVIC base address = 0xE000_E000) RO: read only; WO: write only; R/W: read/write.

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| ISER | 0x100 | R/W | Interrupt Set Enable Control Register | 0x0 |
| ICER | 0x180 | R/W | Interrupt Clear Enable Control Register | 0x0 |
| ISPR | 0x200 | R/W | Interrupt Set Pending Control Register | 0x0 |
| ICPR | 0x280 | R/W | Interrupt Clear Pending Control Register | 0x0 |
| IPR0 | 0x400 | R/W | IRQ0~IRQ3 Interrupt Priority Register | 0x0 |
| IPR1 | 0x404 | R/W | IRQ4~IRQ7 Interrupt Priority Register | 0x0 |
| IPR2 | 0x408 | R/W | IRQ8~IRQ11 Interrupt Priority Register | 0x0 |
| IPR3 | 0x40C | R/W | IRQ12~IRQ15 Interrupt Priority Register | 0x0 |
| IPR4 | 0x410 | R/W | IRQ16~IRQ19 Interrupt Priority Register | 0x0 |
| IPR5 | 0x414 | R/W | IRQ20~IRQ23 Interrupt Priority Register | 0x0 |
| IPR6 | 0x418 | R/W | IRQ24~IRQ27 Interrupt Priority Register | 0x0 |
| IPR7 | 0x41C | R/W | IRQ28~IRQ31 Interrupt Priority Register | 0x0 |

(INTM base address = 0x4004_5B38) RO: read only; WO: write only; R/W: read/write.

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| EGP0 | 0x000 | R/W | External Interrupt Rising Edge Enable Register | 0x0 |
| EGN0 | 0x001 | R/W | External Interrupt Falling Edge Enable Register | 0x0 |

# 24.5Register Description

## 24.5.1 Interrupt Set Enable Control Register (ISER)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | SETENA | Interrupt enable bit<br>Enables one or more interrupts. Each bit represents an interrupt from IRQ0 to IRQ31<br>(vector number from 16 to 47).<br>Write operation:<br>0: Invalid<br>1: Write 1 to enable related interrupts<br>Read operation:<br>0: Disable related interrupt status<br>1: Enable related interrupt status<br>Note: Reading the value of this register indicates that it is currently enabled. | 0x0 |

## 24.5.2 Interrupt Clear Enable Control Register (ICER)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | CLRENA | Interrupt disable bit<br>Disable one or more interrupts. Each bit represents an interrupt from IRQ0 to IRQ31 (vector number from 16 to 47).<br>Read operation:<br>0: Invalid<br>1: Write 1 to enable related interrupts<br>Read operation:<br>0: Disable related interrupt status<br>1: Enable related interrupt status<br>Note: Reading the value of this register indicates that it is currently enabled. | 0x0 |

## 24.5.3 Interrupt Set Pending Control Register (ISPR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | SETPEND | Set interrupt pending bit<br>Write operation:<br>0: Invalid<br>1: Set 1 to pending state. Each bit represents an interrupt from IRQ0 to IRQ31 (vector number from 16 to 47).<br>Read operation:<br>0: The relevant interrupt is not pending<br>1: The relevant interrupt is in the pending state<br>Note: Reading this register indicates that the current state is pending. | 0x0 |

### 24.5.4 Interrupt Clear Pending Control Register (ICPR)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | CLRPEND | Clear interrupt pending bit<br>Write operation:<br>0: Invalid<br>1: Write 1 to clear pending state. Each bit represents an interrupt from IRQ0 to IRQ31 (vector number from 16 to 47).<br>Read operation:<br>0: The relevant interrupt is not pending<br>1: The relevant interrupt is pending<br>Note: Reading this register indicates that the current state is pending. | 0x0 |

### 24.5.5 IRQ0~IRQ3 Interrupt Priority Register (IPR0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:30 | PRI_3 | IRQ3 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 29:24 | - | Reserved | - |
| 23:22 | PRI_2 | IRQ2 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 21:16 | - | Reserved | - |
| 15:14 | PRI_1 | IRQ1 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 13:8 | - | Reserved | - |
| 7:6 | PRI_0 | IRQ0 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 5:0 | - | Reserved | - |

### 24.5.6 IRQ4~IRQ7 Interrupt Priority Register (IPR1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:30 | PRI_7 | IRQ7 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 29:24 | - | Reserved | - |
| 23:22 | PRI_6 | IRQ6 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 21:16 | - | Reserved | - |
| 15:14 | PRI_5 | IRQ5 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 13:8 | - | Reserved | - |
| 7:6 | PRI_4 | IRQ4 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 5:0 | - | Reserved | - |

### 24.5.7 IRQ8~IRQ11 Interrupt Priority Register (IPR2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:30 | PRI_11 | IRQ11 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 29:24 | - | Reserved | - |
| 23:22 | PRI_10 | IRQ10 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 21:16 | - | Reserved | - |
| 15:14 | PRI_9 | IRQ9 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 13:8 | - | Reserved | - |
| 7:6 | PRI_8 | IRQ8 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 5:0 | - | Reserved | - |

### 24.5.8 IRQ12~IRQ15 Interrupt Priority Register (IPR3)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:30 | PRI_15 | IRQ15 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 29:24 | - | Reserved | - |
| 23:22 | PRI_14 | IRQ14 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 21:16 | - | Reserved | - |
| 15:14 | PRI_13 | IRQ13 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 13:8 | - | Reserved | - |
| 7:6 | PRI_12 | IRQ12 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 5:0 | - | Reserved | - |

### 24.5.9 IRQ16~IRQ19 Interrupt Priority Register (IPR4)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:30 | PRI_19 | IRQ19 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 29:24 | - | Reserved | - |
| 23:22 | PRI_18 | IRQ18 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 21:16 | - | Reserved | - |
| 15:14 | PRI_17 | IRQ17 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 13:8 | - | Reserved | - |
| 7:6 | PRI_16 | IRQ16 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 5:0 | - | Reserved | - |

### 24.5.10 IRQ20~IRQ23 Interrupt Priority Register (IPR5)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:30 | PRI_23 | IRQ23 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 29:24 | - | Reserved | - |
| 23:22 | PRI_22 | IRQ22 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 21:16 | - | Reserved | - |
| 15:14 | PRI_21 | IRQ21 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 13:8 | - | Reserved | - |
| 7:6 | PRI_20 | IRQ20 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 5:0 | - | Reserved | - |

### 24.5.11 IRQ24~IRQ27 Interrupt Priority Register (IPR6)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:30 | PRI_27 | IRQ27 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 29:24 | - | Reserved | - |
| 23:22 | PRI_26 | IRQ26 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 21:16 | - | Reserved | - |
| 15:14 | PRI_25 | IRQ25 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 13:8 | - | Reserved | - |
| 7:6 | PRI_24 | IRQ24 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 5:0 | - | Reserved | - |

### 24.5.12 IRQ28~IRQ31 Interrupt Priority Register (IPR7)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:30 | PRI_31 | IRQ31 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 29:24 | - | Reserved | - |
| 23:22 | PRI_30 | IRQ30 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 21:16 | - | Reserved | - |
| 15:14 | PRI_29 | IRQ29 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 13:8 | - | Reserved | - |
| 7:6 | PRI_28 | IRQ28 priority<br>0 represents the highest priority, and 3 represents the lowest priority. | 0x0 |
| 5:0 | - | Reserved | - |

### 24.5.13 External Interrupt Rising Edge Enable Register (EGP0)

The EGP0 and EGN0 registers are used to set the active edge for INTP0 to INTP3. These registers are set through 8-bit memory manipulation instructions.

After a reset signal is generated, the values of these registers become “00H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:4 | - | Reserved | 0x0 |
| 3 | EGP3 | INTP3 external interrupt rising edge enable register:<br>0: Disable external interrupt rising edge<br>1: Enable external interrupt rising edge | 0 |
| 2 | EGP2 | INTP2 external interrupt rising edge enable register:<br>0: Disable external interrupt rising edge<br>1: Enable external interrupt rising edge | 0 |
| 1 | EGP1 | INTP1 external interrupt rising edge enable register:<br>0: Disable external interrupt rising edge<br>1: Enable external interrupt rising edge | 0 |
| 0 | EGP0 | INTP0 external interrupt rising edge enable register:<br>0: Disable external interrupt rising edge<br>1: Enable external interrupt rising edge | 0 |

## 24.5.14 External Interrupt Falling Edge Enable Register (EGN0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:4 | - | Reserved | 0x0 |
| 3 | EGN3 | INTP3 external interrupt falling edge enable register:<br>0: Disable external interrupt falling edge<br>1: Enable external interrupt falling edge | 0 |
| 2 | EGN2 | INTP2 external interrupt falling edge enable register:<br>0: Disable external interrupt falling edge<br>1: Enable external interrupt falling edge | 0 |
| 1 | EGN1 | INTP1 external interrupt falling edge enable register:<br>0: Disable external interrupt falling edge<br>1: Enable external interrupt falling edge | 0 |
| 0 | EGN0 | INTP0 external interrupt falling edge enable register:<br>0: Disable external interrupt falling edge<br>1: Enable external interrupt falling edge | 0 |

| EGPn | EGNn | INTPn pin active edge selection (n=0~3) |
|---|---|---|
| 0 | 0 | Disable detection of edges. |
| 0 | 1 | Falling edge |
| 1 | 0 | Rising edge |
| 1 | 1 | Both edges |

The ports corresponding to the EGPn and EGNn bits are shown in Table 24-1.

Table 24-1 Interrupt request signal corresponding to EGPn bit and EGNn bit

| Detect enable bit | | Interrupt request signal |
|---|---|---|
| EGP0 | EGN0 | INTP0 |
| EGP1 | EGN1 | INTP1 |
| EGP2 | EGN2 | INTP2 |
| EGP3 | EGN3 | INTP3 |

Note 1: If the input port used for the external interrupt function is switched to output mode, an active edge may be detected and an INTPn interrupt may be generated. When switching to the output mode, the port mode register (PMxx) must be set to “0” after disabling the detection of an edge (EGPn, EGNn=0, 0).

Note 2: Please refer to “2.1 Port Functions” for the ports with edge detection.

# Chapter 25 Standby Function

## 25.1Standby Function

The standby function is a function that further reduces the operating current of the system and has the following two modes.

(1) Sleep mode

Sleep mode is the mode in which the CPU is stopped from running the clock. If the high-speed on-chip oscillator or the low-speed on-chip oscillator is oscillating before the sleep mode is set, the clocks continue to oscillate. Although this mode does not reduce the operating current to the level of deep sleep mode, it is an effective mode for wanting to restart processing immediately through interrupt requests or if you want to run frequently in intermittent operations.

(2) Deep sleep mode

Deep sleep mode is a mode that stops the oscillation of the high-speed on-chip oscillator and stops the entire system. The operating current of the CPU can be greatly reduced.

Because deep sleep mode can be released by interrupt requests, intermittent operations can also be performed. However, because the wait time to ensure oscillation stability is required when releasing the deep sleep mode, it is necessary to select the sleep mode if you need to start processing immediately through the interrupt request.

In either mode, registers, flags, and data memory are all left set to before standby mode, and the output latches and output buffers of the input/output ports are also maintained.

Note 1: When shifting to the deep sleep mode, WFI instructions must be executed after stopping peripheral hardware running in the master system clock.

Note 2: To reduce the operating current of the A/D converter, after clearing bit 4 (ADCEN) of the A/D converter control register (CON) and bit 7 (The ADCST conversion is completed by clearing 0 via hardware.) of the control register (CON2) to “0”, execute the WFI instruction after stopping the A/D conversion operation.

Note 3: The option byte can be used to select whether to continue or stop the low-speed internal oscillator in sleep mode or deep sleep mode. For details, please refer to “Chapter 31 Option Byte”.

## 25.2Sleep Mode

### 25.2.1 Setting of Sleep Mode

When the SLEEPDEEP bit of the SCR register is 0, execute the WFI instruction and enter sleep mode. In sleep mode, the CPU stops operating, but the values of the internal registers are still maintained, and the peripheral modules remain in the state they were in before they entered sleep mode. The status of peripheral modules, oscillators, etc. in sleep mode is shown in Table 25-1.

Sleep mode can be set regardless of whether the CPU clock before setup is a high-speed on-chip oscillator clock or a low-speed on-chip oscillator clock.

Caution: When the interrupt request flag is “1” (an interrupt request signal is generated), the interrupt request signal is used to release the sleep mode. Therefore, even if the WFI instruction is executed in this case, it does not shift to the sleep mode.

Table 25-1 Operation status in sleep mode

| Item | | Sleep mode |
|---|---|---|
| System clock | $f_{IH}$ | Operation continues |
| | $f_{IL}$ | The operating state is set via the OSMC register and the SUBCKSEL register, and the set state is retained. |
| CPU | | Operation stopped |
| Code flash memory | | Operation stopped |
| SRAM0 | | Operation stopped |
| SRAM1 | | Operation stopped |
| Port (latch) | | Status before sleep mode was set is retained |
| Universal timer unit | | Operable |
| LSI_timer | | Operable |
| Clock output | | Operable |
| Watchdog timer | | If counting is set to continue before sleep, counting can continue after sleep. |
| DIVSQRT unit | | Calculation stopped |
| Division unit | | Calculation stopped |
| TIMER0/1 | | Operable |
| CCP0/1 | | Operable |
| HALL signal processing | | Operable |
| EPWM | | Operable |
| UART | | Operable |
| IICA | | Operable |
| SSP/SPI | | Operable |
| DMA | | Operable |
| ADC | | Operable |
| PGA0/1/2 | | Operable |
| ACMP0/1 | | Operable |
| DAC | | Operable |
| ADCLDO | | Operable |
| Power-on reset function | | Operable |
| Voltage detection function | | Operable |
| External interrupt | | Operable |
| CRC | High-speed CRC | Operable |
| | Universal CRC | Operation stopped |
| SFR guard function | | Operation stopped |

Note 1: Operation stopped: Operation is automatically stopped before switching to the sleep mode.

Note 2: $f_{IH}$: High-speed on-chip oscillator clock

Note 3: $f_{IL}$: Low-speed on-chip oscillator clock

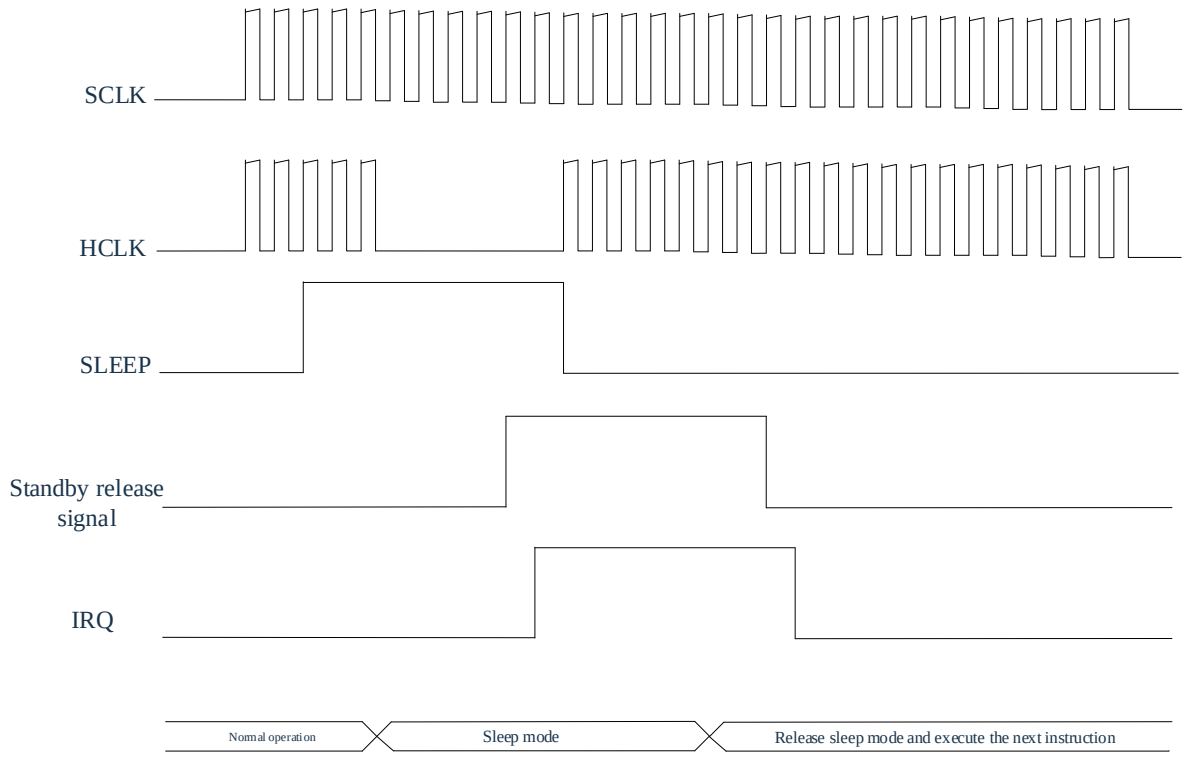
### 25.2.2 Sleep Mode Release

The sleep mode can be released by any interrupt or external reset, POR reset, low voltage detection reset, or WDT reset.

(1) Released by interrupts

When an interrupt is generated and the interrupt is allowed to be accepted, sleep mode is released and the CPU begins processing interrupt services.

Figure 25-1 Release sleep mode by interrupt requests



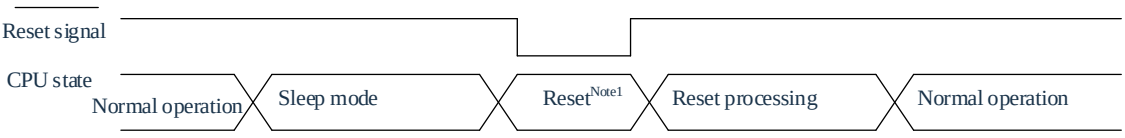

Note 1: From the generation of the standby release signal to the release of sleep mode, it takes 16 clocks to start executing the interrupt service program.

Note 2: Some of the standby release signals cannot be cleared by themselves, but must be cleared by writing to a register. This is usually done by writing to a register in the interrupt service program.

(2) Released by resets

When a reset signal is generated, the CPU is in reset state and the sleep mode is released. As with a normal reset, the program is executed after shifting to the reset vector address.

Figure 25-2 Release sleep mode by resets




Note 1: For reset processing, please refer to “Chapter 26 Reset Function”. For reset processing of power-on reset (POR) and voltage detection (LVD), refer to “Chapter 27 Power-on Reset Circuit”.

# 25.3Deep Sleep Mode

## 25.3.1 Setting of Deep Sleep Mode

When the SLEEPDEEP bit of the SCR register is 1, the WFI instruction is executed and deep sleep mode is entered. In this mode, the CPU, most of the peripheral modules, and the oscillator operation stops. However, the values of the CPU internal registers, the RAM data, the peripheral modules, the state of the I/O are maintained. The operating status of the peripheral module and the oscillator in deep sleep mode is shown in Table 25-2.

Caution: When the interrupt request flag is "1" (an interrupt request signal is generated), the interrupt request signal is used to release deep sleep mode. Therefore, if the WFI instruction is executed in this case, it is released as soon as it enters deep sleep mode. Returns to operation mode after executing the WFI instruction and after a deep sleep mode release time has elapsed.

Table 25-2 Operation status in deep sleep mode

| Item | | Deep sleep mode |
|---|---|---|
| System clock | $f_{IH}$ | Operation stopped |
| | $f_{IL}$ | The operating state is set via the OSMC register and the SUBCKSEL register, and the set state is retained. |
| CPU | | Operation stopped |
| Code flash memory | | Operation stopped |
| SRAM0 | | Operation stopped |
| SRAM1 | | Operation stopped |
| Port (latch) | | Status before deep sleep mode was set is retained |
| Universal timer unit TIMER4 | | Operation disabled |
| LSI_timer | | Operable |
| Clock output | | Operation disabled |
| Watchdog timer | | Operable |
| DIVSQRT unit | | Calculation stopped |
| Division unit | | Calculation stopped |
| TIMER0/1 | | Operation disabled |
| Capture/Compare/PWM (CCP0/1) | | Operation disabled |
| HALL signal processing | | Operation disabled |
| EPWM | | Operation disabled |
| UART | | Operation disabled |
| IICA | | Wake up by address matching |
| SSP/SPI | | Operation disabled |
| DMA | | Accept DMA boot source |
| ADC | | Operation disabled |
| PGA0/1/2 | | Operation disabled |
| ACMP0/1 | | Operation disabled |
| DAC | | Operation disabled |
| ADCLDO | | Operation disabled |

| | | |
|---|---|---|
| Power-on reset function | | Operable |
| Voltage detection function | | Operable |
| External interrupt | | Operable |
| CRC | High-speed CRC | Operation stopped |
| | Universal CRC | Operation stopped |
| SFR guard function | | Operation stopped |

Note 1: Operation stopped: Operation is automatically stopped before switching to the deep sleep mode.

Operation disabled: Operation is stopped before switching to the deep sleep mode.

