32 Bit RISC Microcontroller
TX03 Series

TMPM363F10FG

ARM

Important Notices

Make sure to read read this chapter before using the product.

1 Serial bus interface

There are restrictions on the use of I2C bus mode when the multi-master function is used.

1.1 Description

When the multi-master function is used in I2C bus mode, if these masters start the communications simultaneously, the following phenomena may occur:

1. Communications may be locked up.
2. SCL pulse widths shorten; therefore these pulses may not satisfy I2C Specifications.

1.2 Condition

These phenomena occur only when the multi-master function is used in I2C bus mode. If a single master is used, these phenomena do not occur.

1.3 Workaround

There is no workaround for these phenomena. Perform recovery process by software.

1.4 How to Recover from These Phenomena

Perform recovery process by software.

By using a timer, add timeout process to check whether communication is in a lock-up state.
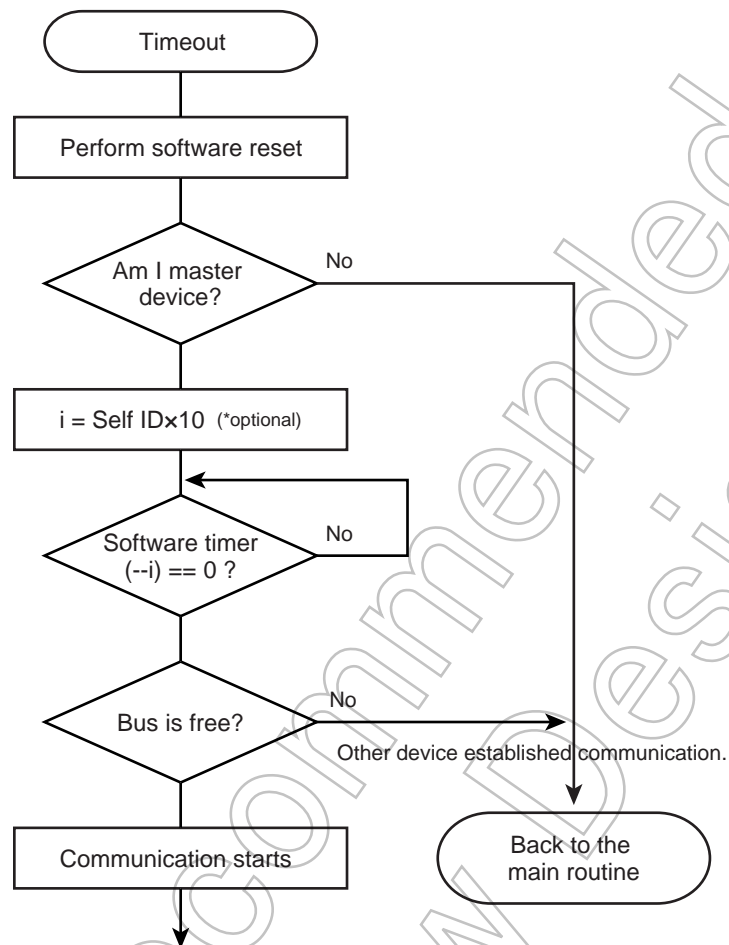
An example of recovery process:

1. Start a timer count synchronously with start of the transmission.
2. If a serial interface interrupt (INTSBIx) does not occur in a certain period, the MCU determines the timeout.
3. If the MCU determines the timeout, communications may be locked up. Perform software reset on the serial bus interface circuit. This circuit is initialized to release communication from the lock up state.
4. Resend transmission data.

Mostly, Process 1 to 4 are enough to recovery; however if the multiple products are connected to the same bus line, add a delay time between each product's recovery process before Process 4 (resending data) is performed. This delay makes a time difference between each master; therefore bus collision can be avoided when the data is sent again.

Example: Recovery process after a timeout is detected.

```
                        ┌──────────────────┐
                        │     Timeout       │
                        └──────────────────┘
                                 │
                 ┌───────────────────────────────┐
                 │   Perform software reset       │
                 └───────────────────────────────┘
                                 │
                            ╱╲         No
                          ╱ Am I master ╲──────────────────────────┐
                          ╲   device?   ╱                          │
                            ╲╱                                     │
                                 │                                 │
                 ┌───────────────────────────────┐                │
                 │  i = Self ID×10  (*optional)   │                │
                 └───────────────────────────────┘                │
                                 │           ◄───────────┐         │
                            ╱╲                 No         │        │
                          ╱ Software timer ╲──────────────┘        │
                          ╲ (--i) == 0 ?   ╱                       │
                            ╲╱                                     │
                                 │                                 │
                            ╱╲          No                         │
                          ╱ Bus is free? ╲────────────────────────│──┐
                          ╲              ╱                         │  │
                            ╲╱        Other device established     │  │
                                 │     communication.              ▼  │
                 ┌───────────────────────────────┐       ┌──────────────────┐
                 │   Communication starts          │       │  Back to the     │
                 └───────────────────────────────┘       │  main routine    │
                                 │                         └──────────────────┘
                                 ▼
```

Introduction: Notes on the description of SFR (Special Function Register) under this specification

An SFR (Special Function Register) is a control register for periperal circuits (IP).

The SFR addressses of IPs are described in the chapter on memory map, and the details of SFR are given in the chapter of each IP.

Definition of SFR used in this specification is in accordance with the following rules.

   a. SFR table of each IP as an example
- SFR tables in each chapter of IP provides register names, addresses and brief descriptions.
- All registers have a 32-bit unique address and the addresses of the registers are defined as follows, with some exceptions: "Base address + (Unique) address"

Base Address = 0x0000_0000

Register name		Address(Base+)
Control register	SAMCR	0x0004
		0x000C

Note: **SAMCR register address is 32 bits wide from the address 0x0000_0004 (Base Address(0x00000000) + unique address (0x0004)).**

Note: **The register shown above is an example for explanation purpose and not for demonstration purpose. This register does not exist in this microcontroller.**

   b. SFR(register)
- Each register basically consists of a 32-bit register (some exceptions).
- The description of each register provides bits, bit symbols, types, initial values after reset and functions.

1.2.2 SAMCR(Control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	MODE	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MODE	TDATA						
After reset	0	0	0	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-10	–	R	"0" can be read.
9-7	MODE[2:0]	R/W	Operation mode settings 000 : Sample mode 0 001 : Sample mode 1 010 : Sample mode 2 011 : Sample mode 3 The settings other than those above: Reserved
6-0	TDATA[6:0]	W	Transmitted data

Note: **The Type is divided into three as shown below.**

R / W   READ WRITE

R   READ

W   WRITE

c. Data descriptopn

Meanings of symbols used in the SFR description are as shown below.

· x:channel numbers/ports

· n,m:bit numbers

d. Register descriptoption

Registers are described as shown below.

· Register name <Bit Symbol>

Exmaple: SAMCR<MODE>="000" or SAMCR<MODE[2:0]>="000"

<MODE[2:0]> indicates bit 2 to bit 0 in bit symbol mode (3bit width).

· Register name [Bit]

Example: SAMCR[9:7]="000"

It indicates bit 9 to bit 7 of the register SAMCR (32 bit width).

Revision History

Date	Revision	Comment
2011/04/13	Tentative 1	First Release of Tentative
2011/06/23	1	First Release
2013/05/31	2	Contents Revised
2023/07/21	3	Contents Revised

Table of Contents

8.   Input / Output Ports

9. DMA Controller(DMAC)

10. Static Memory Controller

15.   Consumer Electronics Control (CEC)

16.   CAN Controller (CAN)

17.   Remote control signal preprocessor(RMC)

18.   USB Host Controller

21. Backup module

22. Analog / Digital Converter (ADC)

23. Real Time Clock (RTC)

24. Flash

25.   ROM protection

26.   RAM Interface

27.   Port Section Equivalent Circuit Schematic

28.   Electrical Characteristics

29.   Package Dimensions

TMPM363F10FG

The TMPM363F10FG is a 32-bit RISC microprocessor series with an ARM Cortex-M3 microprocessor core.

Product name	ROM (FLASH)	RAM	Package
TMPM363F10FG	1024 Kbyte	64 Kbyte	P-LQFP100-1414-0.50H

Features of the TMPM363F10FG are as follows :

1.1 Features

1. ARM Cortex-M3 microprocessor core
    a. Improved code efficiency has been realized through the use of Thumb-2 instruction.
        ・ New 16-bit Thumb instructions for improved program flow
        ・ New 32-bit Thumb instructions for improved performance
        ・ New Thumb mixed 16- / 32-bit instruction set can produce faster, more efficient code.
    b. Both high performance and low power consumption have been achieved.
        [High performance]
        ・ Both high performance and low power consumption have been achieved.
        ・ Division takes between 2 and 12 cycles depending on dividend and devisor
        [Low Power consumption]
        ・ Optimized design using a low power consumption library
        ・ Standby function that stops the operation of the micro controller core
    c. High-speed interrupt response suitable for real-time control
        ・ An interruptible long instruction
        ・ Stack push automatically handled by hardware

2. On Chip program memory and data memory
    ・ On chip RAM : 64Kbyte
    ・ On chip Flash ROM : 1024Kbyte

3. Static memory controller (SMC)
    ・ Up to 16Mbytes access area (Program / Data)
    ・ External data bus (Multiplex bus) : 16-bit data bus width
    ・ Chip select / Wait controller : 4 channels

4. DMA controller (DMAC) : 2 channels
    Transfer can support on chip Memory / Peripheral I/O / External memory

5. 16-bit timer / event counter (TMRB) : 16 channels
    - 16-bit interval timer mode
    - 16-bit event counter mode
    - 16-bit PPG output (4channel timer can start synchronously)
    - Input capture function

6. Watchdog timer (WDT) : 1 channel

    Watchdog timer generates a reset or a non-maskable interrupt (NMI).

7. Serial channnel (SIO/UART) : 5 channels

    Either UART mode or I/O interface can be selected (4byte FIFO equipped)

8. Serial bus interface (I2C/SIO) : 3 channels

    Either I2C bus mode or synchronous 8-bit SIO mode can be selected.

9. I2C bus : 1 channel

10. Synchronous serial port (SSP) : 1 channel
    - Communication protocol that includes SPI: 3 types (SPI/SSI/Microwire)
    - Baud rate: Master mode: 16Mbps (max.), Slave mode: 5.3Mbps (max.)

11. USB2.0 full-speed host : 1 channel
    - Conforms to Universal Serial Bus Specification Rev2.0
    - Conforms to OpenHCI for Release 1.0a
    - Control/Bulk/Interrupt/Isochronous mode
    - Full-speed 12Mbps (Low speed is not available.)

12. CAN 2.0 : 1 channel
    - Version 2.0B Active supported
    - 32 mail boxes
    - Maximum transfer rate : 1 Mbps

13. CEC function (CEC) : 1 channel

    Transmision and reception per 1 byte

14. Remote control signal preprocessor (RMC) : 1 channel

    Can receive up to 72bit data at a time

15. 10-bit AD converter (ADC) : 8 channels
    - Start up with 16-bit timer
    - Fixed channel / scan mode
    - Single / repeat mode
    - AD monitoring 2 channels
    - Conversion time 1.15 μsec (@ fsys = 40 MHz)

16. Key-on wake-up (KWUP) : 4 channels

Dynamic pull-up

17. Real time clock (RTC) : 1channel
   ・ Clock (hour, minute and second)
   ・ Calendar (month, week, date and leap year)
   ・ Alarm (Alarm output)
   ・ Alarm interrupt

18. BACKUP module (BUPMD)

   Low power consumption can be realized by shutdown the power supply except specific part.
   - BACKUP RAM : 8KB
   - Port keep (Keep port status when BACKUP mode is set)
   - CEC Function
   - Remote control signal preprocessor function
   - Key-on wake-up function
   - Real time clock function

19. Interrupt source
   ・ Internal 58 factors : The order of precedence can be set over 7 levels.
      (execpt the watchdog timer interrupt)
   ・ External 8 factors : The order of precedence can be set over 7 levels.

20. Non-maskable interrupt (NMI)

   Non-maskable interrupt (NMI) is generated by a watchdog timer or a $\overline{\text{NMI}}$ pin.

21. Input / output ports (PORT) : 74 pins
   I/O pin : 66 pins
   Input pin : 8 pins

22. Low power consumption mode
   IDLE2, IDLE1, SLEEP, STOP, SLOW, BACKUP (BACKUP SLEEP, BACKUP STOP)

23. Clock generator (CG)
   ・ On chip PLL (Quadrupled or octuple can be selected.)
   ・ Clock gear function : The high-speed clock can be divided into 1/1, 1/2, 1/4 or 1/8.

24. Endian
   Little endian

25. Debug interface
   SWD / SWV / TRACE (DATA 4bit)

26. Maximum operating frequency : 64MHz (48MHz when USB is used.)

27. Operating voltage range

2.7 V to 3.6 V (with on-chip regulator)

3.0 V to 3.6 V (when USB is used)

28. Temperature range
   - -40 degrees to 85 degrees (except Flash writing / erasing)
   - 0 degrees to 70 degrees (during Flash writing / erasing)

29. Package

P-LQFP100-1414-0.50H (14 mm x 14 mm, 0.5 mm pitch)
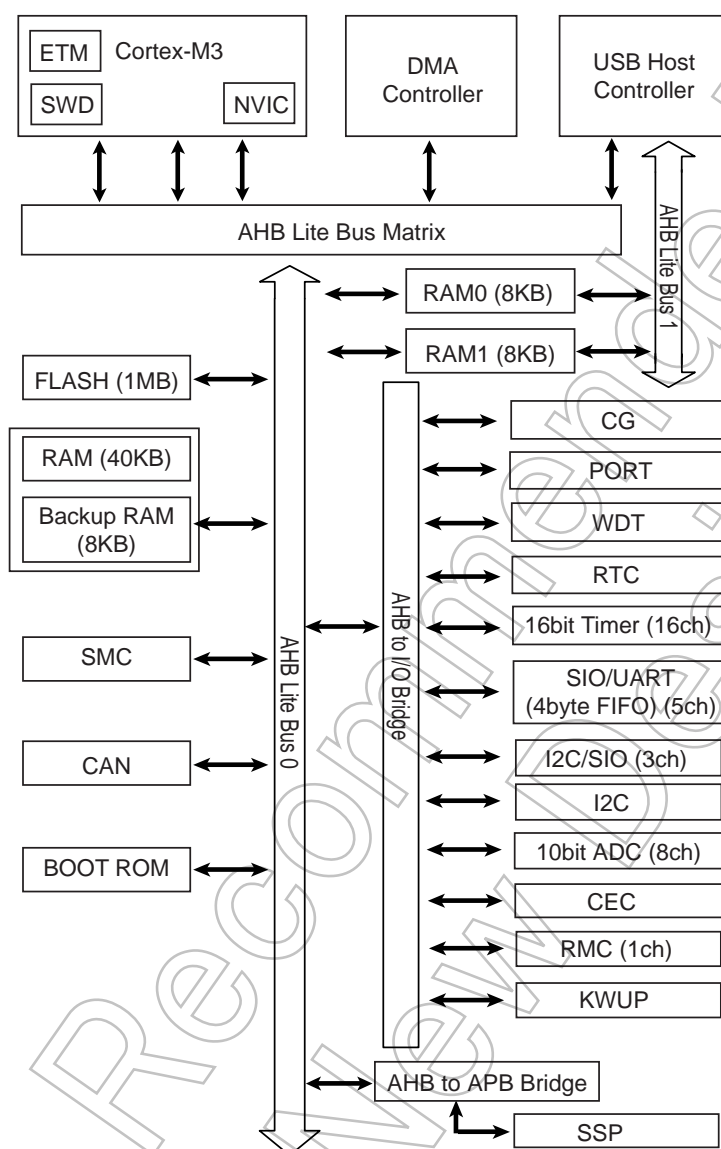
1.2 Block Diagram

Figure 1-1 TMPM363F10FG Block Diagram

1.3 Pin layout (Top view)

Figure 1-2 shows the pin layout of TMPM363F10FG.
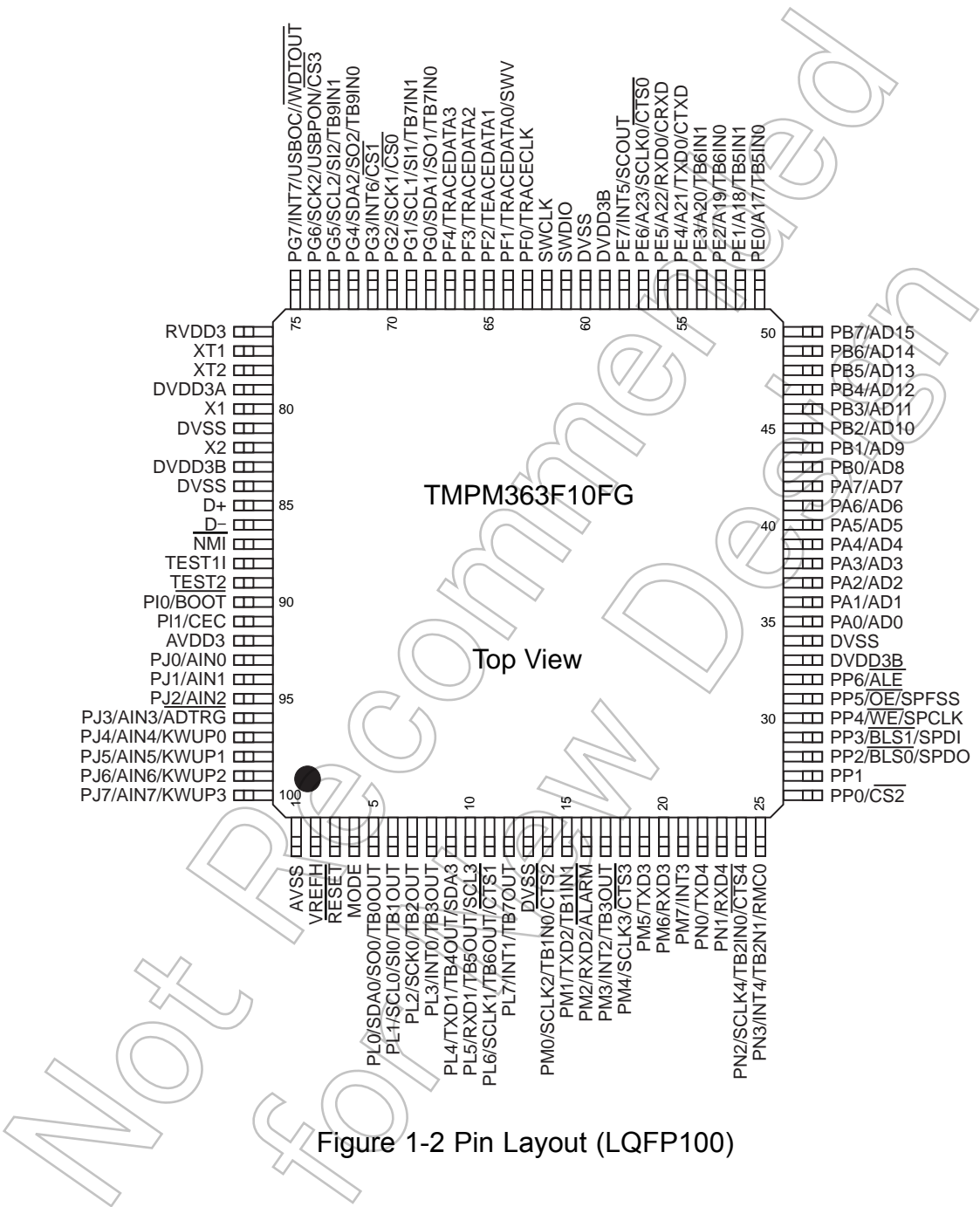
Figure 1-2 Pin Layout (LQFP100)

1.4   Pin names and Functions

Table 1-1 sort input and output pins of TMPM363F10FG by pin or port.

Table 1-1 Pin Names and Functions Sorted by Pin (1/6)

Type	Pin No.	Pin Name	Input / Output	Function
PS	1	AVSS	-	GND pin for AD converter (note) AVSS must be connected to GND even if the AD converter is not used.
PS	2	VREFH	-	Power supply pin for AD converter (note) VREFH must be connected to power supply even if AD converter is not used.
Function	3	$\overline{\text{RESET}}$	Input	Reset input pin (note) With a pull-up and a noise filter (about 30 ns (typ.))
Control	4	MODE	Input	Mode pin (note) MODE pin must be connected to GND.
Function	5	PL0 SDA0/SO0 TB0OUT	I/O I/O Output	I/O port Data in I2C mode / Data in SIO mode 16-bit timer / event counter output
Function	6	PL1 SCL0/SI0 TB1OUT	I/O I/O Output	I/O port Clock in I2C mode / Data in SIO mode 16-bit timer / event counter output
Function	7	PL2 SCK0 TB2OUT	I/O I/O Output	I/O port Clock in SIO mode 16-bit timer / event counter output
Function	8	PL3 INT0 TB3OUT	I/O Input Output	I/O port External interrupt pin 16-bit timer / event counter output
Function	9	PL4 TXD1 TB4OUT SDA3	I/O Output Output I/O	I/O port Serial channel sending serial data 16-bit timer / event counter output I2C data
Function	10	PL5 RXD1 TB5OUT SCL3	I/O Input Output I/O	I/O port Serial channel receiving serial data 16-bit timer / event counter output I2C clock
Function	11	PL6 SCLK1 TB6OUT $\overline{\text{CTS1}}$	I/O I/O Output Input	I/O port Serial channel clock pin 16-bit timer / event counter output Serial channel handshake input pin
Function	12	PL7 INT1 TB7OUT	I/O Input Output	I/O port External interrupt pin 16-bit timer / event counter output
PS	13	DVSS	-	GND pin

Table 1-1 Pin Names and Functions Sorted by Pin (2/6)

Type	Pin No.	PIn Name	Input / Output	Function
Function	14	PM0 SCLK2 TB1IN0 $\overline{CTS2}$	I/O I/O Input Input	I/O port Serial channel clock pin Inputting the 16-bit timer / event countercapture trigger Serial channel handshake input pin
Function	15	PM1 TXD2 TB1IN1	I/O Output Input	I/O port Serial channel sending serial data Inputting the 16-bit timer / event countercapture trigger
Function	16	PM2 RXD2 $\overline{ALARM}$	I/O Input Output	I/O port Serial channel receiving serial data Alarm output
Function	17	PM3 INT2 TB3OUT	I/O Input Output	I/O port External interrupt pin 16-bit timer / event counter output
Function	18	PM4 SCLK3 $\overline{CTS3}$	I/O I/O Input	I/O port Serial channel clock pin Serial channel handshake input pin
Function	19	PM5 TXD3	I/O Output	I/O port Serial channel sending serial data
Function	20	PM6 RXD3	I/O Input	I/O port Serial channel receiving serial data
Function	21	PM7 INT3	I/O Input	I/O port External interrupt pin
Function	22	PN0 TXD4	I/O Output	I/O port Serial channel sending serial data
Function	23	PN1 RXD4	I/O Input	I/O port Serial channel receiving serial data
Function	24	PN2 SCLK4 TB2IN0 $\overline{CTS4}$	I/O I/O Input Input	I/O port Serial channel clock pin Inputting the 16-bit timer / event countercapture trigger Serial channel handshake input pin
Function	25	PN3 INT4 TB2IN1 RMC0	I/O Input Input Input	I/O port External interrupt pin Inputting the 16-bit timer / event countercapture trigger Inputting signal to remote controller
Function	26	PP0 $\overline{CS2}$	I/O Output	I/O port Chip select pin
Function	27	PP1	I/O	I/O port
Function	28	PP2 $\overline{BLS0}$ SPDO	I/O Output Output	I/O port Byte lane pin SSP data output pin
Function	29	PP3 $\overline{BLS1}$ SPDI	I/O Output Input	I/O port Byte lane pin SSP data input pin

Table 1-1 Pin Names and Functions Sorted by Pin (3/6)

Type	Pin No.	PIn Name	Input / Output	Function
Function	30	PP4 $\overline{WE}$ SPCLK	I/O Output I/O	I/O port Write strobe pin SSP clock pin
Function	31	PP5 $\overline{OE}$ SPFSS	I/O Output I/O	I/O port Output enable pin SSP frame / slave select pin
Function	32	PP6 $\overline{ALE}$	I/O Output	I/O port Address latch enable pin
PS	33	DVDD3B	-	Power supply pin
PS	34	DVSS	-	GND pin
Function	35	PA0 AD0	I/O I/O	I/O port Address and data bus
Function	36	PA1 AD1	I/O I/O	I/O port Address and data bus
Function	37	PA2 AD2	I/O I/O	I/O port Address and data bus
Function	38	PA3 AD3	I/O I/O	I/O port Address and data bus
Function	39	PA4 AD4	I/O I/O	I/O port Address and data bus
Function	40	PA5 AD5	I/O I/O	I/O port Address and data bus
Function	41	PA6 AD6	I/O I/O	I/O port Address and data bus
Function	42	PA7 AD7	I/O I/O	I/O port Address and data bus
Function	43	PB0 AD8	I/O I/O	I/O port Address and data bus
Function	44	PB1 AD9	I/O I/O	I/O port Address and data bus
Function	45	PB2 AD10	I/O I/O	I/O port Address and data bus
Function	46	PB3 AD11	I/O I/O	I/O port Address and data bus
Function	47	PB4 AD12	I/O I/O	I/O port Address and data bus
Function	48	PB5 AD13	I/O I/O	I/O port Address and data bus
Function	49	PB6 AD14	I/O I/O	I/O port Address and data bus
Function	50	PB7 AD15	I/O I/O	I/O port Address and data bus

Table 1-1 Pin Names and Functions Sorted by Pin (4/6)

Type	Pin No.	Pin Name	Input / Output	Function
Function	51	PE0 A17 TB5IN0	I/O Output Input	I/O port Address bus Inputting the 16-bit timer / event countercapture trigger
Function	52	PE1 A18 TB5IN1	I/O Output Input	I/O port Address bus Inputting the 16-bit timer / event countercapture trigger
Function	53	PE2 A19 TB6IN0	I/O Output Input	I/O port Address bus Inputting the 16-bit timer / event countercapture trigger
Function	54	PE3 A20 TB6IN1	I/O Output Input	I/O port Address bus Inputting the 16-bit timer / event countercapture trigger
Function	55	PE4 A21 TXD0 CTXD	I/O Output Output Output	I/O port Address bus Serial channel sending serial data CAN sending data
Function	56	PE5 A22 RXD0 CRXD	I/O Output Input Input	I/O port Address bus Serial channel receiving serial data CAN receiving data
Function	57	PE6 A23 SCLK0 $\overline{\text{CTS0}}$	I/O Output I/O Input	I/O port Address bus Serial channel clock pin Serial channel handshake input pin
Function	58	PE7 INT5 SCOUT	I/O Input Output	I/O port External interrupt pin Internal clock output pin
PS	59	DVDD3B	-	Power supply pin
PS	60	DVSS	-	GND pin
Debug	61	SWDIO	I/O	Debug pin
Debug	62	SWCLK	I/O	Debug pin
Function/ Debug	63	PF0 TRACECLK	I/O Output	I/O port Debug pin
Function/ Debug	64	PF1 TRACEDATA0 SWV	I/O Output Output	I/O port Debug pin Debug pin
Function/ Debug	65	PF2 TRACEDATA1	I/O Output	I/O port Debug pin
Function/ Debug	66	PF3 TRACEDATA2	I/O Output	I/O port Debug pin

Table 1-1 Pin Names and Functions Sorted by Pin (5/6)

Type	Pin No.	Pin Name	Input / Output	Function
Function/ Debug	67	PF4 TRACEDATA3	I/O Output	I/O port Debug pin
Function	68	PG0 SDA1/SO1 TB7IN0	I/O I/O Input	I/O port Data in I2C mode / Data in SIO mode Inputting the 16-bit timer / event countercapture trigger
Function	69	PG1 SCL1/SI1 TB7IN1	I/O I/O Input	I/O port Clock in I2C mode / Data in SIO mode Inputting the 16-bit timer / event countercapture trigger
Function	70	PG2 SCK1 $\overline{CS0}$	I/O I/O Output	I/O port Clock in SIO mode Chip select pin
Function	71	PG3 INT6 $\overline{CS1}$	I/O Input Output	I/O port External interrupt pin Chip select pin
Function	72	PG4 SDA2/SO2 TB9IN0	I/O I/O Input	I/O port Data in I2C mode / Data in SIO mode Inputting the 16-bit timer / event countercapture trigger
Function	73	PG5 SCL2/SI2 TB9IN1	I/O I/O Input	I/O port Clock in I2C mode / Data in SIO mode Inputting the 16-bit timer / event countercapture trigger
Function	74	PG6 SCK2 USBPON $\overline{CS3}$	I/O I/O Output Output	I/O port Clock in SIO mode USB connect bus detection (VBus detection) Chip select pin
Function	75	PG7 INT7 $\overline{USBOC}$ $\overline{WDTOUT}$	I/O Input Input Output	I/O port External interrupt pin USB over current Watchdog timer output pin
PS	76	RVDD3	-	Power supply pin
Clock	77	XT1	Input	Connected to a low-speed oscillator.
Clock	78	XT2	Output	Connected to a low-speed oscillator.
PS	79	DVDD3A	-	Power supply pin
Clock	80	X1	Input	Connected to a high-speed oscillator.
PS	81	DVSS	-	GND pin
Clock	82	X2	Output	Connected to a high-speed oscillator.
PS	83	DVDD3B	-	Power supply pin
PS	84	DVSS	-	GND pin

Table 1-1 Pin Names and Functions Sorted by Pin (6/6)

Type	Pin No.	PIn Name	Input / Output	Function
Function	85	D+	I/O	USB data plus
Function	86	D-	I/O	USB data minus
Control	87	$\overline{\text{NMI}}$	Input	Non-maskable interrupt (note) With a noise filter (about 30ns (typical value))
Control	88	TEST1	-	TEST pin (note) TEST pin must be left OPEN.
Control	89	TEST2	-	TEST pin (note) TEST pin must be left OPEN.
Function	90	PI0 $\overline{\text{BOOT}}$	I/O Input	I/O port Setting boot pin TMPM363F10FG goes into single boot mode by sampling "Low" at the rising edge of a $\overline{\text{RESET}}$ pin.
Function	91	PI1 CEC	I/O I/O	I/O port CEC pin (note) Nch open drain port
PS	92	AVDD3	-	Power suppy pin for the AD converter (note) AVDD3 must be connected to power supply even if AD converter is not used.
Function	93	PJ0 AIN0	Input Input	Input port Analog input
Function	94	PJ1 AIN1	Input Input	Input port Analog input
Function	95	PJ2 AIN2	Input Input	Input port Analog input
Function	96	PJ3 AIN3 $\overline{\text{ADTRG}}$	Input Input Input	Input port Analog input External trigger input for AD converter
Function	97	PJ4 AIN4 KWUP0	Input Input Input	Input port Analog input Key-on wake-up pin
Function	98	PJ5 AIN5 KWUP1	Input Input Input	Input port Analog input Key-on wake-up pin
Function	99	PJ6 AIN6 KWUP2	Input Input Input	Input port Analog input Key-on wake-up pin
Function	100	PJ7 AIN7 KWUP3	Input Input Input	Input port Analog input Key-on wake-up pin

1.5 Pin Numbers and Power Supply Pins

Table 1-2 PIn Numbers and Power Supplies

Power supply	Voltage range	Pin No.	PIn mane
DVDD3B	2.7 to 3.6V (When USB is used : 3.0 to 3.6V)	33,59,83	PA,PB,PE,PF,PG,PI,PL,PM,PN,PP XT1,XT2,$\overline{\text{RESET}}$,$\overline{\text{NMI}}$,MODE
DVDD3A		79	X1,X2
AVDD3		92	PJ
RVDD3		76	－

2. Processor Core

The TX03 series has a high-performance 32-bit processor core (the ARM Cortex-M3 processor core). For information on the operations of this processor core, please refer to the "Cortex-M3 Technical Reference Manual" issued by ARM Limited.This chapter describes the functions unique to the TX03 series that are not explained in that document.

2.1 Information on the processor core

The following table shows the revision of the processor core in the TMPM363F10FG.

Refer to the detailed information about the CPU core and architecture, refer to the ARM manual "Cortex-M series processors" in the following URL:

http://infocenter.arm.com/help/index.jsp

Product Name	Core Revision
TMPM363F10FG	r2p0

2.2 Configurable Options

The Cortex-M3 core has optional blocks. The optional blocks of the revision r2p0 are ETM™, MPU and WIC. The following table shows the configurable options in the TMPM363F10FG.

Configurable Options	Implementation
FPB	Two literal comparators Six instruction comparators
DWT	Four comparators
ITM	Present
MPU	Absent
ETM	Present
AHB-AP	Present
AHB Trace Macrocell Interface	Absent
TPIU	Present
WIC	Absent
Debug Port	Serial wire

2.3 Exceptions/ Interruptions

Exceptions and interruptions are described in the following section.

2.3.1 Number of Interrupt Inputs

The number of interrupt inputs can optionally be defined from 1 to 240 in the Cortex-M3 core.

TMPM363F10FG has 66 interrupt inputs. The number of interrupt inputs is reflected in <INTLINESNUM [4:0]> bit of NVIC register. In this product, if read <INTLINESNUM[4:0]> bit, 0x01 is read out.

2.3.2 Number of Priority Level Interrupt Bits

The Cortex-M3 core can optionally configure the number of priority level interrupt bits from 3 bits to 8 bits.

TMPM363F10FG has 3 priority level interrupt bits. The number of priority level interrupt bits is used for assigning a priority level in the interrupt priority registers and system handler priority registers.

2.3.3 SysTick

The Cortex-M3 core has a SysTick timer which can generate SysTick exception.

For the detail of SysTick exception, refer to the section of "SysTick" in the exception and the register of SysTick in the NVIC register.

2.3.4 SYSRESETREQ

The Cortex-M3 core outputs SYSRESETREQ signal when <SYSRESETREQ> bit of Application Interrupt and Reset Control Register are set.

TMPM363F10FG provides the same operation when SYSRESETREQ signal are output.

Note:The reset operation by <SYSRESETREQ> can not used while in SLOW mode.

2.3.5 LOCKUP

When irreparable exception generates, the Cortex-M3 core outputs LOCKUP signal to show a serious error included in software.

TMPM363F10FG does not use this signal. To return from LOCKUP status, it is necessary to use non-maskable interruput (NMI) or reset.

2.3.6 Auxiliary Fault Status register

The Cortex-M3 core provides auxiliary fault status registers to supply additional system fault information to software.

However, TMPM363F10FG is not defined this function. If auxiliary fault status register is read, always "0x0000_0000" is read out.

2.4   Events

The Cortex-M3 core has event output signals and event input signals. An event output signal is output by SEV in-struction execution. If an event is input, the core returns from low-power consumption mode caused by WFE instruc-tion.

TMPM363F10FG does not use event output signals and event input signals. Please do not use SEV instruction and WFE instruction.

2.5   Power Management

The Cortex-M3 core provides power management system which uses SLEEPING signal and SLEEPDEEP sig-nal. SLEEPDEEP signals are output when <SLEEPDEEP> bit of System Control Register is set.

These signals are output in the following circumstances:

-Wait-For-Interrupt (WFI) instruction execution

-Wait-For-Event (WFE) instruction execution

-the timing when interrupt-service-routine (ISR) exit in case that <SLEEPONEXIT> bit of System Control Reg-ister is set.

TMPM363F10FG does not use SLEEPDEEP signal so that <SLEEPDEEP> bit must not be set. And also event signal is not used so that please do not use WFE instruction.

For detail of power management, refer to the Chapter "Clock/Mode control."

2.6   Exclusive access

In Cortex-M3 core, the DCode bus system supports exclusive access. However TMPM363F10FG does not use this function.

3. Debug Interface

3.1 Specification Overview

The TMPM363F10FG contains the Serial Wire Debug Port (SW-DP) unit for interfacing with the debugging tools and the Embedded Trace Macrocell™ (ETM) unit for instruction trace output. Trace data output to the dedicated pins (TRACEDATA[3:0]) via the on-chip Trace Port Interface Unit (TPIU).

For details about SW-DP, ETM and TPIU, refer to "Cortex-M3 Technical Reference Manual".

3.2 SW-DP

SW-DP supports the two-pin Serial Wire Debug Port (SWCLK, SWDIO).

3.3 ETM

ETM supports four data signal pin (TRACEDATA[3:0]), one clock signal pin (TRACECLK) and trace output from SWV.

3.4   Pin functions

The debug interface pin can also be used as general purpose port (PF0 to PF4).

Table 3-1 SW-DP,ETM Debug Function

SW-DP pin name	Port name	SW debug function	
		I / O	Comments
SWDIO	–	I / O	Serial Wire Data Input/Output (Always pull-up)
SWCLK	–	Input	Serial Wire Clock (Always pull-down)
TRACECLK	PF0	Output	TRACE Clock Output
TRACEDATA0 / SWV	PF1	Output	TRACE DATA Output0 / Serial Wire Viewer Output
TRACEDATA1	PF2	Output	TRACE DATA Output1
TRACEDATA2	PF3	Output	TRACE DATA Output2
TRACEDATA3	PF4	Output	TRACE DATA Output3

After reset, PF0 to PF4 pins are configured as general purpose ports. The functions of the debug interface pins need to be programmed as required.

Table 3-2 summarizes the debug interface pin functions and related port settings after reset.

Table 3-2 The Debug Interface Pins functions and Related Port Setting after Reset

Port Name (Bit Name)	Debug Function	Value of related port setting after reset				
		Function (PxFR)	Input (PxIE)	Output (PxCR)	Pull-up (PxPUP)	Pull-down (PxPDN)
PF0	TRACECLK	0	0	0	0	–
PF1	TRACEDATA0 / SWV	0	0	0	0	–
PF2	TRACEDATA1	0	0	0	0	–
PF3	TRACEDATA2	0	0	0	0	–
PF4	TRACEDATA3	0	0	0	0	–

− : Don't care

TOSHIBA

3.5 Peripheral Functions in Halt Mode

When the Cortex-M3 core enters in the halt mode, the watch-dog timer (WDT) automatically stops. Other peripheral functions continue operate.

3.6 Reset Vector Break

TMPM330FDFG/FYFG/FWFG is prohibited from transmission with debug tools while reset caused by $\overline{\text{RESET}}$ pin is effective.When setting a stop by using reset vector, set the following procedure after reset; set break points from the debug tools, then set the application interrupt and the <SYSRESETREQ> bit of the reset control register to reset again.

Note:Do not reset with <SYSRESETREQ> in SLOW mode.

3.7 Connection with a Debug Tool

Concerning a connection with debug tools, refer to manufactures recommendations.

Debug interface pins contain a pull-up resistor and a pull-down resistor.When debug interface pins are connected with external pull-up or pull-down, please pay attention to input level.

4. Memory Map

4.1 Memory Map

The memory maps for the TMPM363F10FG are based on the ARM Cortex-M3 processor core memory map.

The internal ROM is mapped to the code of the Cortex-M3 core memory, the internal RAM is mapped to the SRAM region and the special function register (SFR) is mapped to the peripheral region respectively.

The special function register (SFR) indicates I/O ports and control registers for the peripheral function. The SRAM and SFR regions are all included in the bit-band region.

The CPU register region is the processor core's internal register region.

For more information on the each region, see the "Cortex-M3 Technical Reference Manual".

Note that access to regions indicated as "Fault" causes a memory fault if memory faults are enabled or a hard fault if memory faults are disabled. Do not access the vendor-specific region.

4.1.1   Memory map of the TMPM363F10FG

Figure 4-1 shows the memory map of the TMPM363F10FG.

Address	Region
0xFFFF_FFFF	Vender-Specific
0xE010_0000	
0xE00F_FFFF	CPU Register Region
0xE000_0000	
	Fault
0x63FF_FFFF	External Bus Area
0x6000_0000	
	Fault
0x41FF_FFFF / 0x41FF_F000	SFR
	Fault
0x400F_4FFF / 0x400C_0000	SFR
	Fault
0x4004_1FFF / 0x4004_0000	SFR
	Fault
0x4000_5FFF / 0x4000_0000	SFR
	Fault
0x2000_FFFF / 0x2000_E000	Backup RAM (8K)
0x2000_DFFF / 0x2000_0000	Internal RAM (56K)
	Fault
0x000F_FFFF / 0x0000_0000	Internal ROM (1024K)

Figure 4-1 Memory map

4.2   SFR area detail

This section contains the list of addresses in the SFR are (0x4000_0000 to 0x4000_5FFF, 0x4004_0000 to 0x4004_1FFF, 0x400C_0000 to 0x400F_4FFF, 0x41FF_F000 to 0x41FF_FFFF) assigned to peripheral function.

Access to the Reserved areas in the Table 4-1 is prohibited. As for the SFR area, reading the areas not described in the Table 4-1 yields undefined value. Writing these area is ignored.

Table 4-1 SFR area detail

Start Address	End Address	Peripheral	Reserved		
0x4000_0000	0x4000_0FFF	DMAC	0x4000_0028	to	0x4000_002F
			0x4000_0034	to	0x4000_0037
			0x4000_0500	to	0x4000_050F
			0x4000_0FE0	to	0x4000_0FFF
0x4000_1000	0x4000_1FFF	SMC	0x4000_1000	to	0x4000_1003
			0x4000_1008	to	0x4000_100F
			0x4000_1020	to	0x4000_1023
			0x4000_1200	to	0x4000_1207
			0x4000_1E00	to	0x4000_1E0B
			0x4000_1FE0	to	0x4000_1FFF
0x4000_2000	0x4000_2FFF	CAN			
0x4000_3000	0x4000_3FFF	USB	0x4000_3058	to	0x4000_305F
0x4000_4000	0x4000_5FFF	Reserved			
0x4004_0000	0x4004_0FFF	SSP	0x4004_0028	to	0x4004_0FFF
0x4004_1000	0x4004_1FFF	Reserved			
0x400C_0000	0x400C_FFFF	Port	0x400C_0200	to	0x400C_03FF
			0x400C_0700	to	0x400C_07FF
			0x400C_0A00	to	0x400C_0AFF
			0x400C_0E00	to	0x400C_0EFF
0x400D_0000	0x400D_FFFF	Timer B (16ch)			
0x400E_0000	0x400E_04FF	I2C/SIO(3ch) / I2C(1ch)	0x400E_0400	to	0x400E_041F
0x400E_1000	0x400E_1BFF	SIO/UART(5ch)	0x400E_1500	-	0x400E_1BFF
0x400E_2000	0x400E_203F	CEC			
0x400E_3000	0x400E_31FF	RMC(1ch)	0x400E_3100	to	0x400E_313F
0x400F_0000	0x400F_005B	ADC(8ch)	0x400F_001C	to	0x400F_001F
			0x400F_0024	to	0x400F_002F
0x400F_1000	0x400F_108F	KWUP	0x400F_1010	to	0x400F_107F
0x400F_2000	0x400F_2007	WDT			
0x400F_3000	0x400F_300F	RTC	0x400F_300D		
0x400F_4000	0x400F_4037	CG	0x400F_4028	to	0x400F_402F
			0x400F_4036	to	0x4000_4FFF
0x41FF_F000	0x41FF_F03F	FLASH	0x41FF_F000	to	0x41FF_F007
			0x41FF_F014	to	0x41FF_F017
			0x41FF_F018	to	0x41FF_F01B
			0x41FF_F024	to	0x41FF_F02C
			0x41FF_F033	to	0x41FF_F037
0x41FF_F040	0x41FF_F057	Reserved			
0x41FF_F058	0x41FF_F05B	RAMWAIT			
0x41FF_F060	0x41FF_F093	Reserved			
0x41FF_F0A0	0x41FF_F0BB	Reserved			
0x41FF_F100	0x41FF_F103	SMCMOD			

5. Reset

The TMPM363F10FG has three reset sources: an external reset pin ($\overline{\text{RESET}}$), a watchdog timer (WDT) and the setting <SYSRESETREQ> in the Application Interrupt and Reset Control Register.

For reset from the WDT, refer to the chapter on the WDT.

For reset from <SYSRESETREQ>, refer to "Cortex-M3 Technical Reference Manual".

Note: Do not reset with <SYSRESETREQ> in SLOW mode.

5.1 Cold reset

The power-on sequence must consider the time for the internal regulator and oscillator to be stable. In the TMPM363F10FG, the internal regulator requires at least 700 µs to be stable.

The time required to achieve stable oscillation varies with system. At cold reset, the external reset pin must be kept "Low" for a duration of time sufficiently long enough for the internal regulator and oscillator to be stable.

After the external reset ($\overline{\text{RESET}}$) signal is released, the internal reset signal remains asserted for a further 400µs.

Figure 5-1 shows the power-on sequence.
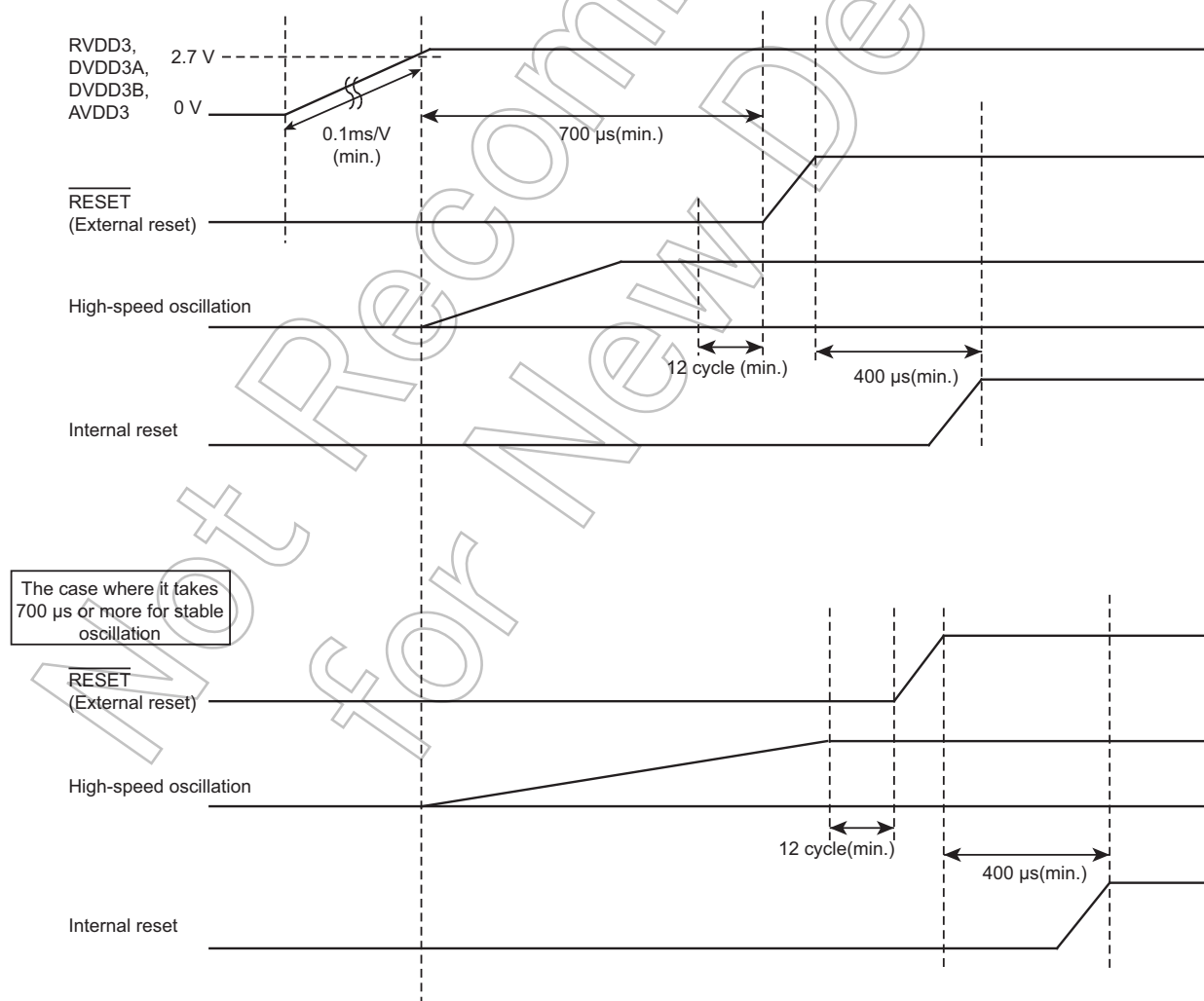
Figure 5-1 Cold Reset Sequence

Note 1: The power supply must be raised (from 0V to 2.7V) at a speed of 0.1ms/V or slower.

Note 2: Turn on the power while the $\overline{\text{RESET}}$ pin is fixed to "Low". When all the power supplies are stabilized within operating voltage, release the reset.

5.2   Warm reset

5.2.1   Reset period

As a precondition, ensure that the power supply voltage is within the operating range and the internal high-frequency oscillator is providing stable oscillation.

To reset the TMPM363F10FG, assert the $\overline{\text{RESET}}$ signal (active low) for a minimum duration of 12 system clocks.

After the external reset ($\overline{\text{RESET}}$) signal is released, the internal reset signal remains asserted for a further 400μs.

5.3   After reset

A warm reset initializes the majority of the Cortex-M3 processor core's system control registers and internal function registers.

The processor core's system debug components (FPB, DWT, ITM) register, the clock generator's CGRSTFLG register and the FCSECBIT register are initialized by a only cold reset.

After reset, the PLL multiplication circuit is inactive and must be enabled in the CGPLLSEL register if needed.

When the reset exception handling is completed, the program branches to the reset interrupt service routine.

Note: The reset operation may alter the internal RAM state.

TOSHIBA

6.   Clock / Mode Control

6.1   Features

The clock/mode control block enables to select clock gear, prescaler clock and warm-up of the PLL clock multi-plication circuit and oscillator.

There is also the low power consumption mode which can reduce power consumption by mode transitions.

This chapter describes how to control clock operating modes and mode transitions.

The clock/mode control block has the following functions:

- Controls the system clock
- Controls the prescaler clock
- Controls the PLL multiplication circuit
- Controls the warm-up timer

In addition to NORMAL mode, the TMPM363F10FG can operate low power mode to reduce power consumption according to its usage conditions.

6.2 Registers

6.2.1 Register List

The following table shows the CG-related registers and addresses.

Base Address = 0x400F_4000

Register name		Address (Base+)
System control register	CGSYSCR	0x0000
Oscillation control register	CGOSCCR	0x0004
Standby control register	CGSTBYCR	0x0008
PLL selection register	CGPLLSEL	0x000C
System clock selection register	CGCKSEL	0x0010

6.2.2 CGSYSCR (System control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	USBHRES	-	-	FCSTOP	-	-	SCOSEL	
After reset	0	0	0	0	0	0	0	1
	15	14	13	12	11	10	9	8
bit symbol	-	-	FPSEL1	FPSEL0	-	PRCK		
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	GEAR		
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-24	−	R	Read as "0".
23	USBHRES	R/W	USB HOST Controller Reset 0: Reset release 1: Reset It is possible to reset USB HOST Controller.
22-21	−	R	Read as "0".
20	FCSTOP	R/W	Stops AD converter clock 0: Operated 1: Stopped It is possible to stop AD converter clock. After reset, AD converter clock is operated. If this bit is set to "1", make sure to confirm that AD conversion is finished or stopped.
19-18	−	R	Read as "0".
17-16	SCOSEL[1:0]	R/W	SCOUT output 00: fs 01: fsys/2 10: fsys 11: φT0 Enables to output the specified clock from SCOUT pin.
15-14	−	R	Read as "0".
13	FPSEL1	R/W	Selects φT0 source clock 0: clock selected by <FPSEL0> 1: fs
12	FPSEL0	R/W	Selects fperiph source clock 0: fgear 1: fc
11	−	R	Read as "0".
10-8	PRCK[2:0]	R/W	Prescaler clock 000: fperiph 100: fperiph/16 001: fperiph/2 101: fperiph/32 010: fperiph/4 110: Reserved 011: fperiph/8 111: Reserved Specifies the prescaler clock to peripheral I/O.
7-3	−	R	Read as "0".
2-0	GEAR[2:0]	R/W	High-speed gear clock (fgear) 000: fc 100: fc/2 001: Reserved 101: fc/4 010: Reserved 110: fc/8 011: Reserved 111: Reserved

6.2.3   CGOSCCR (Oscillation control register)

	31	30	29	28	27	26	25	24
bit symbol	WUPT							
After reset	1	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	WUPT				-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	WUPTL		-	-	-	-	XTEN	XEN
After reset	0	0	0	0	0	0	1	1
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	WUPSEL	PLLON	WUEF	WUEON
After reset	0	0	1	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-20	WUPT[11:0]	R/W	Specifies count time of the warm-up timer. Warm-up counter value for high-speed Warm-up counter value for low-speed (Upper 12 bits) The warm-up counter for high-speed is 16-bit counter and one for low-speed is 18-bit counter. Lower 4 bits of the both counter are masked. Upper 12 bits for high-speed, lower 14 bits for low-speed are compared with warm-up counter.
19-16	–	R	Read as "0".
15-14	WUPTL[1:0]	R/W	Specifies count time of the warm-up timer. Warm-up counter value for low-speed (Lower 2 bits)
13-12	–	R/W	Write as "0".
11-10	–	R	Read as "0".
9	XTEN	R/W	Low-speed oscillator 0: Stop 1:Oscillation
8	XEN	R/W	High-speed oscillator 0: Stop 1:Oscillation
7-4	–	R/W	Write as "0011".
3	WUPSEL	R/W	Warm-up counter 0: High-speed (fosc) 1: Low-speed (fs) Specifies the oscillator to warm-up. A clock generated by the specified oscillator is used for the warm-up timer count.
2	PLLON	R/W	PLL operation 0: Stop 1: Oscillation
1	WUEF	R	Status of warm-up timer (WUP) 0: Warm-up completed. 1: Warm-up operation Enable to monitor the status of the warm-up timer.
0	WUEON	W	Operation of warm-up timer 0: don't care 1: Starting warm-up Enables to start the warm-up timer. Read as "0".

Note 1: Regarding to warm-up time, refer to "6.3.4 Warm-up function".

Note 2: After setting PLL multiplying value, to keep CGOSCCR<PLLON> = "0" (PLL stop) over 100 µs is needed as the PLL initializing stable time.

6.2.4   CGSTBYCR (Standby control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	PTKEEP	DRVE
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	RXTEN	RXEN
After reset	0	0	0	0	0	0	0	1
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-		STBY	
After reset	0	0	0	0	0	0	1	1

Bit	Bit Symbol	Type	Function
31-18	−	R	Read as "0".
17	PTKEEP	R/W	I/O port control in the Backup mode. 0: Output the contents of output latch. 1: Hold the port state when CGSTBYCR<PTKEEP> is changed from "0" to "1".
16	DRVE	R/W	Pin status in STOP mode (note) 0: Inactive 1: Active
15-10	−	R	Read as "0".
9	RXTEN	R/W	Low-speed oscillator operation after releasing the STOP mode. 0: Stop 1: Oscillation
8	RXEN	R/W	High-speed oscillator operation after releasing the STOP mode. 0: Stop 1: Oscillation
7-3	−	R	Read as "0".
2-0	STBY[2:0]	R/W	Low power consumption mode 000: Reserved 001: STOP 010: SLEEP 011: IDLE2 100: Reserved 101: BACKUP STOP 110: BACKUP SLEEP 111: IDLE1

Note 1: I/O Ports which hold their state when CGSTBYCR<PTKEEP> is changed from "0" to "1" are all port except for port I, J, L, M and N. In regarding to release CGSTBYCR<PTKEEP>, refer to section of "Backup module".

Note 2: The value which is shown reserved in above table must be not set.

6.2.5   CGPLLSEL (PLL Selection Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	RS				-	IS		C2S
After reset	0	1	1	1	0	0	1	0
	7	6	5	4	3	2	1	0
bit symbol	ND					-	-	PLLSEL
After reset	0	0	0	1	1	1	1	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as "0".
15-12	RS[3:0]	R/W	Clock multiplied by PLL 0111: 4 times 1010: 8 times Others: Reserved
11	–	R	Read as "0".
10-9	IS[1:0]	R/W	Clock multiplied by PLL 00: 8 times 01: 4 times Others: Reserved
8	C2S	R/W	Clock multiplied by PLL 0: 4 times 1: 8 times
7-3	ND[4:0]	R/W	Clock multiplied by PLL 00011: 4 times 00111: 8 times Others: Reserved
2-1	–	R/W	Write as "1".
0	PLLSEL	R/W	Use PLL 0: fosc 1: f_{PLL} Specifies use or disuse of the clock multiplied by the PLL. fosc is automatically set after reset. Resetting is required when using the PLL.

Note 1:   Select PLL multiplying value which is shown Table 6-1.

Note 2:   Select PLL multiplying value when CGOSCCR<PLLON> = "0" (PLL stop).

Note 3:   After setting PLL multiplying value, to keep (CGOSCCR<PLLON> = "0" (PLL stop) over 100 μs is needed as the PLL initializing stable time.

6.2.6 CGCKSEL (System clock selection register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	SYSCK	SYSCKFLG
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as "0".
1	SYSCK	R/W	System clock (fsys) 0: fgear 1: fs Enable to specify the system clock. Before modifying <SYSCK>, fgear and fs must be stable.
0	SYSCKFLG	R	System clock status 0: fgear 1: fs Shows the status of the system clock. Switching the oscillator with <SYSCK> generates time lag to complete. If the output of the oscillator specified in <SYSCK> is read out by <SYSCLKFLG>, the switching has been completed.

6.3 Clock control

6.3.1 Clock Type

Each clock is defined as follows :

fosc	: Clock input from the X1 and X2 pins
fs	: Clock input from the XT1 and XT2 pins (low-speed clock)
f_{PLL}	: Clock quadrupled or octupled by PLL
fc	: Clock specified by CGPLLSEL<PLLSEL> (high-speed clock)
fgear	: Clock specified by CGSYSCR<GEAR[2:0]> (gear clock)
fsys	: Clock specified by CGCKSEL<SYSCK> (system clock)
fperiph	: Clock specified by CGSYSCR<FPSEL0>
φT0	: Clock specified by CGSYSCR<FPSEL1> (Prescaler clock)

The gear clock fgear and the prescaler clock φT0 are dividable as follows.

Gear clock	: fc, fc/2, fc/4, fc/8
Prescaler clock	: fs, fperiph, fperiph/2, fperiph/4, fperiph/8, fperiph/16, fperiph/32

6.3.2 Initial Values after Reset

Reset operation initializes the clock configuration as follows.

High-speed oscillator	: oscillating
Low-speed oscillator	: oscillating
PLL (Phase locked loop circuit)	: stop
High-speed clock gear	: fc (no frequency dividing)

Reset operation causes all the clock configurations excluding the low-speed clock (fs) to be the same as fosc.

fc = fosc

fsys = fosc

φT0 = fosc

6.3.3 Clock system Diagram

Figure 6-1 shows the clock system diagram.

The input clocks selector shown with an arrow are set as default after reset.
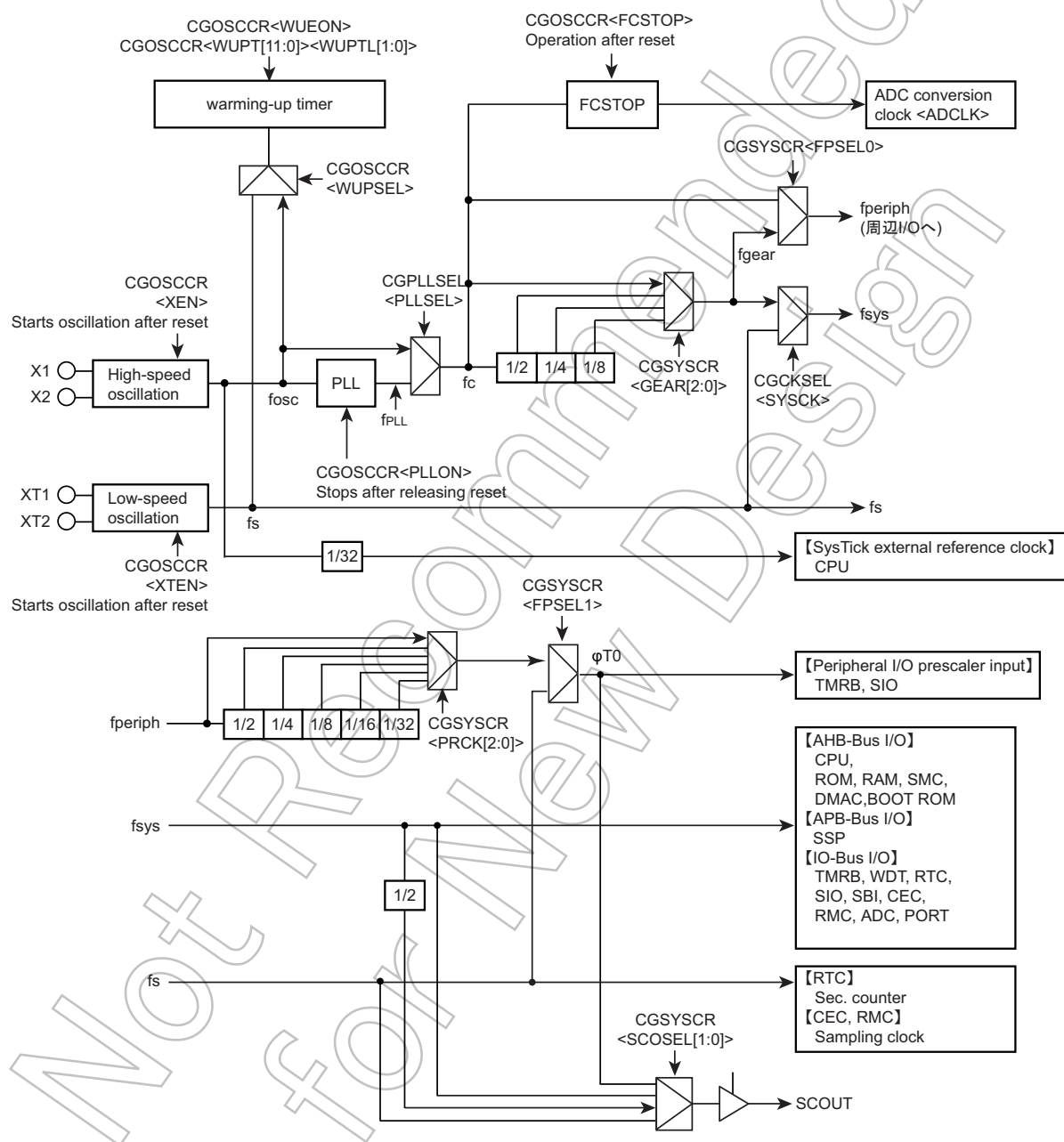
Figure 6-1 Clock Block Diagram

6.3.4   Warm-up function

The warm-up function secures the stability time for the oscillator and the PLL with the warm-up timer.

Refer to "6.6.8 Warm-up" for more detail.

How to use the warm-up function is described.

1.   Specify the count up clock

Specify the count up clock for the warm-up counter in the CGOSCCR<WUPSEL>.

2.   Specify the warm-up counter value

The warm-up time can be calculated by following formula with round 4 bit off, set to bit of <WUPT [11:0]><WUPTL[1:0]>

$$\text{Number of warm-up cycles} = \frac{\text{Warm-up time}}{\text{Warm-up clock cycle}}$$

3.   Start warm-up function and confirm the completion of warm-up

When CGOSCCR<WUEON> is set to "1", the warm-up start a count up. The CGOSCCR <WUEF> is used to confirm the start and completion of warm-up. <WUEF>=1 shows under warm-up and <WUEF>=0 shows completion of warm-up.

For the clock changing, current system clock can be monitored in CGCKSEL<SYSCKFLG>.

The following shows the warm-up setting and example.

1. In transition from SLOW mode to NORMAL mode, set 5ms for warm-up time when using 8MHz oscillator for high-frequency

    The value of warm-up conter is shown below.

$$\frac{\text{Warm-up time}}{\text{Warm-up clock cycle}} = \frac{5\text{ms}}{1/8\text{MHz}} = 40{,}000 \text{ cycles} = 0\text{x}9\text{C}40$$

    The warm-up counter for high-frequency is 16 bits and the lower 4 bits of these is ignored. Therefore, the upper 12 bits of 0x9C40, 0x9C4 is set to CGOSCCR<WUPT[11:0]>.

<example> Transition from SLOW mode to NORMAL mode

	CGOSCCR<WUPT[11:0]> = "0x9C4"	: Warm-up time setting
↻	Read CGOSCCR<WUPT[11:0]>	: Confirm warm-up time reflecting
	CGOSCCR<XEN>="1"	: Enable high-speed oscillation (fosc)
	CGOSCCR<WUEON>="1"	: Enable warm-up counting (WUP)
↻	Read CGOSCCR<WUEF>	: Wait for "0" (end of WUP)
	CGOSCCR<SYSCK>="0"	: system clock changed to high-speed (fgear)
↻	Read CGOSCCR<SYSCKFLG>	: Wait for "0" (the current clock is fgear)
	CGOSCCR<XTEN>="0"	: Disable the low-speed oscillation (fs) (In dual clock mode, it's not required.)

2. Intransition from NORMAL mode to SLOW mode, set 1s for warm-up time when using 32.768kHz oscillator for low-frequency

    The value of warm-up conter is shown below.

$$\frac{\text{Warm-up time}}{\text{Warm-up clock cycle}} = \frac{1\text{s}}{1/32.768\text{kHz}} = 32{,}768 \text{ cycles} = 0\text{x}8000$$

    The warm-up counter for low-frequency is 18 bits and the lower 4 bits of these is ignored. Therefore, the upper 14 bits of 0x8000, 0x0800 is set to CGOSCCR<WUPT[11:0]><WUPTL[1:0]>.

<example> Transmission from NORMAL mode to SLOW mode

CGOSCCR<WUPT[11:0]> = "0x200"	: Warm-up time setting (Upper 12 bits)
CGOSCCR<WUPTL[1:0]> = "00"	: Warm-up time setting (Lower 2 bits)

Read CGOSCCR<WUPT><WUPTL>   : Check warm-up time setting

CGOSCCR<XTEN>="1"	: Enable low-speed oscillation (fs)
CGOSCCR<WUPSEL>="1"	: Select XT1 for warm-up clock
CGOSCCR<WUEON>="1"	: Enable warm-up counting (WUP)

Read CGOSCCR<WUEF>   : Wait for "0" (end of WUP)

CGOSCCR<SYSCK>="1"   : system clock changed to low-speed (fs)

Read CGOSCCR<SYSCKFLG>   : Wait for "1" (the current clock is fs)

CGOSCCR<XEN>="0"   : Disable the high-speed oscillation (fc)
(In dual clock mode, it's not required.)

Note 1: It is not required the warm-up time in using the external clock to be stabled.

Note 2: The warm-up timer operates according to the oscillation clock, and it may contain errors if there is any fluctuation in the oscillation frequency. Therefore, the warm-up time should be taken as approximate time.

Note 3: After setting warm-up count value to OSCCR<WUPT><WUPTL>, wait until confirming of the value to be reflected, then change to the standby mode by WFI instruction.

Note 4: When switching the system clock, ensure that the switching has been completed by reading the CGCKSEL<SYSCKFLG>.

6.3.5 Clock Multiplication Circuit (PLL)

This circuit outputs the f_{PLL} clock that is quadruple / octuple of the high-speed oscillator output clock (fosc). As a result, the input frequency to oscillator can be low, and the internal clock be made high-speed.

6.3.5.1 How to configure the PLL function

The PLL is disabled after reset.

To enable the PLL, set CGPLLSEL<RS[3:0]><IS[1:0]><C2S><ND[4:0]> to multiplying value when CGOSCCR<PLLON> is "0". And set <PLLON> to "1" after 100μs for initialize time of PLL. After 200μs for lock-up time elapses, set CGPLLSEL<PLLSEL> to "1", f_{PLL} which is multipupied by 4 or 8 from fosc is used.

The PLL requires a certain amount of time to be stabilized, which should be secured using the warm-up function or other methods.

As for the 4 or 8 multiplying value, only the following setting are permitted.

Multiplying	<RS[3:0]>	<IS[1:0]>	<C2S>	<ND[4:0]>
4	0111	00	0	0_0011
8	1010	01	1	0_0111

6.3.5.2 Changing PLL multiplying

When number of multiplication is changed, firstly set "0" to CGPLLSEL<PLLSEL>. Secondly read CGPLLSEL<PLLSEL> to check the setting in which multiplication clock is not used (CGPLLSEL<PLLSEL>="0"). Thirdly, set "0" to <PLLON>.

Modify CGPLLSEL<RS[3:0]><IS[1:0]><C2S><ND[4:0]> to multiplying value. And set <PLLON> to "1" after 100μs for initilize time of PLL. After 200μs for lock-up time elapses, Set CGPLLSEL<PLLSEL> to "1".

6.3.5.3   Start PLL sequence

```
┌─────────────────────────────────────────────────┐
│ Initial value after reset                         │
│ CGPLLSEL<PLLSEL> = "0" (Not use PLL)              │
│ CGOSCCR<PLLON> = "0" (PLL stop)                   │
│ CGPLLSEL<RS><IS><C2S><ND> = "4 times"             │
└─────────────────────────────────────────────────┘
                        ⇩
┌─────────────────────────────────────────────────┐
│ PLL multiplying value setting                     │
│ CGPLLSEL<RS><IS><C2S><ND> = multiplying value     │
└─────────────────────────────────────────────────┘
                        ⇩
┌─────────────────────────────────────────────────┐
│ Initialize time                                   │
│ takes 100µs for initialize time of PLL            │
└─────────────────────────────────────────────────┘
                        ⇩
┌─────────────────────────────────────────────────┐
│ PLL operation                                     │
│ CGOSCCR<PLLON> = "1" (PLLì starts)                │
└─────────────────────────────────────────────────┘
                        ⇩
┌─────────────────────────────────────────────────┐
│ Lock-up time                                      │
│ takes 200µs for lock-up time                      │
└─────────────────────────────────────────────────┘
                        ⇩
┌─────────────────────────────────────────────────┐
│ PLL setting                                       │
│ CGPLLSEL<PLLSEL> = "1" (PLL used)                 │
└─────────────────────────────────────────────────┘
                        ⇩
       Possible to use the multiplied system clock
```
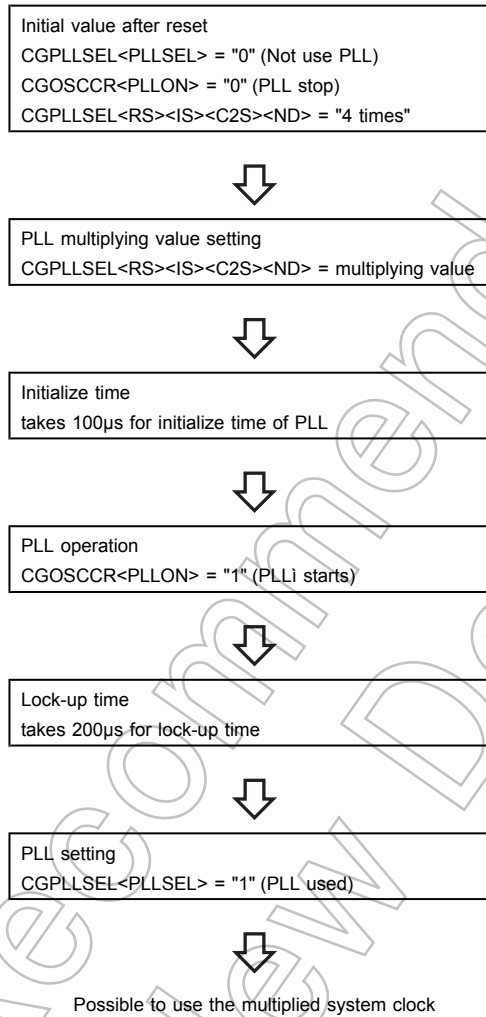
6.3.5.4 Multiplying value change sequence

```
┌─────────────────────────────────────────────┐
│ PLL setting                                   │
│ CGPLLSEL<PLLSEL> = "0" (Not use PLL)          │
│ Confirm that CGPLLSEL<PLLSEL> is "0".         │
└─────────────────────────────────────────────┘
                       ⇩
┌─────────────────────────────────────────────┐
│ PLL operation                                 │
│ CGOSCCR<PLLON> = "0" (PLL stops)              │
└─────────────────────────────────────────────┘
                       ⇩
┌─────────────────────────────────────────────┐
│ PLL multiplying value setting                 │
│ CGPLLSEL<RS><IS><C2S><ND> = multiplying       │
│ value                                         │
└─────────────────────────────────────────────┘
                       ⇩
┌─────────────────────────────────────────────┐
│ Initialize time                               │
│ takes 100µs for initialize time of PLL        │
└─────────────────────────────────────────────┘
                       ⇩
┌─────────────────────────────────────────────┐
│ PLL operation                                 │
│ CGOSCCR<PLLON> = "1" (PLL starts)             │
└─────────────────────────────────────────────┘
                       ⇩
┌─────────────────────────────────────────────┐
│ Lock-up time                                  │
│ takes 200µs for lock-up time                  │
└─────────────────────────────────────────────┘
                       ⇩
┌─────────────────────────────────────────────┐
│ PLL setting                                   │
│ CGPLLSEL<PLLSEL> = "1" (PLL used)             │
└─────────────────────────────────────────────┘
                       ⇩
          Possible to use the multiplied system clock
```
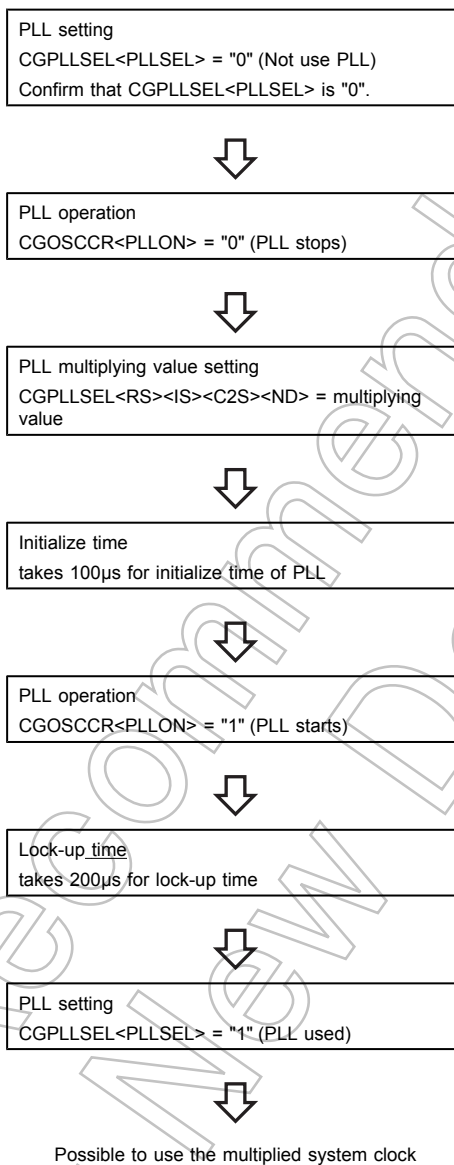
6.3.6 System Clock

The TMPM363F10FG offers two selectable system clocks: low-speed or high-speed.

6.3.6.1 High-speed clock

The high-speed clock is used by multiplying.

Source clock		Frequency	Using PLL
High-speed oscillation	Oscillator	8 to 16MHz	Not use, 4 or 8 multiplying
	External clock	8 to 16MHz	

Note: Regarding PLL multiplying and the frequency of high-speed oscillation, refer to Table 6-1.

The clock devided by CGSYSCR<GEAR[2:0]> is used for a system clock. CGSYSCR<GEAR[2:0]> can be modified in operation, a several time is needed for changing the clock.

The setting example of operation frequency depended on PLL multiplying and the clock gear setting is shown bellow.

Table 6-1 The setting example of operation frequency depended on PLL multiplying and the clock gear setting (Unit : MHz)

Input freq. X1, X2	PLL Multiplying	Min. operating freq.	Max. operating freq.	After reset (PLL = OFF, CG = 1/1)	Clock gear (CG) PLL = @ ON				Clock gear (CG) PLL = @ OFF			
					1/1	1/2	1/4	1/8	1/1	1/2	1/4	1/8
8	4	1	32	8	32	16	8	4	8	4	2	1
9			36	9	36	18	9	4.5	9	4.5	2.25	1.13
10			40	10	40	20	10	5	10	5	2.5	1.25
12			48	12	48	24	12	6	12	6	3	1.5
13.5			54	13.5	54	27	13.5	6.75	13.5	6.75	3.37	1.69
16.0			64	16.0	64	32	16	8	16	8	4	2
8	8	1	64	8	64	32	16	8	8	4	2	1

Note 1: When USB is used, operating frequency must be 48MHz.

Note 2: When ADC is used, ADCLK shold be equal or less than 40MHz by setting ADCLK<ADCLK>.

6.3.6.2 Low-speed clock

The frequency which can be inputted from XT1 and XT2 shown below.

Table 6-2 Range of Low Speed Frequency

Input frequency range	Minimum operating frequency	Maximum operating frequency
30 to 34 (kHz)	30 kHz	34 kHz

Note: CEC uses fs for a sampling clock. If CEC is used, fs must be within 32.768kHz±4%.

6.3.6.3   Setting system clock

To select system clock is used by CGOSCCR and CGCKSEL. After selecting system clock, set CGPLLSEL and CGOSCCR for PLL and CGSYSCR for clock gear.

How to set system clock is shown below.

How to set system clock

```
Initial value after reset
CGOSCCR<XEN> = "1"
(Enable high-speed oscillation)
CGOSCCR<XTEN> = "1"
(Enable low-speed oscillation)
CGCKSEL<SYSCK> = "0"
(Selects fgear for system clock)
CGOSCCR<PLLON> = "0" (stops PLL)
CGPLLSEL<PLLSEL> = "0" (fosc used)
CGSYSCR<GEAR> = "000" (Not divided)
```

```
Case of using high-speed clock
It is possible to use it as it is.
```

```
Case of low-speed clock
CGOSCCR<WUPSEL> = "1"
(Selects low-speed clock)
```

```
<Warm-up operation>
```

```
CGCKSEL<SYSCK> = "1"
(Selects low-speed clock for system clock)
Confirm that CGCKSEL<SYSCKFLG> is "1".
CGOSCCR<XEN> ="0"
(Stops high-speed oscillatir. In dual clock
mode, it's not required.)
```

```
PLL setup
&
Clock gear setting
```

```
When low-speed clock is used as fsys, the us-
age of PLL function and clock gear are pro-
hibited.
```

6.3.7 Prescaler Clock Control

Peripheral I/O has a prescaler for dividing a clock. As the clock φT0 to be input to each prescaler, the "fperiph" clock specified in the CGSYSCR<FPSEL0> can be divided according to the setting in the CGSYSCR<PRCK[2:0]>. After the controller is reset, fperiph/1 is selected as φT0.

Note: To use the clock gear, ensure that you make the time setting such that prescaler output φTn from each peripheral function is slower than fsys (φTn ≤ fsys). Do not switch the clock gear while the timer counter or other peripheral function is operating.

6.3.8 System Clock Pin Output Function

TMPM363F10FG enables to output the system clock from a pin. The SCOUT pin can output the low speed clock fs, the system clock fsys and fsys/2, and the prescaler input clock for peripheral I/O φT0.

Note 1: The phase difference (AC timing) between the system clock output by the SCOUT and the internal clock is not guaranteed.

Note 2: When fsys is output from SCOUT pin, SCOUT pin outputs the unexpected waveform just after changing clock gear. In the case of influencing to system by the unexpected waveform, the output of SCOUT pin shold be disabled when changing theclock gear.

The output clock is selected by setting the CGSYSCR<SCOSEL[1:0]>.

The setting to use as SCOUT pin, refer to "Input/Output port".

Table 6-3 shows the pin status in each mode when the SCOUT pin is set to the SCOUT output.

Table 6-3 SCOUT Output Status in Each Mode

SCOUT selection CGSYSCR	NORMAL	SLOW	IDLE2,1	SLEEP	STOP/BACKUP
<SCOSEL[1:0]> = "00"	Output the fs clock				
<SCOSEL[1:0]> = "01"	Output the fsys/2 clock				
<SCOSEL[1:0]> = "10"	Output the fsys clock				
<SCOSEL[1:0]> = "11"	Output the φT0 clock	Fixed "0" or "1"			

6.4 Modes and Mode Transitions

6.4.1 Mode Transitions

The NORMAL mode and the SLOW mode use the high-speed and low-speed clocks for the system clock respectively.

The IDLE2/1, SLEE, STOP and BACKUP modes can be used as the low power consumption mode that enables to reduce power consumption by halting processor core operation.

When the low-speed clock is not used, the SLOW, SLEEP and BACKUP SLEEP modes cannot be used.

TMPM363F10FG has a BACKUP mode. This mode can reduce power consumption of full width by shutdown main power supply of almost function except particular one.

Figure 6-2 shows mode transition diagram.

For a detail of sleep-on-exit, refer to "Cortex-M3 Technical Reference Manual".
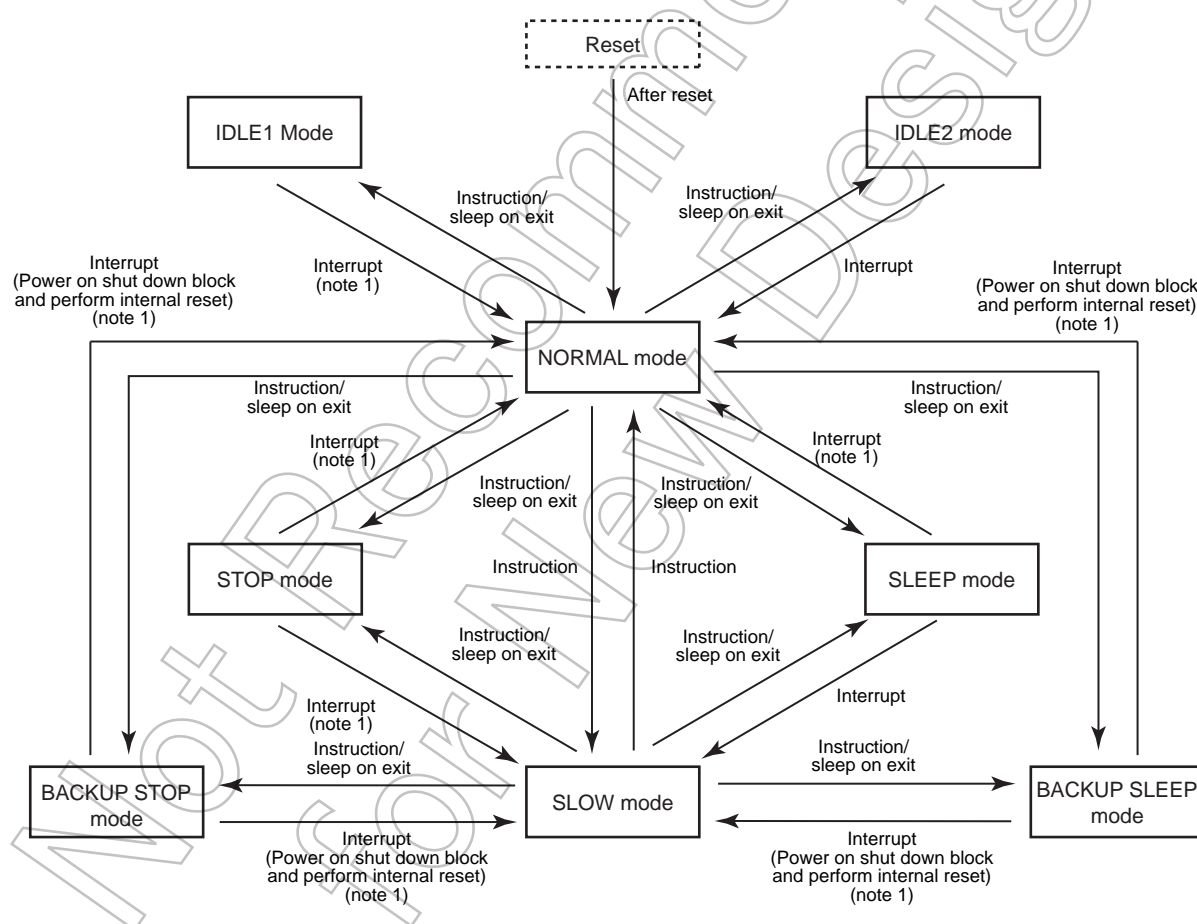
Figure 6-2 Mode Transition Diagram

Note 1: The warm-up is needed. The warm-up time must be set in NORMAL or SLOWmodes before changing to STOP, SLEEP and BACKUP modes. Regarding warm-up time, refer to "6.6.8 Warm-up".

Note 2: When the low-speed clock is not used, the SLOW, SLEEP and BACKUP SLEEP modes can not be used.

Note 3: Transition from SLOW mode to IDLE2 and IDLE1 mode is not available.

6.5 Operation Mode

NORMAL mode and SLOW mode are available. The features of each mode are described in the following section.

6.5.1 NORMAL mode

This mode is to operate the CPU core and the peripheral hardware by using the high-speed clock.

It is shifted to the NORMAL mode after reset. The low-speed clock can also be used.

6.5.2 SLOW mode

This mode is to operate the CPU core and the peripheral hardware by using the low-speed clock with high-speed clock stopped. The SLOW mode reduces power consumption compared to the NORMAL mode.

This mode allows some peripheral functions to operate.

Ther peripheral functions which can be operated are shown in Table 6-6.

Note: In the SLOW mode, be sure not to perform reset using the Application Interrupt and Reset Control Register <SYSRESETREQ> of the Cortex-M3 NVIC register.

6.6   Low Power Consumption Modes

The TMPM363F10FG has low power consumption modes: IDLE1, IDLE2, SLEEP, STOP and BACKUP. To shift to the low power consumption mode, specify the mode in the system control register CGSTBYCR<STBY[2:0]> and execute the WFI (Wait For Interrupt) instruction.In this case, execute reset (Except BACKUP mode) or generate the interrupt to release the mode. Releasing by the interrupt requires settings in advance. See the chapter "Exceptions" for details.

Note 1: The TMPM363F10FG does not offer any event for releasing the low power consumption mode. Transition to the low power consumption mode by executing the WFE (Wait For Event) instruction is prohibited.

Note 2: The TMPM363F10FG does not support the low power consumption mode configured with the SLEEPDEEP bit in the Cortex-M3 core. Setting the <SLEEPDEEP> bit of the system control register is prohibited.

Note 3: Do not release BACKUP mode by reset.

The features of each mode are described as follows.

6.6.1   IDLE Mode (IDLE2, IDLE1)

CPU is stopped in this mode. Each peripheral function has one bit in its control register for enabling or disabling operation in the IDLE mode. When the IDLE mode is enabled, peripheral functions for which operation in the IDLE mode is disabled stop operation and hold the state at that time.

The following peripheral functions can be enabled or disabled in the IDLE mode. For setting details, see the chapter on each peripheral function.

- · 16-bit timer / event counter (TMRB)
- · Serial channel (SIO/UART)
- · Serial bus interface (I2C/SIO)
- · AD converter (ADC)
- · Watchdog timer (WDT)

Note:Pay attention that the counter of watch dog timer function can not be cleared by CPU while in IDLE mode.

6.6.1.1   IDLE2 mode

In the IDLE2 mode, CPU stops.

Operation frequency range and the peripherals performance are equivalent for the normal mode besides power consumption is reduced compared to the NOMAL mode.

6.6.1.2   IDLE1 mode

The IDLE1 mode is decreased power supply ability than IDEL2 to realize low power consumption.

Using the IDLE1 mode, the conditions are refer to "Table 6-6 Operational Status in Each Mode" and Operation frequency must be set as follows; fsys=1MHz (fosc=8MHz, PLL multiplier circuit stop, clock gear 1/8).

6.6.2 SLEEP mode

In the SLEEP mode, the external low-speed oscillator, RTC, CEC, RMC and key-on wakeup can be operated.

By releasing the SLEEP mode, the device returns to the preceding mode of the SLEEP mode and starts operation.

6.6.3 STOP mode

All the internal circuits including the internal oscillator are brought to a stop in the STOP mode.

By releasing the STOP mode, the device returns to the preceding mode of the STOP mode and starts operation.

The STOP mode enables to select the pin status by setting the CGSTBYCR<DRVE>. Table 6-4 shows the pin status in the STOP mode.

Table 6-4 Pin States in the STOP mode

	Pin name	I/O	<DRVE> = 0	<DRVE> = 1
Not port	X1, XT1	Input only	×	×
	X2, XT2	Output only	"High" level output	"High" level output
	RESET, NMI, MODE	Input only	o	o
Port	PL3, PL7, PM3, PM7, PN3, PE7, PG3, PG7, (When used as interrupt pin PxFRn<PxmFn>=1 and input is enabled PxIE<PxmIE>=1) (note)	Input	o	o
		Output	×	Depends on PxCR[m]
	PJ4,PJ5, PJ6, PJ7 (When used as KWUP pin PxFRn<PxmFn>=1 and input is enabled PxIE<PxmIE>=1) (note)	Input	o	o
		Output	×	Depends on PxCR[m]
	PF0, PF1, PF2, PF3, PF4 (When used as trace data output pin xFRn<PxmFn>=1) (note)	Input	×	Depends on PxIE[m]
		Output	Depends on (PxCR[m])	
	PA7 to PA0, PB7 to PB0, PE6 to PE0, PP6 to PP2, PP0 (When used as external bus pin PxFRn<PxmFn>=1. Only data bus pin and input is enabled PxIE <PxmIE>=1) (note)	Input	o	o
		Output	Depends on (PxCR[m])	
	other port pin	Input	×	Depends in PxIE[m]
		Output	×	Depends in PxCR[m]

o : Input or output enabled.

× : Input or output disabled.

Note:x : port number / m : corresponding bit / n: function register number

6.6.4   BACKUP mode (BACKUP STOP, BACKUP SLEEP)

BACKUP mode realizes the lowest power consumption by cutting off the internal power regulator. About more details, refer to the BACKUP module.

6.6.5   Low power Consumption Mode Setting

The low power consumption mode is specified by the setting of the CGSTBYCR <STBY[2:0]>.

Table 6-5 shows the mode setting in the <STBY[2:0]>.

Table 6-5 Low power consumption mode setting

Mode	CGSTBYCR <STBY[2:0]>
STOP	001
SLEEP	010
IDLE2	011
BACKUP STOP	101
BACKUP SLEEP	110
IDLE1	111

Note: Except setting above is prohibited.

6.6.6   Operational Status in Each Mode

Table 6-6 shows the operational status in each mode.

Table 6-6 Operational Status in Each Mode

Block	NORMAL	SLOW	IDLE2	IDLE1 (note 1)	SLEEP	STOP	BACKUP SLEEP	BACKUP STOP
Processor core	o	o	–	–	–	–	×	×
DMAC	o	–	o	–	–	–	×	×
SMC	o	–	o	–	–	–	×	×
I/O port	o	o	o (note 6)	o (note 6)	o (note 6)	o (note 2)	Δ (note 3)	Δ (note 2 / 3)
ADC	o	# (note 5)	Δ	# (note 5)	– (note 5)	– (note 5)	×	×
SIO/UART	o	#	Δ	#	–	–	×	×
I2C/SIO	o	#	Δ	#	–	–	×	×
TMRB	o	o	Δ	#	–	–	×	×
WDT	o	#	Δ	#	–	–	×	×
SSP	o	#	–	–	–	–	×	×
CAN	o	#	–	–	–	–	×	×
USB-HOST	o	# (note 8)	– (note 8)	– (note 8)	– (note 8)	– (note 8)	× (note 8)	× (note 8)
KWUP	o	o	o (note 4)	o (note 4)	o	o (note 4)	Δ	Δ (note 4)
CEC	o	o	Δ	Δ	o	–	Δ	–
RMC	o	o	Δ	Δ	o	–	Δ	–
RTC	o	o	o	o	o	–	Δ	–
CG	o	o	o	o	o	o	o	o
PLL	o	*	Δ	#	#	#	# , ×	# , ×
High-speed oscillator (fc)	o	Δ	o	o	–	–	–	–
Low-speed oscillator (fs)	o	o	·	·	o	–	o	–
Main RAM	o	o	o	o	o	o	×	×
BACKUP RAM (note 2)	o	o	o	o	o	o	o	o

o : Operating

− : Clock stopped automatically after the setting mode. (note7)

Δ : Operating / stopped can be selected by software.

: Before enter the setting mode, must stop these module operations by software.

× : After transition the setting mode, power down these module automatically.

* : After transition the setting mode, must stop these module operations by software.

· : Before enter the setting mode, operating / stopped can be selected by software.

Note 1: In IDLE1 and SLOW mode, the particuler peripheral function shown in Table 6-6 must be stopped. In IDLE1 mode, TMPM363F10FG must be operated on maximam frequency of fsys = 1MHz (fosc=8MHz, PLL stopped, clock gear 1/8).

Note 2: It depends on CGSTBYCR<DRVE>.

Note 3: It depends on CGSTBYCR<PTKEEP>.

Note 4: When low-speed oscillator is stopped or stopped automatically, only static pull-up will be valid.

Note 5: Before transition the setting mode, clear ADMO1D<VREFON> to "0".

Note 6: Port state before transition the low power consumption mode is kept.

Note 7: The clock supplied to the module is stopped automatically after transition the setting mode. Therefore, transit the setting mode after comfirming the stop of the each module.

Note 8: Before transition the setting mode, must make SUSPEND.

6.6.7   Releasing the Low Power Consumption Mode

The low power consumption mode except BACKUP mode can be released by an interrupt request, Non-Maskable Interrupt (NMI) or reset.

The release source that can be used is determined by the low power consumption mode selected.

Details are shown in Table 6-7.

Table 6-7 Release Source in Each Mode

Low power consumption mode			IDLE2	IDLE1 (note 1)	SLEEP	STOP	BACKUP SLEEP (note 2)	BACKUP STOP (note 2)
Release source	Interrupt	INT0 to 4 (note 3)	o	o	o	o	o	o
		INT5 to INT7 (note 3)	o	o	o	o	×	×
		INTRTC	o	o	o	o	×	×
		INTTB0 to F	o	×	×	×	×	×
		INTCAP10 to 20,50 to 70, 90	o	×	×	×	×	×
		INTCAP 11 to 21,51 to 71,91	o	×	×	×	×	×
		INTRX0 to 4, INTTX0 to 4	o	×	×	×	×	×
		INTSBI0 to 3	o	×	×	×	×	×
		INTCECRX, INTCECTX	o	o	o	×	o (note 5)	×
		INTRMCRX0	o	o	o	×	o	×
		INTAD / INTADHP / INTADM0, 1	o	×	×	×	×	×
		INTKWUP	o	o	o	o	o	o
	SysTick interrupt		o	o	×	×	×	×
	NMI (INTWDT)		o	×	×	×	×	×
	NMI ($\overline{\text{NMI}}$ pin)		o	×	o	o	×	×
	RESET ($\overline{\text{RESE}}$ pin)		o	o	o	o	×	×

o :   Starts the interrupt handling after the mode is released. (The reset initializes the LSI)

× :   Unavailable

Note 1:   Refer to "6.6.8 Warm-up" about warm-up time.

Note 2:   After releasing BACKUP mode, initialize the circuit except BACKUP module.

Note 3:   To release the low power consumption mode by using the level mode interrupt, keep the level until the interrupt handling is started. Changing the level before then will prevent the interrupt handling from starting properly.

Note 4:   For shifting to the low power consumption mode, set the CPU to prohibit all the interrupts other than the release source. If not, releasing may be executed by an unspecified for wake up.

Note 5:   INTCECRX (CEC reception interrupt) is source trigger for wake up in the BACKUP SLEEP mode. But INTCECTX (CEC transmission interrupt) is not trigger for wake up in the BACKUP SLEEP mode.

Note 6:   To shift from NORMAL mode to IDLE1 mode, Warm-up time requires more than 100µs. If not, recovery time of MCU internal system is not done when the return from IDLE1 mode.

· Release by interrupt request

   To release the low power consumption mode by an interrupt, the CPU must be set in advance to detect the interrupt. In addition to the setting in the CPU, the clock generator must be set to detect the interrupt to be used to release the SLEEP and STOP modes.

· Release by Non-Maskable Interrupt (NMI)

   There are two kinds of NMI sources: WDT interrupt (INTWDT) and NMI pin. INTWDT can only be used in the IDLE2 mode. The NMI pin can be used to release all the lower power consumption modes except BACKUP and IDLE1 mode.

· Release by reset

   Any low power consumption mode except BACKUP mode can be released by reset from the $\overline{\text{RESET}}$ pin. After that, the mode switches to the NORMAL mode and all the registers are initialized as is the case with normal reset.

   Note that releasing from the STOP mode by reset does not induce the automatic warm-up. Keep the reset signal valid until the oscillator operation becomes stable.

· Release by SysTick interrupt

   SysTick interrupt can only be used in the IDLE mode.

Note: Do not release BACKUP mode by reset.

Refer to "Interrupts" in "Exceptions" for detail.

6.6.8   Warm-up

Mode transition may require the warm-up so that the internal oscillator provides stable oscillation.

In the mode transition from STOP to the NORMAL / SLOW or from IDLE1 / SLEEP to NORMAL, the warm-up counter is activated automatically. And then the system clock output is started after the elapse of configured warm-up time.

It is necessary to select a oscillator to be used for warm-up in the CGOSCCR<WUPSEL> and to set a warm-up time in the CGOSCCR<WUPT[11:0]><WUPTL[1:0]> before executing the instruction to enter the STOP / IDLE1 / SLEEP mode.

In the transition from NORMAL to SLOW / SLEEP, the warm-up is required so that the internal oscillator to stabilize if the low-speed clock is disabled. Enable the low-speed clock and then activate the warm-up by software.

In the transition from SLOW to NORMAL when the high-speed clock is disabled, enable the high-speed clock and then activate the warm-up.

Table 6-8 shows whether the warm-up setting of each mode transition is required or not.

Table 6-8 Warm-up setting in mode transition

Mode transition	Warm-up setting
NORMAL → IDLE2, 1	Not required
NORMAL → SLEEP	Not required (note 1)
NORMAL → SLOW	Not required (note 1)
NORMAL → STOP	Not required
NORMAL → BACKUP SLEEP	Not required (note 1)
NORMAL → BACKUP STOP	Not required
SLOW → NORMAL	Not required (note 2)
SLOW → SLEEP	Not required
SLOW → STOP	Not required
SLOW → BACKUP SLEEP	Not required
SLOW → BACKUP STOP	Not required
IDLE2 → NORMAL	Not required
IDLE1 → NORMAL	Auto-warm-up (note 3) High-speed oscillator : equal or more than 100µs
SLEEP → NORMAL	Auto-warm-up High-speed oscillator : Setting value of warm-up time
SLEEP → SLOW	Not required
STOP → NORMAL	Auto-warm-up High-speed oscillator : Setting value of warm-up time
STOP → SLOW	Auto-warm-up Low-speed oscillator :Setting value of warm-up time
BACKUP SLEEP → NORMAL	Auto-warm-up (note 3) High-speed oscillator : equal or more than 500µs
BACKUP STOP → NORMAL	Auto-warm-up (note 3) High-speed oscillator : equal or more than 500µs
BACKUP SLEEP → SLOW	Auto-warm-up (note 3) Low-speed oscillator : equal or more than 2.5ms
BACKUP STOP → SLOW	Auto-warm-up (note 3) Low-speed oscillator : equal or more than 2.5ms

Note 1: If the low-speed clock is disabled, enable the low-speed clock and then activate the warm-up by software.

Note 2: If the high-speed clock is disabled, enable the high-speed clock and then activate the warm-up by software.

Note 3: Do not set value of warm-up time less than the specified value.

Note 4: Returning to normal mode by reset does not induce the automatic warm-up. Keep the reset signal valid until the oscillator operation becomes stable.

6.6.9 Clock Operation in Mode Transition

The clock operation in mode transition are described Chapter 6.6.9.1 to 6.6.9.4.

6.6.9.1 Transition of operation modes : NORMAL → STOP → NORMAL

When returning to the NORMAL mode from the STOP mode, the warm-up is activated automatically. It is necessary to set the warm-up time before entering the STOP mode.

Returning to the NORMAL mode by reset does not induce the automatic warm-up. Keep the reset signal asserted until the oscillator operation becomes stable.
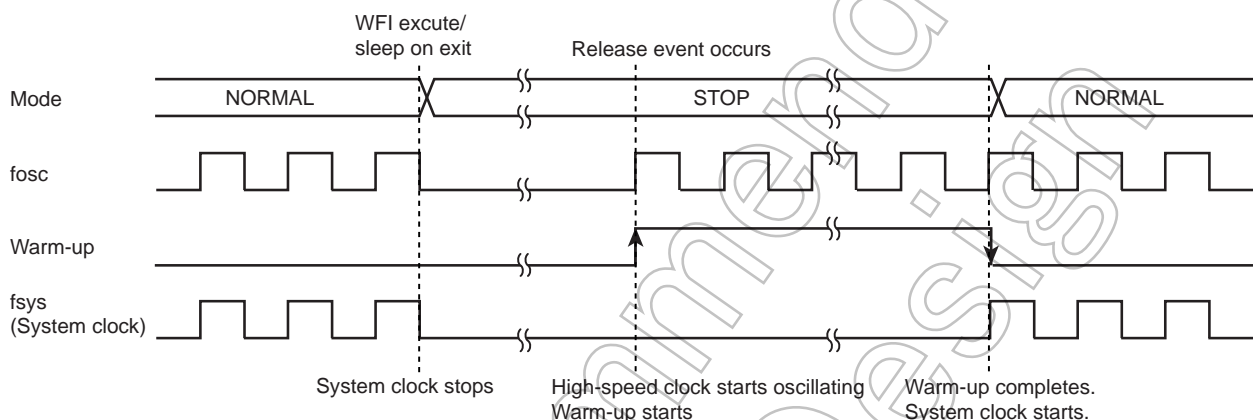
6.6.9.2 Transition of operation modes : NORMAL → SLEEP → NORMAL

When returning the NORMAL mode from the SLEEP mode, the warm-up is activated automatically. It is necessary to set the warm-up time before entering the SLEEP mode.

Returning to the NORMAL mode by reset does not induce the automatic warm-up. Keep the reset signal asserted until the operation becomes stable.
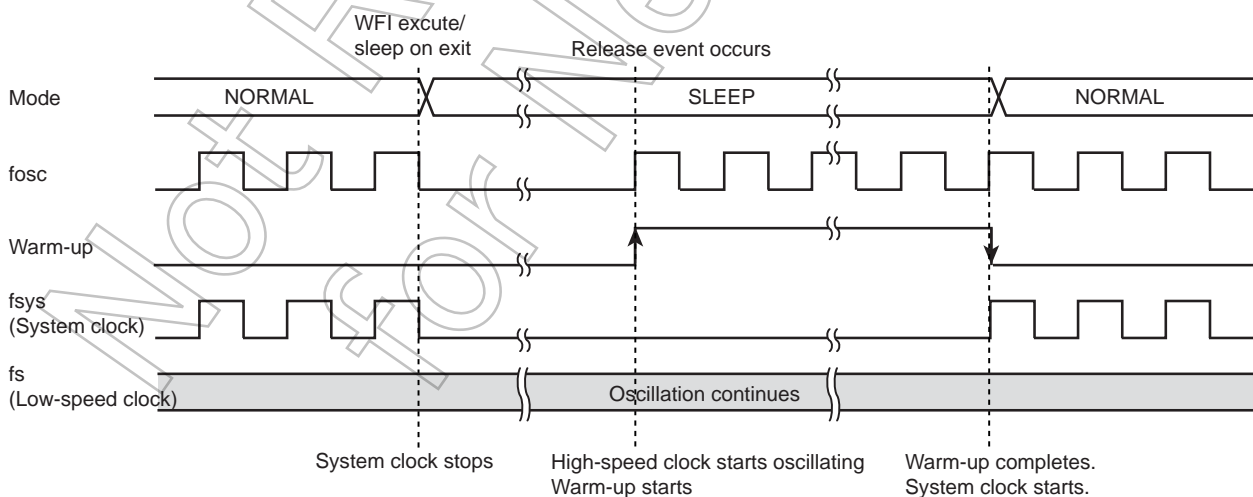
6.6.9.3   Transition of operation modes : SLOW → STOP → SLOW

The warm-up is activated automatically. It is necessary to set the warm-up time before entering the STOP mode.
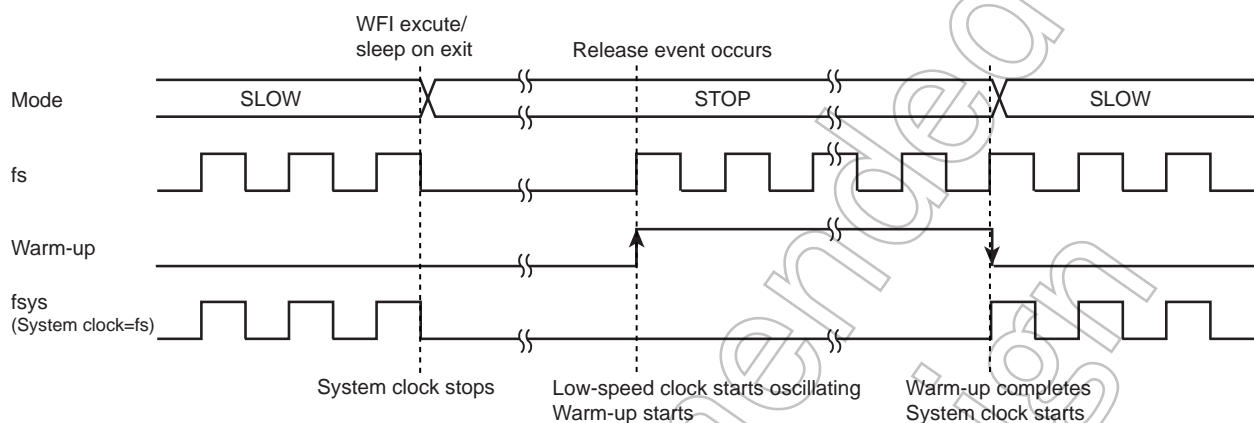
6.6.9.4   Transition of operation modes : SLOW → SLEEP → SLOW

The low-speed clock continues oscillation in the SLEEP mode. There is no need to make a warm-up setting.
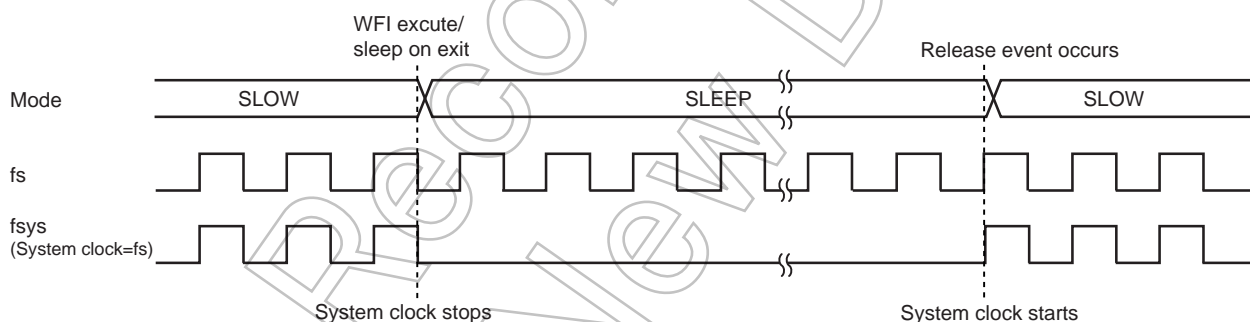
7. Exceptions

This chapter describes features, types and handling of exceptions.

Exceptions have close relation to the CPU core. Refer to "Cortex-M3 Technical Reference Manual" if needed.

7.1 Overview

Exceptions have close relation to the CPU core. Refer to "Cortex-M3 Technical Reference Manual" if needed.

There are two types of exceptions: those that are generated when some error condition occurs or when an instruction to generate an exception is executed; and those that are generated by hardware, such as an interrupt request signal from an external pin or peripheral function.

All exceptions are handled by the Nested Vectored Interrupt Controller (NVIC) in the CPU according to the respective priority levels. When an exception occurs, the CPU stores the current state to the stack and branches to the corresponding interrupt service routine (ISR). Upon completion of the ISR, the information stored to the stack is automatically restored.

7.1.1 Exception types

The following types of exceptions exist in the Cortex-M3.

For detailed descriptions on each exception, refer to "Cortex-M3 Technical Reference Manual".

- Reset
- Non-Maskable Interrupt (NMI)
- Hard Fault
- Memory Management
- Bus Fault
- Usage Fault
- SVCall (Supervisor Call)
- Debug Monitor
- PendSV
- SysTick
- External Interrupt

7.1.2   Handling Flowchart

The following shows how an exception/interrupt is handled. In the following descriptions, [　　　] indicates hardware handling. [　　　] indicates software handling.

Each step is described later in this chapter.

Processing	Description	See
Detection by CG/CPU	The CG/CPU detects the exception request.	Section 7.1.2.1
Handling by CPU	The CPU handles the exception request.	Section 7.1.2.2
Branch to ISR	The CPU branches to the corresponding interrupt service routine (ISR).	
Execution of ISR	Necessary processing is executed.	Section 7.1.2.4
Return from exception	The CPU branches to another ISR or returns to the previous program.	Section 7.1.2.4

7.1.2.1   Exception Request and Detection

(1)   Exception occurrence

Exception sources include instruction execution by the CPU, memory accesses, and interrupt requests from external interrupt pins or peripheral functions.

An exception occurs when the CPU executes an instruction that causes an exception or when an error condition occurs during instruction execution.

An exception also occurs by an instruction fetch from the Execute Never (XN) region or an access violation to the Fault region.

An interrupt request is generated from an external interrupt pin or peripheral function.For interrupts that are used for releasing a standby mode, relevant settings must be made in the clock generator. For details, refer to "7.5 Interrupts".

(2)   Exception detection

If multiple exceptions occur simultaneously, the CPU takes the exception with the highest priority.

Table 7-1 shows the priority of exceptions. "Configurable" means that you can assign a priority level to that exception. Memory Management, Bus Fault and Usage Fault exceptions can be enabled or disabled. If a disabled exception occurs, it is handled as Hard Fault.

Table 7-1 Exception Types and Priority

No.	Exception type	Priority	Description
1	Reset	−3 (highest)	Reset pin, WDT or SYSRETREQ
2	Non-Maskable Interrupt	−2	$\overline{NMI}$ pin or WDT
3	Hard Fault	−1	Fault that cannot activate because a higher-priority fault is being handled or it is disabled
4	Memory Management	Configurable	Exception from the Memory Protection Unit (MPU) (Note 1) Instruction fetch from the Execute Never (XN) region
5	Bus Fault	Configurable	Access violation to the Hard Fault region of the memory map
6	Usage Fault	Configurable	Undefined instruction execution or other faults related to instruction execution
7~10	Reserved	−	
11	SVCall	Configurable	System service call with SVC instruction
12	Debug Monitor	Configurable	Debug monitor when the CPU is not faulting
13	Reserved	−	
14	PendSV	Configurable	Pendable system service request
15	SysTick	Configurable	Notification from system timer
16~	External interrupt	Configurable	External interrupt pin or peripheral function (Note2)

Note 1: **This product does not contain the MPU.**

Note 2: **External interrupts have different sources and numbers in each product. For details, see"7.5.1.5 List of Interrupt Sources".**

(3)   Priority setting

· Priority level

The external interrupt priority is set to the interrupt priority register and other exceptions are set to <PRI_n> bit in the system handler priority register.

The configuration <PRI_n> can be changed, and the number of bits required for setting the priority varies from 3 bits to 8 bits depending on products. Thus, the range of priority values you can specify is different depending on products.

In the case of 8-bit configuration, the priority can be configured in the range from 0 to 255. The highest priority is "0". If multiple elements with the same priority exist, the smaller the number, the higher the priority becomes.

Note: **<PRI_n> bit is defined as a 3-bit configuration with this product.**

· Priority grouping

The priority group can be split into groups. By setting the <PRIGROUP> of the application interrupt and reset control register, <PRI_n> can be divided into the pre-emption priority and the sub priority.

A priority is compared with the pre-emption priority. If the priority is the same as the pre-emption priority, then it is compared with the sub priority. If the sub priority is the same as the priority, the smaller the exception number, the higher the priority.

The Table 7-2 shows the priority group setting. The pre-emption priority and the sub priority in the table are the number in the case that <PRI_n> is defined as an 8-bit configuration.

Table 7-2 Priority grouping setting

<PRIGROUP[2:0]> setting	<PRI_n[7:0]>		Number of pre-emption priorities	Number of subpriorities
	Pre-emption field	Subpriority field		
000	[7:1]	[0]	128	2
001	[7:2]	[1:0]	64	4
010	[7:3]	[2:0]	32	8
011	[7:4]	[3:0]	16	16
100	[7:5]	[4:0]	8	32
101	[7:6]	[5:0]	4	64
110	[7]	[6:0]	2	128
111	None	[7:0]	1	256

Note:If the configuration of <PRI_n> is less than 8 bits, the lower bit is "0". For the example, in the case of 3-bit configuration, the priority is set as <PRI_n[7:5]> and <PRI_n[4:0]> is "00000".

7.1.2.2   Exception Handling and Branch to the Interrupt Service Routine (Pre-emption)

When an exception occurs, the CPU suspends the currently executing process and branches to the interrupt service routine. This is called "pre-emption".

(1)   Stacking

When the CPU detects an exception, it pushes the contents of the following eight registers to the stack in the following order :

- · Program Counter (PC)
- · Program Status Register (xPSR)
- · r0 to r3
- · r12
- · Link Register (LR)

The SP is decremented by eight words by the completion of the stack push.The following shows the state of the stack after the register contents have been pushed.

Old SP →	<previous>
	xPSR
	PC
	LR
	r12
	r3
	r2
	r1
SP →	r0
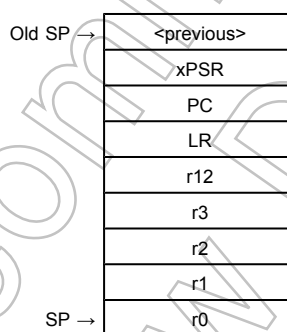
(2)   fetching an ISR

The CPU enables instruction to fetch the interrupt processing with data store to the register.

Prepare a vector table containing the top addresses of ISRs for each exception.After reset, the vector table is located at address 0x0000_0000 in the Code area.By setting the Vector Table Offset Register, you can place the vector table at any address in the Code or SRAM space.

The vector table should also contain the initial value of the main stack.

(3)   Late-arriving

If the CPU detects a higher priority exception before executing the ISR for a previous exception, the CPU handles the higher priority exception first. This is called "late-arriving".

A late-arriving exception causes the CPU to fetch a new vector address for branching to the corresponding ISR, but the CPU does not newly push the register contents to the stack.

(4)   Vector table

The vector table is configured as shown below.

You must always set the first four words (stack top address, reset ISR address, NMI ISR address, and Hard Fault ISR address). Set ISR addresses for other exceptions if necessary.

Offset	Exception	Contents	Setting
0x00	Reset	Initial value of the main stack	Required
0x04	Reset	ISR address	Required
0x08	Non-Maskable Interrupt	ISR address	Required
0x0C	Hard Fault	ISR address	Required
0x10	Memory Management	ISR address	Optional
0x14	Bus Fault	ISR address	Optional
0x18	Usage Fault	ISR address	Optional
0x1C to 0x28	Reserved		
0x2C	SVCall	ISR address	Optional
0x30	Debug Monitor	ISR address	Optional
0x34	Reserved		
0x38	PendSV	ISR address	Optional
0x3C	SysTick	ISR address	Optional
0x40	External Interrupt	ISR address	Optional

7.1.2.3 Executing an ISR

An ISR performs necessary processing for the corresponding exception. ISRs must be prepared by the user.

An ISR may need to include code for clearing the interrupt request so that the same interrupt will not occur again upon return to normal program execution.

For details about interrupt handling, see "7.5 Interrupts".

If a higher priority exception occurs during ISR execution for the current exception, the CPU abandons the currently executing ISR and services the newly detected exception.

7.1.2.4　Exception exit

(1)　Execution after returning from an ISR

When returning from an ISR, the CPU takes one of the following actions :

- Tail-chaining

    If a pending exception exists and there are no stacked exceptions or the pending exception has higher priority than all stacked exceptions, the CPU returns to the ISR of the pending exception.

    In this case, the CPU skips the pop of eight registers and push of eight registers when exiting one ISR and entering another. This is called "tail-chaining".

- Returning to the last stacked ISR

    If there are no pending exceptions or if the highest priority stacked exception is of higher priority than the highest priority pending exception, the CPU returns to the last stacked ISR.

- Returning to the previous program

    If there are no pending or stacked exceptions, the CPU returns to the previous program.

(2)　Exception exit sequence

When returning from an ISR, the CPU performs the following operations :

- Pop eight registers

    Pops the eight registers (PC, xPSR, r0 to r3, r12 and LR) from the stack and adjust the SP.

- Load current active interrupt number

    Loads the current active interrupt number from the stacked xPSR. The CPU uses this to track which interrupt to return to.

- Select SP

    If returning to an exception (Handler Mode), SP is SP_main. If returning to Thread Mode, SP can be SP_main or SP_process.

7.2 Reset Exceptions

Reset exceptions are generated from the following three sources.

Use the Reset Flag (CGRSTFLG) Register of the Clock Generator to identify the source of a reset.

- External reset pin

    A reset exception occurs when an external reset pin changes from "Low" to "High".

- Reset exception by WDT

    The watchdog timer (WDT) has a reset generating feature. For details, see the chapter on the WDT.

- Reset exception by SYSRESETREQ

    A reset can be generated by setting the SYSRESETREQ bit in the NVIC's Application Interrupt and Reset Control Register.

Note: **Do not reset with <SYSRESETREQ> in SLOW mode.**

7.3 Non-Maskable Interrupts (NMI)

Non-maskable interrupts are generated from the following two sources.

Use the NMI Flag (CGNMIFLG) Register of the clock generator to identify the source of a non-maskable interrupt.

- External $\overline{NMI}$ pin

    A non-maskable interrupt is generated when an external $\overline{NMI}$ pin changes from "High" to "Low".

- Non-maskable interrupt by WDT

    The watchdog timer (WDT) has a non-maskable interrupt generating feature. For details, see the chapter on the WDT.

7.4 SysTick

SysTick provides interrupt features using the CPU's system timer.

When you set a value in the SysTick Reload Value Register and enable the SysTick features in the SysTick Control and Status Register, the counter loads with the value set in the Reload Value Register and begins counting down.When the counter reaches "0", a SysTick exception occurs.You may be pending exceptions and use a flag to know when the timer reaches "0".

The SysTick Calibration Value Register holds a reload value for counting 10 ms with the system timer. The count clock frequency varies with each product, and so the value set in the SysTick Calibration Value Register also varies with each product.

Note:In this product, fosc by 32 is used as external reference clock.

7.5 Interrupts

This chapter describes routes, sources and required settings of interrupts.

The CPU is notified of interrupt requests by the interrupt signal from each interrupt source.

It sets priority on interrupts and handles an interrupt request with the highest priority.

Interrupt requests for clearing a standby mode are notified to the CPU via the clock generator. Therefore, appropriate settings must be made in the clock generator.

7.5.1 Interrupt Sources

7.5.1.1 Interrupt route

Figure 7-1 shows an interrupt request route.

The interrupts issued by the peripheral function that is not used to release standby are directly input to the CPU (route1).

The peripheral function interrupts used to release standby (route 2) and interrupts from the external interrupt pin (route 3) are input to the clock generator and are input to the CPU through the logic for releasing standby (route 4 and 5).

If interrupts from the external interrupt pins are not used to release standby, they are directly input to the CPU, not through the logic for standby release (route 6).
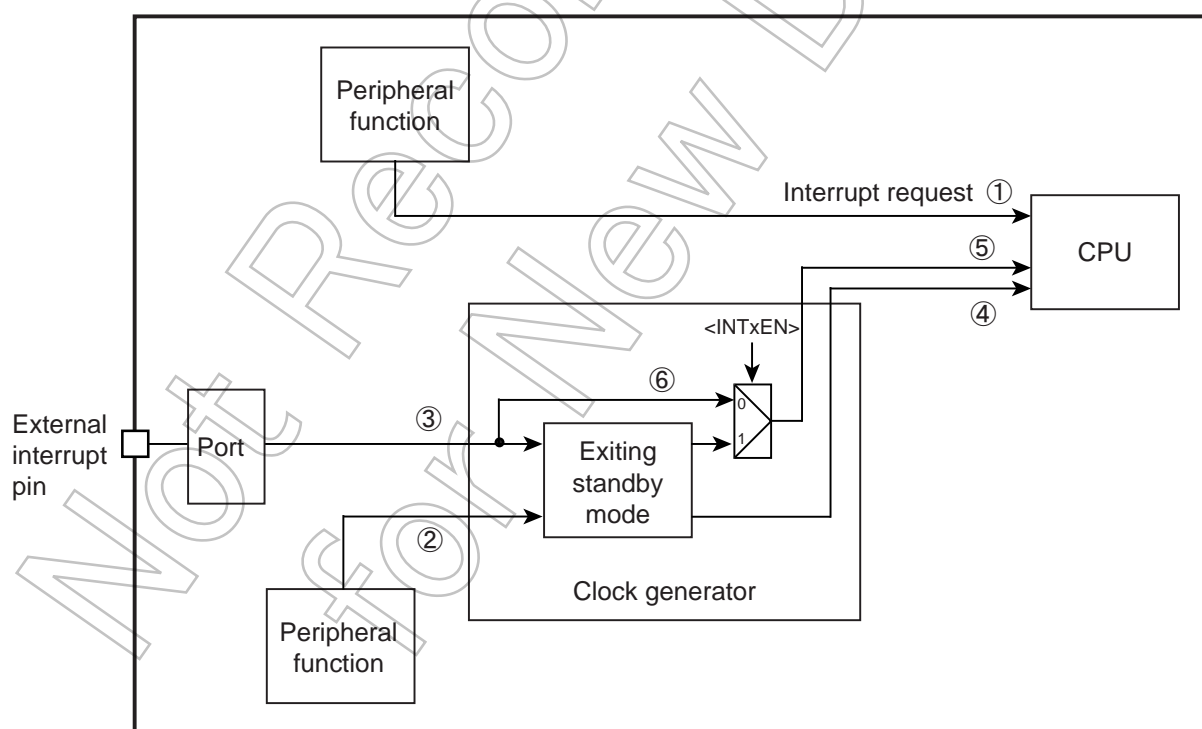
Figure 7-1 Interrupt Route

7.5.1.2 Generation

An interrupt request is generated from an external pin or peripheral function assigned as an interrupt source or by setting the NVIC's Interrupt Set-Pending Register.

- From external pin

   Set the port control register so that the external pin can perform as an interrupt function pin.

- From peripheral function

   Set the peripheral function to make it possible to output interrupt requests.

   See the chapter of each peripheral function for details.

- By setting Interrupt Set-Pending Register (forced pending)

   An interrupt request can be generated by setting the relevant bit of the Interrupt Set-Pending Register.

7.5.1.3 Transmission

An interrupt signal from an external pin or peripheral function is directly sent to the CPU unless it is used to exit a standby mode.

Interrupt requests from interrupt sources that can be used for clearing a standby mode are transmitted to the CPU via the clock generator. For these interrupt sources, appropriate settings must be made in the clock generator in advance. External interrupt sources not used for exiting a standby mode can be used without setting the clock generator.

7.5.1.4 Precautions when using external interrupt pins

If you use external interrupts, be aware the followings not to generate unexpected interrupts.

If input disabled (PxIE<PxmIE>="0"), inputs from external interrupt pins are "High". Also, if external interrupts are not used as a trigger to release standby (route 6 of Figure 7-1), input signals from the external interrupt pins are directly sent to the CPU. Since the CPU recognizes "High" input as an interrupt, interrupts occur if corresponding interrupts are enabled by the CPU as inputs are being disabled.

To use the external interrupt without setting it as a standby trigger, set the interrupt pin input as "Low" and enable it. Then, enable interrupts on the CPU.

7.5.1.5 List of Interrupt Sources

Table 7-3 shows the list of interrupt sources.

Table 7-3 List of Interrupt Sources

No.	Interrupt Source		active level (Releasing standby and interrupt)	CG interrupt mode control register
0	INT0	Interrupt pin 0	Selectable	CGIMCGA
1	INT1	Interrupt pin 1		
2	INT2	Interrupt pin 2		
3	INT3	Interrupt pin 3		
4	INT4	Interrupt pin 4		CGIMCGB
5	INT5	Interrupt pin 5		
6	INT6	Interrupt pin 6		
7	INT7	Interrupt pin 7		
8	Reserved	-	-	-
9	Reserved	-		
10	Reserved	-		
11	Reserved	-		
12	Reserved	-		
13	Reserved	-		
14	Reserved	-		
15	Reserved	-		
16	INTRX0	Serial channel reception 0 interrupt		
17	INTTX0	Serial channel transmission 0 interrupt		
18	INTRX1	Serial channel reception 1 interrupt		
19	INTTX1	Serial channel transmission 1 interrupt		
20	INTRX2	Serial channel reception 2 interrupt		
21	INTTX2	Serial channel transmission 2 interrupt		
22	INTRX3	Serial channel reception 3 interrupt		
23	INTTX3	Serial channel transmission 3 interrupt		
24	INTRX4	Serial channel reception 4 interrupt		
25	INTTX4	Serial channel transmission 4 interrupt		
26	INTSBI0	Serial bus interface 0 interrupt		
27	INTSBI1	Serial bus interface 1 interrupt		
28	INTCECRX	CEC reception interrupt	Rising edge	CGIMCGE
29	INTCECTX	CEC transmission interrupt		
30	INTRMCRX	Remote control signal reception interrupt		
31	Reserved	-	-	-
32	INTRTC	RTC interrupt	Falling edge	CGIMCGF
33	INTKWUP	Key-on wake-up interrupt	High level	
34	INTSBI2	Serial bus interface 2 interrupt		
35	INTSBI3	Serial bus interface 3 interrupt		
36	Reserved	-		
37	INTADHP	Highest priority AD conversion complete interrupt		
38	INTADM0	AD conversion monitoring function 0 interrupt		
39	INTADM1	AD conversion monitoring function 1 interrupt		
40	INTTB0	16-bit timer /event counter 0 match detection interrupt		
41	INTTB1	16-bit timer /event counter 1 match detection interrupt		

Table 7-3 List of Interrupt Sources

No.	Interrupt Source		active level (Releasing standby and interrupt)	CG interrupt mode control register
42	INTTB2	16-bit timer /event counter 2 match detection interrupt		
43	INTTB3	16-bit timer /event counter 3 match detection interrupt		
44	INTTB4	16-bit timer /event counter 4 match detection interrupt		
45	INTTB5	16-bit timer /event counter 5 match detection interrupt		
46	INTTB6	16-bit timer /event counter 6 match detection interrupt		
47	INTTB7	16-bit timer /event counter 7 match detection interrupt		
48	INTTB8	16-bit timer /event counter 8 match detection interrupt		
49	INTTB9	16-bit timer /event counter 9 match detection interrupt		
50	INTTBA	16-bit timer /event counter A match detection interrupt		
51	INTTBB	16-bit timer /event counter B match detection interrupt		
52	INTTBC	16-bit timer /event counter C match detection interrupt		
53	INTTBD	16-bit timer /event counter D match detection interrupt		
54	INTTBE	16-bit timer /event counter E match detection interrupt		
55	INTTBF	16-bit timer /event counter F match detection interrupt		
56	INTUSB	USB interrupt		
57	INTCANGB	CAN status interrupt		
58	INTAD	AD conversion completion interrupt		
59	INTSSPO	SSP interrupt		
60	Reserved	-		
61	Reserved	-		
62	Reserved	-		
63	Reserved	-		
64	Reserved	-		
65	Reserved	-		
66	Reserved	-		
67	Reserved	-		
68	Reserved	-		
69	Reserved	-		
70	Reserved	-		
71	Reserved	-		
72	Reserved	-		
73	Reserved	-		
74	INTCAP10	16-bit timer /event counter 1 input capture interrupt 0		
75	INTCAP11	16-bit timer /event counter 1 input capture interrupt 1		
76	INTCAP20	16-bit timer /event counter 2 input capture interrupt 0		

Table 7-3 List of Interrupt Sources

No.	Interrupt Source		active level (Releasing standby and interrupt)	CG interrupt mode control register
77	INTCAP21	16-bit timer /event counter 2 input capture interrupt 1		
78	INTCANRX	CAN reception interrupt		
79	INTCANTX	CAN transmission interrupt		
80	INTCAP50	16-bit timer /event counter 5 input capture interrupt 0		
81	INTCAP51	16-bit timer /event counter 5 input capture interrupt 1		
82	INTCAP60	16-bit timer /event counter 6 input capture interrupt 0		
83	INTCAP61	16-bit timer /event counter 6 input capture interrupt 1		
84	INTCAP70	16-bit timer /event counter 7 input capture interrupt 0		
85	INTCAP71	16-bit timer /event counter 7 input capture interrupt 1		
86	INTCAP90	16-bit timer /event counter 9 input capture interrupt 0		
87	INTCAP91	16-bit timer /event counter 9 input capture interrupt 1		
88	Reserved	-		
89	Reserved	-		
90	Reserved	-		
91	Reserved	-		
92	Reserved	-		
93	Reserved	-		
94	Reserved	-		
95	Reserved	-		
96	Reserved	-		
97	Reserved	-		
98	INTDMACERR	DMA error status interrupt		
99	INTDMACTC0	DMA terminal count status interrupt		

7.5.1.6 Active level

The active level indicates which change in signal of an interrupt source triggers an interrupt. The CPU recognizes interrupt signals in "High" level as interrupt. Interrupt signals directly sent from peripheral functions to the CPU are configured to output "High" to indicate an interrupt request.

Active level is set to the clock generator for interrupts which can be a trigger to release standby. Interrupt requests from peripheral functions are set as rising-edge or falling-edge triggered. Interrupt requests from interrupt pins can be set as level-sensitive ("High" or "Low") or edge-triggered (rising or falling).

If an interrupt source is used for clearing a standby mode, setting the relevant clock generator register is also required. Enable the CGIMCGx<INTxEN> bit and specify the active level in the CGIMCGx<EMCGx> bits. You must set the active level for interrupt requests from each peripheral function as shown in Table 7-3

An interrupt request detected by the clock generator is notified to the CPU with a signal in "High" level.

Note: **For the CEC reception / transmission, remote control signal reception and real time clock interrupts, set the <INTxEN> bit to "1" and specify the active level, even when they are not used for clearing a standby mode.**

7.5.2   Interrupt Handling

7.5.2.1   Flowchart

The following shows how an interrupt is handled.

The following shows how an exception/interrupt is handled. In the following descriptions, indicates hardware handling. indicates software handling.

Processing	Details	See
Setting for detection **setting for sending interrupt signal**	Set the relevant NVIC registers for detecting interrupts. Set the clock generator as well if each interrupt source is used to clear a standby mode. o Common setting NVIC registers o setting to clear standby mode Clock generator Execute an appropriate setting to send the interrupt signal depending on the interrupt type. o Setting for interrupt from external pin Port o Setting for interrupt from peripheral function Peripheral function (See the chapter of each peripheral function for details.)	"7.5.2.2 Preparation"
Interrupt generation	An interrupt request is generated.	
Not clearing standby mode / Clearing standby mode **CG detects interrupt (clearing standby mode)**	Interrupt lines used for clearing a standby mode are connected to the CPU via the clock generator.	"7.5.2.3 Detection by Clock Generator"
CPU detects interrupt.	The CPU detects the interrupt. If multiple interrupt requests occur simultaneously, the interrupt request with the highest priority is detected according to the priority order.	"7.5.2.4 Detection by CPU"
CPU handles interrupt.	The CPU handles the interrupt. The CPU pushes register contents to the stack before entering the ISR.	"7.5.2.5 CPU processing"

Processing	Details	See
ISR execution	Program for the ISR. Clear the interrupt source if needed.	"7.5.2.6 Interrupt Service Routine (ISR)"
Return to preceding program	Configure to return to the preceding program of the ISR.	

7.5.2.2   Preparation

When preparing for an interrupt, you need to pay attention to the order of configuration to avoid any un-expected interrupt on the way.

Initiating an interrupt or changing its configuration must be implemented in the following order basical-ly. Disable the interrupt by the CPU. Configure from the farthest route from the CPU. Then enable the in-terrupt by the CPU.

To configure the clock generator, you must follow the order indicated here not to cause any unexpec-ted interrupt. First, configure the precondition. Secondly, clear the data related to the interrupt in the clock generator and then enable the interrupt.

The following sections are listed in the order of interrupt handling and describe how to configure them.

1. Disabling interrupt by CPU

2. CPU registers setting

3. Preconfiguration (1) (Interrupt from external pin)

4. Preconfiguration (2) (Interrupt from peripheral function)

5. Preconfiguration (3) (Interrupt Set-Pending Register)

6. Configuring the clock generator

7. Enabling interrupt by CPU

(1)   Disabling interrupt by CPU

To make the CPU for not accepting any interrupt, write "1" to the corresponding bit of the PRI-MASK Register. All interrupts and exceptions other than non-maskable interrupts and hard faults can be masked.

Use "MSR" instruction to set this register.

Interrupt mask register		
PRIMASK	←	"1"(Interrupt disabled)

Note 1: PRIMASK register cannot be modified by the user access level.
Note 2: If a fault causes when "1" is set to the PRIMASK register, it is treated as a hard fault.

(2)   CPU registers setting

You can assign a priority level by writing to <PRI_n> field in an Interrupt Priority Register of the NVIC register.

Each interrupt source is provided with eight bits for assigning a priority level from 0 to 255, but the number of bits actually used varies with each product.Priority level 0 is the highest priority lev-el.If multiple sources have the same priority, the smallest-numbered interrupt source has the highest priority.

You can assign grouping priority by using the <PRIGROUP> in the Application Interrupt and Reset Control Register.

NVIC register		
<PRI_n>	←	"prioryty"
<PRIGROUP>	←	"group priority" (This is configurable if required.)

Note: "n" indicates the corresponding exceptions/interrupts.

This product uses three bits for assigning a priority level.

(3) Preconfiguration (1) (Interrupt from external pin)

Set "1" to the port function register of the corresponding pin. Setting PxFRn[m] allows the pin to be used as the function pin. Setting PxIE[m] allows the pin to be used as the input port.

Port register		
PxFRn<PxmFn>	←	"1"
PxIE<PxmIE>	←	"1"

Note: x: port number / m: corresponding bit / n: function register number In modes other than STOP mode, setting PxIE to enable input enables the corresponding interrupt input regardless of the PxFR setting. Be careful not to enable interrupts that are not used. Also, be aware of the description of "7.5.1.4 Precautions when using external interrupt pins".

(4) Preconfiguration (2) (Interrupt from peripheral function)

The setting varies depending on the peripheral function to be used. See the chapter of each peripheral function for details.

(5) Preconfiguration (3) (Interrupt Set-Pending Register)

To generate an interrupt by using the Interrupt Set-Pending Register, set "1" to the corresponding bit of this register.

NVIC register		
Interrupt Set-Pending [m]	←	"1"

Note: m: corresponding bit

(6) Configuring the clock generator

For an interrupt source to be used for exiting a standby mode, you need to set the active level and enable interrupts in the CGIMCG register of the clock generator. The CGIMCG register is capable of configuring each source.

Before enabling an interrupt, clear the corresponding interrupt request already held. This can avoid unexpected interrupt.To clear corresponding interrupt request, write a value corresponding to the interrupt to be used to the CGICRCG register.See "7.6.3.5 CGICRCG (CG Interrupt Request Clear Register)" for each value.

Interrupt requests from external pins can be used without setting the clock generator if they are not used for exiting a standby mode. However, an "High" pulse or "High"-level signal must be input so that the CPU can detect it as an interrupt request. Also, be aware of the description of "7.5.1.4 Precautions when using external interrupt pins".

Clock generator register		
CGIMCGn<EMCGm>	←	active level
CGICRCG<ICRCG>	←	Value corresponding to the interrupt to be used
CGIMCGn<INTmEN>	←	"1" (interrupt enabled)

Note: n: register number / m: number assigned to interrupt source

(7)   Enabling interrupt by CPU

Enable the interrupt by the CPU as shown below.

Clear the suspended interrupt in the Interrupt Clear-Pending Register. Enable the intended interrupt with the Interrupt Set-Enable Register. Each bit of the register is assigned to a single interrupt source.

Writing "1" to the corresponding bit of the Interrupt Clear-Pending Register clears the suspended interrupt. Writing "1" to the corresponding bit of the Interrupt Set-Enable Register enables the intended interrupt.

To generate interrupts in the Interrupt Set-Pending Register setting, factors to trigger interrupts are lost if pending interrupts are cleared. Thus, this operation is not necessary.

At the end, PRIMASK register is zero cleared.

NVIC register		
Interrupt Clear-Pending [m]	←	"1"
Interrupt Set-Pending [m]	←	"1"
Interrupt mask register		
PRIMASK	←	"0"

Note 1: **m : corresponding bit**
Note 2: **PRIMASK register cannot be modified by the user access level.**

7.5.2.3   Detection by Clock Generator

If an interrupt source is used for exiting a standby mode, an interrupt request is detected according to the active level specified in the clock generator, and is notified to the CPU.

An edge-triggered interrupt request, once detected, is held in the clock generator. A level-sensitive interrupt request must be held at the active level until it is detected, otherwise the interrupt request will cease to exist when the signal level changes from active to inactive.

When the clock generator detects an interrupt request, it keeps sending the interrupt signal in "High" level to the CPU until the interrupt request is cleared in the CG Interrupt Request Clear (CGICRCG) Register. If a standby mode is exited without clearing the interrupt request, the same interrupt will be detected again when normal operation is resumed. Be sure to clear each interrupt request in the ISR.

7.5.2.4 Detection by CPU

The CPU detects an interrupt request with the highest priority.

7.5.2.5 CPU processing

On detecting an interrupt, the CPU pushes the contents of PC, PSR, r0-r3, r12 and LR to the stack then enter the ISR.

7.5.2.6 Interrupt Service Routine (ISR)

An ISR requires specific programming according to the application to be used. This section describes what is recommended at the service routine programming and how the source is cleared.

(1) Pushing during ISR

An ISR normally pushes register contents to the stack and handles an interrupt as required. The Cortex-M3 core automatically pushes the contents of PC, PSR, r0-r3, r12 and LR to the stack. No extra programming is required for them.

Push the contents of other registers if needed.

Interrupt requests with higher priority and exceptions such as NMI are accepted even when an ISR is being executed. We recommend you to push the contents of general-purpose registers that might be rewritten.

(2) Clearing an interrupt source

If an interrupt source is used for clearing a standby mode, each interrupt request must be cleared with the CG Interrupt Request Clear (CGICRCG) Register.

If an interrupt source is set as level-sensitive, an interrupt request continues to exist until it is cleared at its source. Therefore, the interrupt source must be cleared. Clearing the interrupt source automatically clears the interrupt request signal from the clock generator.

If an interrupt is set as edge-sensitive, clear an interrupt request by setting the corresponding value in the CGICRCG register. When an active edge occurs again, a new interrupt request will be detected.

7.6 Exception / Interrupt-Related Registers

The CPU's NVIC registers and clock generator registers described in this chapter are shown below with their respective addresses.

7.6.1 Register List

NVIC registers   Base Address = 0xE000_E000

Register name	Address
SysTick Control and Status Register	0x0010
SysTick Reload Value Register	0x0014
SysTick Current Value Register	0x0018
SysTick Calibration Value Register	0x001C
Interrupt Set-Enable Register 1	0x0100
Interrupt Set-Enable Register 2	0x0104
Interrupt Set-Enable Register 3	0x0108
Interrupt Set-Enable Register 4	0x010C
Interrupt Clear-Enable Register 1	0x0180
Interrupt Clear-Enable Register 2	0x0184
Interrupt Clear-Enable Register 3	0x0188
Interrupt Clear-Enable Register 4	0x018C
Interrupt Set-Pending Register 1	0x0200
Interrupt Set-Pending Register 2	0x0204
Interrupt Set-Pending Register 3	0x0208
Interrupt Set-Pending Register 4	0x020C
Interrupt Clear-Pending Register 1	0x0280
Interrupt Clear-Pending Register 2	0x0284
Interrupt Clear-Pending Register 3	0x0288
Interrupt Clear-Pending Register 4	0x028C
Interrupt Priority Register	0x0400 to 0x0460
Vector Table Offset Register	0x0D08
Application Interrupt and Reset Control Register	0x0D0C
System Handler Priority Register	0x0D18, 0x0D1C, 0x0D20
System Handler Control and State Register	0x0D24

Clock generator register   Base Address = 0x400F_4000

Register name		Address
CG Interrupt Request Clear Register	CGICRCG	0x0014
NMI Flag Register	CGNMIFLG	0x0018
Reset Flag Register	CGRSTFLG	0x001C
CG Interrupt Mode Control Register A	CGIMCGA	0x0020
CG Interrupt Mode Control Register B	CGIMCGB	0x0024
Reserved	-	0x0028
CG Interrupt Mode Control Register D	CGIMCGD	0x002C
CG Interrupt Mode Control Register E	CGIMCGE	0x0030
CG Interrupt Mode Control Register F	CGIMCGF	0x0034
Reserved	-	0x0038
Reserved	-	0x003C

Note: Access to the "Reserved" areas is prohibited.

7.6.2 NVIC Registers

7.6.2.1 SysTick Control and Status Register

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	COUNTFLAG
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	CLKSOURCE	TICKINT	ENABLE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-17	−	R	Read as 0.
16	COUNTFLAG	R/W	0: Timer not counted to 0 1: Timer counted to 0 Returns "1" if timer counted to "0" since last time this was read. Clears on read of any part of the SysTick Control and Status Register.
15-3	−	R	Read as 0.
2	CLKSOURCE	R/W	0: External reference clock (fosc/32) (Note) 1: CPU clock (fsys)
1	TICKINT	R/W	0: Do not pend SysTick 1: Pend SysTick
0	ENABLE	R/W	0: Disable 1: Enable If "1" is set, it reloads with the value of the Reload Value Register and starts operation.

Note: In this product, fosc by 32 is used as external reference clock.

7.6.2.2   SysTick Reload Value Register

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	RELOAD							
After reset	Undefined							
	15	14	13	12	11	10	9	8
bit symbol	RELOAD							
After reset	Undefined							
	7	6	5	4	3	2	1	0
bit symbol	RELOAD							
After reset	Undefined							

Bit	Bit Symbol	Type	Function
31-24	−	R	Read as 0,
23-0	RELOAD	R/W	Reload value Set the value to load into the SysTick Current Value Register when the timer reaches "0".

7.6.2.3   SysTick Current Value Register

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	CURRENT							
After reset	Undefined							
	15	14	13	12	11	10	9	8
bit symbol	CURRENT							
After reset	Undefined							
	7	6	5	4	3	2	1	0
bit symbol	CURRENT							
After reset	Undefined							

Bit	Bit Symbol	Type	Function
31-24	−	R	Read as 0.
23-0	CURRENT	R/W	[Read] Current SysTick timer value [Write] Clear Writing to this register with any value clears it to 0. Clearing this register also clears the <COUNTFLAG> bit of the SysTick Control and Status Register.

7.6.2.4   SysTick Calibration Value Register

	31	30	29	28	27	26	25	24
bit symbol	NOREF	SKEW	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	TENMS							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TENMS							
After reset	0	0	0	0	1	0	0	1
	7	6	5	4	3	2	1	0
bit symbol	TENMS							
After reset	1	1	0	0	0	1	0	0

Bit	Bit Symbol	Type	Function
31	NOREF	R	0: Reference clock provided 1: No reference clock
30	SKEW	R	0: Calibration value is 10 ms. 1: Calibration value is not 10ms.
29-24	–	R	Read as 0.
23-0	TENMS	R	Calibration value Reload value to use for 10 ms timing (0x9C4) by external reference clock (Note).

Note: In the case of a multishot, please use <TENMS>-1.

7.6.2.5 Interrupt Set-Enable Register 1

	31	30	29	28	27	26	25	24
bit symbol	-	SETENA (Interrupt 30)	SETENA (Interrupt 29)	SETENA (Interrupt 28)	SETENA (Interrupt 27)	SETENA (Interrupt 26)	SETENA (Interrupt 25)	SETENA (Interrupt 24)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	SETENA (Interrupt 23)	SETENA (Interrupt 22)	SETENA (Interrupt 21)	SETENA (Interrupt 20)	SETENA (Interrupt 19)	SETENA (Interrupt 18)	SETENA (Interrupt 17)	SETENA (Interrupt 16)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SETENA (Interrupt 7)	SETENA (Interrupt 6)	SETENA (Interrupt 5)	SETENA (Interrupt 4)	SETENA (Interrupt 3)	SETENA (Interrupt 2)	SETENA (Interrupt 1)	SETENA (Interrupt 0)
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	–	R/W	Write as 0.
30-16	SETENA	R/W	Interrupt number [30:16] [Write] 1: Enable [Read] 0: Disabled 1: Enabled Each bit corresponds to the specified number of interrupts. Writing "1" to a bit in this register enables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
15-8	–	R/W	Write as 0.
7-0	SETENA	R/W	Interrupt number [7:0] [Write] 1: Enable [Read] 0: Disabled 1: Enabled Each bit corresponds to the specified number of interrupts. Writing "1" to a bit in this register enables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.6 Interrupt Set-Enable Register 2

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	SETENA (Interrupt 59)	SETENA (Interrupt 58)	SETENA (Interrupt 57)	SETENA (Interrupt 56)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	SETENA (Interrupt 55)	SETENA (Interrupt 54)	SETENA (Interrupt 53)	SETENA (Interrupt 52)	SETENA (Interrupt 51)	SETENA (Interrupt 50)	SETENA (Interrupt 49)	SETENA (Interrupt 48)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	SETENA (Interrupt 47)	SETENA (Interrupt 46)	SETENA (Interrupt 45)	SETENA (Interrupt 44)	SETENA (Interrupt 43)	SETENA (Interrupt 42)	SETENA (Interrupt 41)	SETENA (Interrupt 40)
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SETENA (Interrupt 39)	SETENA (Interrupt 38)	SETENA (Interrupt 37)	-	SETENA (Interrupt 35)	SETENA (Interrupt 34)	SETENA (Interrupt 33)	SETENA (Interrupt 32)
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-28	–	R/W	Write as 0.
27-5	SETENA	R/W	Interrupt number [59:37] [Write] 1: Enable [Read] 0: Disabled 1: Enabled Each bit corresponds to the specified number of interrupts. Writing "1" to a bit in this register enables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
4	–	R/W	Write as 0.
3-0	SETENA	R/W	Interrupt number [35:32] [Write] 1: Enable [Read] 0: Disabled 1: Enabled Each bit corresponds to the specified number of interrupts. Writing "1" to a bit in this register enables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.7   Interrupt Set-Enable Register 3

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	SETENA (Interrupt 87)	SETENA (Interrupt 86)	SETENA (Interrupt 85)	SETENA (Interrupt 84)	SETENA (Interrupt 83)	SETENA (Interrupt 82)	SETENA (Interrupt 81)	SETENA (Interrupt 80)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	SETENA (Interrupt 79)	SETENA (Interrupt 78)	SETENA (Interrupt 77)	SETENA (Interrupt 76)	SETENA (Interrupt 75)	SETENA (Interrupt 74)	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-24	–	R/W	Write as 0.
23-10	SETENA	R/W	Interrupt number [87:74] [Write] 1: Enable [Read] 0: Disabled 1: Enabled Each bit corresponds to the specified number of interrupts. Writing "1" to a bit in this register enables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
9-0	–	R/W	Write as 0.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.8 Interrupt Set-Enable Register 4

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	SETENA (Interrupt 99)	SETENA (Interrupt 98)	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	R	Read as 0,
3-2	SETENA	R/W	Interrupt number [99:98] [Write] 1: Enable [Read] 0: Disabled 1: Enabled Each bit corresponds to the specified number of interrupts. Writing "1" to a bit in this register enables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
1-0	–	R/W	Write as 0.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.9   Interrupt Clear-Enable Register 1

	31	30	29	28	27	26	25	24
bit symbol	-	CLRENA (Interrupt 30)	CLRENA (Interrupt 29)	CLRENA (Interrupt 28)	CLRENA (Interrupt 27)	CLRENA (Interrupt 26)	CLRENA (Interrupt 25)	CLRENA (Interrupt 24)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	CLRENA (Interrupt 23)	CLRENA (Interrupt 22)	CLRENA (Interrupt 21)	CLRENA (Interrupt 20)	CLRENA (Interrupt 19)	CLRENA (Interrupt 18)	CLRENA (Interrupt 17)	CLRENA (Interrupt 16)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CLRENA (Interrupt 7)	CLRENA (Interrupt 6)	CLRENA (Interrupt 5)	CLRENA (Interrupt 4)	CLRENA (Interrupt 3)	CLRENA (Interrupt 2)	CLRENA (Interrupt 1)	CLRENA (Interrupt 0)
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	–	R/W	Write as 0.
30-16	CLRENA	R/W	Interrupt number [30:16] [Write] 1: Disabled [Read] 0: Disabled 1: Enable Each bit corresponds to the specified number of interrupts. It can be performed to enable interrupts and to check if interrupts are disabled. Writing "1" to a bit in this register disables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
15-8	–	R/W	Write as 0.
7-0	CLRENA	R/W	Interrupt number [7:0] [Write] 1: Disabled [Read] 0: Disabled 1: Enable Each bit corresponds to the specified number of interrupts. It can be performed to enable interrupts and to check if interrupts are disabled. Writing "1" to a bit in this register disables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.10   Interrupt Clear-Enable Register 2

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	CLRENA (Interrupt 59)	CLRENA (Interrupt 58)	CLRENA (Interrupt 57)	CLRENA (Interrupt 56)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	CLRENA (Interrupt 55)	CLRENA (Interrupt 54)	CLRENA (Interrupt 53)	CLRENA (Interrupt 52)	CLRENA (Interrupt 51)	CLRENA (Interrupt 50)	CLRENA (Interrupt 49)	CLRENA (Interrupt 48)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	CLRENA (Interrupt 47)	CLRENA (Interrupt 46)	CLRENA (Interrupt 45)	CLRENA (Interrupt 44)	CLRENA (Interrupt 43)	CLRENA (Interrupt 42)	CLRENA (Interrupt 41)	CLRENA (Interrupt 40)
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CLRENA (Interrupt 39)	CLRENA (Interrupt 38)	CLRENA (Interrupt 37)	-	CLRENA (Interrupt 35)	CLRENA (Interrupt 34)	CLRENA (Interrupt 33)	CLRENA (Interrupt 32)
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-28	–	R/W	Write as 0.
27-5	CLRENA	R/W	Interrupt number [59:37] [Write] 1: Disabled [Read] 0: Disabled 1: Enable Each bit corresponds to the specified number of interrupts. It can be performed to enable interrupts and to check if interrupts are disabled. Writing "1" to a bit in this register disables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
4	–	R/W	Write as 0.
3-0	CLRENA	R/W	Interrupt number [35:32] [Write] 1: Disabled [Read] 0: Disabled 1: Enable Each bit corresponds to the specified number of interrupts. It can be performed to enable interrupts and to check if interrupts are disabled. Writing "1" to a bit in this register disables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.11   Interrupt Clear-Enable Register 3

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	CLRENA (Interrupt 87)	CLRENA (Interrupt 86)	CLRENA (Interrupt 85)	CLRENA (Interrupt 84)	CLRENA (Interrupt 83)	CLRENA (Interrupt 82)	CLRENA (Interrupt 81)	CLRENA (Interrupt 80)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	CLRENA (Interrupt 79)	CLRENA (Interrupt 78)	CLRENA (Interrupt 77)	CLRENA (Interrupt 76)	CLRENA (Interrupt 75)	CLRENA (Interrupt 74)	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-24	−	R/W	Write as 0.
23-10	CLRENA	R/W	Interrupt number [87:74] [Write] 1: Disabled [Read] 0: Disabled 1: Enable Each bit corresponds to the specified number of interrupts. It can be performed to enable interrupts and to check if interrupts are disabled. Writing "1" to a bit in this register disables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
9-0	−	R/W	Write as 0.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.12 Interrupt Clear-Enable Register 4

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	CLRENA (Interrupt 99)	CLRENA (Interrupt 98)	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as 0,
3-2	CLRENA	R/W	Interrupt number [99:98] [Write] 1: Disabled [Read] 0: Disabled 1: Enable Each bit corresponds to the specified number of interrupts. It can be performed to enable interrupts and to check if interrupts are disabled. Writing "1" to a bit in this register disables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
1-0	−	R/W	Write as 0.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.13   Interrupt Set-Pending Register 1

	31	30	29	28	27	26	25	24
bit symbol	-	SETPEND (Interrupt 30)	SETPEND (Interrupt 29)	SETPEND (Interrupt 28)	SETPEND (Interrupt 27)	SETPEND (Interrupt 26)	SETPEND (Interrupt 25)	SETPEND (Interrupt 24)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	SETPEND (Interrupt 23)	SETPEND (Interrupt 22)	SETPEND (Interrupt 21)	SETPEND (Interrupt 20)	SETPEND (Interrupt 19)	SETPEND (Interrupt 18)	SETPEND (Interrupt 17)	SETPEND (Interrupt 16)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	SETPEND (Interrupt 7)	SETPEND (Interrupt 6)	SETPEND (Interrupt 5)	SETPEND (Interrupt 4)	SETPEND (Interrupt 3)	SETPEND (Interrupt 2)	SETPEND (Interrupt 1)	SETPEND (Interrupt 0)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31	–	R/W	Write as 0.
30-16	SETPEND	R/W	Interrupt number [30:16] [Write] 1: Pend [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register pends the corresponding interrupt. However, writing "1" has no effect on an interrupt that is already pending or is disabled. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts. Writing "1" to a corresponding bit in the Interrupt Clear-Pending Register clears the bit in this register.
15-8	–	R/W	Write as 0.
7-0	SETPEND	R/W	Interrupt number [7:0] [Write] 1: Pend [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register pends the corresponding interrupt. However, writing "1" has no effect on an interrupt that is already pending or is disabled. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts. Writing "1" to a corresponding bit in the Interrupt Clear-Pending Register clears the bit in this register.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.14 Interrupt Set-Pending Register 2

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	SETPEND (Interrupt 59)	SETPEND (Interrupt 58)	SETPEND (Interrupt 57)	SETPEND (Interrupt 56)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	SETPEND (Interrupt 55)	SETPEND (Interrupt 54)	SETPEND (Interrupt 53)	SETPEND (Interrupt 52)	SETPEND (Interrupt 51)	SETPEND (Interrupt 50)	SETPEND (Interrupt 49)	SETPEND (Interrupt 48)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	SETPEND (Interrupt 47)	SETPEND (Interrupt 46)	SETPEND (Interrupt 45)	SETPEND (Interrupt 44)	SETPEND (Interrupt 43)	SETPEND (Interrupt 42)	SETPEND (Interrupt 41)	SETPEND (Interrupt 40)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	SETPEND (Interrupt 39)	SETPEND (Interrupt 38)	SETPEND (Interrupt 37)	-	SETPEND (Interrupt 35)	SETPEND (Interrupt 34)	SETPEND (Interrupt 33)	SETPEND (Interrupt 32)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-28	–	R/W	Write as 0.
27-5	SETPEND	R/W	Interrupt number [59:37] [Write] 1: Pend [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register pends the corresponding interrupt. However, writing "1" has no effect on an interrupt that is already pending or is disabled. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts. Writing "1" to a corresponding bit in the Interrupt Clear-Pending Register clears the bit in this register.
4	–	R/W	Write as 0.
3-0	SETPEND	R/W	Interrupt number [35:32] [Write] 1: Pend [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register pends the corresponding interrupt. However, writing "1" has no effect on an interrupt that is already pending or is disabled. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts. Writing "1" to a corresponding bit in the Interrupt Clear-Pending Register clears the bit in this register.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.15   Interrupt Set-Pending Register 3

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	SETPEND (Interrupt 87)	SETPEND (Interrupt 86)	SETPEND (Interrupt 85)	SETPEND (Interrupt 84)	SETPEND (Interrupt 83)	SETPEND (Interrupt 82)	SETPEND (Interrupt 81)	SETPEND (Interrupt 80)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	SETPEND (Interrupt 79)	SETPEND (Interrupt 78)	SETPEND (Interrupt 77)	SETPEND (Interrupt 76)	SETPEND (Interrupt 75)	SETPEND (Interrupt 74)	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-24	−	R/W	Write as 0.
23-10	SETPEND	R/W	Interrupt number [87:74] [Write] 1: Pend [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register pends the corresponding interrupt. However, writing "1" has no effect on an interrupt that is already pending or is disabled. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts. Writing "1" to a corresponding bit in the Interrupt Clear-Pending Register clears the bit in this register.
9-0	−	R/W	Write as 0.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.16   Interrupt Set-Pending Register 4

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	SETPEND (Interrupt 99)	SETPEND (Interrupt 98)	-	-
After reset	0	0	0	0	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as 0,
3-2	SETPEND	R/W	Interrupt number [99:98] [Write] 1: Pend [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register pends the corresponding interrupt. However, writing "1" has no effect on an interrupt that is already pending or is disabled. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts. Writing "1" to a corresponding bit in the Interrupt Clear-Pending Register clears the bit in this register.
1-0	−	R/W	Write as 0.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.17 Interrupt Clear-Pending Register 1

	31	30	29	28	27	26	25	24
bit symbol	-	CLRPEND (Interrupt 30)	CLRPEND (Interrupt 29)	CLRPEND (Interrupt 28)	CLRPEND (Interrupt 27)	CLRPEND (Interrupt 26)	CLRPEND (Interrupt 25)	CLRPEND (Interrupt 24)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	CLRPEND (Interrupt 23)	CLRPEND (Interrupt 22)	CLRPEND (Interrupt 21)	CLRPEND (Interrupt 20)	CLRPEND (Interrupt 19)	CLRPEND (Interrupt 18)	CLRPEND (Interrupt 17)	CLRPEND (Interrupt 16)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	CLRPEND (Interrupt 7)	CLRPEND (Interrupt 6)	CLRPEND (Interrupt 5)	CLRPEND (Interrupt 4)	CLRPEND (Interrupt 3)	CLRPEND (Interrupt 2)	CLRPEND (Interrupt 1)	CLRPEND (Interrupt 0)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31	–	R/W	Write as 0.
30-16	CLRPEND	R/W	Interrupt number [30:16] [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register clears the corresponding pending interrupt. However, writing "1" has no effect on an interrupt that is already being serviced. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts.
15-8	–	R/W	Write as 0.
7-0	CLRPEND	R/W	Interrupt number [7:0] [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register clears the corresponding pending interrupt. However, writing "1" has no effect on an interrupt that is already being serviced. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.18 Interrupt Clear-Pending Register 2

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	CLRPEND (Interrupt 59)	CLRPEND (Interrupt 58)	CLRPEND (Interrupt 57)	CLRPEND (Interrupt 56)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	CLRPEND (Interrupt 55)	CLRPEND (Interrupt 54)	CLRPEND (Interrupt 53)	CLRPEND (Interrupt 52)	CLRPEND (Interrupt 51)	CLRPEND (Interrupt 50)	CLRPEND (Interrupt 49)	CLRPEND (Interrupt 48)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	CLRPEND (Interrupt 47)	CLRPEND (Interrupt 46)	CLRPEND (Interrupt 45)	CLRPEND (Interrupt 44)	CLRPEND (Interrupt 43)	CLRPEND (Interrupt 42)	CLRPEND (Interrupt 41)	CLRPEND (Interrupt 40)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	CLRPEND (Interrupt 39)	CLRPEND (Interrupt 38)	CLRPEND (Interrupt 37)	-	CLRPEND (Interrupt 35)	CLRPEND (Interrupt 34)	CLRPEND (Interrupt 33)	CLRPEND (Interrupt 32)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-28	−	R/W	Write as 0.
27-5	CLRPEND	R/W	Interrupt number [59:37] [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register clears the corresponding pending interrupt. However, writing "1" has no effect on an interrupt that is already being serviced. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts.
4	−	R/W	Write as 0.
3-0	CLRPEND	R/W	Interrupt number [35:32] [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register clears the corresponding pending interrupt. However, writing "1" has no effect on an interrupt that is already being serviced. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.19   Interrupt Clear-Pending Register 3

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	CLRPEND (Interrupt 87)	CLRPEND (Interrupt 86)	CLRPEND (Interrupt 85)	CLRPEND (Interrupt 84)	CLRPEND (Interrupt 83)	CLRPEND (Interrupt 82)	CLRPEND (Interrupt 81)	CLRPEND (Interrupt 80)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	CLRPEND (Interrupt 79)	CLRPEND (Interrupt 78)	CLRPEND (Interrupt 77)	CLRPEND (Interrupt 76)	CLRPEND (Interrupt 75)	CLRPEND (Interrupt 74)	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-24	−	R/W	Write as 0.
23-10	CLRPEND	R/W	Interrupt number [87:74] [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register clears the corresponding pending interrupt. However, writing "1" has no effect on an interrupt that is already being serviced. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts.
9-0	−	R/W	Write as 0.

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".

7.6.2.20 Interrupt Clear-Pending Register 4

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	CLRPEND (Interrupt 99)	CLRPEND (Interrupt 98)	-	-
After reset	0	0	0	0	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as 0,
3-2	CLRPEND	R/W	Interrupt number [99:98] [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register clears the corresponding pending interrupt. However, writing "1" has no effect on an interrupt that is already being serviced. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts.
1-0	−	R/W	Write as 0.

Note: **For descriptions of interrupts and interrupt numbers, see Section "7.5.1.5 List of Interrupt Sources".**

7.6.2.21   Interrupt Priority Register

Each interrupt is provided with eight bits of an Interrupt Priority Register.

The following shows the addresses of the Interrupt Priority Registers corresponding to interrupt numbers.

	31 24	23 16	15 8	7 0
0xE000_E400	PRI_3	PRI_2	PRI_1	PRI_0
0xE000_E404	PRI_7	PRI_6	PRI_5	PRI_4
0xE000_E408	–	–	–	–
0xE000_E40C	–	–	–	–
0xE000_E410	PRI_19	PRI_18	PRI_17	PRI_16
0xE000_E414	PRI_23	PRI_22	PRI_21	PRI_20
0xE000_E418	PRI_27	PRI_26	PRI_25	PRI_24
0xE000_E41C	–	PRI_30	PRI_29	PRI_28
0xE000_E420	PRI_35	PRI_34	PRI_33	PRI_32
0xE000_E424	PRI_39	PRI_38	PRI_37	–
0xE000_E428	PRI_43	PRI_42	PRI_41	PRI_40
0xE000_E42C	PRI_47	PRI_46	PRI_45	PRI_44
0xE000_E430	PRI_51	PRI_50	PRI_49	PRI_48
0xE000_E434	PRI_55	PRI_54	PRI_53	PRI_52
0xE000_E438	PRI_59	PRI_58	PRI_57	PRI_56
0xE000_E43C	–	–	–	–
0xE000_E440	–	–	–	–
0xE000_E444	–	–	–	–
0xE000_E448	PRI_75	PRI_74	–	–
0xE000_E44C	PRI_79	PRI_78	PRI_77	PRI_76
0xE000_E450	PRI_83	PRI_82	PRI_81	PRI_80
0xE000_E454	PRI_87	PRI_86	PRI_85	PRI_84
0xE000_E458	–	–	–	–
0xE000_E45C	–	–	–	–
0xE000_E460	PRI_99	PRI_98	–	–

The number of bits to be used for assigning a priority varies with each product. This product uses three bits for assigning a priority.

The following shows the fields of the Interrupt Priority Registers for interrupt numbers 0 to 3. The Interrupt Priority Registers for all other interrupt numbers have the identical fields. Unused bits return "0" when read, and writing to unused bits has no effect.

	31	30	29	28	27	26	25	24
bit symbol		PRI_3		–	–	–	–	–
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol		PRI_2		–	–	–	–	–
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol		PRI_1		–	–	–	–	–
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol		PRI_0		–	–	–	–	–
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-29	PRI_3	R/W	Priority of interrupt number 3
28-24	–	R	Read as 0.
23-21	PRI_2	R/W	Priority of interrupt number 2
20-16	–	R	Read as 0.
15-13	PRI_1	R/W	Priority of interrupt number 1
12-8	–	R	Read as 0.
7-5	PRI_0	R/W	Priority of interrupt number 0
4-0	–	R	Read as 0.

7.6.2.22   Vector Table Offset Register

	31	30	29	28	27	26	25	24
bit symbol	-	-	TBLBASE	TBLOFF				
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	TBLOFF							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBLOFF							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBLOFF	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-30	−	R	Read as 0,
29	TBLBASE	R/W	Table base The vector table is in: 0: Code space 1: SRAM space
28-7	TBLOFF	R/W	Offset value Set the offset value from the top of the space specified in TBLBASE. The offset must be aligned based on the number of exceptions in the table.This means that the minimum alignment is 32 words that you can use for up to 16 interrupts.For more interrupts, you must adjust the alignment by rounding up to the next power of two.
6-0	−	R	Read as 0,

7.6.2.23 Application Interrupt and Reset Control Register

	31	30	29	28	27	26	25	24
bit symbol	VECTKEY/VECTKEYSTAT							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	VECTKEY/VECTKEYSTAT							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ENDIANESS	-	-	-	-	PRIGROUP		
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	SYSRESET REQ	VECTCLR ACTIVE	VECTRESET
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	VECTKEY (Written) / VECTKEYSTAT (Read)	R/W	Register key [Write] Writing to this register requires 0x5FA in the <VECTKEY> field. [Read] Read as 0xFA05.
15	ENDIANESS	R/W	Endianness bit: (Note1) 1: Big endian 0: Little endianl
14-11	–	R	Read as 0,
10-8	PRIGROUP	R/W	Interrupt priority grouping 000: seven bits of pre-emption priority, one bit of subpriority 001: six bits of pre-emption priority, two bits of subpriority 010: five bits of pre-emption priority, three bits of subpriority 011: four bits of pre-emption priority, four bits of subpriority 100: three bits of pre-emption priority, five bits of subpriority 101: two bits of pre-emption priority, six bits of subpriority 110: one bit of pre-emption priority, seven bits of subpriority 111: no pre-emption priority, eight bits of subpriority The bit configuration to split the interrupt priority register <PRI_n> into pre-emption priority and sub priority.
7-3	–	R	Read as 0,
2	SYSRESET REQ	R/W	System Reset Request 1=CPU outputs a SYSRESETREQ signal. (note2)
1	VECTCLR ACTIVE	R/W	Clear active vector bit 1: clear all state information for active NMI, fault, and interrupts. 0: do not clear. This bit self-clears. It it the responsibility of the application to reinitialize the stack.
0	VECTRESET	R/W	System Reset bit 1: reset system. 0: do not reset system. Resets the system, with the exception of debug components (FPB, DWT and ITM) by setting "1" and this bit is also zero cleared.

Note 1: **Little-endian is the default memory format for this product.**

Note 2: **When SYSRESETREQ is output, warm reset is performed on this product. <SYSRESETREQ> is cleared by warm reset.**

7.6.2.24   System Handler Priority Register

Each exception is provided with eight bits of a System Handler Priority Register.

The following shows the addresses of the System Handler Priority Registers corresponding to each exception.

	31 24	23 16	15 8	7 0
0xE000_ED18	PRI_7	PRI_6 (Usage Fault)	PRI_5 (Bus Fault)	PRI_4 (Memory Management)
0xE000_ED1C	PRI_11 (SVCall)	PRI_10	PRI_9	PRI_8
0xE000_ED20	PRI_15 (SysTick)	PRI_14 (PendSV)	PRI_13	PRI_12 (Debug Monitor)

The number of bits to be used for assigning a priority varies with each product. This product uses three bits for assigning a priority.

The following shows the fields of the System Handler Priority Registers for Memory Management, Bus Fault and Usage Fault. Unused bits return "0" when read, and writing to unused bits has no effect.

	31	30	29	28	27	26	25	24
bit symbol	PRI_7			-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	PRI_6			-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	PRI_5			-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PRI_4			-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-29	PRI_7	R/W	Reserved
28-24	−	R	Read as 0,
23-21	PRI_6	R/W	Priority of Usage Fault
20-16	−	R	Read as 0,
15-13	PRI_5	R/W	Priority of Bus Fault
12-8	−	R	Read as 0,
7-5	PRI_4	R/W	Priority of Memory Management
4-0	−	R	Read as 0,

7.6.2.25   System Handler Control and State Register

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	USGFAULT ENA	BUSFAULT ENA	MEMFAULT ENA
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	SVCALL PENDED	BUSFAULT PENDED	MEMFAULT PENDED	USGFAULT PENDED	SYSTICKACT	PENDSVACT	-	MONITOR ACT
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SVCALLACT	-	-	-	USGFAULT ACT	-	BUSFAULT ACT	MEMFAULT ACT
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-19	–	R	Read as 0,
18	USGFAULT ENA	R/W	Usage Fault 0: Disabled 1: Enabled
17	BUSFAUL TENA	R/W	Bus Fault 0: Disable 1: Enable
16	MEMFAULT ENA	R/W	Memory Management 0: Disable 1: Enable
15	SVCALL PENDED	R/W	SVCall 0: Not pended 1: Pended
14	BUSFAULT PENDED	R/W	Bus Fault 0: Not pended 1: Pended
13	MEMFAULT PENDED	R/W	Memory Management 0: Not pended 1: Pended
12	USGFAULT PENDED	R/W	Usage Fault 0: Not pended 1: Pended
11	SYSTICKACT	R/W	SysTick 0: Inactive 1: Active
10	PENDSVACT	R/W	PendSV 0: Inactive 1: Active
9	–	R	Read as 0,
8	MONITORACT	R/W	Debug monitor 0: Inactive 1: Active
7	SVCALLACT	R/W	SVCall 0: Inactive 1: Active
6-4	–	R	Read as 0,

Bit	Bit Symbol	Type	Function
3	USGFAULT ACT	R/W	Usage Fault 0: Inactive 1: Active
2	–	R	Read as 0,
1	BUSFAULT ACT	R/W	Bus Fault 0: Inactive 1: Active
0	MEMFAULT ACT	R/W	Memory management 0: Inactive 1: Active

Note: You must clear or set the active bits with extreme caution because clearing and setting these bits does not re-pair stack contents.

7.6.3   Clock generator registers

7.6.3.1   CGIMCGA (CG Interrupt Mode Control Register A)

	31	30	29	28	27	26	25	24
bit symbol	-	EMCG3			EMST3		-	INT3EN
After reset	0	0	1	0	0	0	Undefined	0
	23	22	21	20	19	18	17	16
bit symbol	-	EMCG2			EMST2		-	INT2EN
After reset	0	0	1	0	0	0	Undefined	0
	15	14	13	12	11	10	9	8
bit symbol	-	EMCG1			EMST1		-	INT1EN
After reset	0	0	1	0	0	0	Undefined	0
	7	6	5	4	3	2	1	0
bit symbol	-	EMCG0			EMST0		-	INT0EN
After reset	0	0	1	0	0	0	Undefined	0

Bit	Bit Symbol	Type	Function
31	–	R	Read as 0,
30-28	EMCG3[2:0]	R/W	active level setting of INT3 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edge
27-26	EMST3[1:0]	R	active level of INT3 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edge
25	–	R	Reads as undefined.
24	INT3EN	R/W	INT3 clear input 0: Disable 1: Enable
23	–	R	Read as 0,
22-20	EMCG2[2:0]	R/W	active level setting of INT2 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edge
19-18	EMST2[1:0]	R	active level of INT2 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edge
17	–	R	Reads as undefined.
16	INT2EN	R/W	INT2 clear input 0:Disable 1: Enable
15	–	R	Read as 0,

Bit	Bit Symbol	Type	Function
14-12	EMCG1[2:0]	R/W	active level setting of INT1 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edge
11-10	EMST1[1:0]	R	active level of INT1 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edge
9	−	R	Reads as undefined.
8	INT1EN	R/W	INT1 clear input 0: Disable 1: Enable
7	−	R	Read as 0,
6-4	EMCG0[2:0]	R/W	active level setting of INT0 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edge
3-2	EMST0[1:0]	R	active level of INT0 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edge
1	−	R	Reads as undefined.
0	INT0EN	R/W	INT0 clear input 0: Disable 1: Enable

Note 1: <EMSTx> is effective only when <EMCGx[2:0]> is set to "100" for both rising and falling edge. The active level used for the reset of standby can be checked by referring <EMSTx>. If interrupts are cleared with the CGICRCG register, <EMSTx> is also cleared.

Note 2: Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

7.6.3.2 CGIMCGB (CG Interrupt Mode Control Register B)

	31	30	29	28	27	26	25	24
bit symbol	-		EMCG7			EMST7	-	INT7EN
After reset	0	0	1	0	0	0	Undefined	0
	23	22	21	20	19	18	17	16
bit symbol	-		EMCG6			EMST6	-	INT6EN
After reset	0	0	1	0	0	0	Undefined	0
	15	14	13	12	11	10	9	8
bit symbol	-		EMCG5			EMST5	-	INT5EN
After reset	0	0	1	0	0	0	Undefined	0
	7	6	5	4	3	2	1	0
bit symbol	-		EMCG4			EMST4	-	INT4EN
After reset	0	0	1	0	0	0	Undefined	0

Bit	Bit Symbol	Type	Function
31	−	R	Read as 0,
30-28	EMCG7[2:0]	R/W	active level setting of INT7 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edge
27-26	EMST7[1:0]	R	active level of INT7 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edge
25	−	R	Reads as undefined.
24	INT7EN	R/W	INT7 clear input 0: Disable 1: Enable
23	−	R	Read as 0,
22-20	EMCG6[2:0]	R/W	active level setting of INT6 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edge
19-18	EMST6[1:0]	R	active level of INT6 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edge
17	−	R	Reads as undefined.
16	INT6EN	R/W	INT6 clear input 0:Disable 1: Enable
15	−	R	Read as 0,
14-12	EMCG5[2:0]	R/W	active level setting of INT5 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edge

Bit	Bit Symbol	Type	Function
11-10	EMST5[1:0]	R	active level of INT5 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edge
9	−	R	Reads as undefined.
8	INT5EN	R/W	INT5 clear input 0: Disable 1: Enable
7	−	R	Read as 0,
6-4	EMCG4[2:0]	R/W	active level setting of INT4 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edge
3-2	EMST4[1:0]	R	active level of INT4 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edge
1	−	R	Reads as undefined.
0	INT4EN	R/W	INT4 clear input 0: Disable 1: Enable

Note 1: <EMSTx> is effective only when <EMCGx[2:0]> is set to "100" for both rising and falling edge. The active level used for the reset of standby can be checked by referring <EMSTx>. If interrupts are cleared with the CGICRCG register, <EMSTx> is also cleared.