Note 2: $f_{IH}$: High-speed on-chip oscillator clock

Note 3: $f_{IL}$: Low-speed on-chip oscillator clock

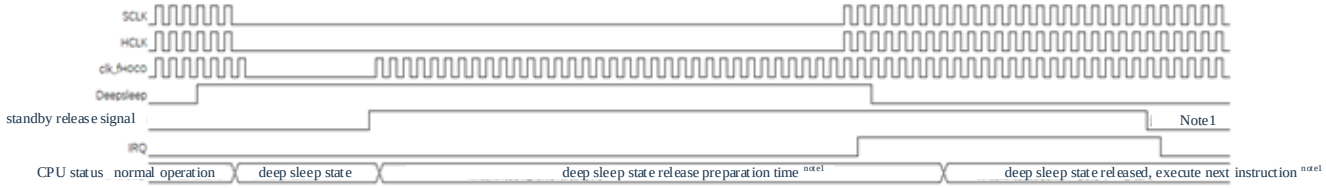
## 25.3.2 Deep Sleep Mode Release

The deep sleep mode can be released by the following two methods.

(a) Released by non-maskable interrupt requests

If an LVD detection, INTP0-3, LSI timer or WDT interrupt request occurs, the deep sleep mode is released. After the oscillation stabilization time, if it is allowed to accept interrupts, it will process the vector interrupt. If it is not allowed to accept interrupts, it executes the instruction at the next address.

Figure 25-3 Release deep sleep mode by interrupt requests




Note 1: Standby release signal: For details of the standby release signal, refer to the section on Interrupt.

Note 2: Deep sleep release preparation time: When the CPU clock is a high-speed on-chip oscillation clock before entering deep sleep mode: at least 20us.

Note 3: Wait: 14 clocks are required from when the time CPU.IRQ is valid to the interrupt service program starts.

Remark: The oscillation accuracy of the high-speed on-chip oscillator clock varies steadily depending on temperature conditions and during deep sleep mode.

(b) Released by generating reset signals

The deep sleep mode is released by generating a reset signal. Then, as with a normal reset, the program is executed after shifting to the reset vector address.

Figure 25-3 Release deep sleep mode by resetting


Note 1: For reset processing, see “Chapter 26 Reset Function”. For reset processing of power-on reset (POR) and voltage detection (LVD), see “Chapter 27 Power-on Reset Circuit”.

# 25.4Deep Sleep Mode With Partial Power Down

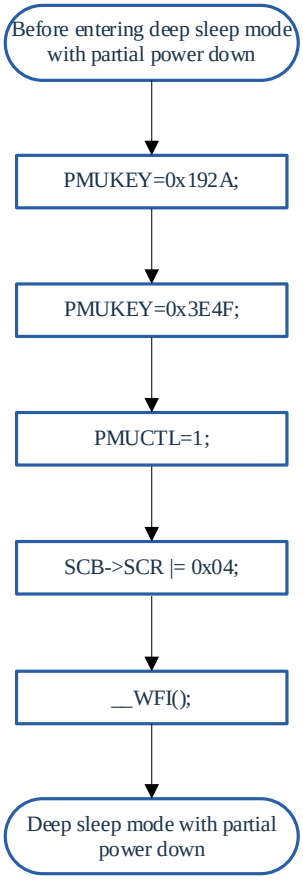
## 25.4.1 Setting of Deep Sleep Mode With Partial Power Down

The deep sleep mode with partial power loss is a deep sleep mode that further saves power consumption by turning off some peripheral power supplies on the basis of deep sleep mode. Enter the partial power-down deep sleep mode needs to configure the PWDNEEN bit of the PMUCTL register, the control bit is written to the power supply mode control protection register (PMUKEY) protection, when the deep sleep mode of partial power down requires reinitialization of the power-down periphery before it can re-operate normally, please refer to Table 25-3 The operation status in the deep sleep mode of the partial power-down is required for details.

When the SCR register has a SLEEPDEEP bit of 1 and the PMUCTL register has a PWDNEEN bit of 1, executing the WFI command can enter a partially powered-down deep sleep mode. In this mode, the CPU and the transmitter stop functioning, and most peripheral modules are powered off. However, the value of the CPU's internal registers, RAM data, the state of the I/O is maintained. The operating status of the peripheral module and the oscillator in the deep sleep mode of partial power failure is shown in Table 25-3.

The PWDNEEN bit of the PMUCTL register is controlled with reference to the section 4.4.8 Power Supply Mode Control Protection Register (PMUKEY) and the 4.4.9 Power Supply Mode Control Register (PMUCTL).

Figure 25-4 Flowchart of entering deep sleep mode with partial power down




Caution: When the interrupt request flag is “1” (an interrupt request signal is generated), the interrupt request signal is used to release the deep sleep mode. Therefore, if the WFI instruction is executed in this case, it is released as soon as the deep sleep mode is entered, and the partial power-down mode is not entered in this case. The WFI command is executed and returns to the operation mode after the deep sleep mode release time.

Table 25-3 Operation status in deep sleep mode with partial power down

| Item | | Deep sleep mode with partial power down |
|---|---|---|
| System clock | $f_{IH}$ | Operation stopped |
| | $f_{IL}$ | The operating state is set via the OSMC register and the SUBCKSEL register, and the set state is retained. |
| CPU | | Operation stopped |
| Code flash memory | | Operation stopped |
| SRAM0 | | Operation stopped |
| SRAM1 | | Operation stopped |
| Port (latch) | | Status before sleep mode was set is retained |
| Universal timer unit TIMER4 | | Operation disabled |
| LSI_timer | | Operable |
| Clock output | | Operation stopped |
| Watchdog timer | | Operable |
| DIVSQRT unit | | Calculation stopped |
| Division unit | | Calculation stopped |
| TIMER0/1 | | Operation disabled |
| Capture/Compare/PWM (CCP0/1) | | Operation disabled |
| HALL signal process | | Operation disabled |
| EPWM | | Operation disabled |
| UART | | Operation disabled |
| IICA | | Operation disabled |
| SSP/SPI | | Operation disabled |
| DMA | | Operation disabled |
| ADC | | Operation stopped |
| PGA0/1/2 | | Operation disabled |
| ACMP0/1 | | Operation disabled |
| DAC | | Operation disabled |
| ADCLDO | | Operation disabled |
| Power-on reset function | | Operable |
| Voltage detection function | | Operable |
| External interrupt | | Operable |
| CRC | High-speed CRC | Operation stopped |
| | Universal CRC | Operation stopped |
| SFR guard function | | Operation stopped |

Note 1: Operation stopped: Operation is automatically stopped before switching to the deep sleep mode with partial power down.

Operation disabled: Operation is stopped before switching to the deep sleep mode with partial power down.

After shifting to a deep sleep mode with partial power down, the power supply to the module is stopped, and the module needs to be re-initialized after being released from the mode.

Note 2: $f_{IH}$: High-speed on-chip oscillator clock

Note 3: $f_{IL}$: Low-speed on-chip oscillator clock

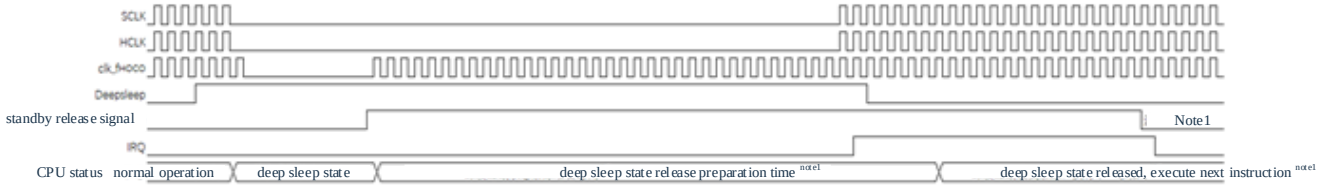
## 25.4.2 Release Deep Sleep Mode With Partial Power Down

Release the deep sleep mode with partial power down by the following 2 methods.

(a) Release deep sleep mode with partial power-down via interrupt requests

If INTP0-3, LSITIMER timer interrupt, LVI interrupt and WDT interrupt are requested, it is possible to release the deep sleep mode with partial power down. After the oscillation stabilization time, if it is allowed to accept interrupts, it will process the vector interrupt. If the interrupts are disabled, the next address is executed.

Figure 25-4 Release deep sleep mode by interrupting requests




Note 1: Standby release signals: INTP0-3, LSITIMER timer interrupt, LVI interrupt and WDT interrupt request signal.

Note 2: When the deep sleep state of partial power-down is ready to be released:

It is necessary to re-initialize the peripheral function and RAM1 function in order to ensure that the program continues to run normally.

(b) Release by generating a reset signal

The deep sleep mode with partial power-down is released by generating a reset signal. Then, as with a normal reset, the program is executed after switching to the reset vector address.

Figure 25-5 Release deep sleep mode with partial power-down by resetting


Note 1: For reset processing, refer to “Chapter 26 Reset Function”. Refer to “Chapter 27 Power-on Reset Circuit” for reset processing of the power-on reset (POR) circuit and voltage detection (LVD) circuit.

# Chapter 26 Reset Function

The following six operations are available to generate a reset signal.

(1) External reset input via RESETB pin.

(2) Internal by watchdog timer program loop detection.

(3) Internal reset by comparison of the supply voltage and detection voltage of power-on reset (POR) circuit.

(4) Internal reset by comparison of supply voltage of the voltage detection circuit (LVD) and detection voltage.

(5) Internal reset by setting the system reset request register bit (AIRCR.SYSRESETREQ) to 1.

(6) Internal reset by illegal memory access.

Internal reset is the same as external reset, and after generating a reset signal, the program is executed starting from the user-defined program start address.

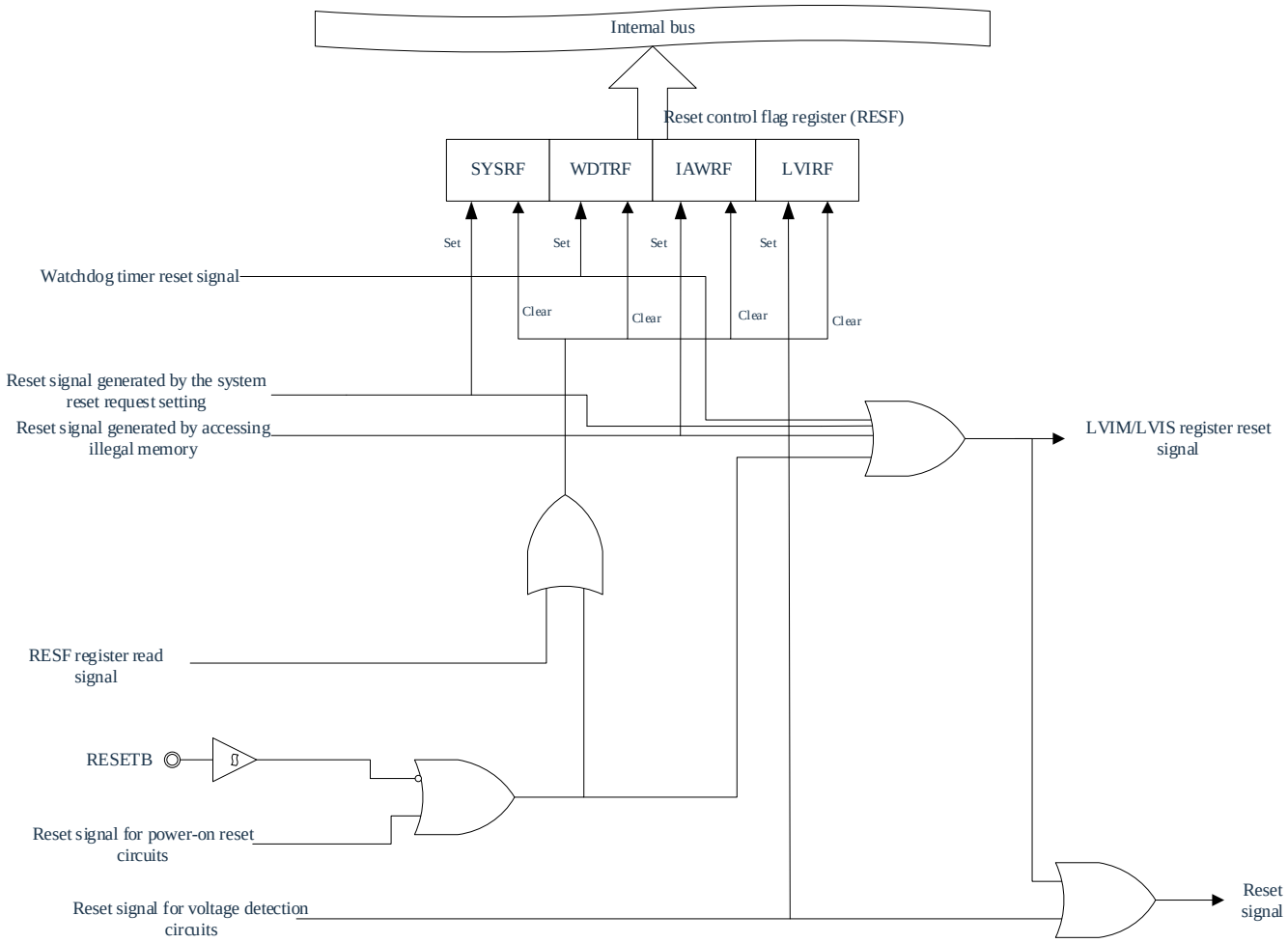
When a low level is supplied to the RESETB pin, or a program runaway is detected by the watchdog timer, or a voltage is detected in the POR and LVD circuits, or the system reset request bit is set, or an illegal memory access occurs, a reset is generated and the hardware changes to the state shown in Table 26-1.

Note 1: When performing an external reset, the RESETB pin must be held low for at least 10μs. If an external reset is performed while the supply voltage is rising, the power must be turned on after supplying a low level to the RESETB pin, and must be held low for at least 10us over the operating voltage range shown in the AC Characteristics of the User's Manual, and then be supplied with a high level.

Note 2: If a reset occurs, each SFR is initialized so that the pins change to the following states:

• During an external or power-on reset, except for P01, P02 and P03 with internal pull-up, all other I/Os are in a high-impedance state.

Figure 26-1 Block diagram of reset function




Note 1: An internal reset of the LVD circuit does not reset the LVD circuit.

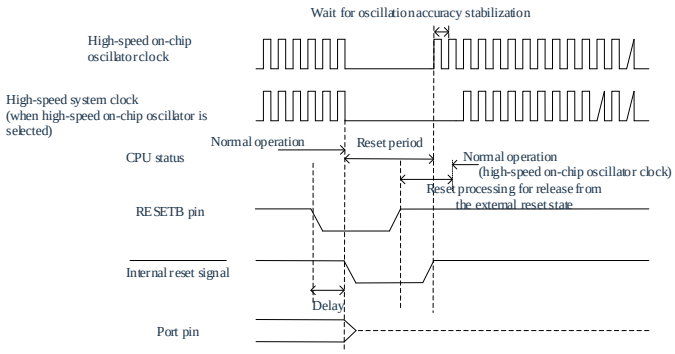
Note 2: LVIM: Voltage detection register

Note 3: LVIS: Voltage detection level register

# 26.1Reset Timing

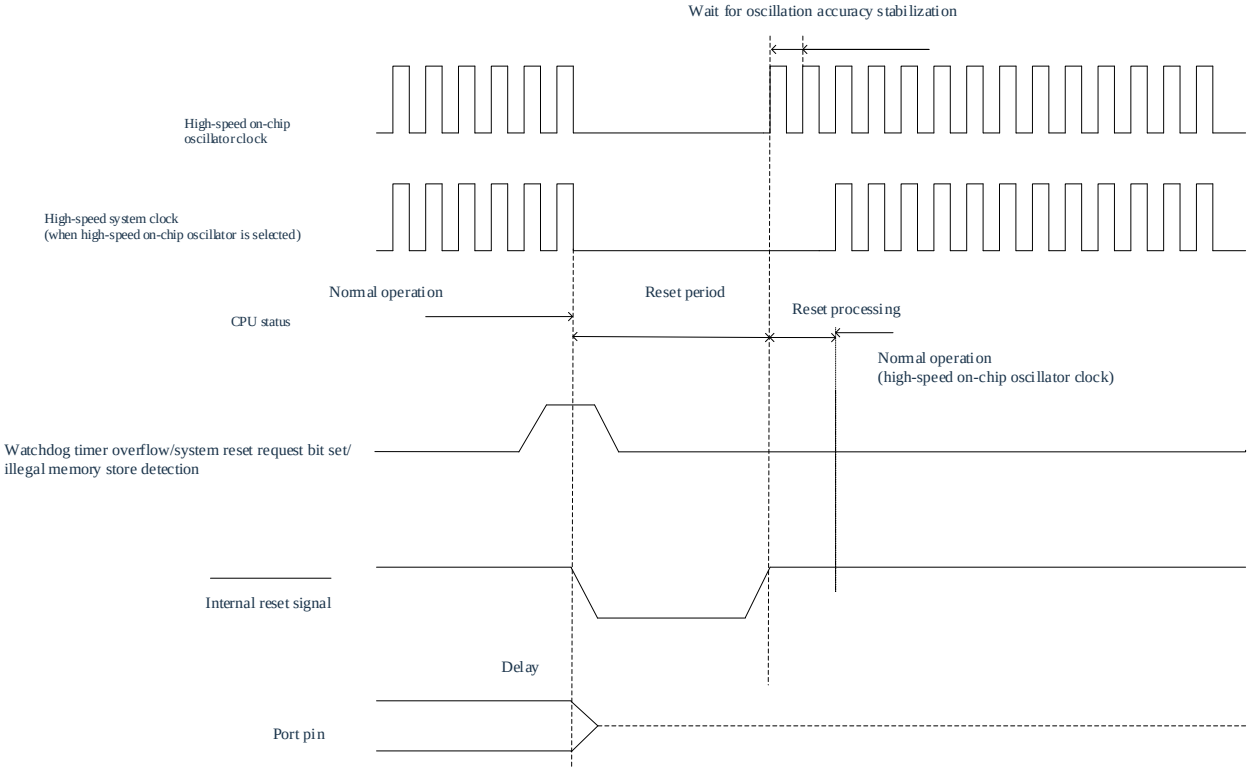
When the RESETB pin is input low, a reset is generated. The reset state is then released if the RESETB pin is input high and the program begins with a high-speed on-chip oscillator clock after the reset process is completed.

Figure 26-2 Reset timing of RESETB input




Release from the reset state is automatic in the case of a reset due to a watchdog timer overflow, execution of a system request, or detection of illegal memory access. After reset processing, execution of the program with the high-speed on-chip oscillator clock as the operating clock starts.

Figure 26-3 Timing of reset due to watchdog timer overflow, set of system reset request bit, or detection of illegal memory access




Note 1: The port pins P01, P02, and P03 have the following states:

- During an external reset, P01 is determined by the external input signal, while P02 and P03 are at a high level.
- During a power-on reset, P01, P02, and P03 are at a high level.

Note 2: The watchdog timer is also reset when an internal reset occurs.

For resets generated by the POR circuit and LVD circuit voltage detection, if $V_{DD} \geq V_{POR}$ or $V_{DD} \geq V_{LVD}$ after the reset,

the reset state is released, and the program starts executing using the high-speed on-chip oscillator clock after the reset processing. For more details, please refer to “Chapter 27 Power-on Reset Circuit” and “Chapter 28 Voltage Detection Circuit”.

Remark: $V_{POR}$: POR supply voltage rising detection voltage

• During an external reset, P01 is determined by the external input signal, while P02 and P03 are at a high level.

$V_{LVD}$: LVD detection voltage

Table 26-1 Operation status during resetting

| Item | | Reset period | | | | | |
|---|---|---|---|---|---|---|---|
| | | External reset RESINB | Watchdog reset | Power-on reset | LVD reset | Write reset register reset | Illegal memory access reset |
| System clock | $f_{IH}$ | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| | $f_{IL}$ | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| CPU | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| Code flash memory | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| SRAM0 | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| SRAM1 | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| Port (latch) | | P01,P02, and P03 are high after power on, and other pins are in high resistance state before and after reset. | P01,P02, and P03 are high after power on, and other pins are in high resistance state before and after reset. | P01,P02, and P03 are high after power on, and other pins are in high resistance state before and after reset. | P01,P02, and P03 are high after power on, and other pins are in high resistance state before and after reset. | P01,P02,P03 are high after power on, and other pins are in high resistance state before and after reset. | P01,P02, and P03 are high after power on, and other pins are in high resistance state before and after reset. |
| Universal timer unit TIMER4 | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| LSI_timer | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| Clock output | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| Watchdog timer | | Calculation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| DIVSQRT unit | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| Division unit | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| TIMER0/1 | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| Capture/Compare/PWM (CCP0/1) | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| HALL signal processing | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| EPWM | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| UART | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| IICA | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| SSP/SPI | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| ADC | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| PGA0/1/2/3 | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| ACMP0/1 | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| DAC | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| ADCLDO | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| Power-on reset function | | Detection operable | Detection operable | Detection operable | Detection operable | Detection operable | Detection operable |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Voltage detection function | | Operation disabled | | | Operable | Operation disabled | Operation disabled |
| External interrupt | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| DMA | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| CRC | High-speed CRC | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| | Universal CRC | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| SFR guard function | | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped | Operation stopped |
| Power-up state after reset completion (option byte configuration) | | Re-configure | Re-configure | Re-configure | Re-configure | Re-configure | Re-configure |

Note 1: P02 is high after power-on (in external input state during external reset), P02, P03 are high, and all other pins are in high resistance state before and after reset.