Note 2: Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

7.6.3.3 CGIMCGE (CG Interrupt Mode Control Register E)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	1	0	0	0	Undefined	0
	23	22	21	20	19	18	17	16
bit symbol	-	EMCGI			EMSTI		-	INTIEN
After reset	0	0	1	0	0	0	Undefined	0
	15	14	13	12	11	10	9	8
bit symbol	-	EMCGH			EMSTH		-	INTHEN
After reset	0	0	1	0	0	0	Undefined	0
	7	6	5	4	3	2	1	0
bit symbol	-	EMCGG			EMSTG		-	INTGEN
After reset	0	0	1	0	0	0	Undefined	0

Bit	Bit Symbol	Type	Function
31	–	R	Read as 0,
30-28	–	R/W	Write optional value.
27-26	–	R	Read as 0,
25	–	R	Read as undefined.
24	–	R/W	Write as 0.
23	–	R	Read as 0,
22-20	EMCGI[2:0]	R/W	active level setting of INTRMCRX standby clear request. Set it as shown below. 011: Rising edge
19-18	EMSTI[1:0]	R	R active level of INTRMCRX standby clear request. 00: – 01: Rising edge 10: Falling edge 11: Both edge
17	–	R	Read as undefined.
16	INTIEN	R/W	INTRMCRX clear input 0:Disable 1: Enable
15	–	R	Read as 0,
14-12	EMCGH[2:0]	R/W	active level setting of INTCECTX standby clear request. Set it as shown below. 011: Riseing edge
11-10	EMSTH[1:0]	R	active level of INTCECTX standby clear request. 00: – 01: Rising edge 10: Falling edge 11: Both edge
9	–	R	Read as undefined
8	INTHEN	R/W	INTCECTX Clear input 0:Disable 1: Enable
7	–	R	Read as 0,
6-4	EMCGG[2:0]	R/W	active level setting of INTCECRX standby clear request. Set it as shown below. 011: Rising edge

Bit	Bit Symbol	Type	Function
3-2	EMSTG[1:0]	R	active level of INTCECRX standby clear request. 00: − 01: Rising edge 10: Falling edge 11: Both edge
1	−	R	Read as undefined.
0	INTGEN	R/W	INTCECRX Clear input 0:Disable 1: Enable

Note 1: <EMSTx> is effective only when <EMCGx[2:0]> is set to "100" for both rising and falling edge. The active level used for the reset of standby can be checked by referring <EMSTx>. If interrupts are cleared with the CGICRCG register, <EMSTx> is also cleared.

Note 2: Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

7.6.3.4 CGIMCGF (CG Interrupt Mode Control Register F)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	EMCGL			EMSTL		-	INTLEN
After reset	0	0	1	0	0	0	Undefined	0
	7	6	5	4	3	2	1	0
bit symbol	-	EMCGK			EMSTK		-	INTKEN
After reset	0	0	1	0	0	0	Undefined	0

Bit	Bit Symbol	Type	Function
31-15	–	R	Read as 0,
14-12	EMCGL[2:0]	R/W	active level setting of INTKWUP standby clear request Set it as shown below. 001: "H" level
11-10	EMSTL[1:0]	R	active level of INTKWUP standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edge
9	–	R	Read as undefined.
8	INTLEN	R/W	INTKWUP clear input 0: Disable 1: Enable
7	–	R	Read as 0,
6-4	EMCGK[2:0]	R/W	active level setting of INTRTC standby clear request Set it as shown below. 010: Falling edge
3-2	EMSTK[1:0]	R	active level of INTRTC standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edge
1	–	R	Read as undefined.
0	INTKEN	R/W	INTRTC clear input 0: Disable 1: Enable

Note 1: <EMSTx> is effective only when <EMCGx[2:0]> is set to "100" for both rising and falling edge. The active level used for the reset of standby can be checked by referring <EMSTx>. If interrupts are cleared with the CGICRCG register, <EMSTx> is also cleared.

Note 2: Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

7.6.3.5   CGICRCG (CG Interrupt Request Clear Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	ICRCG				
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	–	R	Read as 0,
4-0	ICRCG[4:0]	W	Clear interrupt requests. 0_0000: INT0 0_1000: Reserved 1_0000: INTCECRX 0_0001: INT1 0_1001: Reserved 1_0001: INTCECTX 0_0010: INT2 0_1010: Reserved 1_0010: INTRMCRX 0_0011: INT3 0_1011: Reserved 1_0011: Reserved 0_0100: INT4 0_1100: Reserved 1_0100: INTRTC 0_0101: INT5 0_1101: Reserved 1_0101: INTKWUP 0_0110: INT6 0_1110: Reserved 1_0110 to 1_1111: Reserved 0_0111: INT7 0_1111: Reserved Read as 0.

7.6.3.6   CGNMIFLG (NMI Flag Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	NMIFLG1	NMIFLG0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	–	R	Read as 0,
1	NMIFLG1	R	NMI source generation flag 0: not applicable 1:generated from $\overline{\text{NMI}}$ pin.
0	NMIFLG0	R	NMI source generation flag 0: not applicable 1: generated from WDT

Note:<NMIFLG> are cleared to "0" when they are read.

7.6.3.7 CGRSTFLG (Reset Flag Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After pin reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After pin reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After pin reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	SYSRSTF	BUPRSTF	WDTRSTF	PINRSTF	PONRSTF
After pin reset	0	0	0	0	0	0	1	1

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as 0,
4	SYSRSTF	R/W	Debug reset flag (Note1) 0: "0" is written 1: Reset from SYSRESETREQ
3	BUPRSTF	R/W	BACKUP reset flag 0: "0" is written 1: Reset from BACKUP mode release
2	WDTRSTF	R/W	WDT reset flag 0: "0" is written 1: Reset from WDT
1	PINRSTF	R/W	RESET pin flag 0: "0" is written 1: Reset from RESET pin
0	PONRSTF	R/W	Power-on flag 0: "0" is written 1: Reset from power-on reset

Note 1: This flag indicates a reset generated by the SYSRESETREQ bit of the Application Interrupt and Reset Control Register of the CPU's NVIC.

Note 2: This product has power-on reset circuit and this register is initialized only by power-on reset. Therefore, "1" is set to the <PONRSTF> bit in initial reset state right after power-on. Note that this bit is not set by the second and subsequent resets and this register is not cleared automatically. Write "0" to clear the register.

8. Input / Output Ports

8.1 Port Functions

8.1.1 Function list

TMPM363F10FG has 74 ports. Besides the ports function, these ports can be used as I/O pins for peripheral functions.

Table 8-1 shows the port function table.

Table 8-1 Port Function List

Port	PIn	Input / Output	Program-mable Pull-up Pull-down	Schmitt Input	Noise Filter	Program-mable Open-drain	Function pin
Port A							
	PA0	I/O	Pull-up	–	–	o	D0 / AD0
	PA1	I/O	Pull-up	–	–	o	D1 / AD1
	PA2	I/O	Pull-up	–	–	o	D2 / AD2
	PA3	I/O	Pull-up	–	–	o	D3 / AD3
	PA4	I/O	Pull-up	–	–	o	D4 / AD4
	PA5	I/O	Pull-up	–	–	o	D5 / AD5
	PA6	I/O	Pull-up	–	–	o	D6 / AD6
	PA7	I/O	Pull-up	–	–	o	D7 / AD7
Port B							
	PB0	I/O	Pull-up	–	–	o	D8 / AD8
	PB1	I/O	Pull-up	–	–	o	D9 / AD9
	PB2	I/O	Pull-up	–	–	o	D10 / AD10
	PB3	I/O	Pull-up	–	–	o	D11 / AD11
	PB4	I/O	Pull-up	–	–	o	D12 / AD12
	PB5	I/O	Pull-up	–	–	o	D13 / AD13
	PB6	I/O	Pull-up	–	–	o	D14 / AD14
	PB7	I/O	Pull-up	–	–	o	D15 / AD15
Port E							
	PE0	I/O	Pull-up	o	–	o	A17 , TB5IN0
	PE1	I/O	Pull-up	o	–	o	A18 , TB5IN1
	PE2	I/O	Pull-up	o	–	o	A19 , TB6IN0
	PE3	I/O	Pull-up	o	–	o	A20 , TB6IN1
	PE4	I/O	Pull-up	o	–	o	A21 , TXD0 , CTXD
	PE5	I/O	Pull-up	o	–	o	A22 , RXD0 , CRXD
	PE6	I/O	Pull-up	o	–	o	A23 , SCLK0 , $\overline{\text{CTS0}}$
	PE7	I/O	Pull-up	o	o(INT5 only)	o	INT5 , SCOUT
Port F							
	PF0	I/O	Pull-up	o	–	o	TRACECLK
	PF1	I/O	Pull-up	o	–	o	TRACEDATA0 , SWV
	PF2	I/O	Pull-up	o	–	o	TRACEDATA1
	PF3	I/O	Pull-up	o	–	o	TRACEDATA2
	PF4	I/O	Pull-up	o	–	o	TRACEDATA3

Table 8-1 Port Function List

Port	PIn	Input / Output	Program-mable Pull-up Pull-down	Schmitt Input	Noise Filter	Program-mable Open-drain	Function pin
Port G							
	PG0	I/O	Pull-up	o	–	o	SDA1/ SO1 , TB7IN0
	PG1	I/O	Pull-up	o	–	o	SCL1/ SI1 , TB7IN1
	PG2	I/O	Pull-up	o	–	o	SCK1 , $\overline{CS0}$
	PG3	I/O	Pull-up	o	o(INT6 only)	o	INT6 , $\overline{CS1}$
	PG4	I/O	Pull-up	o	–	o	SDA2/ SO2 , TB9IN0
	PG5	I/O	Pull-up	o	–	o	SCL2/ SI2 , TB9IN1
	PG6	I/O	Pull-up	o	–	o	SCK2 , $\overline{CS3}$, USBPON
	PG7	I/O	Pull-up	o	o(INT7 only)	o	INT7 , $\overline{WDTOUT}$, $\overline{USBOC}$
Port I							
	PI0	I/O	Pull-up	o	–	o	$\overline{BOOT}$
	PI1	I/O	–	o	–	o(Note3)	CEC
Port J							
	PJ0	Input	Pull-up	o	–	–	AIN0
	PJ1	Input	Pull-up	o	–	–	AIN1
	PJ2	Input	Pull-up	o	–	–	AIN2
	PJ3	Input	Pull-up	o	–	–	AIN3 , $\overline{ADTRG}$
	PJ4	Input	Pull-up	o	o(KWUP0 only)	–	AIN4 , KWUP0
	PJ5	Input	Pull-up	o	o(KWUP1 only)	–	AIN5 , KWUP1
	PJ6	Input	Pull-up	o	o(KWUP2 only)	–	AIN6 , KWUP2
	PJ7	Input	Pull-up	o	o(KWUP3 only)	–	AIN7 , KWUP3
Port L							
	PL0	I/O	Pull-up	o	–	o	SDA0/ SO0 , TB0OUT
	PL1	I/O	Pull-up	o	–	o	SCL0/ SI0 , TB1OUT
	PL2	I/O	Pull-up	o	–	o	SCK0 , TB2OUT
	PL3	I/O	Pull-up	o	o(INT0 only)	o	INT0 , TB3OUT
	PL4	I/O	Pull-up	o	–	o	TXD1 , TB4OUT , SDA3
	PL5	I/O	Pull-up	o	–	o	RXD1 , TB5OUT , SCL3
	PL6	I/O	Pull-up	o	–	o	SCLK1 , TB6OUT , $\overline{CTS1}$
	PL7	I/O	Pull-up	o	o(INT1 only)	o	INT1 , TB7OUT
Port M							
	PM0	I/O	Pull-up	o	–	o	SCLK2 , TB1IN0 , $\overline{CTS2}$
	PM1	I/O	Pull-up	o	–	o	TXD2 , TB1IN1
	PM2	I/O	Pull-up	o	–	o	RXD2 , $\overline{ALARM}$
	PM3	I/O	Pull-up	o	o(INT2 only)	o	INT2 , TB3OUT
	PM4	I/O	Pull-up	o	–	o	SCLK3 , $\overline{CTS3}$
	PM5	I/O	Pull-up	o	–	o	TXD3
	PM6	I/O	Pull-up	o	–	o	RXD3
	PM7	I/O	Pull-up	o	o	o	INT3
Port N							
	PN0	I/O	Pull-up	o	–	o	TXD4
	PN1	I/O	Pull-up	o	–	o	RXD4
	PN2	I/O	Pull-up	o	–	o	SCLK4 , TB2IN0 , $\overline{CTS4}$
	PN3	I/O	Pull-up	o	o(INT4 only)	o	INT4 , TB2IN1 , RMC

Table 8-1 Port Function List

Port	PIn	Input / Output	Program-mable Pull-up Pull-down	Schmitt Input	Noise Filter	Program-mable Open-drain	Function pin
Port P							
	PP0	I/O	Pull-up	o	–	o	$\overline{CS2}$
	PP1	I/O	Pull-up	–	–	o	–
	PP2	I/O	Pull-up	o	–	o	$\overline{BLS0}$, SPDO
	PP3	I/O	Pull-up	–	–	o	$\overline{BLS1}$, SPDI
	PP4	I/O	Pull-up	–	–	o	$\overline{WE}$, SPCLK
	PP5	I/O	Pull-up	–	–	o	$\overline{OE}$, SPFSS
	PP6	I/O	Pull-up	o	–	o	$\overline{ALE}$

o : Exist

- : Not exist

Note 1: The noise elimination width of the noise filter is approximately 30 ns under typical conditions.

Note 2: The port is always pulled-up in spite of PIPUP.

Note 3: N-ch open drain port

8.1.2 Port Registers Outline

The following registers need to be configured to use ports.

- · PxDATA: Port x data register

    To read / write port data.

- · PxCR: Port x output control register

    To control output.

    PxIE needs to be configured to control input.

- · PxFRn: Port x function register n

    To set function.

    An assigned function can be activated by setting "1".

- · PxOD: Port x open drain control register

    To control the programmable open drain.

    Programmable open drain is function to be materialized pseudo-open-drain by setting the PxOD.

    When PxOD is set "1",output buffer is disabled and pseudo-open-drain is materialized.

- · PxPUP: Port x pull-up control register

    To control programmable pull ups.

- · PxPDN: Port x pull-down control register

    To control programmable pull downs.

- · PxIE : Port x input control register

    To control inputs.

    For avoided through current, default setting prohibits inputs.

8.1.3   Port states in STOP Mode

Input and output in STOP mode are enabled / disabled by the CGSTBYCR<DRVE> .

If PxIE or PxCR is enabled with <DRVE>=1, input or output is enabled respectively in STOP mode.If <DRVE>=0, both input and output are disabled in STOP mode except for some ports even if PxIE or PxCR are enabled.

Table 8-2 shows the pin conditions in STOP mode.

Table 8-2 Port conditions in STOP mode

	Pin name	I/O	<DRVE> = 0	<DRVE> = 1
Exclud-ing port	X1, XT1	Input only	×	×
	X2, XT2	Output only	"High" Level Output	"High"Level Output
	RESET, NMI, MODE	Input only	o	o
Port	PL3, PL7, PM3, PM7, PN3, PE7, PG3, PG7, (When used for interrupt (PxFRn<PxmFn>=1) and input is enabled (PxIE<PxmIE>=1)) (Note)	Input	o	o
		Output	×	Depend on PxCR[m]
	PJ4,PJ5, PJ6, PJ7 (When used for KWUP (PxFRn<PxmFn>=1) and input is enabled (PxIE<PxmIE>=1)) (Note)	Input	o	o
		Output	×	Depend on PxCR[m]
	PF0, PF1, PF2, PF3, PF4 (When used for trace data output (PxFRn<PxmFn>=1)) (Note)	Input	×	Depend on PxCR[m]
		Output	Depend on PxCR[m]	
	PA7-PA0, PB7-PB0, PE6-PE0, PP6-PP2, PP0 (When used for external bus (PxFRn<PxmFn>=1) and input is enabled for data bus (PxIE<PxmIE>=1)) (Note)	Input	o	o
		Output	Depend on PxCR[m]	
	Other ports	Input	×	Depend on PxCR[m]
		Output	×	Depend on PxCR[m]

o : Input or output enabled

× : Input or output disabled

Note:"x" indicates a port number, "m" a corresponding bit and "n" a function register number.

8.2 Port functions

This chapter describes the port registers detail.

This chapter describes only "circuit type" reading circuit configuration. For detailed circuit diagram, refer to "8.3 Block Diagrams of Ports".

8.2.1 Port A (PA0 to PA7)

The port A is a general-purpose, 8-bit input / output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input / output function, the port A performs the external bus interface.

Reset initializes all bits of the port A as general-purpose ports with input, output and pull-up disabled.

If this port is used for the external bus function, PACR, PAFR1 and PAIE must be set to "1".

The port A has a types of function register. If you use the port A as a general-purpose port, set "0" to the corresponding bit of the a registers. If you use the port A as other than a general-purpose port, set "1" to the corresponding bit of the function register.

8.2.1.1 Port A Circuit Type

	7	6	5	4	3	2	1	0
Type	T1	T1	T1	T1	T1	T1	T1	T1

8.2.1.2 Port A Register

Base Address = 0x400C_0000

Register name		Address (Base+)
Port A data register	PADATA	0x0000
Port A output control register	PACR	0x0004
Port A function register 1	PAFR1	0x0008
Port A open drain control register	PAOD	0x0028
Port A pull-up control register	PAPUP	0x002C
Port A input control register	PAIE	0x0038

8.2.1.3   PADATA (Port A data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PA7 to PA0	R/W	Port A data register

8.2.1.4   PACR (Port A output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PA7C	PA6C	PA5C	PA4C	PA3C	PA2C	PA1C	PA0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PA7C to PA0C	R/W	Output 0: Disable 1: Enable

8.2.1.5 PAFR1 (Port A function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PA7F1	PA6F1	PA5F1	PA4F1	PA3F1	PA2F1	PA1F1	PA0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PA7F1	R/W	0 : PORT 1 : D7, AD7
6	PA6F1	R/W	0: PORT 1: D6, AD6
5	PA5F1	R/W	0: PORT 1: D5, AD5
4	PA4F1	R/W	0: PORT 1: D4, AD4
3	PA3F1	R/W	0: PORT 1: D3, AD3
2	PA2F1	R/W	0: PORT 1: D2, AD2
1	PA1F1	R/W	0: PORT 1: D1, AD1
0	PA0F1	R/W	0: PORT 1: D0, AD0

8.2.1.6 PAOD (Port A open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PA7OD	PA6OD	PA5OD	PA4OD	PA3OD	PA2OD	PA1OD	PA0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31 to 8	−	R	Read as "0".
7 to 0	PA7OD to PA0OD	R/W	0 : CMOS 1 : Open-drain

8.2.1.7 PAPUP (Port A pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PA7UP	PA6UP	PA5UP	PA4UP	PA3UP	PA2UP	PA1UP	PA0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PA7UP to PA0UP	R/W	Pull-Up 0: Disable 1: Enable

8.2.1.8   PAIE (Port A input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PA7IE	PA6IE	PA5IE	PA4IE	PA3IE	PA2IE	PA1IE	PA0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PA7IE to PA0IE	R/W	Input 0: Disable 1: Enable

8.2.2   Port B (PB0 to PB7)

The port B is a general-purpose, 8-bit input / output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input / output function, the port B performs the external bus interface.

Reset initializes all bits of the port B as general-purpose ports with input, output and pull-up disabled.

The port B has a types of function register. If you use the port B as a general-purpose port, set "0" to the corresponding bit of the a registers. If you use the port B as other than a general-purpose port, set "1" to the corresponding bit of the function register.

8.2.2.1   Port B Circuit Type

	7	6	5	4	3	2	1	0
Type	T1	T1	T1	T1	T1	T1	T1	T1

8.2.2.2   Port B Register

Base Address = 0x400C_0100

Register name		Address (Base+)
Port B data register	PBDATA	0x0000
Port B output control register	PBCR	0x0004
Port B function register 1	PBFR1	0x0008
Port B open drain control register	PBOD	0x0028
Port B pull-up control register	PBPUP	0x002C
Port B input control register	PBIE	0x0038

8.2.2.3 PBDATA (Port B data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PB7 to PB0	R/W	Port B data register

8.2.2.4 PBCR (Port B output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PB7C	PB6C	PB5C	PB4C	PB3C	PB2C	PB1C	PB0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PB7C to PB0C	R/W	Output 0: Disable 1: Enable

8.2.2.5   PBFR1 (Port B function register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PB7F1	PB6F1	PB5F1	PB4F1	PB3F1	PB2F1	PB1F1	PB0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PB7F1	R/W	0 : PORT 1 : D15, AD15
6	PB6F1	R/W	0: PORT 1: D14, AD14
5	PB5F1	R/W	0: PORT 1: D13, AD13
4	PB4F1	R/W	0: PORT 1: D12, AD12
3	PB3F1	R/W	0: PORT 1: D11, AD11
2	PB2F1	R/W	0: PORT 1: D10, AD10
1	PB1F1	R/W	0: PORT 1: D9, AD9
0	PB0F1	R/W	0: PORT 1: D8, AD8

8.2.2.6   PBOD (Port B open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PB7OD	PB6OD	PB5OD	PB4OD	PB3OD	PB2OD	PB1OD	PB0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7-0	PB7OD to PB0OD	R/W	0 : CMOS 1 : Open-drain

8.2.2.7   PBPUP (Port B pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PB7UP	PB6UP	PB5UP	PB4UP	PB3UP	PB2UP	PB1UP	PB0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7-0	PB7UP to PB0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.2.8   PBIE (Port B input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PB7IE	PB6IE	PB5IE	PB4IE	PB3IE	PB2IE	PB1IE	PB0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PB7IE to PB0IE	R/W	Input 0: Disable 1: Enable

8.2.3 Port E (PE0 to PE7)

The port E is a general-purpose, 8-bit input / output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input / output function, the port E performs the external bus interface, the serial channel, external signal interrupt, the 16-bit timer / counter, CAN controller function and clock output function.

Reset initializes all bits of the port E as general-purpose ports with input, output and pull-up disabled.

The Port E has three types of of function register. If you use the port E as a general-purpose port, set "0" to the corresponding bit of the three registers. If you use the port E as other than a general-purpose port, set "1" to the corresponding bit of the function register. Do not set "1" to the some function registers at the same time.

Note: In modes other than STOP mode, interrupt input is enabled regardless of the PxFR register setting if input is enabled in PxIE. Make sure to disable unused interrupts when programming the device.

8.2.3.1 Port E Circuit Type

	7	6	5	4	3	2	1	0
Type	T6	T4	T37	T36	T3	T3	T3	T3

8.2.3.2 Port E register

Base Address = 0x400C_0400

Register name		Address (Base+)
Port E data register	PEDATA	0x0000
Port E output control register	PECR	0x0004
Port E function register 1	PEFR1	0x0008
Port E function register 2	PEFR2	0x000C
Port E function register 3	PEFR3	0x0010
Port E open drain control register	PEOD	0x0028
Port E pull-up control register	PEPUP	0x002C
Port E input control register	PEIE	0x0038

8.2.3.3   PEDATA (Port E data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PE7 to PE0	R/W	Port E data register

8.2.3.4   PECR (Port E output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PE7C	PE6C	PE5C	PE4C	PE3C	PE2C	PE1C	PE0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PE7C to PE0C	R/W	Output 0: Disable 1: Enable

8.2.3.5 PEFR1 (Port E function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	PE6F1	PE5F1	PE4F1	PE3F1	PE2F1	PE1F1	PE0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	−	R/W	Write "0".
6	PE6F1	R/W	0: PORT 1:A23
5	PE5F1	R/W	0: PORT 1: A22
4	PE4F1	R/W	0: PORT 1: A21
3	PE3F1	R/W	0: PORT 1: A20
2	PE2F1	R/W	0: PORT 1: A19
1	PE1F1	R/W	0: PORT 1: A18
0	PE0F1	R/W	0: PORT 1: A17

8.2.3.6　PEFR2 (Port E function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PE7F2	PE6F2	PE5F2	PE4F2	PE3F2	PE2F2	PE1F2	PE0F2
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PE7F2	R/W	0 : PORT 1 : INT5
6	PE6F2	R/W	0: PORT 1:SCLK0
5	PE5F2	R/W	0: PORT 1: RXD0
4	PE4F2	R/W	0: PORT 1: TXD0
3	PE3F2	R/W	0 : PORT 1 : TB6IN1
2	PE2F2	R/W	0: PORT 1: TB6IN0
1	PE1F2	R/W	0: PORT 1: TB5IN1
0	PE0F2	R/W	0: PORT 1: TB5IN0

8.2.3.7   PEFR3 (Port E function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PE7F3	PE6F3	PE5F3	PE4F3	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PE7F3	R/W	0 : PORT 1 : SCOUT
6	PE6F3	R/W	0 : PORT 1 : $\overline{CTS0}$
5	PE5F3	R/W	0 : PORT 1 : CRXD
4	PE4F3	R/W	0 : PORT 1 : CTXD
3 to 0	−	R	Read as "0".

8.2.3.8   PEOD (Port E open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PE7OD	PE6OD	PE5OD	PE4OD	PE3OD	PE2OD	PE1OD	PE0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PE7OD to PE0OD	R/W	0 : CMOS 1 : Open-drain

8.2.3.9   PEPUP (Port E pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PE7UP	PE6UP	PE5UP	PE4UP	PE3UP	PE2UP	PE1UP	PE0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PE7UP to PE0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.3.10   PEIE (Port E input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PE7IE	PE6IE	PE5IE	PE4IE	PE3IE	PE2IE	PE1IE	PE0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PE7IE to PE0IE	R/W	Intput 0: Disable 1: Enable

8.2.4   Port F (PF0 to PF4)

The port F is a general-purpose, 5-bit input / output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input / output function, the port F performs the debug interface function.

Reset initializes all bits of the port F as general-purpose ports with input, output and pull-up disabled.

The Port F has one types of of function register. If you use the port F as a general-purpose port, set "0" to the corresponding bit of the three registers. If you use the port F as other than a general-purpose port, set "1" to the corresponding bit of the function register.

8.2.4.1   Port F Circuit Type

	7	6	5	4	3	2	1	0
Type	–	–	–	T7	T7	T7	T7	T7

8.2.4.2   Port F Register

Base Address = 0x400C_0500

Register name		Address (Base+)
Port F data register	PFDATA	0x0000
Port F output control register	PFCR	0x0004
Port F function register 1	PFFR1	0x0008
Port F open drain control register	PFOD	0x0028
Port F pull-up control register	PFPUP	0x002C
Port F input control register	PFIE	0x0038

8.2.4.3　PFDATA (Port F data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PF4	PF3	PF2	PF1	PF0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	TypF	Function
31-5	−	R	Read as "0".
4-0	PF4 to PF0	R/W	Port F data register

8.2.4.4　PFCR (Port F output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PF4C	PF3C	PF2C	PF1C	PF0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4-0	PF4C to PF0C	R/W	Output 0: Disable 1: Enable

8.2.4.5 PFFR1 (Port F function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PF4F1	PF3F1	PF2F1	PF1F1	PF0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4	PF4F1	R/W	0: PORT 1: TRACEDATA3
3	PF3F1	R/W	0: PORT 1: TRACEDATA2
2	PF2F1	R/W	0: PORT 1: TRACEDATA1
1	PF1F1	R/W	0: PORT 1: TRACEDATA0 / SWV
0	PF0F1	R/W	0: PORT 1: TRACECLK

8.2.4.6   PFOD (Port F open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PF4OD	PF3OD	PF2OD	PF1OD	PF0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4-0	PF4OD to PF0OD	R/W	0 : CMOS 1 : Open-drain

8.2.4.7   PFPUP (Port F pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PF4UP	PF3UP	PF2UP	PF1UP	PF0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4-0	PF4UP to PF0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.4.8 PFIE (Port F input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PF4IE	PF3IE	PF2IE	PF1IE	PF0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4-0	PF4IE to PF0IE	R/W	Input 0: Disable 1: Enable

8.2.5.3 PGDATA (Port G data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7-0	PG7 to PG0	R/W	Port G data register

8.2.5.4 PGCR (Port G output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PG7C	PG6C	PG5C	PG4C	PG3C	PG2C	PG1C	PG0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7-0	PG7C to PG0C	R/W	Output 0: Disable 1: Enable

8.2.5.5   PGFR1 (Port G function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PG7F1	PG6F1	PG5F1	PG4F1	PG3F1	PG2F1	PG1F1	PG0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PG7F1	R/W	0: PORT 1:INT7
6	PG6F1	R/W	0: PORT 1: SCK2
5	PG5F1	R/W	0: PORT 1: SCL2 / SI2
4	PG4F1	R/W	0: PORT 1: SDA2 / SO2
3	PG3F1	R/W	0: PORT 1: INT6
2	PG2F1	R/W	0: PORT 1: SCK1
1	PG1F1	R/W	0: PORT 1: SCL1 / SI1
0	PG0F1	R/W	0: PORT 1: SDA1 / SO1

8.2.5.6 PGFR2 (Port G function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PG7F2	PG6F2	PG5F2	PG4F2	-	-	PG1F2	PG0F2
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PG7F2	R/W	0 : PORT 1 : $\overline{USBOC}$
6	PG6F2	R/W	0 : PORT 1 : USBPON
5	PG5F2	R/W	0: PORT 1: TB9IN1
4	PG4F2	R/W	0: PORT 1: TB9IN0
3-2	−	R	Read as "0".
1	PG1F2	R/W	0: PORT 1: TB7IN1
0	PG0F2	R/W	0: PORT 1: TB7IN0

8.2.5.7　PGFR3 (Port G function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PG7F3	PG6F3	-	-	PG3F3	PG2F3	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PG7F3	R/W	0 : PORT 1 : $\overline{\text{WDTOUT}}$
6	PG6F3	R/W	0 : PORT 1 : $\overline{\text{CS3}}$
5-4	−	R	Read as "0".
3	PG3F3	R/W	0 : PORT 1 : $\overline{\text{CS1}}$
2	PG2F3	R/W	0 : PORT 1 : $\overline{\text{CS0}}$
1 to 0	−	R	Read as "0".

8.2.5.8   PGOD (Port G open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PG7OD	PG6OD	PG5OD	PG4OD	PG3OD	PG2OD	PG1OD	PG0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PG7OD to PG0OD	R/W	0 : CMOS 1 : Open-drain

8.2.5.9   PGPUP (Port G pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PG7UP	PG6UP	PG5UP	PG4UP	PG3UP	PG2UP	PG1UP	PG0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PG7UP to PG0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.5.10 PGIE (Port G input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PG7IE	PG6IE	PG5IE	PG4IE	PG3IE	PG2IE	PG1IE	PG0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PG7IE to PG0IE	R/W	Input 0: Disable 1: Enable

8.2.6   Port I (PI0 to PI1)

The port I is a general-purpose, 2-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose port function, the port I performs CEC input and the operation mode setting.

Reset initializes PI0 as general-purpose ports with input, output and pull-up disabled. Reset initializes PI1, PI2 and PI3 as general-purpose ports with input, output disabled.

PI1 is N-ch open-drain ports.

The Port I has one types of of function register. If you use the port I as a general-purpose port, set "0" to the corresponding bit of the three registers. If you use the port I as other than a general-purpose port, set "1" to the corresponding bit of the function register.

While $\overline{RESET}$ pin is "Low", input and pull-up of PI0($\overline{BOOT}$) are enabled. At the rising edge of $\overline{RESET}$ pin, if PI0($\overline{BOOT}$) is "High", TMPM363F10FG enters the single chip mode and boots from the on chip Flash ROM. If PI0($\overline{BOOT}$) is "Low", TMPM363F10FG enters the single boot mode and boots from the on chip BOOTROM. For details of the single boot mode, refer to "Flash memory operation".

8.2.6.1   Port I Circuit Type

	7	6	5	4	3	2	1	0
Type	–	–	–	–	–	–	T15	T14

8.2.6.2   Port I register

Base Address = 0x400C_0800

Register name		Address (Base+)
Port I data register	PIDATA	0x0000
Port I output control register	PICR	0x0004
Port I function register 1	PIFR1	0x0008
Port I open drain control register	PIOD	0x0028
Port I pull-up control register	PIPUP	0x002C
Port I input control register	PIIE	0x0038

8.2.6.3   PIDATA (Port I data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	PI1	PI0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-2	−	R/W	Write "0".
1-0	PI1 to PI0	R/W	Port I data register

8.2.6.4 PICR (Port I output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	PI1C	PI0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	R	Read as "0".
3-2	–	R/W	Write "0".
1-0	PI1C-PI0C	R/W	Output 0: Disable 1: Enable

8.2.6.5   PIFR1 (Port I function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	PI1F1	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-2	−	R/W	Write "0".
1	PI1F1	R/W	0: PORT 1: CEC
0	−	R/W	Write "0".

8.2.6.6 PIOD (Port I open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	PI0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-2	−	R/W	Write "0".
1	−	R	Read as "0".
0	PI0OD	R/W	0 : CMOS 1 : Open-drain

8.2.6.7 PIPUP (Port I pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	PI0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	−	R	Read as "0".
0	PI0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.6.8   PIIE (Port I input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	PI1IE	PI0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-2	−	R/W	Write "0".
1-0	PI1IE-PI0IE	R/W	Input 0: Disable 1: Enable

8.2.7 Port J (PJ0 to PJ7)

The port J is an 8-bit input port. Besides the general-purpose input function, the port J performs the AD converter and the key-on wake-up function.

Reset initializes all bits of the port J as general-purpose ports with input and pull-up disabled.

The Port J has one types of of function register. If you use the port J as a general-purpose port, set "0" to the corresponding bit of the three registers. If you use the port J as other than a general-purpose port, set "1" to the corresponding bit of the function register.

Note: Unless you use all the bits of port J as analog input pins, conversion accurary may be reduced.Be sure to verify that this causes no problem on your system.

8.2.7.1 Port J Circuit Type

	7	6	5	4	3	2	1	0
Type	T19	T19	T19	T19	T18	T17	T17	T17

8.2.7.2 Port J register

Base Address = 0x400C_0900

Register name		Address (Base+)
Port J data register	PJDATA	0x0000
Port J function register 2	PJFR2	0x000C
Port J pull-up control register	PJPUP	0x002C
Port J input control register	PJIE	0x0038

8.2.7.3   PJDATA (Port J data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
After reset	1	1	1	1	1	1	1	1

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PJ7 to PJ0	R	Port J data register

8.2.7.4   PJFR2 (Port J function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PJ7F2	PJ6F2	PJ5F2	PJ4F2	PJ3F2	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PJ7F2	R/W	0: PORT 1: KWUP3
6	PJ6F2	R/W	0: PORT 1: KWUP2
5	PJ5F2	R/W	0: PORT 1: KWUP1
4	PJ4F2	R/W	0: PORT 1: KWUP0
3	PJ3F2	R/W	0: PORT 1: $\overline{\text{ADTRG}}$
2-0	−	R	Read as "0".

8.2.7.5   PJPUP (Port J pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PJ7UP	PJ6UP	PJ5UP	PJ4UP	PJ3UP	PJ2UP	PJ1UP	PJ0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PJ7UP to PJ0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.7.6   PJIE (Port J input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PJ7IE	PJ6IE	PJ5IE	PJ4IE	PJ3IE	PJ2IE	PJ1IE	PJ0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PJ7IE to PJ0IE	R/W	Input 0: Disable 1: Enable

8.2.8 Port L (PL0 to PL7)

The port L is a general-purpose, 8-bit input / output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input / output function, the port L performs the serial channel , the serial bus interface, the external signal interrupt input and 16bit timer / event counter function.

Reset initializes all bits of the port L as general-purpose ports with input, output and pull-up disabled.

The Port L has three types of of function register. If you use the port L as a general-purpose port, set "0" to the corresponding bit of the three registers. If you use the port L as other than a general-purpose port, set "1" to the corresponding bit of the function register. Do not set "1" to the some function registers at the same time.

Note:: In modes other than STOP mode, interrupt input is enabled regardless of the PxFR register setting
if input is enabled in PxIE. Make sure to disable unused interrupts when programming the device.

8.2.8.1 Port L Circut Type

	7	6	5	4	3	2	1	0
Type	T21	T24	T23	T22	T21	T20	T20	T20

8.2.8.2 Port L register

Base Address = 0x400C_0B00

Register name		Address (Base+)
Port L data register	PLDATA	0x0000
Port L output control register	PLCR	0x0004
Port L function register 1	PLFR1	0x0008
Port L function register 2	PLFR2	0x000C
Port L function register 3	PLFR3	0x0010
Port L open drain control register	PLOD	0x0028
Port L pull-up control register	PLPUP	0x002C
Port L input control register	PLIE	0x0038

8.2.8.3 PLDATA (Port L data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PL7 to PL0	R/W	Port L data register

8.2.8.4 PLCR (Port L output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PL7C	PL6C	PL5C	PL4C	PL3C	PL2C	PL1C	PL0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PL7C to PL0C	R/W	Output 0: Disable 1: Enable

8.2.8.5 PLFR1 (Port K function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PL7F1	PL6F1	PL5F1	PL4F1	PL3F1	PL2F1	PL1F1	PL0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PL7F1	R/W	0: PORT 1:INT1
6	PL6F1	R/W	0: PORT 1: SCLK1
5	PL5F1	R/W	0: PORT 1: RXD1
4	PL4F1	R/W	0: PORT 1: TXD1
3	PL3F1	R/W	0: PORT 1: INT0
2	PL2F1	R/W	0: PORT 1: SCK0
1	PL1F1	R/W	0: PORT 1: SCL0 / SI0
0	PL0F1	R/W	0: PORT 1: SDA0 / SO0

8.2.8.6 PLFR2 (Port L function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PL7F2	PL6F2	PL5F2	PL4F2	PL3F2	PL2F2	PL1F2	PL0F2
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PL7F2	R/W	0: PORT 1: TB7OUT
6	PL6F2	R/W	0: PORT 1: TB6OUT
5	PL5F2	R/W	0: PORT 1: TB5OUT
4	PL4F2	R/W	0: PORT 1: TB4OUT
3	PL3F2	R/W	0: PORT 1: TB3OUT
2	PL2F2	R/W	0: PORT 1: TB2OUT
1	PL1F2	R/W	0: PORT 1: TB1OUT
0	PL0F2	R/W	0: PORT 1: TB0OUT

8.2.8.7   PLFR3 (Port L function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	PL6F3	PL5F3	PL4F3	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6	PL6F3	R/W	0 : PORT 1 : $\overline{CTS1}$
5	PL5F3	R/W	0 : PORT 1 : SCL3
4	PL4F3	R/W	0 : PORT 1 : SDA3
3-0	−	R	Read as "0".

8.2.8.8   PLOD (Port L open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PL7OD	PL6OD	PL5OD	PL4OD	PL3OD	PL2OD	PL1OD	PL0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PL7OD to PL0OD	R/W	0 : CMOS 1 : Open-drain

8.2.8.9   PLPUP (Port L pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PL7UP	PL6UP	PL5UP	PL4UP	PL3UP	PL2UP	PL1UP	PL0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PL7UP to PL0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.8.10   PLIE (Port L input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PL7IE	PL6IE	PL5IE	PL4IE	PL3IE	PL2IE	PL1IE	PL0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PL7IE to PL0IE	R/W	Input 0: Disable 1: Enable

8.2.9   Port M (PM0 to PM7)

The port M is a general-purpose, 8-bit input / output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input / output function, the port M performs the serial channel, the external signal interrupt input,16bit timer / event counter and the alarm output function.

Reset initializes all bits of the port M as general-purpose ports with input, output and pull-up disabled.

The Port M has three types of of function register. If you use the port M as a general-purpose port, set "0" to the corresponding bit of the three registers. If you use the port M as other than a general-purpose port, set "1" to the corresponding bit of the function register. Do not set "1" to the some function registers at the same time.

Note: In modes other than STOP mode, interrupt input is enabled regardless of the PxFR register setting if
      input is enabled in PxIE. Make sure to disable unused interrupts when programming the device.

8.2.9.1   Port Circuit Type

	7	6	5	4	3	2	1	0
Type	T30	T29	T28	T27	T21	T23	T26	T25

8.2.9.2   Port M register

Base Address = 0x400C_0C00

Register name		Address (Base+)
Port M data register	PMDATA	0x0000
Port M output control register	PMCR	0x0004
Port M function register 1	PMFR1	0x0008
Port M function register 2	PMFR2	0x000C
Port M function register 3	PMFR3	0x0010
Port M open drain control registert	PMOD	0x0028
Port M pull-up control register	PMPUP	0x002C
Port M input control register	PMIE	0x0038

8.2.9.3   PMDATA (Port M data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PM7	PM6	PM5	PM4	PM3	PM2	PM1	PM0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PM7 to PM0	R/W	Port M data register

8.2.9.4   PMCR (Port M output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PM7C	PM6C	PM5C	PM4C	PM3C	PM2C	PM1C	PM0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PM7C to PM0C	R/W	Output 0: Disable 1: Enable

8.2.9.5 PMFR1 (Port M function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PM7F1	PM6F1	PM5F1	PM4F1	PM3F1	PM2F1	PM1F1	PM0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PM7F1	R/W	0: PORT 1: INT3
6	PM6F1	R/W	0: PORT 1: RXD3
5	PM5F1	R/W	0: PORT 1: TXD3
4	PM4F1	R/W	0: PORT 1: SCLK3
3	PM3F1	R/W	0: PORT 1: INT2
2	PM2F1	R/W	0: PORT 1: RXD2
1	PM1F1	R/W	0: PORT 1: TXD2
0	PM0F1	R/W	0: PORT 1: SCLK2

8.2.9.6   PMFR2 (Port M function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PM3F2	PM2F2	PM1F2	PM0F2
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3	PM3F2	R/W	0: PORT 1: TB3OUT
2	PM2F2	R/W	0: PORT 1: $\overline{\text{ALARM}}$
1	PM1F2	R/W	0: PORT 1: TB1IN1
0	PM0F2	R/W	0: PORT 1: TB1IN0

8.2.9.7 PMFR3 (Port M function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PM4F3	-	-	-	PM0F3
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4	PM4F3	R/W	0 : PORT 1 : $\overline{CTS3}$
3-1	−	R	Read as "0".
0	PM0F3	R/W	0 : PORT 1 : $\overline{CTS2}$

8.2.9.8 PMOD (Port M open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PM7OD	PM6OD	PM5OD	PM4OD	PM3OD	PM2OD	PM1OD	PM0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PM7OD to PM0OD	R/W	0 : CMOS 1 : Open-drain

8.2.9.9   PMPUP (Port M pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PM7UP	PM6UP	PM5UP	PM4UP	PM3UP	PM2UP	PM1UP	PM0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PM7UP to PM0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.9.10   PMIE (Port M input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PM7IE	PM6IE	PM5IE	PM4IE	PM3IE	PM2IE	PM1IE	PM0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PM7IE to PM0IE	R/W	Input 0: Disable 1: Enable

8.2.10   Port N (PN0 to PN3)

The port N is a general-purpose, 4-bit input / output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input / output function, the port N performs the serial channel, the external signal interrupt input, 16bit timer / counter and the remote control signal preprocessor input function.

Reset initializes all bits of the port N as general-purpose ports with input, output and pull-up disabled.

The Port N has three types of of function register. If you use the port N as a general-purpose port, set "0" to the corresponding bit of the three registers. If you use the port N as other than a general-purpose port, set "1" to the corresponding bit of the function register. Do not set "1" to the some function registers at the same time.

Note: In modes other than STOP mode, interrupt input is enabled regardless of the PxFR register setting if input is enabled in PxIE. Make sure to disable unused interrupts when programming the device.

8.2.10.1   Port N Circuit Type

	7	6	5	4	3	2	1	0
Type	–	–	–	–	T30	T25	T29	T28

8.2.10.2   Port N register

Base Address = 0x400C_0D00

Register name		Address (Base+)
Port N data register	PNDATA	0x0000
Port N output control register	PNCR	0x0004
Port N function register 1	PNFR1	0x0008
Port N function register 2	PNFR2	0x000C
Port N function register 3	PNFR3	0x0010
Port N open drain control registert	PNOD	0x0028
Port N pull-up control register	PNPUP	0x002C
Port N input control register	PNIE	0x0038

8.2.10.3   PNDATA (Port N data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PN3	PN2	PN1	PN0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-4	−	R/W	Write "0".
3-0	PN3 to PN0	R/W	Port N data register

8.2.10.4   PNCR (Port N output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PN3C	PN2C	PN1C	PN0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-4	−	R/W	Write "0".
3-0	PN3C to PN0C	R/W	Output 0: Disable 1: Enable

8.2.10.5 PNFR1 (Port N function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PN3F1	PN2F1	PN1F1	PN0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-4	−	R/W	Write "0".
3	PN3F1	R/W	0: PORT 1: INT4
2	PN2F1	R/W	0: PORT 1: SCLK4
1	PN1F1	R/W	0: PORT 1: RXD4
0	PN0F1	R/W	0: PORT 1: TXD4

8.2.10.6   PNFR2 (Port N function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PN3F2	PN2F2	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-6	−	R/W	Write "0".
5-4	−	R	Read as "0".
3	PN3F2	R/W	0: PORT 1: TB2IN1
2	PN2F2	R/W	0: PORT 1: TB2IN0
1-0	−	R	Read as "0".

8.2.10.7 PNFR3 (Port N function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PN3F3	PN2F3	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-6	−	R/W	Write "0".
5-4	−	R	Read as "0".
3	PN3F3	R/W	0 : PORT 1 : RMC
2	PN2F3	R/W	0 : PORT 1 : $\overline{CTS4}$
1 to 0	−	R	Read as "0".

8.2.10.8 PNOD (Port N open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PN3OD	PN2OD	PN1OD	PN0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-4	−	R/W	Write "0".
3-0	PN3OD to PN0OD	R/W	0 : CMOS 1 : Open-drain

8.2.10.9 PNPUP (Port N pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PN3UP	PN2UP	PN1UP	PN0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-4	−	R/W	Write "0".
3-0	PN3UP to PN0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.10.10   PNIE (Port N input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PN3IE	PN2IE	PN1IE	PN0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-4	−	R/W	Write "0".
3-0	PN3IE to PN0IE	R/W	Input 0: Disable 1: Enable

8.2.11   Port P (PP0 to PP6)

The port P is a general-purpose, 7-bit input / output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input / output function, the port P performs the external bus interface, and SSP function.

Reset initializes all bits of the port P as general-purpose ports with input, output and pull-up disabled.

The Port P has two types of of function register. If you use the port P as a general-purpose port, set "0" to the corresponding bit of the two registers. If you use the port P as other than a general-purpose port, set "1" to the corresponding bit of the function register. Do not set "1" to the both function registers at the same time.

8.2.11.1   Port P Circuit Type

	7	6	5	4	3	2	1	0
Type	–	T5	T35	T34	T33	T32	T31	T5

8.2.11.2   Port P register

Base Address = 0x400C_0F00

Register name		Address (Base+)
Port P data register	PPDATA	0x0000
Port P output control register	PPCR	0x0004
Port P function register 1	PPFR1	0x0008
Port P function register 2	PPFR2	0x000C
Port P open drain control register	PPOD	0x0028
Port P pull-up control register	PPPUP	0x002C
Port P input control register	PPIE	0x0038

8.2.11.3   PPDATA (Port P data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	PP6	PP5	PP4	PP3	PP2	PP1	PP0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6-0	PP6 to PP0	R/W	Port P data register

8.2.11.4   PPCR (Port P output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	PP6C	PP5C	PP4C	PP3C	PP2C	PP1C	PP0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6-0	PP6C to PP0C	R/W	Output 0: Disable 1: Enable

8.2.11.5 PPFR1 (Port P function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	PP6F1	PP5F1	PP4F1	PP3F1	PP2F1	PP1F1	PP0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6	PP6F1	R/W	0: PORT 1: $\overline{\text{ALE}}$
5	PP5F1	R/W	0: PORT 1: $\overline{\text{OE}}$
4	PP4F1	R/W	0: PORT 1: $\overline{\text{WE}}$
3	PP3F1	R/W	0: PORT 1: $\overline{\text{BLS1}}$
2	PP2F1	R/W	0: PORT 1: $\overline{\text{BLS0}}$
1	PP1F1	R/W	Write "0".
0	PP0F1	R/W	0: PORT 1: $\overline{\text{CS2}}$

8.2.11.6 PPFR2 (Port P function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	PP5F2	PP4F2	PP3F2	PP2F2	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	–	R	Read as "0".
5	PP5F2	R/W	0: PORT 1: SPFSS
4	PP4F2	R/W	0: PORT 1: SPCLK
3	PP3F2	R/W	0: PORT 1: SPDI
2	PP2F2	R/W	0: PORT 1: SPDO
1-0	–	R	Read as "0".

8.2.11.7 PPOD (Port P open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	PP6OD	PP5OD	PP4OD	PP3OD	PP2OD	PP1OD	PP0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6-0	PP6OD to PP0OD	R/W	0 : CMOS 1 : Open-drain

8.2.11.8 PPPUP (Port P pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	PP6UP	PP5UP	PP4UP	PP3UP	PP2UP	PP1UP	PP0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6-0	PP6UP to PP0UP	R/W	Pull-up 0: Disable 1: Enable

8.2.11.9   PPIE (Port P input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	PP6IE	PP5IE	PP4IE	PP3IE	PP2IE	PP1IE	PP0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6-0	PP6IE to PP0IE	R/W	Input 0: Disable 1: Enable

8.3   Block Diagrams of Ports

8.3.1   Port Types

The ports are classified as shown below. Please refer to the following pages for the block diagrams of each port type.

Dot lines in the figure indicate the part of the equivalent circuit described in the "Block diagrams of ports".

Table 8-3 Function lists

Type	Input/ Output	Function1	Function2	Function3	Analog	Pull-up	Program-mable open-drain	Note
T1	I/O	I/O	–	–	–	R	o	
T2	I/O	Output	Output	–	–	R	o	
T3	I/O	Output	Input	–	–	R	o	
T4	I/O	Output	I/O	Input	–	R	o	
T5	I/O	Output	–	–	–	R	o	
T6	I/O	Input (int)	Output	–	–	R	o	
T7	I/O	Output	–	–	–	R	o	
T8	I/O	I/O	Input	–	–	R	o	
T9	I/O	I/O	–	Input	–	R	o	
T10	I/O	Input (int)	–	Input	–	R	o	
T11	I/O	Input (int)	–	Output	–	R	o	
T12	I/O	I/O	Input	–	–	R	o	
T13	I/O	Input (int)	Input	–	–	R	o	
T14	I/O	–	–	–	–	R	o	$\overline{BOOT}$ input enabled during reset
T15	I/O	Input	–	–	–	–	–	Nch open drain port
T16	I/O	Input (int)	–	–	–	NoR	o	
T17	Input	–	–	–	o	R	–	
T18	Input	Input	–	–	o	R	–	
T19	Input	Input	–	–	o	R	–	
T20	I/O	I/O	Output	–	–	R	o	
T21	I/O	Input	Output	–	–	R	o	
T22	I/O	Output	Output	–	–	R	o	
T23	I/O	Input	Output	–	–	R	o	
T24	I/O	I/O	Output	Input	–	R	o	
T25	I/O	I/O	Input	Input	–	R	o	
T26	I/O	Output	Input	–	–	R	o	
T27	I/O	I/O	–	Output	–	R	o	
T28	I/O	Output	–	–	–	R	o	
T29	I/O	Input	–	–	–	R	o	
T30	I/O	Input (int)	–	–	–	R	o	
T31	I/O	–	–	–	–	R	o	
T32	I/O	Output	Output	–	–	R	o	
T33	I/O	Output	Input	–	–	R	o	
T34	I/O	Output	I/O	–	–	R	o	
T35	I/O	Output	–	–	–	R	o	

int : Interrupt input   R : Forced disable during reset

- : Not exist   NoR : Unaffected by reset

o : exist

Table 8-3 Function lists

Type	Input/Output	Function1	Function2	Function3	Analog	Pull-up	Programmable open-drain	Note
T36	I/O	Output	Output	Output	–	R	o	
T37	I/O	Output	Input	Input	–	R	o	
T38	I/O	I/O	Output	Output	–	R	o	
T39	I/O	Input	Input	Output	–	R	o	

int : Interrupt input   R : Forced disable during reset

- : Not exist   NoR : Unaffected by reset
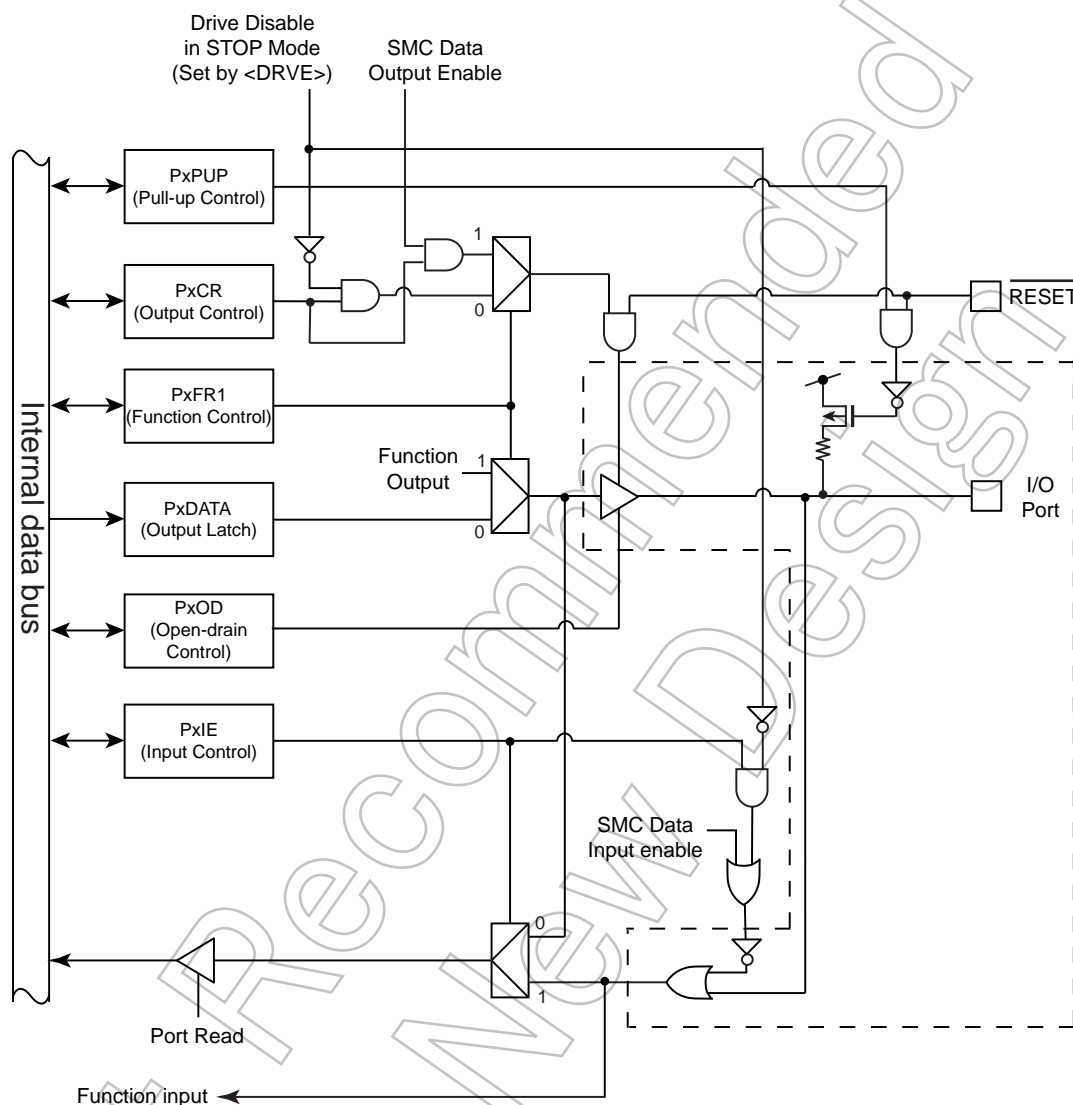
o : exist

8.3.2   Type T1

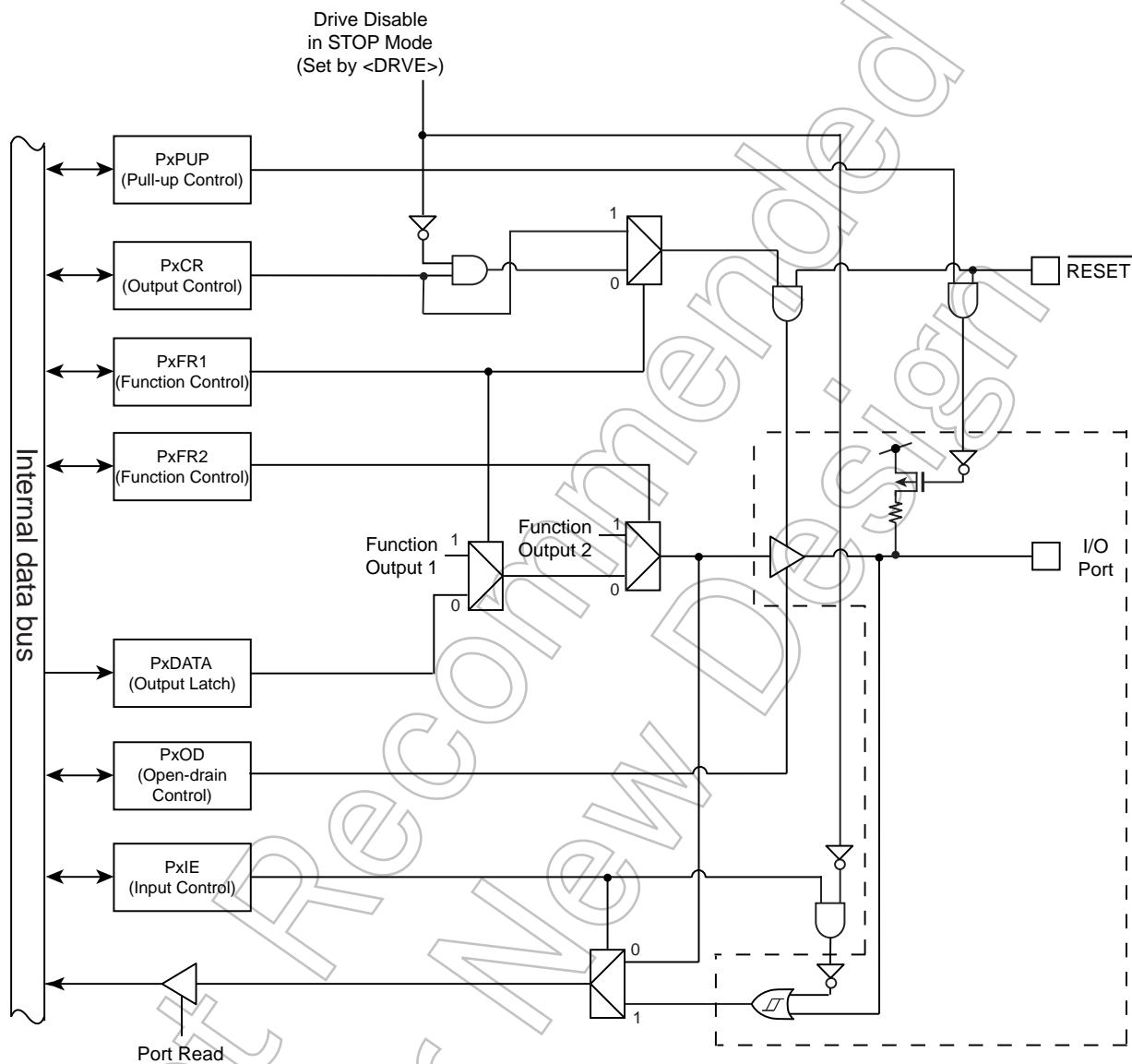

Figure 8-1 Port Type T1

8.3.3 Type T2
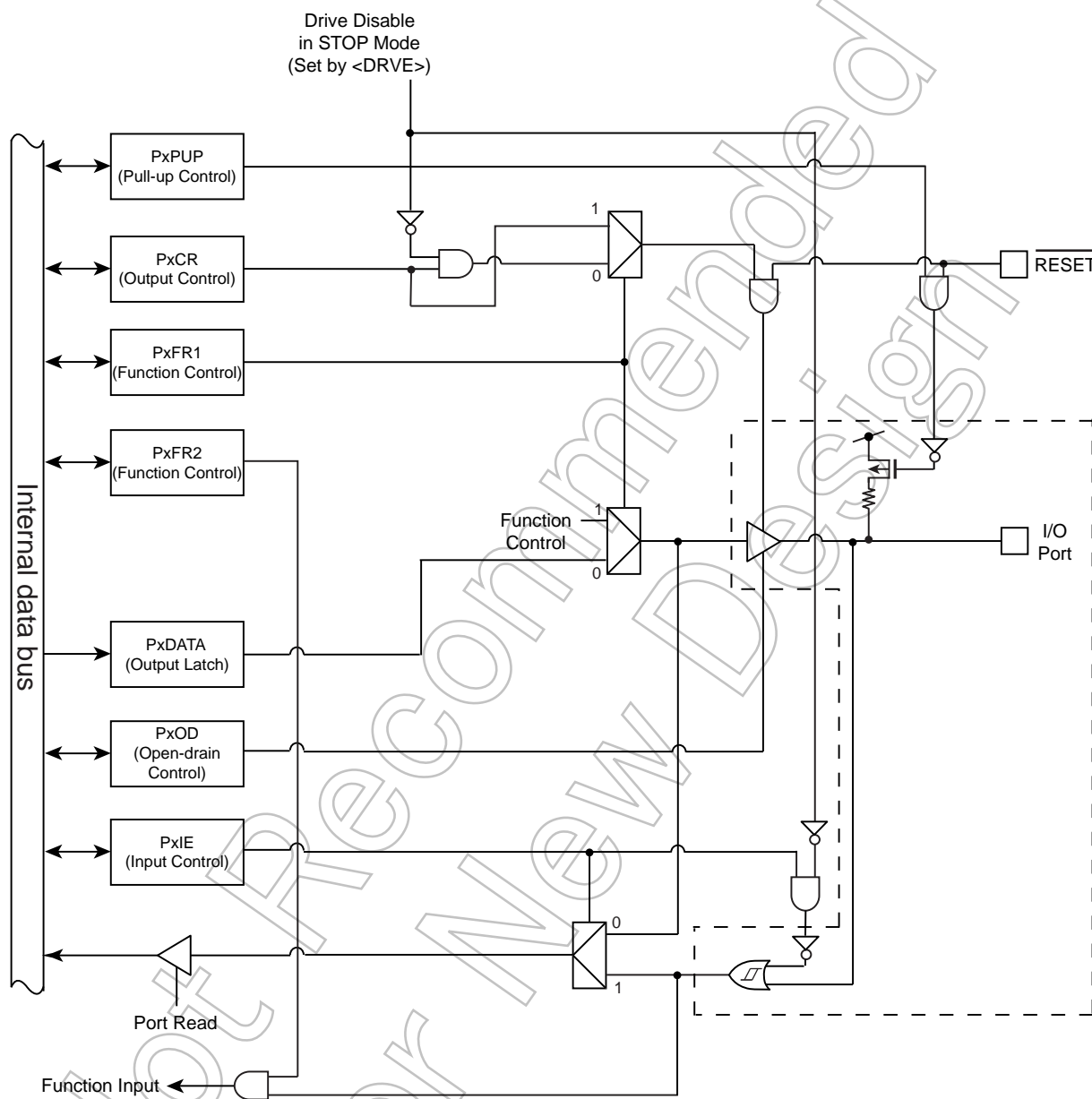

Figure 8-2 Port Type T2

8.3.4 Type T3

Figure 8-3 Port Type T3

8.3.5 Type T4

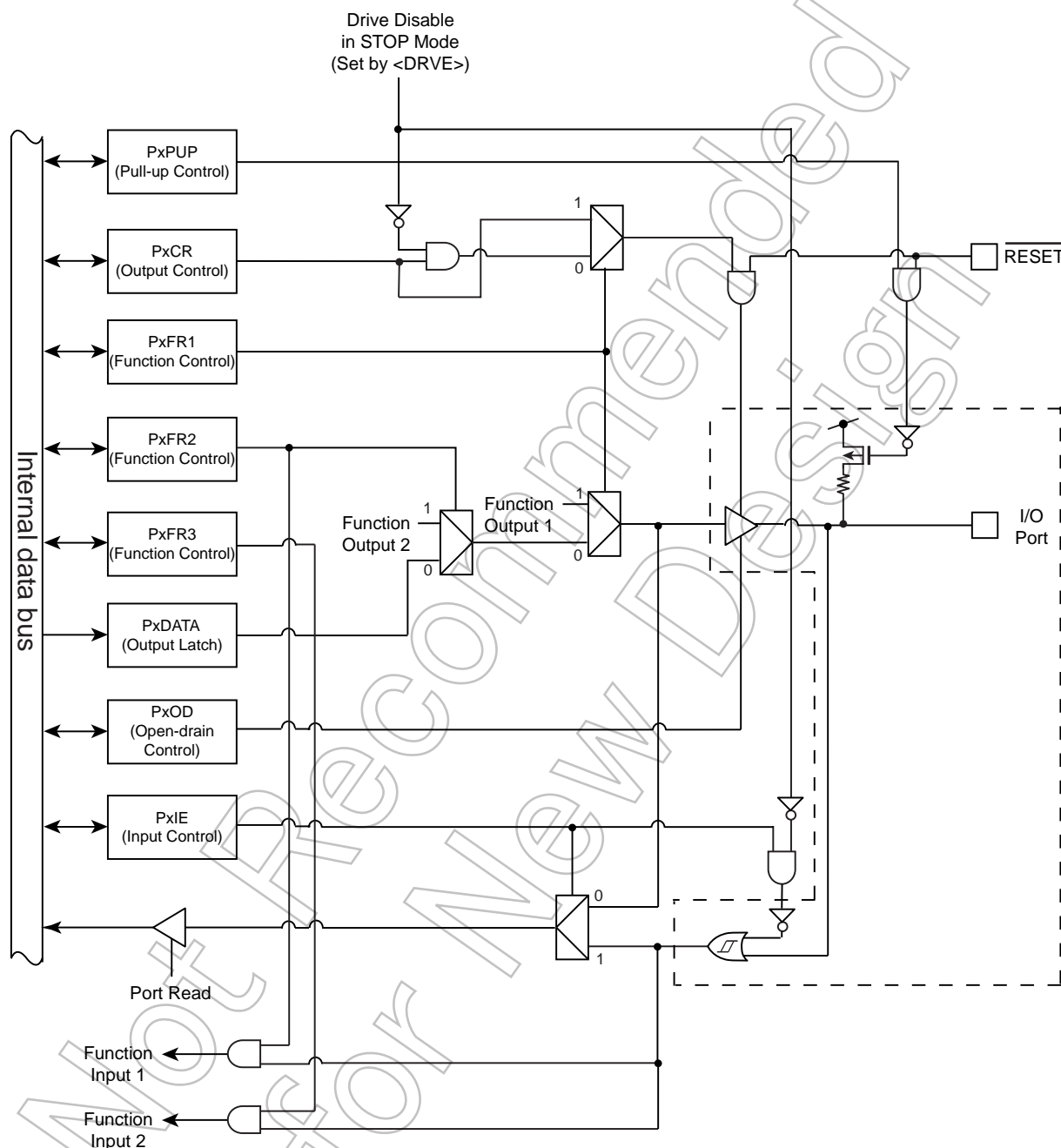Figure 8-4 Port Type T4

8.3.6   Type T5
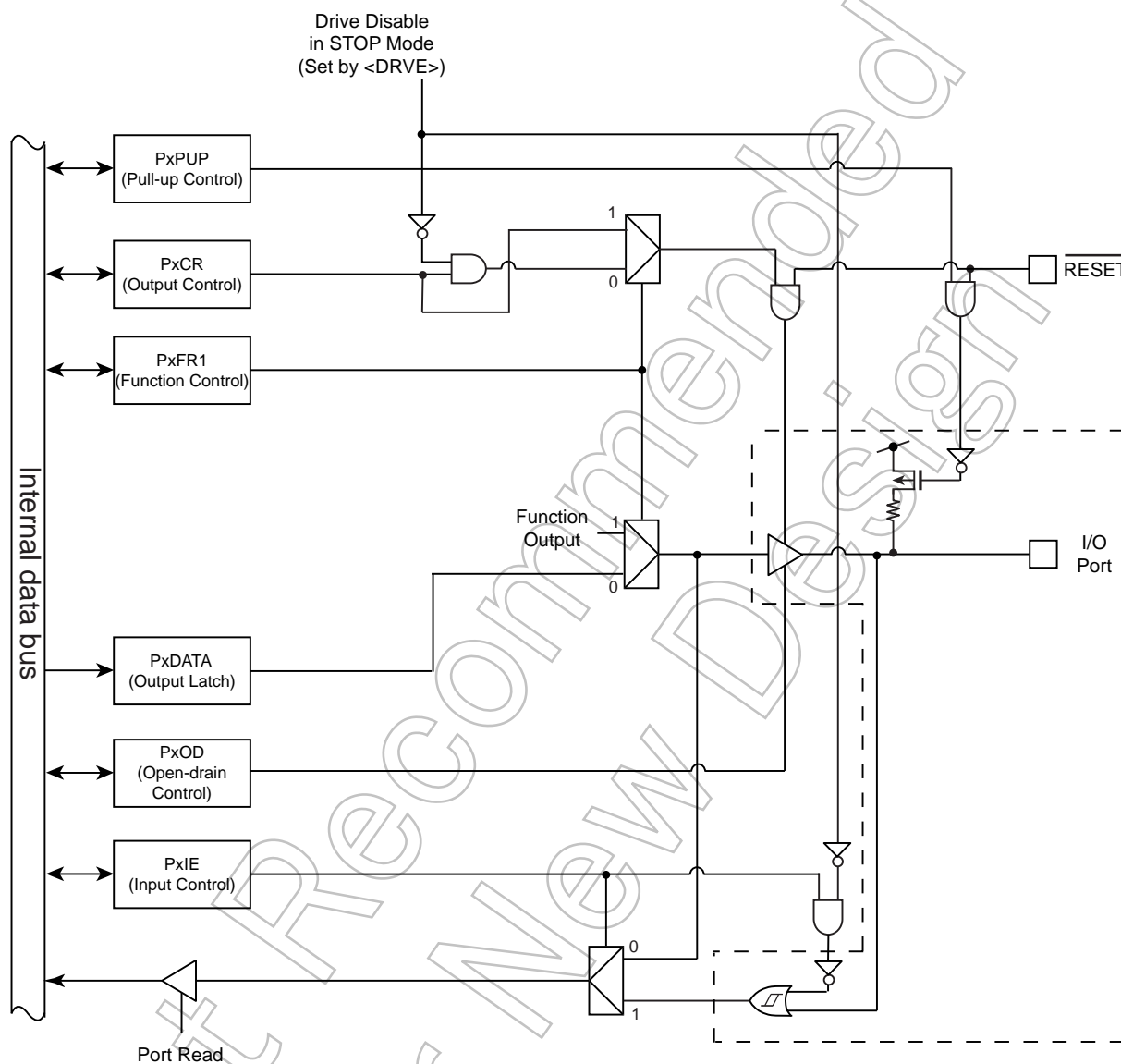

Figure 8-5 Port Type T5

8.3.7 Type T6

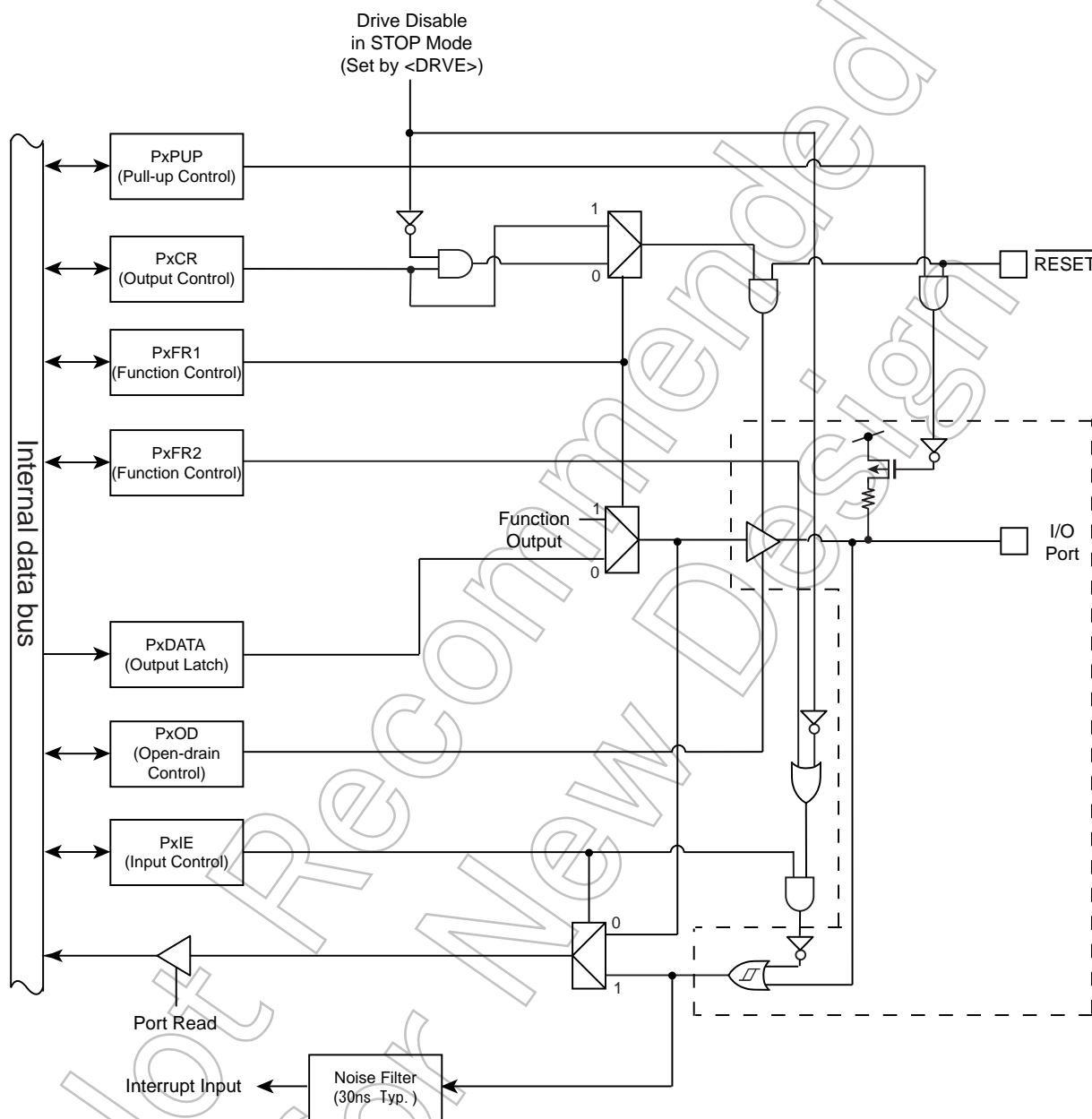
Figure 8-6 Port Type T6

8.3.8   Type T7

Figure 8-7 Port Type T7

8.3.9 Type T8
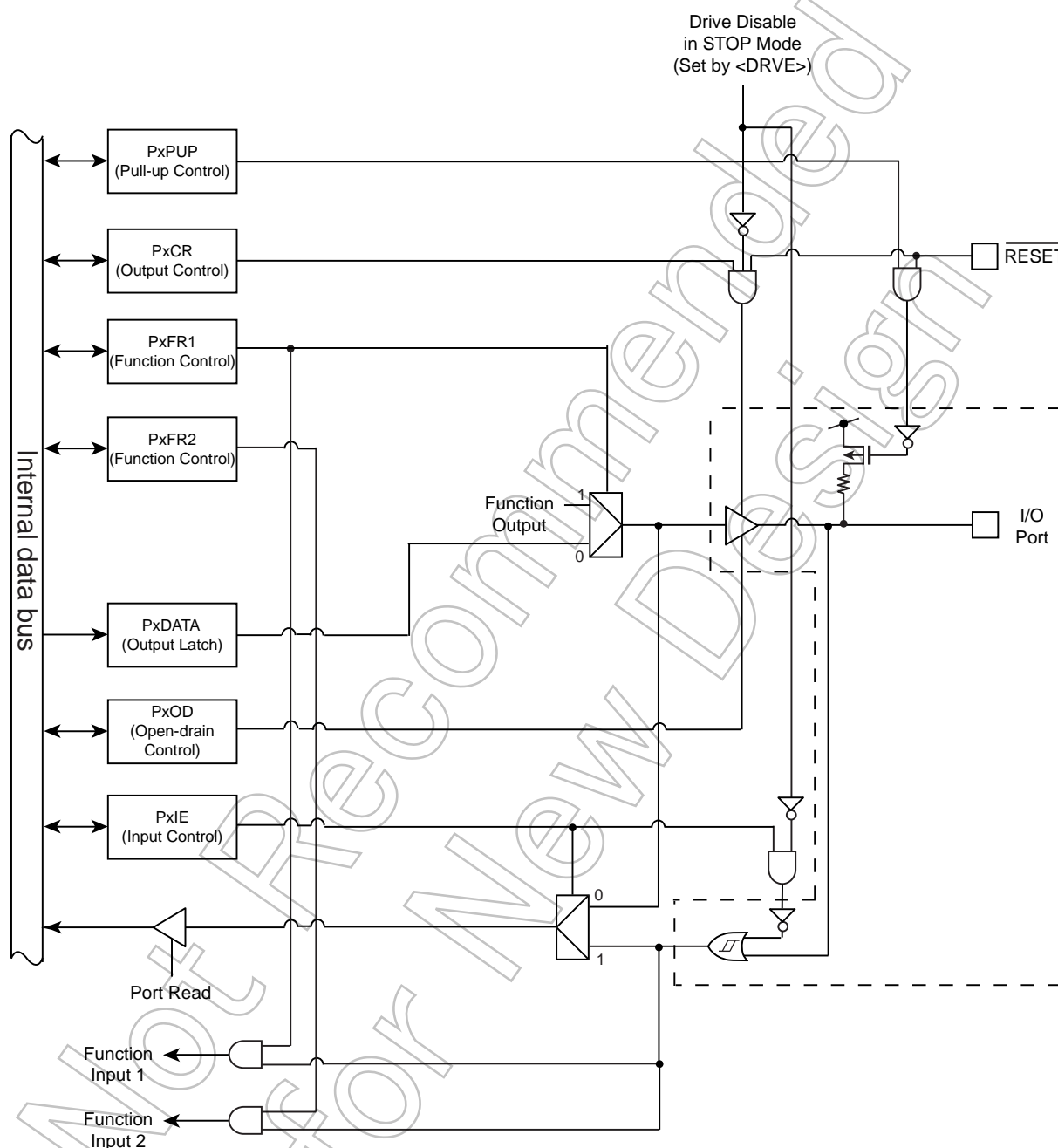
Figure 8-8 Port Type T8

8.3.10   Type T9
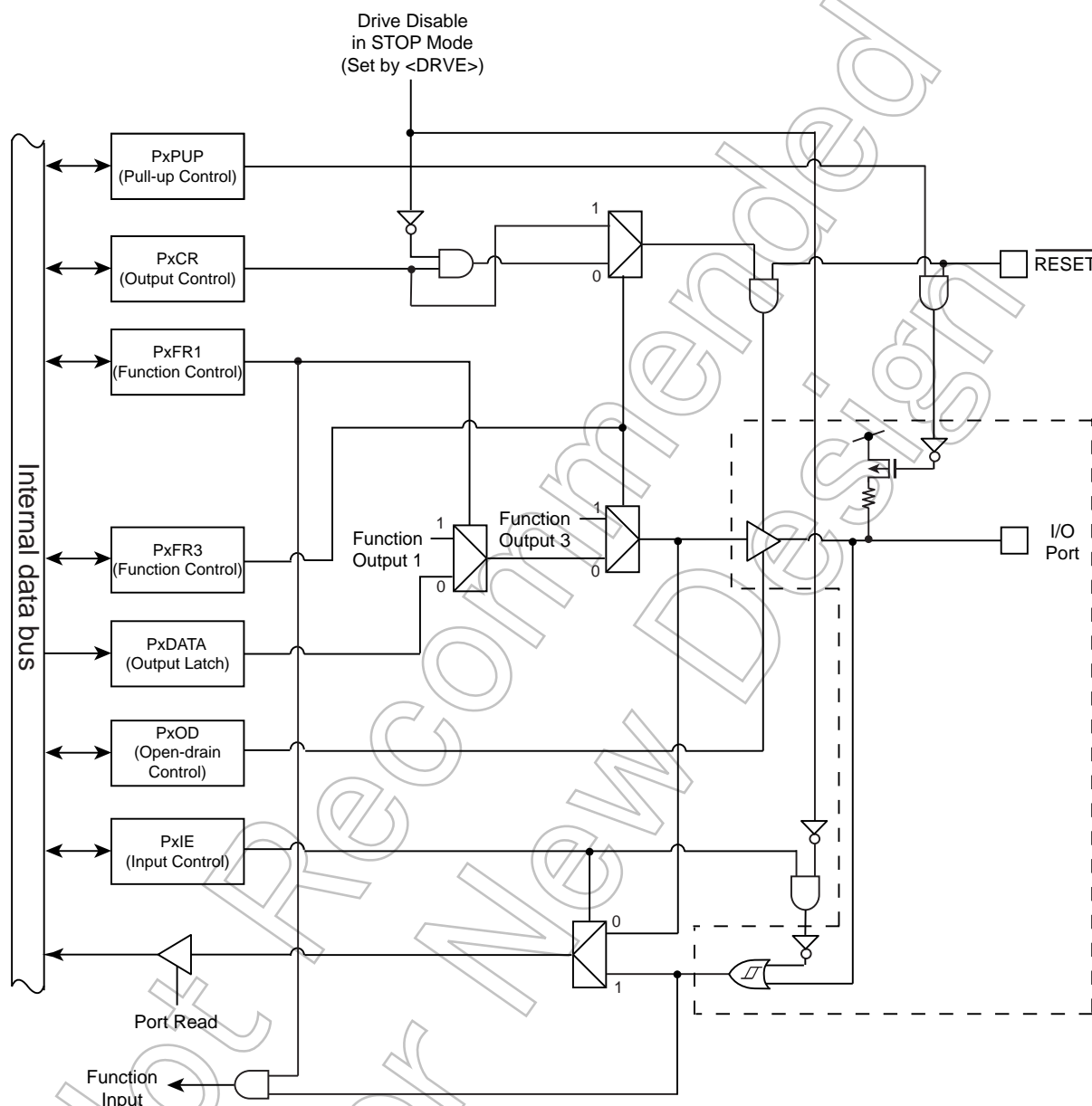
Figure 8-9 Port Type T9

8.3.11   Type T10
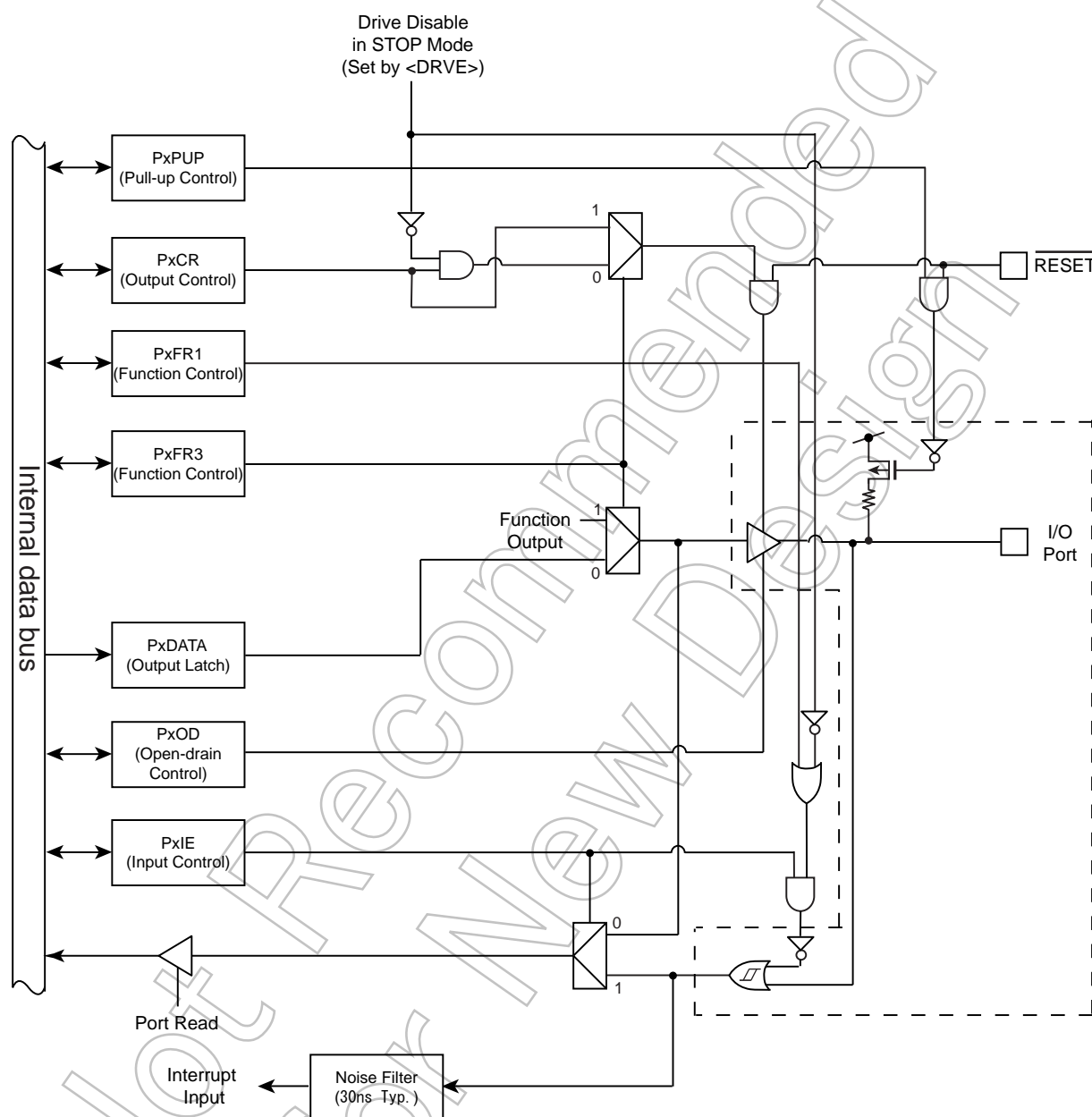
Figure 8-10 Port Type T10

8.3.12   Type T11
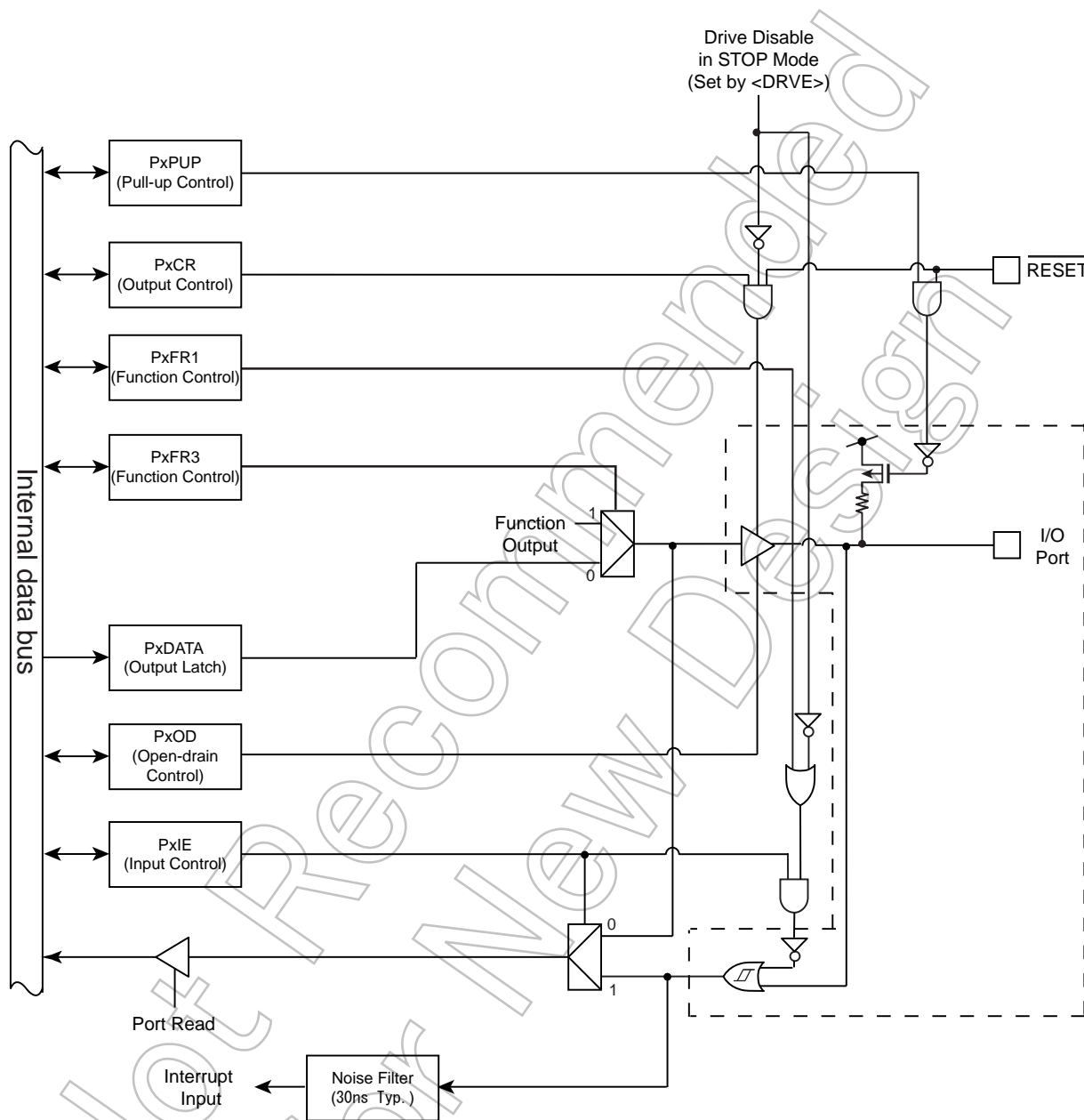
Figure 8-11 Port Type T11

8.3.13   Type T12
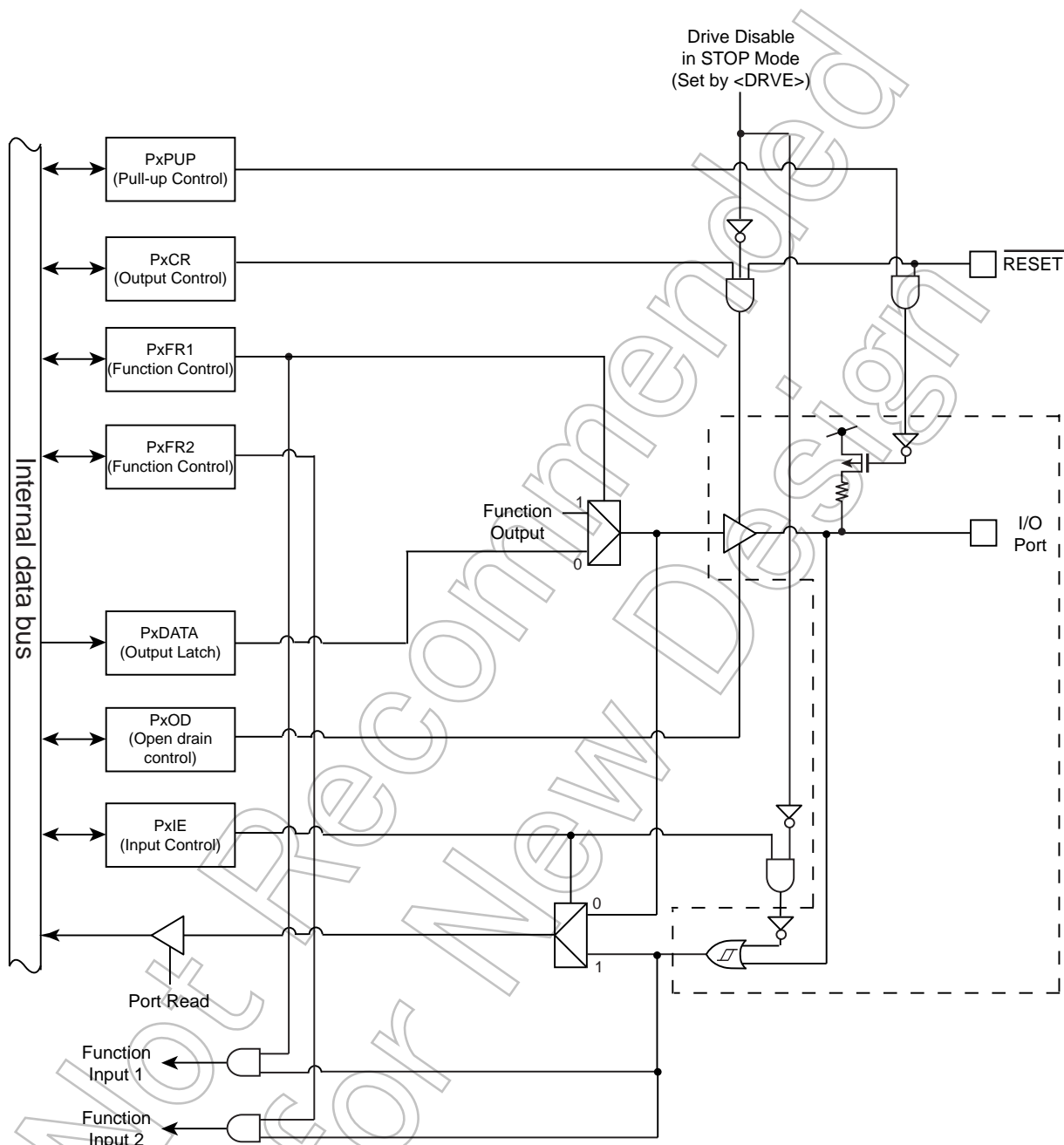
Figure 8-12 Port Type T12

8.3.14 TypeT13
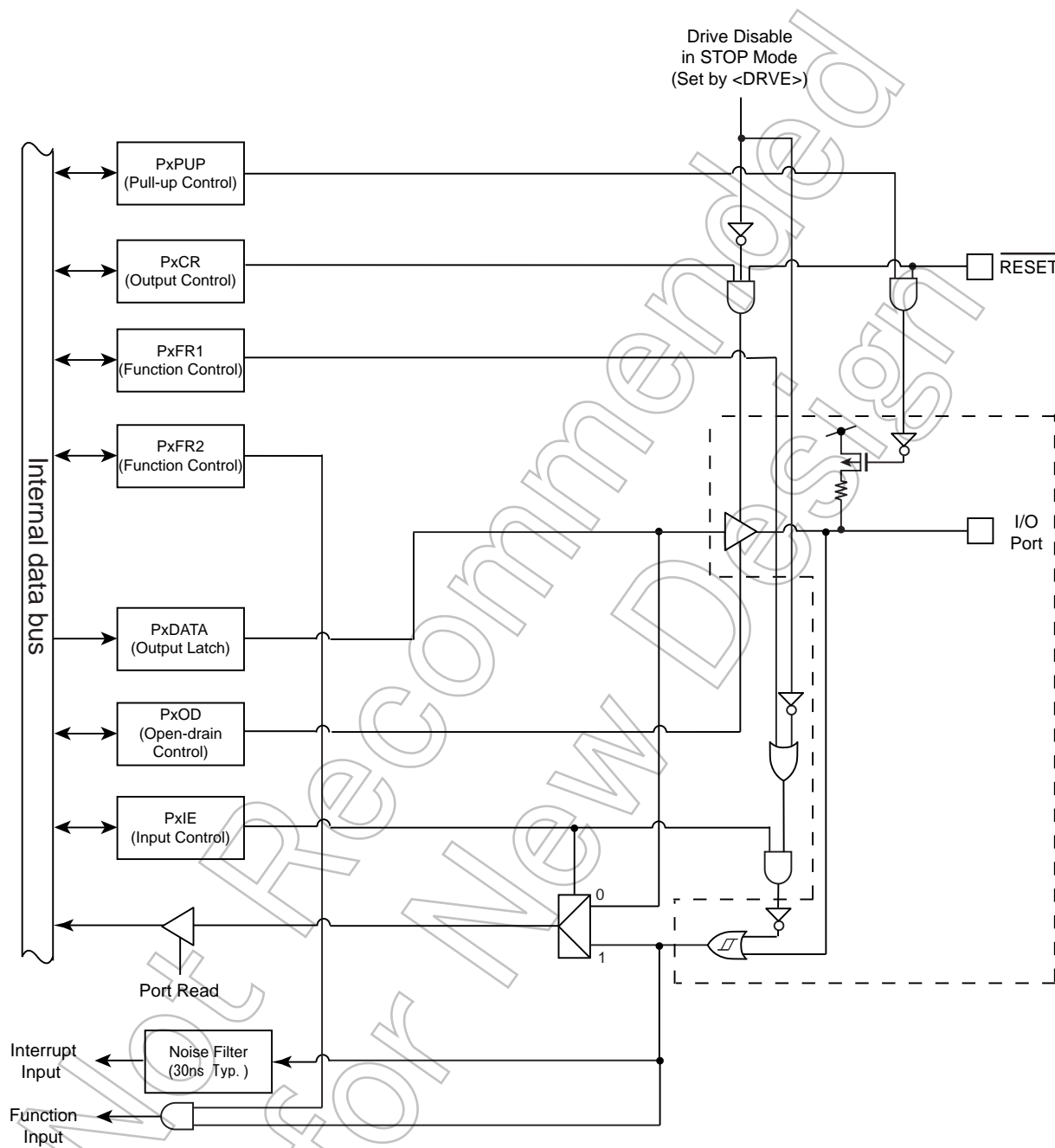
Figure 8-13 Port Type T13

8.3.15 Type T14

Figure 8-14 Port Type T14

8.3.16 Type T15
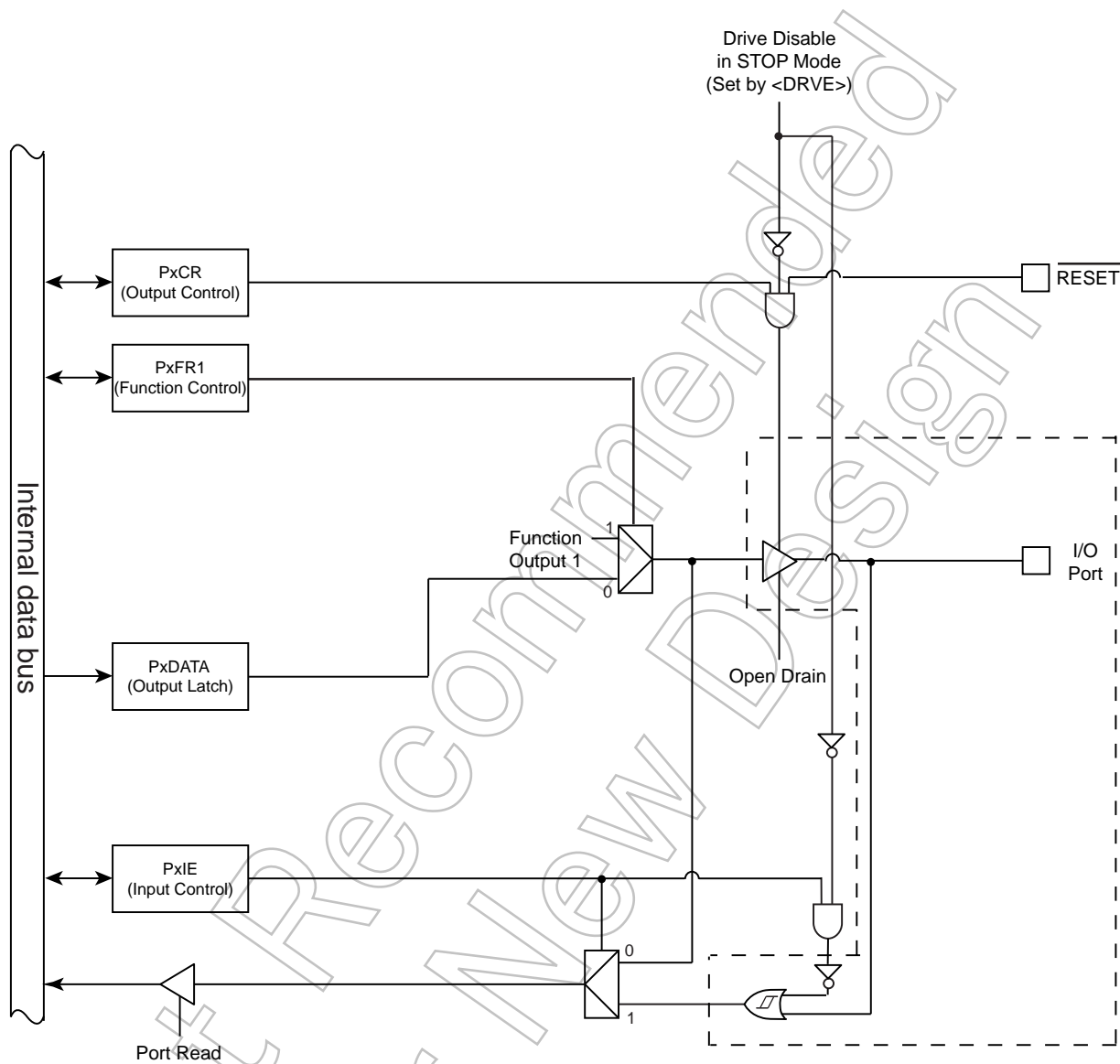
Figure 8-15 Port Type T15

8.3.17 TypeT16
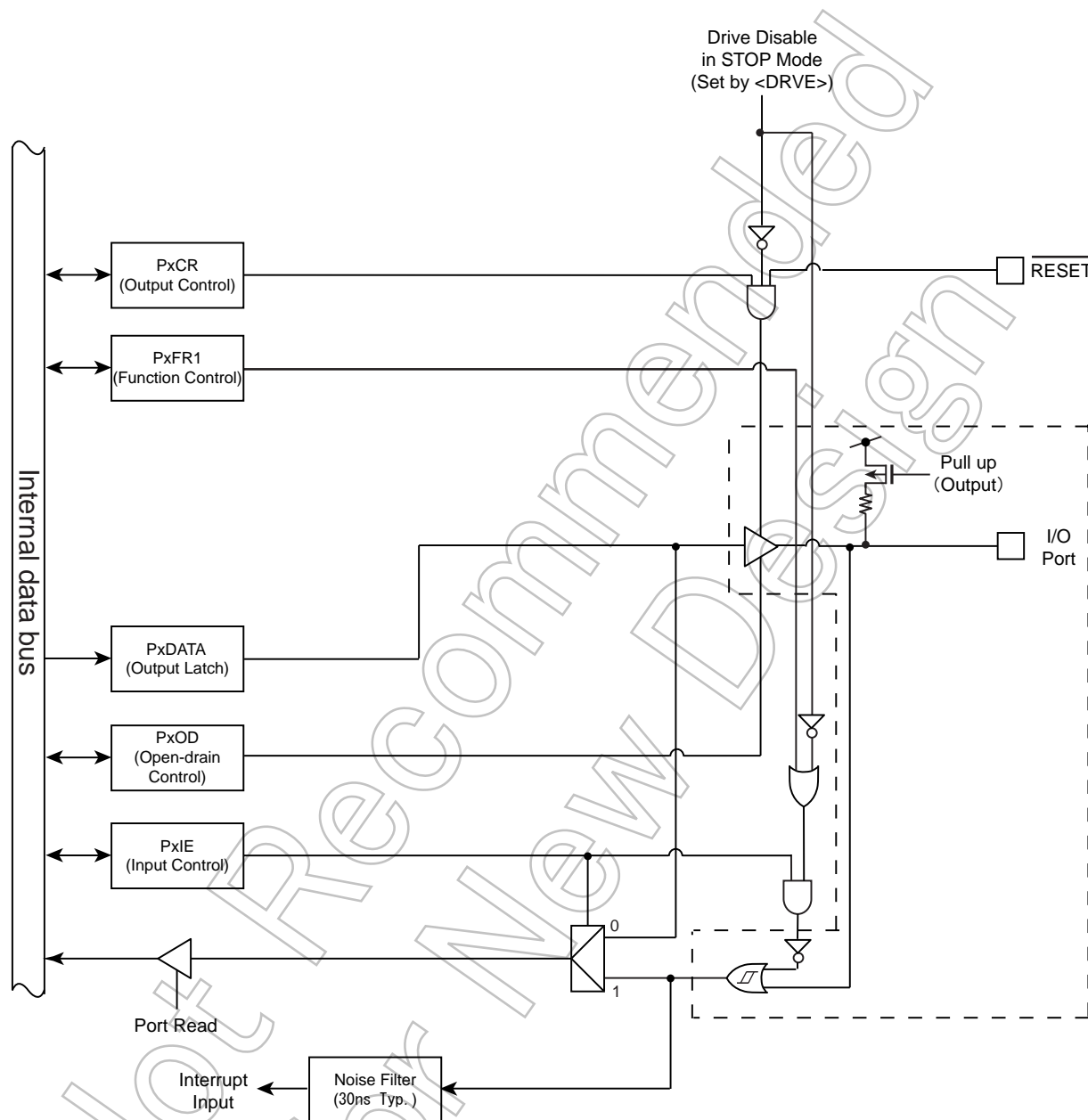
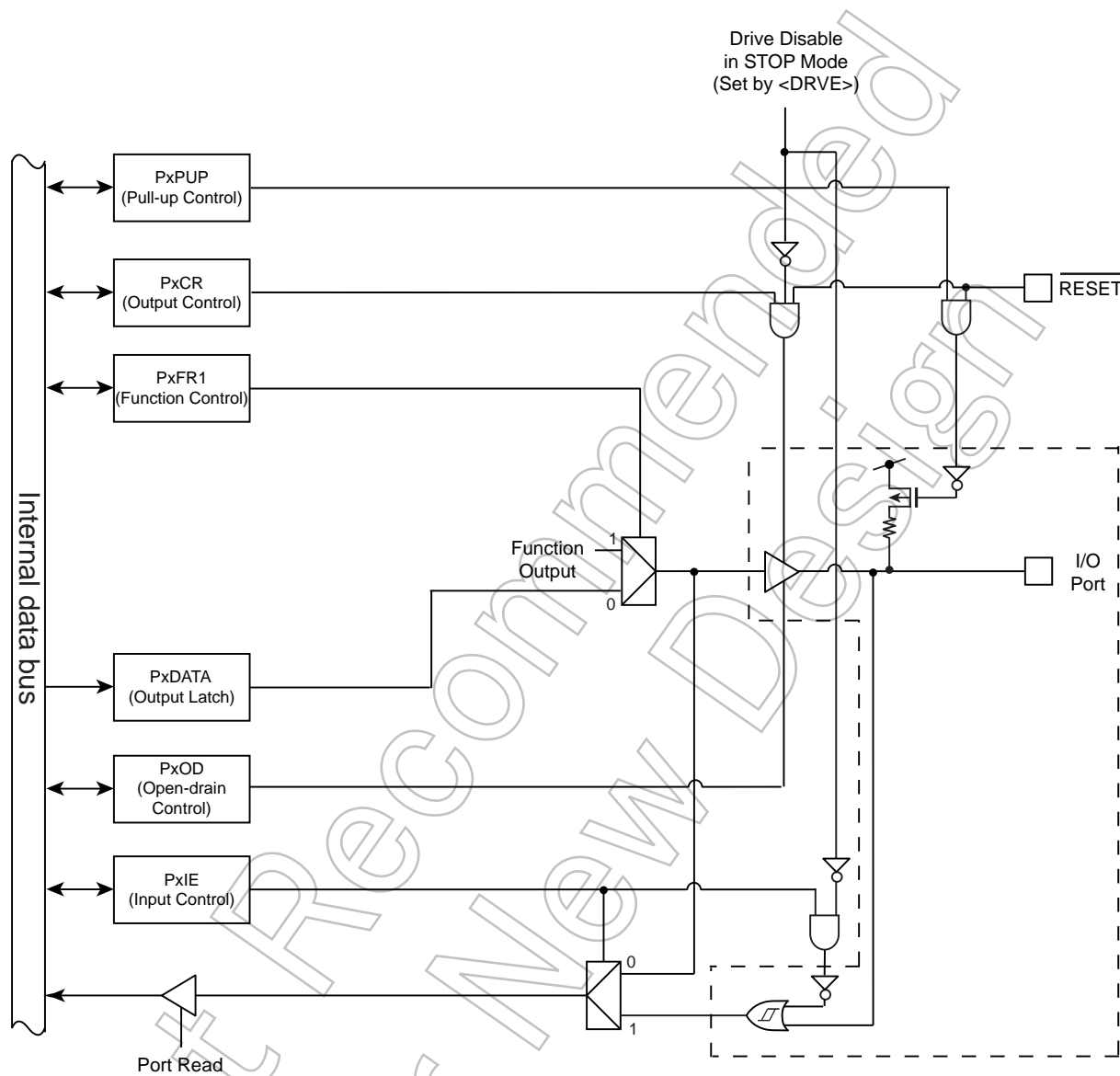Figure 8-16 Port Type T16