Note 2: $f_{IH}$: High-speed on-chip clock; $f_{IL}$: Low-speed on-chip clock

# 26.2Registers for Confirming the Reset Source

## 26.2.1 Register Mapping

(Reset control base address = 0x4002_0440) RO: read only; WO: write only; R/W: read/write.

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| RESF | 0x000 | RO | Reset control flag register | - |

## 26.2.2 Reset Control Flag Register (RESF)

The microcontroller has multiple internal reset generation sources. The Reset Control Flag register (RESF) holds the reset source where the reset request occurs. The RESF register can be read by an 8-bit memory manipulation instruction.

Through the input of RESETB, the power-on reset (POR) circuit is triggered, and reading the RESF register clears the SYSRF, WDTRF, IAWRF, and LVIRF flags. To determine the reset source, the value of the RESF register must be saved to any RAM, and then the reset source can be identified by checking its value stored in RAM.

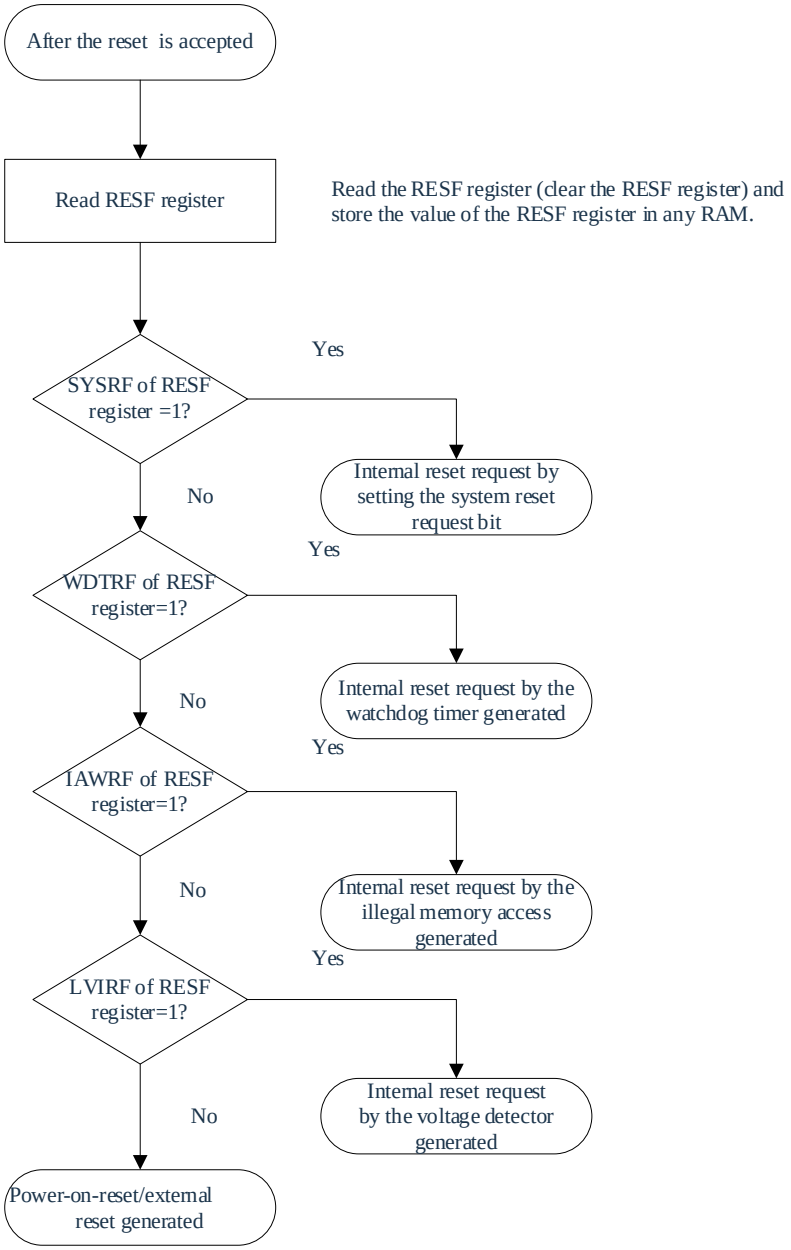
| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | SYSRF | Internal reset request resulting from the system reset request bit being set<br>0: No internal reset request is generated or the RESF register is cleared.<br>1: An internal reset request is generated. | - |
| 6:5 | - | Reserved | - |
| 4 | WDTRF | Internal reset request generated by the watchdog timer (WDT)<br>0: No internal reset request is generated or the RESF register is cleared.<br>1: An internal reset request is generated. | - |
| 3:2 | - | Reserved | - |
| 1 | IAWRF | Access to internal reset requests generated by illegal memory<br>0: No internal reset request is generated or the RESF register is cleared.<br>1: An internal reset request is generated. | - |
| 0 | LVIRF | Internal reset requests generated by the voltage detection circuit (LVD)<br>0: No internal reset request is generated or the RESF register is cleared.<br>1: An internal reset request is generated. | - |

The status of the RESF register when a reset request occurs is shown in the following table.

<table>
<tr><th>Reset source<br>Flag</th><th>RESETB input</th><th>Reset by POR</th><th>Reset generated by system reset request bit set</th><th>Reset generated by WDT</th><th>Reset generated by accessing illegal memory</th><th>Reset generated by LVD</th></tr>
<tr><td>SYSRF</td><td rowspan="4">Cleared to “0”</td><td rowspan="4">Cleared to “0”</td><td>Set to “1”</td><td>Held</td><td rowspan="2">Held</td><td rowspan="3">Held</td></tr>
<tr><td>WDTRF</td><td rowspan="3">Held</td><td>Set to “1”</td></tr>
<tr><td>IAWRF</td><td rowspan="2">Held</td><td>Set to “1”</td></tr>
<tr><td>LVIRF</td><td>Held</td><td>Set to “1”</td></tr>
</table>

Figure 26-4 shows the procedure for checking a reset source.

Figure 26-4 Example of procedure for checking reset source




Caution: The above process is an example of the confirmation step.

# Chapter 27 Power-on Reset Circuit

## 27.1Functions of Power-on-Reset Circuit

The power-on-reset circuit (POR) has the following functions.

- Generates internal reset signal at power on.

The reset signal is released when the supply voltage ($V_{DD}$) exceeds the detection voltage ($V_{POR}$). Note that the reset state must be retained until the operating voltage becomes in the range defined in AC Characteristics. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal.

- Compares supply voltage ($V_{DD}$) and detection voltage ($V_{PDR}$), generates internal reset signal when $V_{DD} < V_{PDR}$.

Note that, after power is supplied, this should be placed in the deep sleep mode, or in the reset state by utilizing the voltage detection circuit or externally input reset signal, before the operation voltage falls below the range defined in AC Characteristics. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

Note 1: When the power-on reset circuit generates an internal reset signal, the reset control flag register (RESF) is cleared to “00H”.

Note 2: The CMS32M65xx microcontroller incorporates multiple hardware functions that generate an internal reset signal. A flag that indicates the reset source is located in the reset control flag register (RESF) for when an internal reset signal is generated by the watchdog timer (WDT), voltage-detector (LVD), system reset request bit setting, or illegal-memory access. The RESF register is not cleared to 00H and the flag is set to 1 when an internal reset signal is generated by the watchdog timer (WDT), voltage-detector (LVD), system reset request bit setting, or illegal-memory access. For details of RESF register, refer to “Chapter 26 Reset Function”.

Note 3: $V_{POR}$: POR power supply rise detection voltage

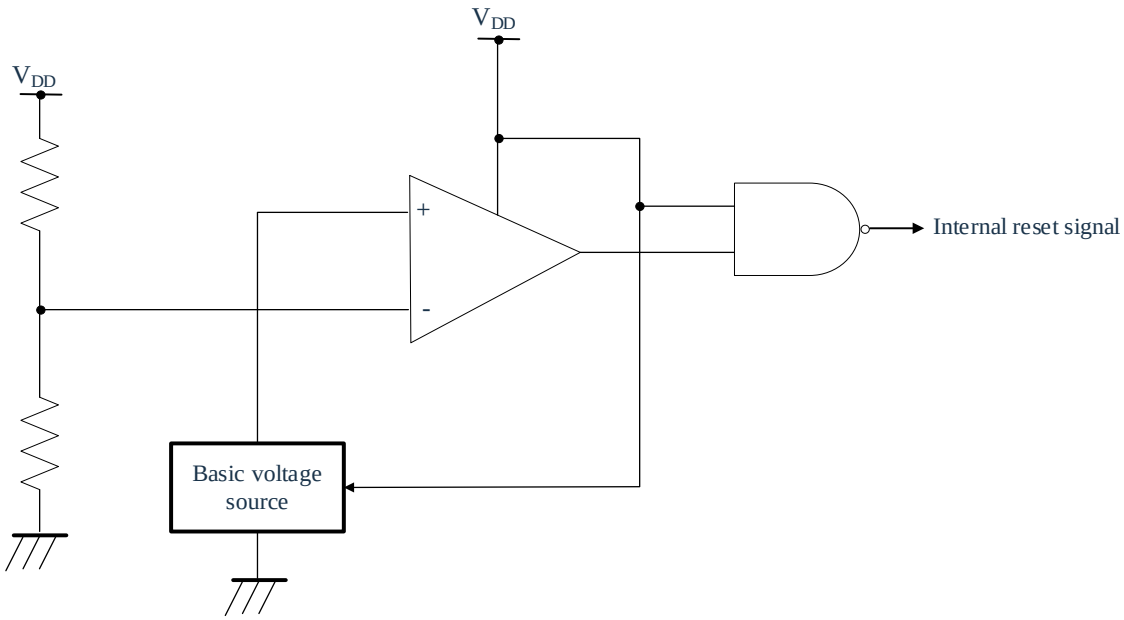
Note 4: $V_{PDR}$: POR power supply fall detection voltage

Note 5: For details, refer to the POR circuit characteristics in the data sheet.

## 27.2Structure of Power-on Reset Circuit

The block diagram of the power-on reset circuit is shown in Figure 27-1.

Figure 27-1 Block diagram of power-on reset circuit

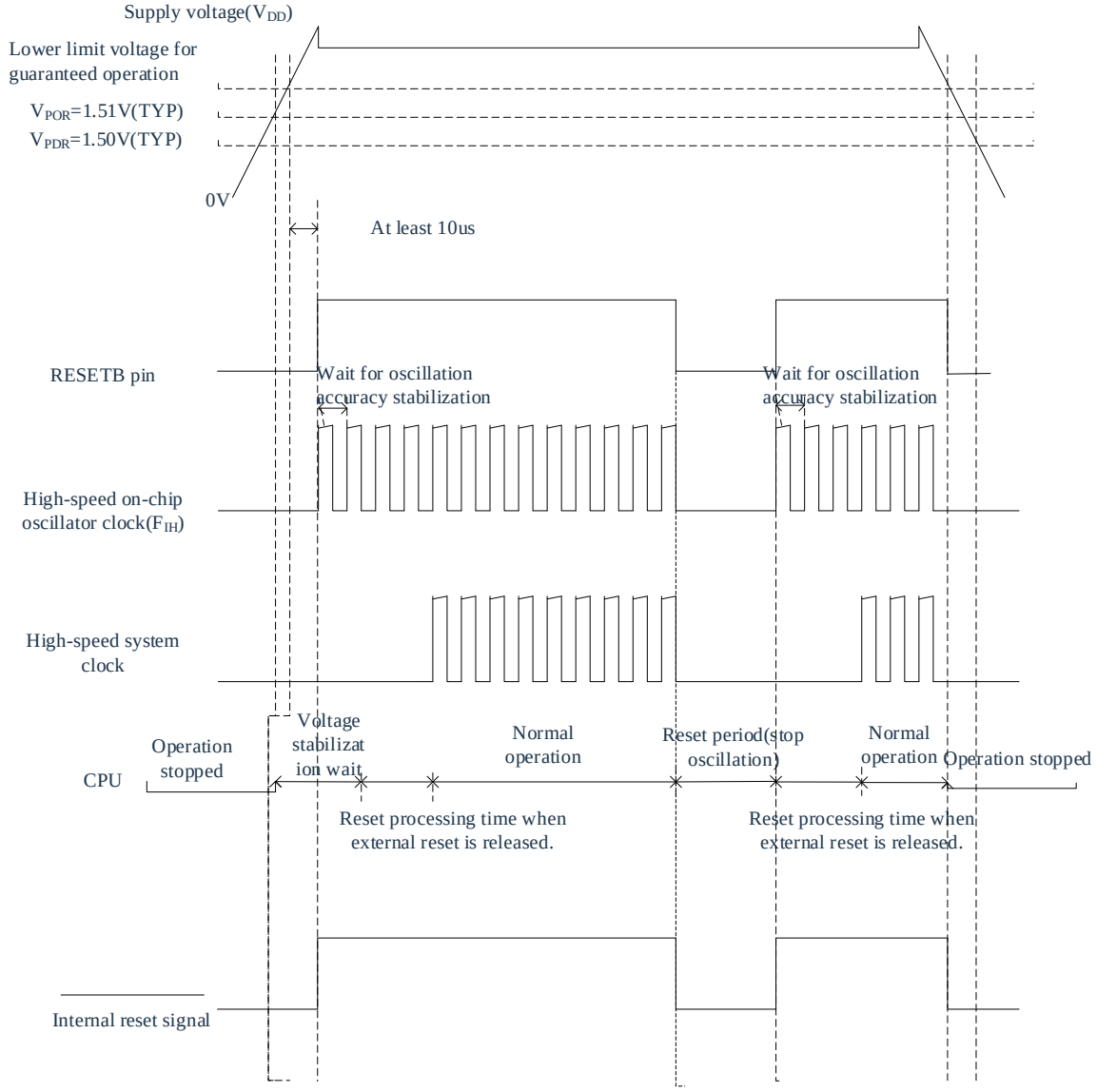


# 27.3Operation of Power-on Reset Circuit

The timing of the internal reset signal generation for the power-on reset circuit and the voltage detection circuit is shown below.

Figure 27-2 Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (1/3)

(1) When the externally input reset signal on the RESETB pin is used




Note 1: The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed on-chip oscillator clock.

Note 2: When the power supply voltage rises, the power supply voltage must be maintained by external reset before it reaches the working voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset through deep sleep mode transfer, voltage detection circuitry, or external reset before the supply voltage falls below the operating voltage range. When restarting operation, you must confirm that the supply voltage has returned to the operating voltage range.

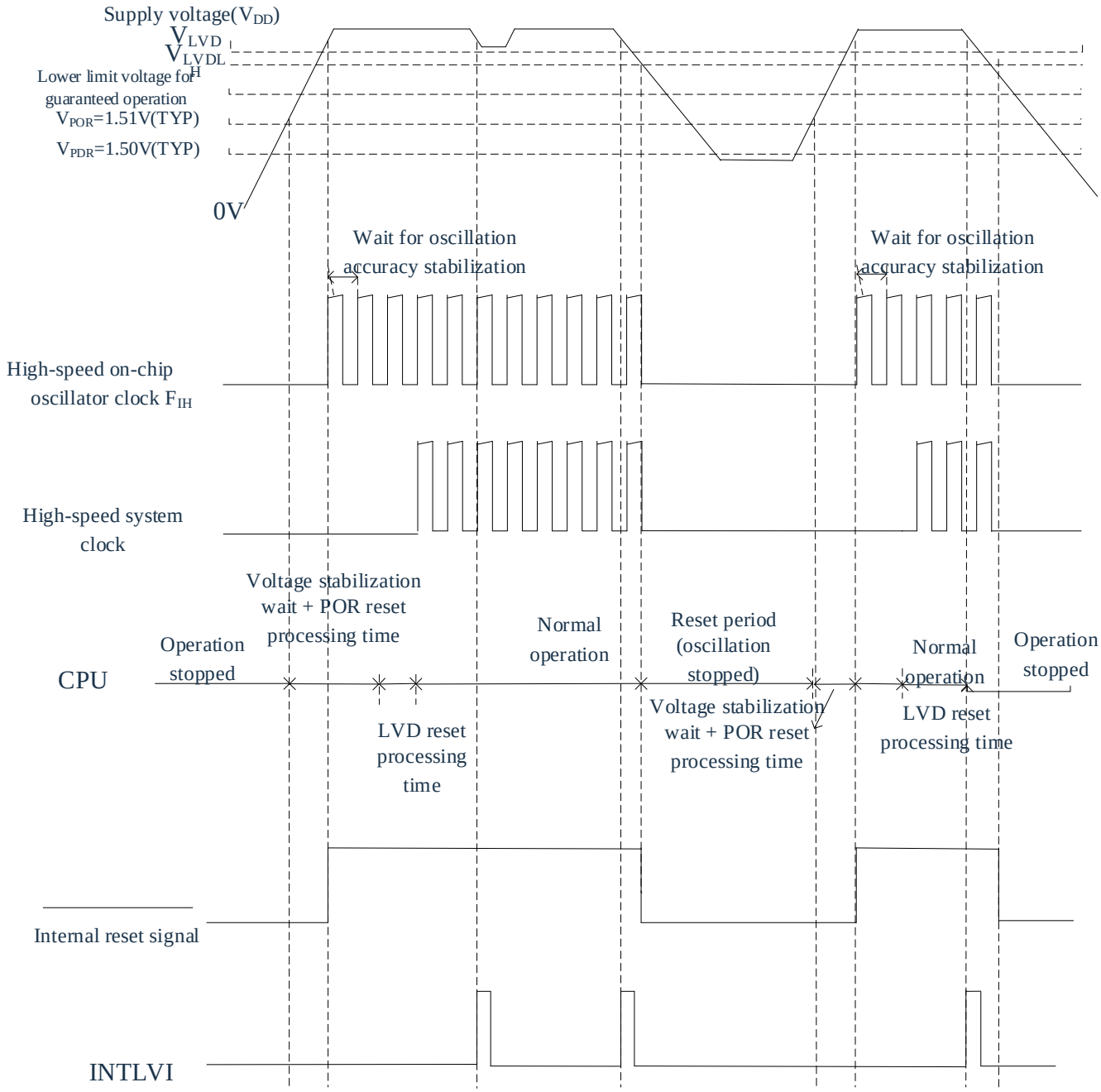
Note 3: $V_{POR}$: POR power supply rise detection voltage

Note 4: $V_{PDR}$: POR power supply fall detection voltage

Note 5: When LVD is OFF, the external reset of RESETB pin must be used. For details, please refer to “Chapter 28 Voltage Detection Circuit”.

Figure 27-2 Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (2/3)

(2) LVD interrupt & reset mode (option byte 000C1: LVIMDS1, LVIMDS0=1, 0)




Note 1: The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed on-chip oscillator clock.

Note 2: After generating the interrupt request signal (INTLVI), the LVIV bit and the LVIMD bit of the voltage detection level register (LVIS) are automatically set to "1". Therefore, considering the possibility that the power supply voltage may return to the high voltage detection voltage ($V_{LVDH}$) or higher without falling below the low voltage detection voltage ($V_{LVDL}$), follow the steps in "Figure 28-6 Setting Procedure for Confirmation/Reset of Operating Voltage" and "Figure 28-7 Setting Procedure for Interrupt and Reset" after generating an INTLVI.

Note 3: The time until normal operation begins includes the "Voltage Stabilization Wait + POR Reset Processing Time" after $V_{POR}$ (1.51V (TYP.)) is reached as well as the "LVD Reset Processing Time" after the LVD detection level ($V_{LVDH}$) is reached.