8.3.18   Type T17
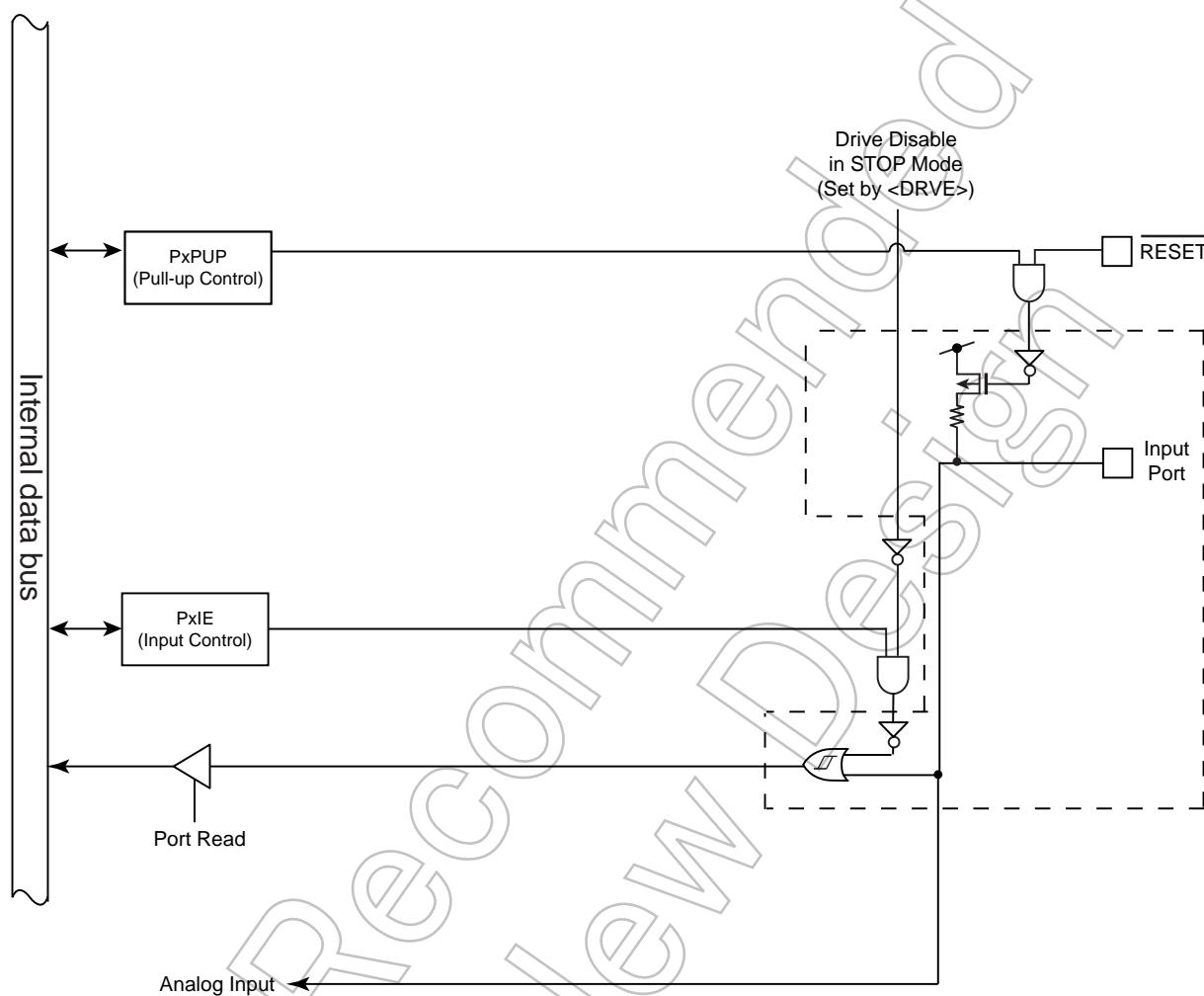
Figure 8-17 Port Type T17

8.3.19   Type T18
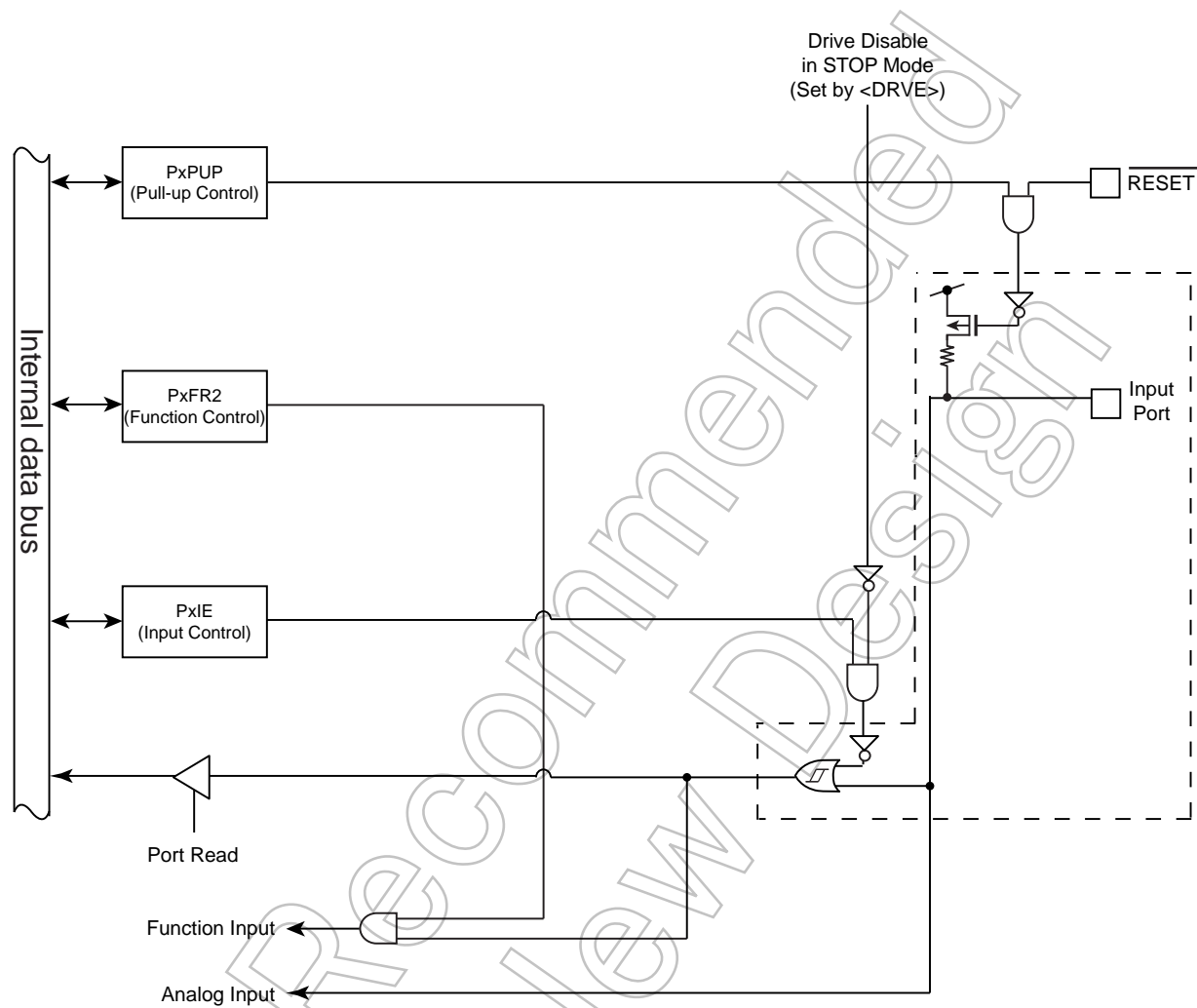
Figure 8-18 Port Type T18

8.3.20   Type T19
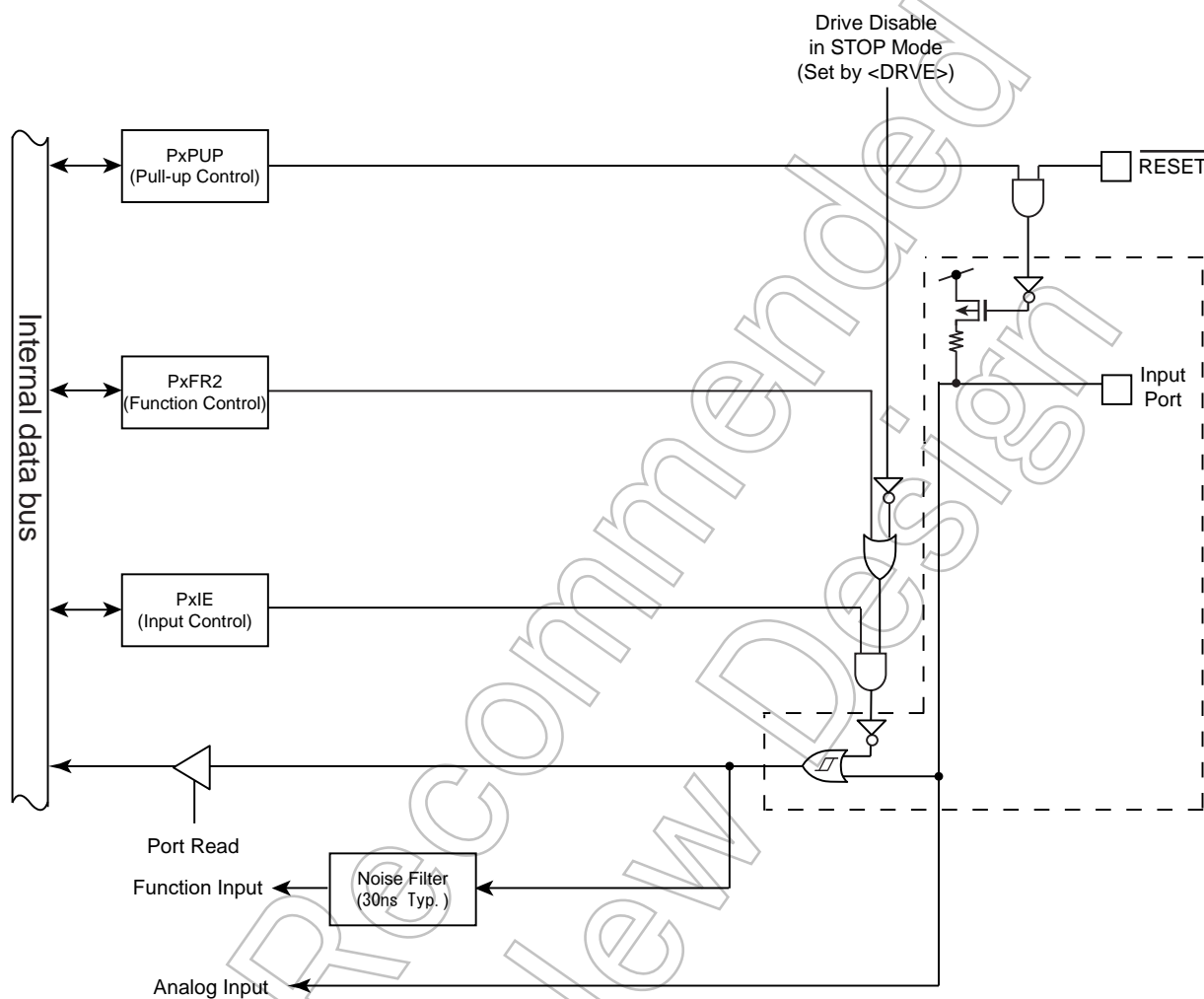
Figure 8-19 Port Type T19

8.3.21 Type T20
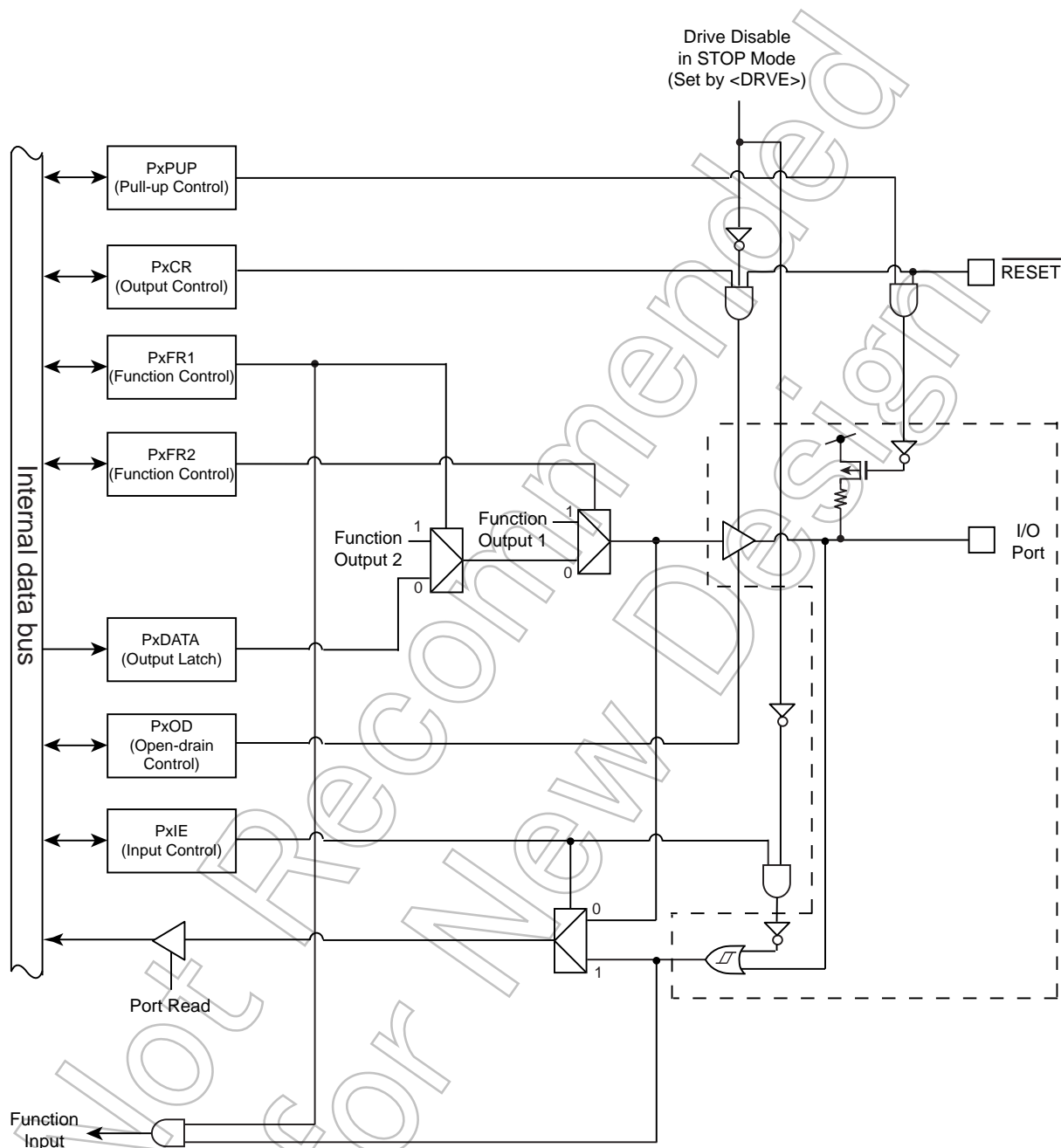
Figure 8-20 Port Type T20

8.3.22   Type T21
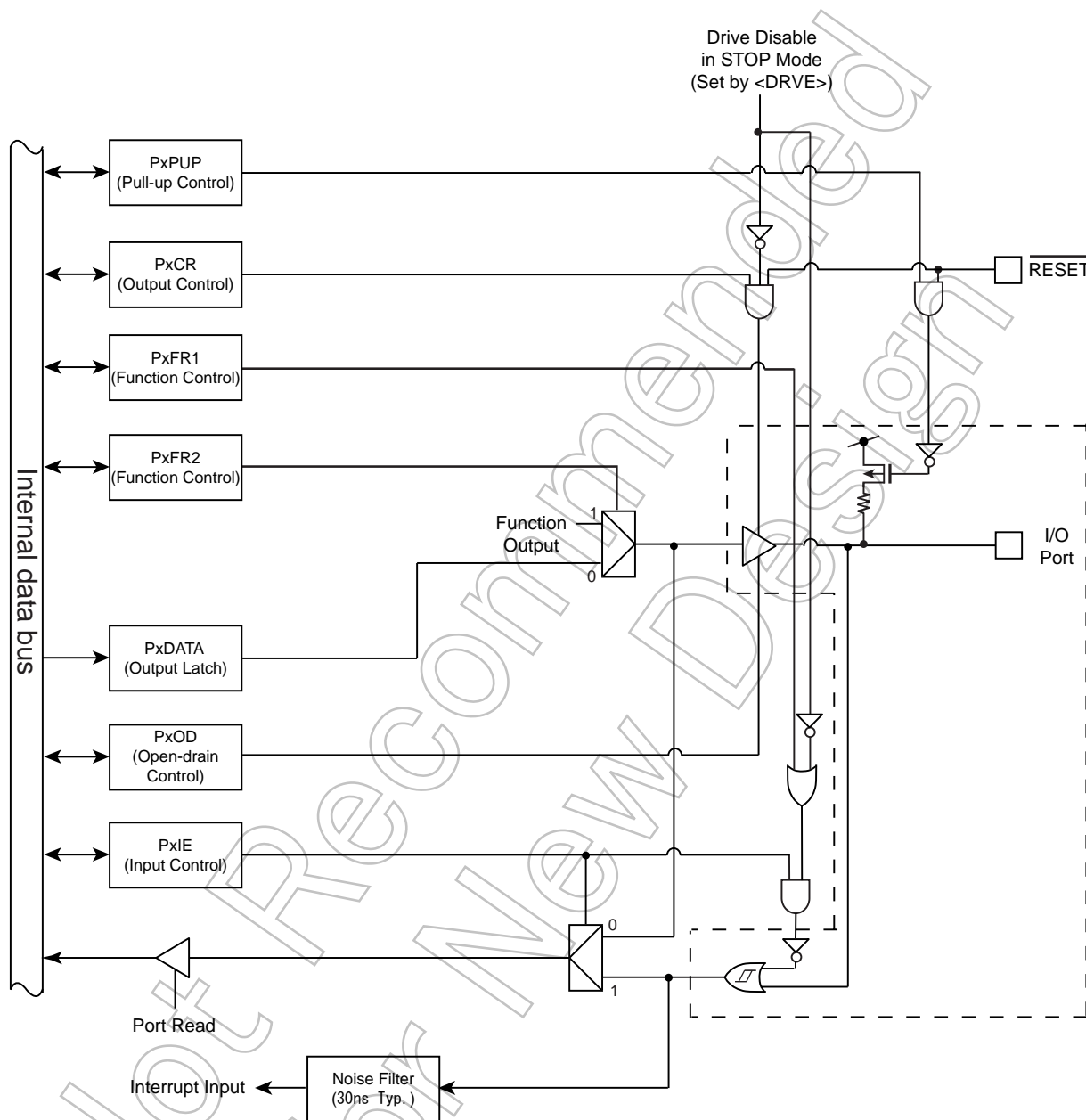
Figure 8-21 Port Type T21

8.3.23 Type T22
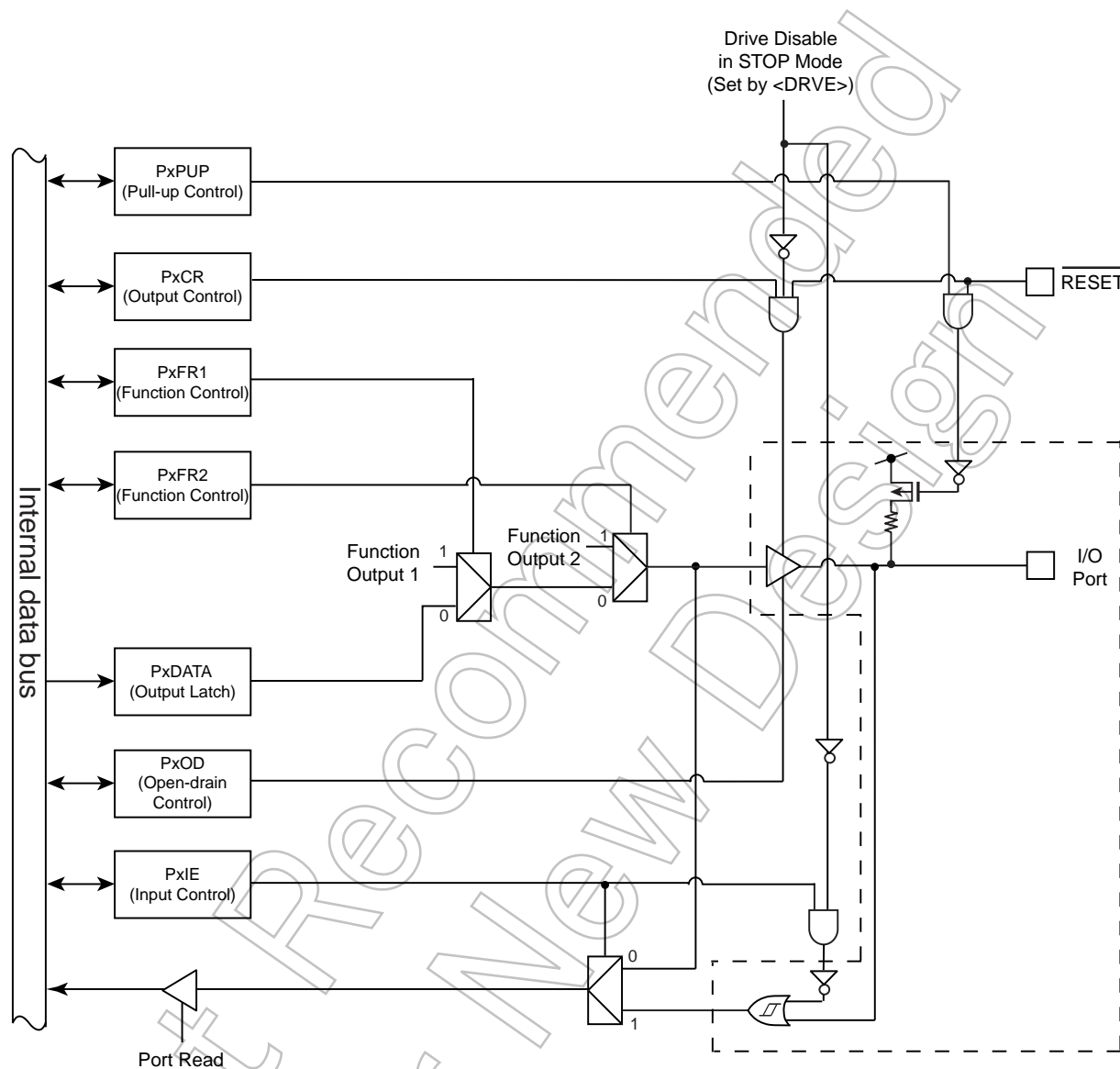
Figure 8-22 Port Type T22

8.3.24   Type T23
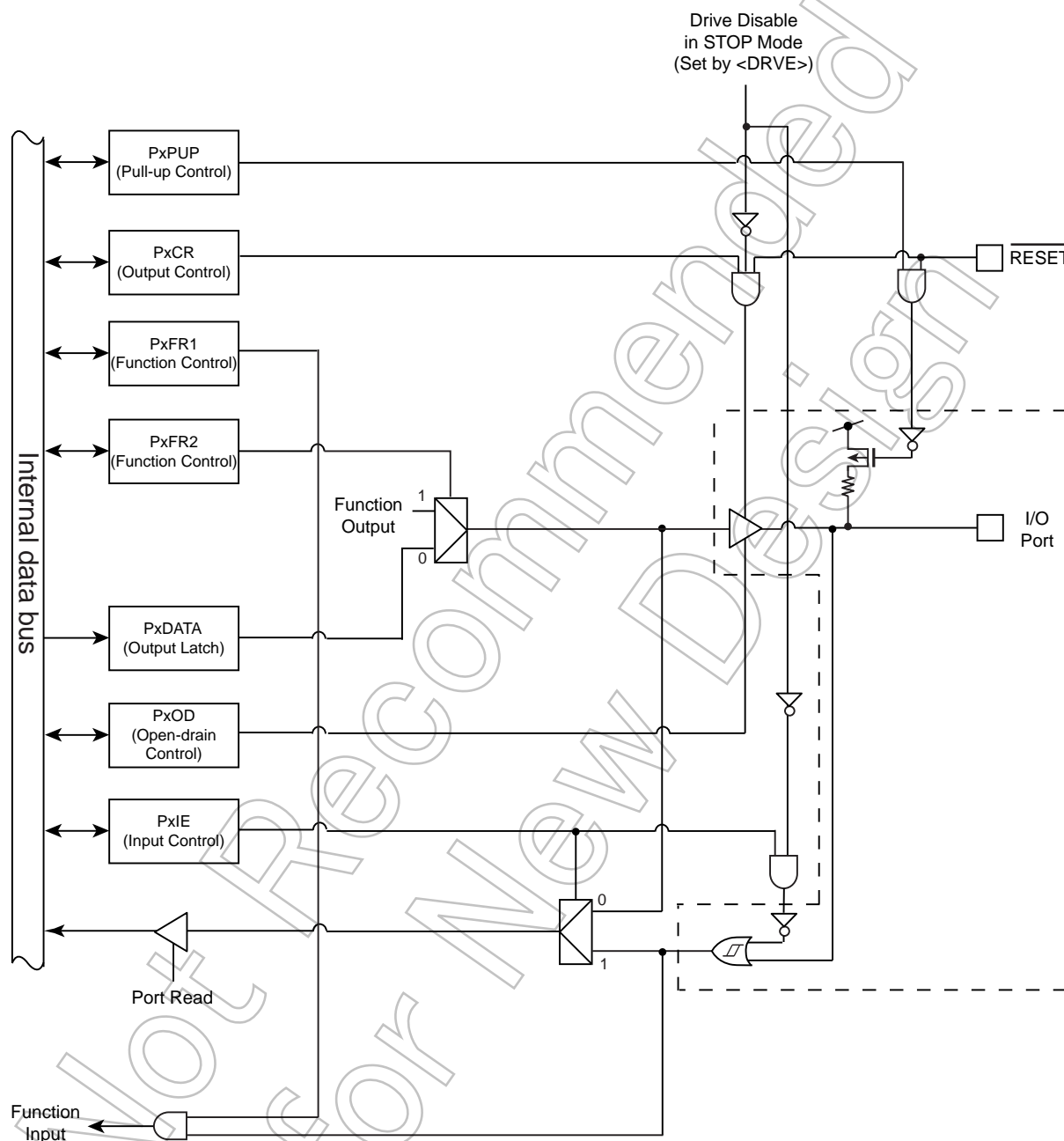
Figure 8-23 Port Type T23

8.3.25 Type T24
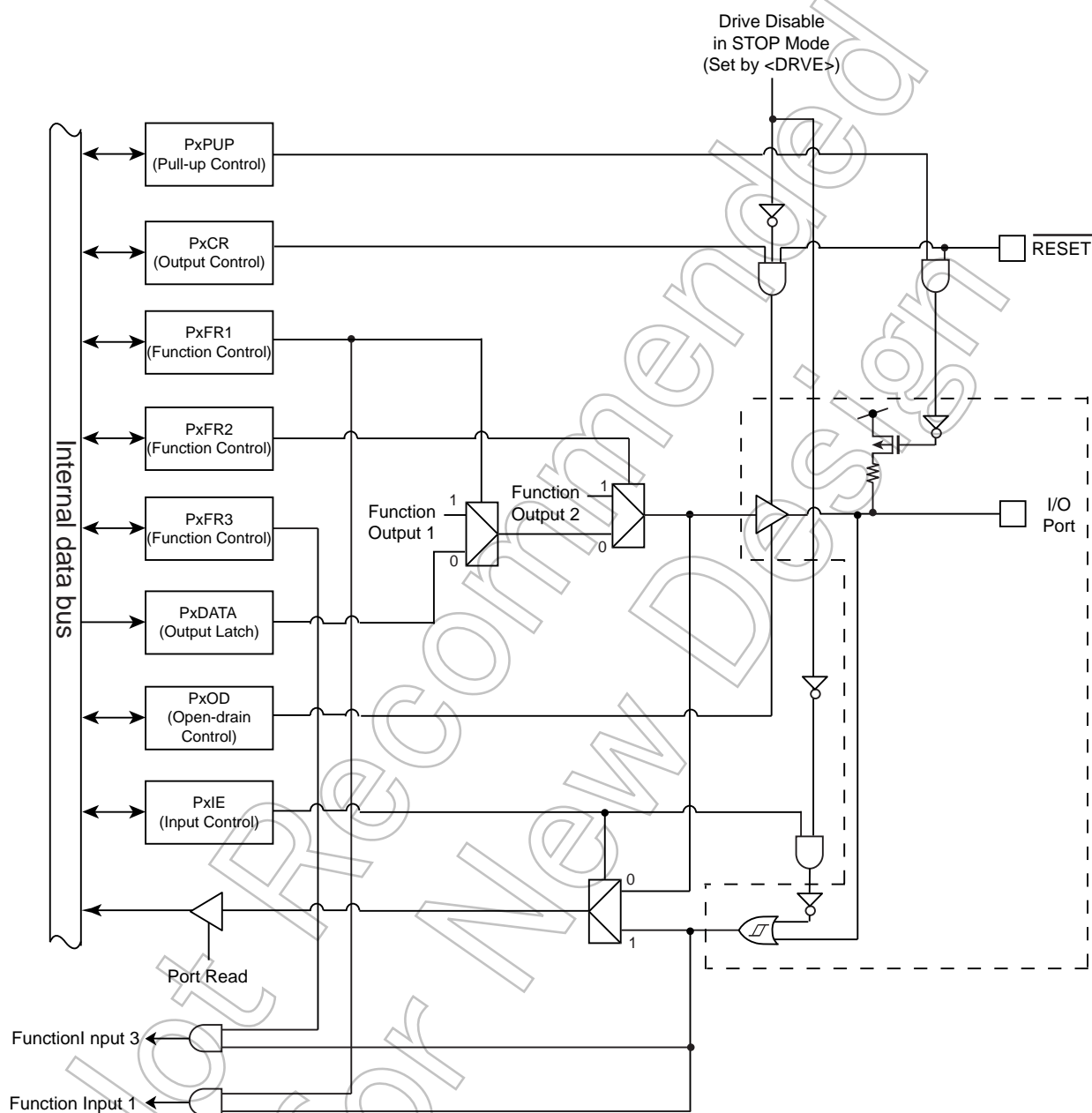
Figure 8-24 Port Type T24

8.3.26   Type T25
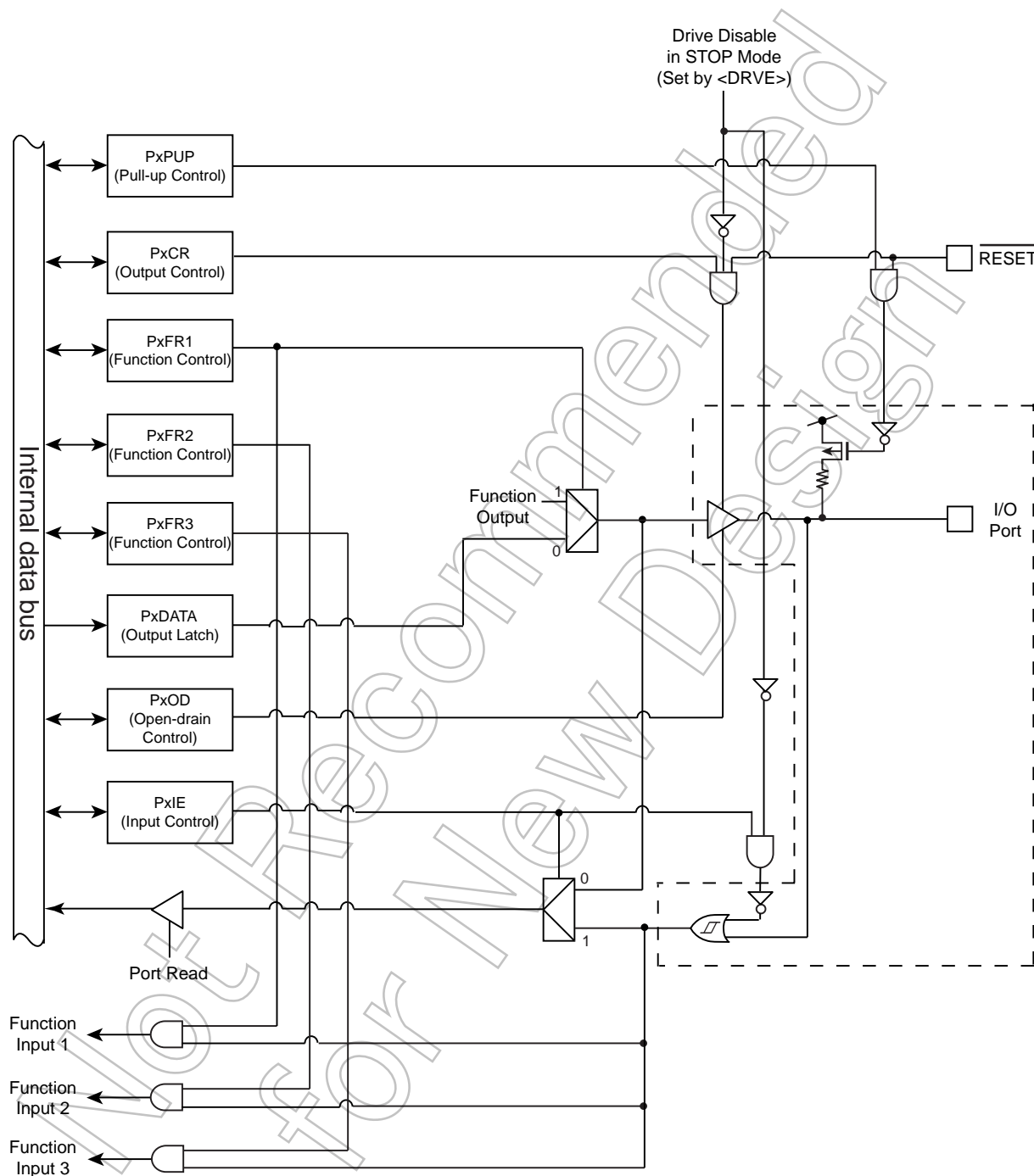
Figure 8-25 Port Type T25

8.3.27 TypeT26
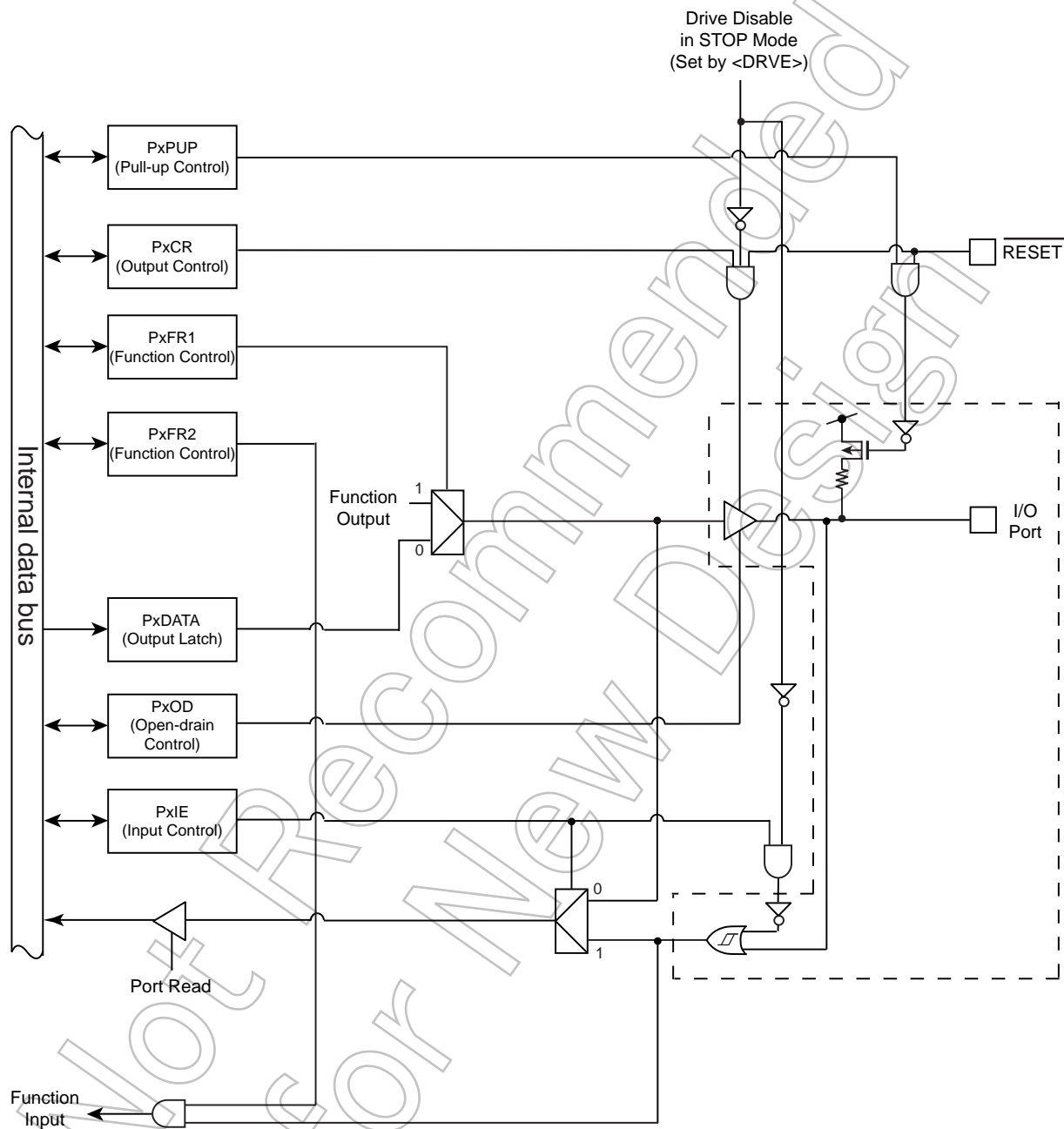
Figure 8-26 Port Type T26

8.3.28   Type T27
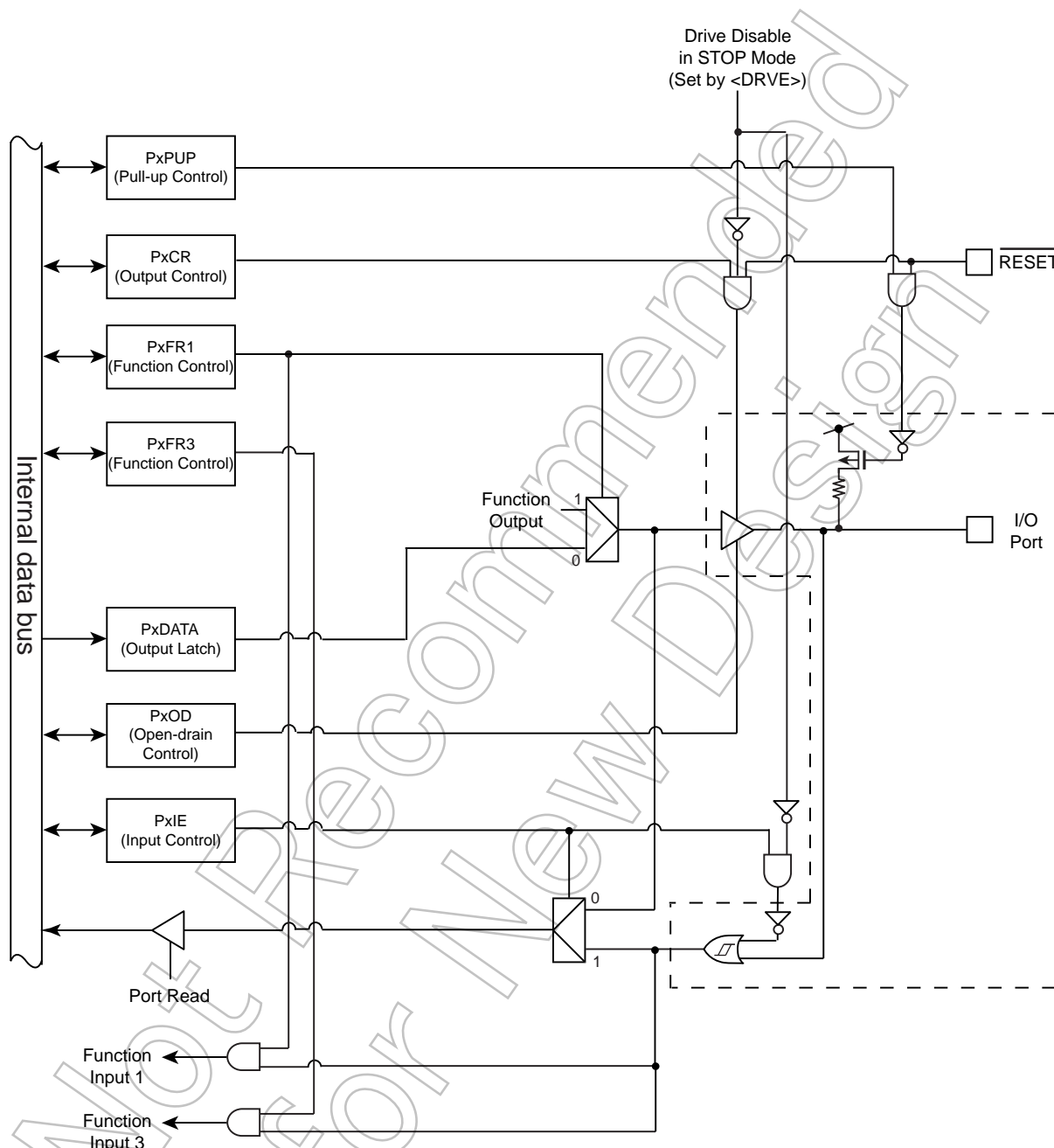
Figure 8-27 Port Type T27

8.3.29   Type T28
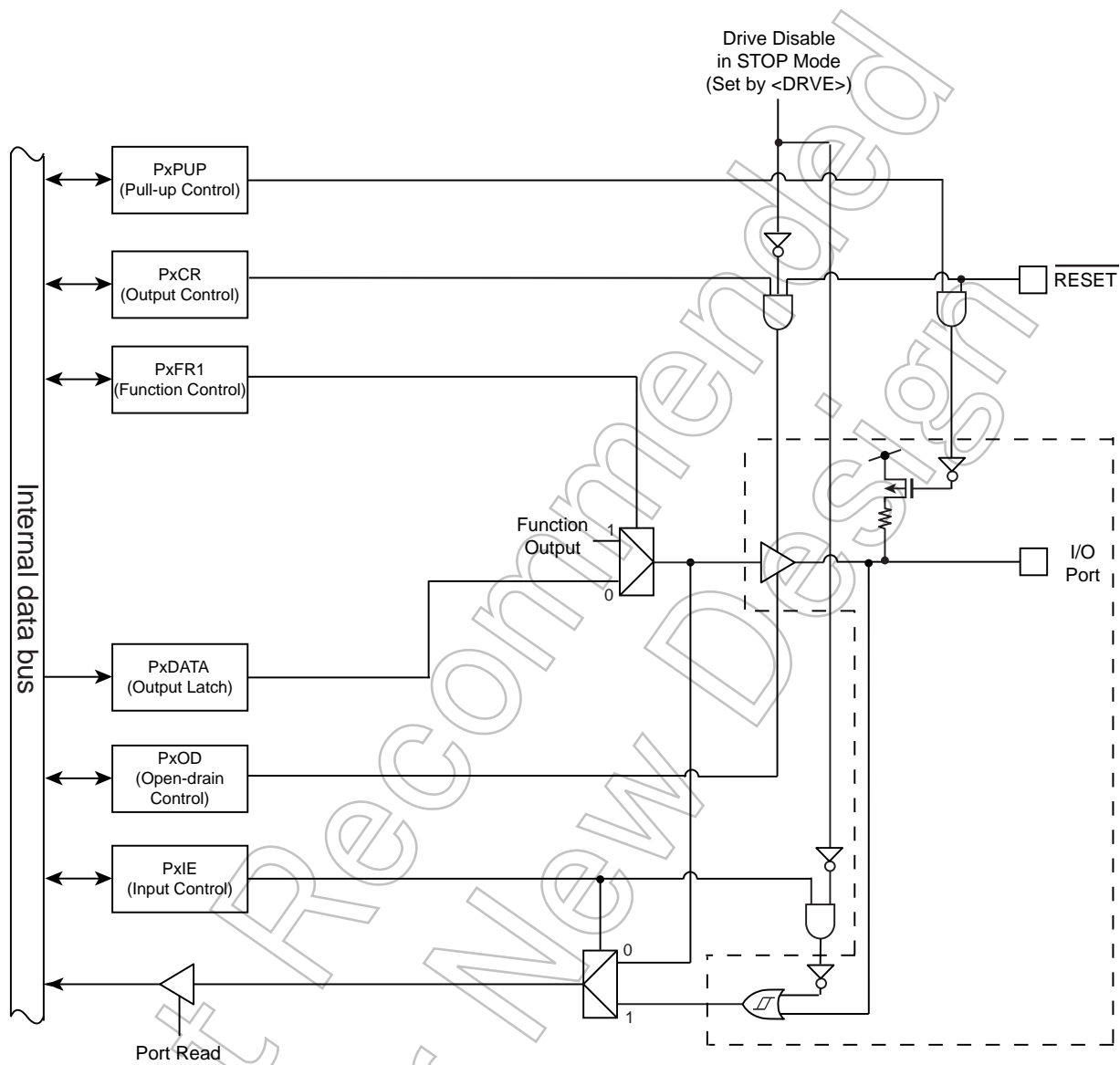
Figure 8-28 Port Type T28

8.3.30　Type T29
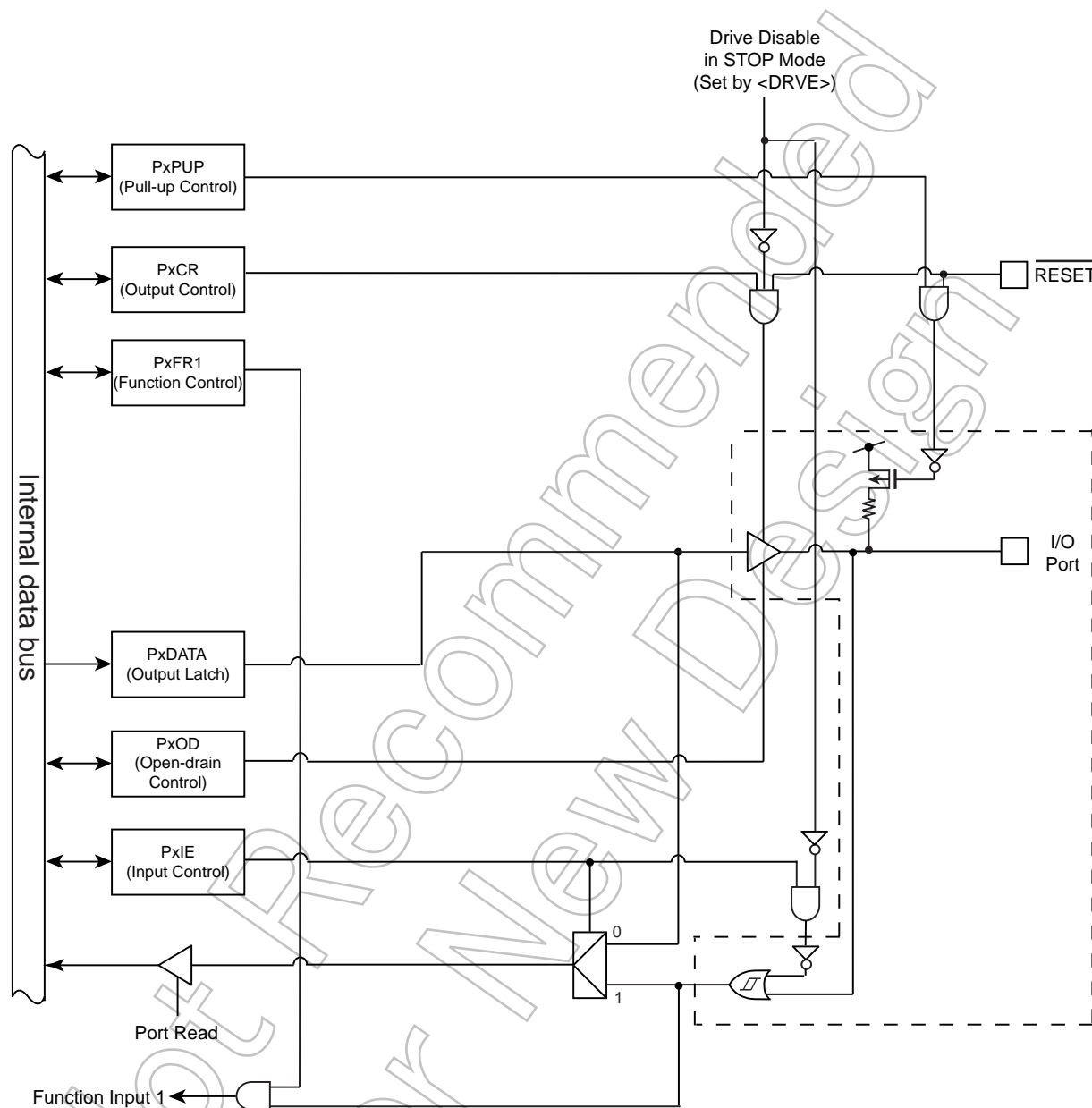
Figure 8-29 Port Type T29

8.3.31   Type T30
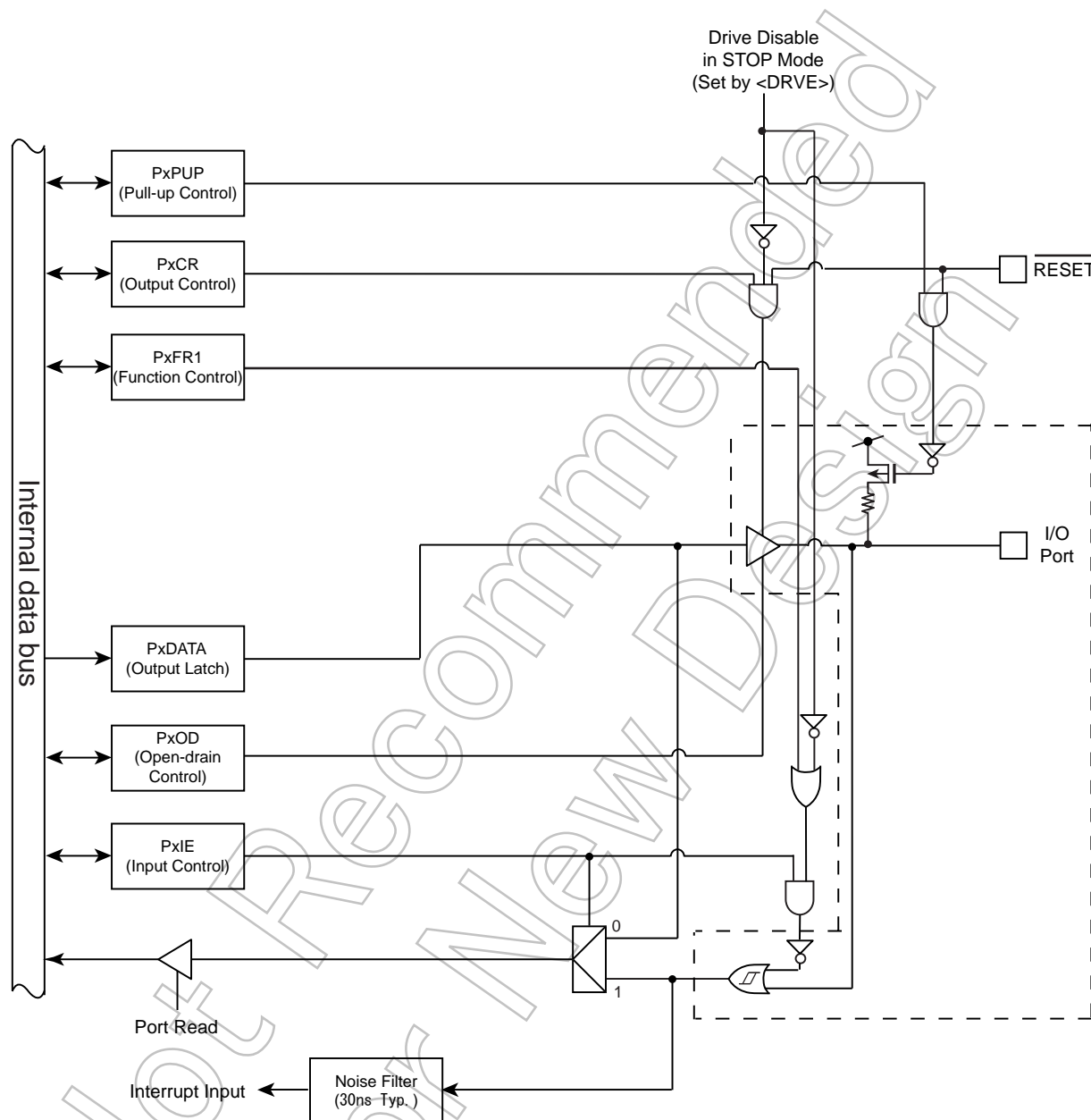
Figure 8-30 Port Type T30

8.3.32   Type T31
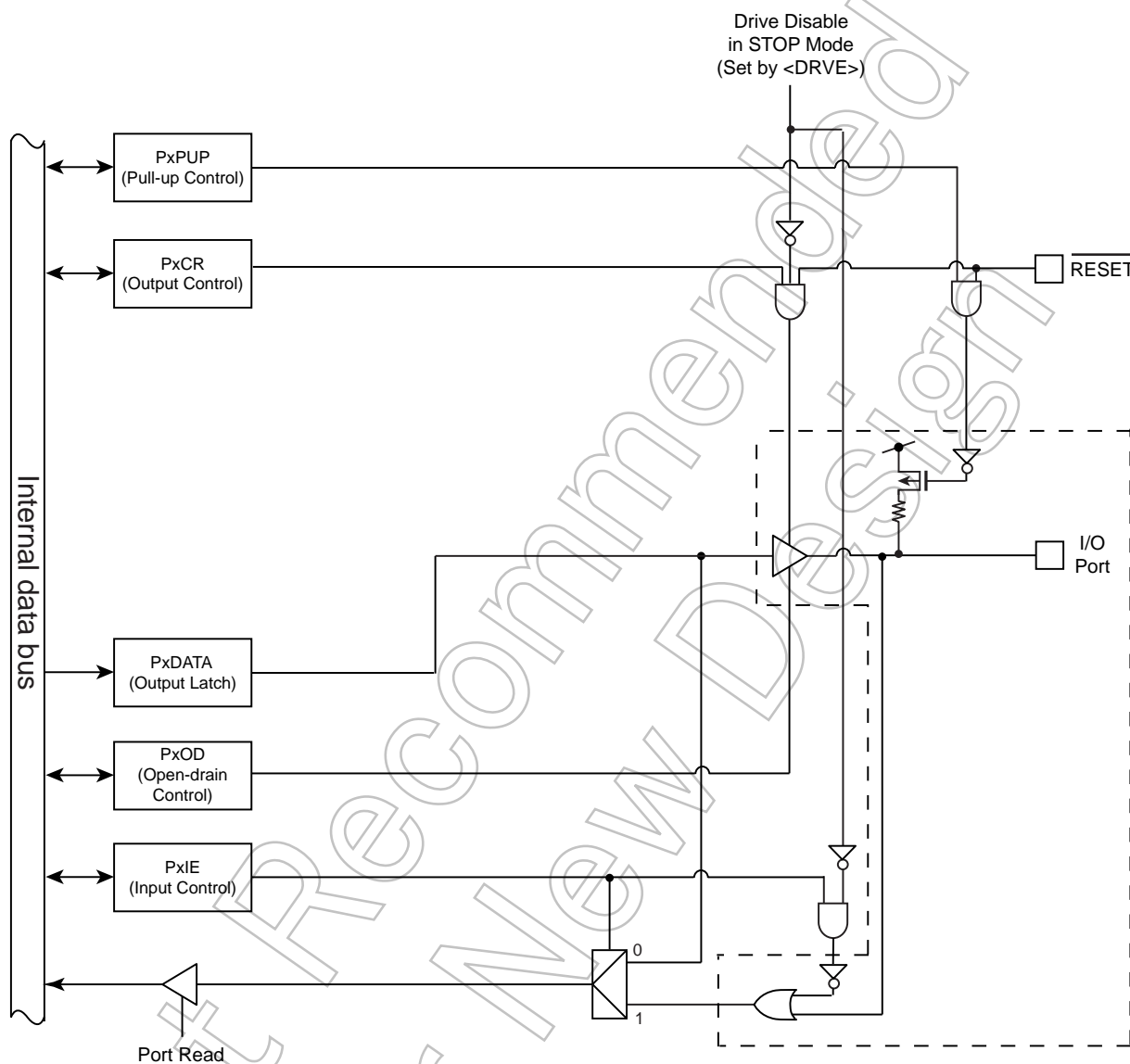
Figure 8-31 Port Type T31

## 8.3.33   Type T32
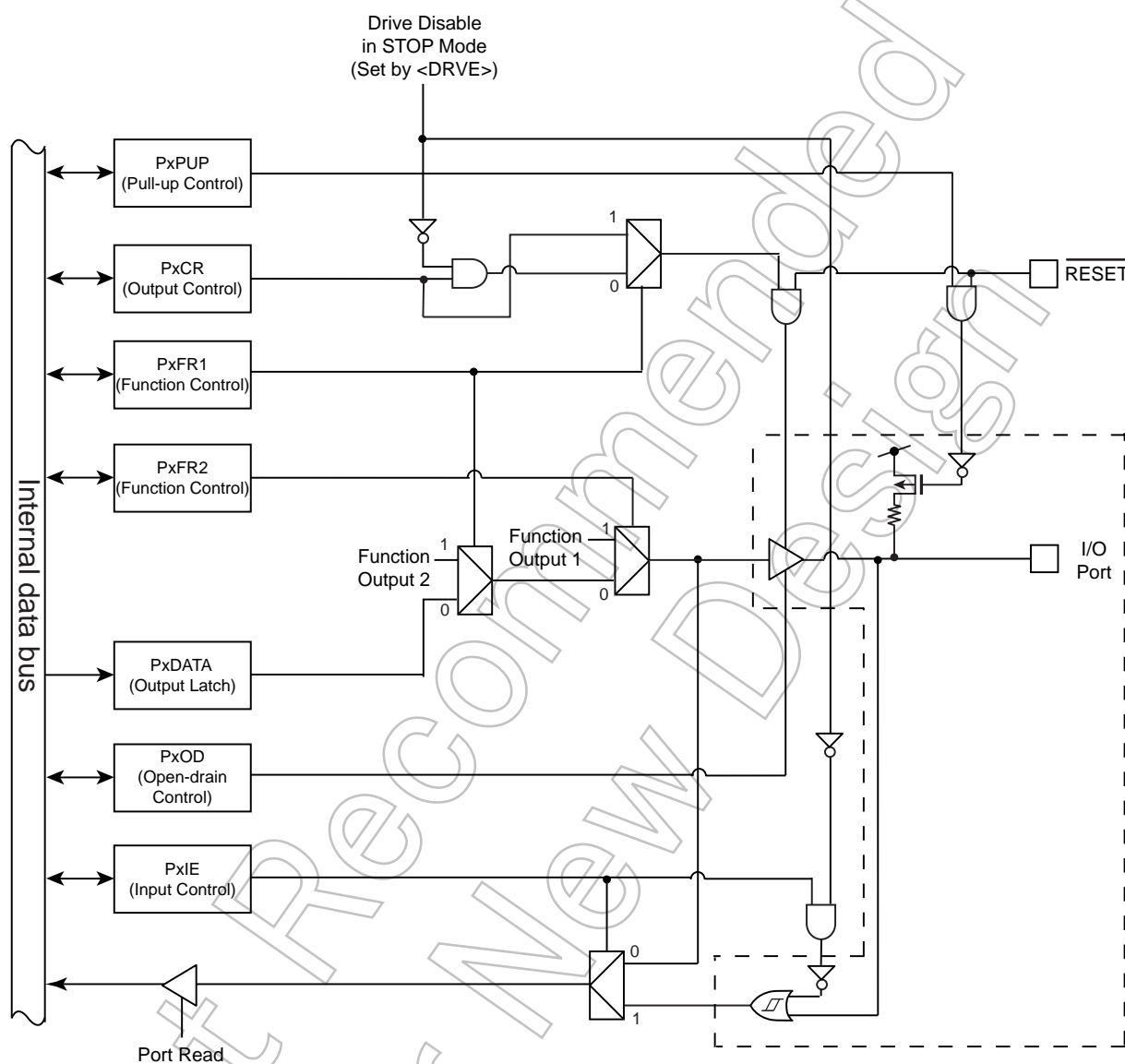
Figure 8-32 Port Type T32

8.3.34   Type T33
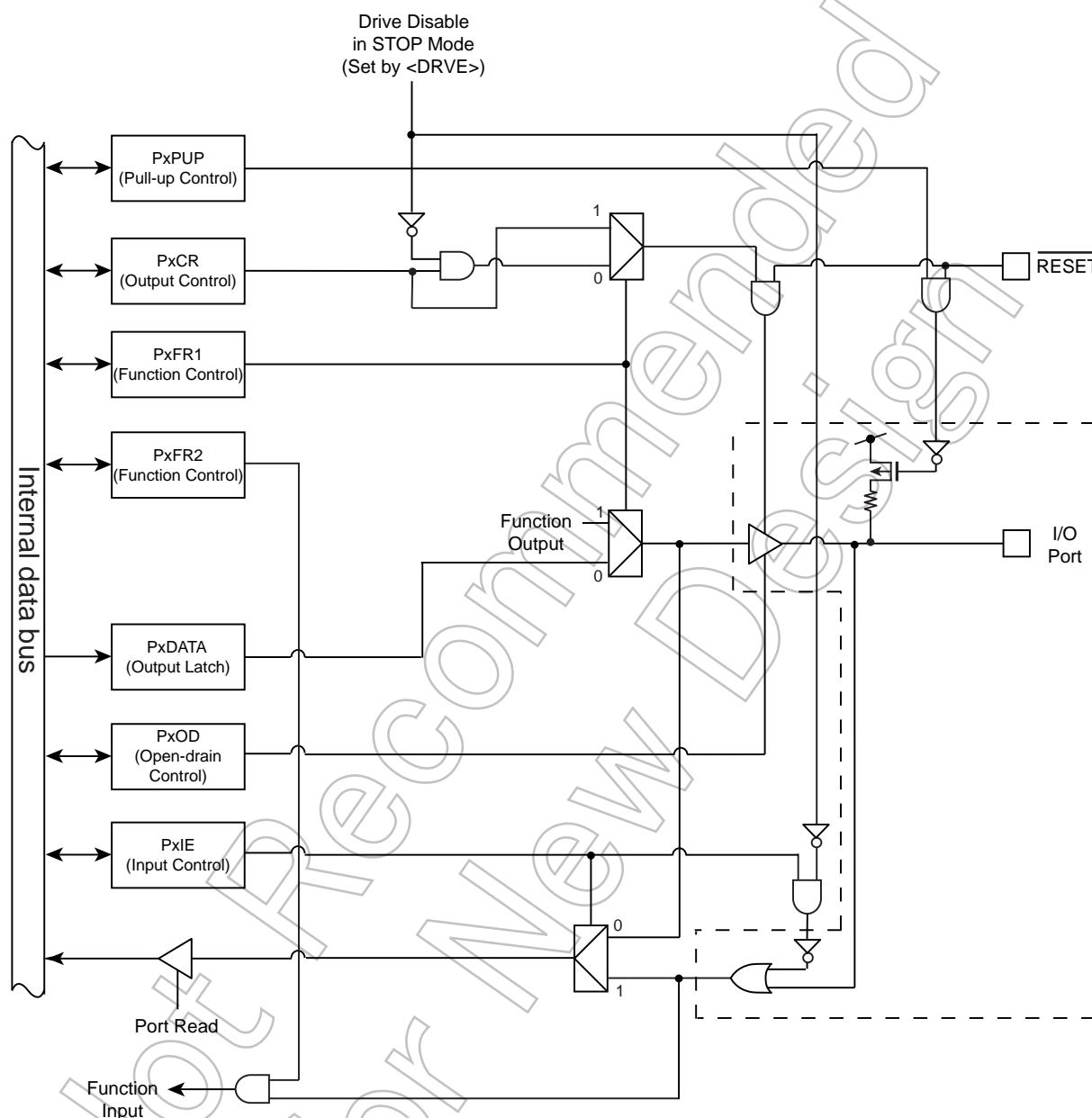
Figure 8-33 Port Type T33

8.3.35 Type T34
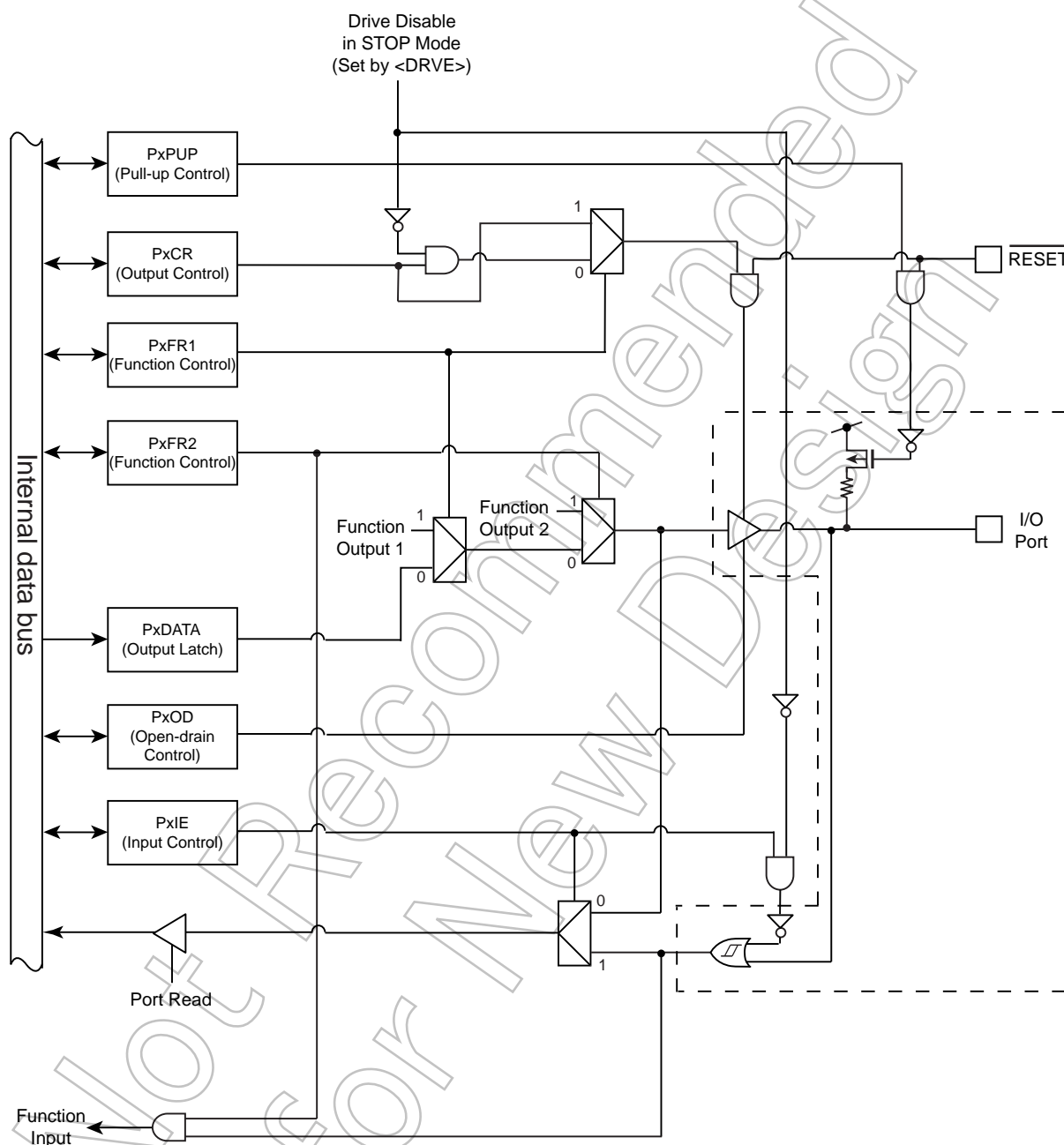
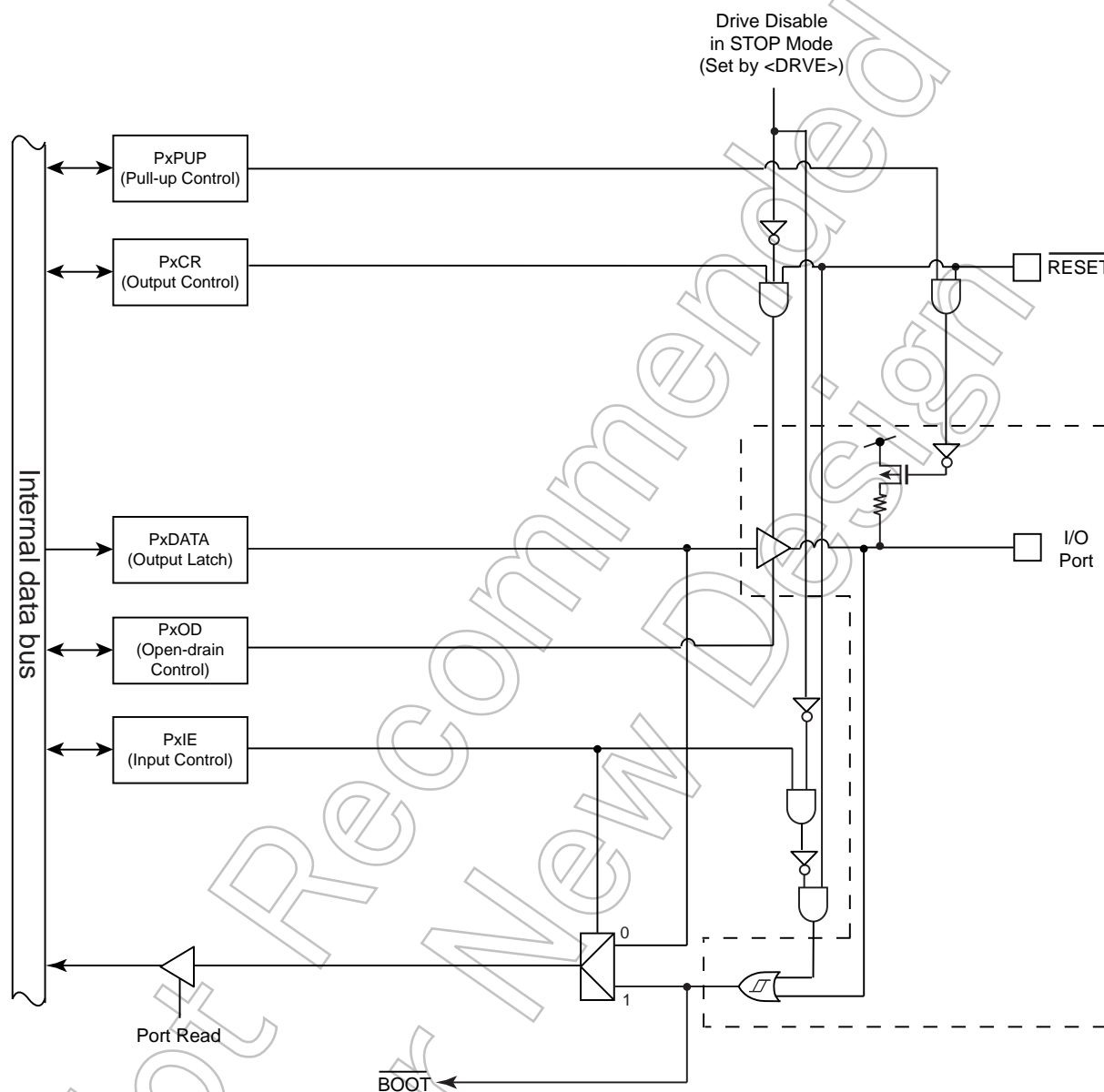

Figure 8-34 Port Type T34

8.3.36   Type T35
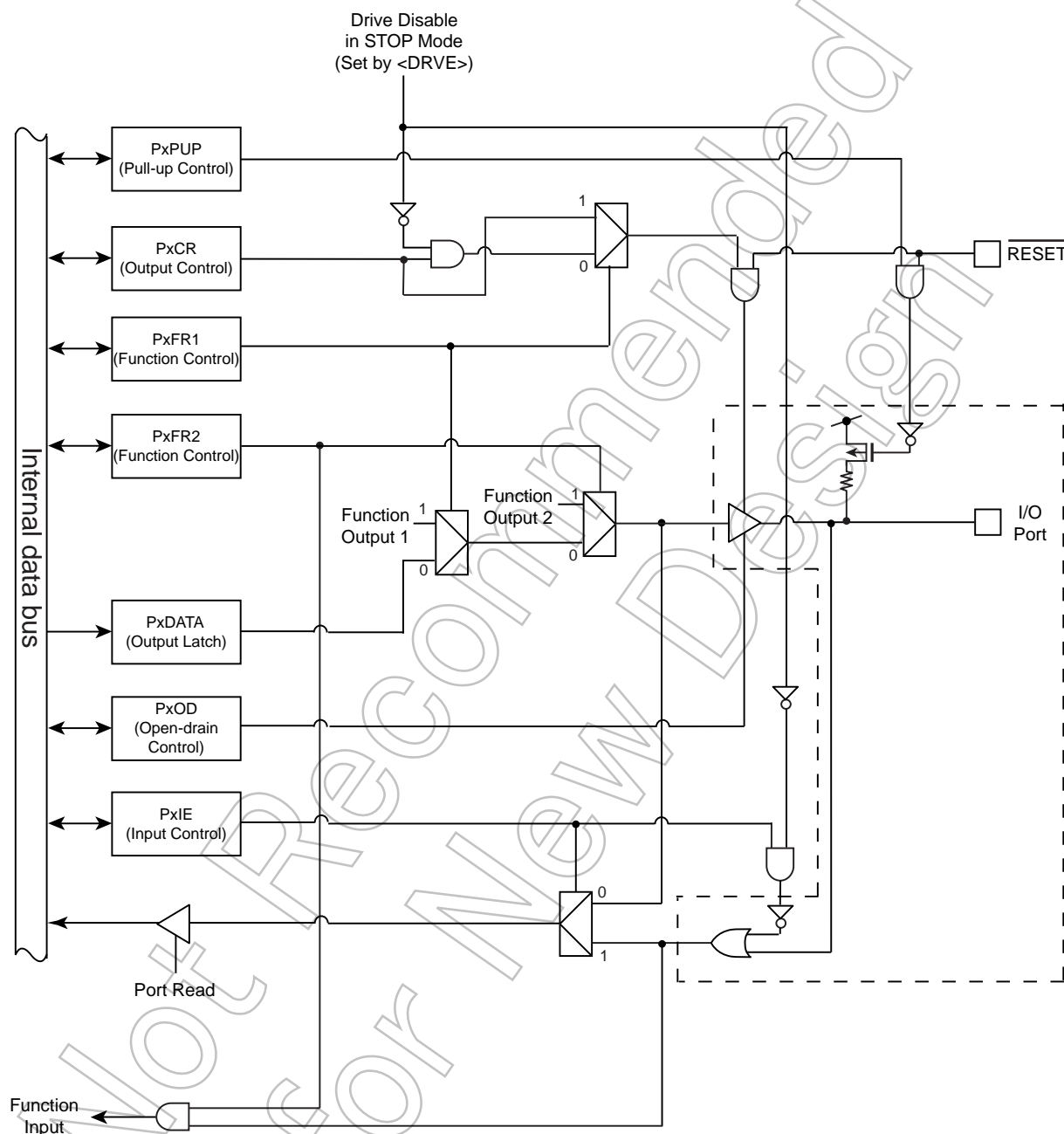
Figure 8-35 Port Type T35

### 8.3.37 Type T36
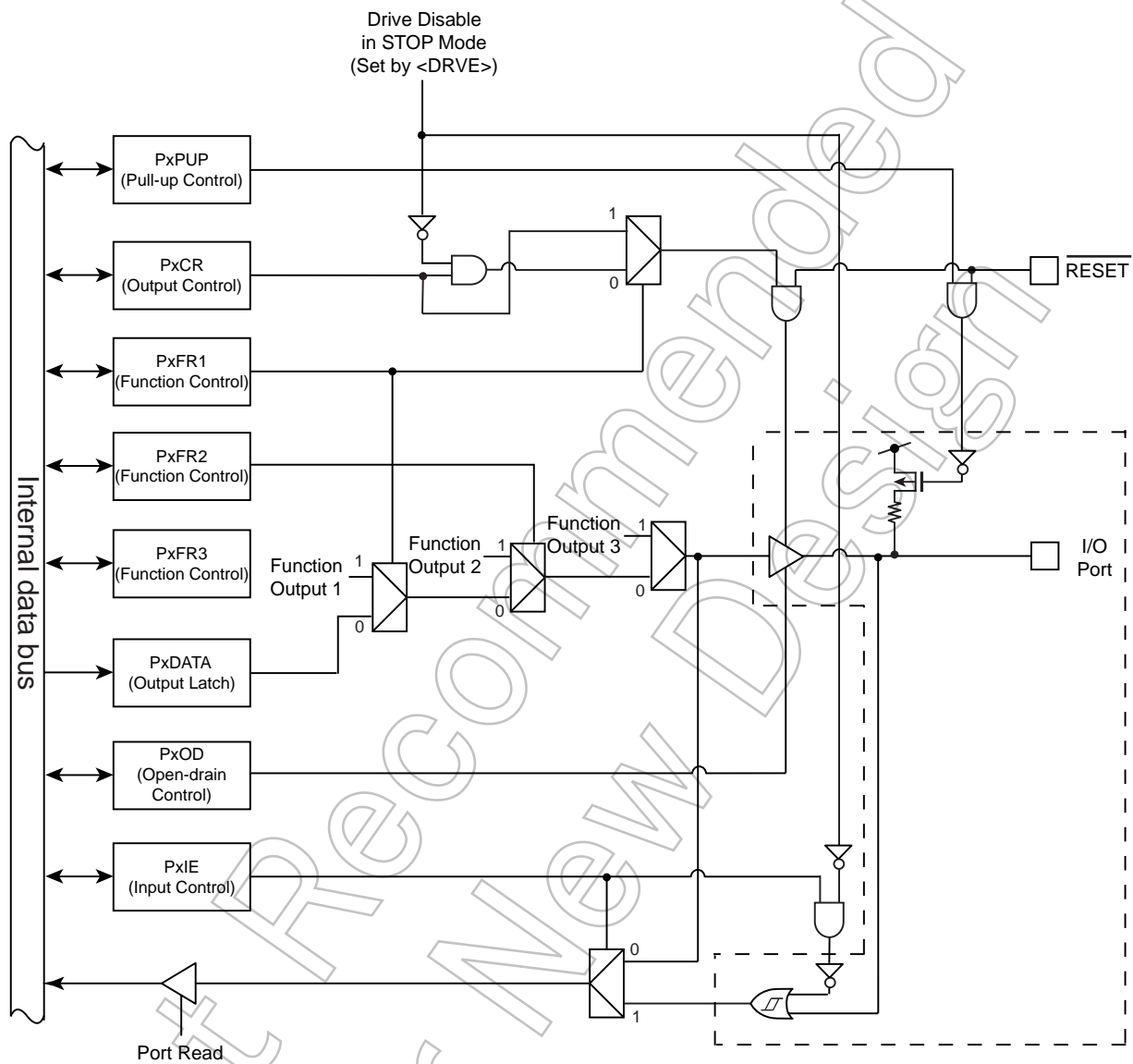
Figure 8-36 Port Type T36

8.3.38   Type T37
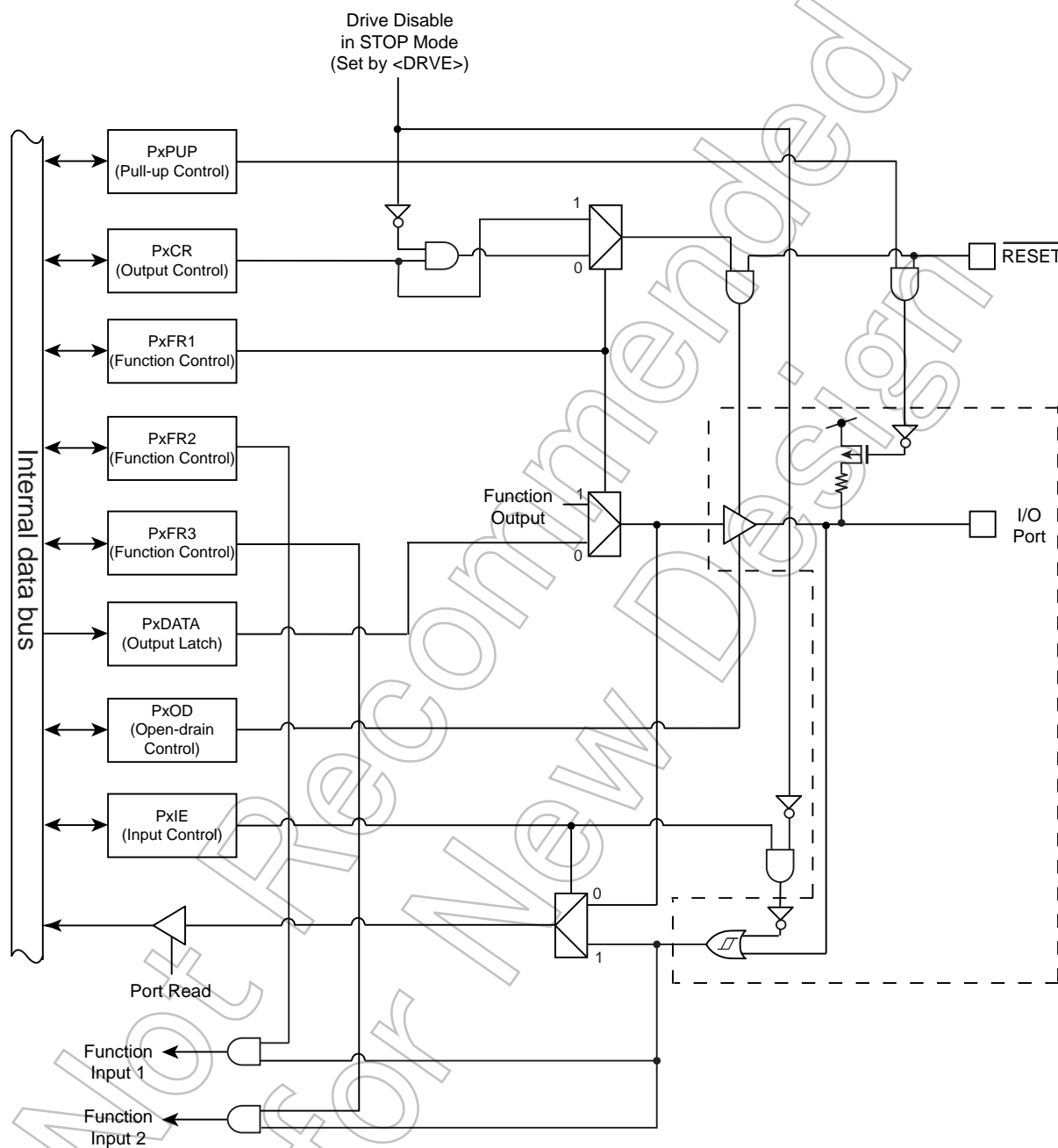
Figure 8-37 Port Type T37

8.3.39 Type T38
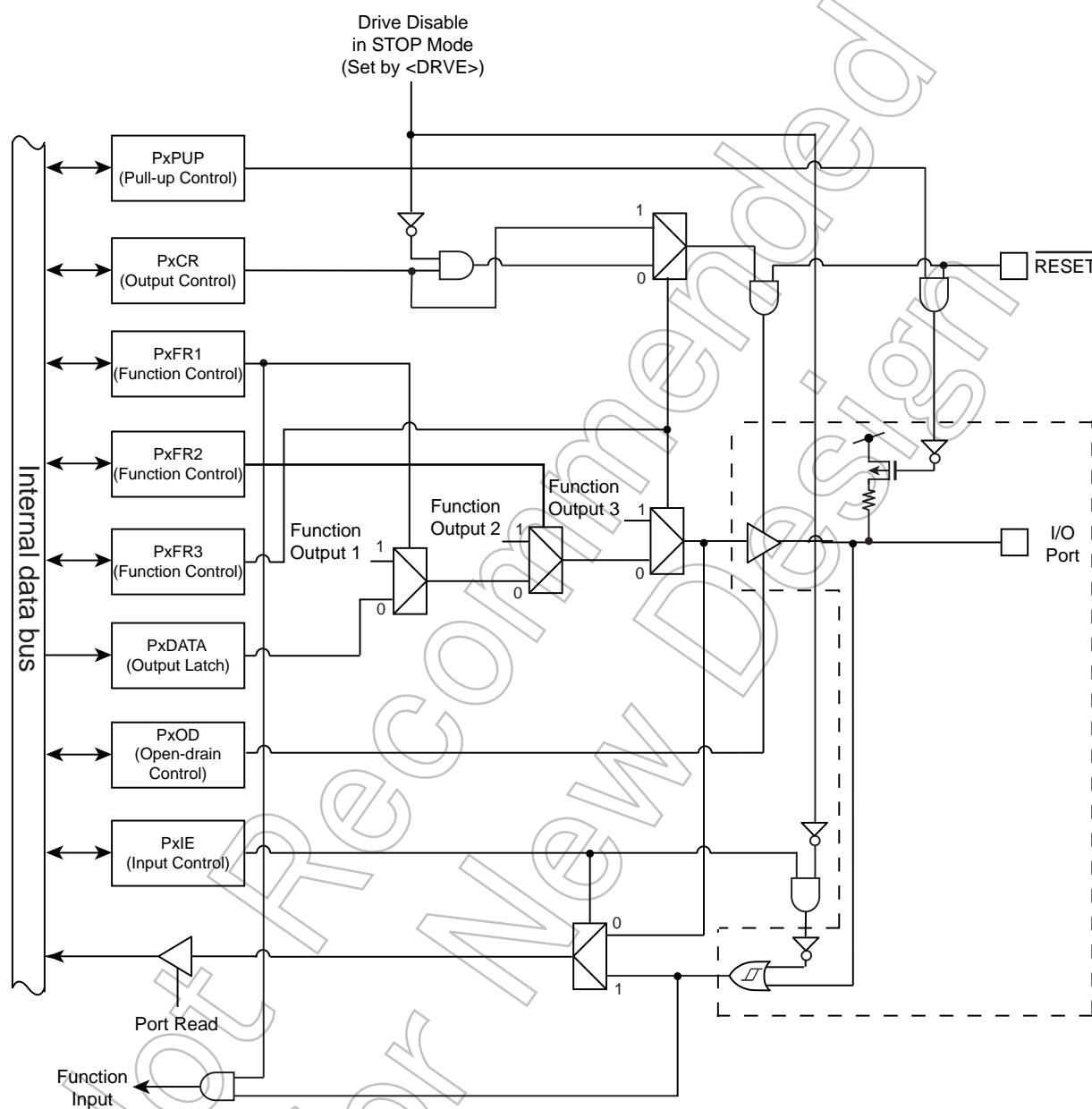
Figure 8-38 Port Type T38

8.3.40 Type T39
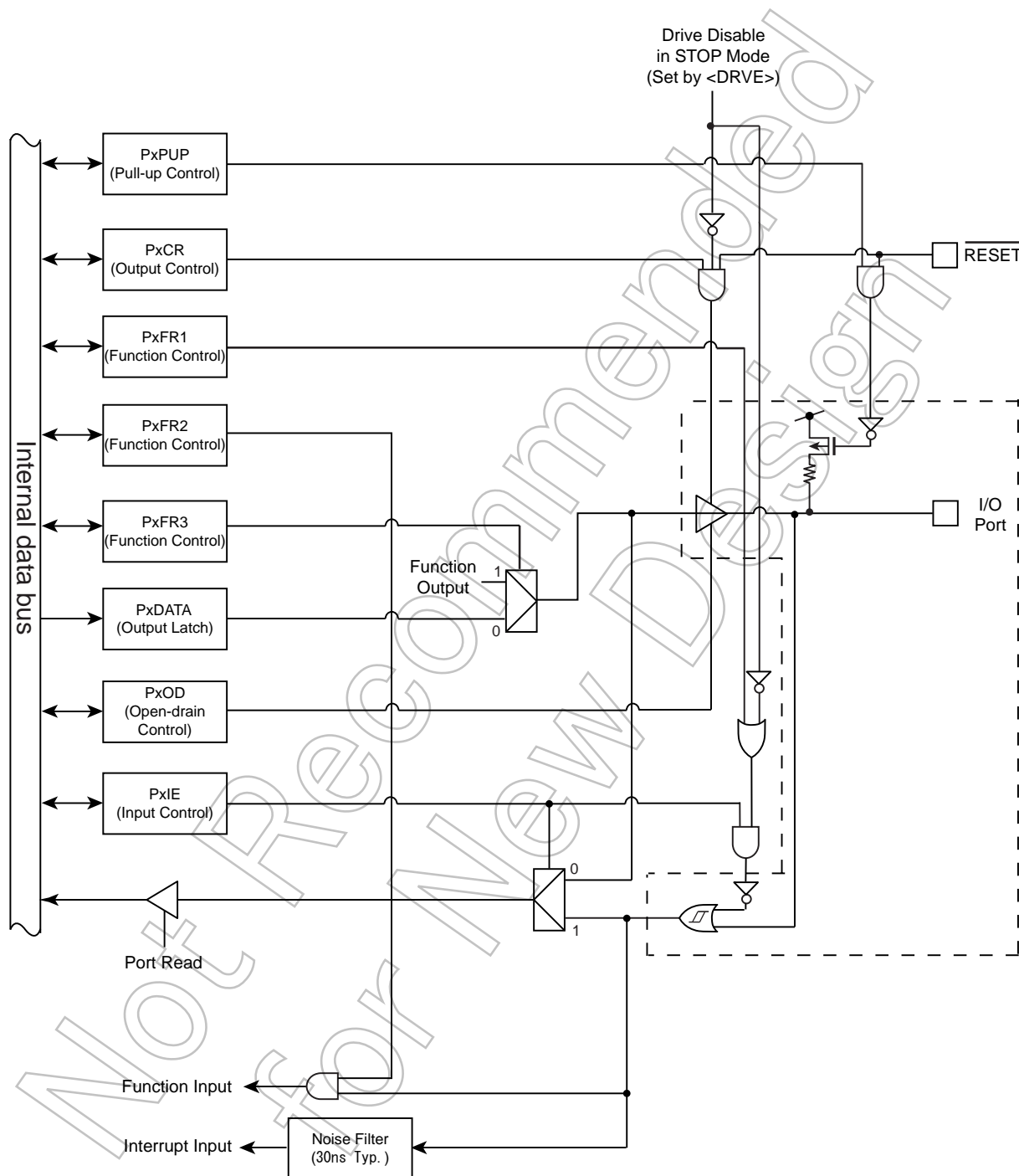
Figure 8-39 Port Type T39

8.4   Appendix (Port setting List)

The following table shows the register setting for each function.

Initialization of the ports where the o does not exist in the "After reset" field is set to "0" for all register settings.

Setting for the bit "×" can be arbitrarily-specified.

8.4.1   Port A setting

Table 8-4 Port Setting List (Port A)

Pin	Port Type	Function	After reset	PACR	PAFR1	PAOD	PAPUP	PAIE
PA0	T1	Input Port		0	0	×	×	1
		Output Port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PA1	T1	Input Port		0	0	×	×	1
		Output Port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PA2	T1	Input Port		0	0	×	×	1
		Output Port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PA3	T1	Input Port		0	0	×	×	1
		Output Port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PA4	T1	Input Port		0	0	×	×	1
		Output Port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PA5	T1	Input Port		0	0	×	×	1
		Output Port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PA6	T1	Input Port		0	0	×	×	1
		Output Port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PA7	T1	Input Port		0	0	×	×	1
		Output Port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1

8.4.2 Port B Setting

Table 8-5 Port Setting List (Port B)

Pin	Port Type	Function	After reset	PBCR	PBFR1	PBOD	PBPUP	PBIE
PB0	T1	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PB1	T1	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PB2	T1	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PB3	T1	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PB4	T1	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PB5	T1	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PB6	T1	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1
PB7	T1	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		Data (I/O) / Address data (I/O)		1	1	×	×	1

2023/07/21

8.4.3 Port E Setting

Table 8-6 Port Setting List (Port E)

Pin	Port Type	Function	After reset	PECR	PEFR1	PEFR2	PEFR3	PEOD	PEPUP	PEIE
PE0	T3	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		Address (Output)		1	1	0	0	×	×	0
		TB5IN0 (Input)		0	0	1	0	×	×	1
PE1	T3	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		Address (Output)		1	1	0	0	×	×	0
		TB5IN1 (Input)		0	0	1	0	×	×	1
PE2	T3	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		Address (Output)		1	1	0	0	×	×	0
		TB6IN0 (Input)		0	0	1	0	×	×	1
PE3	T3	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		Address (Output)		1	1	0	0	×	×	0
		TB6IN1 (Input)		0	0	1	0	×	×	1
PE4	T36	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		Address (Output)		1	1	0	0	×	×	0
		TXD0 (Output)		1	0	1	0	×	×	0
		CTXD (Output)		1	0	0	1	×	×	0
PE5	T37	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		Address (Output)		1	1	0	0	×	×	0
		RXD0 (Input)		0	0	1	0	×	×	1
		CRXD (Input		0	0	0	1	×	×	1
PE6	T4	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		Address (Output)		1	1	0	0	×	×	0
		SCLK0 (Input)		0	0	1	0	×	×	1
		SCLK0 (Output)		1	0	1	0	×	×	0
		$\overline{CTS0}$ (Input)		0	0	0	1	×	×	1
PE7	T6	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		INT5 (Input)		0	0	1	0	×	×	1
		SCOUT (Output)		1	0	0	1	×	×	0

8.4.4　Port F Setting

Table 8-7 Port Setting List (Port F)

Pin	Port Type	Function	After reset	PFCR	PFFR1	PFOD	PFPUP	PFIE
PF0	T7	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		TRACECLK (Output)		1	1	×	×	0
PF1	T7	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		TRACEDATA0/SWV (Output)		1	1	×	×	0
PF2	T7	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		TRACEDATA1 (Output)		1	1	×	×	0
PF3	T7	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		TRACEDATA2 (Output)		1	1	×	×	0
PF4	T7	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		TRACEDATA3 (Output)		1	1	×	×	0

8.4.5 Port G Setting

Table 8-8 Port Setting List (Port G)

Pin	Port Type	Function	After reset	PGCR	PGFR1	PGFR2	PGFR3	PGOD	PGPUP	PGIE
PG0	T8	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SO1 (Output)		1	1	0	0	×	×	0
		SDA1 (I/O)		1	1	0	0	1	×	1
		TB7IN0 (Input)		0	0	1	0	×	×	1
PG1	T8	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SI1 (Input)		0	1	0	0	×	×	1
		SCL1 (I/O)		1	1	0	0	1	×	1
		TB7IN1 (Input)		0	0	1	0	×	×	1
PG2	T9	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SCK1 (Input)		0	1	0	0	×	×	1
		SCK1 (Output)		1	1	0	0	×	×	0
		$\overline{CTS0}$ (Input)		0	0	0	1	×	×	1
PG3	T10	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		INT6 (Input)		0	1	0	0	×	×	1
		$\overline{CTS1}$ (Input)		0	0	0	1	×	×	1
PG4	T8	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SO2 (Output)		1	1	0	0	×	×	0
		SDA2 (I/O)		1	1	0	0	1	×	1
		TB9IN0 (Input)		0	0	1	0	×	×	1
PG5	T8	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SI2 (Input)		0	1	0	0	×	×	1
		SCL2 (I/O)		1	1	0	0	1	×	1
		TB9IN1 (Input)		0	0	1	0	×	×	1
PG6	T38	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SCK2 (Input)		0	1	0	0	×	×	1
		SCK2 (Output)		1	1	0	0	×	×	0
		USBPON (Output)		1	0	1	0	×	×	0
		$\overline{CTS3}$ (Input)		0	0	0	1	×	×	1
PG7	T39	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		INT7 (Input)		0	1	0	0	×	×	1
		$\overline{USBOC}$ (Input)		0	0	1	0	×	×	1
		$\overline{WDTOUT}$ (Output)		1	0	0	1	×	×	0

8.4.6   Port I Setting

Table 8-9 Port Setting List (Port I)

Pin	Port Type	Function	After reset	PICR	PIFR1	PIOD	PIPUP	PIIE
PI0	T14	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
PI1	T15	Input port		0	0	×	×	1
		Output port		1	0	×	×	0
		CEC (Input)		0	1	×	×	1

Note: The PI0 input and pull-up are enabled and act as $\overline{\text{BOOT}}$ input pin while a $\overline{\text{RESET}}$ is in "Low" state.

8.4.7   Port J Setting

Table 8-10 Port Setting List (Port J)

Pin	Port Type	Function	After reset	PJFR2	PJPUP	PJIE
PJ0	T17	Input port		0	×	1
		Analog input		0	0	0
PJ1	T17	Input port		0	×	1
		Analog input		0	0	0
PJ2	T17	Input port		0	×	1
		Analog input		0	0	0
PJ3	T18	Input port		0	×	1
		Analog input		0	0	0
		$\overline{\text{ADTRG}}$ (Input)		1	×	1
PJ4	T19	Input port		0	×	1
		Analog input		0	0	0
		KWUP0 (Input)		1	×	1
PJ5	T19	Input port		0	×	1
		Analog input		0	0	0
		KWUP1 (Input)		1	×	1
PJ6	T19	Input port		0	×	1
		Analog input		0	0	0
		KWUP2 (Input)		1	×	1
PJ7	T19	Input port		0	×	1
		Analog input		0	0	0
		KWUP3 (Input)		1	×	1

8.4.8 Port L Setting

Table 8-11 Port Setting List (Port L)

Pin	Port Type	Function	After reset	PLCR	PLFR1	PLFR2	PLFR3	PLOD	PLPUP	PLIE
PL0	T20	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SO0 (Output)		1	1	0	0	×	×	0
		SDA0 (I/O)		1	1	0	0	1	×	1
		TB0OUT(Output)		1	0	1	0	×	×	0
PL1	T20	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SI0 (Input)		0	1	0	0	×	×	1
		SCL0 (I/O)		1	1	0	0	1	×	1
		TB1OUT(Output)		1	0	1	0	×	×	0
PL2	T20	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SCK0 (Input)		0	1	0	0	×	×	1
		SCK0 (Output)		1	1	0	0	×	×	0
		TB2OUT(Output)		1	0	1	0	×	×	0
PL3	T21	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		INT0 (Input)		0	1	0	0	×	×	1
		TB3OUT(Output)		1	0	1	0	×	×	0
PL4	T22	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		TXD1 (Output)		1	1	0	0	×	×	0
		SDA3 (I/O)		1	0	0	1	1	×	1
		TB4OUT(Output)		1	0	1	0	×	×	0
PL5	T23	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		RXD1 (Input)		0	1	0	0	×	×	1
		SCL3 (I/O)		1	0	0	1	1	×	1
		TB5OUT(Output)		1	0	1	0	×	×	0
PL6	T24	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SCLK1 (Input)		0	1	0	0	×	×	1
		SCLK1 (Output)		1	1	0	0	×	×	0
		TB6OUT(Output)		1	0	1	0	×	×	0
		$\overline{\text{CTS1}}$ (Input)		0	0	0	1	×	×	1
PL7	T21	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		INT1 (Input)		0	1	0	0	×	×	1
		TB7OUT(Output)		1	0	1	0	×	×	0

8.4.9   Port M Setting

Table 8-12 Port Setting List (Port M)

Pin	Port Type	Function	After reset	PMCR	PMFR1	PMFR2	PMFR3	PMOD	PMPUP	PMIE
PM0	T25	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SCLK2 (Input)		0	1	0	0	×	×	1
		SCLK2 (Output)		1	1	0	0	×	×	0
		TB1IN0 (Input)		0	0	1	0	×	×	1
		CTS2 (Input)		0	0	0	1	×	×	1
PM1	T26	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		TXD2 (Output)		1	1	0	0	×	×	0
		TB1IN1 (Input)		0	0	1	0	×	×	1
PM2	T23	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		RXD2 (Input)		0	1	0	0	×	×	1
		ALARM (Output)		1	0	1	0	×	×	0
PM3	T21	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		INT2 (Input)		0	1	0	0	×	×	1
		TB3OUT(Output)		1	0	1	0	×	×	0
PM4	T27	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SCLK3 (Input)		0	1	0	0	×	×	1
		SCLK3 (Output)		1	1	0	0	×	×	0
		CTS3 (Input)		0	0	0	1	×	×	1
PM5	T28	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		TXD3 (Output)		1	1	0	0	×	×	0
PM6	T29	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		RXD3 (Input)		0	1	0	0	×	×	1
PM7	T30	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		INT3 (Input)		0	1	0	0	×	×	1

8.4.10 Port N setting

Table 8-13 Port Setting List (Port N)

Pin	Port Type	Function	After reset	PNCR	PNFR1	PNFR2	PNFR3	PNOD	PNPUP	PNIE
PN0	T28	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		TXD4 (Output)		1	1	0	0	×	×	0
PN1	T29	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		RXD2 (Input)		0	1	0	0	×	×	1
PN2	T25	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		SCLK4 (Input)		0	1	0	0	×	×	1
		SCLK4 (Output)		1	1	0	0	×	×	0
		TB2IN0 (Input)		0	0	1	0	×	×	1
		$\overline{CTS4}$ (Input)		0	0	0	1	×	×	1
PN3	T30	Input port		0	0	0	0	×	×	1
		Output port		1	0	0	0	×	×	0
		INT4 (Input)		0	1	0	0	×	×	1
		TB2IN1 (Input)		0	0	1	0	×	×	1
		RMC (Input)		0	0	0	1	×	×	1

8.4.11 Port P Setting

Table 8-14 Port Setting List (Port P)

Pin	Port Type	Function	After reset	PPCR	PPFR1	PPFR2	PPOD	PPPUP	PPIE
PP0	T5	Input port		0	0	0	×	×	1
		Output port		1	0	0	×	×	0
		$\overline{CS2}$ (Output)		1	1	0	×	×	0
PP1	T31	Input port		0	0	0	×	×	1
		Output port		1	0	0	×	×	0
PP2	T32	Input port		0	0	0	×	×	1
		Output port		1	0	0	×	×	0
		$\overline{BLS0}$ (Output)		1	1	0	×	×	0
		SPDO (Output)		1	0	1	×	×	0
PP3	T33	Input port		0	0	0	×	×	1
		Output port		1	0	0	×	×	0
		$\overline{BLS1}$ (Output)		1	1	0	×	×	0
		SPDI (Input)		0	0	1	×	×	1
PP4	T34	Input port		0	0	0	×	×	1
		Output port		1	0	0	×	×	0
		$\overline{WE}$ (Output)		1	1	0	×	×	0
		SPCLK (Input)		0	0	1	×	×	1
		SPCLK (Output)		1	0	1	×	×	0
PP5	T35	Input port		0	0	0	×	×	1
		Output port		1	0	0	×	×	0
		$\overline{OE}$ (Output)		1	1	0	×	×	0
		SPFSS (Input)		0	0	1	×	×	1
		SPFSS (Output)		1	0	1	×	×	0
PP6	T5	Input port		0	0	0	×	×	1
		Output port		1	0	0	×	×	0
		$\overline{ALE}$ (Output)		1	1	0	×	×	0

9. DMA Controller(DMAC)

9.1 Overview

The table below lists its major functions.

Table 9-1 DMA controller functions (1 Unit)

Item	Function		Description
Number of channels	2ch		-
Number of DMA request	16		-
DMA Start up trigger	Hardware start		Started with DMA request for peripheral circuit.
	Software start		Started with a write to the DMACxSoftBReq register.
Bus master	32bit × 1 (AHB)		-
Priority	High : CH0 Low : CH1		Fixed
FIFO	4word × 2ch (1word = 32bit)		-
Bus width	8/16/32bit		Settable individually for transfer source and destination.
Burst size	1/4/8/16/32/64/128/256		-
Number of transfers	up to 4095		-
Address	Transfer source address	increment not increment	It is possible to specify whether Source and Destination addresses should increment or should not increment. (Address wrapping is not supported.)
	Transfer destination address	increment not increment	
Endian	Littleendian is supported.		-
Transfer type	Peripheral to Memory Memory to Peripheral Memory to Memory Peripheral to Peripheral		When "Memory to Memory" is selected, hardware start for DMA startup is not supported. Refer to the DMACxCnConfiguration for more information. Particular peripheral can be assigned as Source or Distination when "Peripheral to Peripheral" is selected. Regarding to peripheral assigned, refer to "9.4.1 Peripheral function supported with Peripheral to Peripheral Transfer". TMPM363F10FG does not support Peripheral to Peripheral.
Interrupt function	Transfer end interrupt (INTDMACxTC) Error interrupt (INTDMACxERR)		-
Special Function	Scatter/gather function		-

9.2 DMA transfer type

Table 9-2 DMA transfer type

No.	DMA transfer type	Circuit generated DMA request	DMA request type	Description
1	Memory to Peripheral	Peripheral (Destination)	Burst request	In case of 1word transmission, set to the "1" for burst size of DMA controller.
2	Peripheral to Memory	Peripheral (Source)	Burst request / single request	If the amount of transfer data is not an integral multiple of the burst size, both burst and single request can be used. If amount of transfer data is more or equal than burst size, the single request is ignored and the burst transfer is used. If it becomes less than burst size, the single transfer is used.
3	Memory то Memory	DMAC	None	Enabling the DMAC starts data transfer without DMAC request. (Select Memory to Memory mode, set DMACxCnConfiguration<E> to "1") When All transfer data is transferred completely or when the DMAC channel is disabled, DMAC is stopped.

No.	DMA transfer type	Circuit generated DMA request	DMA request type	Transfer size	Source	Destination
4	Peripheral to Peripheral	Peripheral (Source)	Burst request / single request	(1)An integral multiple of the burst size	Burst request	Burst request
		Peripheral (Destination)	Burst request	(2)Not an integral multiple of the burst size	Burst request / single request	-

Note: When much data is transferred in memory to memory, we recommend that a lower priority channel is used. If a lower priority channel is used, a higher priority channel can be started to transfer during a lower priority channel is transferring. If a higher priority channel is used, a lower priority channel can not be started to transfer during a higher priority channel is transferring.
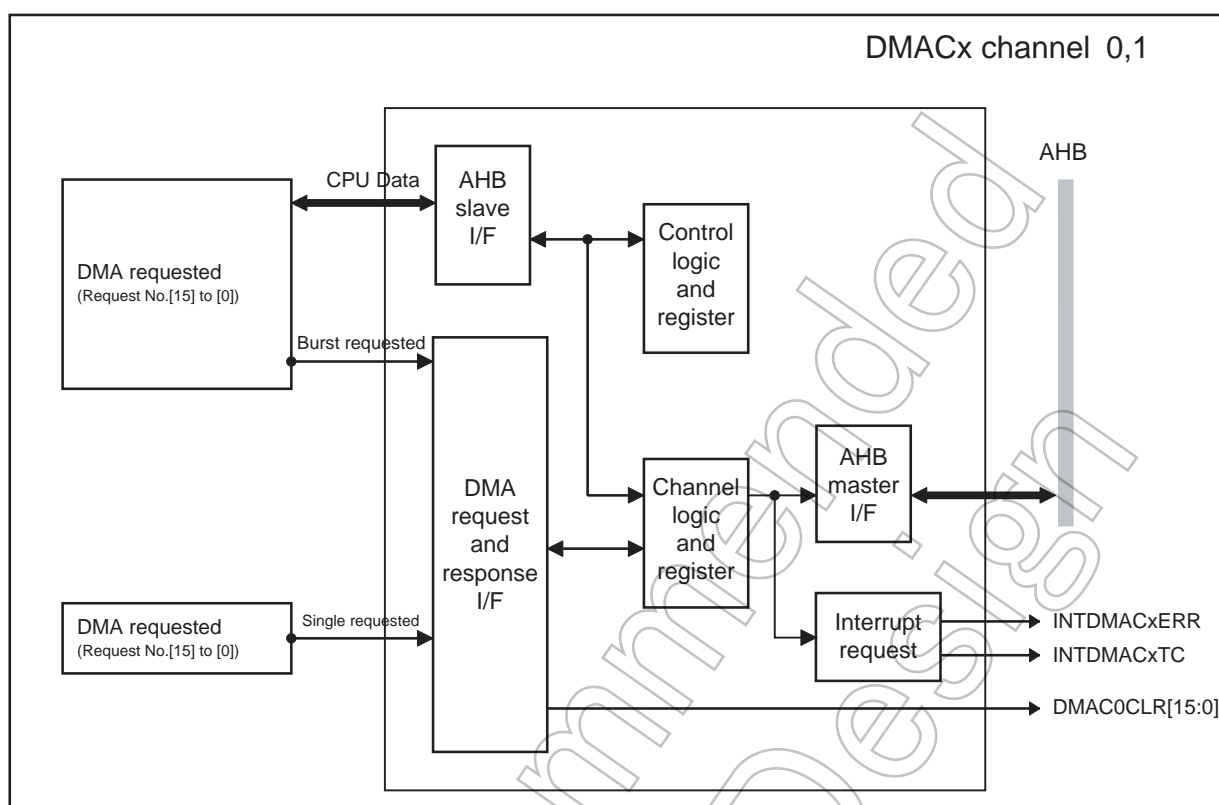
9.3   Block diagram

Figure 9-1 DMAC Block diagram

9.4   Product information of TMPM363F10FG

9.4.1   Peripheral function supported with Peripheral to Peripheral Transfer

Peripheral functions (Register) supported with Peripheral to Peripheral Transfer are shown below.

TMPM363F10FG does not support Peripheral to Peripheral Transfer.

9.4.2   DMA request

DMA request against each DMA request no. are shown as bellows.

Table 9-3 DMA request factor

DMA request No.	Corresponding peripheral	
	ch0,ch1	
	Burst request	Single request
0	SIO0/UART0 Transmission / Reception	–
1	SIO1/UART1 Transmission / Reception	–
2	SIO2/UART2 Transmission / Reception	–
3	SIO3/UART3 Transmission / Reception	–
4	SIO4/UART4 Transmission / Reception	–
5	–	–
6	–	–
7	–	–
8	–	–
9	–	–
10	–	–
11	–	–
12	SSP Transmission	–
13	SSP Reception	SSP Reception
14	–	–
15	Normal AD Conversion End	–

9.4.3   Interrupt request

Transfer complete interrupt	Error interrupt
INTDMACTC	INTDMACERR

9.4.4   Base address of registers

Base Address
0x4000_0000

9.5   Description of Registers

9.5.1   DMAC register list

The function and address for each register are shown bellow.

Register Name		Address (Base+)
DMAC Interrupt Status Register	DMACxIntStaus	0x0000
DMAC Interrupt Terminal Count Status Register	DMACxIntTCStatus	0x0004
DMAC Interrupt Terminal Count Clear Register	DMACxIntTCClear	0x0008
DMAC Interrupt Error Status Register	DMACxIntErrorStatus	0x000C
DMAC Interrupt Error Clear Register	DMACxIntErrClr	0x0010
DMAC Raw Interrupt Terminal Count Status Register	DMACxRawIntTCStatus	0x0014
DMAC Raw Error Interrupt Status Register	DMACxRawIntErrorStatus	0x0018
DMAC Enabled Channel Register	DMACxEnbldChns	0x001C
DMAC Software Burst Request Register	DMACxSoftBReq	0x0020
DMAC Software Single Request Register	DMACxSoftSReq	0x0024
Reserved	-	0x0028
Reserved	-	0x002C
DMAC Configuration Register	DMACxConfiguration	0x0030
Reserved	-	0x0034
DMAC Channel0 Source Address Register	DMACxC0SrcAddr	0x0100
DMAC Channel0 Destination Address Register	DMACxC0DestAddr	0x0104
DMAC Channel0 Linked List Item Register	DMACxC0LLI	0x0108
DMAC Channel0 Control Register	DMACxC0Control	0x010C
DMAC Channel0 Configuration Register	DMACxC0Configuration	0x0110
DMAC Channel1 Source Address Register	DMACxC1SrcAddr	0x0120
DMAC Channel1 Destination Address Register	DMACxC1DestAddr	0x0124
DMAC Channel1 Linked List Item Register	DMACxC1LLI	0x0128
DMAC Channel1 Control Register	DMACxC1Control	0x012C
DMAC Channel 1 Configuration Register	DMACxC1Configuration	0x0130

Note 1: Access the registers by using word (32bit) reads and word writes.

Note 2: Access to the "Reserved" area is prohibited.

Note 3: For the registers prepared for every channel, if the channel structure is the same, unit number is expressed as "x" and channel number is expresses as "n".

Note 4: When the register which is not assigned with an each channel is read after the register which is assigned with an each channel is written, one machine cycle is inserted between the instructions or read the register which is not assigned with an each channel twice.

9.5.2   DMACxIntStatus (DMAC Interrupt Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntStatus1	IntStatus0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

| Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntStatus1	R	Status of DMAC channel 1 transfer end interrupt. 0 : Interrupt not requested 1 : Interrupt requested Status of the DMAC interrupt generation after passing through the transfer end interrupt enable register and error interrupt enable register. An interrupt is requested when there is a transfer error or when the counter completes counting.
0	IntStatus0	R	Status of DMAC channel 0 interrupt generation. 0 : Interrupt not requested 1 : Interrupt requested Status of the DMAC interrupt generation after passing through the transfer end interrupt enable register and error interrupt enable register. An interrupt is requested when there is a transfer error or when the counter completes counting.
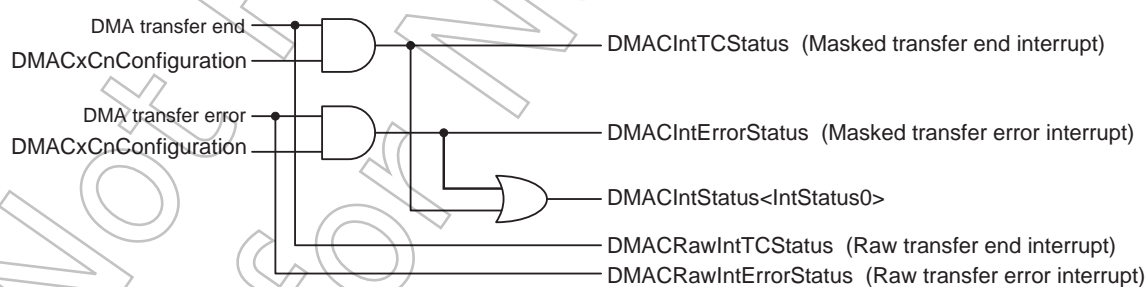
Figure 9-2 Interrupt-related block diagram

9.5.3 DMACxIntTCStatus (DMAC Interrupt Terminal Count Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntTCStatus1	IntTCStatus0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntTCStatus1	R	Status of DMAC channel 1 transfer end interrupt. 0 : Interrupt not requested 1 : Interrupt requested The status of post-enable transfer end interrupt generation.
0	IntTCStatus0	R	Status of DMAC channel 0 transfer end interrupt. 0 : Interrupt not requested 1 : Interrupt requested The status of post-enable transfer end interrupt generation.

9.5.4   DMACxIntTCClear (DMAC Interrupt Terminal Count Clear Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntTCClear1	IntTCClear0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntTCClear1	W	Clear DMAC channel 1 transfer end interrupt. 0 : Do nothing 1 : Clear The DMACxIntTCStatus<IntTCStatus1> will be cleared when "1" is written.
0	IntTCClear0	W	Clear DMAC channel 0 transfer end interrupt. 0 : Do nothing 1 : Clear The DMACxIntTCStatus<IntTCStatus0> will be cleared when "1" is written.

9.5.5 DMACxIntErrorStatus (DMAC Interrupt Error Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntErrStatus1	IIntErrStatus0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntErrStatus1	R	Status of DMAC channel 1 error interrupt generation. 0 : Interrupt not requested 1 : Interrupt requested Shows error interrupt status after enabled.
0	IntErrStatus0	R	Status of DMAC channel 0 error interrupt generation. 0 : Interrupt not requested 1 : Interrupt requested Shows error interrupt status after enabled.

9.5.6   DMACxIntErrClr (DMAC Interrupt Error Clear Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntErrClr1	IntErrClr0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntErrClr1	W	Clear DMAC channel 1 transfer end interrupt. 0 : Do nothing 1 : Clear The DMACxIntErrorStatus<IntErrStatus1> will be cleared when "1" is written.
0	IntErrClr0	W	Clear DMAC channel 0 transfer end interrupt. 0 : Do nothing 1 : Clear The DMACxIntErrorStatus<IntErrStatus0> will be cleared when "1" is written.

9.5.7 DMACxRawIntTCStatus (DMAC Raw Interrupt Terminal Count Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	RawIntTCS1	RawIntTCS0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	RawIntTCS1	R	Status of DMAC channel 1 pre-enable transfer end interrupt generation 0 : Interrupt not requested 1 : Interrupt requested
0	RawIntTCS0	R	Status of DMAC channel 0 pre-enable transfer end interrupt generation 0 : Interrupt not requested 1 : Interrupt requested

9.5.8　DMACxRawIntErrorStatus (DMAC Raw Error Interrupt Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	RawIntErrS1	RawIntErrS0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	RawIntErrS1	R	Status of DMAC channel 1 pre-enable error interrupt. 0 : Interrupt not requested 1 : Interrupt requested
0	RawIntErrS0	R	Status of DMAC channel 0 pre-enable error interrupt. 0 : Interrupt not requested 1 : Interrupt requested

9.5.9 DMACxEnbldChns (DMAC Enabled Channel Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	EnabledCH1	EnabledCH0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	EnabledCH1	R	DMA channel 1 enable status. 0 : Disable 1 : Enable After finishing all the total transfer number of times in DMACxCnControl register (the value becomes the zero), the this flag is clearded.
0	EnabledCH0	R	DMA channel 0 enable status. 0 : Disable 1 : Enable After finishing all the total transfer number of times in DMACxCnControl register (the value becomes the zero), the this flag is clearded.

9.5.10　DMACxSoftBReq (DMAC Software Burst Request Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	SoftBReq15	SoftBReq14	SoftBReq13	SoftBReq12	SoftBReq11	SoftBReq10	SoftBReq9	SoftBReq8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SoftBReq7	SoftBReq6	SoftBReq5	SoftBReq4	SoftBReq3	SoftBReq2	SoftBReq1	SoftBReq0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	-	Write as zero.
15	SoftBReq15	R/W	DMA burst request by software (Request No. [15]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
14	SoftBReq14	R/W	DMA burst request by software (Request No. [14]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
13	SoftBReq13	R/W	DMA burst request by software (Request No. [13]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
12	SoftBReq12	R/W	DMA burst request by software (Request No. [12]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
11	SoftBReq11	R/W	DMA burst request by software (Request No. [11]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
10	SoftBReq10	R/W	DMA burst request by software (Request No. [10]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
9	SoftBReq9	R/W	DMA burst request by software (Request No. [9]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested

Bit	Bit Symbol	Type	Function
8	SoftBReq8	R/W	DMA burst request by software (Request No. [8]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
7	SoftBReq7	R/W	DMA burst request by software (Request No. [7]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
6	SoftBReq6	R/W	DMA burst request by software (Request No. [6]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
5	SoftBReq5	R/W	DMA burst request by software (Request No. [5]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
4	SoftBReq4	R/W	DMA burst request by software (Request No. [4]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
3	SoftBReq3	R/W	DMA burst request by software (Request No. [3]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
2	SoftBReq2	R/W	DMA burst request by software (Request No. [2]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
1	SoftBReq1	R/W	DMA burst request by software (Request No. [1]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
0	SoftBReq0	R/W	DMA burst request by software (Request No. [0]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested

Note 1: Do not execute DMA requests by software and hardware at the same time.

Note 2: Refer to "9.4.2 DMA request" for DMA request number. Clear "0" to bit corresponded with the DMA request number which has no burst request.

9.5.11　DMACxSoftSReq (DMAC Software Single Request Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	SoftSReq15	SoftSReq14	SoftSReq13	SoftSReq12	SoftSReq11	SoftSReq10	SoftSReq9	SoftSReq8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SoftSReq7	SoftSReq6	SoftSReq5	SoftSReq4	SoftSReq3	SoftSReq2	SoftSReq1	SoftSReq0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	-	Write as zero.
15	SoftSReq15	R/W	DMA single request by software (Request No. [15]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
14	SoftSReq14	R/W	DMA single request by software (Request No. [14]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
13	SoftSReq13	R/W	DMA single request by software (Request No. [13]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
12	SoftSReq12	R/W	DMA single request by software (Request No. [12]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
11	SoftSReq11	R/W	DMA single request by software (Request No. [11]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
10	SoftSReq10	R/W	DMA single request by software (Request No. [10]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
9	SoftSReq9	R/W	DMA single request by software (Request No. [9]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
8	SoftSReq8	R/W	DMA single request by software (Request No. [8]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested

Bit	Bit Symbol	Type	Function
7	SoftSReq7	R/W	DMA single request by software (Request No. [7]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
6	SoftSReq6	R/W	DMA single request by software (Request No. [6]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
5	SoftSReq5	R/W	DMA single request by software (Request No. [5]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
4	SoftSReq4	R/W	DMA single request by software (Request No. [4]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
3	SoftSReq3	R/W	DMA single request by software (Request No. [3]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
2	SoftSReq2	R/W	DMA single request by software (Request No. [2]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
1	SoftSReq1	R/W	DMA single request by software (Request No. [1]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
0	SoftSReq0	R/W	DMA single request by software (Request No. [0]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested

Note 1: Do not execute DMA requests by software and hardware at the same time.

Note 2: Refer to "9.4.2 DMA request" for DMA request number. Clear "0" to bit corresponded with the DMA request number which has no single request.

9.5.12   DMACxConfiguration (DMAC Configuration Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	M	E
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	M	R/W	Write as zero.
0	E	R/W	DMA circuit control 0 : Stop 1 : Operate When circuit stops, the registers for the DMA circuit cannot be written or read. When operating the DMA, always set <E>="1".

9.5.13 DMACxCnSrcAddr (DMAC Channelx Source Address Register)

	31	30	29	28	27	26	25	24
bit symbol	SrcAddr							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	SrcAddr							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	SrcAddr							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SrcAddr							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	SrcAddr[31:0]	R/W	Sets a DMA transfer source address. Make sure to confirm the source address and the bit width before setting. The below are the restrictions in setting of source address bit width.

Source address bit width DMACxCnControl<Swidth[2:0]>	Setting of least significant address
000 : Byte (8 bits)	no restriction
001 : Half word (16 bits)	Setting as multiples of 2, (0x0,0x02,0x4,0x06,0x8,0xA,0xC…)
010 : Word (32 bits)	Setting as multiples of 4, (0x0,0x4,0x8,0xC…)

Because enabling channel "n" (DMACxCnConfiguration<E>="1") updates the data written in the registers, set DMACxCnSrcAddr before enabling the channels.

When the DMA is operating, the value in the DMACxCnSrcAddr register sequentially changes, so the read values are not fixed.

And do not update DMACxCnSrcAddr during transfer. To change DMACxCnSrcAddr, be sure to disable the channel "n" (DMACxCnConfiguration<E>="0") before change.

9.5.14   DMACxCnDestAddr (DMAC Channelx Destination Address Register)

	31	30	29	28	27	26	25	24
bit symbol	DestAddr							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	DestAddr							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	DestAddr							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DestAddr							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	DestAddr[31:0]	R/W	Sets a DMA transfer destination address. Make sure to confirm the destination address and the bit width before setting. The below are the restrictions in setting of destination address bit width. table below

Destination address bit width DMACxCControl<Dwidth[2:0]>	Setting of least significant address
000 : Byte (8 bits)	no restriction
001 : Half word (16 bits)	Setting as multiples of 2, (0x0,0x02,0x4,0x06,0x8,0xA,0xC…)
010 : Word (32 bits)	Setting as multiples of 4, (0x0,0x4,0x8,0xC…)

Do not update DMACxCnDestAddr during transfer. To change DMACxCnDestAddr, be sure to disable the channel "n" (DMACxCnConfiguration<E>="0") before change.

9.5.15 DMACxCnLLI (DMAC Channelx Linked List Item Register)

	31	30	29	28	27	26	25	24
bit symbol	LLI							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	LLI							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	LLI							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	LLI						-	-
After reset	0	0	0	0	0	0	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-2	LLI[29:0]	R/W	Sets the first address of the next transfer information. Set a value smaller than 0xFFFF_FFF0. When <LLI> = 0, LLI is the last chain. After DMA transfer finishes, the DMA channel is disabled.
1-0	-	R/W	Write as zero.

Note:For <LLI> detailed operation, see "9.6 Special Functions".

9.5.16   DMACxCnControl (DMAC Channeln Control Register)

	31	30	29	28	27	26	25	24
bit symbol	I	-	-	-	DI	SI	-	-
After reset	0	Undefined	Undefined	Undefined	0	0	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	Dwidth			Swidth			DBSize	
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	DBSize	SBSize			TransferSize			
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TransferSize							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	I	R/W	Register for enabling a transfer interrupt. 0 : Disable 1 : Enable The transfer end interrupt is generated by setting <I>="1" and DMACxCnConfiguration<ITC>="1". When the scatter/gather function is used in the last transfer DMAC setting flow and by setting this bit to enable, to generate the transfer end interrupt is enable only at the last transfer. To generate interrupt during normal transfer, set this bit to "1" and change to enable mode.
30-28	-	W	Write as zero.
27	DI	R/W	Increment the transfer destination address 0 : Do not increment 1 : Increment
26	SI	R/W	Increment the transfer source address 0 : Do not increment 1 : Increment
25-24	-	W	Write as zero.
23-21	Dwidth[2:0]	R/W	Transfer destination bit width. 000 : Byte (8 bits) 001 : Half-word (16 bits) 010 : Word (32 bits) other: Reserved Refer to Table 9-4 for the setting value.
20-18	Swidth[2:0]	R/W	Transfer source bit width 000: Byte (8 bits) 001: Half-word (16 bits) 010 : Word (32 bits) other: Reserved Refer to Table 9-4 for the setting value.
17-15	DBSize[2:0]	R/W	Transfer destination burst size: (Note 1) 000: 1 beat 100: 32 beats 001: 4 beats 101: 64 beats 010: 8 beats 110: 128 beats 011: 16 beats 111: 256 beats Refer to Table 9-4 for the setting value.

Bit	Bit Symbol	Type	Function
14-12	SBSize[2:0]	R/W	Transfer source burst size: (Note 1) 000: 1 beat 100: 32 beats 001: 4 beats 101: 64 beats 010: 8 beats 110: 128 beats 011: 16 beats 111: 256 beats Refer to Table 9-4 for the setting value.
11-0	TransferSize[11:0]	R/W	Set the total number of transfers. Amount of transfer data per a bit width specified by transfer source bit witdth (4 bytes / 2 bytes / 1byte) is set into <TransderSize[11:0]>. Because the burst size shows amount of transfer data at every DMA request internally, amount of transfer data is never changed even if any burst size is set when transfer source bit width and total number of transfers are not changed. The value of <TransferSize[11:0]> is decremented to 0 by the DMA transferring. If this is read, the value which is the number of data not to transfer. The total number of transfers is used as the unit for the transfer source bit width. For examples: When <Swidth>="000" (8bit), the number of transfers is expressed in the units of byte. When <Swidth>="001" (16bit), the number of transfers is expressed in the units of half word. When <Swidth>="010" (32bit), the number of transfers is expressed in the units of word.

Note: The burst size to be set with DBsize and SBsize has no connections with the HBURST for the AHB bus.

Table 9-4 How to decide the value of <Dwidth[2:0]>,<Swidth[2:0]>, <DBSize[2:0]>, <SBSize[2:0]>

<Dwidth[2:0]> / <Swidth[2:0]>	Set the number so that the following expression is satisfied: Transfer source bit width × Total number of transfers = Transfer destination bit width × N (N : Integer number) (ex.1) Bit width of transfer source:8 bit, bit width of transfer destination:32 bit, total number of transfers:25 times 8 bit × 25 times = 200 bit (25 byte) N = 200 ÷ 32 = 6.25 word Since 6.25 is not an integer number, the above setting is invalid. If the transfer source bit width is smaller than the transfer destination bit width, care must be taken when setting the total number of transfers. (ex.2) Bit width of transfer source :32 bit, bit width of transfer destination:16 bit, total number of transfers: 13 times 32 bit × 13 times = 416 bit (13 word) N = 416 ÷ 16 = 26 half_word Since 26 is an integer number, the above setting is valid.
<DBSize[2:0]> / <SBSize[2:0]>	When "Peripheral to Memory" or "Memory to Peripheral" is performed, peripheral circuits generates DMA request signal to indicate the preparation is ready. This signal triggers to execute data transfers. (In the case of "Memory to Memory", only software start is used.) Set the burst size to define the amount of data transferred from peripherals per DMA request signal. This register is used with FIFO buffer that can be contained multiple data.

9.5.17   DMACxCnConfiguration (DMAC Channel n Configuration Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	Halt	Active	Lock
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ITC	IE	FlowCntrl			-	DestPeripheral	
After reset	0	0	0	0	0	Undefined	0	0
	7	6	5	4	3	2	1	0
bit symbol	DestPeripheral		-	SrcPeripheral				E
After reset	0	0	Undefined	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-19	-	W	Write as zero.
18	Halt	R/W	Controls accepting a DMA request 0 : Accept a DMA request 1 : Ignore a DMA request
17	Active	R	Indicates whether data is present in the channel FIFO. 0 : No data exists in the FIFO 1 : Data exists in the FIFO
16	Lock	R/W	Sets a locked transfer (Non-divided transfer). 0 : Disable locked transfer 1: Enable locked transfer When locked transfer is enabled, as many burst transfers as specified are consecutively executed without releasing the bus. For detailed operation, see "9.6 Special Functions".
15	ITC	R/W	Transfer end interrupt enable register. 0 : Disable interrupt 1 : Enable interrupt
14	IE	R/W	Error interrupt enable register 0 :Disable interrupt 1: Enable interrupt
13-11	FlowCntrl[2:0]	R/W	Sets transfer method <table><tr><td><FlowCntrl[2:0]> setting value</td><td>Transfer method</td></tr><tr><td>000:</td><td>Memory to Memory (Note)</td></tr><tr><td>001:</td><td>Memory to Peripheral</td></tr><tr><td>010:</td><td>Peripheral to Memory</td></tr></table>011 to 111: Reserved
10	-	W	Write as zero.
9-6	DestPeripheral [3:0]	R/W	Sets transfer destination peripheral (Note 2) Refer to "9.4.2 DMA request". When a memory is the transfer destination, this setting is ignored.
5	-	W	Write as zero.
4-1	SrcPeripheral [3:0]	R/W	Sets transfer source peripheral (Note 2) Refer to "9.4.2 DMA request". When a memory is the transfer source, this setting is ignored.

Bit	Bit Symbol	Type	Function
0	E	R/W	Channel enable 0 : Disable 1 : Enable This bit can be used to enable/disable the channels. (This bit works as start bit when "Memory to Memory" is selected.) Amount of transfer data specified by DMACxCnControl <TransferSize> is completed, the corresponding <E> is cleared to "0" automatically. Disabling channels during transfer loses the data in the FIFO. Initialize all the channels before restart. To pause the transfer, stop the DMA request by using the <HALT>, and poll the data until the <Active> becomes "0" and then disable the channel with the <E> bit.

Note 1: When "Memory to Memory" is selected, hardware start for DMA startup is not supported. Write "1" <E> for starting transfer.

Note 2: When DMACxENableChns<EnabledCHx> is enabled and the corresponding DMACxCnConfiguration<Halt> is set to "1", write them after channel enable bit (E:bit0) is clear to "0". Without this, in the case of the slave error is occurred when writing them, the error is recovered by reset. Regarding slave error, when the width and address of transfer have mismatch, this error is occurred.

9.6   Special Functions

9.6.1   Scatter/gather function

When removing a part of image data and transferring it, image data cannot be handled as consecutive data, and the address changes dramatically depending on the special rule. Since DMA can transfer data only by using consecutive addresses, it is necessary to make required settings at locations where addresses changes.

A part of screen image is cut out.

Addresses are not continuous.

Screen image   Screen data
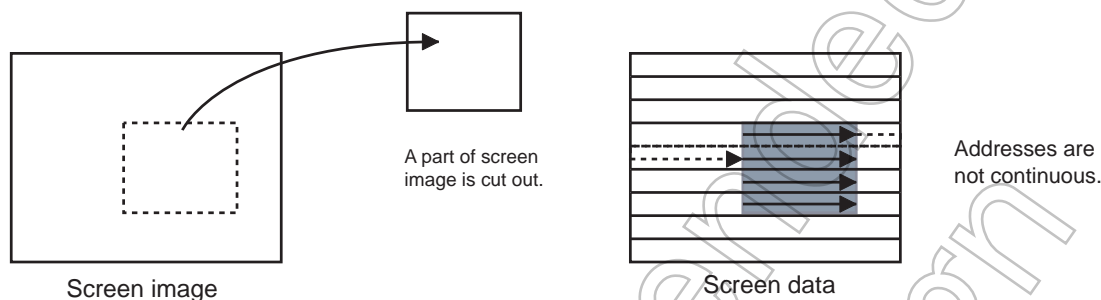
The scatter/gather function can consecutively operate DMA settings (transfer source address, destination address, number of transfers, and transfer bus width) by re-loading them each time a specified number of DMA executions have completed via a pre-set "Linked List" where the CPU does not need to control the operation.

Setting "1" in the DMACxCnLLI register enables/disables the operation.

The items that can be set with Linked List are configured with the following 4 words:

1.   DMACxCnSrcAddr

2.   DMACxCnDestAddr

3.   DMACxCnLLI

4.   DMACxCnControl

They can be used with the interrupt operation.

An interrupt depends on the count end interrupt enable bit of the DMACxCnControl register, and can be generated at the end of each LLI. When this bit is used, a condition can be added even during transfer using LLI to perform branch operation, etc. To clear the interrupt, control the appropriate bit of the DMACxIntTCClear register.

9.6.2 Linked list operation

To operate the scatter/gather function, a transfer source and source data areas need to be defined by creating a set of Linked Lists first.

Each setting is called LLI (LinkedList).

Each LLI controls the transfer of one block of data. Each LLI indicates normal DMA setting and controls transfer of successive data. Each time each DMA transfer is complete, the next LLI setting will be loaded to continue the DMA operation (Daisy Chain).

An example of the setting is shown below.

1. The first DMA transfer setting should be made directly in the DMA register.

2. The second and subsequent DMA transfer settings should be written in the addresses of the memory set in "next LLI AddressX."

3. To stop up to N'th DMA transfer, set "next LLI AddressX" to 0x0000_0000.
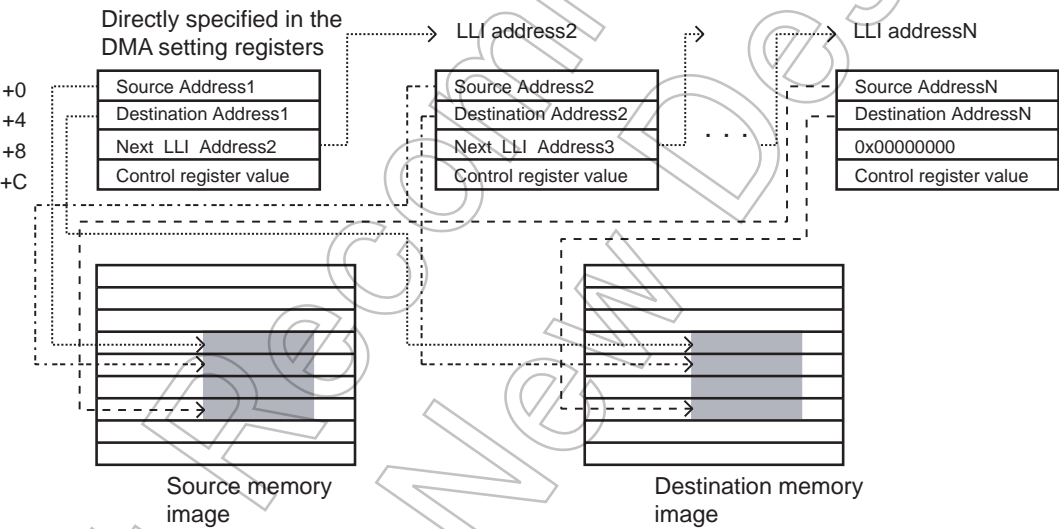
When transferring data in the area enclosed by the square

	Setting register	Setting parameter
+0	DMACxCnSrcAddr	:0x0A200
+4	DMACxCnDestAddr	:Destination address 1
+8	DMACxCnLLI	:0x200000
+C	DMACxCnControl	:Set the number of burst transfers and the number of transfers, etc.

Linked List

0x200000	0x0B200(SrcAddr)
+4	Dest Addr2
+8	0x200010
+C	Control register value

0x200010	0x0C200(SrcAddr)
+4	Dest Addr3
+8	0x00000000
+C	Control register value

←Indicates that a sequence of transfers ends with this LLI.
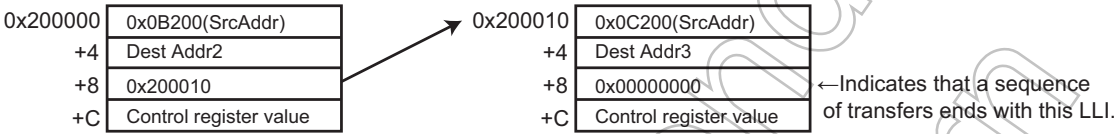
10. Static Memory Controller

TMPM363F10FG contains static memory (NOR type Flash memory and SRAM) controller with asynchronous access.

Note 1: Execute the WFI instruction after confirming the external memory access is completed.

Note 2: The external memory can not be used as a FIFO becase the dummy read cycle in the reading cycle from an external bus may be occurred.

10.1 Function Overview

Outline function is shown in Table 10-1.

Table 10-1 Out line function of Static memory Controller

Item	
Supported Memory type and bus connection	Asynchronous access memory (NOR Flash memory, SRAM, etc.) Multiplex bus supported
Data bus width	16bit data bus width
Memory map	64MB access area is supported and divide into four CS signal and area. CS0 : 0x6000_0000 to 0x60FF_FFFF (16MB) CS1 : 0x6100_0000 to 0x61FF_FFFF (16MB) CS2 : 0x6200_0000 to 0x62FF_FFFF (16MB) CS3 : 0x6300_0000 to 0x63FF_FFFF (16MB)
Timing adjustment	Can be controlled AC timing by registers.
Clock (SMCCLK)	fsys / 2
External control signals	Multiplex bus : AD0 to AD15, A17 to A23, $\overline{OE}$, $\overline{WE}$, $\overline{ALE}$, $\overline{CS0}$ to $\overline{CS3}$, $\overline{BLS0}$, $\overline{BLS1}$

10.2 Block diagram

The block diagram of SMC is shown as below.
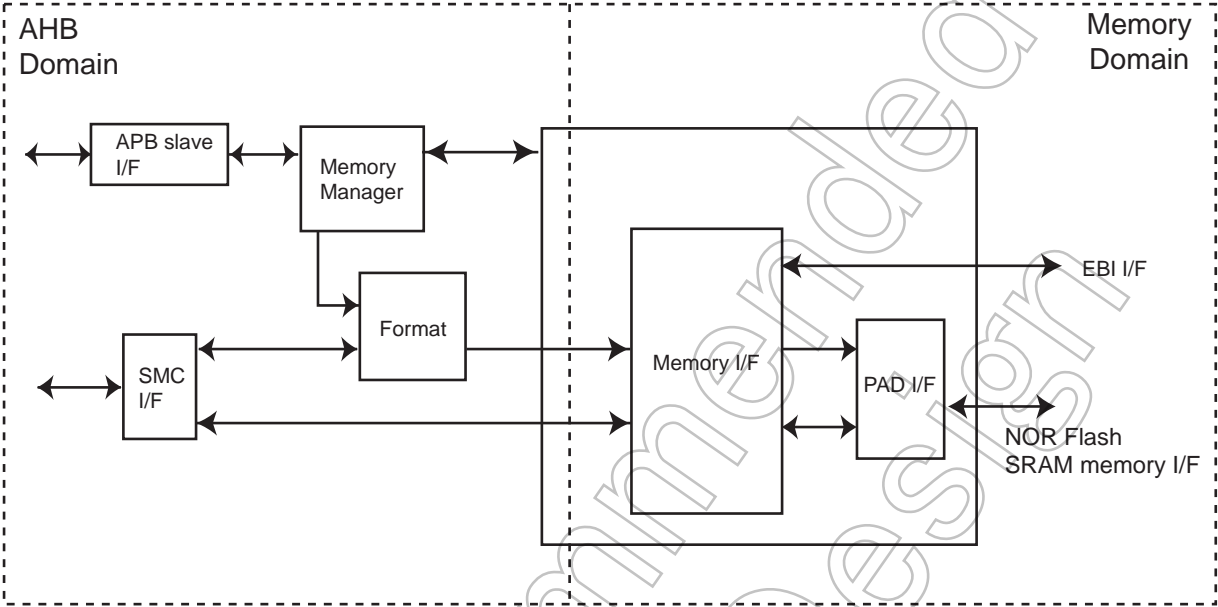
Figure 10-1 SMC Block Diagram

10.3 Description of Registers

10.3.1 SFR List

The following lists the SFRs.

Base Address = 0x4000_1000

Register name		Address (Base+)
Reserved	-	0x0000
SMC Memory Interface Configuration Register	smc_memif_cfg	0x0004
Reserved	-	0x0008
Reserved	-	0x000C
SMC Direct Command Register	smc_direct_cmd	0x0010
SMC Set Cycles Register	smc_set_cycles	0x0014
SMC Set Opmode Register	smc_set_opmode	0x0018
Reserved	-	0x0020
SMC SRAM Cycles Registers <0>	smc_sram_cycles0_0	0x0100
SMC Opmode Registers <0>	smc_opmode0_0	0x0104
SMC SRAM Cycles Registers <1>	smc_sram_cycles0_1	0x0120
SMC Opmode Registers <1>	smc_opmode0_1	0x0124
SMC SRAM Cycles Registers <2>	smc_sram_cycles0_2	0x0140
SMC Opmode Registers <2>	smc_opmode0_2	0x0144
SMC SRAM Cycles Registers <3>	smc_sram_cycles0_3	0x0160
SMC Opmode Registers <3>	smc_opmode0_3	0x0164
Reserved	-	0x0200 to 0x0204,0x0E00 to 0x0E08,0x0FE0 to 0x0FFC

Base Address = 0x41FF_F100

Register name		Address (Base+)
SMC Mode Register	SMCMDMODE	0x0000

Note 1: **Access the registers by using word reads and word writes.**

Note 2: **Do not access at reserved address.**

10.3.2   SMCMDMODE (Mode Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	IFSMC MUXMD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-1	−	R/W	Write as "0".
0	IFSMCMUXMD	R/W	SMC memory bus mode setting 1:Multiplex bus mode

Note 1: Do not change <IFSMCMUXMD> during SMC operation.

Note 2: Set <IFSMCMUXMD> to "1".

10.3.3   smc_memif_cfg (SMC Memory Interface Configuration Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	memory_width		memory_chips		memory_type	
After reset	Undefined	Undefined	0	1	1	1	0	1

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as undefined.
5-4	memory_width [1:0]	R	Maximum external SMC memory bus width 01 : 16 bits Others : Don't care
3-2	memory_chips [1:0]	R	The number of supported memory CS 00 : 1 chip 01 : 2 chip 10 : 3 chip 11 : 4 chip
1-0	memory_type [1:0]	R	Supported memory types : SRAM When <IFSMCMUXMD> is "1", read as "11". Others : Don't care

10.3.4 smc_direct_cmd (SMC Direct Command Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	chip_select	
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	chip_select	cmd_type		-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-26	−	W	Write as "0".
25-23	chip_select[2:0]	W	CS selection 000 : CS0 001 : CS1 010 : CS2 011 : CS3 100 to 111 : Setting prohibition Select objective Chip Select terminal
22-21	cmd_type[1:0]	W	Update set_opmode register and set_cycles register value 10 : Update registers Others : Setting prohibition
20-0	−	W	Write as "0".

```
Start
  |
  v
Set smc_set_cycle register as timing parameter
and set smc_set_opmode as operation mode
  |
  v
Select the external ChipSelect and set
smc_direct_cmd register then updating
  |
  v
End
```
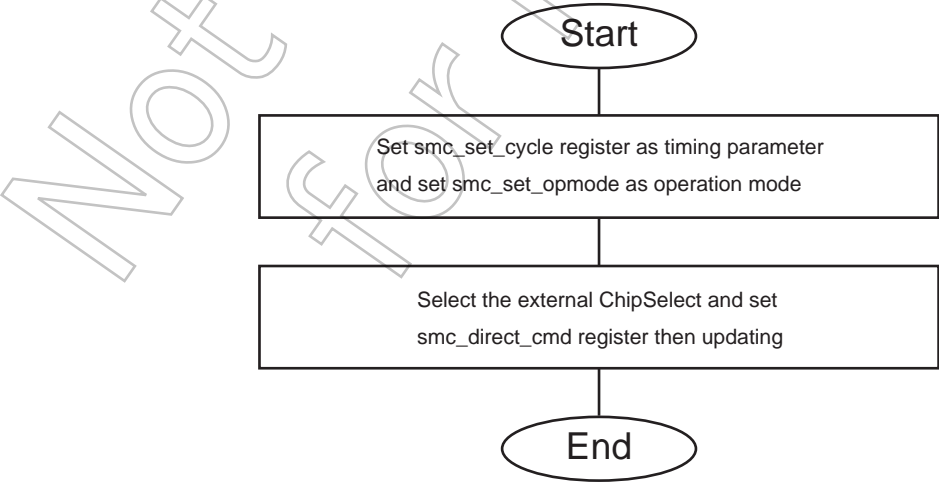
10.3.5   smc_set_cycles (SMC Set Cycles Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	Set_t5			Set_t4
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	Set_t4		Set_t3			Set_t2		
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	Set_t1				Set_t0			
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-20	–	W	Write as "0".
19-17	Set_t5[2:0]	W	Set value of t_{TR} 000 : Setting prohibition 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
16-14	Set_t4[2:0]	W	Set value of t_{PC} 000 : Setting prohibition 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock Page access is not supported in multiplex bus mode. t_{PC} is effective only separate bus mode.
13-11	Set_t3[2:0]	W	Set value of t_{WP} (note) 000 : Setting prohibition 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock In multiplex mode, write pulse width (t_{WP}) increase for one more clock pulse against for value of <Set_t3>.
10-8	Set_t2[2:0]	W	Set value of t_{CEOE} (note) 000 : Setting prohibition 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
7-4	Set_t1[3:0]	W	Set value of t_{WC} (note) 0000 : Setting prohibition 0011 to 1111 : SMCCLK × 3 clock to SMCCLK × 15 clock
3-0	Set_t0[3:0]	W	Set value of t_{RC} (note) 0000 : Setting prohibition 0010 to 1111 : SMCCLK × 2 clock to SMCCLK × 15 clock

This register is provided to adjust the access cycle of static memory and should be set to satisfy the A.C. specifications of the memory to be used. Adjust base clock is SMCCLK : fsys/2.

To validate SMC set cycles register setting, it is necessary to execute update register command on smc_direct_cmd register.

Note: It needs to keep below relation.

<Set_t3>,<Set_t1>	Multiplex bus mode :(t_{WP} + SMCCLK × 3 clock) ≤ t_{WC}
<Set_t2>,<Set_t0>	Multiplex bus mode :(t_{CEOE} + SMCCLK × 1 clock) ≤ t_{RC}

10.3.6 smc_set_opmode (SMC Set Opmode Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	set_adv	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	set_rd_bl			-	set_mw	
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-12	–	W	Write as "0".
11	set_adv	W	ALE signal 1 : Address latch enable ($\overline{\text{ALE}}$) used (select when multiplex bus mode is used.)
10-6	–	W	Write as "0".
5-3	set_rd_bl[2:0]	W	Setting bits for Burst length of data read 000 : 1 beat 001 : 4 beats Others : Reserved
2	–	W	Write as "0".
1-0	set_mw[1:0]	W	Holding register of the memory data bus width set value 01 : 16 bits Others : Reserved Setting bits for data bus width.

To validate SMC set opmode register settings, it is necessary to execute update register command on smc_direct_cmd register.

Note: Set <set_adv> to "1".