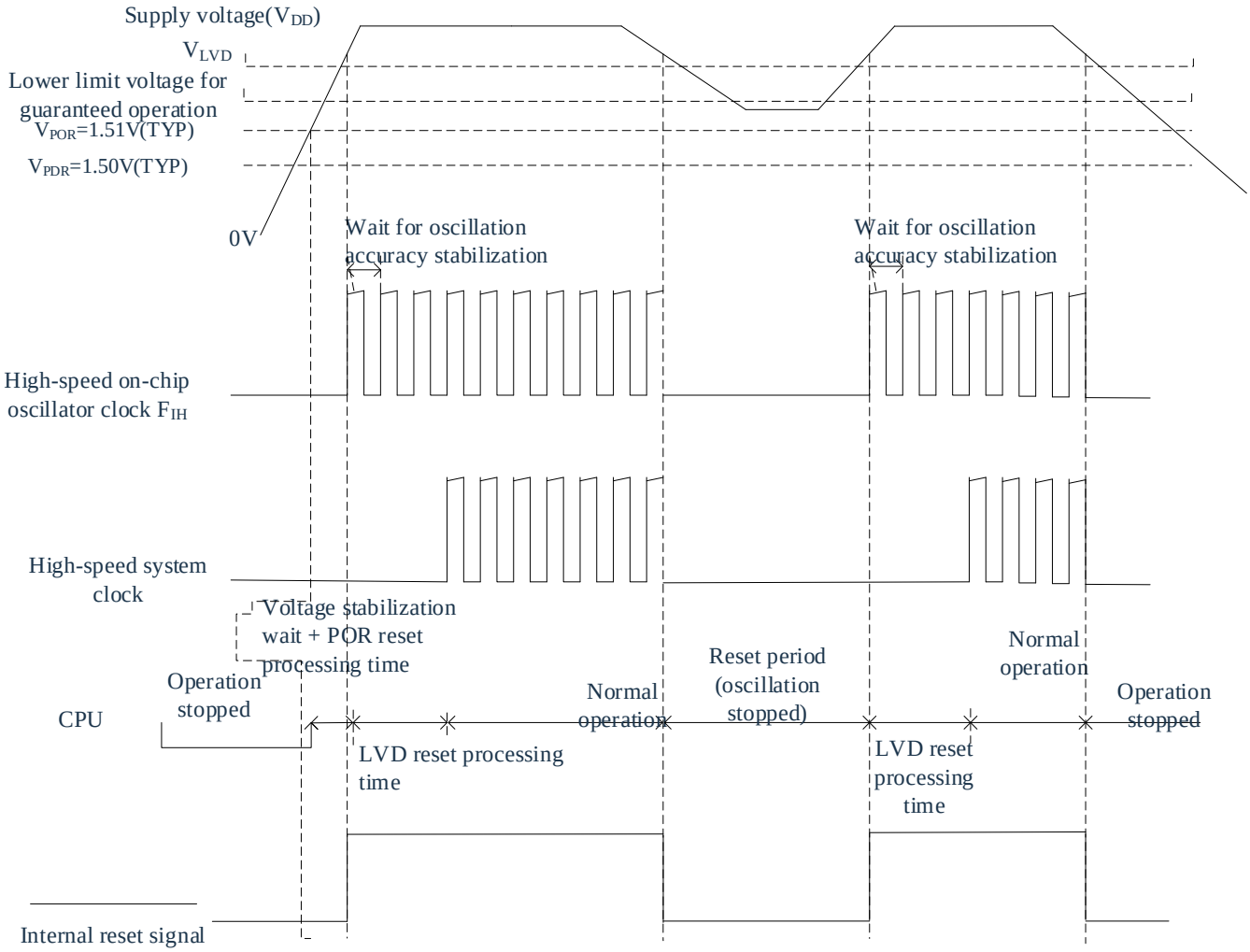
Note 4: $V_{LVDH}$, $V_{LVDL}$ : LVD detection voltage

Note 5: $V_{POR}$: POR power supply rise detection voltage

Note 6: $V_{PDR}$ : POR power supply fall detection voltage

Figure 27-2 Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (3/3)

(3)LVD reset mode (option byte 000C1H: LVIMDS1 = 1, LVIMDS0 = 1)




Note 1: The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed on-chip oscillator clock.

Note 2: The time until normal operation starts includes the following LVD reset processing time after the LVD detection level ($V_{LVD}$) is reached as well as the voltage stabilization wait + POR reset processing time after the VPOR (1.51 V(TYP.)) is reached.

Note 3: When the power supply voltage is below the lower limit for operation and the power supply voltage is then restored after an internal reset is generated only by the voltage detection circuit (LVD), the following LVD reset processing time is required after the LVD detection level ($V_{LVD}$) is reached.

Note 4: $V_{LVDH}$, $V_{LVDL}$: LVD detection voltage

Note 5: $V_{POR}$: POR supply voltage rise detection voltage

Note 6: $V_{PDR}$: POR supply voltage fall detection voltage

Note 7: When the LVD interrupt mode is selected (option byte 000C1H: LVIMD1 = 0, LVIMD0 = 1), the time until normal operation starts after power is powered on is the same as the time specified in Figure 27-2 (3/3).

# Chapter 28 Voltage Detection Circuit

## 28.1Functions of Voltage Detection Circuit

The voltage detection circuit sets the operating mode and detection voltage ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$) by option byte (000C1H). The voltage detection circuit (LVD) has the following functions.

- The internal reset or internal interrupt signal is generated by comparing the supply voltage ($V_{DD}$) with the detection voltage ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$).
- The detection voltage of the supply voltage ($V_{LVDH}$, $V_{LVDL}$) can be selected from 12 detection levels by means of option bytes (see “Chapter 31 Option Byte”).
- It can also operate in deep sleep mode.
- When the supply voltage rises, the reset state must be maintained by the voltage detection circuit or external reset before the supply voltage reaches the operating voltage range shown in the AC characteristics of the datasheet; when the supply voltage falls, the reset state must be set by the deep sleep mode transfer, voltage detection circuit or external reset before the supply voltage falls below the operating voltage range. The operating voltage range depends on the setting of the user option byte.

(a) Interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0 of option byte)

Two detection voltages ($V_{LVDH}$, $V_{LVDL}$) are selected by the option byte 000C1H. The high voltage detection level ($V_{LVDH}$) is used to release the reset or generate an interrupt, and the low voltage detection level ($V_{LVDL}$) is used to generate a reset.

(b) Reset mode (LVIMDS1, LVIMDS0=1, 1 for option byte)

A detection voltage ($V_{LVD}$) selected by option byte 000C1H is used to generate or release the reset.

(c) Interrupt mode (option byte of LVIMDS1, LVIMDS0=0, 1)

A detection voltage ($V_{LVD}$) selected by option byte 000C1H is used to generate an interrupt or to release the reset. In each mode, the following interrupt signals and internal reset signals are generated.

| Mode | Interrupt & reset mode | Reset mode | Interrupt mode |
|---|---|---|---|
| Configuration | (LVIMDS1, LVIMDS0=1, 0) | (LVIMDS1, LVIMDS0=1, 1) | (LVIMDS1, LVIMDS0=0, 1) |
| Operation process | When the operating voltage drops, an interrupt request signal is generated when $V_{DD} < V_{LVDH}$ is detected; when $V_{DD} < V_{LVDL}$ is detected, an internal reset is generated. When $V_{DD} \geq V_{LVDH}$ is detected, an internal reset is released. | When $V_{DD} \geq V_{LVD}$ is detected, an internal reset is released; when $V_{DD} < V_{LVD}$ is detected, an internal reset is generated. | After a reset occurs, an internal reset state of LVD continues until $V_{DD} \geq V_{LVD}$. When $V_{DD} \geq V_{LVD}$ is detected, an internal reset of LVD is released. After the internal reset of LVD is released, if $V_{DD} < V_{LVD}$ or $V_{DD} \geq V_{LVD}$ is detected, then an interrupt request signal (INTLVI) is generated. |

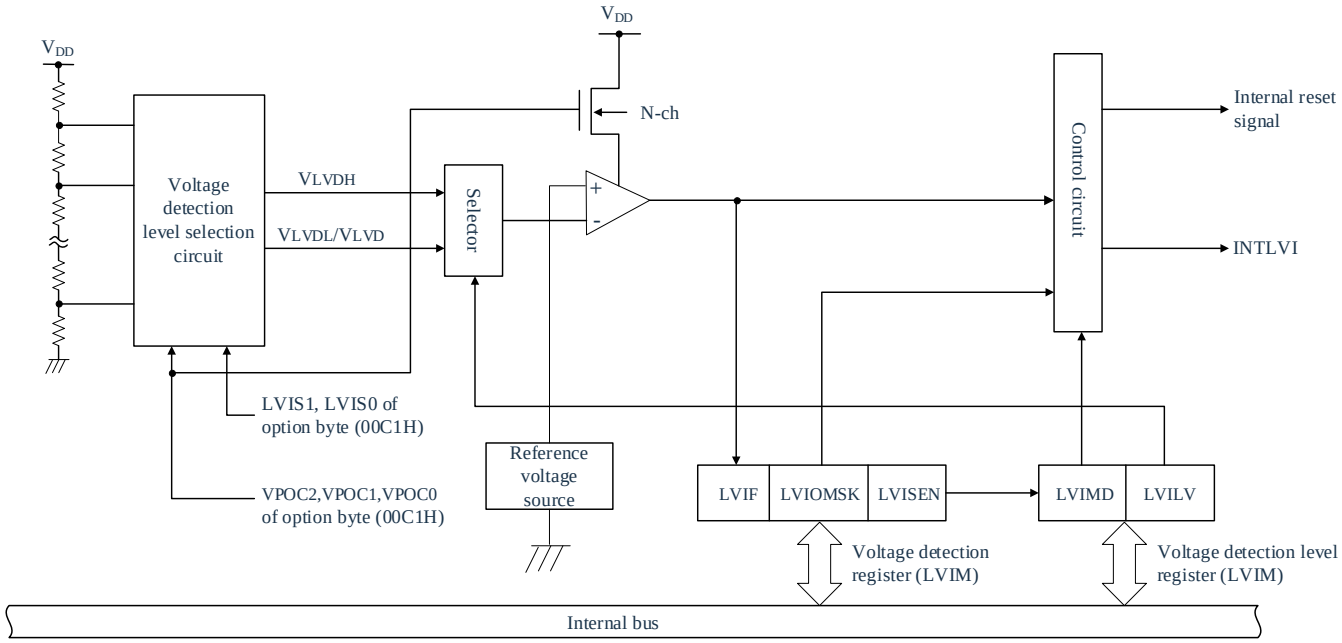
When the voltage detection circuit is in operation, it is possible to check whether the power supply voltage is greater than or less than the detection voltage by reading the voltage detection flag (LVIF: bit 0 of the voltage detection register (LVIM)).

If a reset occurs, bit 0 (LVIRF) of the reset control flag register (RESF) is set to “1”. For details of the RESF register, please refer to “Chapter 26 Reset Function”.

# 28.2Structure of Voltage Detection Circuit

The block diagram of the voltage detection circuit is shown in Figure 28-1.

Figure 28-1 Block diagram of voltage detection circuit




# 28.3Register Mapping

(Voltage detection base address = 0x4002_0441) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| LVIM | 0x000 | R/W | Voltage detection register | 0x0 |
| LVIS | 0x001 | R/W | Voltage detect level register | 0x0 |

# 28.4Registers for Controlling Voltage Detection Circuit

The voltage detection circuit is controlled by the following registers.

- Voltage detection register (LVIM)
- Voltage detection level register (LVIS)

### 28.4.1 Voltage Detection Register (LVIM)

This register is set to enable or disable overwriting of the voltage detection level register (LVIS), and to confirm the masking status of the LVD output. The LVIM register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes “00H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | LVISEN[Note1] | Enable/disable setting of voltage detection level register (LVIS)<br>0: Disable<br>1: Enable | - |
| 6:2 | - | Reserved | - |
| 1 | LVIOMSK | Mask status flag for LVD output<br>0: LVD output masking is invalid.<br>1: LVD output masking is valid. [Note2] | - |
| 0 | LVIF | Voltage detection flag<br>0: Supply voltage ($V_{DD}$) ≥ detection voltage ($V_{LVD}$) or LVD is OFF.<br>1: Supply voltage ($V_{DD}$) < detection voltage ($V_{LVD}$) | - |

Note 1: It can only be set when the interrupt & reset mode is selected (the LVIMDS1 bit and the LVIMDS0 bit of the option bytes are “1” and “0” respectively), the initial value cannot be changed in other modes.

Note 2: Only when the interrupt & reset mode is selected (the LVIMDS1 bit and the LVIMDS0 bit of the option byte are “1” and “0” respectively). The LVIOMSK bit automatically changes to “1” during the following periods, masking the reset or interrupt generated by LVD.

- When LVISEN=1
- Waiting time from the occurrence of LVD interrupt to the stabilization of LVD detection voltage
- Waiting time from changing the value of the LVILV bit until the LVD detection voltage stabilizes.

## 28.4.2 Voltage Detection Level Register (LVIS)

This is a register that sets the voltage detection level.

The LVIS register is set by an 8-bit memory manipulation instruction. After generating a reset signal, the value of this register changes to “00H/01H/81H” Note1.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | LVIMD Note1 | Operation mode of voltage detection<br>0: Interrupt mode<br>1: Reset mode | 0 |
| 6:1 | - | Reserved | - |
| 0 | LVILV Note2 | LVD detection level<br>0: High voltage detection level ($V_{LVDH}$)<br>1: Low voltage detection level ($V_{LVDL}$ or $V_{LVD}$) | 0 |

Note 1: The reset value varies depending on the setting of the reset source and option bytes. When an LVD reset occurs, this register is not cleared to “00H”.

When a reset other than LVD occurs, the values of this register are as follows:

• LVIMDS1, LVIMDS0 of option bytes =1, 0: 00H

• LVIMDS1, LVIMDS0 of option bytes =1, 1: 81H

• LVIMDS1, LVIMDS0 of option bytes =0, 1: 01H

Note 2: Write “0” only if interrupt & reset mode is selected (the LVIMDS1 bit and the LVIMDS0 bit for option bytes are “1” and “0” respectively). In other cases, it cannot be set. In interrupt & reset mode, value substitution is performed automatically by generating a reset or interrupt.

Note 3: To rewrite the LVIS registers, it must be done in accordance with the steps in Figure 28-6 and Figure 28-7.

Note 4: Option byte 000C1H selects the mode of operation of the LVD and the detection voltage ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$) for each mode. For details of the user option byte (000C1H/010C1H), refer to “Chapter 31 Option Byte”.

# 28.5Operation of Voltage Detection Circuit

## 28.5.1 When Used as Reset Mode

The operation mode (reset mode (LVIMDS1, LVIMDS0=1, 1)) and the detection voltage ($V_{LVD}$) are set via the option byte 000C1H. If the reset mode is set, operation starts with the following initial settings.

- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to “0” (disable rewriting the voltage detection level register (LVIS))
- Set the initial value of the voltage detection level register (LVIS) to “81H”. Set bit7(LVIMD) to “1” (reset mode). Set bit0 (LVILV) to “1” (voltage detection level: $V_{LVD}$).

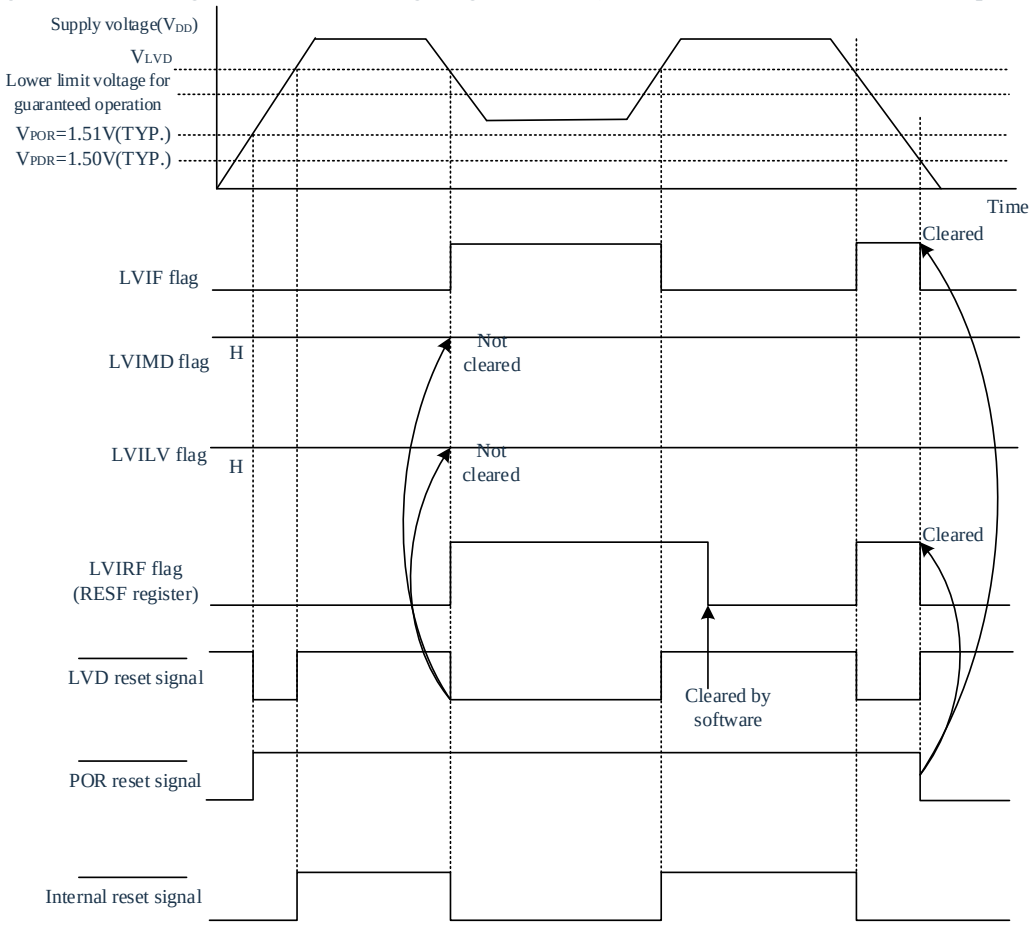
- Operation of LVD reset mode

When the power is turned on, the reset mode (LVIMDS1, LVIMDS0=1, 1 of the option byte) keeps the internal reset state of LVD until the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$). If the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$), the internal reset is released.

When the operating voltage falls, an internal reset of LVD is generated if the supply voltage ($V_{DD}$) is below the voltage detection level ($V_{LVD}$)

The timing of the internal reset signal generation for LVD reset mode is shown in Figure 28-2.

Figure 28-2 Timing of internal reset signal generation (LVIMDS1, LVIMDS0=1, 1 for option byte)




Note 1: $V_{POR}$: POR power supply rise detection voltage

Note 2: $V_{PDR}$: POR power supply fall detection voltage

# 28.6When Used As Interrupt Mode

The operation mode (interrupt mode (LVIMDS1, LVIMDS0=0, 1)) and the detection voltage ($V_{LVD}$) are set via the option byte 000C1H. If the interrupt mode is set, operation starts with the following initial settings.

- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to “0” (disables rewriting the voltage detection level register (LVIS)).
- Set the initial value of the voltage detection level register (LVIS) to “01H”. Set bit7 (LVIMD) to “0” (interrupt mode). Set bit0(LVILV) to “1” (voltage detection level: VLVD).

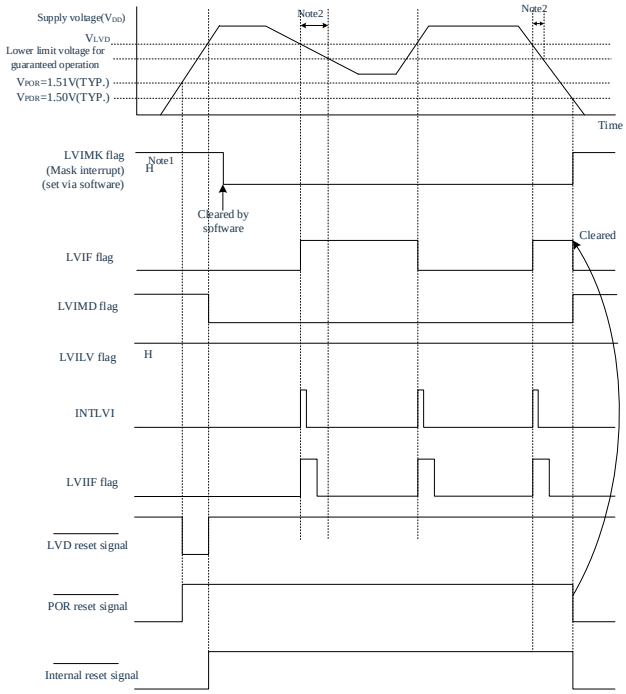
- Operation of LVD interrupt mode

After generating a reset, the interrupt mode (LVIMDS1, LVIMDS0 of the option byte =0, 1) maintains the internal reset state of the LVD until the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$). If the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$), the internal reset of the LVD is released.

If the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$) after the internal reset of the LVD is released, an interrupt request signal (INTLVI) of the LVD is generated. When the operating voltage drops, it must be set to the reset state by deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the datasheet. When restarting operation, it must be verified that the supply voltage has returned to the operating voltage range.

The timing of the interrupt request signal generation for LVD interrupt mode is shown in Figure 28-3.

Figure 28-3 Timing of interrupt signal generation (LVIMDS1, LVIMDS0 of option byte =0, 1)




Note 1: After generating a reset signal, the LVIMK flag changes to “1”.

Note 2: When the operating voltage drops, it must be reset by deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the data sheet. When restarting operation, it must be verified that the supply voltage returns to the operating voltage range.

Remark: $V_{POR}$: POR power supply rise detection voltage, $V_{PDR}$: POR power supply fall detection voltage

## 28.7When Used as Interrupt & Reset Mode

The operation mode (interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0)) and the detection voltage ($V_{LVDH}$, $V_{LVDL}$) are set via the option byte 000C1H.