10.3.7   smc_sram_cycles0_0 (SMC SRAM Cycles Registers 0 <0>)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-		t_tr		-
After reset	Undefined	Undefined	Undefined	Undefined	0	0	1	0
	15	14	13	12	11	10	9	8
bit symbol	-	-		t_wp			t_ceoe	
After reset	1	0	1	1	0	0	1	1
	7	6	5	4	3	2	1	0
bit symbol		t_wc				t_rc		
After reset	1	1	0	0	1	1	0	0

Bit	Bit Symbol	Type	Function
31-20	−	W	Write as "0".
19-17	t_tr[2:0]	R	Turn around cycle time 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
16-14	−	R	Read as undefined.
13-11	t_wp[2:0]	R	$\overline{WE}$ pulse cycle time 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
10-8	t_ceoe[2:0]	R	Delay cycle time to $\overline{OE}$ assert 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
7-4	t_wc[3:0]	R	Write cycle time 0011 to 1111 : SMCCLK × 3 clock to SMCCLK × 15 clock
3-0	t_rc[3:0]	R	Read cycle time 0010 to 1111 : SMCCLK × 2 clock to SMCCLK × 15 clock

10.3.8 smc_sram_cycles0_1 (SMC SRAM Cycles Registers 0 <1>)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	t_tr			-
After reset	Undefined	Undefined	Undefined	Undefined	0	0	1	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	t_wp			t_ceoe		
After reset	1	0	1	1	0	0	1	1
	7	6	5	4	3	2	1	0
bit symbol	t_wc				t_rc			
After reset	1	1	0	0	1	1	0	0

Bit	Bit Symbol	Type	Function
31-20	–	W	Write as "0".
19-17	t_tr[2:0]	R	Turn around cycle time 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
16-14	–	R	Read as undefined.
13-11	t_wp[2:0]	R	$\overline{WE}$ pulse cycle time 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
10-8	t_ceoe[2:0]	R	Delay cycle time to $\overline{OE}$ assert 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
7-4	t_wc[3:0]	R	Write cycle time 0011 to 1111 : SMCCLK × 3 clock to SMCCLK × 15 clock
3-0	t_rc[3:0]	R	Read cycle time 0010 to 1111 : SMCCLK × 2 clock to SMCCLK × 15 clock

10.3.9 smc_sram_cycles0_2 (SMC SRAM Cycles Registers 0 <2>)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-		t_tr		-
After reset	Undefined	Undefined	Undefined	Undefined	0	0	1	0
	15	14	13	12	11	10	9	8
bit symbol	-	-		t_wp			t_ceoe	
After reset	1	0	1	1	0	0	1	1
	7	6	5	4	3	2	1	0
bit symbol			t_wc				t_rc	
After reset	1	1	0	0	1	1	0	0

Bit	Bit Symbol	Type	Function
31-20	−	W	Write as "0".
19-17	t_tr[2:0]	R	Turn around cycle time 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
16-14	−	R	Read as undefined.
13-11	t_wp[2:0]	R	$\overline{WE}$ pulse cycle time 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
10-8	t_ceoe[2:0]	R	Delay cycle time to $\overline{OE}$ assert 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
7-4	t_wc[3:0]	R	Write cycle time 0011 to 1111 : SMCCLK × 3 clock to SMCCLK × 15 clock
3-0	t_rc[3:0]	R	Read cycle time 0010 to 1111 : SMCCLK × 2 clock to SMCCLK × 15 clock

10.3.10   smc_sram_cycles0_3 (SMC SRAM Cycles Registers 0 <3>)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-		t_tr		-
After reset	Undefined	Undefined	Undefined	Undefined	0	0	1	0
	15	14	13	12	11	10	9	8
bit symbol	-	-		t_wp			t_ceoe	
After reset	1	0	1	1	0	0	1	1
	7	6	5	4	3	2	1	0
bit symbol		t_wc				t_rc		
After reset	1	1	0	0	1	1	0	0

Bit	Bit Symbol	Type	Function
31-20	−	W	Write as "0".
19-17	t_tr[2:0]	R	Turn around cycle time 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
16-14	−	R	Read as undefined.
13-11	t_wp[2:0]	R	$\overline{\text{WE}}$ pulse cycle time 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
10-8	t_ceoe[2:0]	R	Delay cycle time to $\overline{\text{OE}}$ assert 001 to 111 : SMCCLK × 1 clock to SMCCLK × 7 clock
7-4	t_wc[3:0]	R	Write cycle time 0011 to 1111 : SMCCLK × 3 clock to SMCCLK × 15 clock
3-0	t_rc[3:0]	R	Read cycle time 0010 to 1111 : SMCCLK × 2 clock to SMCCLK × 15 clock

10.3.11 smc_opmode0_0 (SMC Opmode Registers 0<0>)

	31	30	29	28	27	26	25	24
bit symbol	address_match							
After reset	0	1	1	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	adv	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	1	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	rd_bl			-	mw	
After reset	Undefined	Undefined	0	0	0	Undefined	1	0

Bit	Bit Symbol	Type	Function
31-24	address_match [7:0]	R	Start address of CS0 area Read as "0x60".
23-16	−	R	Read as "0xFF".
15-12	−	R	Read as undefined.
11	adv	R	Address latch enable signal 0 : Address latch enable signal ($\overline{ALE}$) not used 1 : Address latch enable signal ($\overline{ALE}$) used
10-6	−	R	Read as undefined.
5-3	rd_bl[2:0]	R	Burst length of data read 000 : 1 beat 001 : 4 beats 010 to 111 : Don't care
2	−	R	Read as undefined.
1-0	mw[1:0]	R	Data bus width of CS0 01 : 16 bits Others : Don't care

Note:Do not access the external memory area except set CS area.

10.3.12 smc_opmode0_1 (SMC Opmode Registers 0<1>)

	31	30	29	28	27	26	25	24
bit symbol				address_match				
After reset	0	1	1	0	0	0	0	1
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	adv	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	1	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-		rd_bl		-		mw
After reset	Undefined	Undefined	0	0	0	Undefined	1	0

Bit	Bit Symbol	Type	Function
31-24	address_match [7:0]	R	Start address of CS1 area Read as "0x61".
23-16	–	R	Read as "0xFF".
15-12	–	R	Read as undefined.
11	adv	R	Address latch enable signal 0 : Address latch enable signal ($\overline{ALE}$) not used 1 : Address latch enable signal ($\overline{ALE}$) used
10-6	–	R	Read as undefined.
5-3	rd_bl[2:0]	R	Burst length of data read 000 : 1 beat 001 : 4 beats 010 to 111 : Don't care
2	–	R	Read as undefined.
1-0	mw[1:0]	R	Data bus width of CS1 01 : 16 bits Others : Don't care

Note:Do not access the external memory area except set CS area.

10.3.13   smc_opmode0_2 (SMC Opmode Registers 0<2>)

	31	30	29	28	27	26	25	24
bit symbol	address_match							
After reset	0	1	1	0	0	0	1	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	adv	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	1	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	rd_bl			-	mw	
After reset	Undefined	Undefined	0	0	0	Undefined	1	0

Bit	Bit Symbol	Type	Function
31-24	address_match [7:0]	R	Start address of CS2 area Read as "0x62".
23-16	−	R	Read as "0xFF".
15-12	−	R	Read as undefined.
11	adv	R	Address latch enable signal 0 : Address latch enable signal (ALE) not used 1 : Address latch enable signal (ALE) used
10-6	−	R	Read as undefined.
5-3	rd_bl[2:0]	R	Burst length of data read 000 : 1 beat 001 : 4 beats 010 to 111 : Don't care
2	−	R	Read as undefined.
1-0	mw[1:0]	R	Data bus width of CS2 01 : 16 bits Others : Don't care

Note: Do not access the external memory area except set CS area.

10.3.14 smc_opmode0_3 (SMC Opmode Registers 0<3>)

	31	30	29	28	27	26	25	24
bit symbol	address_match							
After reset	0	1	1	0	0	0	1	1
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	adv	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	1	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	rd_bl			-	mw	
After reset	Undefined	Undefined	0	0	0	Undefined	1	0

Bit	Bit Symbol	Type	Function
31-24	address_match [7:0]	R	Start address of CS3 area Read as "0x63".
23-16	−	R	Read as "0xFF".
15-12	−	R	Read as undefined.
11	adv	R	Address latch enable signal 0 : Address latch enable signal ($\overline{ALE}$) not used 1 : Address latch enable signal ($\overline{ALE}$) used
10-6	−	R	Read as undefined.
5-3	rd_bl[2:0]	R	Burst length of data read 000 : 1 beat 001 : 4 beats 010 to 111 : Don't care
2	−	R	Read as undefined.
1-0	mw[1:0]	R	Data bus width of CS3 01 : 16 bits Others : Don't care

Note:Do not access the external memory area except set CS area.

10.4   External Bus Cycle

10.4.1   Multiplex mode

10.4.1.1 t_{RC} / t_{CEOE} setting example

t_{RC} =4, t_{CEOE} =1 (smc_set_cycles = 0x0002B1C4)

		t_{TR}	t_{PC}	t_{WP}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
Setting value	0	001(1)	010(2)	110(6)	001(1)	1100(C)	0100(4)
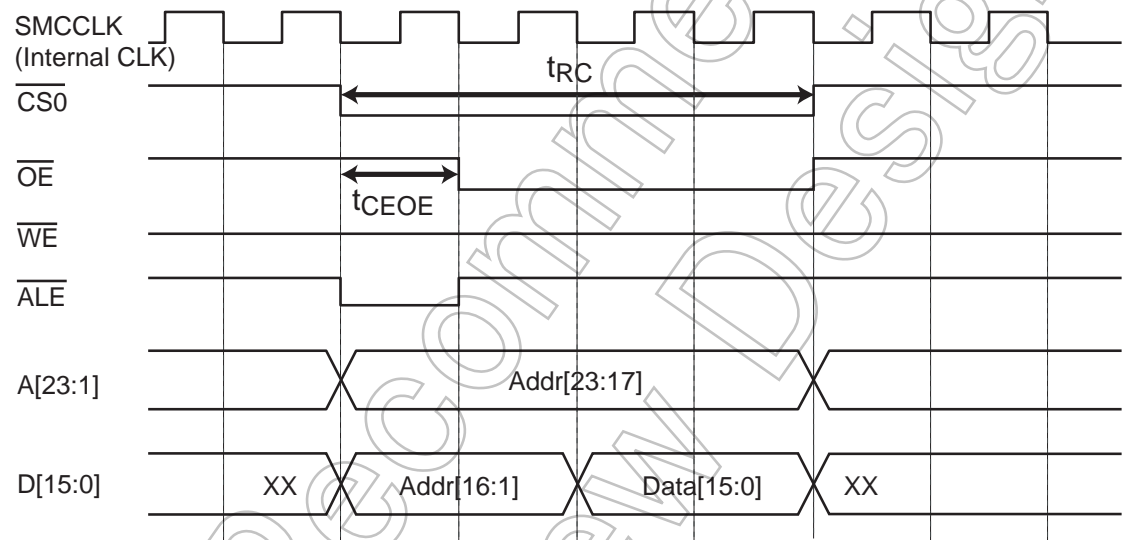
Figure 10-2 Asynchronous Read

10.4.1.2 t_{WC} / t_{WP} setting example

t_{WC} =5, t_{WP} =1 (smc_set_cycles = 0x00028B5C)

		t_{TR}	t_{PC}	t_{WP}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
Setting value	0	001(1)	010(2)	001(1)	011(3)	0101(5)	1100(C)
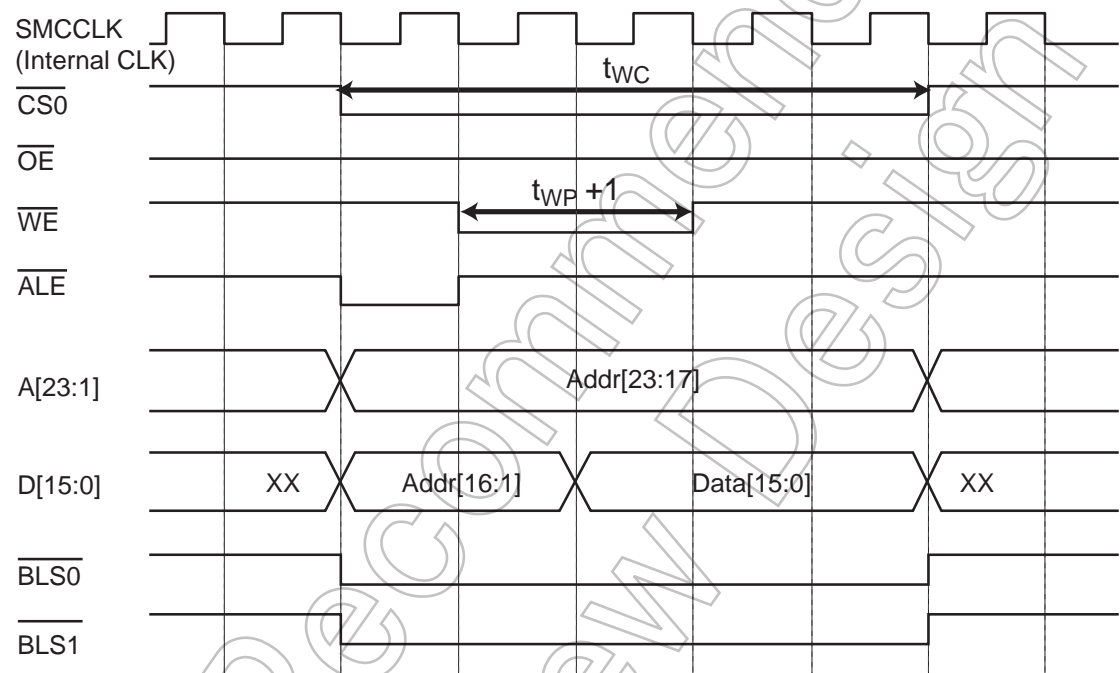
Figure 10-3 Asynchronous Write

10.4.1.3 t_{TR} setting example

t_{TR} =1 (smc_set_cycles = 0x00029144)

		t_{TR}	t_{PC}	t_{WP}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
Setting value	0	001(1)	010(2)	010(2)	001(1)	0100(4)	0100(4)
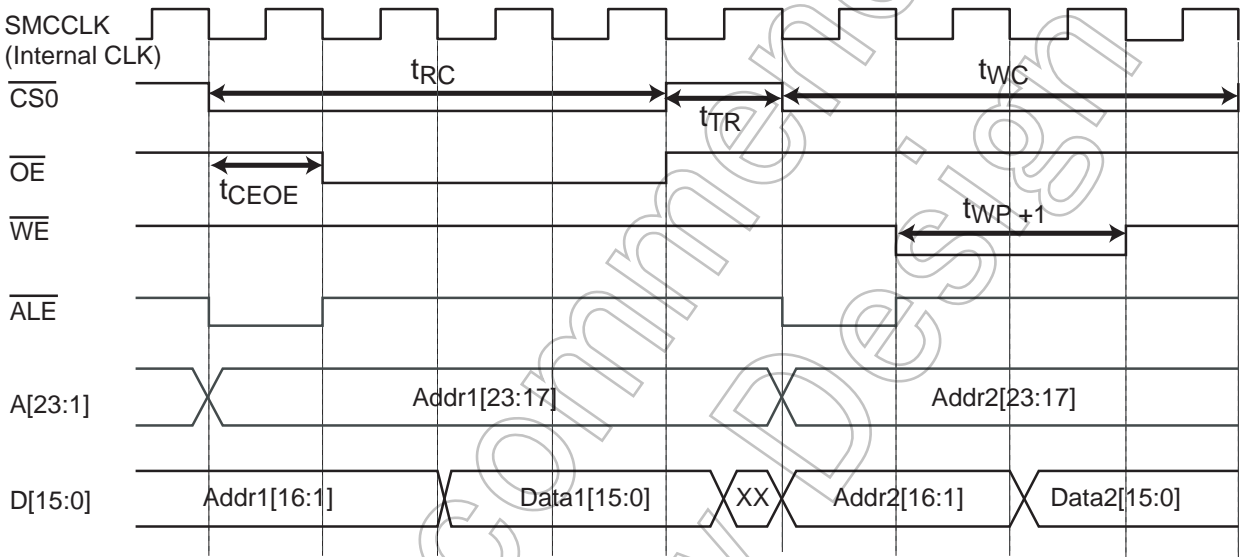
Figure 10-4 Asynchronous Read and Asynchronous Write

10.5   Connection example for external memory

Below figures show connection example for external 16bit NOR-Flash and 16bit SRAM.
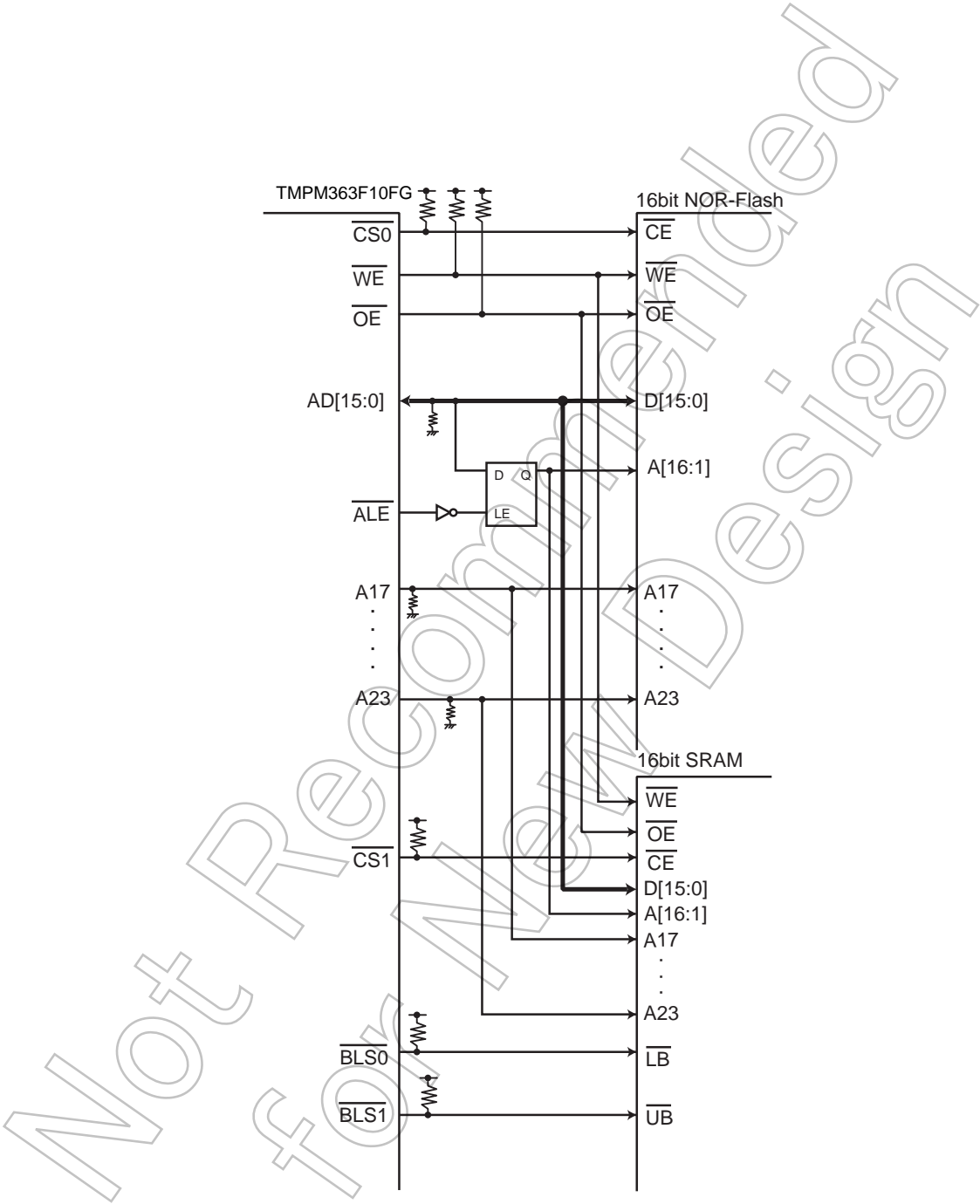
Figure 10-5 Connection example for external 16bit SRAM and NOR-Flash (Multiplex mode)

11.   16-bit Timer / Event Counters (TMRB)

11.1   Outline

TMRB operate in the following four operation modes:

- · 16-bit interval timer mode
- · 16-bit event counter mode
- · 16-bit programmable pulse generation mode (PPG)
- · Timer synchronous mode

The use of the capture function allows TMRB to perform the following three measurements.

- · One shot pulse output by an external trigger
- · Frequency measurement
- · Pulse width measurement
- · Time difference measurement

In the following explanation of this section, "x" indicates a channel number.

11.2 Differences in the Specifications

TMPM363F10FG contains 16-channel of TMRB.

Each channel functions independently and the channels operate in the same way except for the differences in their specification as shown in Table 11-1.

Some of the channels can put the capture trigger and the synchronous start trigger on other channels.

1. The flip-flop output of TMRB 0, 4, 8 and C can be used as the capture trigger of other channels.
   - TB0OUT → available for TMRB5 through TMRB7
   - TB4OUT → available for TMRB1 through TMRB3
   - TB8OUT → available for TMRBD through TMRBF
   - TBCOUT → available for TMRB9 through TMRBB

2. The start trigger of the timer synchronous mode (with TBxRUN)
   - TMRB0 → can start TMRB0 through TMRB3 synchronously
   - TMRB4 → can start TMRB4 through TMRB7 synchronously
   - TMRB8 → can start TMRB8 through TMRBB synchronously
   - TMRBC → can start TMRBC through TMRBF synchronously

Note: TMRB8 to TMRBF do not have timer output.

Table 11-1 Differences in the Specifications of TMRB Modules

Specification Channel	External pins		Trigger function between timers		Interrupt	
	External clock / capture trigger input pins Signal	Timer flip-flop output pin Signal	Capture trigger	Synchronous start trigger channel	Capture in- terrupt	TMRB interrupt
TMRB0	−	TB0OUT	−	TMRB0	−	INTTB0
TMRB1	TB1IN0 TB1IN1	TB1OUT	TB4OUT	TMRB0	INTCAP10 INTCAP11	INTTB1
TMRB2	TB2IN0 TB2IN1	TB2OUT	TB4OUT	TMRB0	INTCAP20 INTCAP21	INTTB2
TMRB3	− (Connect to SCLK3)	TB3OUT	TB4OUT	TMRB0	−	INTTB3
TMRB4	−	TB4OUT	−	TMRB4	−	INTTB4
TMRB5	TB5IN0 TB5IN1	TB5OUT (Connect to SIO0 to SIO3)	TB0OUT	TMRB4	INTCAP50 INTCAP51	INTTB5
TMRB6	TB6IN0 TB6IN1	TB6OUT (Connect to SIO4)	TB0OUT	TMRB4	INTCAP60 INTCAP61	INTTB6
TMRB7	TB7IN0 TB7IN1	TB7OUT	TB0OUT	TMRB4	INTCAP70 INTCAP71	INTTB7
TMRB8	−	−	−	TMRB8	−	INTTB8
TMRB9	TB9IN0 TB9IN1	−	−	TMRB8	INTCAP90 INTCAP91	INTTB9
TMRBA	−	− (Connect to CEC)	−	TMRB8	−	INTTBA
TMRBB	−	− (Connect to RMC)	−	TMRB8	−	INTTBB
TMRBC	−	−	−	TMRBC	−	INTTBC
TMRBD	−	−	−	TMRBC	−	INTTBD
TMRBE	−	−	−	TMRBC	−	INTTBE
TMRBF	−	−	−	TMRBC	−	INTTBF

11.3   Configuration

Each channel consists of a 16-bit up-counter, two 16-bit timer registers (double-buffered), two 16-bit capture registers, two comparators, a capture input control, a timer flip-flop and its associated control circuit.Timer operation modes and the timer flip-flop are controlled by a register.
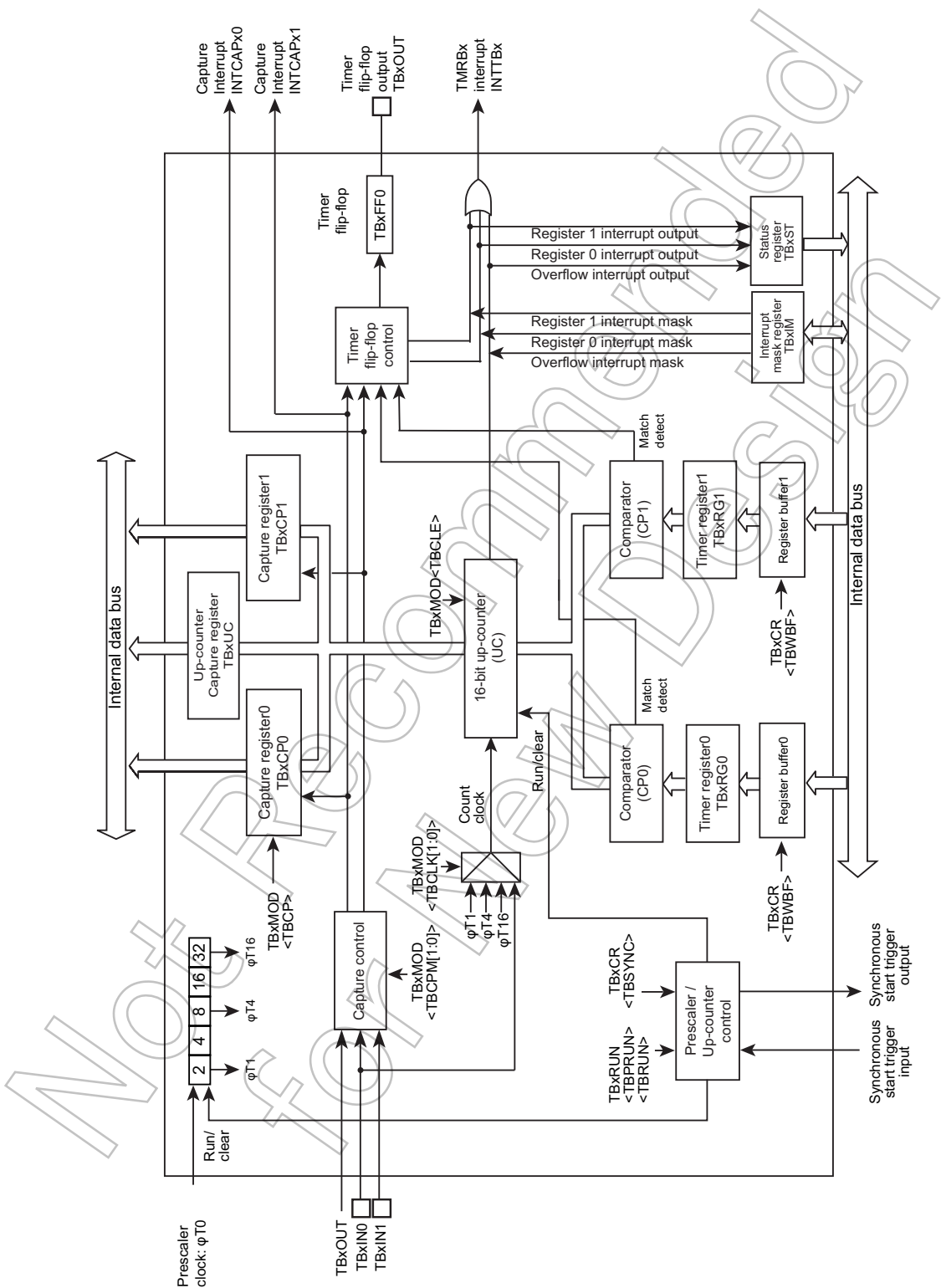
Figure 11-1 TMRBx Block Diagram (x= 0 to 2, 4 to F)
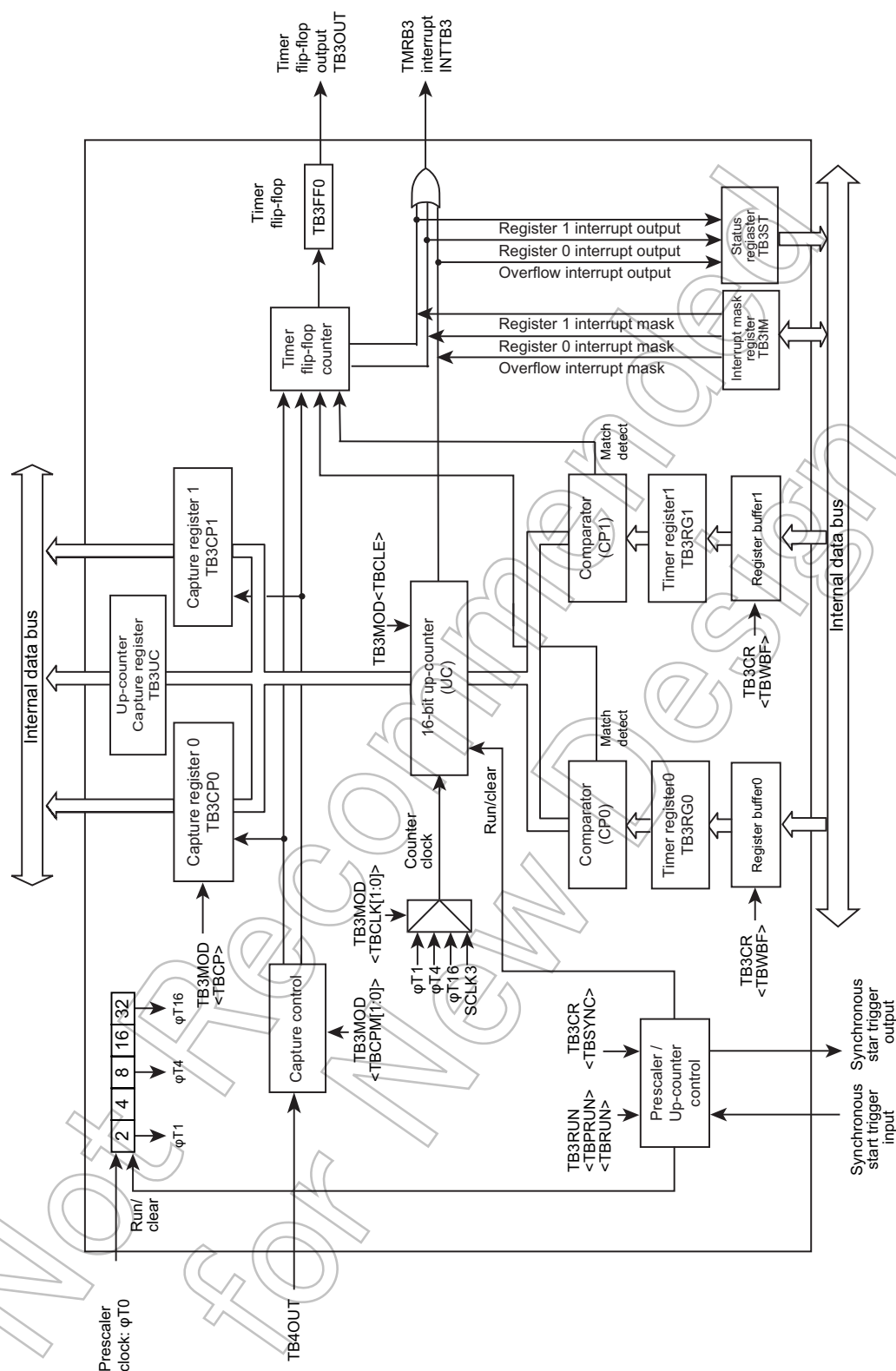
Figure 11-2 TMRBx Block Diagram (x= 3)

11.4   Registers

11.4.1   Register list according to channel

The following table shows the register names and addresses of each channel.

Channel x	Base Address
Channel 0	0x400D_0000
Channel 1	0x400D_0100
Channel 2	0x400D_0200
Channel 3	0x400D_0300
Channel 4	0x400D_0400
Channel 5	0x400D_0500
Channel 6	0x400D_0600
Channel 7	0x400D_0700
Channel 8	0x400D_0800
Channel 9	0x400D_0900
Channel A	0x400D_0A00
Channel B	0x400D_0B00
Channel C	0x400D_0C00
Channel D	0x400D_0D00
Channel E	0x400D_0E00
Channel F	0x400D_0F00

Register name (x=0 to F)		Address (Base+)
Enable register	TBxEN	0x0000
RUN register	TBxRUN	0x0004
Control register	TBxCR	0x0008
Mode register	TBxMOD	0x000C
Flip-flop control register	TBxFFCR	0x0010
Status register	TBxST	0x0014
Interrupt mask register	TBxIM	0x0018
Up counter capture register	TBxUC	0x001C
Timer register 0	TBxRG0	0x0020
Timer register 1	TBxRG1	0x0024
Capture register 0	TBxCP0	0x0028
Capture register 1	TBxCP1	0x002C

11.4.2   TBxEN (Enable register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBEN	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TBEN	R/W	TMRBx operation 0: Disable 1: Enable Specifies the TMRB operation. When the operation is disabled, no clock is supplied to the other registers in the TMRB module. This can reduce power consumption. (This disables reading from and writing to the other registers except TBxEN register.) To use the TMRB, enable the TMRB operation (set to "1") before programming each register in the TMRB module. If the TMRB operation is executed and then disabled, the settings will be maintained in each register.
6-0	−	R	Read as "0".

11.4.3 TBxRUN (RUN register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	TBPRUN	-	TBRUN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	–	R	Read as "0".
2	TBPRUN	R/W	Prescaler operation 0: Stop & clear 1: Count
1	–	R	Read as "0".
0	TBRUN	R/W	Count operation 0: Stop & clear 1: Count

Note: When the counter is stopped (<TBRUN>="0") and TBxUC<TBUC[15:0]> is read, the value which was captured when the counter was operated is read.

11.4.4   TBxCR (Control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBWBF	-	TBSYNC	-	I2TB	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0"
7	TBWBF	R/W	Double buffer 0: Disable 1: Enable
6	−	R/W	Write as "0".
5	TBSYNC	R/W	Synchronous mode switching 0: individual (unit of channel) 1: synchronous
4	−	R	Read as "0".
3	I2TB	R/W	Operation at IDLE mode 0: Stop 1:Operation
2	−	R	Read as "0".
1-0	−	R/W	Write as "0".

Note:Do not modify TBxCR during operating TMRB.

11.4.5 TBxMOD (Mode register)

x=0 to 2, 4 to F

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	TBCP	TBCPM		TBCLE	TBCLK	
After reset	0	0	1	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6	−	R/W	Write as "0".
5	TBCP	W	Capture control by software 0: Capture by software 1: Don't care When "0" is written, the capture register 0 (TBxCP0) takes count value. Read as "1".
4-3	TBCPM[1:0]	R/W	Capture timing 00: Disable 01: TBxIN0↑ TBxIN1↑ Takes count values into capture register 0 (TBxCP0) upon rising of TBxIN0 pin input. Takes count values into capture register 1 (TBxCP1) upon rising of TBxIN1 pin input. 10: TBxIN0↑ TBxIN0↓ Takes count values into capture register 0 (TBxCP0) upon rising of TBxIN0 pin input. Takes count values into capture register 1 (TBxCP1) upon falling of TBxIN0 pin input. 11: TBxOUT↑ TBxOUT↓ Takes count values into capture register 0 (TBxCP0) upon rising of 16-bit timer match output (TBxOUT) and into capture register 1 (TBxCP1) upon falling of TBxOUT. (TMRB1 to 3: TB4OUT , TMRB5 to 7: TB0OUT , TMRB9 to B: TBCOUT , TMRBD to F: TB8OUT)
2	TBCLE	R/W	Up-counter control 0: Disables clearing of the up-counter. 1: Enables clearing of the up-counter. Clears and controls the up-counter. When "0" is written, it disables clearing of the up-counter. When "1" is written, it clears up counter when there is a match with Timer Regsiter1 (TBxRG1).
1-0	TBCLK[1:0]	R/W	Selects the TMRBx source clock. 00: TBxIN0 pin input 01: φT1 10: φT4 11: φT16

Note:TMRB0, 3, 4, 8 and A does not have TBxIN0 input and TBxIN1 input.

x=3

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	TBCLE	TBCLK	
After reset	0	0	1	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6	−	R/W	Write as "0".
5	−	W	Write as "1".
4-3	−	R/W	Write as "00".
2	TBCLE	R/W	Up-counter control 0: Disables clearing of the up-counter. 1: Enables clearing of the up-counter. Clears and controls the up-counter. When "0" is written, it disables clearing of the up-counter. When "1" is written, it clears up counter when there is a match with Timer Regsiter1 (TBxRG1).
1-0	TBCLK[1:0]	R/W	Selects the TMRBx source clock. 00: SCLK3 pin input 01: φT1 10: φT4 11: φT16

11.4.6 TBxFFCR (Flip-flop control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	TBC1T1	TBC0T1	TBE1T1	TBE0T1	TBFF0C	
After reset	1	1	0	0	0	0	1	1

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-6	−	R	Read as "1".
5	TBC1T1	R/W	TBxFF0 reverse trigger when the up-counter value is taken into the TBxCP1. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when the up-counter value is taken into the Capture register 1 (TBxCP1).
4	TBC0T1	R/W	TBxFF0 reverse trigger when the up-counter value is taken into the TBxCP0. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when the up-counter value is taken into the Capture register 0 (TBxCP0).
3	TBE1T1	R/W	TBxFF0 reverse trigger when the up-counter value is matched with TBxRG1. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when the up-counter value is matched with the Timer register 1 (TBxRG1).
2	TBE0T1	R/W	TBxFF0 reverse trigger when the up-counter value is matched with TBxRG0. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when an up-counter value is matched with the Timer register 0 (TBxRG0).
1-0	TBFF0C[1:0]	R/W	TBxFF0 control 00: Invert Reverses the value of TBxFF0 (reverse by using software). 01: Set Sets TBxFF0 to "1". 10: Clear Clears TBxFF0 to "0". 11: Don't care * This is always read as "11".

Note: Do not modify TBxFFCR during operating TMRB.

11.4.7 TBxST (Status register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	INTTBOF	INTTB1	INTTB0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	INTTBOF	R	Overflow flag 0:No overflow occurs 1:Overflow occurs When an up-counter is overflow, "1" is set.
1	INTTB1	R	Match flag (TBxRG1) 0:No detection of a mach 1:Detects a match with TBxRG1 When a match with the timer register 1 (TBxRG1) is detected, "1" is set.
0	INTTB0	R	Match flag (TBxRG0) 0:No match is detected 1:Detects a match with TBxRG0 When a match with the timer register 0 (TBxRG0) is detected, "1" is set.

Note 1: The factors only which is not masked by TBxIM output interrupt request to the CPU. Even if the mask setting is done, the flag is set.

Note 2: The flag is cleared by reading the TBxST register.To clear the flag, TBxST register should be read.

11.4.8 TBxIM (Interrupt mask register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	TBIMOF	TBIM1	TBIM0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	TBIMOF	R/W	Overflow interrupt mask 0:Disable 1:Enable Sets the up-counter overflow interrupt to disable or enable.
1	TBIM1	R/W	Match interrupt mask (TBxRG1) 0:Disable 1:Enable Sets the match interrupt mask with the Timer register 1 (TBxRG1) to enable or disable.
0	TBIM0	R/W	Match interrupt mask (TBxRG0) 0:Disable 1:Enable Sets the match interrupt mask with the Timer register 0 (TBxRG0) to enable or disable.

Note:Even if TBxIM setting is done, TBxST is set.

11.4.9   TBxUC (Up counter capture register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBUC							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBUC							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as "0".
15-0	TBUC[15:0]	R	Captures a value by reading up-counter out. If TBxUC is read, current up-counter value can be captured.

Note: When the counter is operated and TBxUC is read, the value of the up counter is captured and read.

11.4.10 TBxRG0 (Timer register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBRG0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBRG0							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as "0".
15-0	TBRG0[15:0]	R/W	Sets a value comparing to the up-counter.

11.4.11 TBxRG1 (Timer register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBRG1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBRG1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as "0".
15-0	TBRG1[15:0]	R/W	Sets a value comparing to the up-counter.

11.4.12   TBxCP0 (Capture register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBCP0							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	TBCP0							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-0	TBCP0[15:0]	R	A value captured from the up-counter is read.

11.4.13   TBxCP1 (Capture register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBCP1							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	TBCP1							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-0	TBCP1[15:0]	R	A value captured from the up-counter is read.

11.5   Description of Operations for Each Circuit

The channels operate in the same way, except for the differences in their specifications as shown in Table 11-1.

11.5.1   Prescaler

There is a 4-bit prescaler to generate the source clock for up-counter UC.

The prescaler input clock φT0 is fs, fperiph/1, fperiph/2, fperiph/4, fperiph/8, fperiph/16 or fperiph/32 selected by CGSYSCR<FPSEL1> in the CG. The peripheral clock, fperiph, is either fgear, a clock selected by CGSYSCR<FPSEL0> in the CG, or fc, which is a clock before it is divided by the clock gear.

The operation or the stoppage of a prescaler is set with TBxRUN<TBPRUN> where writing "1" starts counting and writing "0" clears and stops counting. Below tables show prescaler output clock resolutions.

Table 11-2 Prescaler Output Clock Resolutions (fc = 64MHz, fs = 32.768kHz)

Select φT0 CGSYSCR <FPSEL1>	Select peripheral clock CGSYSCR <FPSEL0>	Select gear clock CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function		
				φT1	φT4	φT16
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0312 μs)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			001 (fperiph/2)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			010 (fperiph/4)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			011 (fperiph/8)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			100 (fperiph/16)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			101 (fperiph/32)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			001 (fperiph/2)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			010 (fperiph/4)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			011 (fperiph/8)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.8 μs)

Table 11-2 Prescaler Output Clock Resolutions (fc = 64MHz, fs = 32.768kHz)

Select φT0 CGSYSCR <FPSEL1>	Select peripheral clock CGSYSCR <FPSEL0>	Select gear clock CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function		
				φT1	φT4	φT16
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0312 μs)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
		100 (fc/2)	000 (fperiph/1)	–	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
		101 (fc/4)	000 (fperiph/1)	–	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
		110 (fc/8)	000 (fperiph/1)	–	–	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	–	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
1	*	*	*	fs/2 (61μs)	$fs/2^3$ (244 μs)	$fc/2^5$ (977 μs)

Note 1: The prescaler output clock φTn must be selected so that φTn < fsys is satisfied (so that φTn is slower than fsys).

Note 2: Do not change the clock gear while the timer is operating.

Note 3: "." denotes a setting prohibited. "*" denotes a don't care.

Table 11-3 Prescaler Output Clock Resolutions (fc = 48MHz, fs = 32.768kHz)

Select φT0 CGSYSCR <FPSEL1>	Select peripheral clock CGSYSCR <FPSEL0>	Select gear clock CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function		
				φT1	φT4	φT16
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.04 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			001 (fperiph/2)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			010 (fperiph/4)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			011 (fperiph/8)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			100 (fperiph/16)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
			101 (fperiph/32)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			001 (fperiph/2)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			010 (fperiph/4)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			011 (fperiph/8)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
			100 (fperiph/16)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
			101 (fperiph/32)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{12}$ (85.33 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			001 (fperiph/2)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			010 (fperiph/4)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
			011 (fperiph/8)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
			100 (fperiph/16)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{12}$ (85.33 μs)
			101 (fperiph/32)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{13}$ (170.67 μs)

Table 11-3 Prescaler Output Clock Resolutions (fc = 48MHz, fs = 32.768kHz)

Select φT0 CGSYSCR <FPSEL1>	Select peripheral clock CGSYSCR <FPSEL0>	Select gear clock CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function		
				φT1	φT4	φT16
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.04 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		100 (fc/2)	000 (fperiph/1)	−	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		101 (fc/4)	000 (fperiph/1)	−	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	−	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		110 (fc/8)	000 (fperiph/1)	−	−	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	−	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	−	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
1	*	*	*	fs/2 (61μs)	$fs/2^3$ (244 μs)	$fc/2^5$ (977 μs)

Note 1: The prescaler output clock φTn must be selected so that φTn < fsys is satisfied (so that φTn is slower than fsys).

Note 2: Do not change the clock gear while the timer is operating.

Note 3: "." denotes a setting prohibited. "*" denotes a don't care.

Table 11-4 Prescaler Output Clock Resolutions (fc = 32MHz, fs = 32.768kHz)

Select φT0 CGSYSCR <FPSEL1>	Select peripheral clock CGSYSCR <FPSEL0>	Select gear clock CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function		
				φT1	φT4	φT16
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0625 μs)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
			001 (fperiph/2)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			010 (fperiph/4)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			011 (fperiph/8)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			100 (fperiph/16)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			001 (fperiph/2)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			010 (fperiph/4)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			011 (fperiph/8)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			001 (fperiph/2)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			010 (fperiph/4)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			011 (fperiph/8)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			100 (fperiph/16)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
			101 (fperiph/32)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)	$fc/2^{13}$ (256.0 μs)

TOSHIBA

Table 11-4 Prescaler Output Clock Resolutions (fc = 32MHz, fs = 32.768kHz)

Select φT0 CGSYSCR <FPSEL1>	Select peripheral clock CGSYSCR <FPSEL0>	Select gear clock CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function		
				φT1	φT4	φT16
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0625 µs)	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)
			001 (fperiph/2)	$fc/2^2$ (0.125 µs)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)
			010 (fperiph/4)	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
			011 (fperiph/8)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
			100 (fperiph/16)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)
			101 (fperiph/32)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)
		100 (fc/2)	000 (fperiph/1)	–	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)
			001 (fperiph/2)	$fc/2^2$ (0.125 µs)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)
			010 (fperiph/4)	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
			011 (fperiph/8)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
			100 (fperiph/16)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)
			101 (fperiph/32)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)
		101 (fc/4)	000 (fperiph/1)	–	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)
			001 (fperiph/2)	–	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)
			010 (fperiph/4)	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
			011 (fperiph/8)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
			100 (fperiph/16)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)
			101 (fperiph/32)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)
		110 (fc/8)	000 (fperiph/1)	–	–	$fc/2^5$ (1.0 µs)
			001 (fperiph/2)	–	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)
			010 (fperiph/4)	–	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
			011 (fperiph/8)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
			100 (fperiph/16)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)
			101 (fperiph/32)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)
1	*	*	*	fs/2 (61µs)	$fs/2^3$ (244 µs)	$fc/2^5$ (977 µs)

Note 1: The prescaler output clock φTn must be selected so that φTn < fsys is satisfied (so that φTn is slower than fsys).

Note 2: Do not change the clock gear while the timer is operating.

Note 3: "." denotes a setting prohibited. "*" denotes a don't care.

2023/07/21

11.5.2 Up-counter (UC)

UC is a 16-bit binary counter.

- Source clock

    UC source clock, specified by TBxMOD<TBCLK[1:0]>, can be selected from either three types - φT1, φT4 and φT16 - of prescaler output clock or the external clock of the TBxIN0 pin.

- Counter start / stop

    Counter operation is specified by TBxRUN<TBRUN>. UC starts counting if <TBRUN> = "1", and stops counting and clears counter value if <TBRUN> = "0".

- Timing to clear UC

    1. When a match is detected.

        By setting TBxMOD<TBCLE> = "1", UC is cleared if when the comparator detects a match between counter value and the value set in TBxRG1. UC operates as a free-running counter if TBxMOD<TBCLE> = "0".

    2. When UC stops

        UC stops counting and clears counter value if TBxRUN<TBRUN> = "0".

- UC overflow

    If UC overflow occurs, the INTTBx overflow interrupt is generated.

11.5.3 Timer registers (TBxRG0, TBxRG1)

TBxRG0 and TBxRG1 are registers for setting values to compare with up-counter values and two registers are built into each channel. If the comparator detects a match between a value set in this timer register and that in a UC up-counter, it outputs the match detection signal.

TBxRG0 and TBxRG1 are consisted of the double-buffered configuration which are paired with register buffers. The double buffering is disabled in the initial state.

Controlling double buffering disable or enable is specified by TBxCR<TBWBF> bit. If <TBWBF> = "0", the double buffering becomes disable. If <TBWBF> = "1", it becomes enable. When the double buffering is enabled, a data transfer from the register buffer to the timer register (TBxRG0/1) is done in the case that UC is matched with TBxRG1.When the counter is stopped even if double buffering is enabled, the double buffering operates as a single buffer, and an immediate data can be written to the TBxRG0 and TBxRG1.

11.5.4 Capture

This is a circuit that controls the timing of latching values from the UC up-counter into the TBxCP0 and TBxCP1 capture registers. The timing with which to latch data is specified by TBxMOD<TBCPM[1:0]>.

Software can also be used to import values from the UC up-counter into the capture register; specifically, UC values are taken into the TBxCP0 capture register each time "0" is written to TBxMOD<TBCP>.

11.5.5 Capture registers (TBxCP0, TBxCP1)

This register captures an up-counter (UC) value.

11.5.6 Up-counter capture register (TBxUC)

Other than the capturing functions shown above, the current count value of the UC can be captured by reading the TBxUC registers.

11.5.7 Comparators (CP0, CP1)

This register compares with the up-counter (UC) and the value setting of the Timer Register (TBxRG0 and TBxRG1) to detect whether there is a match or not. If a match is detected, INTTBx is generated.

11.5.8 Timer Flip-flop (TBxFF0)

The timer flip-flop (TBxFF0) is reversed by a match signal from the comparator and a latch signal to the capture registers. It can be enabled or disabled to reverse by setting the TBxFFCR<TBC1T1, TBC0T1, TBE1T1, TBE0T1>.

The value of TBxFF0 becomes undefined after a reset. The flip-flop can be reversed by writing "00" to TBxFFCR<TBFF0C[1:0]>. It can be set to "1" by writing "01," and can be cleared to "0" by writing "10."

The value of TBxFF0 can be output to the Timer output pin (TBxOUT). If the timer output is performed, the corresponding port settings must be programmed beforehand.

11.5.9 Capture interrupt (INTCAPx0, INTCAPx1)

Interrupts INTCAPx0 and INTCAPx1 can be generated at the timing of latching values from the UC up-counter into the TBxCP0 and TBxCP1 capture registers. The interrupt timing is specified by the CPU.

11.6 Description of Operations for Each Mode

11.6.1 16-bit interval Timer Mode

In the case of generating constant period interrupt, set the interval time to the Timer register (TBxRG1) to generate the INTTBx interrupt.

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops count operation.
Interrupt Set-Enable Register	←	*	*	*	*	*	*	*	*	Permits INTTBx interrupt by setting corresponding bit to "1".
TBxFFCR	←	X	X	0	0	0	0	1	1	Disable to TBxFF0 reverse trigger
TBxMOD	←	X	0	1	0	0	1	*	*	Changes to prescaler output clock as input clock. Specifies capture function to disable.
							(** = 01, 10, 11)			
TBxRG1	←	*	*	*	*	*	*	*	*	Specifies a time interval. (16 bits)
	←	*	*	*	*	*	*	*	*	
TBxRUN	←	*	*	*	*	*	1	X	1	Starts TMRBx.

Note: X; Don't care
　　　 −; No change

11.6.2 16-bit Event Counter Mode

It is possible to make it the event counter by using an input clock as an external clock (TBxIN0 pin input).

The up-counter counts up on the rising edge of TBxIN0 pin input. It is possible to read the count value by capturing value using software and reading the captured value.

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops count TMRBx.
Set PORT registers.										Allocates corresponding port to TBxIN0.
TBxFFCR	←	X	X	0	0	0	0	1	1	Disable to TBxFF0 reverse trigger.
TBxMOD	←	X	0	1	0	0	0	0	0	Changes to TBxIN0 as an input clock.
TBxRUN	←	*	*	*	*	*	1	X	1	Starts TMRBx.
TBxMOD	←	X	0	0	0	0	0	0	0	Software capture is done.

Note: X; Don't care
　　　 −; No change

11.6.3   16-bit PPG (Programmable Pulse Generation) Output Mode

Square waves with any frequency and any duty (programmable square waves) can be output. The output pulse can be either low-active or high-active

Programmable square waves can be output from the TBxOUT pin by triggering the timer flip-flop (TBxFF) to reverse when the set value of the up-counter (UC) matches the set values of the timer registers (TBxRG0 and TBxRG1). Note that the set values of TBxRG0 and TBxRG1 must satisfy the following requirement:

Set value of TBxRG0 < Set value of TBxRG1
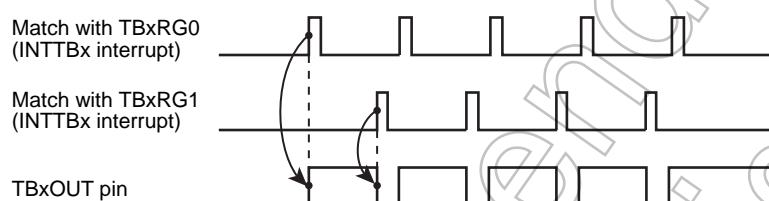
Figure 11-3 Example of Output of Programmable Pulse Generation (PPG)

In this mode, by enabling the double buffering of TBxRG0, the value of register buffer 0 is shifted into TBxRG0 when the set value of the up-counter matches the set value of TBxRG1. This facilitates handling of small duties.
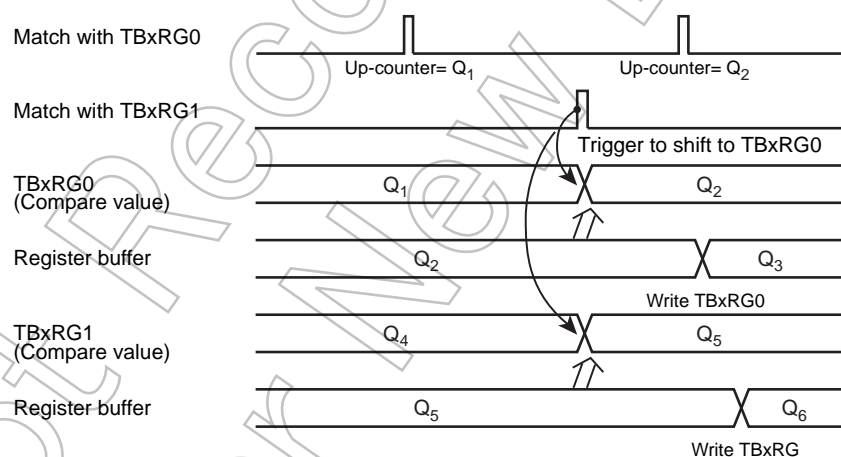
Figure 11-4 Register Buffer Operation

The block diagram of this mode is shown below.
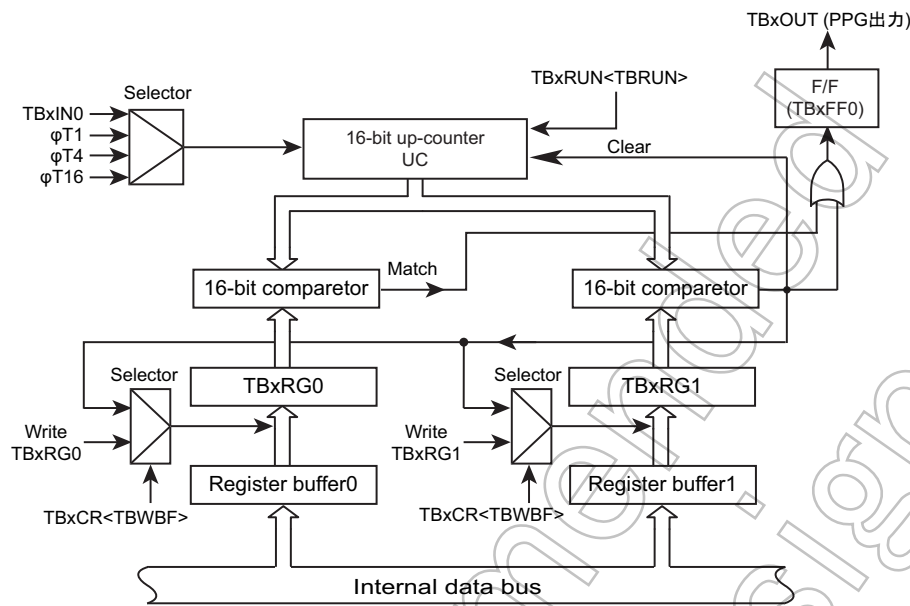
Figure 11-5 Block Diagram of 16-bit PPG Mode

Each register in the 16-bit PPG output mode must be programmed as listed below.

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops count operation.
TBxCR	←	0	0	−	X	−	X	0	0	Disable double buffering.
TBxRG0	←	*	*	*	*	*	*	*	*	Specifies a duty. (16 bits))
	←	*	*	*	*	*	*	*	*	
TBxRG1	←	*	*	*	*	*	*	*	*	Specifies a cycle. (16 bits)
	←	*	*	*	*	*	*	*	*	
TBxCR	←	1	0	−	X	−	X	0	0	Enables the TBxRG0 double buffering.
										(Changes the duty / cycle when the INTTBx interrupt is generated)
TBxFFCR	←	X	X	0	0	1	1	1	0	Specifies to trigger TBxFF0 to reverse when a match with TBxRG0 or TBxRG1 is detected, and sets the initial value of TBxFF0 to "0".
TBxMOD	←	X	0	1	0	0	1	*	*	Designates the prescaler output clock as the input clock, and disables the capture function.
								(** = 01, 10, 11)		
Set PORT registers.										Allocates corresponding port to TBxOUT.
TBxRUN	←	*	*	*	*	*	1	X	1	Starts TMRBx.

Note: X; Don't care −; No change

11.6.4 Timer synchronous mode

This mode enables the timers to start synchronously.

If the mode is used with PPG output, the output can be applied to drive a motor.

TMRB is consisted of pairs of 4-channel TMRB. If one channel starts, remaining 3 channels can be start synchronously. In the TMPM363F10FG, the following combinations allow to use.

Start trigger channel (Master channel)	Synchronous operation channel (Slave channel)
TMRB0	TMRB1, TMRB2, TMRB3
TMRB4	TMRB5, TMRB6, TMRB7
TMRB8	TMRB9, TMRBA, TMRBB
TMRBC	TMRBD, TMRBE, TMRBF

Use of the timer synchronous mode is specified in TBxCR<TBSYNC> bit.

- <TBSYNC> = "0" : Timer operates individually.
- <TBSYNC> = "1" : Timers operates synchronously.

Set "0" to the <TBSYNC> bit in the master channel.

If <TBSYNC>= "1" is set in the slave channel, the start timing is synchronized with master channel start timing. Setting of start timing for TBxRUN<TBPRUN, TBRUN> bit in the slave channel is not required.

11.7 Applications using the Capture Function

The capture function can be used to develop many applications, including those described below:

1. One-shot pulse output triggered by an external pulse

2. Frequency measurement

3. Pulse width measurement

4. Time difference measurement

11.7.1 One-shot pulse output triggered by an external pulse

One-shot pulse output triggered by an external pulse is carried out as follows:

The 16-bit up-counter is made to count up by putting it in a free-running state using the prescaler output clock. An external pulse is input through the TBxIN0 pin. A trigger is generated at the rising of the external pulse by using the capture function and the value of the up-counter is taken into the capture registers (TBxCP0).

The CPU must be programmed so that an interrupt INTCAPx0 is generated at the rising of an external trigger pulse. This interrupt is used to set the timer registers (TBxRG0) to the sum of the TBxCP0 value (c) and the delay time (d), (c + d), and set the timer registers (TBxRG1) to the sum of the TBxRG0 values and the pulse width (p) of one-shot pulse, (c + d + p). TBxRG1 change must be completed before the next match.

In addition, the timer flip-flop control registers(TBxFFCR<TBE1T1, TBE0T1>) must be set to "11". This enables triggering the timer flip-flop (TBxFF0) to reverse when UC matches TBxRG0 and TBxRG1. This trigger is disabled by the INTTBx interrupt after a one-shot pulse is output.

Symbols (c), (d) and (p) used in the text correspond to symbols c, d and p in "Figure 11-6 One-shot Pulse Output (With Delay)".
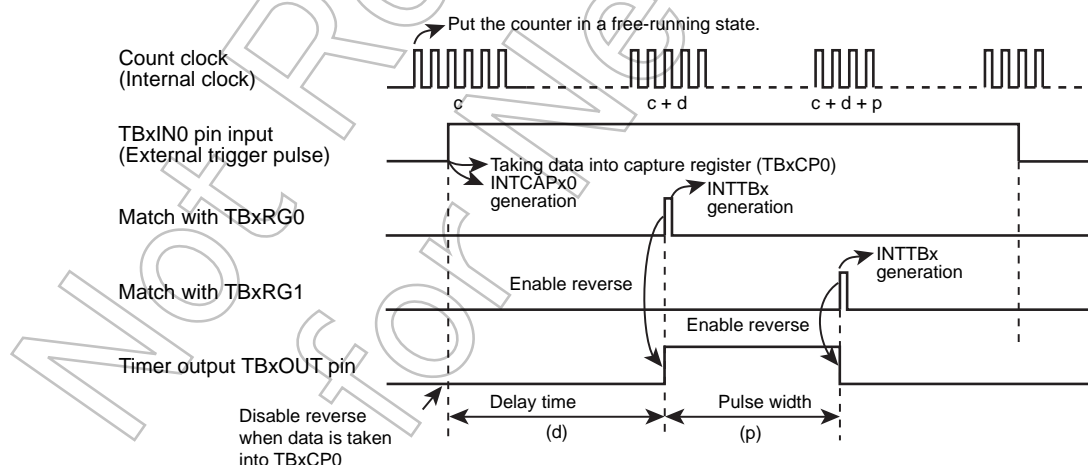
Figure 11-6 One-shot Pulse Output (With Delay)

The followings show the settings in the case that 2 ms width one-shot pulse is output after 3ms by triggering TBxIN0 input at the rising edge. (φT1 is selected for counting.)

		7	6	5	4	3	2	1	0	
[Main processing] Capture setting by TBxIN0										
Set PORT registers.										Allocates corresponding port to TBxIN0.
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops count operation.
TBxMOD	←	X	0	1	0	1	0	0	1	Changes source clock to φT1. Fetches a count value into the TBxCP0 at the rising edge of TBxIN0.
TBxFFCR	←	X	X	0	0	0	0	1	0	Clears TBxFF0 reverse trigger and disables.
Set PORT registers.										Allocates corresponding port to TBxOUT.
Interrupt Set-Enable Register	←	*	*	*	*	*	*	*	*	Permits to generate interrupts specified by INTCAPx0 interrupt corresponding bit by setting to "1".
TBxRUN	←	*	*	*	*	*	1	X	1	Starts the TMRBx module.
[Processing of INTCAPx0 interrupt service routine] Pulse output setting										
TBxRG0	←	*	*	*	*	*	*	*	*	Sets count value. (TBxCAP0 + 3ms/φT1)
	←	*	*	*	*	*	*	*	*	
TBxRG1	←	*	*	*	*	*	*	*	*	Sets count value.(TBxCAP0 + (3+2)ms/φT1)
	←	*	*	*	*	*	*	*	*	
TBxFFCR	←	X	X	–	–	1	1	–	–	Reverses TBxFF0 if UC consistent with TBxRG0 and TBxRG1.
TBxIM	←	X	X	X	X	X	1	0	1	Masks except TBxRG1 correspondence interrupt.
Interrupt Set-Enable Register	←	*	*	*	*	*	*	*	*	Permits to generate interrupt specified by INTTBx interrupt corresponding bit setting to "1".
[Processing of INTTBx interrupt service routine] Output disable										
TBxFFCR	←	X	X	–	–	0	0	–	–	Clears TBxFF0 reverse trigger setting.
Interrupt enable clear register	←	*	*	*	*	*	*	*	*	Prohibits interrupts specified by INTTBx interrupt corresponding bit by setting to "1".

Note: X; Don't care

       –; No change

If a delay is not required, TBxFF0 is reversed when data is taken into TBxCP0, and TBxRG1 is set to the sum of the TBxCP0 value (c) and the one-shot pulse width (p), (c + p), by generating the INTCAPx0 interrupt. TBxRG1 change must be completed before the next match.

TBxFF0 is enabled to reverse when UC matches with TBxRG1, and is disabled by generating the INTTBx interrupt.
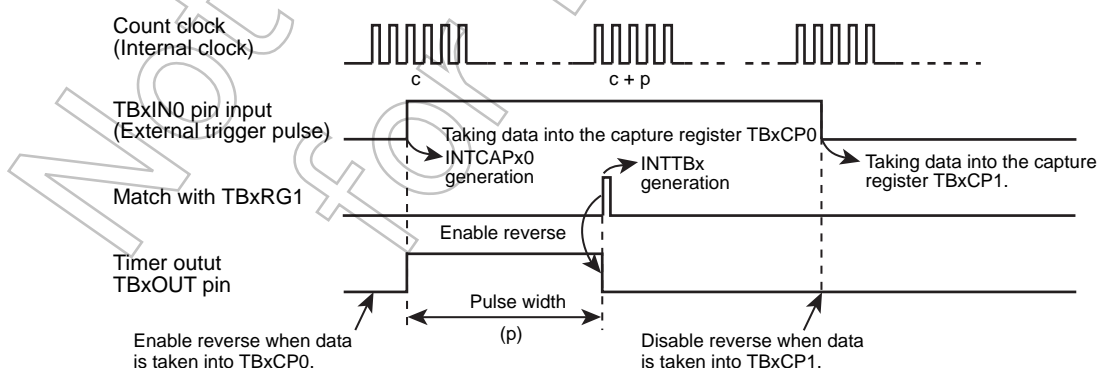
Figure 11-7 One-shot Pulse Output Triggered by an External Pulse (Without Delay)

11.7.2 Frequency measurement

The frequency of an external clock can be measured by using the capture function.

To measure frequency, another 16-bit timer is used in combination with the 16-bit event counter mode. As an example, we explain with TMRB5 and TMRB0. TB0OUT of the 16-bit timer TMRB0 is used to specify the measurement time.

TMRB5 count clock selects TB5IN0 input and performs count operation by using external clock input. If TB5MOD<TBCPM[1:0]> is set "11", TMRB5 count clock takes the counter value into the TB5CP0 at the rising edge of TB0OUT and takes the counter value into TB5CP1 at the falling edge of TB0OUT.

This setting allows a count value of the 16-bit up-counter UC to be taken into the capture register (TB5CP0) upon rising of a timer flip-flop output (TB0OUT) of the 16-bit timer (TMRB0), and an UC counter value to be taken into the capture register (TB5CP1) upon falling of TB0OUT of the 16-bit timer (TMRB0).

A frequency is then obtained from the difference between TB5CP0 and TB5CP1 based on the measurement, by generating the INTTB0 16-bit timer interrupt.

For example, if the difference between TB5CP0 and TB5CP1 is 100 and the level width setting value of TB0OUT is 0.5 s, the frequency is 200 Hz (100 ÷ 0.5 s = 200 Hz).
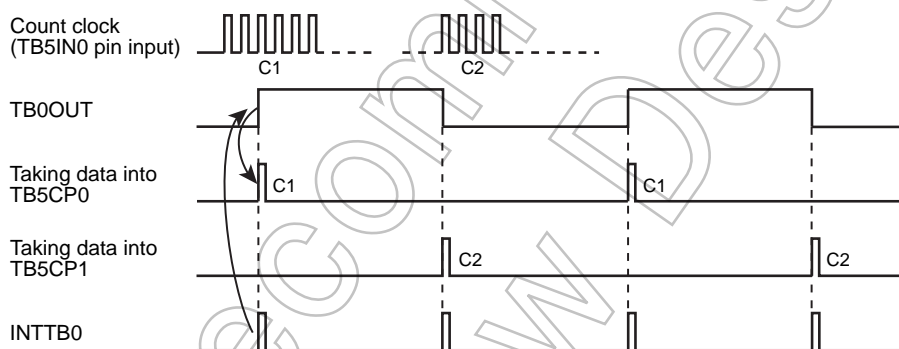
Figure 11-8 Frequency Measurement

11.7.3 Pulse width measurement

By using the capture function, the "High" level width of an external pulse can be measured. Specifically, by putting it in a free-running state using the prescaler output clock, an external pulse is input through the TBxIN0 pin and the up-counter (UC) is made to count up. A trigger is generated at each rising and falling edge of the external pulse by using the capture function and the value of the up-counter is taken into the capture registers (TBxCP0, TBxCP1). The CPU must be programmed so that INTCAPx1 is generated at the falling edge of an external pulse input through the TBxIN0 pin.

The "High" level pulse width can be calculated by multiplying the difference between TBxCP0 and TBxCP1 by the clock cycle of an internal clock.

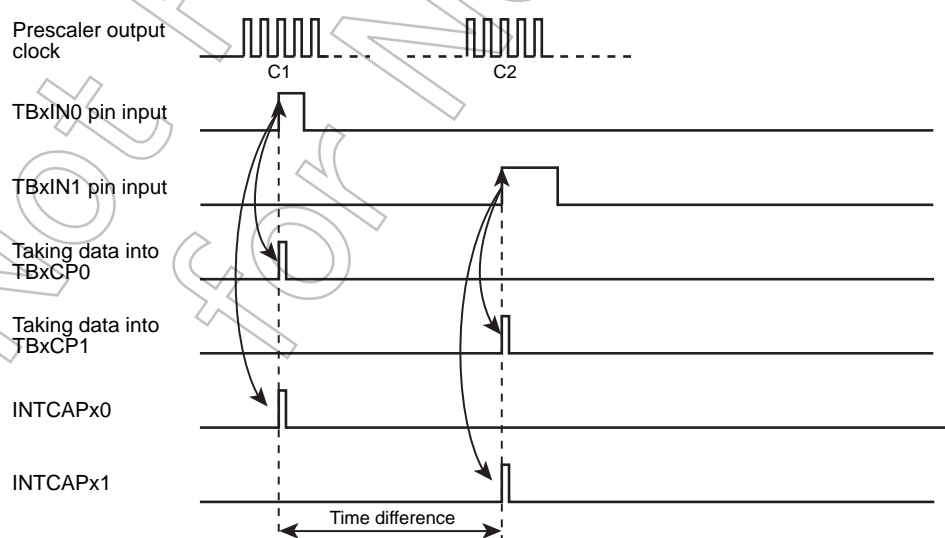For example, if the difference between TBxCP0 and TBxCP1 is 100 and the cycle of the prescaler output clock is 0.5 μs, the pulse width is 100 × 0.5 μs = 50 μs.

Caution must be exercised when measuring pulse widths exceeding the UC maximum count time which is dependant upon the source clock used. The measurement of such pulse widths must be made using software.

The "Low" level width of an external pulse can also be measured. In such cases, the difference between C2 generated the first time and C1 generated the second time is initially obtained by performing the second stage of INTCAPx0 interrupt processing as shown in "Figure 11-9 Pulse Width Measurement" and this difference is multiplied by the cycle of the prescaler output clock to obtain the "Low" level width.
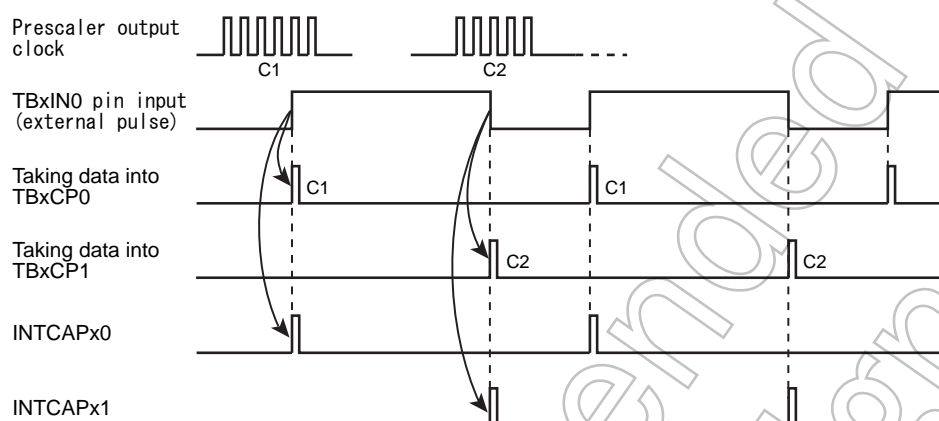
Figure 11-9 Pulse Width Measurement

11.7.4 Time Difference Measurement

The time difference of two events can be measured by the capture function. The up-counter (UC) is made to count up by putting it in a free-running state using the prescaler output clock.

The value of UC is taken into the capture register (TBxCP0) at the rising edge of the TBxIN0 pin input pulse. The CPU must be programmed to generate INTCAPx0 interrupt at this time.

The value of UC is taken into the capture register (TBxCP1) at the rising edge of the TBxIN1 pin input pulse. The CPU must be programmed to generate INTCAPx1 interrupt at this time.

The time difference can be calculated by multiplying the difference between TBxCP1 and TBxCP0 by the clock cycle of an internal clock.

Figure 11-10 Time Difference Measurement

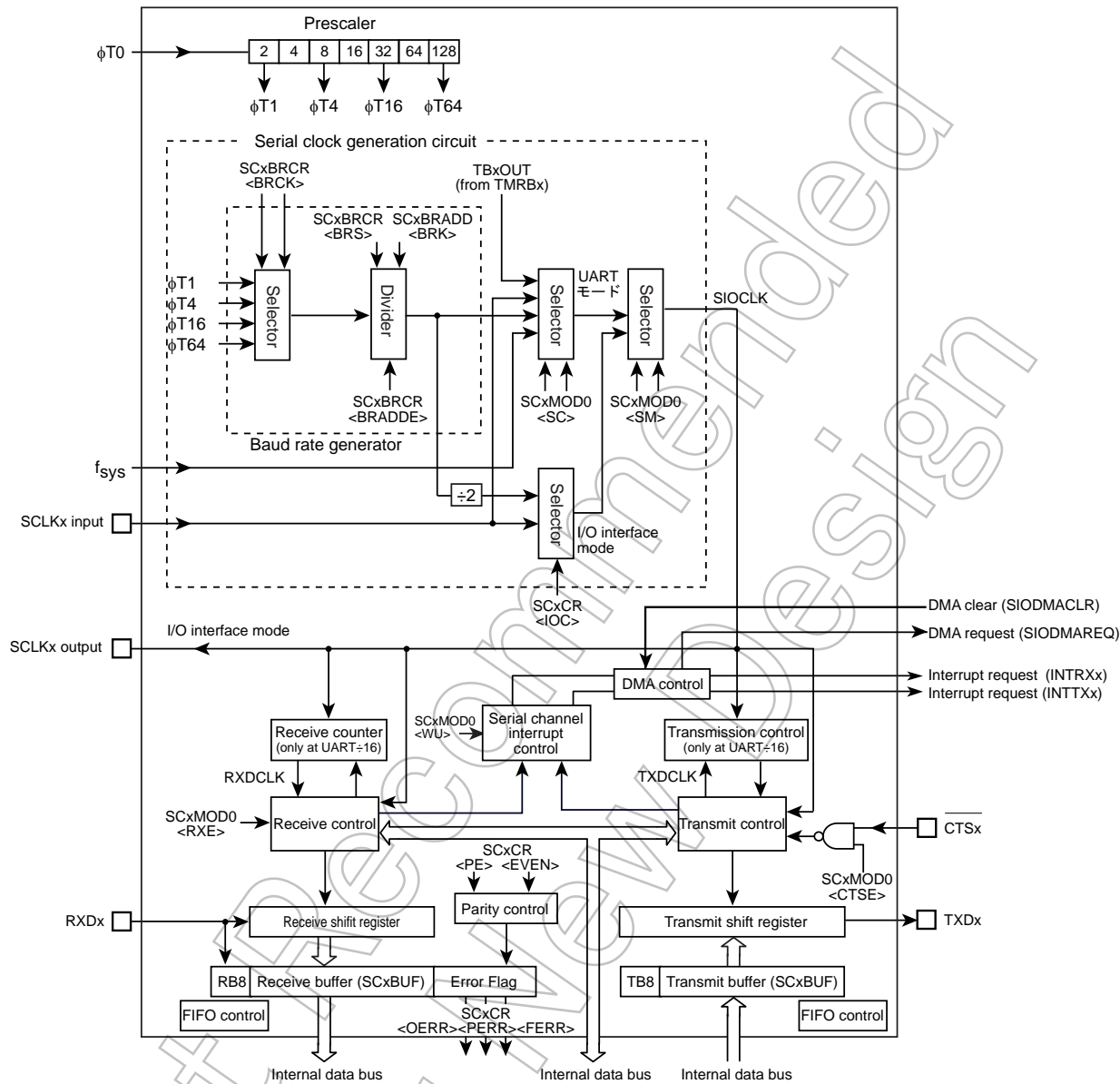# 12.3 Configuration

Figure 12-1 shows SIO block diagram.

Figure 12-1 SIO Block Diagram

12.4 Registers Description

12.4.1 Registers List in Each Channel

The below table shows registers and addresses for each register.

Channel x	Base Address
Channel0	0x400E_1000
Channel1	0x400E _ 1100
Channel2	0x400E _ 1200
Channel3	0x400E _ 1300
Channel4	0x400E _ 1400

Register name (x=0 to 4)		Address(Base+)
Enable register	SCxEN	0x0000
Buffer register	SCxBUF	0x0004
Control register	SCxCR	0x0008
Mode control register 0	SCxMOD0	0x000C
Baud rate generator control register	SCxBRCR	0x0010
Baud rate generator control register 2	SCxBRADD	0x0014
Mode control register 1	SCxMOD1	0x0018
Mode control register 2	SCxMOD2	0x001C
RX FIFO configuration register	SCxRFC	0x0020
TX FIFO configuration register	SCxTFC	0x0024
RX FIFO status register	SCxRST	0x0028
TX FIFO status register	SCxTST	0x002C
FIFO configuration register	SCxFCNF	0x0030

Note 1: **Do not modify any control register when data is being transmitted or received.**

Note 2: **Do not clear SCxMOD0<RXE> when data is being received.**

Note 3: **Do not clear SCxMOD1<TXE> when data is being transmitted.**

12.4.2 SCxEN (Enable Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	SIOE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	–	R	Read as "0".
0	SIOE	R/W	SIO operation 0: disable 1: Operation Specifies the SIO operation. To use the SIO, set <SIOE> = "1". When the operation is disabled, no clock is supplied to the other registers in the SIO module. This can reduce the power consumption. If the SIO operation is executed and then disabled, the settings will be maintained in each register except for SCxTFC<TIL>.

Note 1: **In case that SCxEN<SIOE>="0" (Stop SIO operation) or the operation mode is changed to IDLE mode with SCxMOD1<I2SC>="0" (Stop SIO operation in IDLE mode), SCxTFC is initiailzed again.**

Note 2: **In the DMA transfer using transmit / receive interrupt of SIO, firstly generate software reset by setting SCxMOD2<SWRST[1:0]>, next, enable DMAC (DMA request waiting state), and then start transmit / receive of SIO.**

12.4.3　SCxBUF (Buffer Register)

SCxBUF works as a transmit buffer or FIFO for write operation and as a receive buffer or FIFO for read operation.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TB / RB							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	TB[7:0] / RB [7:0]	R/W	[write] TB :Transmit buffer / FIFO [read] RB : Receive buffer / FIFO

12.4.4 SCxCR (Control Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	RB8	R	Receive data bit 8 (For UART) 9th bit of the received data in the 9 bits UART mode.
6	EVEN	R/W	Parity (For UART) 0: Odd 1: Even Selects even or odd parity. "0" :odd parity, "1" : even parity The parity bit can be used only in the 7-bit or 8-bit UART mode.
5	PE	R/W	Adding parity (for UART) 0: Disabled 1: Enabled Controls enabling / disabling parity The parity bit can be used only in the 7-bit or 8-bit UART mode.
4	OERR	R	Overrun error flag (Note) 0: Normal operation 1: Error
3	PERR	R	Parity / Underrun error flag (Note) 0: Normal operation 1: Error
2	FERR	R	Framing error flag (Note) 0: Normal operation 1: Error
1	SCLKS	R/W	Selecting input clock edge (For I/O Interface) Set to "0" in the clock output mode. 0: Data in the transmit buffer is sent to TXDx pin one bit at a time on the falling edge of SCLKx. Data from RXDx pin isrecieved in the recieve buffer one bit at a time on the rising edge of SCLKx. In this case, the SCLKx starts from high level. 1: Data in the transmit buffer is sent to TXDx pin one bit at a time on the rising edge of SCLKx. Data from RXDx pin isrecieved in the recieve buffer one bit at a time on the falling edge of SCLKx. In this case, the SCLKx starts from low level.
0	IOC	R/W	Selecting clock (For I/O Interface) 0: Baud rate generator 1: SCLK pin input

12.4.5 SCxMOD0 (Mode Control Register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TB8	CTSE	RXE	WU	SM		SC	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TB8	R/W	Transmit data bit 8 (For UART) Writes the 9th bit of transmit data in the 9 bits UART mode.
6	CTSE	R/W	Handshake function control (For UART) 0: CTS disabled 1: CTS enabled Controls handshake function. Setting "1" enables handshake function using CTS pin.
5	RXE	R/W	Receive control (Note1) (Note2) 0: Disabled 1: Enabled
4	WU	R/W	Wake-up function (For UART) 0: Disabled 1: Enabled This function is available only at 9-bit UART mode. In other mode, this function has no meaning. When it is set to be enabled, Interrupt occurs only when RB9 = "1" at 9-bit in the UART mode.
3-2	SM[1:0]	R/W	Specifies transfer mode. 00: I/O interface mode 01: 7-bit length UART mode 10: 8-bit length UART mode 11: 9-bit length UART mode
1-0	SC[1:0]	R/W	Serial transfer clock (For UART) 00: Timer TB 9OUT Refer to Table 12-1. 01: Baud rate generator 10: Internal clock fsys 11: External clock (SCLK input) (As for the I/O interface mode, the serial transfer clock can be set in the control register (SCxCR).

Note 1: **Set <RXE> after specifying each of mode registers (SCxMOD0, SCxMOD1, SCxMOD2).**

Note 2: **Do not clear SCxMOD0<RXE> when data is being received.**

12.4.6 SCxMOD1 (Mode Control Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	I2SC	FDPX		TXE	SINT			-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	I2SC	R/W	IDLE 0: Stop 1: Operate Specifies the IDLE mode operation.
6-5	FDPX[1:0]	R/W	Transfer mode setting 00: Transfer prohibited 01: Half duplex (Receive) 10: Half duplex (Transmit) 11: Full duplex Configures the transfer mode in the I/O interface mode. Also configures the FIFO if it is enabled. In the UART mode, it is used only to specify the FIFO configuration.
4	TXE	R/W	Transmit control (Note1) (Note2) 0: Disabled 1: Enabled This bit enables transmission and is valid for all the transfer modes.
3-1	SINT[2:0]	R/W	Interval time of continuous transmission (For I/O interface) 000: None 001: 1SCLK 010: 2SCLK 011: 4SCLK 100: 8SCLK 101: 16SCLK 110: 32SCLK 111: 64SCLK This parameter is valid only for the I/O interface mode when SCLK pin output is selected. In other modes, this function has no meaning. Specifies the interval time of continuous transmission when double buffering or FIFO is enabled in the I/O interface mode.
0	−	R/W	Write to "0".

Note 1: **Specify all the modes first and then enable the <TXE> bit.**

Note 2: **Do not stop the transmit operation (by setting <TXE> = "0") when data is being transmitted.**

Note 3: **In the DMA transfer using transmit / receive interrupt of SIO, the full duplex transmission can not be used.**

Note 4: **In case that SCxEN<SIOE>="0" (Stop SIO operation) or the operation mode is changed to IDLE mode with SCxMOD1<I2SC>="0" (Stop SIO operation in IDLE mode), SCxTFC is initiailzed again.**

12.4.7   SCxMOD2 (Mode Control Register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBEMP	RBFLL	TXRUN	SBLEN	DRCHG	WBUF	SWRST	
After reset	1	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TBEMP	R	Transmit buffer empty flag. 0: Full 1: Empty If double buffering is disabled, this flag is insignificant. This flag shows that the transmit double buffers are empty. When data in the transmit double buffers is moved to the transmit shift register and the double buffers are empty, this bit is set to "1". Writing data again to the double buffers sets this bit to "0".
6	RBFLL	R	Receive buffer full flag. 0: Empty 1: Full If double buffering is disabled, this flag is insignificant. This is a flag to show that the receive double buffers are full. When a receive operation is completed and received data is moved from the receive shift register to the receive double buffers, this bit changes to "1" while reading this bit changes it to "0".
5	TXRUN	R	In transmission flag 0: Stop 1: Operate This is a status flag to show that data transmission is in progress. <TXRUN> and <TBEMP> bits indicate the following status. | <TXRUN>	<TBEMP>	Status | |---|---|---| | 1 | − | Transmission in progress | | 0 | 1 | Transmission completed | | | 0 | Wait state with data in transmit buffer. |
4	SBLEN	R/W	STOP bit (For UART) 0: 1-bit 1: 2-bit This specifies the length of transmission stop bit in the UART mode. On the receive side, the decision is made using only a single bit regardless of the <SBLEN> setting.
3	DRCHG	R/W	Setting transfer direction 0: LSB first 1: MSB first Specifies the direction of data transfer in the I/O interface mode. In the UART mode, set this bit to LSB first.
2	WBUF	R/W	Double buffer 0: Disabled 1: Enabled This parameter enables or disables the transmit / receive double buffers to transmit (in both SCLK output / input modes) and receive (in SCLK output mode) data in the I/O interface mode and to transmit data in the UART mode. When receiving data in the I/O interface mode (SCLK input) and UART mode, double buffering is enabled in both case that "0" or "1" is set to <WBUF> bit.

Bit	Bit Symbol	Type	Function
1-0	SWRST[1:0]	R/W	Software reset Overwriting "01" in place of "10" generates a software reset. When this software reset is executed, the following bits are initialized and the transmit/receive circuit, the transmit circuit and the FIFO become initial state (see Note1 and Note2).

Register	Bit
SCxMOD0	<RXE>
SCxMOD1	<TXE>
SCxMOD2	<TBEMP>, <RBFLL>, <TXRUN>
SCxCR	<OERR>, <PERR>, <FERR>

Note 1: **While data transmission is in progress, any software reset operation must be executed twice in succession.**

Note 2: **A software reset requires 2 clocks-duration at the time between the end of recognition and the start of execution of software reset instruction.**

12.4.8 SCxBRCR (Baud Rate Generator Control Register), SCxBRADD (Baud Rate Generator Control Register 2)

The division ratio of the baud rate generator can be specified in the registers shown below.

SCxBRCR

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	BRADDE	BRCK		BRS			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	−	R/W	Write to "0".
6	BRADDE	R/W	N + (16 − K)/16 divider function (For UART) 0: Disabled 1: Enabled This division function can only be used in the UART mode.
5-4	BRCK[1:0]	R/W	Select input clock to the baud rate generator. 00: φT1 01: φT4 10: φT16 11: φT64
3-0	BRS[3:0]	R/W	Division ratio "N" 0000: 16 0001: 1 0010: 2 : 1111: 15

SCxBRADD

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	BRK			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-0	BRK[3:0]	R/W	Specify K for "N + (16 − K)/16" division (For UART) 0000: Prohibited 0001: K = 1 0010: K = 2 : 1111: K = 15

Table 12-2 lists the setting of baud rate generator division ratio.