If the interrupt & reset mode is set, the operation starts with the following initial settings.

- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to “0” (disables rewriting the voltage detection level register (LVIS)).
- Set the initial value of the voltage detection level register (LVIS) to “00H”. Set bit7 (LVIMD) to “0” (interrupt mode). Set bit0(LVILV) to “0” (high voltage detection level: $V_{LVDH}$).

- Operation of LVD interrupt & reset mode

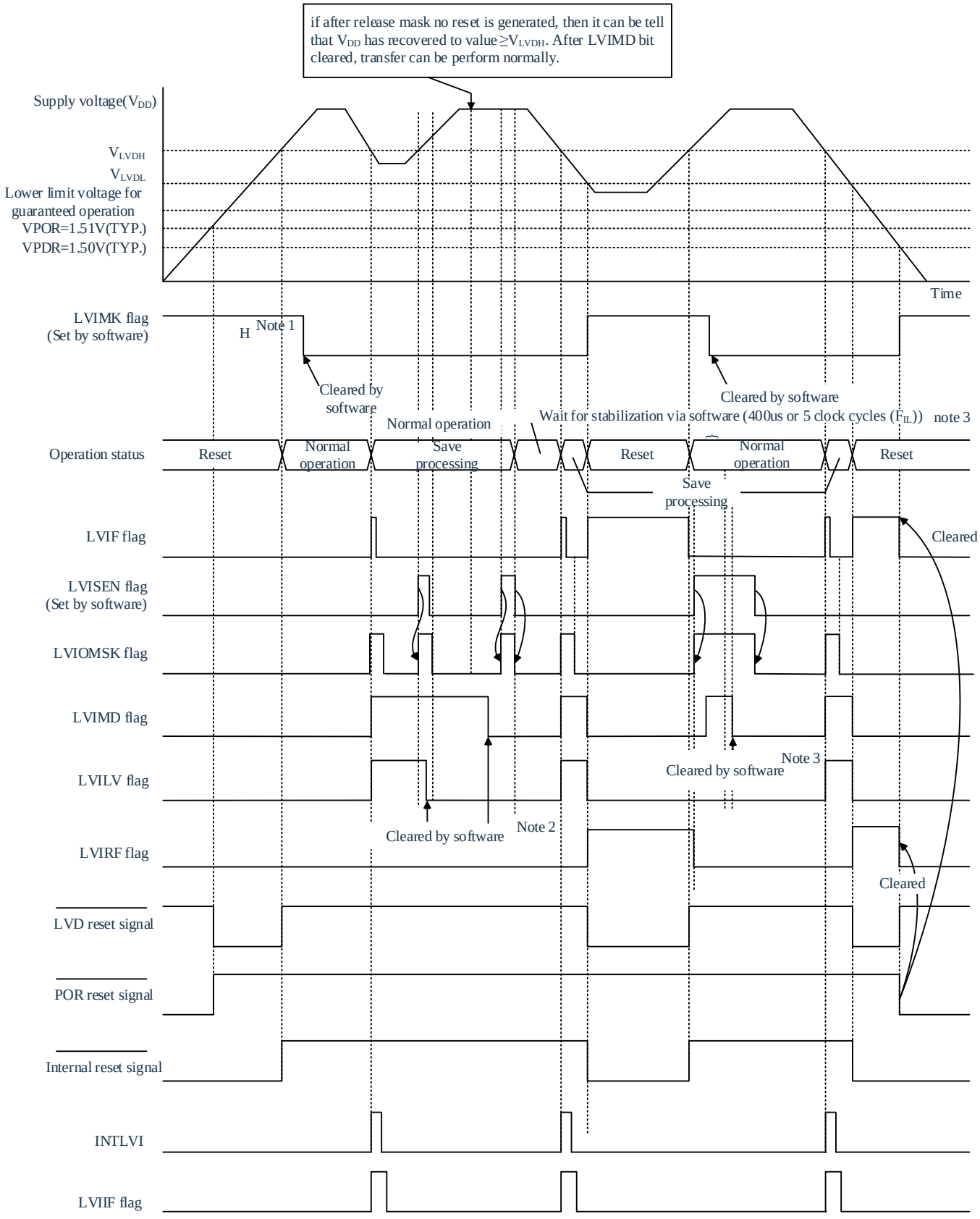
When power is turned on, the interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0 of the option byte) maintains the internal reset state of the LVD until the power supply voltage ($V_{DD}$) exceeds the high voltage detection level ($V_{LVDH}$). If the supply voltage ($V_{DD}$) exceeds the high voltage detection level ($V_{LVDH}$), the internal reset is released.

When the operating voltage drops, if the supply voltage ($V_{DD}$) is below the high voltage detection level ($V_{LVDH}$), an interrupt request signal (INTLVI) is generated for the LVD and any stacking process can be performed. After that, if the supply voltage ($V_{DD}$) is below the low voltage detection level ($V_{LVDL}$), an internal reset of the LVD is generated. However, after INTLVI occurs, no interrupt request signal is generated even if the supply voltage ($V_{DD}$) returns to the high voltage detection voltage ($V_{LVDH}$) or higher without falling below the low voltage detection voltage ($V_{LVDL}$).

When using LVD interrupt & reset mode, you must follow “Figure 28-6: Setting procedure for confirmation /reset of operating voltage” and “Figure 28-7: Initial setting procedure for interrupt & reset mode”.

The timing of the internal reset signal and interrupt signal generation in LVD interrupt & reset mode is shown in Figure 28-4.

Figure 28-4 Reset & interrupt signal generation timing (LVIMDS1, LVIMDS0=1, 0) (1/2)




Note 1: After the reset signal is generated, the LVIMK flag becomes “1”.

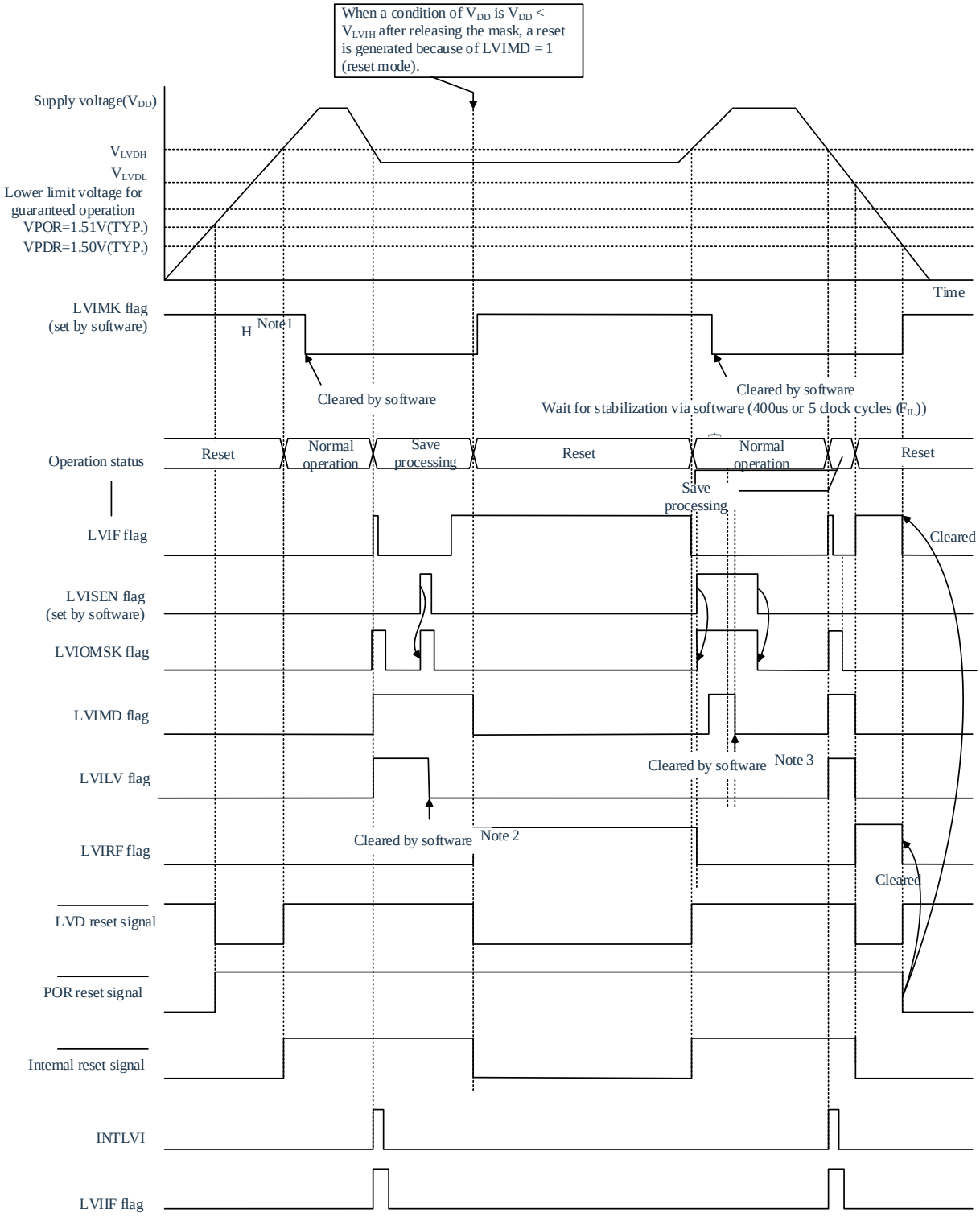
Note 2: When using the interrupt & reset mode, you must follow “Figure 28-6: Setting procedure for confirmation /reset of operating voltage” after an interrupt occurs.

Note 3: When using the interrupt&reset mode, you must follow the steps in “Figure 28-7: Initial setting procedure for interrupt & reset mode” after the reset is released.

Note 4: $V_{POR}$: POR power supply rise detection voltage

$V_{PDR}$: POR power supply fall detection voltage

Figure 28-5 Reset & interrupt signal generation timing (LVIMDS1, LVIMDS0=1, 0) (2/2)




Note 1: The LVIMK flag is set to “1” by reset signal generation.

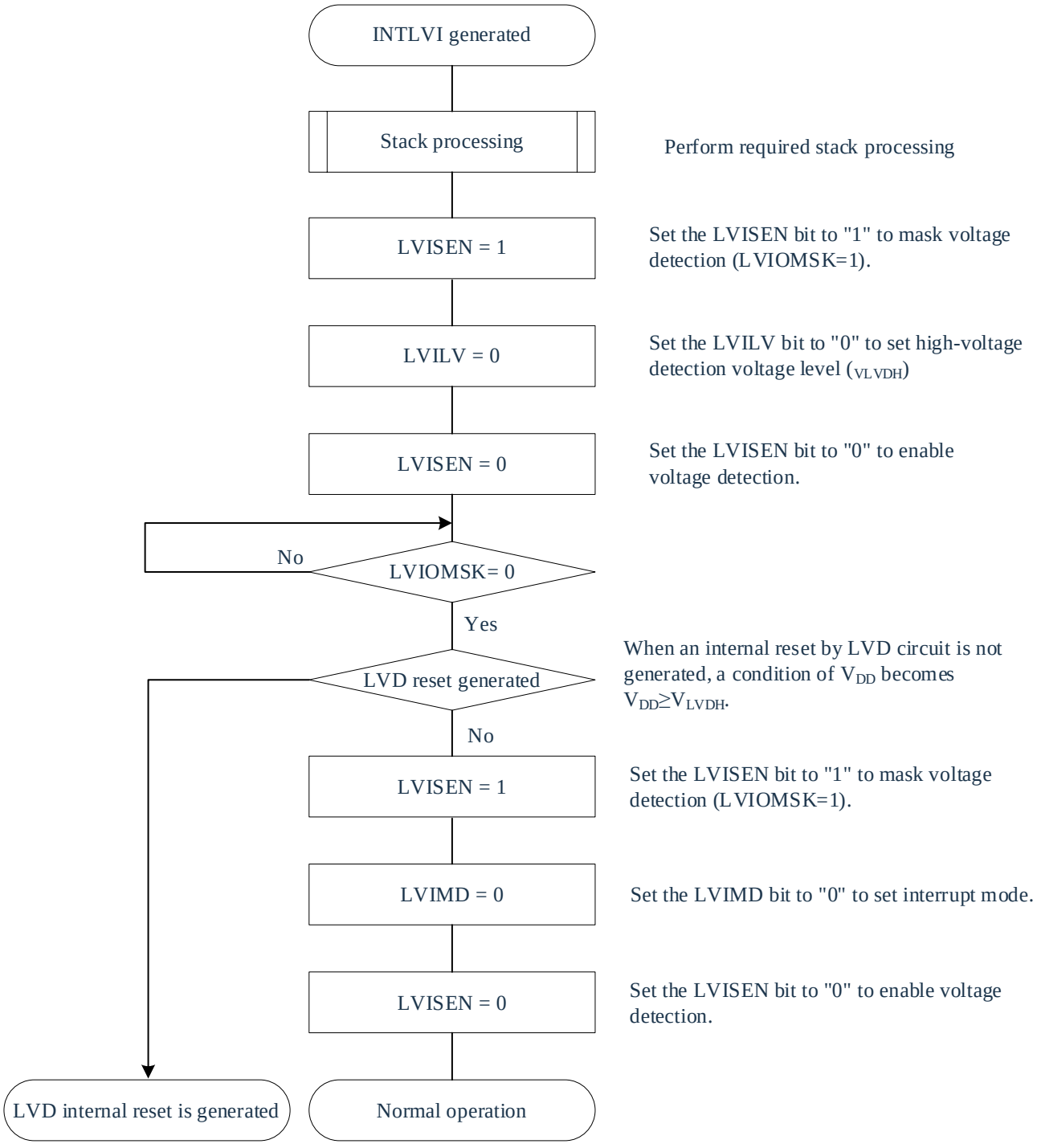
Note 2: When using the interrupt & reset mode, you must follow “Figure 28-6: Setting procedure for confirmation /reset of operating voltage” after an interrupt occurs.

Note 3: When using the interrupt&reset mode, you must follow the steps in “Figure 28-7: Initial setting procedure for interrupt & reset mode” after the reset is released.

Remark: $V_{POR}$: POR power supply rise detection voltage

$V_{PDR}$: POR power supply fall detection voltage

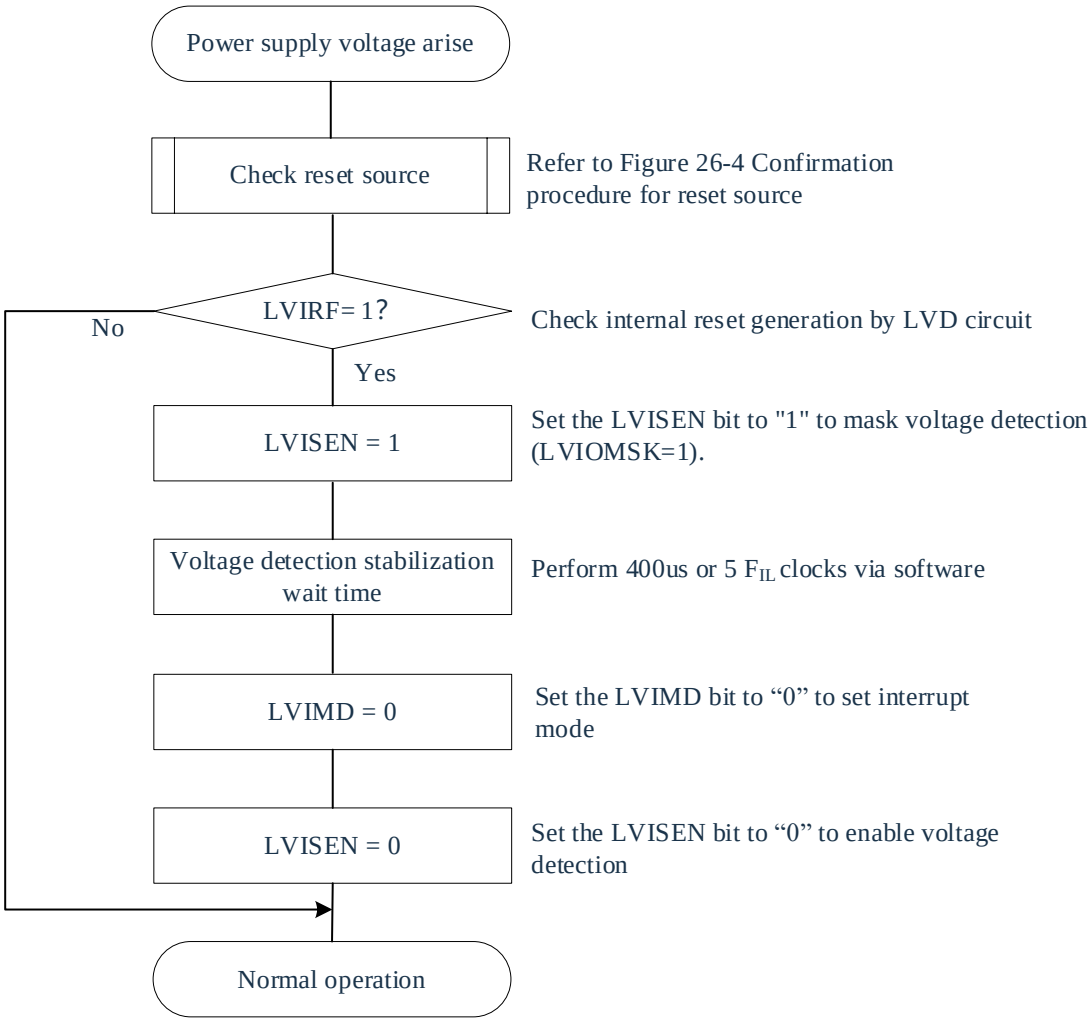
Figure 28-6 Setting procedure for confirmation/reset of operating voltage




If the interrupt & reset mode is set (LVIMDS1, LVIMDS0=1, 0), it will take 400us or 5 $F_{IL}$ clocks for the voltage detection to stabilize after the LVD reset (LVIRF=1) is released. The LVIMD bit must be cleared to “0” for initialization after waiting for the voltage detection to stabilize. The LVISEN bit must be set to “1” during the count of the voltage detection stabilization time and when rewriting the LVIMD bit to block the generation of resets or interrupts generated by LVD.

The initial setting procedure for interrupt & reset mode is shown in Figure 28-7.

Figure 28-7 nitial setting procedure for interrupt & reset mode




Remark: $F_{IL}$: Low-speed on-chip oscillator clock frequency

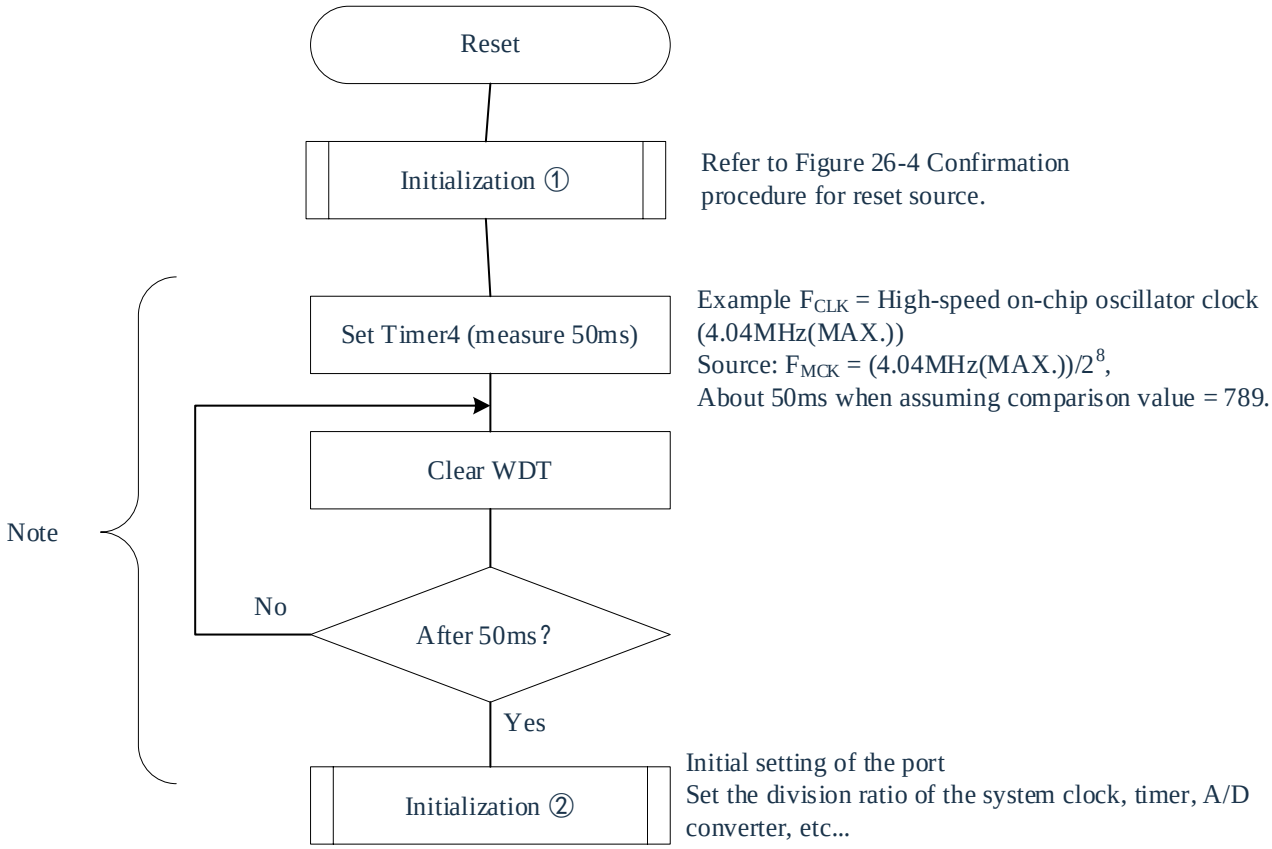
# 28.8Cautions for Voltage Detection Circuit

(1) Voltage fluctuation when power is supplied

In a system where the supply voltage ($V_{DD}$) fluctuates for a certain period in the vicinity of the LVD detection voltage, the system may be repeatedly reset and released from the reset status. In this case, the time from release of reset to the start of the operation of the microcontroller can be arbitrarily set by taking the following action.