Table 12-2 Setting division ratio

	<BRADDE> = "0"	<BRADDE> = "1" (Note1) (Only UART mode)
<BRS>	Specify "N" (Note2) (Note3)	
<BRK>	No setting required	Specify "K" (Note4)
Division ratio	Divide by N	$N + \dfrac{(16 - K)}{16}$ division

Note 1: **To use the "N + (16 - K)/16" division function, be sure to set <BRADDE> to "1" after setting the K value to <BRK>. The "N + (16 - K)/16" division function can only be used in the UART mode.**

Note 2: **As a division ratio, 1 ("0001") or 16 ("0000") can not be applied to N when using the "N + (16 - K)/16" division function in the UART mode.**

Note 3: **The division ratio "1" of the baud rate generator can be specified only when the double buffering is used in the I/O interface mode.**

Note 4: **Specifying "K = 0" is prohibited.**

12.4.9 SCxFCNF (FIFO Configuration Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	RFST	TFIE	RFIE	RXTXCNT	CNFG
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7-5	-	R/W	Be sure to write "000".
4	RFST	R/W	Bytes used in RX FIFO 0: Maximum 1: Same as FILL level of RX FIFO When RX FIFO is enabled, the number of RX FIFO bytes to be used is selected (Note1) 0: The maximum number of bytes of the FIFO configured (see also <CNFG>). 1: Same as the fill level for receive interrupt generation specified by SCxRFC <RIL[1:0]>
3	TFIE	R/W	TX interrupt for TX FIFO 0:Disabled 1:Enabled When TX FIFO is enabled, transmit interrupts are enabled or disabled by this parameter.
2	RFIE	R/W	RX interrupt for RX FIFO 0:Disabled 1:Enabled When RX FIFO is enabled, receive interrupts are enabled or disabled by this parameter.
1	RXTXCNT	R/W	Automatic disable of RXE / TXE 0: None 1: Auto disabled Controls automatic disabling of transmission and reception. Setting "1" enables to operate as follows Half duplex RX: When receive shift register, the receive buffer and the RX FIFO are filled, SCxMOD0<RXE> is automatically set to "0" to inhibit further reception. Half duplex TX: When the TX FIFO, the transmit buffer and the transmit shift register is empty, SCxMOD1<TXE> is automatically set to "0" to inhibit further transmission. Full duplex: When either of the above two conditions is satisfied, TXE/RXE are automatically set to "0" to inhibit further transmission and reception.
0	CNFG	R/W	Enables FIFO 0: Disabled 1: Enabled Enabled bit for FIFO. (note2) If <CNFG> is set to "1", the SCxMOD1 <FDPX[1:0]> setting automatically configures FIFO as follows: Half duplex RX: RX FIFO 4 bytes Half duplex TX: TX FIFO 4 bytes Full duplex: RX FIFO 2 bytes + TX FIFO 2 bytes

Note 1: **Regarding TX FIFO, the maximum number of bytes being configured is always available. The available number of bytes is the bytes already written to the TX FIFO.**

Note 2: **The FIFO can not use in 9bit UART mode.**

Note 3: **In the DMA transfer using transmit / receive interrupt of SIO, the FIFO function can not be used.**

12.4.10 SCxRFC (RX FIFO Configuration Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RFCS	RFIS	-	-	-	-	RIL	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	RFCS	W	RX FIFO clear (Note1) 1: Clear When SCxRFC<RFCS> is set to "1", the receive FIFO is cleared and SCxRST<RLVL> is "000". And also the read pointer is initialized.
6	RFIS	R/W	Select interrupt generation condition 0: when the data reaches to the specified fill level. 1: when the data reaches to the specified fill level or the data exceeds the specified fill level at the time data is read.
5-2	−	R	Read as "0".
1-0	RIL[1:0]	R/W	FIFO fill level to generate RX interrupts <table><tr><td></td><td>Half duplex</td><td>Full duplex</td></tr><tr><td>00</td><td>4 bytes</td><td>2 bytes</td></tr><tr><td>01</td><td>1 byte</td><td>1 byte</td></tr><tr><td>10</td><td>2 bytes</td><td>2 bytes</td></tr><tr><td>11</td><td>3 bytes</td><td>1 byte</td></tr></table>

Note 1: **To use TX/RX FIFO buffer, TX / RX FIFO must be cleared after setting the SIO transfer mode (half duplex / full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").**

Note 2: **DMA transfer is not started by an interrupt generated in the fill level of FIFO.**

12.4.11 SCxTFC (TX FIFO Configuration Register) (Note2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TFCS	TFIS	-	-	-	-	TIL	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TFCS	W	TX FIFO clear (Note1) 1: Clear When SCxTST<TFCS> is set to "1", the transmit FIFO is cleared and SCxTST<TLVL> is "000". And also the write pointer is initialized.
6	TFIS	R/W	Selects interrupt generation condition. 0: An interrupt is generated when the data reaches to the specified fill level. 1: An interrupt is generated when the data reaches to the specified fill level or the data can not reach the specified fill level at the time data is read.
5-2	−	R	Read as "0".
1-0	TIL[1:0]	R/W	Fill level which transmit interrupt is occurred.<table><tr><td></td><td>Half duplex</td><td>Full duplex</td></tr><tr><td>00</td><td>Empty</td><td>Empty</td></tr><tr><td>01</td><td>1 byte</td><td>1 byte</td></tr><tr><td>10</td><td>2 bytes</td><td>Empty</td></tr><tr><td>11</td><td>3 bytes</td><td>1 byte</td></tr></table>

Note 1: **To use TX/RX FIFO buffer, TX/RX FIFO must be cleared after setting the SIO transfer mode (half duplex/full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").**

Note 2: **After you perform the following operations, configure the SCxTFC register again.**

SCxEN<SIOE> = "0" (SIO operation stop)

Conditions are as follows:SCxMOD1<I2SC> = "0" (operation is prohibited in IDLE mode) and releasing the low power consumption mode which is started by the WFI (Wait For Interrupt) instruction.

Note 3: **DMA transfer is not started by an interrupt generated in the fill level of FIFO.**

12.4.12   SCxRST (RX FIFO Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ROR	-	-	-	-	RLVL		
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	ROR	R	RX FIFO Overrun (Note) 0: Not generated 1: Generated
6-3	−	R	Read as "0".
2-0	RLVL[2:0]	R	Status of RX FIFO fill level 000: Empty 001: 1 byte 010: 2 bytes 011: 3 bytes 100: 4 bytes

Note: **The <ROR> bit is cleared to "0" when receive data is read from the SCxBUF register.**

12.4.13 SCxTST (TX FIFO Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TUR	-	-	-	-	TLVL		
After reset	1	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TUR	R	TX FIFO Under run (Note) 0: Not generated 1: Generated
6-3	−	R	Read as "0".
2-0	TLVL[2:0]	R	Status of TX FIFO level 000: Empty 001: 1 byte 010: 2 byte 011: 3 byte 100: 4 byte

Note: **The <TUR> bit is cleared to "0" when transmit data is written to the SCxBUF register.**

12.5 Operation in Each Mode

Table 12-3 shows the modes and data format.

Table 12-3 Mode and Data format

Mode	Mode type	Data length	Transfer direction	Specifies whether to use parity bits	STOP bit length (Transmit)
Mode 0	Synchronous communication mode (IO interface mode)	8 bit	LSB first / MSB first	-	-
Mode 1	Asynchronous communication mode (UART mode)	7 bit	LSB first	o	1 bit or 2 bit
Mode 2		8 bit		o	
Mode 3		9bit		×	

Mode 0 is synchronous communication and can be used to extend I/O. This mode transmits and receives data in synchronization with SCLK. SCLK can be used for both input and output.

The direction of data transfer can be selected from LSB first and MSB first. This mode is not allowed either to add a parity or STOP bits.

The mode 1, mode 2 and mode 3 are asynchronous modes and the transfer direction is fixed to the LSB first.

Parity bits can be added in the mode 1 and mode 2. The mode 3 has a wake-up function in which the master controller can start up slave controllers via the serial link (multi-controller system).

STOP bit in transmission can be selected from 1 bit and 2 bits. The STOP bit length in reception is fixed to a one bit.

12.6   Data Format

12.6.1   Data Format List
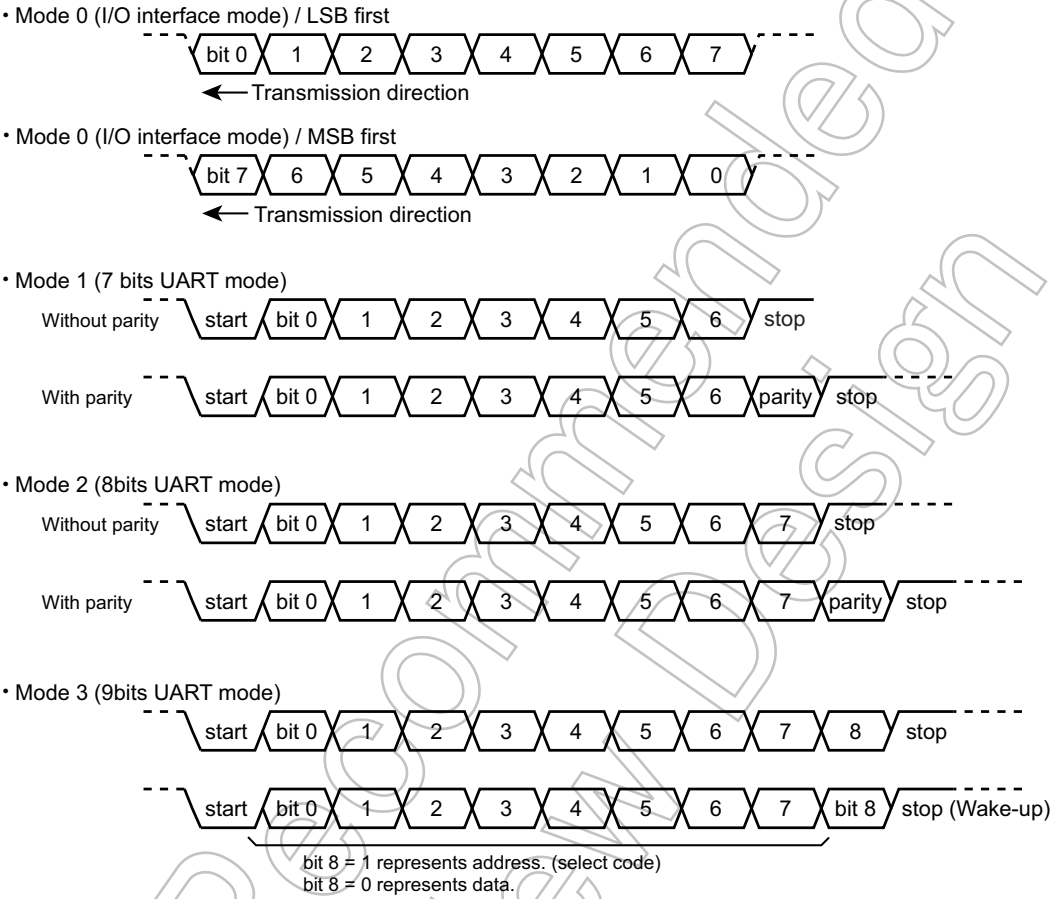
Figure 12-2 shows data format.

Figure 12-2 Data Format

12.6.2   Parity Control

The parity bit can be added only in the 7- or 8-bit UART mode.

Setting "1" to SCxCR<PE> enables the parity.

The SCxCR<EVEN> selects either even or odd parity.

12.6.2.1   Transmission

Upon data transmission, the parity control circuit automatically generates the parity with the data in the transmit buffer.

After data transmission is complete, the parity bit will be stored in SCxBUF<TB7> in the 7-bit UART mode SCxMOD<TB8> in the 8-bit UART mode.

The <PE> and <EVEN> settings must be completed before data is written to the transmit buffer.

12.6.2.2   Receiving Data

If the received data is moved from the receive shift register to the receive buffer, a parity is generated.

In the 7-bit UART mode, the generated parity is compared with the parity stored in SCxBUF<RB7>, while in the 8-bit UART mode, it is compared with the one in SCxCR<RB8>.

If there is any difference, a parity error occurs and the <PERR> of the SCxCR register is set to "1".

In use of the FIFO, <RERR> indicates that a parity error was generated in one of the received data.

12.6.3   STOP Bit Length

The length of the STOP bit in the UART transmission mode can be selected from one bit or two bits by setting the SCxMOD2<SBLEN>. The length of the STOP bit data is determined as one-bit when it is received regardless of the setting of this bit.

12.7 Clock Control

12.7.1 Prescaler

There is a 7-bit prescaler to divide a prescaler input clock φT0 by 2, 8, 32 and 128.

Use the CGSYSCR register in the clock / mode control block to select the input clock φT0 of the prescaler.

The prescaler becomes active only when the baud rate generator is selected as a transfer clock by SCxMOD0<SC[1:0]> = "01".

Table 12-4 (operation frequency 64MHz), Table 12-5 (operation frequency 48MHz) and Table 12-6 (operation frequency 32MHz) show the resolution of the input clock to the baud rate generator.

Table 12-4 Clock resolution to the Baud Rate Generator fc = 64 MHz, fs = 32.768kHz

φT0 selection CGSYSCR <FPSEL1>	Peripheral clock selection CGSYSCR <FPSEL0>	Clock gear value CGSYSCR <GEAR[2:0]>	Prescaler clock selection CGSYSCR <PRCK[2:0]>	Prescaler output clock resolution			
				φT1	φT4	φT16	φT64
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0312 μs)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			001 (fperiph/2)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			010 (fperiph/4)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			011 (fperiph/8)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.0 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			001 (fperiph/2)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			010 (fperiph/4)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			011 (fperiph/8)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0μs)
			100 (fperiph/16)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.0 μs)
			101 (fperiph/32)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0μs)	$fc/2^{14}$ (256.0 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			001 (fperiph/2)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (32.0 μs)
			010 (fperiph/4)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0μs)
			011 (fperiph/8)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.0 μs)
			100 (fperiph/16)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0μs)	$fc/2^{14}$ (256.0 μs)
			101 (fperiph/32)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.0 μs)	$fc/2^{15}$ (512.0 μs)

Table 12-4 Clock resolution to the Baud Rate Generator fc = 64 MHz, fs = 32.768kHz

φT0 selection CGSYSCR <FPSEL1>	Peripheral clock selection CGSYSCR <FPSEL0>	Clock gear value CGSYSCR <GEAR[2:0]>	Prescaler clock selection CGSYSCR <PRCK[2:0]>	Prescaler output clock resolution			
				φT1	φT4	φT16	φT64
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (00312 μs)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		100 (fc/2)	000 (fperiph/1)	−	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		101 (fc/4)	000 (fperiph/1)	−	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)
			001 (fperiph/2)	−	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		110 (fc/8)	000 (fperiph/1)	−	−	$fc/2^5$ (0.8 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	−	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	−	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
1	*	*	*	$fs/2$ (61μs)	$fs/2^3$ (244 μs)	$fs/2^5$ (977 μs)	$fs/2^7$ (3.91 ms)

Note 1: **The prescaler output clock φTn must be selected so that the relationship "φTn ≤ fsys/2" is satisfied (so that φTn is slower than fsys).**

Note 2: **Do not change the clock gear while SIO is operating.**

Note 3: **The "−" indicates that the setting is prohibited and the "*" indicates don't care in the above table.**

2023/07/21

Table 12-5 Clock resolution to the Baud Rate Generator fc = 48 MHz, fs = 32.768kHz

φT0 selection CGSYSCR <FPSEL1>	Peripheral clock selection CGSYSCR <FPSEL0>	Clock gear value CGSYSCR <GEAR[2:0]>	Prescaler clock selection CGSYSCR <PRCK[2:0]>	Prescaler output clock resolution			
				φT1	φT4	φT16	φT64
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0417 μs)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0833 μs)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
			011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.0833 μs)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			001 (fperiph/2)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
			010 (fperiph/4)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			011 (fperiph/8)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			100 (fperiph/16)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
			101 (fperiph/32)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
			001 (fperiph/2)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^9$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			010 (fperiph/4)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			011 (fperiph/8)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
			100 (fperiph/16)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)
			101 (fperiph/32)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3μs)	$fc/2^{14}$ (341 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			001 (fperiph/2)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			010 (fperiph/4)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
			011 (fperiph/8)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)
			100 (fperiph/16)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3μs)	$fc/2^{14}$ (341 μs)
			101 (fperiph/32)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)	$fc/2^{15}$ (683 μs)

Table 12-5 Clock resolution to the Baud Rate Generator fc = 48 MHz, fs = 32.768kHz

φT0 selection CGSYSCR <FPSEL1>	Peripheral clock selection CGSYSCR <FPSEL0>	Clock gear value CGSYSCR <GEAR[2:0]>	Prescaler clock selection CGSYSCR <PRCK[2:0]>	Prescaler output clock resolution			
				φT1	φT4	φT16	φT64
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0417 µs)	$fc/2^3$ (0.167 µs)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)
			001 (fperiph/2)	$fc/2^2$ (0.0833 µs)	$fc/2^4$ (0.333 µs)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)
			010 (fperiph/4)	$fc/2^3$ (0.167 µs)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)	$fc/2^9$ (10.7 µs)
			011 (fperiph/8)	$fc/2^4$ (0.333 µs)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)	$fc/2^{10}$ (21.3 µs)
			100 (fperiph/16)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)	$fc/2^9$ (10.7 µs)	$fc/2^{11}$ (42.7 µs)
			101 (fperiph/32)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)	$fc/2^{10}$ (21.3 µs)	$fc/2^{12}$ (85.3 µs)
		100 (fc/2)	000 (fperiph/1)	−	$fc/2^3$ (0.167 µs)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)
			001 (fperiph/2)	$fc/2^2$ (0.0833 µs)	$fc/2^4$ (0.333 µs)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)
			010 (fperiph/4)	$fc/2^3$ (0.167 µs)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)	$fc/2^9$ (10.7 µs)
			011 (fperiph/8)	$fc/2^4$ (0.333 µs)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)	$fc/2^{10}$ (21.3 µs)
			100 (fperiph/16)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)	$fc/2^8$ (10.7 µs)	$fc/2^{11}$ (42.7 µs)
			101 (fperiph/32)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)	$fc/2^{10}$ (21.3 µs)	$fc/2^{12}$ (85.3 µs)
		101 (fc/4)	000 (fperiph/1)	−	$fc/2^3$ (0.167 µs)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)
			001 (fperiph/2)		$fc/2^4$ (0.333 µs)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)
			010 (fperiph/4)	$fc/2^3$ (0.167 µs)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)	$fc/2^9$ (10.7 µs)
			011 (fperiph/8)	$fc/2^4$ (0.333 µs)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)	$fc/2^{10}$ (21.3 µs)
			100 (fperiph/16)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)	$fc/2^9$ (10.7 µs)	$fc/2^{11}$ (42.7 µs)
			101 (fperiph/32)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)	$fc/2^{10}$ (21.3 µs)	$fc/2^{12}$ (85.3 µs)
		110 (fc/8)	000 (fperiph/1)	−	−	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)
			001 (fperiph/2)	−	$fc/2^4$ (0.333 µs)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)
			010 (fperiph/4)	−	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)	$fc/2^9$ (10.7 µs)
			011 (fperiph/8)	$fc/2^4$ (0.333 µs)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)	$fc/2^{10}$ (21.3 µs)
			100 (fperiph/16)	$fc/2^5$ (0.667 µs)	$fc/2^7$ (2.67 µs)	$fc/2^9$ (10.7 µs)	$fc/2^{11}$ (42.7 µs)
			101 (fperiph/32)	$fc/2^6$ (1.33 µs)	$fc/2^8$ (5.33 µs)	$fc/2^{10}$ (21.3 µs)	$fc/2^{12}$ (85.3 µs)
1	*	*	*	fs/2 (61µs)	$fs/2^3$ (244 µs)	$fs/2^5$ (977 µs)	$fs/2^7$ (3.91 ms)

Note 1: **The prescaler output clock φTn must be selected so that the relationship "φTn ≤ fsys/2" is satisfied (so that φTn is slower than fsys).**

Note 2: **Do not change the clock gear while SIO is operating.**

Note 3: **The "−" indicates that the setting is prohibited and the "*" indicates don't care in the above table.**

Table 12-6 Clock resolution to the Baud Rate Generator fc = 32 MHz, fs = 32.768kHz

φT0 selection CGSYSCR \<FPSEL1\>	Peripheral clock selection CGSYSCR \<FPSEL0\>	Clock gear value CGSYSCR \<GEAR[2:0]\>	Prescaler clock selection CGSYSCR \<PRCK[2:0]\>	Prescaler output clock resolution			
				φT1	φT4	φT16	φT64
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0625 µs)	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
			001 (fperiph/2)	$fc/2^2$ (0.125 µs)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
			010 (fperiph/4)	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)
			011 (fperiph/8)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)
			100 (fperiph/16)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)	$fc/2^{11}$ (64.0 µs)
			101 (fperiph/32)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)	$fc/2^{12}$ (128.0 µs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.125 µs)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
			001 (fperiph/2)	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)
			010 (fperiph/4)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)
			011 (fperiph/8)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)	$fc/2^{11}$ (64.0 µs)
			100 (fperiph/16)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)	$fc/2^{12}$ (128.0 µs)
			101 (fperiph/32)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)	$fc/2^{11}$ (64.0 µs)	$fc/2^{13}$ (256.0 µs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)
			001 (fperiph/2)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)
			010 (fperiph/4)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)	$fc/2^{11}$ (64.0 µs)
			011 (fperiph/8)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)	$fc/2^{12}$ (128.0 µs)
			100 (fperiph/16)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)	$fc/2^{11}$ (64.0 µs)	$fc/2^{13}$ (256.0 µs)
			101 (fperiph/32)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)	$fc/2^{12}$ (128.0 µs)	$fc/2^{14}$ (512.0 µs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)
			001 (fperiph/2)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)	$fc/2^{11}$ (64.0 µs)
			010 (fperiph/4)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)	$fc/2^{12}$ (128.0 µs)
			011 (fperiph/8)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16.0 µs)	$fc/2^{11}$ (64.0 µs)	$fc/2^{13}$ (256.0 µs)
			100 (fperiph/16)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32.0 µs)	$fc/2^{12}$ (128.0 µs)	$fc/2^{14}$ (512.0 µs)
			101 (fperiph/32)	$fc/2^9$ (16.0 µs)	$fc/2^{11}$ (64.0 µs)	$fc/2^{13}$ (256.0 µs)	$fc/2^{15}$ (1024 µs)

12.7.2 Serial Clock Generation Circuit

The serial clock circuit is a block to generate transmit and receive clocks (SIOCLK) and consists of the circuits in which clocks can be selected by the settings of the baud rates generator and modes.

12.7.2.1 Baud Rate Generator

The baud rate generator generates transmit and receive clocks to determine the serial channel transfer rate.

(1) Buad Rate Generator input clock

The input clock of the baud rate generator is selected from the prescaler outputs divided by 2, 8, 32 and 128.

This input clock is selected by setting the SCxBRCR<BRCK>.

(2) Baud Rate Generator output clock

The frequency division ratio of the output clock in the baud rate generator is set by SCxBRCR and SCxBRADD.

The following frequency divide ratios can be used; 1/N frequency division in the I/O interface mode, either 1/N or N + (16-K)/16 in the UART mode.

The table below shows the frequency division ratio which can be selected.

Mode	Divide Function Setting SCxBRCR<BRADDE>	Divide by N SCxBRCR<BRS>	Divide by K SCxBRADD<BRK>
I/O interface	Divide by N	1 to 16 (Note)	-
UART	Divide by N	1 to 16	-
	N + (16-K)/16 division	2 to 15	1 to 15

Note: **1/N (N=1)frequency division ratio can be used only when a double buffer is enabled.**

12.7.2.2 Clock Selection Circuit

A clock can be selected by setting the modes and the register.

Modes can be specified by setting the SCxMOD0<SM>.

The input clock in I/O interface mode is selected by setting SCxCR.

The clock in UART mode is selected by setting SCxMOD0<SC>.

(1) Transfer Clock in I/O interface mode

Table 12-7 shows clock selection in I/O interface mode.

Table 12-7 Clock selection in I/O interface Mode

Mode SCxMOD0\<SM>	Input / Output selection SCxCR\<IOC>	Clock edge selection SCxCR\<SCLKS>	Clock of use
I/O interface mode	SCLK output	Set to "0" (Fixed to the rising edge)	Divided by 2 of the baud rate generator output
	SCLK input	Rising edge	SCLK input rising edge
		Falling edge	SCLK input falling edge

To get the highest baud rate, the baud rate generator must be set as below.

Note: **When deciding clock settings, make sure that AC electrical character is satisfied.**

- Clock / mode control block settings
  - fc = 40MHz
  - fgear = 40MHz (CGSYSCR\<GEAR[2:0]> = "000" : fc selected)
  - φT0=40Mhz (CGSYSCR\<PRCK[2:0]>="000" : 1 division ratio)

- SIO settings (if double buffer is used)
  - Clock (SCxBRCR\<BRCK[1:0]> = "00" : φT1 selected) = 20MHz
  - Divided clock frequency (SCxBRCR\<BRS[3:0]> = "0001":1 division ratio) = 20MHz

  1 division ratio can be selected if double buffer is used. In this case, baud rate is 10Mbps because 20MHz is divided by 2.

- SIO settings (if double buffer is not used)
  - Clock (SCxBRCR\<BRCK[1:0]> = "00":φT1 selected) = 20MHz
  - Divided clock frequency (SCxBRCR\<BRS[3:0]> = "0010" : 2 division ratio) = 10MHz

  2 division ratio is the highest if double buffer is not used. In this case, baud rate is 5Mbps because 10MHz is divided by 2.

To use SCLK input, the following conditions must be satisfied.

- If double buffer is used
  - SCLK cycle > 6/fsys

  The highest buad rate is less than 40 ÷ 6 = 6.66 Mbps.

- If double buffer is not used
  - SCLK cycle > 8/fsys

  The highest baud rate is less than 40 ÷ 8 = 5.0 Mbps.

(2)   Transfer clock in the UART mode

Table 12-8 shows the clock selection in the UART mode. In the UART mode, selected clock is divided by 16 in the receive counter or the transmit counter before use.

Table 12-8 Clock selection in UART Mode

Mode SCxMOD0<SM>	Clock selection SCxMOD0<SC>
UART Mode	Timer output
	Baud rate generator
	fsys
	SCLK input

The examples of baud rate in each clock settings.

- If baud rate generator is used.
    - fc = 40MHz
    - fgear = 40MHz (CGSYSCR<GEAR[2:0]> = "000" : fc selected)
    - φT0 = 40MHz (CGSYSCR<PRCK[2:0]> = "000" : 1 division ratio)
    - Clock = φT1 = 20MHz (SCxBRCR<BRCK[1:0]> = "00" : φT1 selected)

    The highest baud rate is 1.25MHz because 20MHz is divided by 16.

    Table 12-9 shows examples of baud rate when the baud rate generator is used with the following clock settings.

    - fc = 9.8304MHz
    - fgear = 9.8304MHz (CGSYSCR<GEAR[2:0]> = "000" : fc selected)
    - ΦT0 = 4.9152MHz (CGSYSCR<PRCK[2:0]> = "001" : 2 division ratio)

Table 12-9 Example of UART Mode Baud Rate (Using the Baud Rate Generator)

fc [MHz]	Division ratio N (SCxBRCR<BRS[3:0]>)	φT1 (fc/4)	φT4 (fc/16)	φT16 (fc/64)	φT64 (fc/256)
9.830400	2	76.800	19.200	4.800	1.200
	4	38.400	9.600	2.400	0.600
	8	19.200	4.800	1.200	0.300
	16	9.600	2.400	0.600	0.150

Unit : kbps

- If the SCLK input is used

    To use SCLK input, the following conditions must be satisfied.

    - SCLK cycle > 2/fsys

    The highest baud rate must be less than $40 \div 2 \div 16 = 1.25$ Mbps.

- If fsys is used

    Since the highest value of fsys is 40MHz, the highest baud rate is $40 \div 16 = 2.5$Mbps.

- If timer output is used

    To enable the timer output, the following condition must be set: a timer flip-flop output inverts when the value of the counter and that of TBxRG1 match. The SIOCLK clock frequency is "Setting value of TBxRG1 × 2".

    Baud rate can be obtained by using the following formula.

Baud rate calculation

$$\text{Transfer rate} = \frac{\text{Clock frequency selected by CGSYSCR<PRCK[1:0]>}}{(\text{TBxRG1} \times 2) \times 2 \times 16}$$

In the case the timer prescaler clock fT1(2 divition ratio) is selected

One clock cycle is a period that the timer flip-flop is inverted twice.

Table 12-10 shows the examples of baud rates when the timer output is used with the following clock settings.

- fc = 32MHz / 9.8304MHz / 8MHz
- fgear = 32MHz / 9.8304MHz / 8MHz (CGSYSCR<GEAR[2:0]> = "000" :fc selected)
- φT0 = 16MHz / 4.9152MHz / 4MHz (CGSYSCR<PRCK[2:0]> = "001" :2 division)
- Timer count clock

    = 4MHz / 1.2287MHz / 1MHz (TBxMOD<TBCLK[1:0]> = "01" :φT1 selected)

Table 12-10 Example of UART Mode Baud Rate (Using the Timer Output)

TBxRG setting	fc		
	32MHz	9.8304MHz	8MHz
0x0001	250	76.8	62.5
0x0002	125	38.4	31.25
0x0003	-	25.6	-
0x0004	62.5	19.2	15.625
0x0005	50	15.36	12.5
0x0006	-	12.8	-
0x0008	31.25	9.6	-
0x000A	25	7.68	6.25
0x0010	15.625	4.8	-
0x0014	12.5	3.84	3.125

Unit : kbps

12.8 Transmit / Receive Buffer and FIFO

12.8.1 Configuration

Figure 12-3 shows the configuration of transmit buffer, receive buffer and FIFO.

Appropriate settings are required for using buffer and FIFO. The configuration may be predefined depending on the mode.
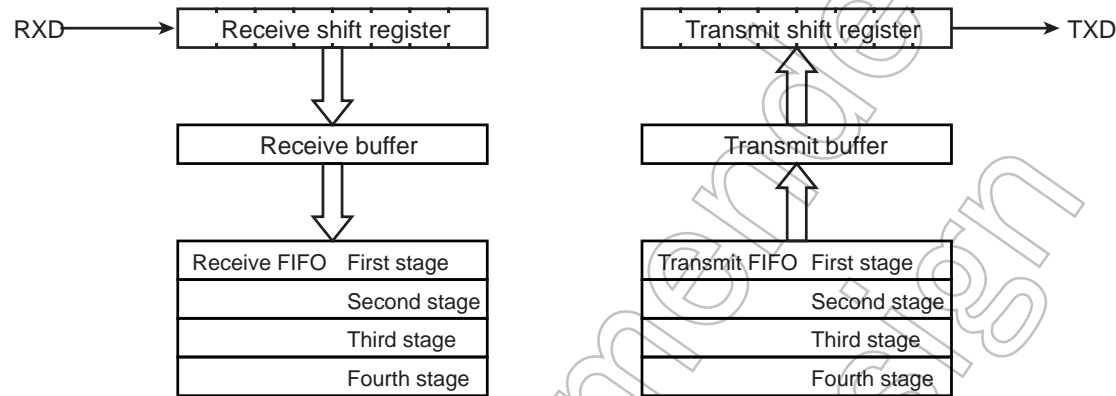
Figure 12-3 The Configuration of Buffer and FIFO

12.8.2 Transmit / Receive Buffer

Transmit buffer and receive buffer are double-buffered. The buffer configuration is specified by SCxMOD2<WBUF>.

In the case of using a receive buffer, if SCLK input is set to generate clock output in the I/O interface mode or the UART mode is selected, it's double buffered despite the <WBUF> settings. In other modes, it's according to the <WBUF> settings.

Table 12-11 shows correlation between modes and buffers.

Table 12-11 Mode and buffer Composition

Mode		SCxMOD2<WBUF>	
		"0"	"1"
UART	Transmit	Single	Double
	Receive	Double	Double
I/O interface (SCLK input)	Transmit	Single	Double
	Receive	Double	Double
I/O interface (SCLK output)	Transmit	Single	Double
	Receive	Single	Double

12.8.3 FIFO

In addition to the double buffer function above described, 4-byte FIFO can be used.

To enable FIFO, enable the double buffer by setting SCxMOD2<WBUF> to "1" and SCxFCNF<CNFG> to "1". The FIFO buffer configuration is specified by SCxMOD1<FDPX[1:0]>.

Note: **To use TX/RX FIFO buffer, TX/RX FIFO must be cleared after setting the SIO transfer mode (half duplex/ full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").**

Table 12-12 shows correction between modes and FIFO.

Table 12-12 Mode and FIFO Composition

	SCxMOD1<FDPX[1:0]>	RX FIFO	TX FIFO
Half duplex RX	"01"	4byte	-
Half duplex TX	"10"	-	4byte
Full duplex	"11"	2byte	2byte

12.9 Status Flag

The SCxMOD2 register has two types of flag. This bit is significant only when the double buffer is enabled.

<RBFLL> is a flag to show that the receive buffer is full. When one frame of data is received and the data is moved from the receive shift register to the receive buffers, this bit changes to "1" while reading this bit changes it to "0".

<TBEMP> shows that the transmit buffers are empty. When data in the transmit buffers is moved to the transmit shift register, this bit is set to "1" When data is set to the transmit buffers, the bit is cleared to "0".

12.10 Error Flag

Three error flags are provided in the SCxCR register. The meaning of the flags is changed depending on the modes. The table below shows the meaning in each mode.

These flags are cleared to "0" after reading the SCxCR register.

Mode	Flag		
	<OERR>	<PERR>	<FERR>
UART	Overrun error	Parity error	Framing error
I/O interface (SCLK input)	Overrun error	Underrun error (When using double buffer or FIFO)	Fixed to "0"
		Fixed to "0" (When a double buffer and FIFO unused)	
I/O interface (SCLK output)	Undefined	Undefined	Fixed to "0"

12.10.1 OERR Flag

In both UART and I/O interface modes, this bit is set to "1" when an error is generated by completing the reception of the next frame of receive data before the receive buffer has been read. If the receive FIFO is enabled, the received data is automatically moved to the receive FIFO and no overrun error will be generated until the receive FIFO is full (or until the usable bytes are fully occupied).

In the I/O interface with SCLK output mode, the SCLK output stops upon setting the flag.

Note:**To switch the I/O interface SCLK output mode to other modes, read the SCxCR register and clear the overrun flag.**

12.10.2　PERR Flag

This flag indicates a parity error in the UART mode and an under-run error in the I/O interface mode.

In the UART mode, <PERR> is set to "1" when the parity generated from the received data is different from the parity received.

In the I/O interface mode, <PERR> is set to "1" under the following conditions when a double buffer is enabled.

In the SCLK input mode, <PERR> is set to "1" when the SCLK is input after completing data output of the transmit shift register with no data in the transmit buffer.

In the SCLK output mode, <PERR> is set to "1" after completing output of all data and the SCLK output stops.

Note:**To switch the I/O interface SCLK output mode to other modes, read the SCxCR register and clear the underrun flag.**

12.10.3　FERR Flag

A framing error is generated if the corresponding stop bit is determined to be "0" by sampling the bit at around the center. Regardless of the stop bit length settings in the SCxMOD2<SBLEN>register, the stop bit status is determined by only 1.

This bit is fixed to "0" in the I/O interface mode.

12.11   Receive

12.11.1   Receive Counter

The receive counter is a 4-bit binary counter and is up-counted by SIOCLK.

In the UART mode, sixteen SIOCLK clock pulses are used in receiving a single data bit and the data symbol is sampled at the seventh, eighth, and ninth pulses. From these three samples, majority logic is applied to decide the received data.

12.11.2   Receive Control Unit

12.11.2.1   I/O interface mode

In the SCLK output mode with SCxCR <IOC> set to "0", the RXD pin is sampled on the rising edge of the shift clock outputted to the SCLK pin.

In the SCLK input mode with SCxCR <IOC> set to "1", the serial receive data RXD pin is sampled on the rising or falling edge of SCLK input signal depending on the SCxCR <SCLKS> setting.

12.11.2.2   UART Mode

The receive control unit has a start bit detection circuit, which is used to initiate receive operation when a normal start bit is detected.

12.11.3   Receive Operation

12.11.3.1   Receive Buffer

The received data is stored by 1 bit in the receive shift register. When a complete set of bits has been stored, the interrupt INTTRX is generated.

When the double buffer is enabled, the data is moved to the receive buffer (SCxBUF) and the receive buffer full flag (SCxMOD2<RBFLL>) is set to "1". The receive buffer full flag is "0" cleared by reading the receive buffer. The receive buffer flag does not have any value for the single buffer.
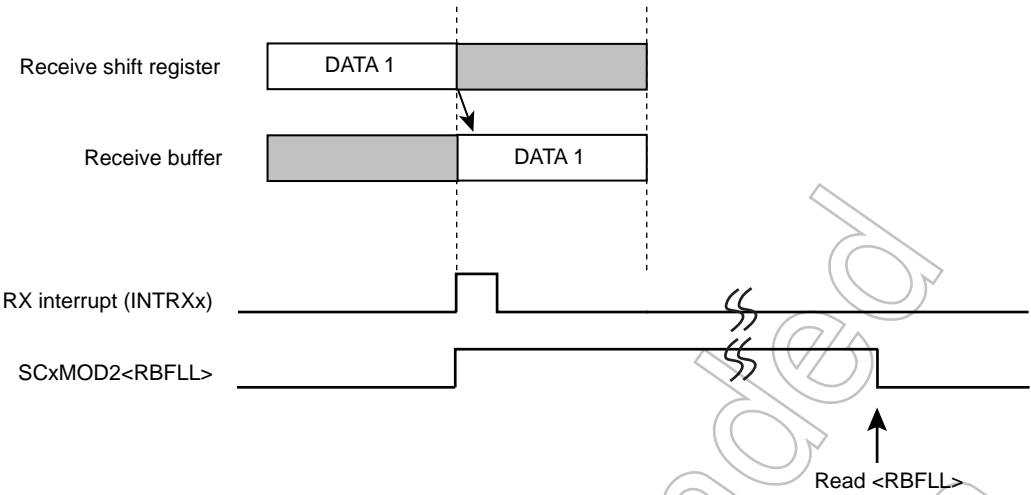
Figure 12-4 Receive Buffer Operation

12.11.3.2   Receive FIFO Operation

When FIFO is enabled, the received data is moved from receive buffer to receive FIFO and the receive buffer full flag is cleared immediately. An interrupt will be generated according to the SCxRFC<RIL> setting.

Note:**When the data with parity bit are received in UART mode by using the FIFO, the parity error flag is shown the occurring the parity error in the received data.**

The configurations and operations in the half duplex RX mode are described as follows.

SCxMOD1[6:5] =01	:Transfer mode is set to half duplex mode
SCxFCNF[4:0] = 10111	:Automatically inhibits continuous reception after reaching the fill level.
	: The number of bytes to be used in the receive FIFO is the same as the interrupt generation fill level.
SCxRFC[1:0] = 00	:The fill level of FIFO in which generated receive interrupt is set to 4-byte
SCxRFC[7:6] = 11	:Clears receive FIFO and sets the condition of interrupt generation.

After setting of the above FIFO configuration, the data reception is started by writing "1" to the SCxMOD0<RXE>. When the data is stored all in the receive shift register, receive buffer and receive FIFO, SCxMOD0<RXE> is automatically cleared and the receive operations finished.

In the above condition, if the cutaneous reception after reaching the fill level is enabled, it becomes possible to receive a data continuously by reading the data in the FIFO.
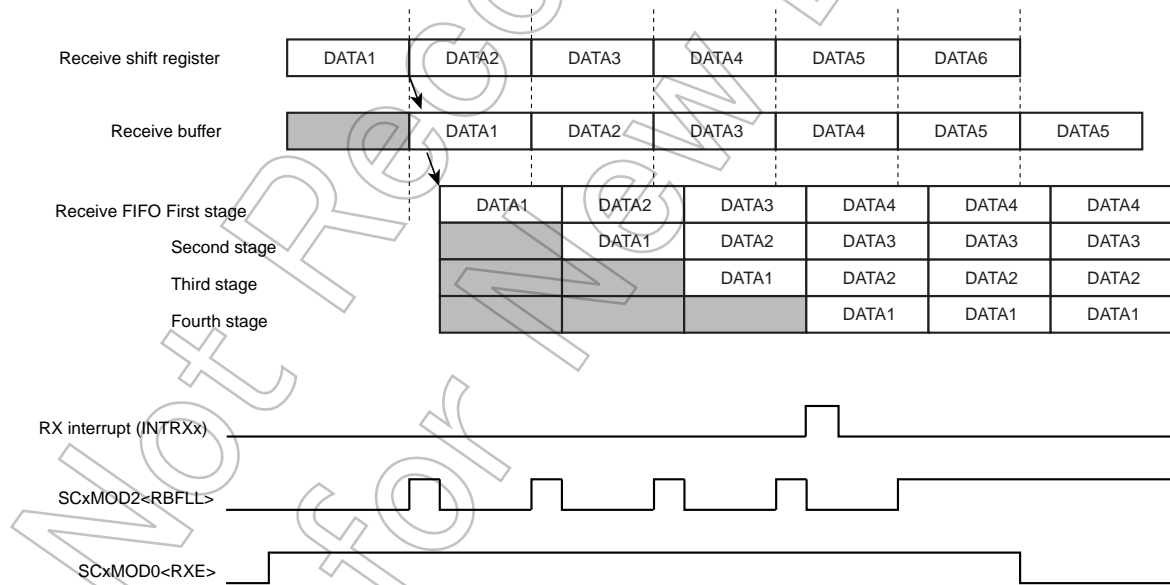
Figure 12-5 Receive FIFO Operation

12.11.3.3 I/O interface mode with SCLK output

In the I/O interface mode and SCLK output setting, SCLK output stops when all received data is stored in the receive buffer and FIFO. Thus, in this mode, the overrun error flag has no meaning.

The timing of SCLK output stop and re-output depends on receive buffer and FIFO.

(1) Case of single buffer

Stop SCLK output after receiving a data. In this mode, I/O interface can transfer each data with the transfer device by hand-shake.

When the data in a buffer is read, SCLK output restarts.

(2) Case of double buffer

Stop SCLK output after receiving the data into a receive shift register and a receive buffer.

When the data is read, SCLK output restarts.

(3) Case of FIFO

Stop SCLK output after receiving the data into a shift register, received buffer and FIFO.

When one byte data is read, the data in the received buffer is transferred into FIFO and the data in the receive shift register is transferred into the received buffer and SCLK output restarts.

And if SCxFCNF<RXTXCNT>is set to "1", SCLK stops and receive operation stops with clearing SCxMOD0<RXE> bit, too.

12.11.3.4 Read Received Data

In spite of enabling or disabling FIFO, read the received data from the receive buffer (SCxBUF).

When receive FIFO is disabled, the buffer full flag SCxMOD2<RBFLL> is cleared to "0" by this reading. In the case of the next data can be received in the receive shift register before reading a data from the receive buffer. The parity bit to be added in the 8-bit UART mode as well as the most significant bit in the 9-bit UART mode will be stored in SCxCR<RB8>.

When the receive FIFO is available, the 9-bit UART mode is prohibited because up to 8-bit data can be stored in FIFO. In the 8-bit UART mode, the parity bit is lost but parity error is determined and the result is stored in SCxCR<PERR>.

12.11.3.5 Wake-up Function

In the 9-bit UART mode, the slave controller can be operated in the wake-up mode by setting the wake-up function SCxMOD0 <WU> to "1". In this case, the interrupt INTRXx will be generated only when SCxCR<RB8> is set to "1".

12.11.3.6   Overrun Error

When FIFO is disabled, the overrun error occurs and an overrun error is without completing reading data before receiving the next data. When an overrun error occurs, a content of receive buffer and SCxCR<RB8> is not lost, but a content of receive shift register is lost.

When FIFO is enabled, overrun error is occurred and set overrun flag by no reading the data before moving the next data into received buffer when FIFO is full. In this case, the contents of FIFO are not lost.

In the I/O interface mode with SCLK output setting, the clock output automatically stops, so this flag has no meaning.

Note:**When the mode is changed from I/O interface SCLK output mode to the other mode, read
        SCxCR and clear overrun flag.**

12.12   Transmission

12.12.1   Transmission Counter

The transmit counter is a 4-bit binary counter and is counted by SIOCLK as in the case of the receive counter.

In UART mode, it generates a transmit clock (TXDCLK) on every 16th clock pulse.
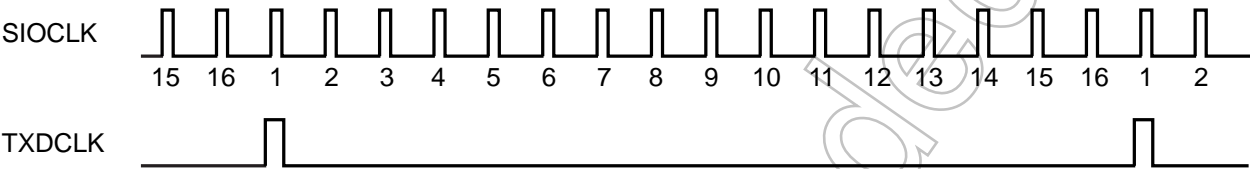
Figure 12-6 Generation of Transmission Clock in UART Mode

12.12.2   Transmission Control

12.12.2.1   I/O interface Mode

In the SCLK output mode with SCxCR<IOC> set to "0", each bit of data in the transmit buffer is output-ted to the TXD pin on the falling edge of the shift clock outputted from the SCLK pin.

In the SCLK input mode with SCxCR<IOC> set to "1", each bit of data in the transmit buffer is output-ted to the TXD pin on the rising or falling edge of the SCLK input signal according to the SCxCR<SCLKS> setting.

12.12.2.2   UART Mode

When the transmit data is written in the transmit buffer, data transmission is initiated on the rising edge of the next TXDCLK and the transmit shift clock signal is also generated.

12.12.3   Transmit Operation

12.12.3.1   Operation of Transmission Buffer

If double buffering is disabled, the CPU writes data only to Transmit shift Buffer and the transmit interrupt INTTXx is generated upon completion of data transmission.

If double buffering is enabled (including the case the transmit FIFO is enabled), data written to the transmit buffer is moved to the transmit shift register. The INTTXx interrupt is generated at the same time and the transmit buffer empty flag (SCxMOD2<TBEMP>) is set to "1". This flag indicates that the next transmit data can be written. When the next data is written to the transmit buffer, the <TBEMP> flag is cleared to "0".
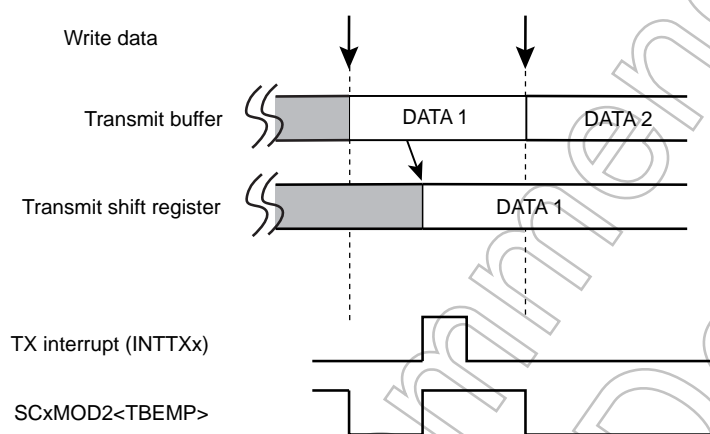
Figure 12-7 Operation of Transmission Buffer (Double buffer is enabled)

12.12.3.2   Transmit FIFO Operation

When FIFO is enabled, the maximum 5-byte data can be stored using the transmit buffer and FIFO. Once transmission is enabled, data is transferred to the transmit shift register from the transmit buffer and start transmission. If data exists in the FIFO, the data is moved to the transmit buffer immediately, and the <TBEMP> flag is cleared to "0".

Note:**To use TX FIFO buffer, TX FIFO must be cleared after setting the SIO transfer mode (half duplex/ full duplex) and enabling FIFO (SCxFCNF<CNFG>="1").**

Settings and operations to transmit 4-byte data stream by setting the transfer mode to half duplex are shown as below.

SCxMOD1[6:5] =10	:Transfer mode is set to half duplex.
SCxFCNF[4:0] = 11011	:Transmission is automatically disabled if FIFO becomes empty.
	:The number of bytes to be used in the receive FIFO is the same as the interrupt
	:generation fill level.
SCxTFC[1:0] = 00	:Sets the interrupt generation fill level to "0".
SCxTFC[7:6] = 11	:Clears receive FIFO and sets the condition of interrupt generation.
SCxFCNF[0] = 1	:Enable FIFO

After above settings are configured, data transmission can be initiated by writing 5 bytes of data to the transmit buffer or FIFO, and setting the SCxMOD1<TXE> bit to "1". When the last transmit data is moved to the transmit buffer, the transmit FIFO interrupt is generated. When transmission of the last data is completed, the clock is stopped and the transmission sequence is terminated.

Once above settings are configured, if the transmission is not set as auto disabled, the transmission should lasts writing transmit data.
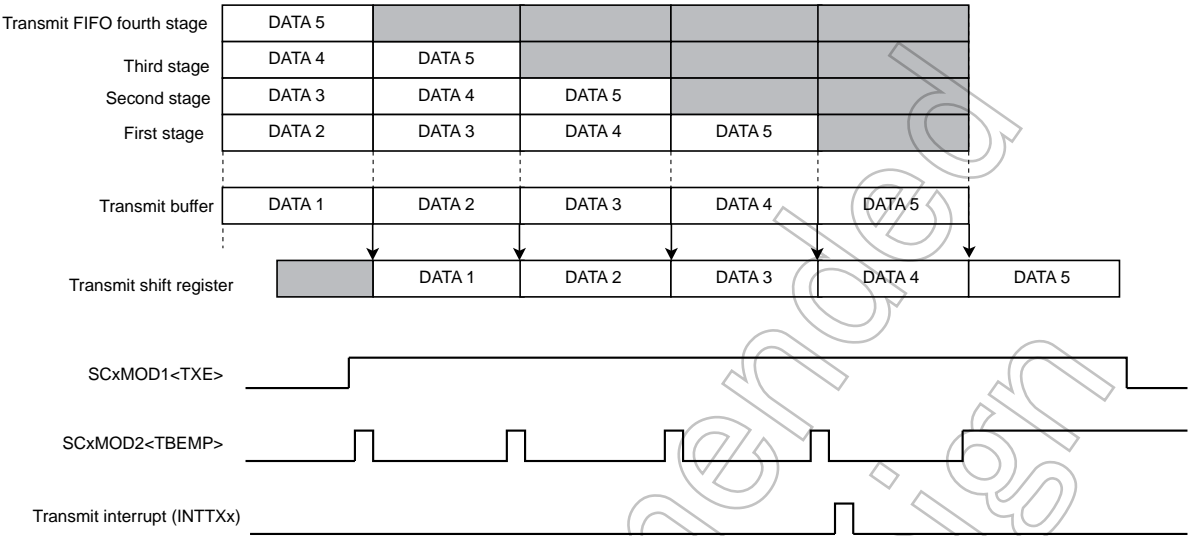
12.12.3.3   I/O interface Mode/Transmission by SCLK Output

If SCLK is set to generate clock in the I/O interface mode, the SCLK output automatically stops when all data transmission is completed and underrun error will not occur.

The timing of suspension and resume of SCLK output is different depending on the buffer and FIFO usage.

(1)   Single Buffer

The SCLK output stops each time one frame of data is transferred. Handshaking for each data with the other side of communication can be enabled. The SCLK output resumes when the next data is written in the buffer.

(2)   Double Buffer

The SCLK output stops upon completion of data transmission of the transmit shift register and the transmit buffer. The SCLK output resumes when the next data is written in the buffer.

(3)   FIFO

The transmission of all data stored in the transmit shift register, transmit buffer and FIFO is completed, the SCLK output stops. The next data is written, SCLK output resumes.

If SCxFCNF<RXTXCNT> is configured, SCxMOD0<TXE> bit is cleared at the same time as SCLK stop and the transmission stops.

12.12.3.4   Underrun Error

If the transmit FIFO is disabled in the I/O interface SCLK input mode and if no data is set in transmit buffer before the next frame clock input, which occurs upon completion of data transmission from transmit shift register, an under-run error occurs and SCxCR<PERR> is set to "1".

In the I/O interface mode with SCLK output setting, the clock output automatically stops, so this flag has no meaning/

Note:**Before switching the I/O interface SCLK output mode to other modes, read the SCxCR register and clear the underrun flag.**

12.13 Handshake Function

The function of the handshake is to enable frame-by-frame data transmission by using the CTS (Clear to send) pin and to prevent overrun errors. This function can be enabled or disabled by SCxMOD0<CTSE>.

When the $\overline{\text{CTS}}$ pin is set to "High" level, the current data transmission can be completed but the next data transmission is suspended until $\overline{\text{CTS}}$ pin returns to the "Low" level. However in this case, the INTTXx interrupt is generated in the normal timing, the next transmit data is written in the transmit buffer, and it waits until it is ready to transmit data.

Note 1: **If the $\overline{\text{CTS}}$ signal is set to "High" during transmission, the next data transmission is suspended after the current transmission is completed.**

Note 2: **Data transmission starts on the first falling edge of the TXDCLK clock after $\overline{\text{CTS}}$ is set to "Low".**

Although no $\overline{\text{RTS}}$ pin is provided, a handshake control function can easily implemented by assigning one bit of the port for the $\overline{\text{RTS}}$ function. By setting the port to "High" level upon completion of data reception (in the receive interrupt routine), the transmit side can be requested to suspend data transmission.
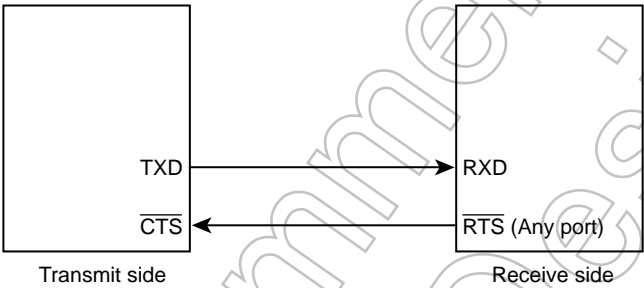
Figure 12-8 Handshake Function
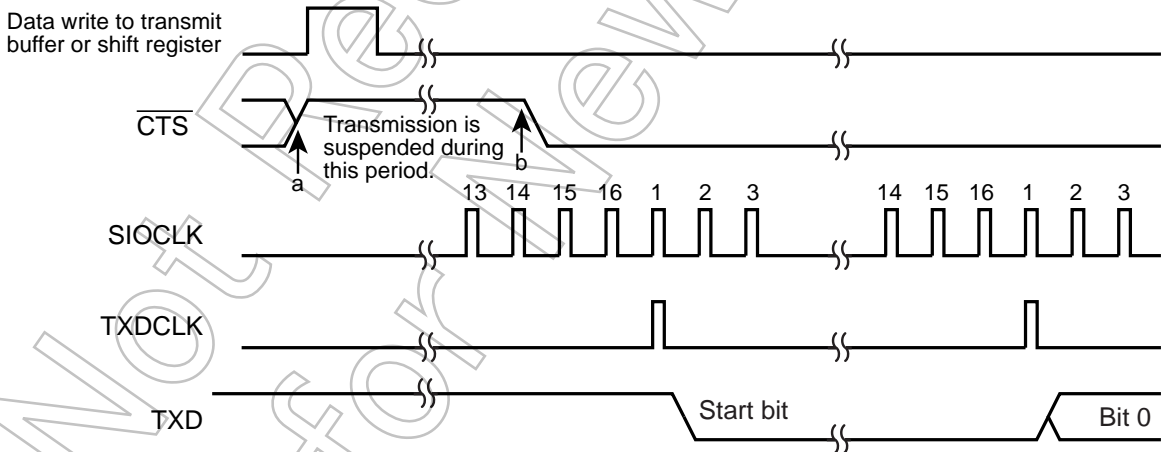
Figure 12-9 $\overline{\text{CTS}}$ Signal timing

12.14 Interrupt / Error Generation Timing

12.14.1 RX Interrupt

Figure 12-10 shows the data flow of receive operation and the route of read.

RXD ──────→ Receive shift register ──────→

If the receive buffer is emply,
the data is moved.

Receive buffer ──────→

If the RX FIFO is not full,
the data is moved.

Receive FIFO First stage
second stage
Third stage
Fourth stage ──────→

(1)Reading in the single buffer configuration :
An interrupt is generated after receiving all bits.

(2)Reading in the doule buffer configuration :
An interrupt is generated when the data is moved to
the receive buffer.

(3)Reading in use the FIFO :
An interrupt is generated
When the data is moved to the FIFO
or when reading the FIFO.

Figure 12-10 Receive Buffer / FIFO Configuration Diagram

12.14.1.1 Single Buffer / Double Buffer

RX interrupts are generated at the time depends on the transfer mode and the buffer configurations, which are given follows.

Buffer Configuration	UART modes	I/O interface modes
Single Buffer	–	Immediately after the raising / falling edge of the last SCLK (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Double Buffer	Around the center of the first stop bit	Immediately after the raising / falling edge of the last SCLK (Rising or falling is determined according to SCxCR<SCLKS> setting.) On data transfer from the shift register to the buffer by reading buffer.

Note:**Interrupts are not generated when an overrun error occurs.**

12.14.1.2 FIFO

When the FIFO is used, a receive interrupt occurs depending on the timing described in Table 12-13 and the condition specified with SCxRFC<RFIS>.

Table 12-13 Receive Interrupt Conditions in use of FIFO

SCxRFC <RFIS>	Interrupt conditions	Interrupt generation timing
"0"	When FIFO fill level (SCxRST<RLVL[2:0]>) = Receive FIFO fill level to generate receive interrupt <RIL[1:0] >	・When received data is transferred from receive buffer to receive FIFO
"1"	When FIFO fill level (SCxRST<RLVL[2:0]>) ≥ Receive FIFO fill level to generate receive interrupt <RIL[1:0]>	・When received data is transferred from receive buffer to receive FIFO ・When received data is read from receive FIFO

12.14.2   TX interrupt

Figure 12-11 shows the data flow of transmit operation and the route of read.
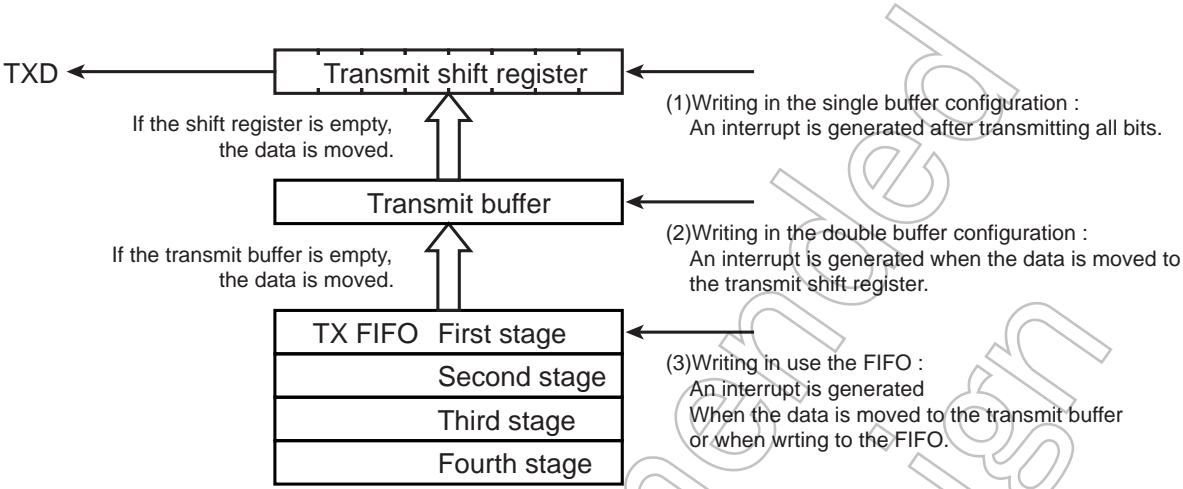
Figure 12-11 Transmit Buffer / FIFO Configuration Diagram

12.14.2.1   Single Buffer / Double Buffer

TX interrupts are generated at the time depends on the transfer mode and the buffer configurations, which are given as follows.

Buffer Configuration	UART modes	I/O interface modes
Single Buffer	Just before the stop bit is sent	Immediately after the raising / falling edge of the last SCLK (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Double Buffer	When a data is moved from the transmit buffet to the transmit shift register.	

Note: **If double buffer is enabled, a interrupt is also generated when the data is moved from the buffer to the shift register by writing to the buffer.**

12.14.2.2   FIFO

When the FIFO is used, a transmit interrupt occurs depending on the timing described in Table 12-14 and the condition specified with SCxTFC<TFIS>.

Table 12-14 Transmit Interrupt conditions in use of FIFO

SCxTFC <TFIS>	Interrupt condition	Interrupt generation timing
"0"	When FIFO fill level (SCxTST<TLVL[2:0]>) = Transmit FIFO fill level to generate transmit interrupt <TIL[1:0] >	・When transmitted data is transferred from transmit FIFO to transmit buffer
"1"	When FIFO fill level (SCxTST<TLVL[2:0]>) ≤ Transmit FIFO fill level to generate transmit interrupt <TIL[1:0] >	・When transmitted data is transferred from transmit FIFO to transmit buffer ・When transmit data is write into transmit FIFO

12.14.3   Error Generation

12.14.3.1   UART Mode

Modes	9 bits	7 bits 8 bits 7 bits + parity 8bits + parity
Framing error Overrun error	Around the center of stop bit	
Parity Error	−	Around center of parity bit

12.14.3.2   I/O Interface Mode

Overrun error	Immediately after the raising / falling edge of the last SCLK (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Underrun error	Immediately after the rising or falling edge of the next SCLK. (Rising or falling is determined according to SCxCR<SCLKS> setting.)

Note:**Over-run error and Under-run error have no meaning in SCLK output mode.**

12.15   Software Reset

Software reset is generated by writing SCxMOD2<SWRST[1:0]> as "10" followed by "01". As a result, SCxMOD0<RXE>, SCxMOD1<TXE>, SCxMOD2<TBEMP><RBFLL><TXRUN>, SCxCR<OERR><PERR> <FERR> are initialized. And the receive circuit, the transmit circuit and the FIFO become initial state. Other states are maintained.

12.16 Operation in Each Mode

12.16.1 Mode 0 (I/O Interface Mode)

Mode 0 consists of two modes, the SCLK output mode to output synchronous clock and the SCLK input mode to accept synchronous clock from an external source.

The following operational descriptions are for the case use of FIFO is disabled. For details of FIFO operation, refer to the previous sections describing receive/transmit FIFO functions.

12.16.1.1 Transmitting Data

(1) SCLK Output Mode

- If the transmit double buffer is disabled (SCxMOD2<WBUF> = "0")

  Data is output from the TXD pin and the clock is output from the SCLK pin each time the CPU writes data to the transmit buffer. When all data is output, an interrupt (INTTXx) is generated.

- If the transmit double buffer is enabled (SCxMOD2<WBUF> = "1")

  Data is moved from the transmit buffer to the transmit shift register when the CPU writes data to the transmit buffer while data transmission is halted or when data transmission from the transmit buffer (shift register) is completed. Simultaneously, the transmit buffer empty flag SCxMOD2<TBEMP> is set to "1", and the INTTXx interrupt is generated.

  When data is moved from the transmit buffer to the transmit shift register, if the transmit buffer has no data to be moved to the transmit shift register, INTTXx interrupt is not generated and the SCLK output stops.
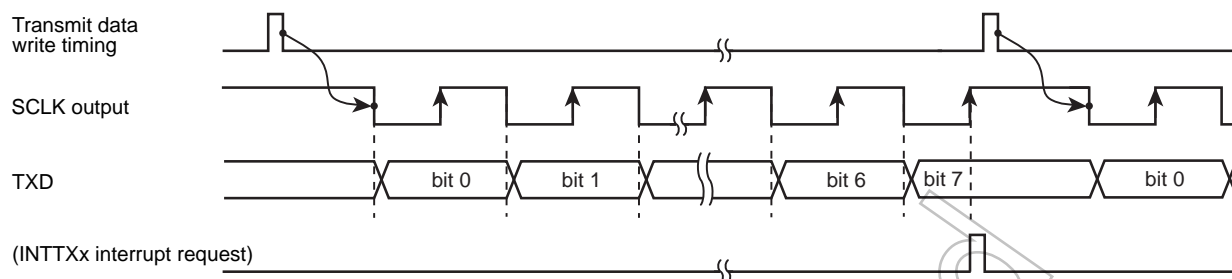
<WBUF> = "0" (if double buffering is disabled)
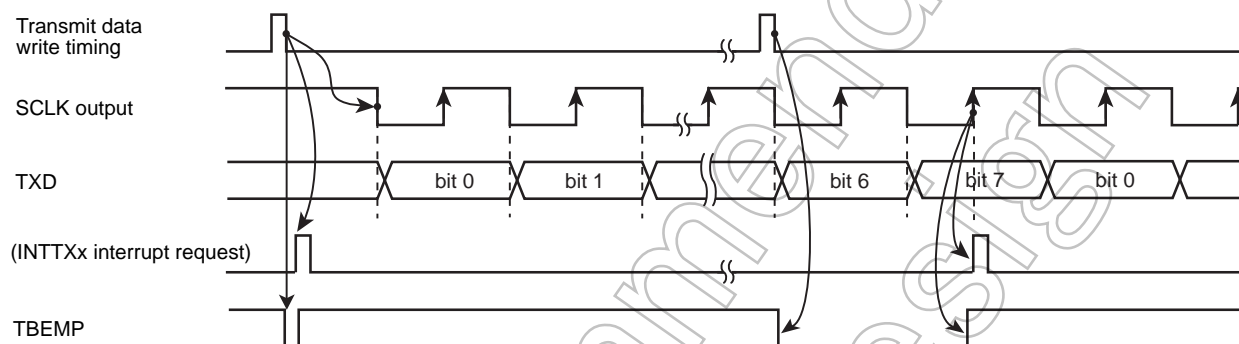
<WBUF> = "1" (If double buffering is enabled and there is data in buffer)
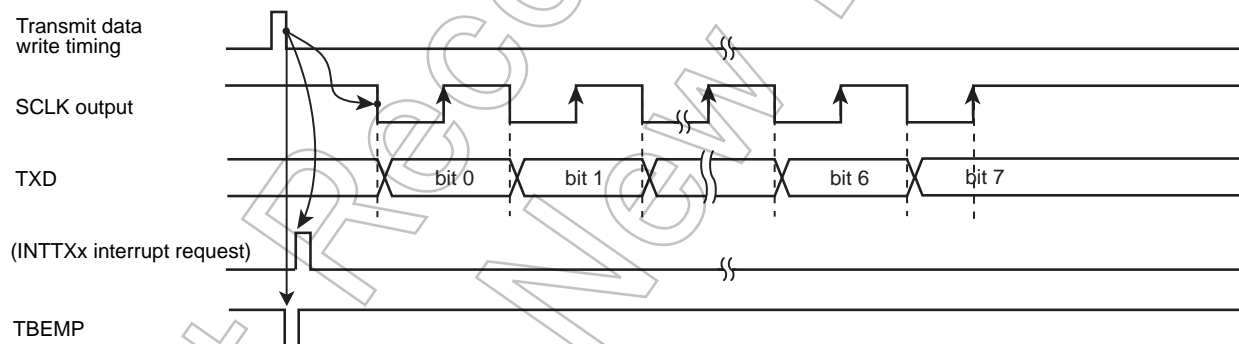
<WBUF> = "1" (if double buffering is enabled and there is no data in buffer)

Figure 12-12 Transmit Operation in the I/O Interface Mode (SCLK Output Mode)

(2) SCLK Input Mode

- If double buffering is disabled (SCxMOD2<WBUF> = "0")

  If the SCLK is input in the condition where data is written in the transmit buffer, 8-bit data is outputted from the TXD pin. When all data is output, an interrupt INTTXx is generated. The next transmit data must be written before the timing point "A" as shown in Figure 12-13.

- If double buffer is enabled (SCxMOD2<WBUF> = "1")

  Data is moved from the transmit buffer to the transmit shift register when the CPU writes data to the transmit buffer before the SCLK input becomes active or when data transmission from the transmit shift register is completed. Simultaneously, the transmit buffer empty flag SCxMOD2<TBEMP> is set to "1", and the INTTXx interrupt is generated.

  If the SCLK input becomes active while no data is in the transmit buffer, although the internal bit counter is started, an under-run error occurs and 8-bit dummy data (0xFF) is sent.

Transmit data write timing

SCLK input (<SCLKS>=0 Rising edge mode)

SCLK input (<SCLKS>=1 Falling edge mode)

TXD

(INTTXx interrupt request)

<WBUF> = "0" (if double buffering is disabled)

Transmit data write timing

SCLK input (<SCLKS>=0 Rising edge mode)

SCLK input (<SCLKS>=1 Falling edge mode)

TXD

(INTTXx interrupt request)

TBEMP

<WBUF> = "1" (if double beffering is enabled and there is data in beffer2)

Transmit data write timing

SCLK input (<SCLKS>=0 Rising edge mode)

SCLK input (<SCLKS>=1 Falling edge mode)

TXD

(INTTXx interrupt request)

TBEMP

PERR (Function to detect underrun error)

<WBUF> = "1" (if double buffering is enabled and there is no data in buffer2)

Figure 12-13 Transmit Operation in the I/O Interface Mode (SCLK Input Mode)

12.16.1.2 Receive

(1) SCLK Output Mode

The SCLK output can be started by setting the receive enable bit SCxMOD0<RXE> to "1".

- If double buffer is disabled (SCxMOD2<WBUF> = "0")

    A clock pulse is outputted from the SCLK pin and the next data is stored into the shift register each time the CPU reads received data. When all the 8 bits are received, the INTRXx interrupt is generated.

- If double buffer is enabled (SCxMOD2<WBUF> = "1")

    Data stored in the shift register is moved to the receive buffer and the receive buffer can receive the next frame. A data is moved from the shift register to the receive buffer, the receive buffer full flag SCxMOD2<RBFLL> is set to "1" and the INTRXx is generated.

    While data is in the receive buffer, if the data cannot be read from the receive buffer before completing reception of the next 8 bits, the INTRXx interrupt is not generated and the SCLK output stops. In this state, reading data from the receive buffer allows data in the shift register to move to the receive buffer and thus the INTRXx interrupt is generated and data reception resumes.
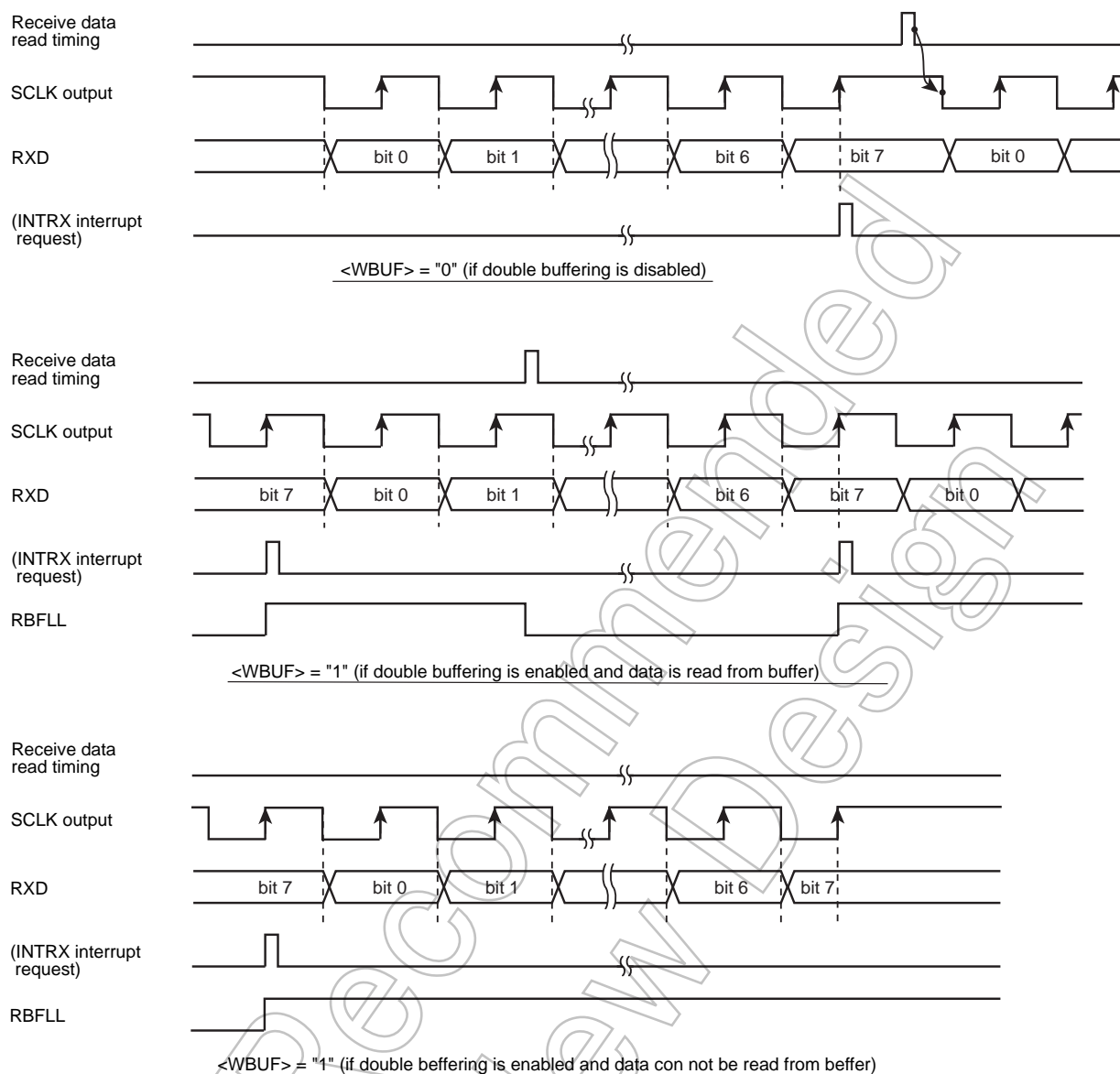
Figure 12-14 Receive Operation in the I/O Interface Mode (SCLK Output Mode)

(2) SCLK Input Mode

In the SCLK input mode, receiving double buffering is always enabled, the received frame can be moved to the receive buffer from the shift register, and the receive buffer can receive the next frame successively.

The INTRXx receive interrupt is generated each time received data is moved to the receive buffer.
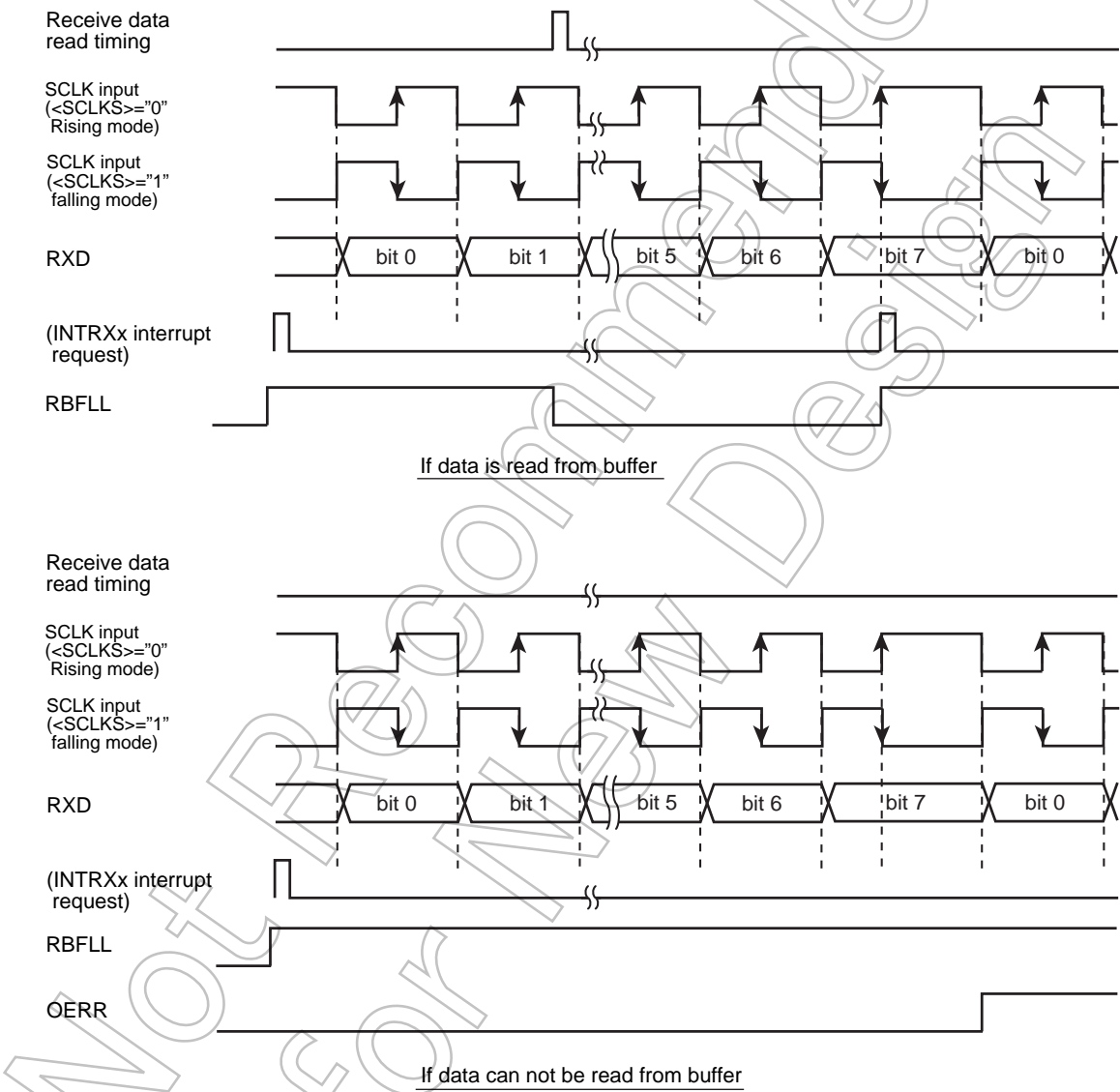
Figure 12-15 Receive Operation in the I/O Interface Mode (SCLK Input Mode)

12.16.1.3   Transmit and Receive (Full duplex)

(1)   SCLK Output Mode

·   If SCxMOD2<WBUF> is set to "0" and the double buffers are disabled

SCLK is outputted when the CPU writes data to the transmit buffer.

Subsequently, 8 bits of data are shifted into receive buffer and the INTRXx receive interrupt is generated. Concurrently, 8 bits of data written to the transmit buffer are outputted from the TXD pin, the INTTXx transmit interrupt is generated when transmission of all data bits has been completed. Then, the SCLK output stops.

The next round of data transmission and reception starts when the data is read from the receive buffer and the next transmit data is written to the transmit buffer by the CPU. The order of reading the receive buffer and writing to the transmit buffer can be freely determined. Data transmission is resumed only when both conditions are satisfied.

·   If SCxMOD2<WBUF> is set to "1" and the double buffers are enabled

SCLK is outputted when the CPU writes data to the transmit buffer.

8 bits of data are shifted into the receive shift register, moved to the receive buffer, and the INTRXx interrupt is generated. While 8 bits of data is received, 8 bits of transmit data is outputted from the TXD pin. When all data bits are sent out, the INTTXx interrupt is generated and the next data is moved from the transmit buffer to the transmit shift register.

If the transmit buffer has no data to be moved to the transmit buffer (SCxMOD2<TBEMP> = 1) or when the receive buffer is full (SCxMOD2<RBFULL> = 1), the SCLK output is stopped. When both conditions, receive data is read and transmit data is written, are satisfied, the SCLK output is resumed and the next round of data transmission and reception is started.
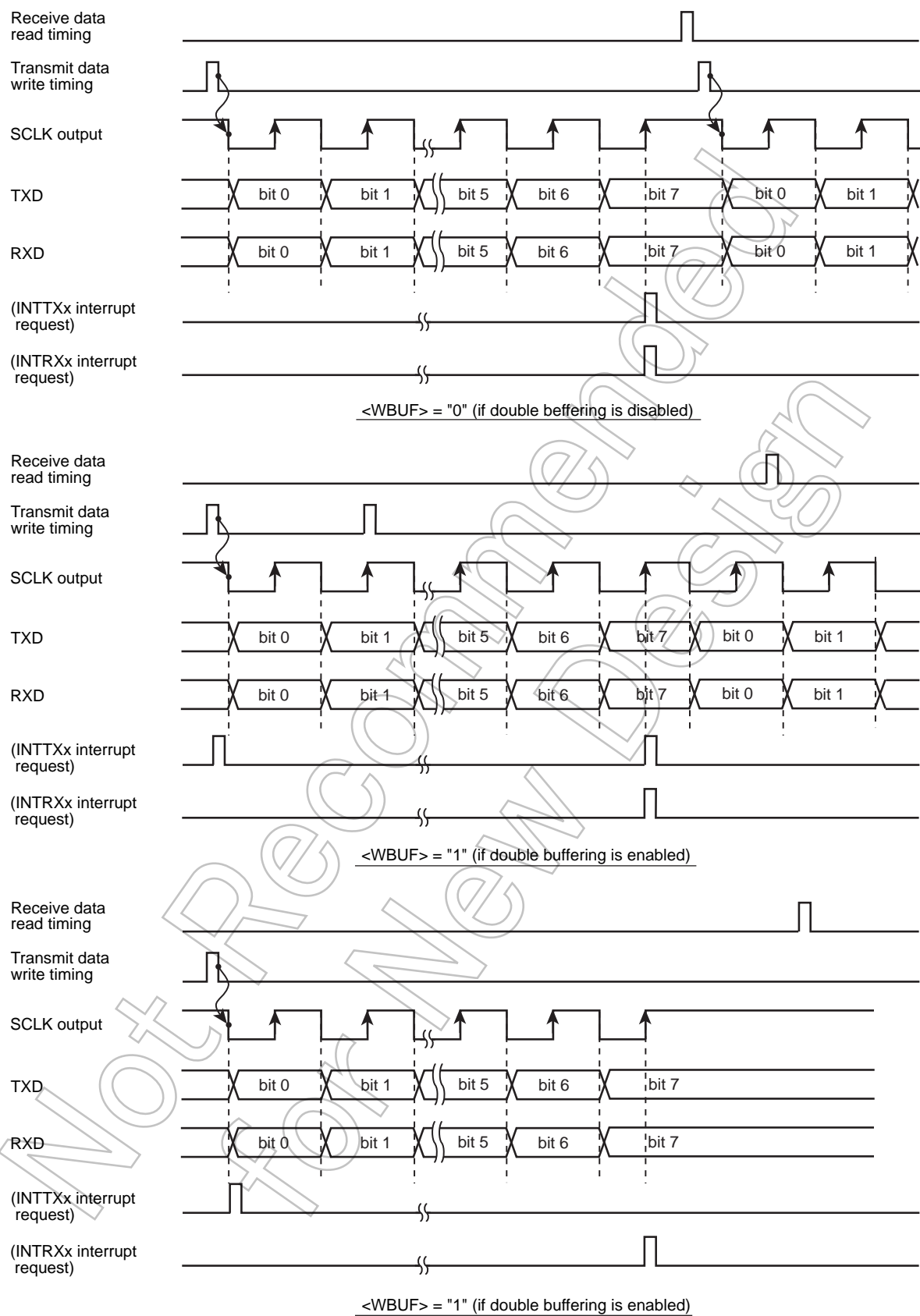
Figure 12-16 Transmit / Receive Operation in the I/O Interface Mode (SCLK Output Mode)

(2)   SCLK Input Mode

·   If SCxMOD2<WBUF> is set to "0" and the transmit double buffer is disabled

When receiving data, double buffer is always enabled regardless of the SCxMOD2 <WBUF> settings.

8-bit data written in the transmit buffer is outputted from the TXD pin and 8 bit of data is shifted into the receive buffer when the SCLK input becomes active.The INTTXx interrupt is generated upon completion of data transmission. The INTTRXx interrupt is generated when the data is moved from shift register to receive buffer after completion of data reception.

Note that transmit data must be written into the transmit buffer before the SCLK input for the next frame (data must be written before the point A in Figure 10-17). Data must be read before completing reception of the next frame data.

·   If SCxMOD2<WBUF> is set to "1" and the double buffer is enabled.

The interrupt INTRXx is generated at the timing the transmit buffer data is moved to the transmit shift register after completing data transmission from the transmit shift register. At the same time, data received is shifted to the shift register, it is moved to the receive buffer, and the INTRXx interrupt is generated.

Note that transmit data must be written into the transmit buffer before the SCLK input for the next frame (data must be written before the point A in Figure 12-17). Data must be read before completing reception of the next frame data.

Upon the SCLK input for the next frame, transmission from transmit shift register (in which data has been moved from transmit buffer) is started while receive data is shifted into receive shift register simultaneously.

If data in receive buffer has not been read when the last bit of the frame is received, an over-run error occurs. Similarly, if there is no data written to transmit buffer when SCLK for the next frame is input, an under-run error occurs.
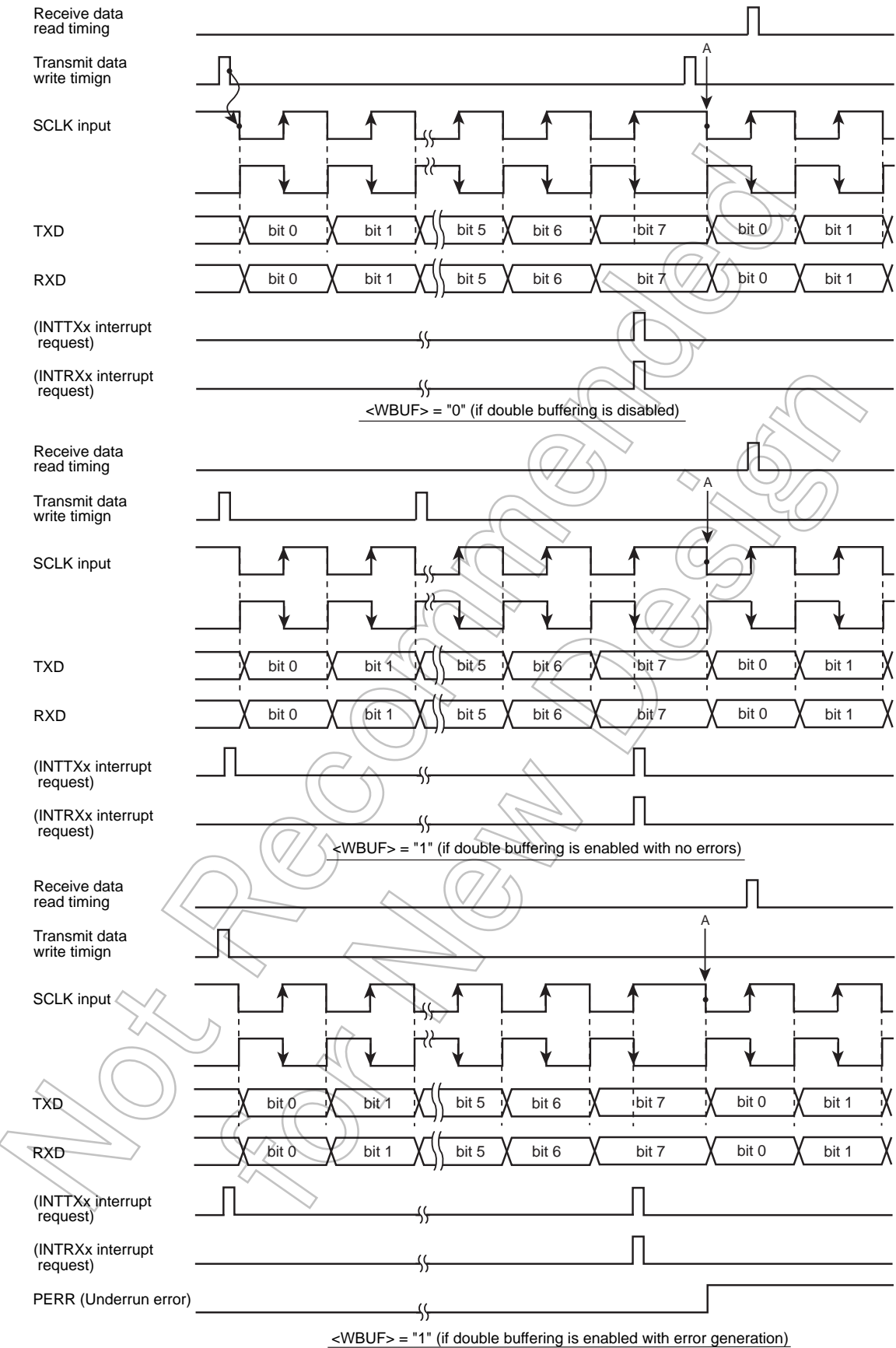
Figure 12-17 Transmit / Receive Operation in the I/O Interface Mode (SCLK Input Mode)

12.16.2   Mode 1 (7-bit UART Mode)

The 7-bit UART mode can be selected by setting the serial mode control register (SCxMOD<SM[1:0]>) to "01".

In this mode, parity bits can be added to the transmit data stream; the serial mode control register (SCxCR<PE>) controls the parity enable/disable setting. When <PE> is set to "1" (enable), either even or odd parity may be selected using the SCxCR<EVEN> bit. The length of the stop bit can be specified using SCxMOD2<SBLEN>.