After releasing the reset signal, wait for the supply voltage fluctuation period of each system by means of a software counter that uses a timer, and then initialize the ports.

Figure 28-8 Example of software processing when the supply voltage fluctuation near the LVD detection voltage does not exceed 50ms

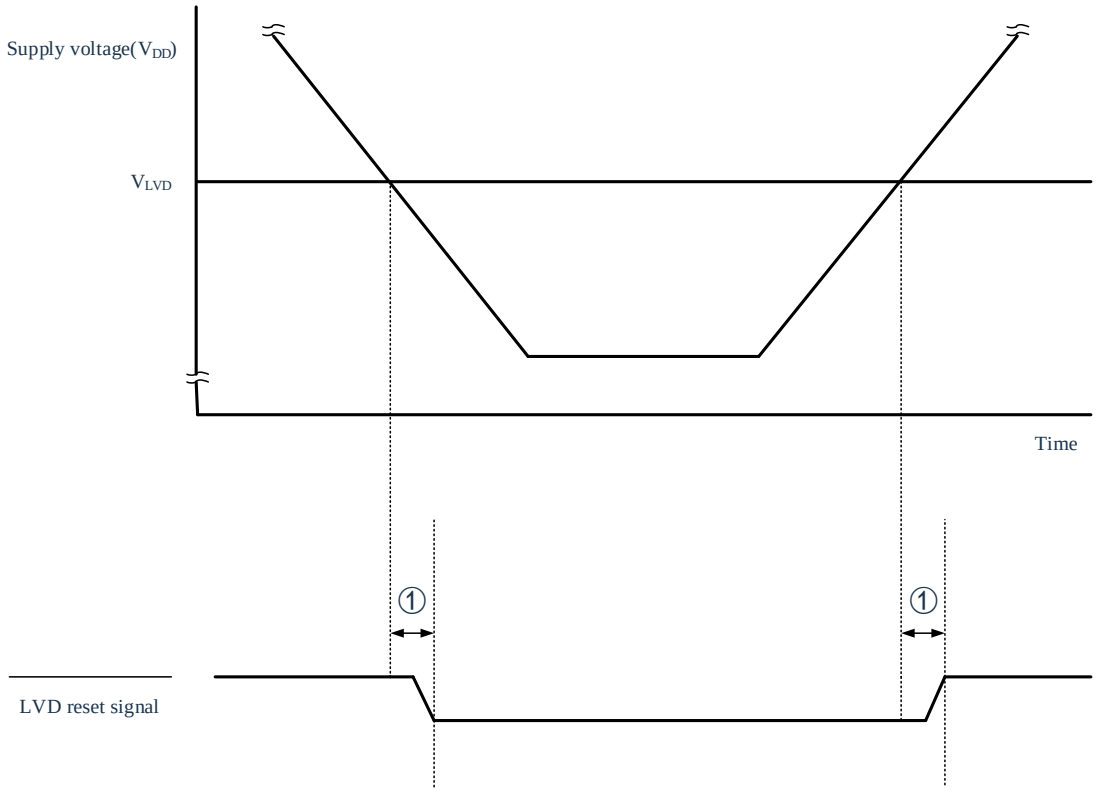



Note 1: If the reset occurs again during this period, it is not switched to initialization processing ②.

(2) Delay from the time LVD reset source is generated until the time LVD reset has been generated or released

A delay occurs from the time the supply voltage ($V_{DD}$) < LVD detection voltage ($V_{LVD}$) is met to the time the LVD reset is generated. Similarly, a delay occurs from the time the LVD detection voltage ($V_{LVD}$) ≤ the supply voltage ($V_{DD}$) to the time the LVD reset is released (see Figure 28-9).

Figure 28-9 Delay from generation of LVD reset source to generation or release of LVD reset




① Detection delay (300us(MAX.))

(3) When the power is turned on with LVD set to OFF

When LVD is set to OFF, an external reset must be performed using the RESETB pin.

When performing an external reset, the RESETB pin must be input low for at least 10us. If an external reset is performed while the supply voltage is rising, the power must be turned on after a low level is input to the RESETB pin, and must be held low for at least 10us within the operating voltage range shown in the AC characteristics of the datasheet, followed by a high level.

(4) When LVD is set to OFF in LVD interrupt mode and the operating voltage drops

If the operating voltage drops when LVD is set to OFF and LVD interrupt mode is set, it must be reset by deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the data sheet. When restarting operation, it is necessary to verify that the supply voltage is restored in the operating voltage range.

# Chapter 29 Safety Functions

## 29.1 Overview of Safety Functions

The following safety functions are provided in the CMS32M67xx to comply with the IEC60730 and IEC61508 safety standards.

These functions enable the microcontroller to self-diagnose abnormalities and stop operating if an abnormality is detected.

(1) Flash memory CRC operation function (high-speed CRC, universal CRC)

This detects data errors in the flash memory by performing CRC operations. Two CRC functions are provided in the CMS32M65xx that can be used according to the application or purpose of use.

- "High-speed CRC" … The CPU can be stopped and a high-speed check executed on its entire code flash memory area during the initialization routine.
- "Universal CRC" … This can be used for checking various data in addition to the code flash memory area while the CPU is running.

(2) SFR guard function

This prevents SFRs from being rewritten when the CPU freezes.

(3) Frequency detection function

This function allows a self-check of the CPU/peripheral hardware clock frequencies using the general-purpose timer unit.

(4) A/D test function

This is used to perform a self-check of the A/D converter by performing A/D conversion of the A/D converter's positive and negative reference voltages, analog input channel (ANI), temperature sensor output voltage, and internal reference voltage.

(5) Digital output signal level detection function for input/output ports

When the input/output port is in output mode, the output level of the pin can be read.

## 29.2Register Mapping

The safety functions use the following registers for each function.

| Register name | Function |
|---|---|
| • Flash CRC control register (CRC0CTL)<br>• Flash CRC operation result register (PGCRCL) | Flash CRC operation function<br>(High-speed CRC) |
| • CRC input register (CRCIN)<br>• CRC data register (CRCD) | CRC calculation function<br>(Universal CRC) |
| • Special SFR protection control register (SFRGD) | SFR guard function |
| • Timer input/output select register 0 (TIOS0) | Frequency detection function |
| • A/D test register (CON2) | A/D test function |
| • Port mode selection register (PMS) | Digital output signal level detection function for input/output pins |

(Flash memory CRC base address = 0x4002_1810) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CRC0CTL | 0x000 | R/W | Flash memory CRC control register | 0x0 |
| PGCRCL | 0x002 | R/W | Flash memory CRC operation result register | 0x0 |

(Universal CRC base address = 0x4004_32FA) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| CRCIN | 0x4004_33AC | R/W | Flash memory CRC control register | 0x0 |
| CRCD | 0x4004_32FA | R/W | Flash memory CRC operation result register | 0x0 |

(SFR base address = 0x4004_0478) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| SFRGD | 0x000 | R/W | SFR guard control register | 0x0 |

(Port control base address = 0x4004_087B) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| PMS | 0x000 | R/W | Port mode select register | 0x0 |

(UID base address = 0x0050_0894) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| UID0 | 0x000 | RO | Product unique ID bit [31:0] | - |
| UID1 | 0X004 | RO | Product unique ID bit [63:32] | - |
| UID2 | 0X008 | RO | Product unique ID bit [95:64] | - |
| UID3 | 0X00C | RO | Product unique ID bit [127:96] | - |

# 29.3Operation of Safety Functions

## 29.3.1 Flash CRC Operation Function (High-Speed CRC)

The IEC60730 standard mandates the checking of data in the flash memory, and recommends using CRC to do it. The high-speed CRC can be used to check the entire code flash memory area during the initialization routine.

The high-speed CRC performs an operation by reading 32-bit data per clock from the flash memory while stopping the CPU. This function therefore can finish a check in a shorter time (for example, 1820.44 μs@72 MHz with 128-KB flash memory Note1).

The CRC generator polynomial used complies with CRC-16-CCITT "$X^{16}+X^{12}+X^5+1$".

Bit31 - bit0 operates with MSB first priority.

Note 1: This is the system clock, the relationship between the system clock and the clock of flash CRC operation is 4 : 1, the system clock is 72MHZ, then the clock of flash CRC operation is 18MHZ.

Note 2: The operation result is different because the universal CRC operates in LSB first order.

Flash memory CRC control register (CRC0CTL)

This register is used to control the operation of the high-speed CRC, as well as to specify the operation range. The CRC0CTL register can be set by an 8-bit memory manipulation instruction. Reset signal generation clears this register to 00H.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | CRC0EN | High-speed CRC operation control<br>0: Operation stopped<br>1: Start the operation by executing the WFE instruction | 0 |
| 6 | CRCCHK124 | 124K operation range selection<br>0: The range is controlled by bits [5:0]<br>1: 00000H ~1EFFBH(124K-4byte) | 0 |
| 5:0 | FEA | High-speed CRC operation range<br>0000: 00000H ~1FFBH(8K-4byte)<br>0001: 00000H ~3FFBH(16K-4byte)<br>0010: 00000H ~5FFBH(24K-4byte)<br>0011: 00000H ~7FFBH(32K-4byte)<br>0100: 00000H ~9FFBH(40K-4byte)<br>0101: 00000H ~BFFBH(48K-4byte)<br>0110: 00000H ~DFFBH(56K-4byte)<br>0111: 00000H ~FFFBH(64K-4byte)<br>1000: 00000H ~11FFBH(72K-4byte)<br>1001: 00000H ~13FFBH(80K-4byte)<br>1010: 00000H ~15FFBH(88K-4byte)<br>1011: 00000H ~17FFBH(96K-4byte)<br>1100: 00000H ~19FFBH(104K-4byte)<br>1101: 00000H ~1BFFBH(112K-4byte)<br>1110: 00000H ~1DFFBH(120K-4byte)<br>1111: 00000H ~1FFFBH(128K-4byte) | 0x0 |

Note 1: Bit4~5 must be set to 0.

Note 2: Input the expected CRC operation result value to be used for comparison in the lowest 4 bytes of the flash memory. Note that the operation range will thereby be reduced by 4 bytes.

#### 29.3.1.1 Flash Memory CRC Operation Result Register L (PGCRCL)

This register is used to store the results of the high-speed CRC operation.

The PGCRCL register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

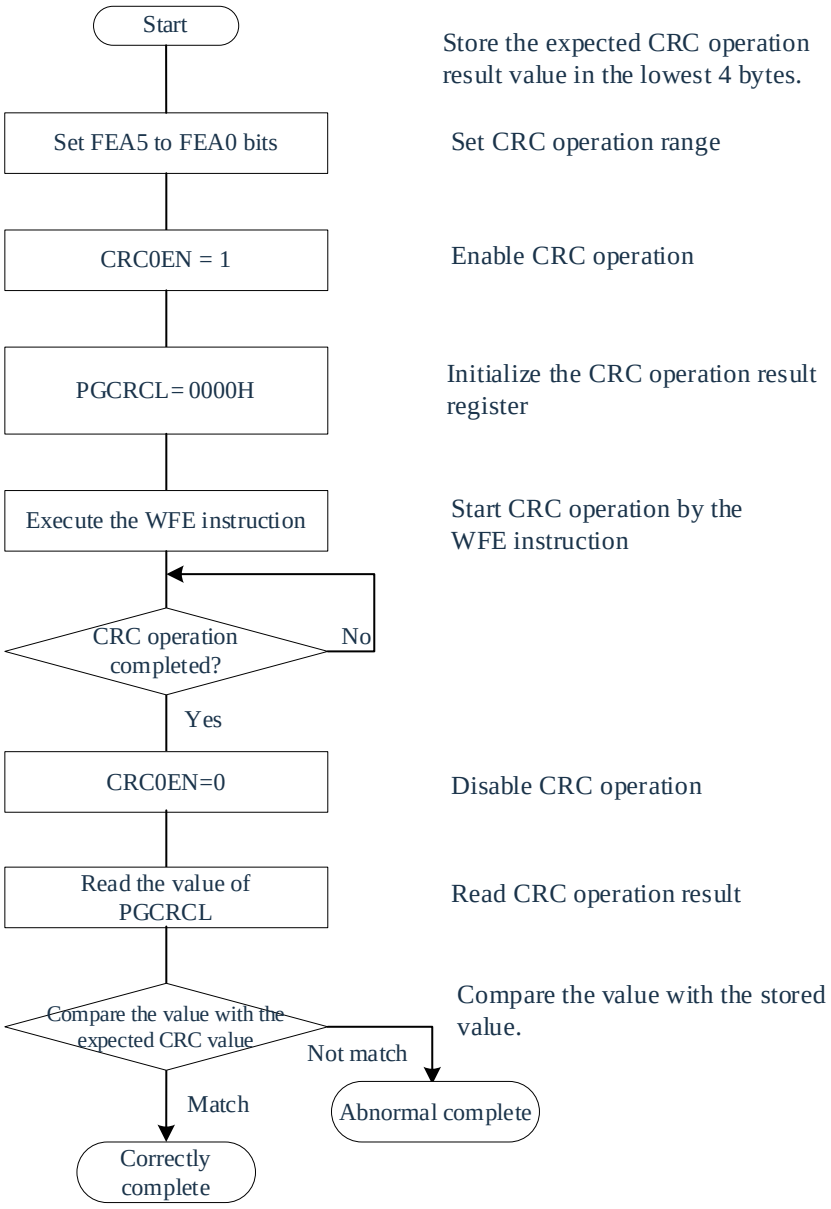
| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:0 | PGCRCL | Store the results of the high-speed CRC operation.<br>0000H ~FFFFH | 0x0 |

Note: The PGCRCL register can only be written if CRC0EN (bit 7 of the CRC0CTL register) = 1.

The flowchart of the flash memory CRC operation function (high-speed CRC) is shown in Figure 29-1.

＜Operation flow＞

Figure 29-1: Flow chart of flash CRC operation function (high-speed CRC)




Note 1: The CRC operation is executed only on the code flash.

Note 2: Store the expected CRC operation value in the area below the operation range in the code flash.

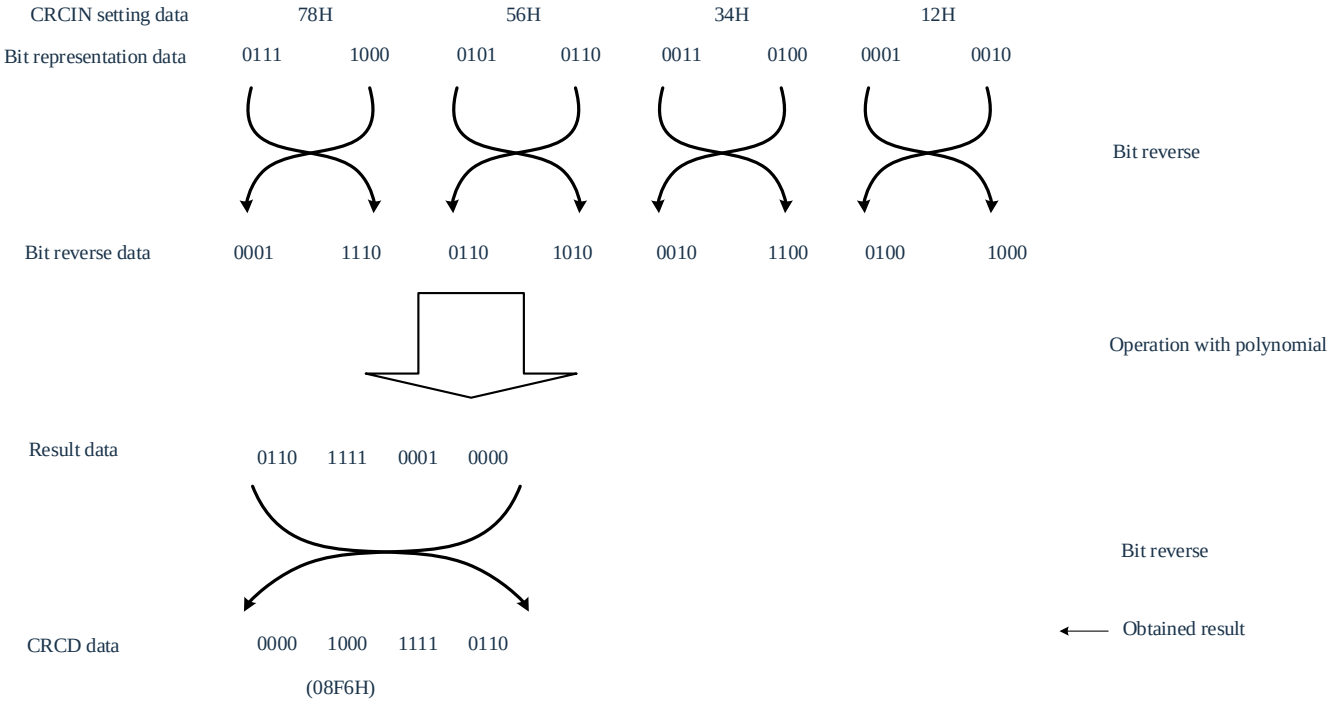
### 29.3.1.2 CRC Operation Function (Universal CRC)

In order to guarantee safety during operation, the IEC61508 standard mandates the checking of data even while the CPU is operating.

The universal CRC operation can be executed as a peripheral function while the CPU is operating. The universal CRC can be used for checking various data in addition to the code flash memory area. The data to be checked can be specified by using software (a user-created program).

The universal CRC operation can be executed in the main system clock operation mode as well as the subsystem clock operation mode.

The CRC generator polynomial used is "$X^{16}+X^{12}+X^5+1$" of CRC-16-CCITT. The data to be input is inverted in bit order and then calculated to allow for LSB-first communication. For example, if the data 12345678H is sent from the LSB, values are written to the CRCIN register in the order of 78H, 56H, 34H, and 12H, enabling a value of 08F6H to be obtained from the CRCD register. This is the result obtained by executing a CRC operation on the bit rows shown below, which consist of the data 12345678H inverted in bit order.




Note: Because the debugger rewrites the software break setting line to a break instruction during program execution, the CRC operation result differs if a software break is set in the CRC operation target area.

### 29.3.1.3 CRC Input Register (CRCIN)

The CRCIN register is an 8-bit register that is used to set the CRC operation data of general-purpose CRC. The possible setting range is 00H to FFH.

The CRCIN register can be set by an 8-bit memory manipulation instruction. Reset signal generation clears this register to 00H.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | CRCIN | Universal CRC data input 00H ~FFH | 0x0 |

### 29.3.1.4 CRC Data Register (CRCD)

This register is used to store the CRC operation result of the universal CRC. The setting range is 0000H to FFFFH.

After 1 clock of CPU/peripheral hardware clock ($F_{CLK}$) has elapsed from the time CRCIN register is written, the CRC operation result is stored to the CRCD register. The CRCD register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

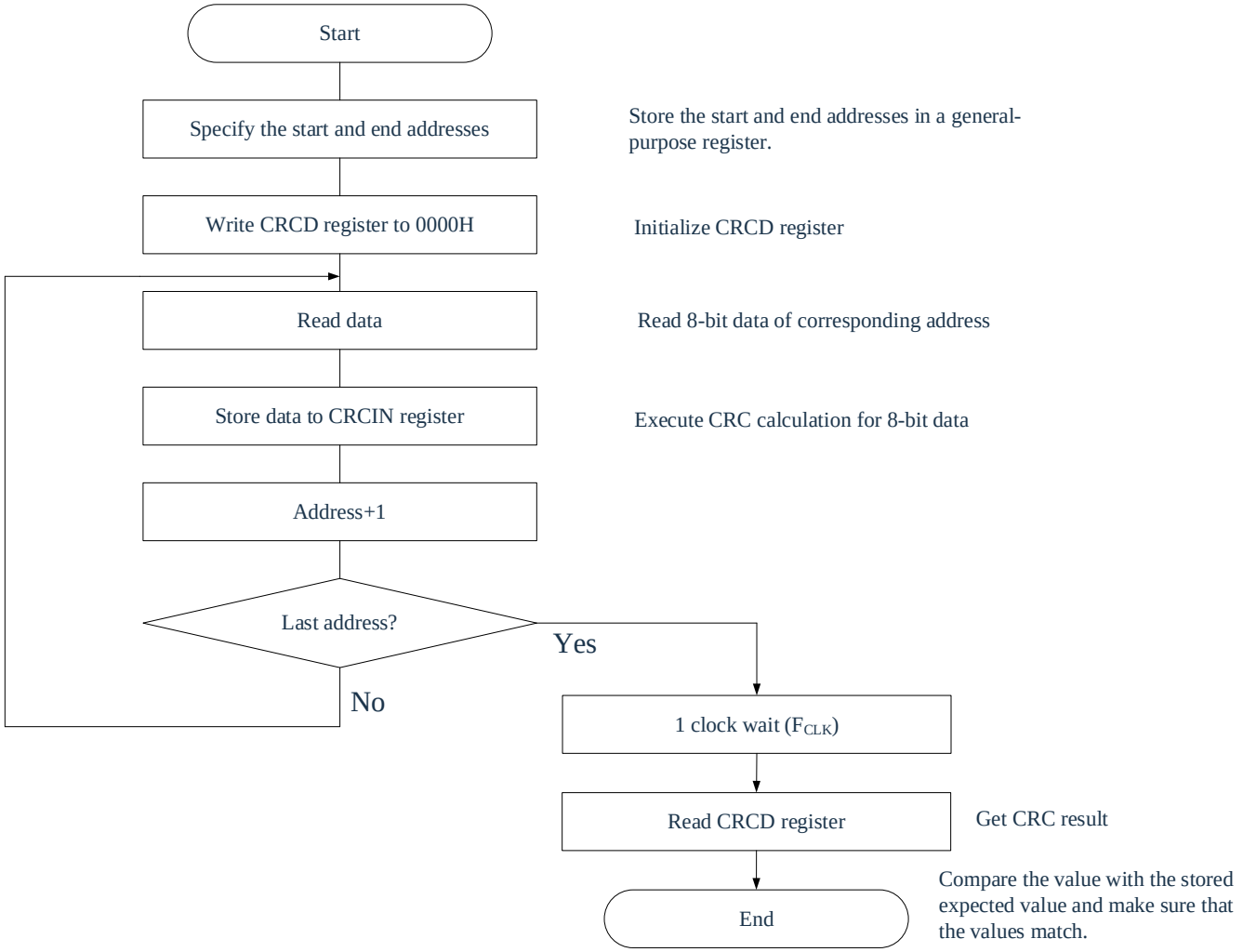
| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:0 | CRCD | Store the universal CRC operation results (0000H ~FFFFH) | 0x0 |

Note 1: To read the write value of the CRCD register, the CRCD register must be read before the CRCIN register is written.

Note 2: If a write operation to the CRCD register competes with the saving of an operation result, the write operation is ignored.

＜Operation flow＞

Figure 29-2 CRC operation function (universal CRC)



## 29.3.2 SFR Guard Function

In order to ensure safety during operation, the IEC61508 standard requires that even if the CPU is out of control, it is necessary to protect important SFR from being rewritten. The SFR protection function is used to protect data from the control registers of the comparator function, port function, interrupt function, clock control function, and voltage detection circuitry.

If the SFR protection function is set, the write operation of the protected SFR is invalid, but it can be read normally.

### 29.3.2.1 SFR Guard Control Register (SFRGD)

This register controls whether the SFR guard function is valid.

The GPORT bit and the GCSC bit are used for SFR guard function.

The SFRGD register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes “00H”.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | Reserved | - | - |
| 2 | GPORT | Protection of control registers for port functions<br>0: Invalid. Can read and write control registers for port functions.<br>1: Valid. Write operation of the control register of the port function is invalid, and it can be read.<br>[Protected SFR]PMxx, PUxx, PDxx, POMxx, PMCxx, PxxCFG[Note]. | 0 |
| 1 | - | Reserved | - |
| 0 | GCSC | Clock control function, voltage detection circuit control register protection<br>0: Invalid. Can read/write clock control function, voltage detection circuit control register.<br>1: Valid. Clock control function and write operation of the control register of the voltage detection circuit are invalid and read operation is enabled.<br>[Protected SFR] CSC, OSTS, CKC, PERx, OSMC, LVIM, LVIS. | 0 |

Note: Pxx (port register) is not protected.

### 29.3.3 Frequency Detection Function

The IEC60730 standard mandates checking that the oscillation frequency is correct.

By using the CPU/peripheral hardware clock frequency ($F_{CLK}$) and measuring the pulse width of the input signal to channel 1 of the Timer4, whether the proportional relationship between the two clock frequencies is correct can be determined.

Note that, however, if one or both clock operations are completely stopped, the proportional relationship between the clocks cannot be determined.

＜Clocks to be compared＞

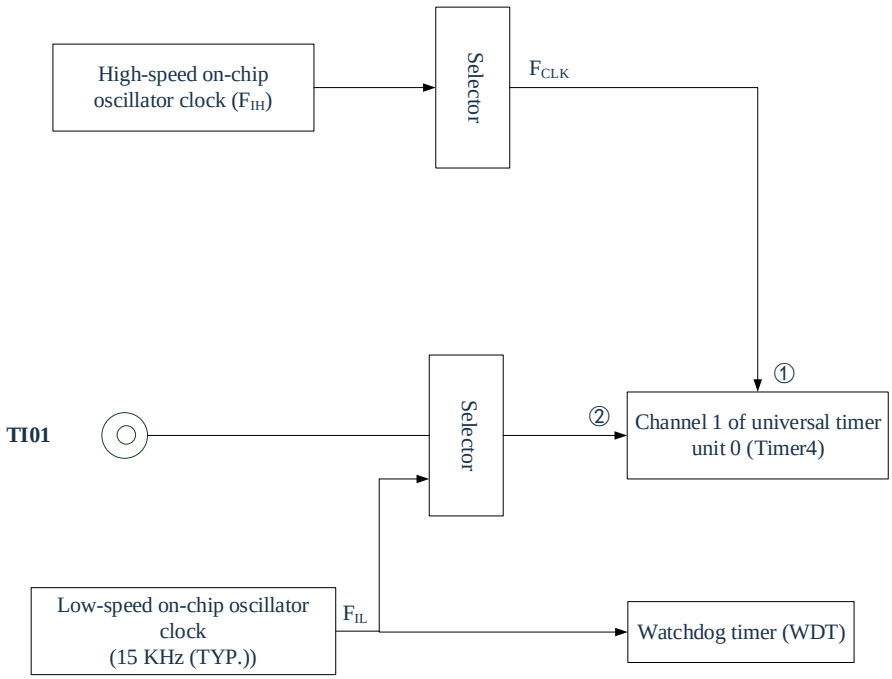
① CPU/peripheral hardware clock frequency ($F_{CLK}$):

- High-speed on-chip oscillator clock ($F_{IH}$)

② Input to Chanel 1 of the Timer4:

- Timer input to channel 1 (TI01)
- Low-speed on-chip oscillator clock ($F_{IL}$: 15kHz(TYP.))

Figure 29-3 Structure of frequency detection function




If the measurement result of the input pulse interval is an abnormal value, it can be judged as “clock frequency abnormality”. For the measurement method of the input pulse interval, refer to “5.7.4 Operation as input pulse interval measurement”.

Note: Can only be selected in the products incorporating the subsystem clock.

#### 29.3.3.1 Timer Input/Output Select Register 0 (TIOS0)

Refer to Section 5.2.11 for the register description.

### 29.3.4 A/D Test Function

The IEC60730 standard mandates testing the A/D converter. The A/D test function checks whether or not the A/D converter is operating normally by executing A/D conversions of the A/D converter's positive and negative reference voltages, analog input channel (ANI), temperature sensor output voltage, and the internal reference voltage.

The analog multiplexer can be checked using the following procedure.

1) Select the ANIx pin for A/D conversion using the CON2 register (ADCSWCHS = 00100).
2) Perform A/D conversion for the ANIx pin (conversion result 1-1).
3) Select the A/D converter's negative reference voltage for A/D conversion using the CON2 register (ADCSWCHS = 11000)
4) Perform A/D conversion of the negative reference voltage of the A/D converter (conversion result 2-1).
5) Select the ANIx pin for A/D conversion using the CON2 register (ADCSWCHS = 00100).
6) Perform A/D conversion for the ANIx pin (conversion result 1-2).
7) Select the A/D converter's positive reference voltage for A/D conversion using the CON2 register (ADCSWCHS = 10111)
8) Perform A/D conversion of the positive reference voltage of the A/D converter (conversion result 2-2).
9) Select the ANIx pin for A/D conversion using the CON2 register (ADCSWCHS = 00100).
10) Perform A/D conversion for the ANIx pin (conversion result 1-3).
11) Check that the conversion results 1-1, 1-2, and 1-3 are equal.
12) Check that the A/D conversion result 2-1 is all zero and conversion result 2-2 is all one.

Using the procedure above can confirm that the analog multiplexer is selected and all wiring is connected.

Note 1: If the analog input voltage is variable during A/D conversion in steps 1)~10) above, use another method to check the analog multiplexer

Note 2: The conversion results might contain an error. Consider an appropriate level of error when comparing the conversion results.

#### 29.3.4.1 A/D Test Register (CON2)

This register selects the positive (+) reference voltage, negative (–) reference voltage, analog input channel (ANxx), output voltage of the temperature sensor, and the internal reference voltage (1.45V) as the A/D conversion targets.

When using the A/D test function, specify the following settings:

- Select negative reference voltage as the target of A/D conversion for zero-scale measurement.
- Select positive reference voltage as the target of A/D conversion for full-scale measurement.

Please refer to Section 19.5.2 for A/D registers and descriptions.

## 29.3.5 Digital Output Signal Level Detection Function for Input/Output Pins

The IEC60730 standard mandates confirming that the I/O functions are normal.

Input/Output Pin Digital Output Signal Level Detection Function reads the digital output level of a pin when the pin is in output mode.

### 29.3.5.1 Port Mode Select Register (PMS)

This register selects whether to read the value of the port's output latch or the output level of the pin when the pin is in output mode (PMmn bit of the Port Mode Register (PMm) is "0").

The PMS register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "00H".

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:1 | - | Reserved | - |
| 0 | PMS0 | Selection of reading data when the pin is in output mode<br>0: Read the value of the Pmn register.<br>1: Read the digital output level of the pin. | 0 |

Note: m=0~5, n=0~7.

### 29.3.6 Product Unique ID Register

The unique ID of the product is perfect for:

Used as a serial number (e.g. USB character serial number or other terminal applications).

Used as a password, this unique ID is used in conjunction with a software encryption and decryption algorithm when writing flash memory to improve the security of the code in the flash memory.

Used to activate a bootstrap process with a safety mechanism

The reference number provided by the 128-bit product unique ID is unique to any microcontroller in any case. Under any circumstances, the user cannot modify this ID.

Product unique ID register 0 (UID0)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | - | Product unique ID register bit [31:0], the value of which is programmed at the factory. | - |

Product unique ID register 1 (UID1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | - | Product unique ID register bit [63:32], the value of which is programmed at the factory. | - |

Product unique ID register 2 (UID2)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | - | Product unique ID register bit [95:64], the value of which is programmed at the factory. | - |

Product unique ID register 3 (UID3)

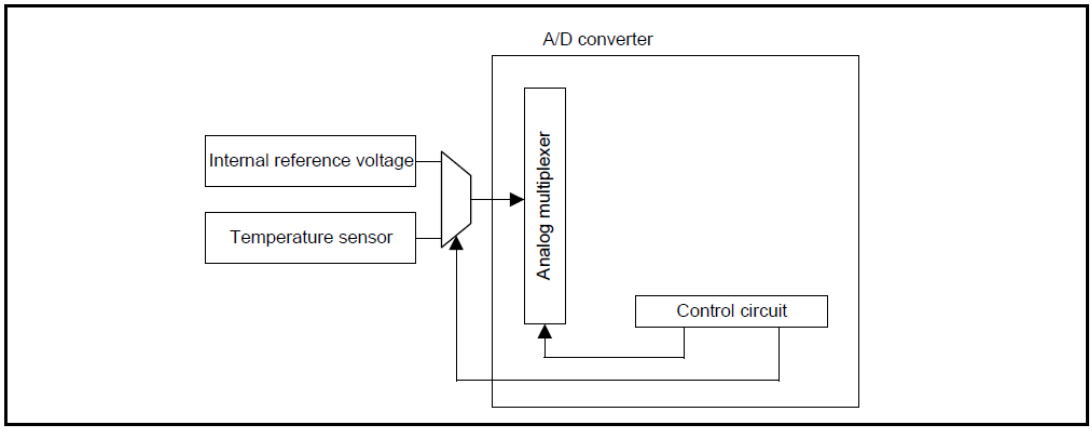
| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:0 | - | Product unique ID register bit [127:96], the value of which is programmed at the factory. | - |

# Chapter 30 Temperature Sensor

## 30.1Functions of Temperature Sensors

The on-chip temperature sensor measures and monitors the core temperature of the product, thus ensuring reliable operation of the product. The voltage output by the temperature sensor is proportional to the core temperature, and there is a linear relationship between the voltage and temperature. Its output voltage is supplied to the ADC for conversion. Figure 30-1 shows a block diagram of a temperature sensor.

Figure 30-1 Temperature sensor block diagram




## 30.2Register Mapping

RO: Read only, WO: Write only, R/W: Read/Write

| Register | Address | R/W | Description | Reset value |
|---|---|---|---|---|
| TSN25 | 0x0050066C | RO | Temperature Sensor Calibration Data Register | - |

## 30.3Temperature Sensor Register

### 30.3.1 Temperature Sensor Calibration Data Register TSN25

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 15:12 | - | Reserved | - |
| 11:0 | TSN25 | Calibration data, automatically loaded at power on or reset startup, and each chip has its own calibration data. | - |

## 30.4Instructions for Using Temperature Sensors

The temperature (T) is proportional to the sensor voltage output (Vs), so the temperature is calculated as follows:

T = (Vs - V1) / slope + 25℃

T: Measured temperature (℃)

Vs: Output voltage of the temperature sensor at temperature measurement (V)

V1: Voltage output at 25℃ measured by temperature sensor (V)

Slope: Temperature slope of the temperature sensors(V/℃), slope = -3.5 mV/℃

Remark: Temperature sensors have low accuracy and are not recommended for use in applications where high accuracy is required.

# Chapter 31 Option Byte

## 31.1Functions of Option Bytes

Addresses 000C0H~000C3H, 500004H of the flash emory form an option byte area.

Option bytes consist of user option byte (000C0H to 000C2H) and flash memory data protection option byte (000C3H, 500004H). When powered on or reset is initiated, the specified function is set with reference to the option byte. When using this product, the following functions must be set by the option byte. For bits that do not have configuration capabilities, you cannot change the initial value.

Caution: Regardless of whether or not to use each function, you must set the option byte.

### 31.1.1 User Option Bytes (000C0H~000C2H)

(1) 000C0H

- Operation of watchdog timer
  - Enable or disable counter operation.
  - Enable or stop counter operation in sleep/deep sleep mode.
- Setting of watchdog timer overflow time
  - Setting of window open period of watchdog timer
- Setting of interval interrupt of watchdog timer
  - Whether or not to use the interval interrupt is selectable.

(2) 000C1H

- Setting of LVD operation mode
  - Interrupt & reset mode.
  - Reset mode.
  - Interrupt mode.
  - LVD off (by controlling the externally input reset signal on the RESETB pin)
- Setting of LVD detection level ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$)

Note 1 : When the supply voltage rises, the reset state must be maintained by voltage detection circuits or external resets before the supply voltage reaches the operating voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset by transferring in deep sleep mode, voltage detection circuitry, or external reset before the supply voltage falls below the operating voltage range.

Note 2 : The operating voltage range depends on the setting of the user option byte (000C2H).

(3) 000C2H

- Setting of the frequency of the high-speed on-chip oscillator
  - Select from 2MHz~36MHz, 64MHz, and 72MHz

## 31.2Flash Memory Data Protection Option Bytes (000C3H, 500004H)

- Control of flash memory data protection when debugging on-chip

  Level0: Read/write/erase operations on flash data are enabled via debugger.

  Level1: Chip erase operations on flash data via debugger are enabled, read/write operations are disabled.

  Level2: Operations on flash data via debugger are disabled.

## 31.3Register Mapping

| Register | Address | R/W | Description | Reset value |
|---|---|---|---|---|
| Option byte 0 | 0x00C0H | R/W | Watchdog Timer Status Control Register | 0xFF |
| Option byte 1 | 0x00C1H | R/W | LVD Status Control Register | 0xFF |
| Option byte 2 | 0x00C2H | R/W | High-Speed On-Chip Oscillator Frequency Control Register | 0xEC |
| Option byte 3 | 0x00C3H | R/W | On-chip Debug Flash Data Protection Control Register 1 | 0xFF |
| Option byte 4 | 0x500004H | R/W | On-chip Debug Flash Data Protection Control Register 2 | 0xFF |

# 31.4User Option Bytes

## 31.4.1 User Option Byte 0 (000C0H)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7 | WDTINT | Interval interrupt of watchdog timer<br>0: Interval interrupt is not used.<br>1: When 75% of the overflow time + 1/2FIL is reached, an interval interrupt is generated. | 1 |
| 6:5 | WINDOW[1:0] | When watchdog timer window opens<br>0X: Settings are disabled.<br>10: 75%<br>11: 100% | 0x3 |
| 4 | WDTON | Controlling counter operation of watchdog timer<br>0: Disable counter operation (stop counting after the reset is released).<br>1: Enable counter operation (start counting after the reset is released). | 1 |
| 3:1 | WDTCS[2:0] | Overflow time of watchdog timer ($F_{IL}$=15kHz)<br>000: $2^6/F_{IL}$ (4.3ms)<br>001: $2^7/F_{IL}$ (8.5ms)<br>010: $2^8/F_{IL}$ (17.0ms)<br>011: $2^9/F_{IL}$ (34.0ms)<br>100: $2^{11}/F_{IL}$ (135.9ms)<br>101: $2^{13}/F_{IL}$ (543.5ms)<br>110: $2^{14}/F_{IL}$ (1086.9ms)<br>111: $2^{16}/F_{IL}$ (4347.8ms) | 0x7 |
| 0 | WDSTBYON | Counter operation control (sleep mode) of watchdog timer<br>0: In sleep mode, counter operations are stopped Note1.<br>1: In sleep mode, counter operations are enabled. | 1 |

Note 1: When the WDSTBYON bit is “0”, regardless of the values of the WINDOW1 bit and the WINDOW0 bit, it is 100% during window opening.

Note 2: $F_{IL}$: Low-speed on-chip oscillator clock frequency

## 31.4.2 User Option Byte 1 (000C1H)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:5 | VPOC[2:0] | Detection voltage setting | 0x7 |
| 4 | - | Reserved (Set to 1) | 1 |
| 3:2 | LVIS[1:0] | Detection voltage setting | 0x2 |
| 1:0 | LVIMDS[1:0] | Mode selection<br>10: Interrupt & reset mode<br>11: Reset mode<br>01: Interrupt mode | 0x3 |

- LVD settings (interrupt & reset mode)

| Detect voltage | | | Setting value of option byte | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $V_{LVDH}$ | | $V_{LVDL}$ | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising | Falling | Falling | | | | | | LVIMDS1 | LVIMDS0 |
| 1.77V | 1.73V | 1.63V | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 1.88V | 1.84V | | | | | 0 | 1 | | |
| 2.92V | 2.86V | | | | | 0 | 0 | | |
| 1.98V | 1.94V | 1.84V | | 0 | 1 | 1 | 0 | | |
| 2.09V | 2.04V | | | | | 0 | 1 | | |
| 3.13V | 3.06V | | | | | 0 | 0 | | |
| 2.61V | 2.55V | 2.45V | | 1 | 0 | 1 | 0 | | |
| 2.71V | 2.65V | | | | | 0 | 1 | | |
| 3.75V | 3.67V | | | | | 0 | 0 | | |
| 2.92V | 2.86V | 2.75V | | 1 | 1 | 1 | 0 | | |
| 3.02V | 2.96V | | | | | 0 | 1 | | |
| 4.06V | 3.98V | | | | | 0 | 0 | | |
| — | | | Settings other than above are prohibited. | | | | | | |

Note 1: Set the bit4 to “1”.

Note 2: For details of LVD circuit, please refer to “Chapter 28 Voltage Detection Circuit”.

Note 3: The detection voltage is a TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

• LVD setting (reset mode)

| Detect voltage | | Setting value of option byte | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising | Falling | | | | | | LVIMDS1 | LVIMDS0 |
| 1.67V | 1.63V | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1.77V | 1.73V | | 0 | 0 | 1 | 0 | | |
| 1.88V | 1.84V | | 0 | 1 | 1 | 1 | | |
| 1.98V | 1.94V | | 0 | 1 | 1 | 0 | | |
| 2.09V | 2.04V | | 0 | 1 | 0 | 1 | | |
| 2.50V | 2.45V | | 1 | 0 | 1 | 1 | | |
| 2.61V | 2.55V | | 1 | 0 | 1 | 0 | | |
| 2.71V | 2.65V | | 1 | 0 | 0 | 1 | | |
| 2.81V | 2.75V | | 1 | 1 | 1 | 1 | | |
| 2.92V | 2.86V | | 1 | 1 | 1 | 0 | | |
| 3.02V | 2.96V | | 1 | 1 | 0 | 1 | | |
| 3.13V | 3.06V | | 0 | 1 | 0 | 0 | | |
| 3.75V | 3.67V | | 1 | 0 | 0 | 0 | | |
| 4.06V | 3.98V | | 1 | 1 | 0 | 0 | | |
| — | | Settings other than above are prohibited. | | | | | | |

Note 1: Set the bit4 to “1”.

Note 2: For details of LVD circuit, please refer to “Chapter 28 Voltage Detection Circuit”.

Note 3: The detection voltage is a TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

• LVD setting (interrupt mode)

| Detect voltage | | Setting value of option byte | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising | Falling | | | | | | LVIMDS1 | LVIMDS0 |
| 1.67V | 1.63V | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1.77V | 1.73V | | 0 | 0 | 1 | 0 | | |
| 1.88V | 1.84V | | 0 | 1 | 1 | 1 | | |
| 1.98V | 1.94V | | 0 | 1 | 1 | 0 | | |
| 2.09V | 2.04V | | 0 | 1 | 0 | 1 | | |
| 2.50V | 2.45V | | 1 | 0 | 1 | 1 | | |
| 2.61V | 2.55V | | 1 | 0 | 1 | 0 | | |
| 2.71V | 2.65V | | 1 | 0 | 0 | 1 | | |
| 2.81V | 2.75V | | 1 | 1 | 1 | 1 | | |
| 2.92V | 2.86V | | 1 | 1 | 1 | 0 | | |
| 3.02V | 2.96V | | 1 | 1 | 0 | 1 | | |
| 3.13V | 3.06V | | 0 | 1 | 0 | 0 | | |
| 3.75V | 3.67V | | 1 | 0 | 0 | 0 | | |
| 4.06V | 3.98V | | 1 | 1 | 0 | 0 | | |
| — | | Settings other than above are prohibited. | | | | | | |

Note 1: Set the bit4 to “1”.

Note 2: For details of LVD circuit, please refer to “Chapter 28 Voltage Detection Circuit”.

Note 3: The detection voltage is a TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

• Setting when LVD is OFF (external reset input using RESETB pin)

| Detect voltage | | Setting value of option byte | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVDH}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising | Falling | | | | | | LVIMDS1 | LVIMDS0 |
| — | — | 1 | × | × | × | × | × | 1 |
| — | | Settings other than above are prohibited. | | | | | | |

Caution 1: Set the bir4 to “1”.

Caution 2: When the supply voltage rises, the reset state must be maintained by the voltage detection circuit or external reset before the supply voltage reaches the operating voltage range shown in the AC Characteristics of the datasheet; when the supply voltage falls, the reset state must be reset by the transfer of the sleep mode, the voltage detection circuit, or the external reset before the supply voltage falls below the operating voltage range. The operating voltage range depends on the setting of the user option byte (000C2H).

Note 1: ×: Ingore

Note 2: For details of LVD circuit, please refer to “Chapter 28 Voltage Detection Circuit”.

Note 3: The detection voltage is a TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

### 31.4.3 User Option Byte 2 (000C2H)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:5 | - | Reserved (Set to 1) | 0x7 |
| 4:0 | FRQSE[4:0] | High-speed on-chip oscillator clock frequency selection | 0x0C |

| FRQSE4 | FRQSEL3 | FRQSEL2 | FRQSEL1 | FRQSEL0 | High-speed on-chip oscillator clock frequency | |
|---|---|---|---|---|---|---|
| | | | | | $F_{HOCO}$ | $F_{IH}$ |
| 1 | x | 0 | 0 | 0 | 72MHz | 72MHz |
| 1 | x | 0 | 0 | 1 | 72MHz | 36MHz |
| 1 | x | 0 | 1 | 0 | 72MHz | 18MHz |
| 1 | x | 0 | 1 | 1 | 72MHz | 9MHz |
| 1 | x | 1 | 0 | 0 | 72MHz | 4.5MHz |
| 0 | x | 0 | 0 | 0 | 64MHz | 64MHz |
| 0 | x | 0 | 0 | 1 | 64MHz | 32MHz |
| 0 | x | 0 | 1 | 0 | 64MHz | 16MHz |
| 0 | x | 0 | 1 | 1 | 64MHz | 8MHz |
| 0 | x | 1 | 0 | 0 | 64MHz | 4MHz |
| 0 | x | 1 | 0 | 1 | 64MHz | 2MHz |
| Other than the above | | | | | Settings are prohibited | |

Note 1: Bits 7 to 5 must be set to “1”.

Note 2: Operating frequency range and operating voltage range vary depending on each operating mode of the flash memory. For details, refer to AC Characteristics in the datasheet.

### 31.4.4 Flash Memory Data Protection Option Byte 1 (000C3H)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | OCDEN[7:0] | Control of flash memory data protection | 0xFF |

### 31.4.5 Flash Memory Data Protection Option Byte 2 (500004H)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:0 | OCDM[7:0] | Control of flash memory data protection | 0xFF |

| OCDM | OCDEN | Control of flash memory data protection |
|---|---|---|
| 3C | C3 | Manipulation of flash data via debugger is disabled. |
| Other than 3C | C3 | Chip erase operation on flash data via debugger is enabled, read/write operation is disabled. |
| Other than the above | | Read/write/erase operations on flash data via debugger are enabled. |

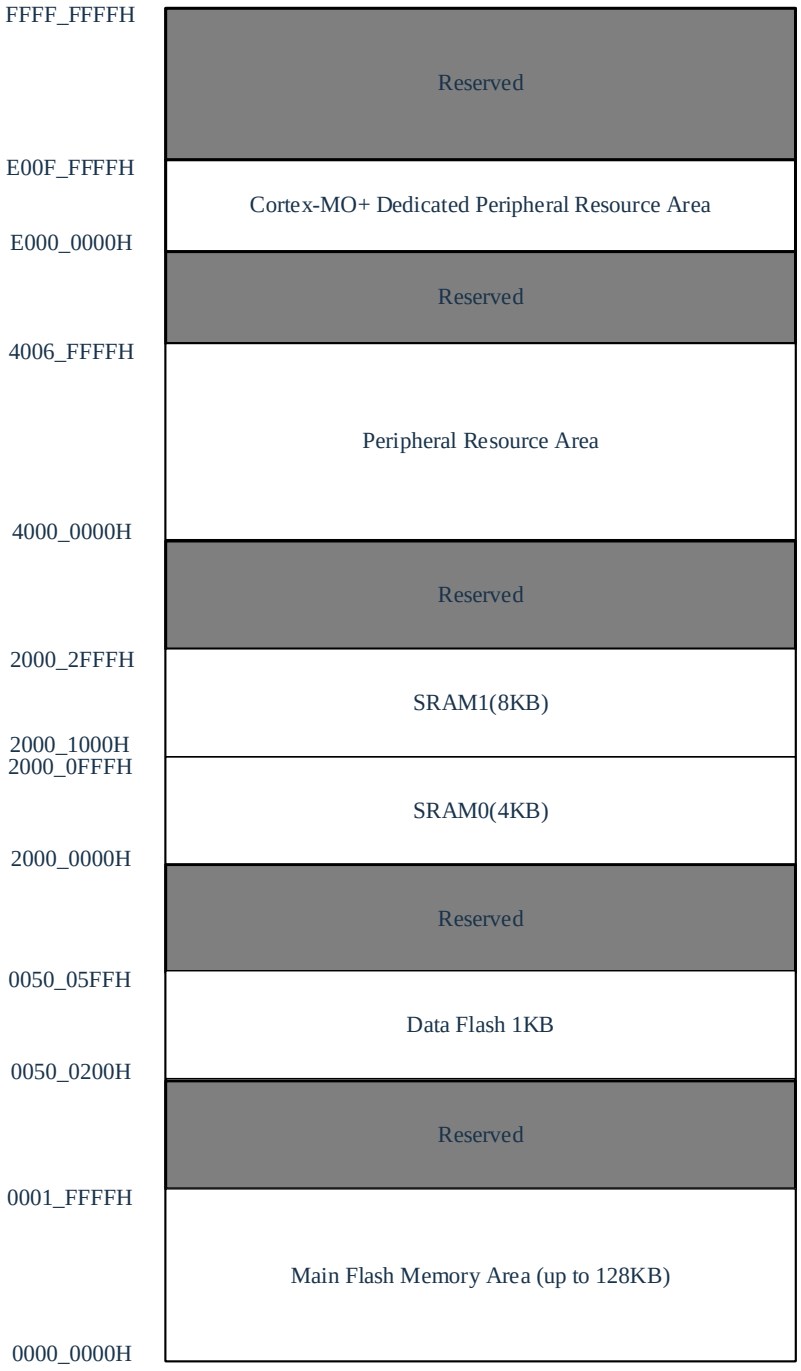
Note: The 50_0004H address belongs to the data flash memory area. If you use this address for data storage, make sure that the value will not cause the protection option to be set incorrectly.

# Chapter 32 Flash Control

## 32.1Overivew of Flash Control

This product contains a 128KB flash memory, which is divided into 256 Sectors, each with a capacity of 512 Bytes. It can be used as program memory and data memory. This module supports erasing, programming and reading operations for this memory.

## 32.2Structure of Flash Memory



# 32.3Register Mapping

(Flash control base address = 0x4002_0000) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| FLSTS | 0x000 | R/W | Flash Status Register | 0x0 |
| FLOPMD1 | 0x004 | R/W | Flash Operation Control Register 1 | 0x0 |
| FLOPMD2 | 0x008 | R/W | Flash Operation Control Register 2 | 0x0 |
| FLERMD | 0x00C | R/W | Flash Erase Control Register | 0x0 |
| FLCERCNT | 0x010 | R/W | Flash Chip Erase Time Control Register | 0xCE |
| FLSERCNT | 0x014 | R/W | Flash Page Erase Time Control Register | 0x149 |
| FLPROCNT | 0x01C | R/W | Flash Write Time Control Register | 0XB0007E |
| FLPROT | 0x020 | R/W | Flash Write Protect Register | 0x0 |

# 32.4Register Description

## 32.4.1 Flash Write Protection Register (FLPROT)

Flash protection register is a register used to protect the flash operation control register.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:1 | PRKEY | WRP write protection<br>78h: Enable rewriting WRP<br>Other: Disable rewriting WRP | 0x0 |
| 0 | WRP | Operation register (FLOPMD1/FLOPMD2) write protection<br>1: Enable rewriting FLOPMD1/FLOPMD2<br>0: Disable rewriting FLOPMD1/ FLOPMD2 | 0 |

## 32.4.2 Flash Operation Control Register (FLOPMD1)

Flash operation control register is used to set the erase and write operations of flash.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | FLOPMD1 | Flash operation select bit:<br>0x55: When FLOPMD2=0xAA: Erase<br>0xAA: When FLOPMD2=0x55: Write<br>0x00: When FLOPMD2=0x00: Read<br>Other than the above: Settings are disabled | 0x0 |

### 32.4.3 Flash Operation Control Register (FLOPMD2)

Flash operation control register is used to set the erase and write operations of flash.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | FLOPMD2 | Flash operation select bit:<br>0xAA: When FLOPMD1=0x55: Erase<br>0x55: When FLOPMD1=0xAA: Write<br>0x00: When FLOPMD1=0x00: Read<br>Other than the above: Settings are disabled | 0x0 |

### 32.4.4 Flash Erase Control Register (FLERMD)

Flash erase control register is used to set the type of flash erase operation.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:5 | - | Reserved | - |
| 4:3 | ERMD | Erase operation control bit:<br>0: Sector erase, no hardware check after erase<br>1: Chip erase Note<br>2: Sector erase, hardware check after erase<br>3: Settings are disabled | 0x0 |
| 2:0 | - | Reserved | - |

Note: Chip erase only erases the code flash area, not the data flash area. And chip erase does not support hardware check.

### 32.4.5 Flash Status Register (FLSTS)

The status of the flash controller can be queried through the status register.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 7:3 | - | Reserved | - |
| 2 | EVF(Note2) | Flash erase hardware check error flag:<br>0: No hardware check error after the flash erase<br>1: A hardware check error is triggered after the flash erase | 0 |
| 1 | - | Reserved | - |
| 0 | OVF(Note1) | Flash erase/write operation complete flag:<br>0: The flash erase/write operation is not completed<br>1: The flash erase/write operation is completed | 0 |

Note 1: The OVF needs to be cleared by writing “1” through software. If it is not cleared, the next erase operation cannot be performed.

Note 2: The EVF needs to be cleared by writing “1” through software.

## 32.4.6 Flash Chip Erase Time Control Register (FLCERCNT)

FLCERCNT register enables to set the flash chip erase time.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | Load | Chip erase time setting selection Note<br>0: Erase time is set by hardware<br>1: Erase time is set by software (FLCERCNT[10:0]) | 0 |
| 30:11 | - | Reserved | - |
| 10:0 | FLCERCNT | Software erase time setting<br>Chip erase time = (CERCNT*2048*Tfclk), which meets the hardware requirement of >30ms | 0xCE |

Note: When the master clock is an on-chip high-speed OCO or the external input clock is<=20M, the hardware setting time can be used without setting FLCERCNT.

## 32.4.7 Flash Sector Erase Time Control Register (FLSERCNT)

FLSERCNT register enables to set the flash sector erase time.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | Load | Sector erase time setting selection Note<br>0: Erase time is set by hardware<br>1: Erase time is set by software (FLSERCNT[10:0]) | 0 |
| 30:11 | - | Reserved | - |
| 10:0 | FLSERCNT | Software erase time setting<br>sector erase time = (FLSERCNT*256*Tfclk) which meets the hardware requirement of >2ms | 0x149 |

Note: When the master clock is an on-chip high-speed OCO or the external input clock is<=20M, the time can be set in hardware without setting FLSERCNT.

## 32.4.8 Flash Write Time Control Register (FLPROCNT)

FLPROCNT register enables to set the flash word write time.

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31 | Load1 | Write action setup time ($T_{PGS}$) setting[Note 1]<br>0: Write action setup time by hardware<br>1: Write action setup time by software FLPGSCNT[12:0] | 0 |
| 30:29 | - | Reserved | - |
| 28:16 | FLPGSCNT | Write action setup time by software<br>Write action setup time = (PGSCNT*Tfclk), which meets the hardware requirement of>70us | 0xB0 |
| 15 | Load0 | Write time ($T_{PROG}$) setting selection [Note 2:]<br>0: Write time is set by hardware<br>1: Write time is set by software FLPROCNT[8:0] | 0 |
| 14:10 | - | Reserved | - |
| 9:0 | FLPROCNT | Software write time setting<br>Write time = (FLPROCNT*Tfclk), which meets the hardware requirement of >7.5us | 0x7E |

Note 1: When the master clock is an on-chip high-speed OCO or the external input clock is<=20M, you can set the time by hardware without setting the FLPGSCNT.

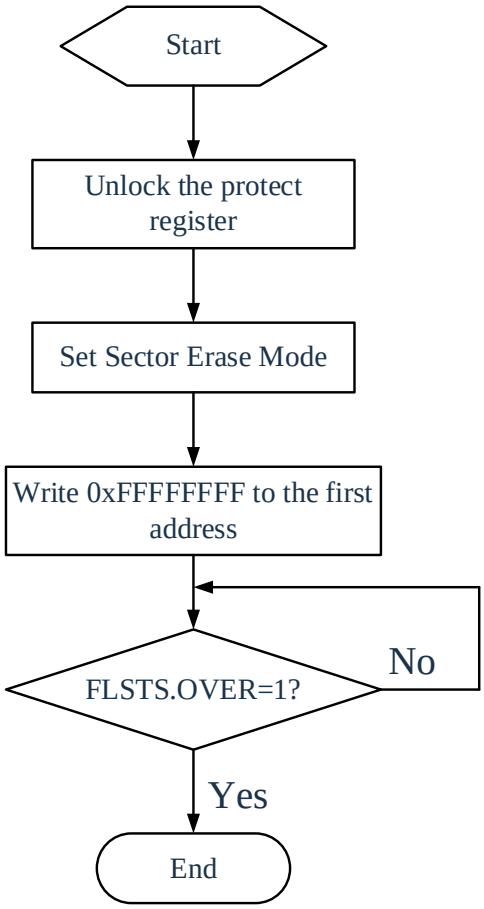
Note 2: When the master clock is an on-chip high-speed OCO or the external input clock is<=20M, you can set the time by hardware without setting the FLPROCNT.

# 32.5How To Operate Flash

## 32.5.1 Sector Erase

The Sector erase time is realized by the hardware or can be configured by FLSERCNT. The operation flow is as follows.

1) Set FLERMD.ERMD0 to 1'b0, select the sector erase mode, and set the value of ERMD1 according to whether or not hardware check is required.
2) Set FLPROT to 0xF1, unprotect FLOPMD. Then set FLOPMD1 to 0x55, FLOPMD2 to 0xAA, and
3) Write arbitrary data to the first address of the erase target sector. Example: * ((unsigned long *)0x00000200)=0xffffffff.
4) Query the status register FLSTS.OVF through software, when OVF=1, it means the erase operation is completed.
5) If hardware check after erase is set (ERMD1=1), you can determine whether the verification is correct by software for FLSTS.EVF.
6) Before the next operation, set the software to “1” to clear the FLSTS.



### 32.5.2 Chip Erase

Chip erase, and the erase time are implemented by hardware and can also be configured via FLCERCNT. The operation process is as follows

1) Set FLERMD. ERMD0 to 1'b1, and select chip erase mode;
2) Set FLPROT to 0xF1 to unprotect FLOPMD. Then set FLOPMD1 to 0x55 and FLOPMD2 to 0xAA.
3) Write arbitrary data to any address in the flash area of the code.
4) Query the status register FLSTS.OVF through software, when OVF=1, it means the erase operation is completed.
5) Before the next operation, set the software to “1” to clear the FLSTS.

### 32.5.3 Word Program

Word programming and write time are implemented by hardware and can also be configured via PROCNT. The operation process is as follows:

1) Set FLPROT to 0xF1, unprotect FLOPMD. Then set FLOPMD1 to 0xAA, FLOPMD2 to 0x55, and
2) Write the corresponding data to the target address.
3) Query the status register FLSTS.OVF through software, when OVF=1, it means the write operation is completed.
4) Before the next operation, set the software to “1” to clear the FLSTS.

## 32.6Cautions for Flash Operation

1) Flash memory has strict time requirements for the control signal of erasing and programming operation, and the timing of the control signal is not qualified, which will cause the erase operation and programming operation to fail. The setting of the erase and write parameters can be implemented by hardware, or it can be modified by modifying the parameter registers; When using on-chip high-speed OCO, MAINOSC/ external input clock = 20M, it is recommended to use hardware-set erase and write parameters without setting parameter registers.
2) If the erase/write operation is executed from flash, the CPU stops fetching and the hardware automatically waits for the completion of the operation to proceed to the next instruction. If the operation is executed from RAM, the CPU will not stop fetching and can continue the next instruction.
3) If the CPU executes an instruction to enter deep sleep while the flash is in programming operation, the system waits for the programming action to end before entering deep sleep.

# Chapter 33 Test

## 33.1Register Mapping

(TEST base address = 0x4006_9000) RO: Read only, WO: Write only, R/W: Read/Write

| Register | Offset value | R/W | Description | Reset value |
|---|---|---|---|---|
| TEST1 | 0x000 | R/W | Test Register 1 | 0x0 |
| LOCK | 0x00C | R/W | Test Register Access Enable Control Register | 0x0 |

## 33.2Register Description

### 33.2.1 Test Register 1(TEST1)

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:3 | - | Reserved | - |
| 2 | PGA123_OEN | PGA123 output to PAD<br>0: Disable<br>1: Enable (output to P53) | 0 |
| 1:0 | - | Set to 0. | 0x0 |

### 33.2.2 Test Register Access Enable Control Register

| Bit | Symbol | Description | Reset value |
|---|---|---|---|
| 31:8 | - | Reserved | - |
| 7:0 | LOCK | When LOCK=0x55, enable operation of the test register.<br>When LOCK = other values, disable operation of the test register. | 0x0 |

# Chapter 34 Appendix Revision History

| Version # | Date | Decription of Changes |
|---|---|---|
| V1.0.0 | Apr. 2025 | Formal version<br>1) Changed some section names<br>2) Revised section 14.3.6<br>Revised section 22.5.1 |