The following table shows the control register settings for transmitting in the following data format.

```
 - - - ⌐‾⌐ start ⟩⟨ bit 0 ⟩⟨ 1 ⟩⟨ 2 ⟩⟨ 3 ⟩⟨ 4 ⟩⟨ 5 ⟩⟨ 6 ⟩⟨ even parity ⟩ stop
                  ◄─── Transmission direction (Transmission rate of 2400 bps @fc = 9.8304 MHz)
```

Clocking condition
- System clock :   High-speed (fc)
- High-speed clock gear :   x1 (fx)
- Prescaler clock :   fperiph/2 (fperiph = fsys)

		7	6	5	4	3	2	1	0	
SCxMOD0	←	x	0	-	0	0	1	0	1	Set 7-bit UART mode
SCxCR	←	x	1	1	x	x	x	0	0	Even parity enabled
SCxBRCR	←	0	0	1	0	0	1	0	0	Set 2400bps
SCxBUF	←	*	*	*	*	*	*	*	*	Set transmit data

x : don't care - : no change

12.16.3   Mode 2 (8-bit UART Mode)

The 8-bit UART mode can be selected by setting SCxMOD0<SM[1:0]> to "10". In this mode, parity bits can be added and parity enable/disable is controlled using SCxCR<PE>. If <PE> = "1" (enabled), either even or odd parity can be selected using SCxCR<EVEN>.

The control register settings for receiving data in the following format are as follows :

```
 - - - ⌐‾⌐ start ⟩⟨ bit 0 ⟩⟨ 1 ⟩⟨ 2 ⟩⟨ 3 ⟩⟨ 4 ⟩⟨ 5 ⟩⟨ 6 ⟩⟨ 7 ⟩⟨ odd parity ⟩ stop
                  ◄─── Transmission direction (Transmission rate of 9600 bps @fc = 9.8304 MHz)
```

Clocking condition
- System clock :   High-speed (fc)
- High-speed clock gear :   x1 (fc)
- Prescaler clock :   fperiph/2 (fperiph = fsys)
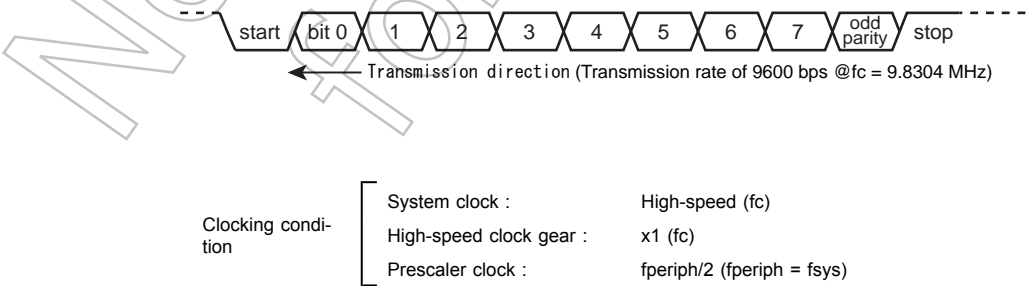
		7	6	5	4	3	2	1	0	
SCxMOD0	←	x	0	0	0	1	0	0	1	Set 8-bit UART mode
SCxCR	←	x	0	1	x	x	x	0	0	Odd parity enabled
SCxBRCR	←	0	0	0	1	0	1	0	0	Set 9600bps
SCxMOD0	←	-	-	1	-	-	-	-	-	Reception enabled

x : don't care - : no change

12.16.4   Mode 3 (9-bit UART Mode)

The 9-bit UART mode can be selected by setting SCxMOD0<SM[1:0]> to "11". In this mode, parity bits must be disabled (SCxCR<PE> = "0").

The most significant bit (9th bit) is written to bit 7 <TB8> of the serial mode control register 0 (SCxMOD0) for transmitting data. The data is stored in bit 7 <RB8> of the serial control register SCxCR.

When writing or reading data to/from the buffers, the most significant bit must be written or read first before writing or reading to/from SCxBUF. The stop bit length can be specified using SCxMOD2<SBLEN>.

12.16.4.1   Wake-up Function

In the 9-bit UART mode, slave controllers can be operated in the wake-up mode by setting the wake-up function control bit SCxMOD0<WU> to "1".

In this case, the interrupt INTRXx will be generated only when SCxCR<RB8> is set to "1".

Note:**The TXD pin of the slave controller must be set to the open drain output mode using the ODE register.**
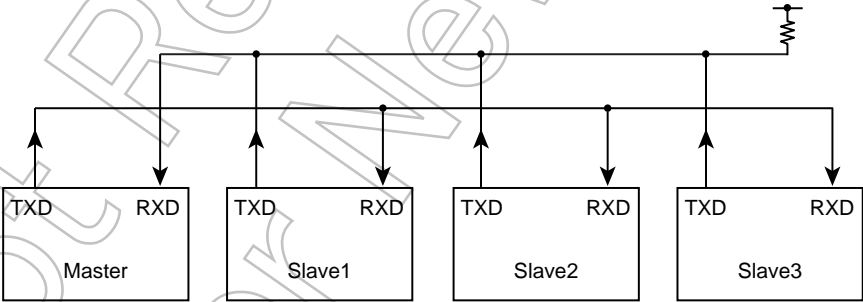
Figure 12-18 Serial Links to Use Wake-up Function

12.16.4.2   Protocol

1.   Select the 9-bit UART mode for the master and slave controllers.

2.   Set SCxMOD<WU> to "1" for the slave controllers to make them ready to receive data.

3.   The master controller is to transmit a single frame of data that includes the slave controller se-
     lect code (8 bits). In this, the most significant bit (bit 8) <TB8> must be set to "1".
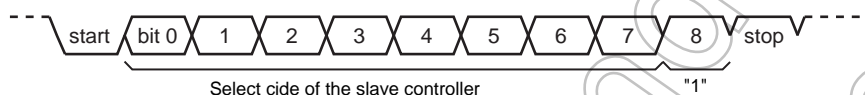
4.   Each slave controller receives the above data frame; if the code received matches with the con-
     troller's own select code, it clears the <WU> bit to "0".

5.   The master controller transmits data to the designated slave controller (the controller of which
     SCxMOD<WU> bit is cleared to "0"). In this, the most significant bit (bit 8) <TB8> must be
     set to "0".
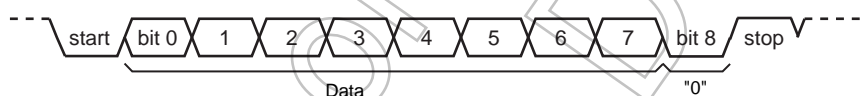
6.   The slave controllers with the <WU> bit set to "1" ignore the receive data because the most sig-
     nificant bit (bit 8) <RB8> is set to "0" and thus no interrupt (INTRXx) is generated.Also, the
     slave controller with the <WU> bit set to "0" can transmit data to the master controller to in-
     form that the data has been successfully received.

13. Synchronous Serial Port (SSP)

13.1 Overview

This LSI contains the SSP (Synchronous Serial Port) with 1 channel. This channel has the following features.

Communication protocol		Three types of synchronous serial ports including the SPI ・ Motorola SPI (SPI) frame format ・ TI synchronous (SSI) frame format ・ National Microwire (Microwire) frame format
Operation mode		Master/slave mode
Transmit FIFO		16bits wide / 8 tiers deep
Receive FIFO		16bits wide / 8 tiers deep
Transmitted/received data size		4 to 16 bits
Interrupt type		Transmit interrupt Receive interrupt Receive overrun interrupt Time-out interrupt
Communication speed	In master mode	fsys (64MHz)/ 4 (max. 16Mbps)
	In slave mode	fsys (64MHz)/ 12 (max. 5.3Mbps)
DMA		Supported
Internal test function		The internal loopback test mode is available.
Control pin		SPCLK,SPFSS,SPDO,SPDI

13.2 Block Diagram
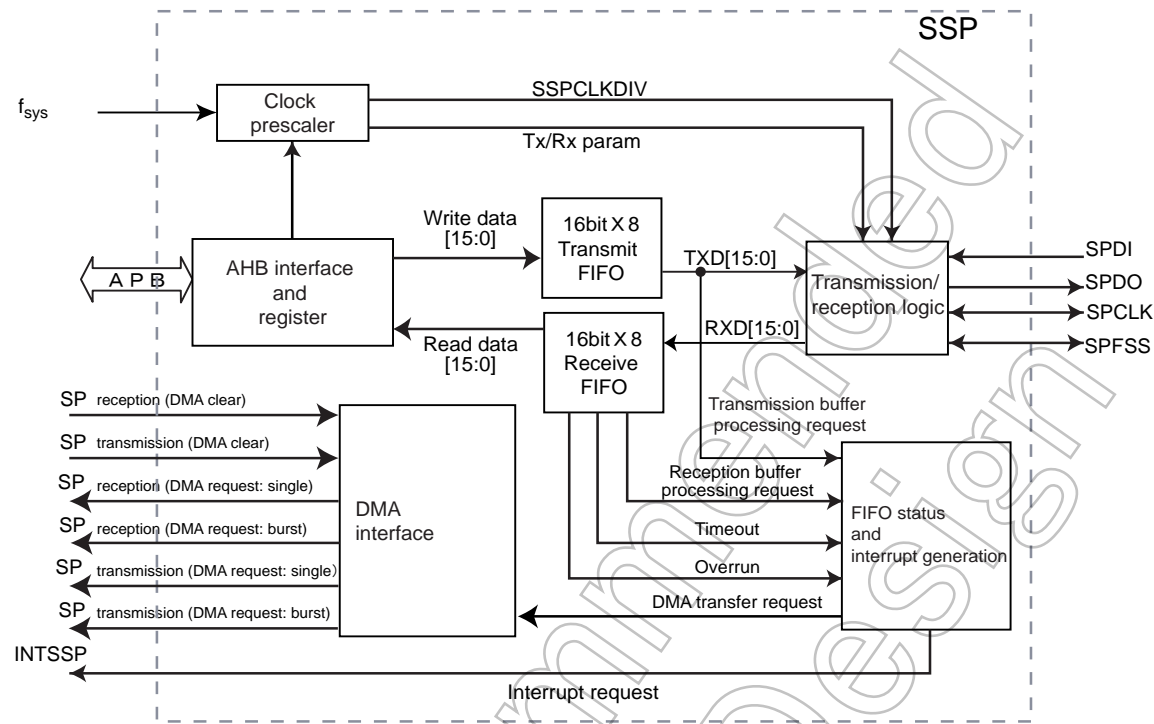
Figure 13-1 SSP block diagram

13.3 Register

13.3.1 Register List

Base Address = 0x4004_0000

Register Name		Address (Base+)
Control register 0	SSPCR0	0x0000
Control register 1	SSPCR1	0x0004
Receive FIFO (read) and transmit FIFO (write) data register	SSPDR	0x0008
Status register	SSPSR	0x000C
Clock prescale register	SSPCPSR	0x0010
Interrupt enable/disable register	SSPIMSC	0x0014
Pre-enable interrupt status register	SSPRIS	0x0018
Post-enable interrupt status register	SSPMIS	0x001C
Interrupt clear register	SSPICR	0x0020
DMA control register	SSPDMACR	0x0024
Reserved	-	0x0028 to 0x0FFC

Note 1: These registers in the above table allows to access only word (32 bits) basis.

Note 2: Access to the "Reserved" area is prohibited.

13.3.2 SSPCR0(Control register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	SCR							
After Reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SPH	SPO	FRF		DSS			
After Reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	W	Write as "0".
15-8	SCR[7:0]	R/W	For serial clock rate setting. Parameter : 0x00 to 0xFF. Bits to generate the SSP transmit bit rate and receive bit rate. This bit rate can be obtained by the following equation. Bit rate = f_{sys} / (\<CPSDVSR\> × (1+ \<SCR\>)) \<CPSDVSR\> is an even number between 2 to 254, which is programmed by the SSPCPSR register, and \<SCR\> takes a value between 0 to 255.
7	SPH	R/W	SPCLK phase: 0 : Captures data at the 1st clock edge. 1 : Captures data at the 2nd clock edge. This is applicable to Motorola SPI frame format only. Refer to Section "Motorola SPI frame format"
6	SPO	R/W	SPCLK polarity: 0:SPCLK is in Low state. 1:SPCLK is in High state. This is applicable to Motorola SPI frame format only. Refer to Section "Motorola SPI frame format"
5-4	FRF[1:0]	R/W	Frame format: 00: SPI frame format 01: SSI serial frame format 10: Microwire frame format 11: Reserved, undefined operation
3-0	DSS[3:0]	R/W	Data size select: 0000: Reserved, undefined operation — 1000: 9 bits data 0001: Reserved, undefined operation — 1001: 10 bits data 0010: Reserved, undefined operation — 1010: 11 bits data 0011: 4 bits data — 1011: 12 bits data 0100: 5 bits data — 1100: 13 bits data 0101: 6 bits data — 1101: 14 bits data 0110: 7 bits data — 1110: 15 bits data 0111: 8 bits data — 1111: 16 bits data

Note: Set a clock prescaler to SSPCR0\<SCR[7:0]\> = 0x00 , SSPCPSR\<CPSDVSR[7:0]\> = 0x02, when slave mode is selected.

13.3.3   SSPCR1(Control register1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	SOD	MS	SSE	LBM
After Reset	Undefined	Undefined	Undefined	Undefined	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	W	Write as "0".
3	SOD	R/W	Slave mode SPDO output control: 0: Enable 1: Disable Slave mode output disable. This bit is relevant only in the slave mode (<MS>="1").
2	MS	R/W	Master/slave mode select: (Note) 0: Device configured as a master. 1: Device configured as a slave.
1	SSE	R/W	SSP enable/disable 0: Disable 1: Enable
0	LBM	R/W	Loop back mode 0: Normal serial port operation enabled. 1: Output of transmit serial shifter is connected to input of receive serial shifter internally.

Note: This bit is for switching between master and slave. Be sure to configure in the following steps in slave mode and in transmission.

1)	Set to slave mode	:<MS>=1
2)	Set transmit data in FIFO	:<DATA>=0x****
3)	Set SSP to Enable.	:<SSE>=1

13.3.4 SSPDR(Data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	DATA							
After Reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DATA							
After Reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	W	Write as "0".
15-0	DATA[15:0]	R/W	Transmit/receive FIFO data: 0x0000 to 0xFFFF Read: Receive FIFO Write: Transmit FIFO If the data size used in the program is less than 16bits, write the data to fit LSB.The transmit control circuit ignores unused bits of MSB side. The receive control circuit receives the data to fit LSB automatically.

13.3.5   SSPSR(Status register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	BSY	RFF	RNE	TNF	TFE
After Reset	Undefined	Undefined	Undefined	0	0	0	1	1

Bit	Bit Symbol	Type	Function
31-5	−	W	Write as "0".
4	BSY	R	Busy flag: 0: Idle 1: Busy <BSY>="1" indicates that the SSP is currently transmitting and/or receiving a frame or the transmit FIFO is not empty.
3	RFF	R	Receive FIFO full flag: 0: Receive FIFO is not full. 1: Receive FIFO is full.
2	RNE	R	Receive FIFO empty flag: 0: Receive FIFO is empty. 1: Receive FIFO is not empty.
1	TNF	R	Transmit FIFO full flag: 0: Transmit FIFO is full. 1: Transmit FIFO is not full.
0	TFE	R	Transmit FIFO empty flag: 0: Transmit FIFO is not empty. 1: Transmit FIFO is empty.

13.3.6   SSPCPSR (Clock prescale register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	CPSDVSR							
After Reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	W	Write as "0".
7-0	CPSDVSR[7:0]	R/W	Clock prescale divisor: Set an even number from 2 to 254. Clock prescale divisor: Must be an even number from 2 to 254, depending on the frequency of fsys. The least significant bit always returns zero when read.

Note: Set a clock prescaler to SSPCR0<SCR[7:0]> = 0x00 , SSPCPSR<CPSDVSR[7:0]> = 0x02, when slave mode is selected.

13.3.7 SSPIMSC (Interrupt enable/disable register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	TXIM	RXIM	RTIM	RORIM
After Reset	Undefined	Undefined	Undefined	Undefined	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	W	Write as "0".
3	TXIM	R/W	Transmit FIFO interrupt enable: 0: Disable 1: Enable Enable or disable a conditional interrupt to occur if the transmit FIFO is half empty or less.
2	RXIM	R/W	Receive FIFO interrupt enable: 0: Disable 1: Enable Enable or disable a conditional interrupt to occur if the receive FIFO is half full or less.
1	RTIM	R/W	Receive time-out interrupt enable: 0: Disable 1: Enable Enable or disable a conditional interrupt to indicate that data exists in the receive FIFO to the time-out period and data is not read.
0	RORIM	R/W	Receive overrun interrupt enable: 0: Disable 1: Enable Enable or disable a conditional interrupt to indicate that data was written when the receive FIFO was in the full condition.

13.3.8   SSPRIS (Pre-enable interrupt status register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	TXRIS	RXRIS	RTRIS	RORRIS
After Reset	Undefined	Undefined	Undefined	Undefined	1	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	W	Write as "0".
3	TXRIS	R	Pre-enable transmit interrupt flag: 0: Interrupt not present 1: Interrupt present
2	RXRIS	R	Pre-enable receive interrupt flag: 0: Interrupt not present 1: Interrupt present
1	RTRIS	R	Pre-enable timeout interrupt flag: 0: Interrupt not present 1: Interrupt present
0	RORRIS	R	Pre-enable overrun interrupt flag: 0: Interrupt not present 1: Interrupt present

13.3.9   SSPMIS (Post-enable interrupt status register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	TXMIS	RXMIS	RTMIS	RORMIS
After Reset	Undefined	Undefined	Undefined	Undefined	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	W	Write as "0".
3	TXMIS	R	Post-enable transmit interrupt flag: 0: Interrupt not present 1: Interrupt present
2	RXMIS	R	Post-enable receive interrupt flag: 0: Interrupt not present 1: Interrupt present
1	RTMIS	R	Post-enable time-out interrupt flag: 0: Interrupt not present 1: Interrupt present
0	RORMIS	R	Post-enable overrun interrupt flag: 0: Interrupt not present 1: Interrupt present

13.3.10 SSPICR (Interrupt clear register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	RTIC	RORIC
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-2	−	W	Write as "0".
1	RTIC	W	Clear the time-out interrupt flag: 0: Invalid 1: Clear
0	RORIC	W	Clear the overrun interrupt flag: 0: Invalid 1: Clear

13.3.11 SSPxDMACR (DMA control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	TXDMAE	RXDMAE
After Reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	−	W	Write as "0".
1	TXDMAE	R/W	Transmit FIFO DMA control: 0:Disable 1:Enable
0	RXDMAE	R/W	Transmit FIFO DMA control: 0:Disable 1:Enable

13.4 Overview of SSP

This LSI contains the SSP with 1channels.

The SSP is an interface that enables serial communications with the peripheral devices with three types of synchronous serial interface functions.

The SSP performs serial-parallel conversion of the data received from a peripheral device.

The transmit buffers data in the independent 16-bit wide and 8-layered transmit FIFO in the transmit mode, and the receive buffers data in the 16-bit wide and 8-layered receive FIFO in receive mode. Serial data is transmitted via SPDO and received via SPDI.

The SSP contains a programmable prescaler to generate the serial output clock SPCLK from the input clock fsys. The operation mode, frame format, and data size of the SSP are programmed in the control registers SSPCR0 and SSPCR1.

13.4.1 Clock prescaler

When configured as a master, a clock prescaler comprising two free-running serially linked counters is used to provide the serial output clock SPCLK.

You can program the clock prescaler through the SSPCPSR register, to divide fsys by a factor of 2 to 254 in steps of two. Because the least significant bit of the SSPCPSR register is not used, division by an odd number is not possible.

The output of the prescaler is further divided by a factor of 1 to 256, which is obtained by adding 1 to the value programmed in the SSPCR0 register, to give the master output clock SPCLK.

Bitrate = fsys / (<CPSDVSR>×(1+<SCR>))
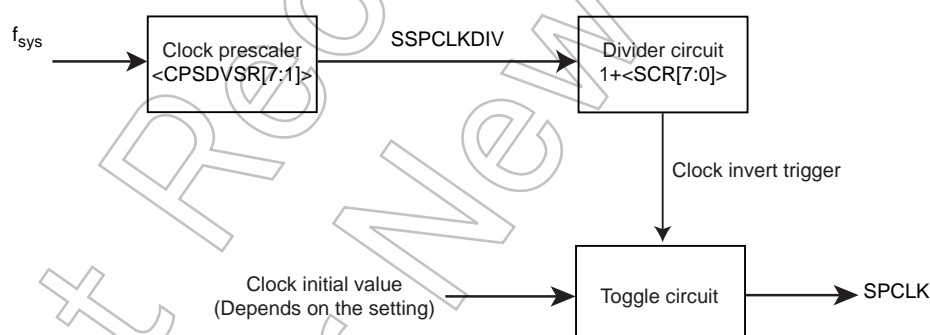
13.4.2 Transmit FIFO

This is a 16-bit wide, 8-layered transmit FIFO buffer, which is shared in master and slave modes.

13.4.3 Receive FIFO

This is a 16-bit wide 8-layered receive FIFO buffer, which is shared in master and slave modes.

13.4.4   Interrupt generation logic

The interrupts, each of which can be masked separately, are generated.

Transmit interrupt	A conditional interrupt to occur when the transmit FIFO has free space more than (including half) of the entire capacity. (Number of valid data items in the transmit FIFO ≤ 4)
Receive interrupt	A conditional interrupt to occur when the receive FIFO has valid data more than half (including half) the entire capacity. (Number of valid data items in the receive FIFO ≥ 4)
Time-out interrupt	A conditional interrupt to indicate that the data exists in the receive FIFO to the time-out period.
Overrun interrupt	Conditional interrupts indicating that data is written to receive FIFO when it is full.

Also, The individual masked sources are combined into a single interrupt. When any of the above interrupts is asserted, the combined interrupt INTSSP is asserted.
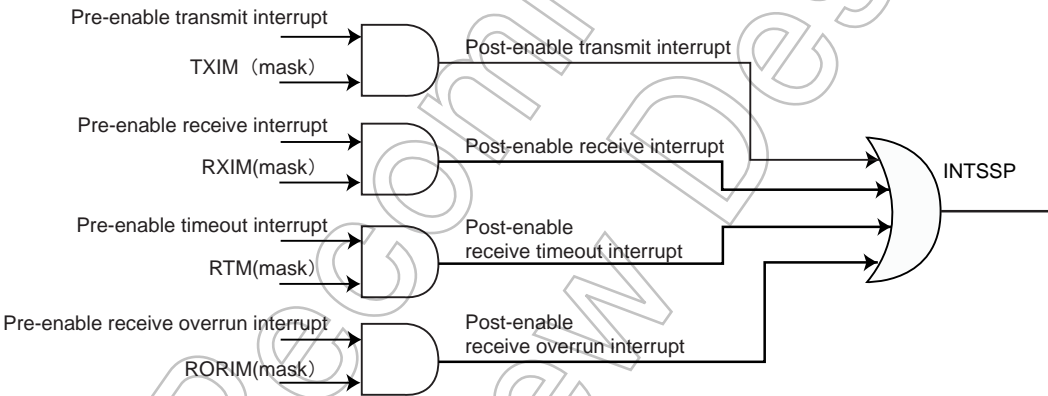
a.   Transmit interrupt

The transmit interrupt is asserted when there are four or fewer valid entries in the transmit FIFO. The transmit interrupt is also generated when the SSP operation is disabled (SSPCR1 <SSE> = "0").

The first transmitted data can be written in the FIFO by using this interrupt.

b.   Receive interrupt

The receive interrupt is asserted when there are four or more valid entries in the receive FIFO.

c.   Time-out interrupt

The time-out interrupt is asserted when the receive FIFO is not empty and the SSP has remained idle for a fixed 32-bit period (bit rate). This mechanism ensures that the user is aware that data is still present in the receive FIFO and requires servicing. This operation occurs in both master and slave modes. When the time-out interrupt is generated, read all data from the receive FIFO. Even if all the data is not read, data can be transmitted / received if the receive FIFO has a free space and the number of data to be transmitted does not exceed the free space of the receive FIFO. When transfer starts, the timeout interrupt will be cleared. If data is transmitted / received when the receive FIFO has no free space, the time-out interrupt will not be cleared and an overrun interrupt will be generated.

SPCLK

Receive FIFO empty flag
SSPSR<RNE>

During data transfer

Internal down counter enable

bit rate

Receive timeout interrupt enable
SSPIMSC<RTIM>

Receive timeout interrupt
SSPMIS<RTMIS>
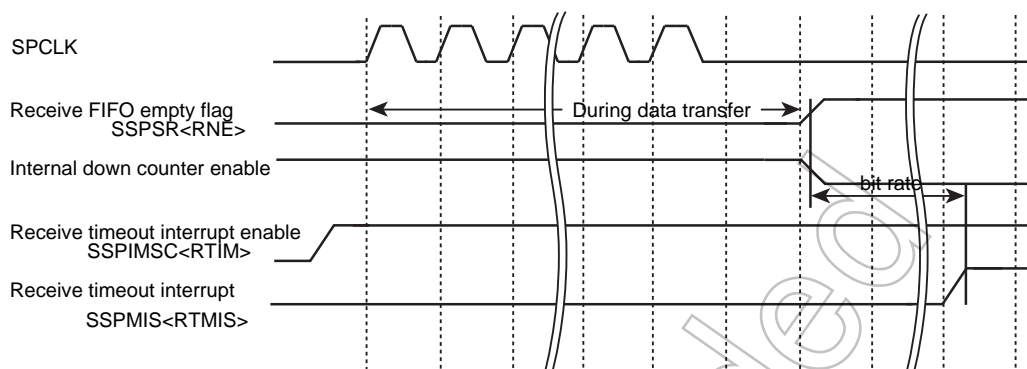
d. Overrun interrupt

When the next data (9th data item) is received when the receive FIFO is already full, an overrun interrupt is generated immediately after transfer. The data received after the overrun interrupt is generated (including the 9th data item) will become invalid and be discarded. However, if data is read from the receive FIFO while the 9th data item is being received (before the interrupt is generated), the 9th received data will be written in the receive FIFO as valid data. To perform transfer properly when the overrun interrupt has been generated, write "1" to SSPICR<RORIC> register, and then read all data from the receive FIFO. Even if all the data is not read, data can be transmitted / received if the receive FIFO has free space and the number of data to be transmitted does not exceed the free space of the receive FIFO. Note that if the receive FIFO is not read (provided that the receive FIFO is not empty) within a certain 32-bit period (bit rate) after the overrun interrupt is cleared, a time-out interrupt will be generated.

13.4.5 DMA interface

The DMA operation of the SSP is controlled through SSPxDMACR register.

When there are more data than the watermark level (half of the FIFO) in the receive FIFO, the receive DMA request is asserted.

When the amount of data left in the transmit FIFO is less than the watermark level (half of the FIFO), the transmit DMA request is asserted.

To clear the transmit/receive DMA request, an input pin for the transmit/receive DMA request clear signals, which are asserted by the DMA controller, is provided.

Set the DMA burst length to four words.

Note: For the remaining three words, the SSP does not assert the burst request.

Each request signal remains asserted until the relevant DMA clear signal is asserted. After the request clear signal is deasserted, a request signal can become active again, depending on the conditions described above. All request signals are deasserted if the SSP is disabled or the DMA enable signal is cleared.

The following table shows the trigger points for DMABREQ, for both the transmit and receive FIFOs.

Watermark level	Burst length	
	Transmit (number of empty locations)	Receive (number of filled locations)
1/2	4	4

13.5   SSP operation

13.5.1   Initial setting for SSP

Settings for the SSP communication protocol must be made with the SSP disabled.

Control registers SSPCR0 and SSPCR1 need to configure this SSP as a master or slave operating under one of the following protocols. In addition, make the settings related to the communication speed in the clock prescale registers SSPCPSR and SSPCR0 <SCR>.

This SSP supports the following protocols:

- ·   SPI
- ·   SSI
- ·   Microwire

13.5.2   Enabling SSP

The transfer operation starts when the operation is enabled with the transmitted data written in the transmit FIFO, or when transmitted data is written in the transmit FIFO with the operation enabled.

However, if the transmit FIFO contains only four or fewer entries when the operation is enabled, a transmit interrupt will be generated. This interrupt can be used to write the initial data.

Note:When the SSP is in the SPI slave mode and the SPFSS pin is not used, be sure to transmit data of one byte or more in the FIFO before enabling the operation. If the operation is enabled with the transmit FIFO empty, the transfer data will not be output correctly.

13.5.3   Clock ratios

When setting a frequency for fsys , the following conditions must be met.

- ·   In master mode

    f_{SPCLK} (maximum) $\rightarrow$ f_{sys} /4

    f_{SPCLK} (minimum) $\rightarrow$ f_{sys} /(254×256)
- ·   In slave mode

    f_{SPCLK} (maximum) $\rightarrow$ f_{sys} /12

    f_{SPCLK} (minimum) $\rightarrow$ f_{sys} /(254×256)

Note:The maximum baud-rate in the master mode is equal or less than 16Mbps.

13.6   Frame Format

Each frame format is between 4 and 16 bits wide depending on the size of data programmed, and is transmitted starting from the MSB.

- Serial clock (SPCLK)

  Signals remain "Low" in the SSI and Microwire formats and as inactive in the SPI format while the SSP is in the idle state. In addition, data is output at the set bit rate only during data transmission.

- Serial frame (SPFSS)

  In the SPI and Microwire frame formats, signals are set to "Low" active and always asserted to "Low" during frame transmission.

  In the SSI frame format, signals are asserted only during 1 bit rate before each frame transmission. In this frame format, output data is transmitted at the rising edge of SPCLK and the input data is received at its falling edge.

Refer to Section "13.6.1" to "13.6.3" for details of each frame format.

13.6.1   SSI frame format

In this mode, the SSP is in idle state, SPCLK and SPFSS are forcedly set to "Low", and the transmit data line SPDO becomes Hi-Z. When data is written in the transmit FIFO, the master outputs "High" pulses of 1 SPCLK to the SPFSS line. The transmitted data will be transferred from the transmit FIFO to the transmit serial shift register. Data of 4 to 16 bits will be output from the SPDO pin at the next rising edge of SPCLK.

Likewise, the received data will be input starting from the MSB to the SPDI pin at the falling edge of SPCLK. The received data will be transferred from the serial shift register into the receive FIFO at the rising edge of SPCLK after its LSB data is latched.
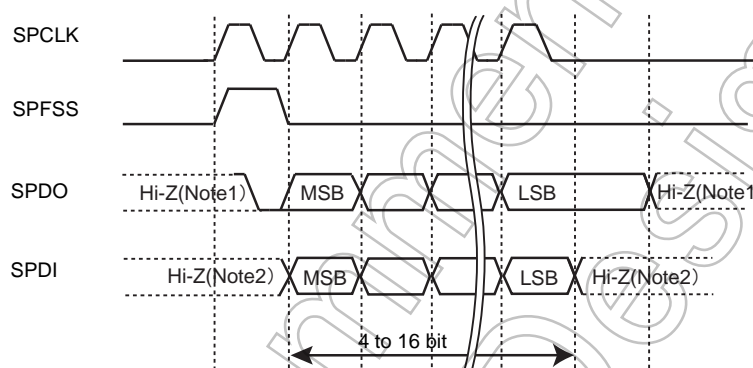
Figure 13-2 SSI frame format (transmission/reception during single transfer)
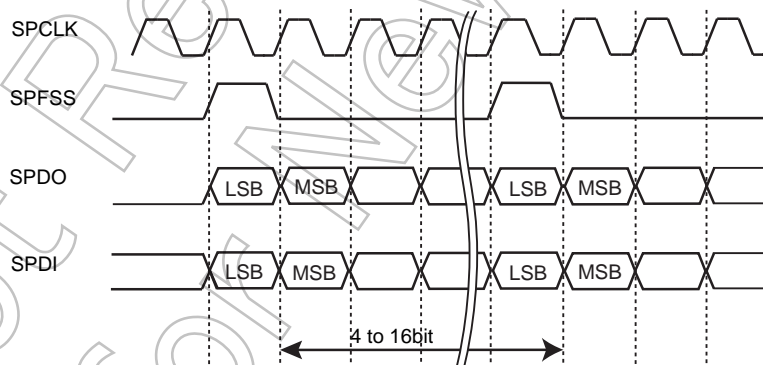
Figure 13-3 SSI frame format (transmission/reception during continuous transfer)

Note 1: When transmission is disable, SPDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note 2: SPDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

13.6.2 SPI frame format

The SPI interface has 4 lines. SPFSS is used for slave selection. One of the main features of the SPI format is that the <SPO> and <SPH> bits in the SSPCR0 register can be used to set the SPCLK operation timing.

SSPCR0 <SPO> is used to set the level at which SPCLK in idle state is held.

SSPCR0 <SPH> is used to select the clock edge at which data is latched.

	SSPCR0<SPO>	SSPCR0<SPH>
0	"Low" state	Capture data at the 1st clock edge.
1	"High" state	Capture data at the 2nd clock edge.
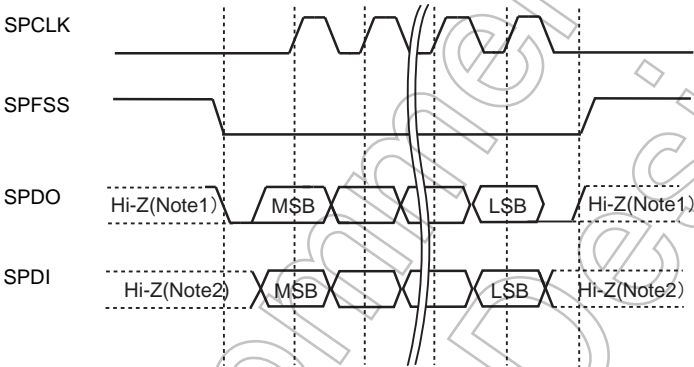
Figure 13-4 SPI frame format (single transfer, <SPO>="0" & <SPH>="0")
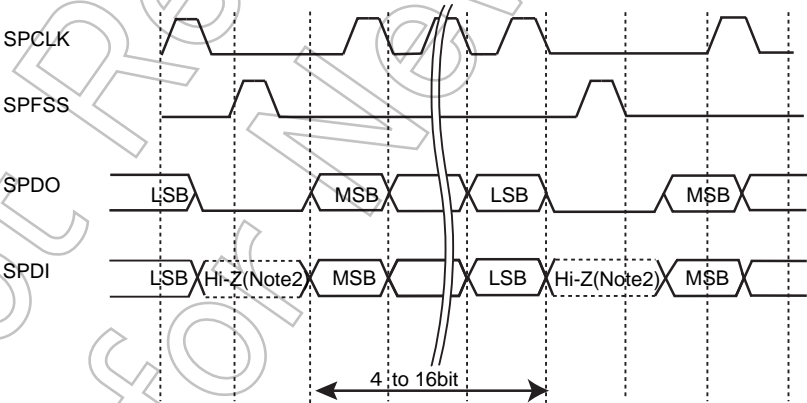
Figure 13-5 SPI frame format (continuous transfer,<SPO>="0" & <SPH>="0")

Note 1: When transmission is disable, SPDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note 2: SPDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

With this setting <SPO>="0", during the idle period:

- The SPCLK signal is set to "Low".
- SPFSS is set to "High".
- The transmit data line SPDO is set to "Low".

If the SSP is enabled and valid data exists in the transmit FIFO, the SPFSS master signal driven by "Low" notifies of the start of transmission. This enables the slave data in the SPDI input line of the master.

When a half of the SPCLK period has passed, valid master data is transferred to the SPDO pin. Both the master data and slave data are now set. When another half of SPCLK has passed, the SPCLK master clock pin becomes "High". After that, the data is captured at the rising edge of the SPCLK signal and transmitted at its falling edge.

In the single transfer, the SPFSS line will return to the idle "High" state when all the bits of that data word have been transferred, and then one cycle of SPCLK has passed after the last bit was captured.

However, for continuous transfer, the SPFSS signal must be pulsed at HIGH between individual data word transfers. This is because change is not enabled when the slave selection pin freezes data in its peripheral register and the <SPH> bit is logical 0.

Therefore, to enable writing of serial peripheral data, the master device must drive the SPFSS pin of the slave device between individual data transfers. When the continuous transfer is completed, the SPFSS pin will return to the idle state when one cycle of SPCLK has passed after the last bit is captured.

13.6.3   Microwire frame format

The Microwire format uses a special master/slave messaging method, which operates in half-duplex mode. In this mode, when a frame begins, an 8-bit control message is transmitted to the slave. During this transmission, no incoming data is received by the SSP. After the message has been transmitted, the slave decodes it, and after waiting one serial clock after the last bit of the 8-bit control message has been sent, it responds with the requested data. The returned data can be 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits.
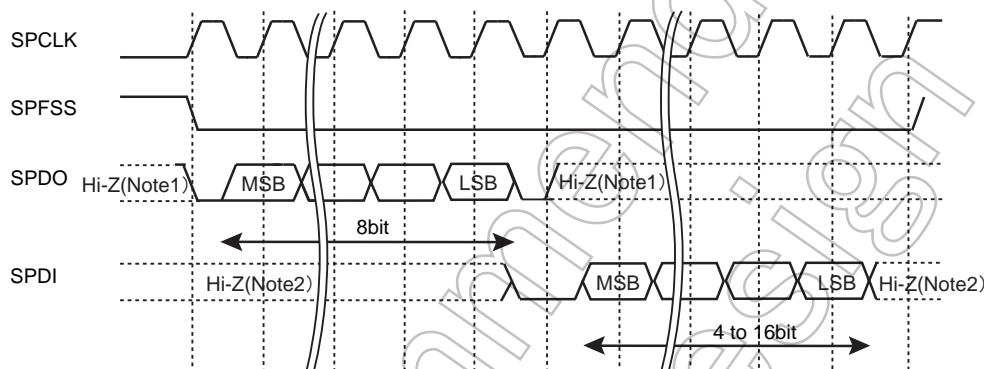
Figure 13-6 Microwire frame format (single transfer)

Note 1:   When transmission is disabled, SPDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to fix the voltage level.

Note 2:   SPDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to fix the voltage level.

Though the Microwire format is similar to the SPI format, it uses the master/slave message transmission method for half-duplex communications. Each serial transmission is started by an 8-bit control word, which is sent to the off-chip slave device. During this transmission, the SSP does not receive input data. After the message has been transmitted, the off-chip slave decodes it, and after waiting one serial clock after the last bit of the 8-bit control message has been sent, responds with the requested data. The returned data can be 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits. With this configuration, during the idle period:

·   The SPCLK signal is set to "Low".
·   SPFSS is set to "High".
·   The transmit data line SPDO is set to "Low".

A transmission is triggered by writing a control byte to the transmit FIFO. The falling edge of SPFSS causes the value stored in the bottom entry of the transmit FIFO to be transferred to the serial shift register for the transmit logic, and the MSB of the 8-bit control frame to be shifted out onto the SPDO pin.

SPFSS remains "Low" and the SPDI pin remains tristated during this transmission. The off-chip serial slave device latches each control bit into its serial shifter on the rising edge of each SPCLK.

After the last bit is latched by the slave device, the control byte is decoded during a one clock wait-state, and the slave responds by transmitting data back to the SSP. Each bit is driven onto SPDI line on the falling edge of SPCLK.

The SSP in turn latches each bit on the rising edge of SPCLK. At the end of the frame, for single transfers, the SPFSS signal is pulled "High" one clock period after the last bit has been latched in the receive serial shifter, which causes the data to be transferred to the receive FIFO.

Note: The off-chip slave device can tristate the receive line either on the falling edge of SPCLK after the
LSB has been latched by the receive shifter, or when the SPFSS pin goes "High".
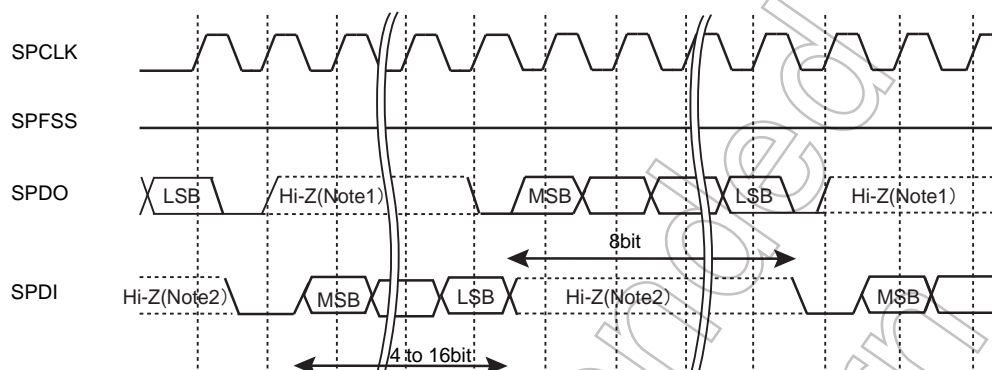
Figure 13-7 Microwire frame format (continuous transfer)

Note 1: When transmission is disabled, SPDO terminal doesn't output and is high impedance status. This terminal
needs to add suitable pull-up/down resistance to fix the voltage level.

Note 2: SPDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance sta-
tus, this terminal needs to add suitable pull-up/down resistance to fix the voltage level.

For continuous transfers, data transmission begins and ends in the same manner as a single transfer. Howev-
er, the SPFSS line is continuously asserted (held Low) and transmission of data occurs back to back.

The control byte of the next frame follows directly after the LSB of the received data from the current
frame. Each of the received values is transferred from the receive shifter on the falling edge of SPCLK, after
the LSB of the frame has been latched into the SSP.

Note: [Example of connection] The SSP does not support dynamic switching between the master and
slave in the system. Each sample SSP is configured and connected as either a master or slave.

14. Serial Bus Interface (I2C/SIO)

The TMPM363F10FG contains 4 Serial Bus Interface (I2C/SIO) channels, in which the following two operating modes are included:

- I2C bus mode (with multi-master capability)
- Clock-synchronous 8-bit SIO mode

In the I2C bus mode, the I2C/SIO is connected to external devices via SCL and SDA.

In the clock-synchronous 8-bit SIO mode, the I2C/SIO is connected to external devices via SCK, SI and SO.

The following table shows the programming required to put the I2C/SIO in each operating mode.

Table 14-1 Port settings for using serial bus interface

channel	Operating mode	pin	Port Function Register	Port Output Control Register	Port Input Control Register	Port Open Drain Output Control Register
SBI0	I2C bus mode	SCL0 :PL1 SDA0 :PL0	PLFR1[1:0] = 11	PLCR[1:0] = 11	PLIE[1:0] = 11	PLOD[1:0] = 11
	SIO mode	SCK0 :PL2 SI0 :PL1 SO0 :PL0	PLFR1[2:0] = 111	PLCR[2:0] = 101(SCK0 output) PLCR[2:0] = 001(SCK0 input)	PLIE[2:0] = 010(SCK0 output) PLIE[2:0] = 110(SCK0 input)	PLOD[2:0] = xxx
SBI1	I2C bus mode	SCL1 :PG1 SDA1 :PG0	PGFR1[1:0] = 11	PGCR[1:0] = 11	PGIE[1:0] = 11	PGOD[1:0] = 11
	SIO mode	SCK1 :PG2 SI1 :PG1 SO1 :PG0	PGFR1[2:0] = 111	PGCR[2:0] = 101(SCK1 output) PGCR[2:0] = 001(SCK1 input)	PGIE[2:0] = 010(SCK1 output) PGIE[2:0] = 110(SCK1 input)	PGOD[2:0] = xxx
SBI2	I2C bus mode	SCL2 :PG5 SDA2 :PG4	PGFR1[5:4] = 11	PGCR[5:4] = 11	PGIE[5:4] = 11	PGOD[5:4] = 11
	SIO mode	SCK2 :PG6 SI2 :PG5 SO2 :PG4	PGFR1[6:4] = 111	PGCR[6:4] = 101(SCK2 output) PGCR[6:4] = 001(SCK2 input)	PGIE[6:4] = 010(SCK2 output) PGIE[6:4] = 110(SCK2 input)	PGOD[6:4] = xxx
SBI3	I2C bus mode	SCL3 :PL5 SDA3 :PL4	PLFR3[5:4] = 11	PLCR[5:4] = 11	PLIE[5:4] = 11	PLOD[5:4] = 11
	SIO mode	SCK3 :- SI3 :- SO3 :-	-	-	-	-

Note:x: Don't care

14.1 Configuration

The configuration is shown in Figure 14-1.
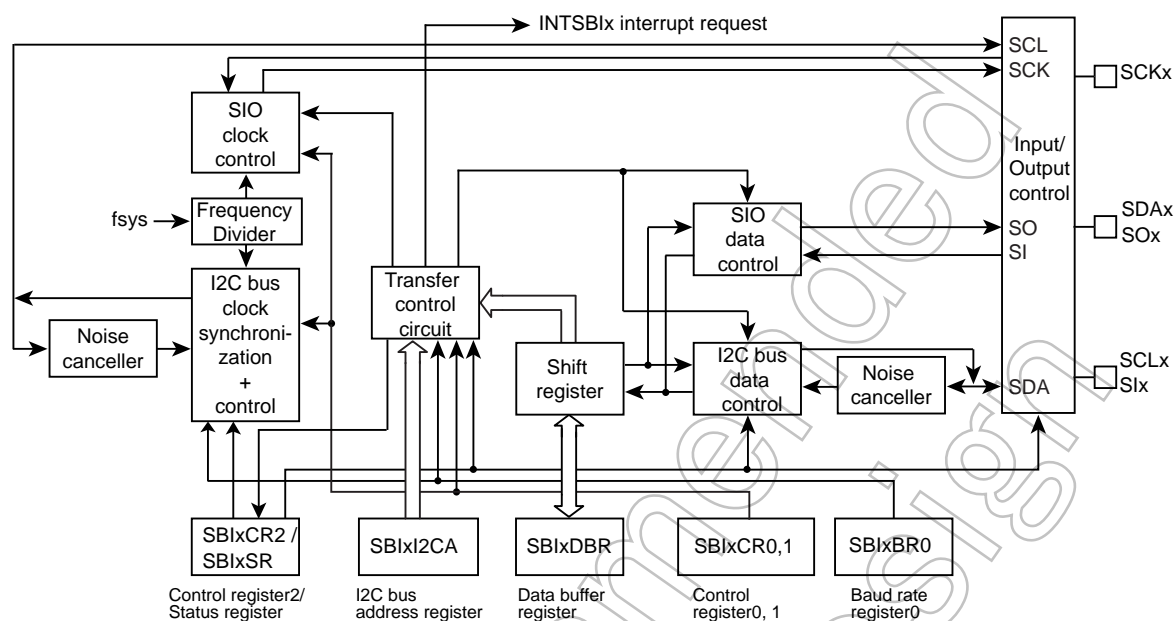
Figure 14-1 (I2C/SIO) Block Interface

14.2   Register

The following registers control the serial bus interface and provide its status information for monitoring.

The register below performs different functions depending on the mode. For details, refer to "14.4 Control Registers in the I2C Bus Mode" and "14.7 Control register of SIO mode".

14.2.1   Registers for each channel

The tables below show the registers and register addresses for each channel.

Channel x	Base Address
Channel0	0x400E_0000
Channel1	0x400E_0100
Channel2	0x400E_0200
Channel3	0x400E_0300

Register name(x=0,1,2,3,)		Address(Base+)
Control register 0	SBIxCR0	0x0000
Control register 1	SBIxCR1	0x0004
Data buffer register	SBIxDBR	0x0008
I2C bus address register	SBIxI2CAR	0x000C
Control register 2	SBIxCR2 (writing)	0x0010
Status register	SBIxSR (reading)	
Baud rate register 0	SBIxBR0	0x0014

14.3   I2C Bus Mode Data Format

Figure 14-2 shows the data formats used in the I2C bus mode.

(a) Addressing format

(b) Addressing format (with repeated start condition)

(c) Free data format (master-transmitter to slave-receiver)

Note)   S : Start condition
R/W : Direction bit
ACK : Acknowledge bit
P : Stop condition
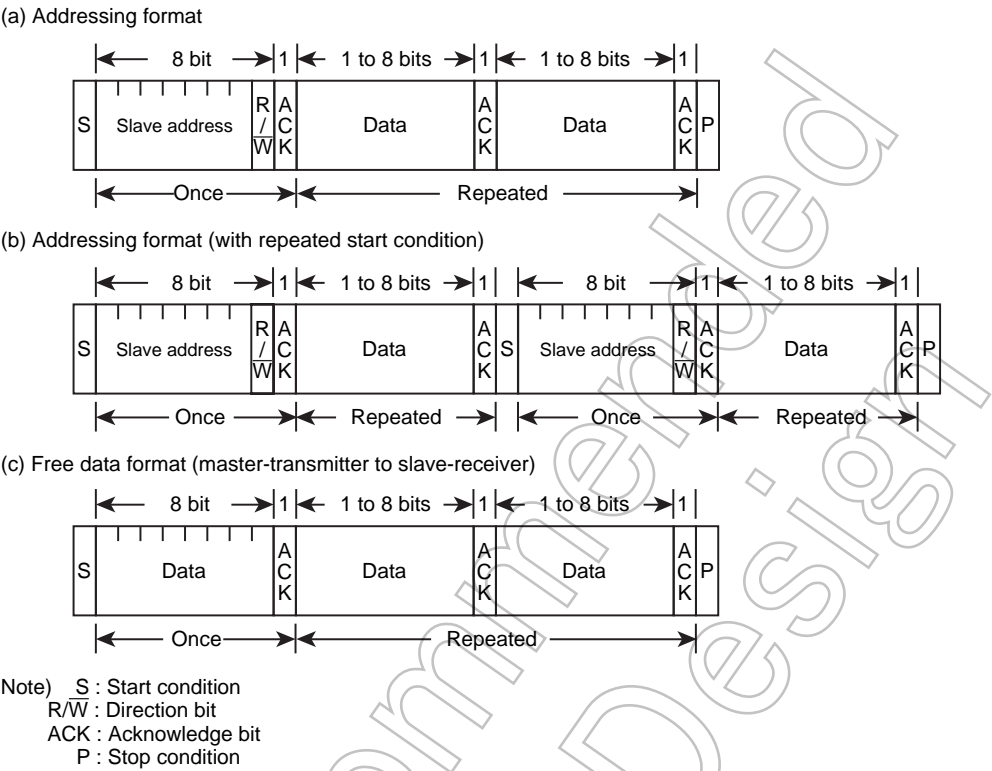
Figure 14-2 I2C Bus Mode Data Formats

14.4   Control Registers in the I2C Bus Mode

The following registers control the serial bus interface in the I2C bus mode and provide its status information for monitoring.

14.4.1   SBIxCR0(Control register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SBIEN	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	SBIEN	R/W	Serial bus interface operation 0:Disable 1:Enable To use the serial bus interface, enable this bit first. For the first time in case of setting to enable, the relevant SBI registers can be read or written. Since all clocks except SBIxCR0 stop if this bit is disabled, power consumption can be reduced by disabling this bit. If this bit is disabled after it's been enabled once, the settings of each register are retained.
6-0	-	R	Read as 0.

Note:To use the serial bus interface, enable this bit first.

14.4.2 SBIxCR1(Control register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol		BC		ACK	-	SCK2	SCK1	SCK0 / SWRMON
After reset	0	0	0	0	1	0	0	1(Note3)

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-5	BC[2:0]	R/W	Select the number of bits per transfer (Note 1)
4	ACK	R/W	Master mode 0: Acknowledgement clock pulse is not generated. 1: Acknowledgement clock pulse is generated. Slave mode 0: Acknowledgement clock pulse is not counted. 1: Acknowledgement clock pulse is counted.
3	-	R	Read as 1.
2-1	SCK[2:1]	R/W	Select internal SCL output clock frequency (Note 2).
0	SCK[0]	W	
	SWRMON	R	On reading <SWRMON>: Software reset status monitor 0:Software reset operation is in progress. 1:Software reset operation is not in progress.

BC[2:0] table:

<BC>	When <ACK> = 0 Number of clock cycles	When <ACK> = 0 Data length	When <ACK> = 1 Number of clock cycles	When <ACK> = 1 Data length
000	8	8	9	8
001	1	1	2	1
010	2	2	3	2
011	3	3	4	3
100	4	4	5	4
101	5	5	6	5
110	6	6	7	6
111	7	7	8	7

SCK table:

SCK	n	Frequency
000	n = 5	615 kHz
001	n = 6	471 kHz
010	n = 7	320 kHz
011	n = 8	195 kHz
100	n = 9	110 kHz
101	n = 10	58 kHz
110	n = 11	30 kHz
111		reserved

System Clock: fsys ($= 64MHz$)

Clock gear : fc/1

Frequency = $\dfrac{fsys}{2^n + 72}$ [Hz]

Note 1: **Clear <BC[2:0]> to "000" before switching the operation mode to the SIO mode.**

Note 2: **For details on the SCL line clock frequency, refer to "14.5.1 Serial Clock".**

Note 3: **After a reset, the <SCK[0]/SWRMON> bit is read as "1". However, if the SIO mode is selected at the SBIxCR2 register, the initial value of the <SCK[0]> bit is "0".**

Note 4: **The initial value for selecting a frequency is <SCK[2:0]>=000 and is independent of the read initial value.**

Note 5: **When <BC[2:0]>="001" and <ACK>="0" in master mode, SCL line may be fixed to "L" by falling edge of SCL line after generation of STOP condition and the other master devices can not use the bus. In the case of bus which is connected with several master devices, the bumber of bits per transfer should be set equal or more than 2 before generation of STOP condition.**

14.4.3 SBIxCR2(Control register 2)

This register serves as SBIxSR register by reading it.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MST	TRX	BB	PIN	SBIM		SWRST	
After reset	0	0	0	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	MST	W	Select master/slave 0: Slave mode 1: Master mode
6	TRX	W	Select transmit/ receive 0: Receive 1: Transmit
5	BB	W	Start/stop condition generation 0: Stop condition generated 1: Start condition generated
4	PIN	W	Clear INTSBIx interrupt request 0: – 1: Clear interrupt request
3-2	SBIM[1:0]	W	Select serial bus interface operating mode (Note) 00: Port mode (Disables a serial bus interface output) 01: SIO mode 10: I2C bus mode 11: Reserved
1-0	SWRST[1:0]	W	Software reset generation Write "10" followed by "01" to generate a reset. When writing, set <SBIM[1:0]> to "10"; I2Cbus mode.

Note: **Make sure that modes are not changed during a communication session.Ensure that the bus is free before switching the operating mode to the port mode. Ensure that the port is at the "High" level before switching the operating mode from the port mode to the I2C bus or clock-synchronous 8-bit SIO mode.**

14.4.4　SBIxSR (Status Register)

This register serves as SBIxCR2 by writing to it.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MST	TRX	BB	PIN	AL	AAS	ADO	LRB
After reset	0	0	0	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	MST	R	Master/slave selection monitor 0: Slave mode 1: Master mode
6	TRX	R	Transmit/receive selection monitor 0: Receive 1: Transmit
5	BB	R	I2C bus state monitor 0: Free 1: Busy
4	PIN	R	INTSBIx interrupt request monitor 0:Interrupt request generated 1: Interrupt request cleared
3	AL	R	Arbitration lost detection 0: − 1:Detected
2	AAS	R	Slave address match detection 0: − 1: Detected (This bit is set when the general call is detected as well.)
1	ADO	R	General call detection 0: − 1:Detected
0	LRB	R	Last received bit monitor 0:Last received bit "0" 1:Last received bit "1"

14.4.5 SBIxBR0(Serial bus interface baud rate register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	I2SBI	-	-	-	-	-	-
After reset	1	0	1	1	1	1	1	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	-	R	Read as 1.
6	I2SBI	R/W	Operation at the IDLE mode 0: Stop 1: Operate
5-1	-	R	Read as 1.
0	-	R/W	Be sure to write "0".

14.4.6 SBIxDBR (Serial bus interface data buffer register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DB							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-0	DB[7:0]	R (Receive)/ W (Transmit)	Receive data / Transmit data

Note 1: **The transmission data must be written in to the register from the MSB (bit 7). The received data is stored in the LSB.**

Note 2: **Since SBIxBDR has independent buffers for writing and reading, a written data cannot be read. Thus, read-modify-write instructions, such as bit manipulation, cannot be used.**

14.4.7   SBIxI2CAR (I2Cbus address register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SA							ALS
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-1	SA[6:0]	R/W	Set the slave address when the SBI acts as a slave device.
0	ALS	R/W	Specify address recognition mode. 0: Recognize its slave address. 1: Do not recognize its slave address (free-data format).

Note 1: **Please set the bit 0 <ALS> of I2C bus address register SBIxI2CAR to "0", except when you use a free data format. It operates as a free data format when setting it to "1". Selecting the master fixes to transmission. Selecting the slave fixes to reception.**

Note 2: **Do not set SBIxI2CAR to "0x00" in slave mode. (If SBIxI2CAR is set to "0x00", it's recognized that the slave address matches the START byte ("0x01") of the I2C standard received in slave mode.)**

14.5 Control in the I2C Bus Mode

14.5.1 Serial Clock

14.5.1.1 Clock source

SBIxCR1<SCK[2:0]> specifies the maximum frequency of the serial clock to be output from the SCL pin in the master mode.

$$t_{LOW} = 2^{n-1}/fsys + 58/fsys$$
$$t_{HIGH} = 2^{n-1}/fsys + 14/fsys$$
$$fscl = 1/(t_{LOW} + t_{HIGH})$$
$$= \frac{fsys}{2^n + 72}$$

SBIxCR1<SCK[2:0]>	n
000	5
001	6
010	7
011	8
100	9
101	10
110	11
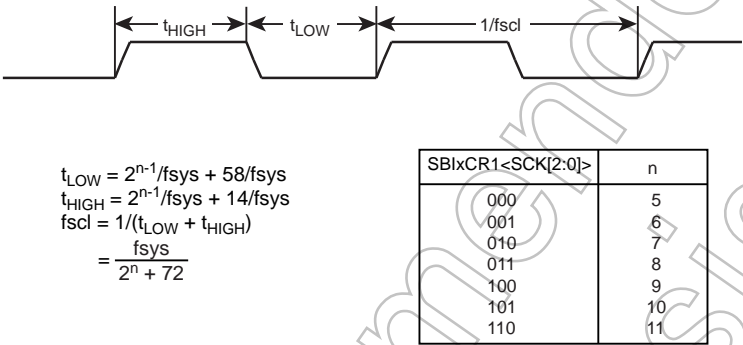
Figure 14-3 Clock source

Note: **The maximum speeds in the standard and high-speed modes are specified to 100kHz and 400kHz respectively following the communications standards. Notice that the internal SCL clock frequency is determined by the fsys used and the calculation formula shown above.**

14.5.1.2 Clock Synchronization

The I2C bus is driven by using the wired-AND connection due to its pin structure. The first master that pulls its clock line to the "Low" level overrides other masters producing the "High" level on their clock lines. This must be detected and responded by the masters producing the "High" level.

Clock synchronization assures correct data transfer on a bus that has two or more master.

For example, the clock synchronization procedure for a bus with two masters is shown below.
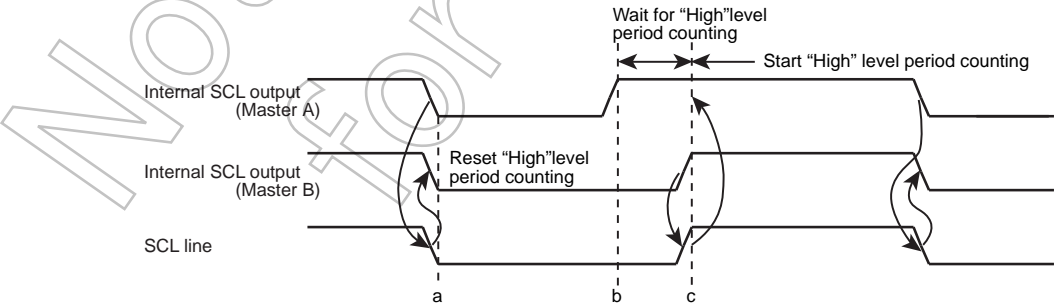
Figure 14-4 Example of Clock Synchronization

At the point a, Master A pulls its internal SCL output to the "Low" level, bringing the SCL bus line to the "Low" level. Master B detects this transition, resets its "High" level period counter, and pulls its internal SCL output level to the "Low" level.

Master A completes counting of its "Low" level period at the point b, and brings its internal SCL output to the "High" level. However, Master B still keeps the SCL bus line at the "Low" level, and Master A stops counting of its "High" level period counting.After Master A detects that Master B brings its internal SCL output to the "High" level and brings the SCL bus line to the "High" level at the point c, it starts counting of its "High" level period.

After that Master finishes counting the "High" level period, the Master pulls the SCL pin to "Low" and the SCL bus line becomes "Low".

This way, the clock on the bus is determined by the master with the shortest "High" level period and the master with the longest "Low" level period among those connected to the bus.

14.5.2 Setting the Acknowledgement Mode

Setting SBIxCR1<ACK> to "1" selects the acknowledge mode.When operating as a master, the SBI adds one clock for acknowledgment signal. In slave mode, the clock for acknowledgement signals is counted. In transmitter mode, the SBI releases the SDAx pin during clock cycle to receive acknowledgement signals from the receiver. In receiver mode, the SBI pulls the SDAx pin to the "Low" level during the clock cycle and generates acknowledgement signals. Also in slave mode, if a general-call address is received, the SBI pulls the SDAx pin to the "Low" level during the clock cycle and generates acknowledgement signals.

By setting <ACK> to "0", the non-acknowledgment mode is activated. When operating as a master, the SBI does not generate clock for acknowledgement signals. In slave mode, the clock for acknowledgement signals is counted.

14.5.3 Setting the Number of Bits per Transfer

SBIxCR1<BC[2:0]> specifies the number of bits of the next data to be transmitted or received.

Under the start condition, <BC[2:0]> is set to "000", causing a slave address and the direction bit to be transferred in a packet of eight bits. At other times, <BC[2:0]> keeps a previously programmed value.

14.5.4 Slave Addressing and Address Recognition Mode

Setting "0" to SBIxI2CAR<ALS> and a slave address in SBIxI2CAR<SA[6:0]> sets addressing format, and then the SBI recognizes a slave address transmitted by the master device and receives data in the addressing format.

If <ALS> is set to "1", the SBI does not recognize a slave address and receives data in the free data format. In the case of free data format, a slave address and a direction bit are not recognized; they are recognized as data immediately after generation of the start condition.

14.5.5 Operating mode

The setting of SBIxCR2<SBIM[1:0]> controls the operating mode. To operate in I2C mode, ensure that the serial bus interface pins are at "High" level before setting <SBIM[1:0]> to "10". Also, ensure that the bus is free before switching the operating mode to the port mode.

14.5.6 Configuring the SBI as a Transmitter or a Receiver

Setting SBIxCR2<TRX> to "1" configures the SBI as a transmitter. Setting <TRX> to "0" configures the SBI as a receiver.

At the slave mode:

- · when data is transmitted in the addressing format.
- · when the received slave address matches the value specified at SBIxI2CAR.
- · when a general-call address is received; i.e., the eight bits following the start condition are all zeros.

If the value of the direction bit (R/$\overline{W}$) is "1", <TRX> is set to "1" by the hardware. If the bit is "0", <TRX> is set to "0".

As a master device, the SBI receives acknowledgement from a slave device. If the direction bit of "1" is transmitted, <TRX> is set to "0" by the hardware. If the direction bit is "0", <TRX> changes to "1". If the SBI does not receive acknowledgement, <TRX> retains the previous value.

<TRX> is cleared to "0" by the hardware when it detects the stop condition on the bus or the arbitration lost.

If SBI is used in free data format, <TRX> is not changed by the hardware.

14.5.7 Configuring the SBI as a Master or a Slave

Setting SBIxCR2<MST> to "1" configures the SBI to operate as a master device.

Setting <MST> to "0" configures the SBI as a slave device. <MST> is cleared to "0" by the hardware when it detects the stop condition on the bus or the arbitration lost.

14.5.8 Generating Start and Stop Conditions

When SBIxSR<BB> is "0", writing "1" to SBIxCR2<MST, TRX, BB, PIN> causes the SBI to start a sequence for generating the start condition and to output the slave address and the direction bit prospectively written in the data buffer register. <ACK> must be set to "1" in advance.
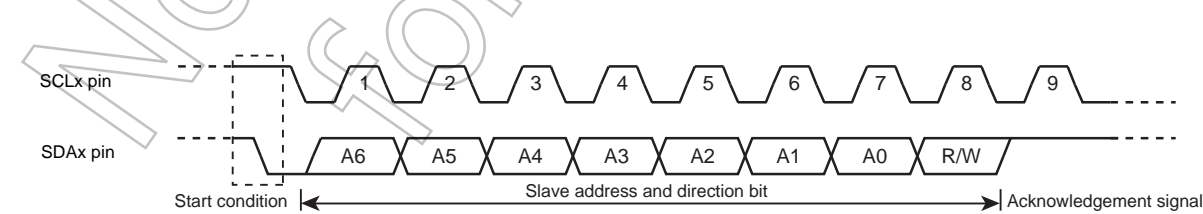
Figure 14-5 Generating the Start Condition and a Slave Address

When <BB> is "1", writing "1" to <MST, TRX, PIN> and "0" to <BB> causes the SBI to start a sequence for generating the stop condition on the bus. The contents of <MST, TRX, BB, PIN> should not be altered until the stop condition appears on the bus.

If SCL bus line is pulled "Low" by other devices when the stop condition is generated, the stop condition is generated after the SCL line is released.
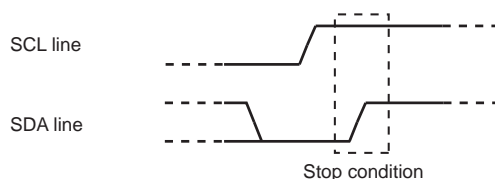
Figure 14-6 Generating the Stop Condition

SBIxSR<BB> can be read to check the bus state. <BB> is set to "1" when the start condition is detected on the bus (the bus is busy), and cleared to "0" when the stop condition is detected (the bus is free).

14.5.9   Interrupt Service Request and Release

In master mode, a serial bus interface request (INTSBIx) is generated when the transfer of the number of clock cycles set by <BC> and <ACK> is completed.

In slave mode, INTSBIx is generated under the following conditions.

- · After output of the acknowledge signal which is generated when the received slave address matches the slave address set to SBIxI2CAR<SA[6:0]>.
- · After the acknowledge signal is generated when a general-call address is received.
- · When the slave address matches or a data transfer is completed after receiving a general-call address.

In the address recognition mode (<ALS> = "0"), INTSBIx is generated when the received slave address matches the values specified at SBIxI2CAR or when a general-call (eight bits data following the start condition is all "0") is received.

When an interrupt request (INTSBIx) is generated, SBIxCR2<PIN> is cleared to "0". While <PIN> is cleared to "0", the SBI pulls the SCL line to the "Low" level.

<PIN> is set to "1" when data is written to or read from SBIxDBR. It takes a period of t_{LOW} for the SCL line to be released after <PIN> is set to "1". When the program writes "1" to <PIN>, it is set to "1". However, writing "0" does not clear this bit to "0".

Note: When arbitration is lost in master mode, <PIN> is not cleared to "0" if the slave address does not match (INTSBIx is generated).

14.5.10   Arbitration Lost Detection Monitor

The I2C bus has the multi-master capability (there are two or more masters on a bus), and requires the bus arbitration procedure to ensure correct data transfer.

A master that attempts to generate the start condition while the bus is busy loses bus arbitration, with no start condition occurring on the SDA and SCL lines.The I2C-bus arbitration takes place on the SDA line.

The arbitration procedure for two masters on a bus is shown below.

Up until the point a, Master A and Master B output the same data. At the point a, Master A outputs the "Low" level and Master B outputs the "High" level.

Then Master A pulls the SDA bus line to the "Low" level because the line has the wired-AND connection. When the SCL line goes high at the point b, the slave device reads the SDA line data, i.e., data transmitted by Master A. At this time, data transmitted by Master B becomes invalid.

This condition of Master B is called "Arbitration Lost". Master B releases its SDA pin, so that it does not affect the data transfer initiated by another master. If two or more masters have transmitted exactly the same first data word, the arbitration procedure continues with the second data word.
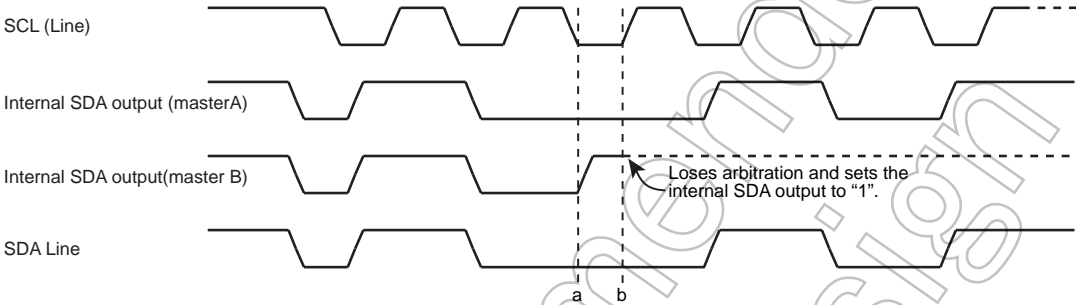
Figure 14-7 Lost Arbitration

A master compares the SDA bus line level and the internal SDA output level at the rising of the SCL line. If there is a difference between these two values, Arbitration Lost occurs and SBIxSR<AL> is set to "1".

When <AL> is set to "1", SBIxSR<MST, TRX> are cleared to "0", causing the SBI to operate as a slave receiver.Therefore, the serial bus interface circuit stops the clock output during data transfer after <AL> is set to "1".

<AL> is cleared to "0" when data is written to or read from SBIxDBR or data is written to SBIxCR2.
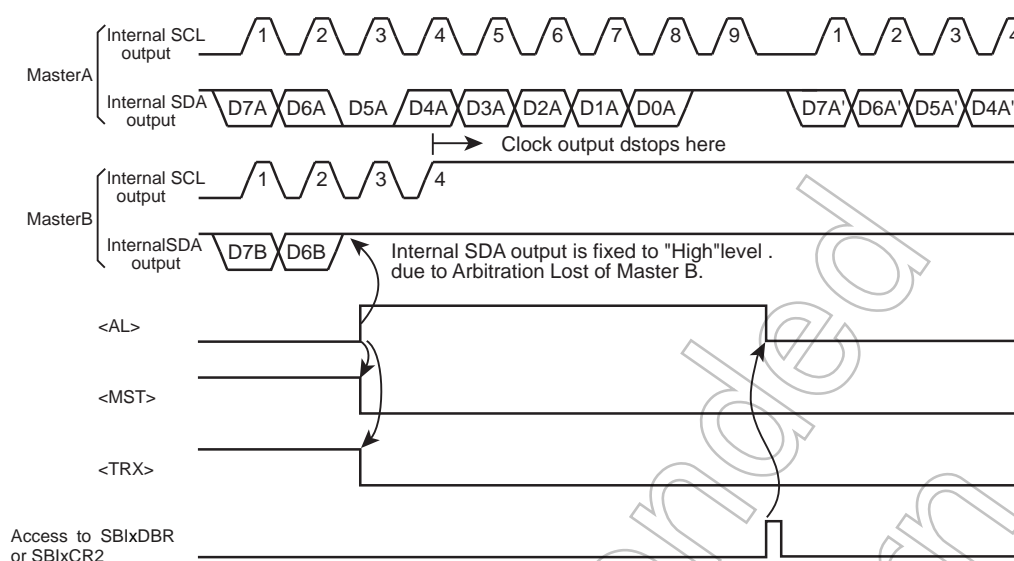
Figure 14-8 Example of Master B Lost Arbitration (D7A = D7B, D6A = D6B)

14.5.11   Slave Address Match Detection Monitor

When the SBI operates as a slave device in the address recognition mode (SBIxI2CAR<ALS>="0"), SBIxSR<AAS> is set to "1" on receiving the general-call address or the slave address that matches the value specified at SBIxI2CAR.

When <ALS> is "1", <AAS> is set to "1" when the first data word has been received. <AAS> is cleared to "0" when data is written to or read from SBIxDBR.

14.5.12   General-call Detection Monitor

When the SBI operates as a slave device, SBIxSR<ADO> is set to "1" when it receives the general-call address; i.e., the eight bits following the start condition are all zeros.

<ADO> is cleared to "0" when the start or stop condition is detected on the bus.

14.5.13   Last Received Bit Monitor

SBIxSR<LRB> is set to the SDA line value that was read at the rising of the SCL line.

In the acknowledgment mode, reading SBIxSR<LRB> immediately after generation of the INTSBIx interrupt request causes ACK signal to be read.

14.5.14   Data Buffer Register (SBIxDBR)

Reading or writing SBIxDBR initiates reading received data or writing transmitted data.

When the SBI is acting as a master, setting a slave address and a direction bit to this register generates the start condition.

14.5.15 Baud Rate Register (SBIxBR0)

The SBIxBR0<I2SBI> register determines if the SBI operates or not when it enters the IDLE mode.

This register must be programmed before executing an instruction to switch to the standby mode.

14.5.16 Software Reset

If the serial bus interface circuit locks up due to external noise, it can be initialized by using a software reset.

Writing "10" followed by "01" to SBIxCR2<SWRST[1:0]> generates a reset signal that initializes the serial bus interface circuit. When writing, set <SBIM[1:0]> to "10"; I2Cbus mode. After a reset, all control registers and status flags are initialized to their reset values. When the serial bus interface is initialized, <SWRST> is automatically cleared to "0".

Note: A software reset causes the SBI operating mode to switch from the I2C mode to the port mode.

14.6   Data Transfer Procedure in the I2C Bus ModeI2C

14.6.1   Device Initialization

First, program SBIxCR1<ACK, SCK[2:0]>. Writing "000" to SBIxCR1<BC[2:0]> at the time.

Next, program SBIxI2CAR by specifying a slave address at <SA[6:0]> and an address recognition mode at <ALS>. (<ALS> must be cleared to "0" when using the addressing format).

To configure the Serial Bus Interface as a slave receiver, ensure that the serial bus interface pin is at "High" first. Then write "0" to SBIxCR2<MST, TRX, BB>, "1" to <PIN>, "10" to <SBIM[1:0]> and "0" to the bit 1 and 0.

Note: Initialization of the serial bus interface circuit must be completed within a period that any device does not generate start condition after all devices connected to the bus were initialized. If this rule is not followed, data may not be received correctly because other devices may start transfer before the initialization of the serial bus interface circuit is completed.

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	0	0	X	0	X	X	X	Specifies ACK and SCL clock.
SBIxI2CAR	←	X	X	X	X	X	X	X	X	Specifies a slave address and an address recognition mode.
SBIxCR2	←	0	0	0	1	1	0	0	0	Configures the SBI as a slave receiver.

Note: X; Don't care

14.6.2   Generating the Start Condition and a Slave Address

14.6.2.1   Master mode

In the master mode, the following steps are required to generate the start condition and a slave address.

First, ensure that the bus is free (<BB> = "0"). Then, write "1" to SBIxCR1<ACK> to select the acknowledgment mode. Write to SBIxDBR a slave address and a direction bit to be transmitted.

When <BB> = "0", writing "1111" to SBIxCR2<MST, TRX, BB, PIN> generates the start condition on the bus. Following the start condition, the SBI generates nine clocks from the SCL pin. The SBI outputs the slave address and the direction bit specified at SBIxDBR with the first eight clocks, and releases the SDA line in the ninth clock to receive an acknowledgment signal from the slave device.

The INTSBIx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to "0". In the master mode, the SBI holds the SCL line at the "Low" level while <PIN> is = "0".<TRX> changes its value according to the transmitted direction bit at generation of the INTSBIx interrupt request, provided that an acknowledgment signal has been returned from the slave device.

Note: To output salve address, check with software that the bus is free before writing to SBIxDBR. If this rule is not followed, data being output on the bus may get ruined.

Settings in main routine

```
                        7   6   5   4   3   2   1   0

Reg.        ←   SBIxSR
Reg.        ←   Reg. e 0x20
if Reg.     ≠   0x00                                    Ensures that the bus is free.

Then

SBIxCR1     ←   X   X   X   1   0   X   X   X          Selects the acknowledgement mode.
SBIxDBR     ←   X   X   X   X   X   X   X   X          Specifies the desired slave address and direction.
SBIxCR2     ←   1   1   1   1   1   0   0   0          Generates the start condition.
            Example of INTSBI0 interrupt routine
            Clears the interrupt request.
            Processing
            End of interrupt
```

14.6.2.2   Slave mode

In the slave mode, the SBI receives the start condition and a slave address.

After receiving the start condition from the master device, the SBI receives a slave address and a direction bit from the master device during the first eight clocks on the SCL line.

If the received address matches its slave address specified at SBIxI2CAR or is equal to the general-call address, the SBI pulls the SDA line to the "Low" level during the ninth clock and outputs an acknowledgment signal.

The INTSBIx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to "0". In the slave mode, the SBI holds the SCL line at the "Low" level while <PIN> is "0".
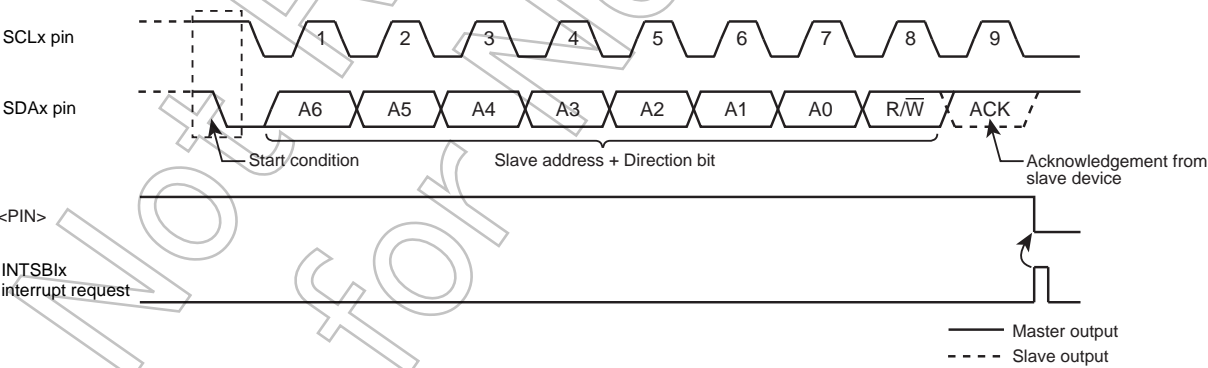
Figure 14-9 Generation of the Start Condition and a Slave Address

14.6.3   Transferring a Data Word

At the end of a data word transfer, the INTSBIx interrupt is generated to test <MST> to determine whether the SBI is in the master or slave mode.

14.6.3.1   Master mode (<MST> = "1")

Test <TRX> to determine whether the SBI is configured as a transmitter or a receiver.

(1)   Transmitter mode (<TRX> = "1")

Test <LRB>. If <LRB> is "1", that means the receiver requires no further data.

The master then generates the stop condition as described later to stop transmission.

If <LRB> is "0", that means the receiver requires further data.If the next data to be transmitted has eight bits, the data is written into SBIxDBR. If the data has different length, <BC[2:0]> and <ACK> are programmed and the transmit data is written into SBIxDBR.Writing the data makes <PIN> to "1", causing the SCL pin to generate a serial clock for transferring a next data word, and the SDA pin to transfer the data word.

After the transfer is completed, the INTSBIx interrupt request is generated, <PIN> is cleared to "0", and the SCL pin is pulled to the "Low" level.

To transmit more data words, test <LRB> again and repeat the above procedure.

INTSBIx interrupt

if MST = 0

Then go to the slave-mode processing.

if TRX = 0

Then go to the receiver-mode processing.

if LRB = 0

Then go to processing for generating the stop condition.

SBIxCR1   ←   X   X   X   X   0   X   X   X   Specifies the number of bits to be transmitted and specify whether ACK is required.

SBIxDBR   ←   X   X   X   X   X   X   X   X   Writes the transmit data.

End of interrupt processing.

Note: **X; Don't care**
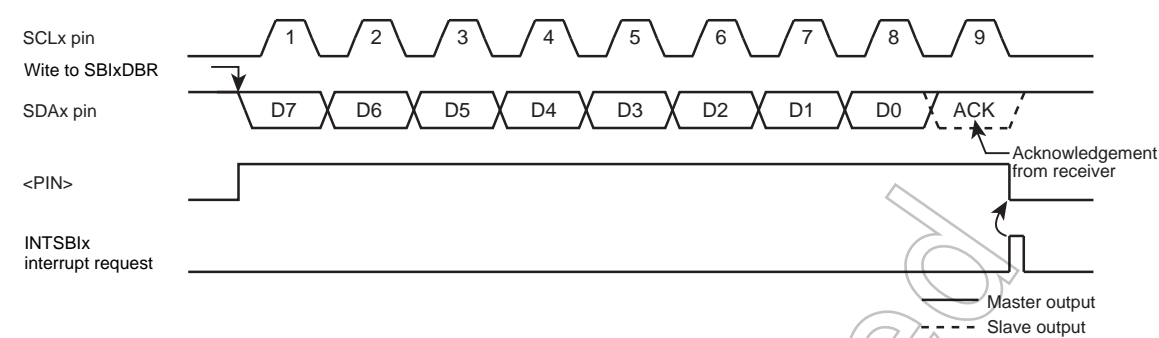
Figure 14-10 <BC[2:0]>= "000",<ACK>= "1" (Transmitter Mode)

(2) Receiver mode (<TRX> = "0")

If the next data to be transmitted has eight bits, the transmit data is written into SBIxDBR.

If the data has different length, <BC[2:0]> and <ACK> are programmed and the received data is read from SBIxDBR to release the SCL line. (The data read immediately after transmission of a slave address is undefined.)On reading the data, <PIN> is set to "1", and the serial clock is output to the SCL pin to transfer the next data word.In the last bit, when the acknowledgment signal becomes the "Low" level, "0" is output to the SDA pin.

After that, the INTSBIx interrupt request is generated, and <PIN> is cleared to "0", pulling the SCL pin to the "Low" level.Each time the received data is read from SBIxDBR, one-word transfer clock and an acknowledgement signal are output.
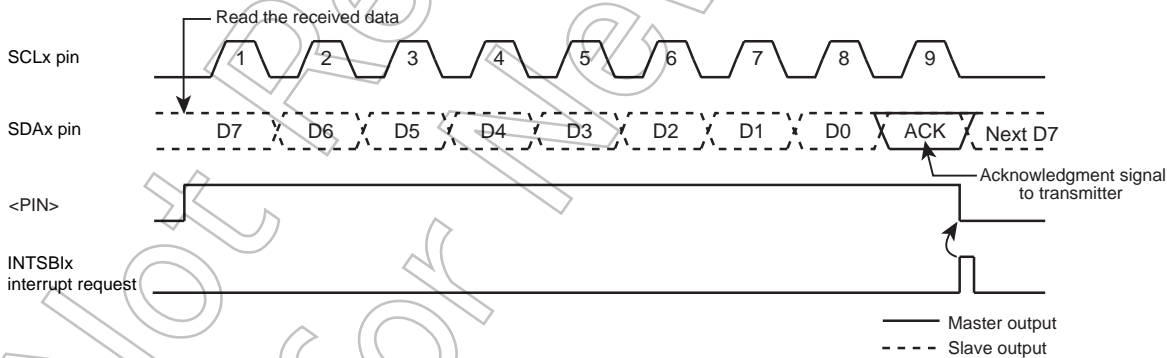
Figure 14-11 <BC[2:0]>= "000",<ACK>= "1" (Receiver Mode)

To terminate the data transmission from the transmitter, <ACK> must be cleared to "0" immediately before reading the data word second to last.

This disables generation of an acknowledgment clock for the last data word.

When the transfer is completed, an interrupt request is generated. After the interrupt processing, <BC[2:0]> must be set to "001" and the data must be read so that a clock is generated for 1-bit transfer.

At this time, the master receiver holds the SDA bus line at the "High" level, which signals the end of transfer to the transmitter as an acknowledgment signal.

In the interrupt processing for terminating the reception of 1-bit data, the stop condition is generated to terminate the data transfer.
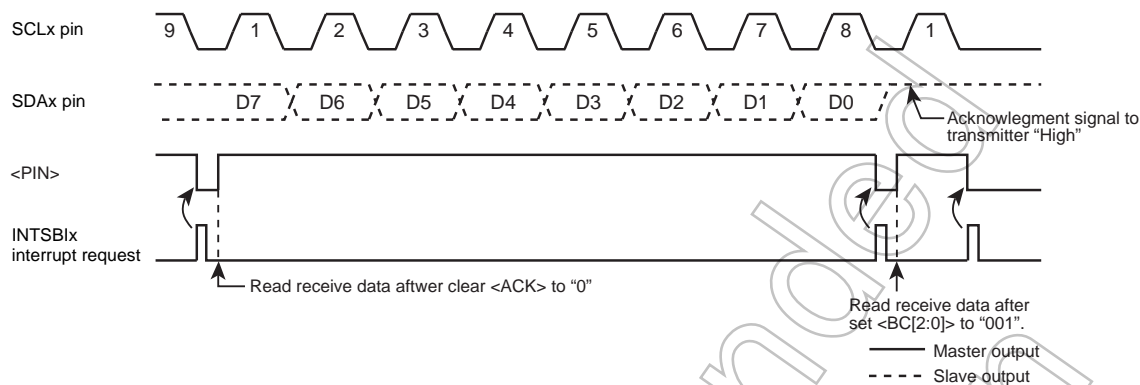
Figure 14-12 Terminating Data Transmission in the Master Receiver Mode

Example: When receiving N data word

INTSBIx interrupt (after data transmission)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	X	X	X	X	0	X	X	X	Sets the number of bits of data to be received and specify whether ACK is required.
Reg.	←	SBIxDBR								Reads dummy data.

End of interrupt

INTSBIx interrupt (first to (N-2)th data reception)

		7	6	5	4	3	2	1	0	
Reg.	←	SBIxDBR								Reads the first to (N-2)th data words.

End of interrupt

INTSBIx interrupt ((N-1)th data reception)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	X	X	X	0	0	X	X	X	Disables generation of acknowledgement clock.
Reg.	←	SBIxDBR								Reads the (N-1)th data word.

End of interrupt

INTSBIx interrupt (Nth data reception)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	0	1	0	0	X	X	X	Disables generation of acknowledgement clock.
Reg.	←	SBIxDBR								Reads the Nth data word.

End of interrupt

INTSBIx interrupt (after completing data reception)

Processing to generate the stop condition.   Terminates the data transmission.

End of interrupt

Note: X; Don't care

14.6.3.2 Slave mode (<MST> = "0")

In the slave mode, the SBI generates the INTSBIx interrupt request on four occasions:

1) when the SBI has received any slave address from the master.

2) when the SBI has received a general-call address.

3) when the received slave address matches its address.

4) when a data transfer has been completed in response to a general-call.

Also, if the SBI detects Arbitration Lost in the master mode, it switches to the slave mode.

Upon the completion of data word transfer in which Arbitration Lost is detected, the INTSBIx interrupt request is generated, <PIN> is cleared to "0", and the SCL pin is pulled to the "Low" level.

When data is written to or read from SBIxDBR or when <PIN> is set to "1", the SCLx pin is released after a period of t_{LOW}.

In the slave mode, the normal slave mode processing or the processing as a result of Arbitration Lost is carried out.

SBIxSR<AL>, <TRX>, <AAS> and <ADO> are tested to determine the processing required.

"Table 14-2 Processing in Slave Mode"shows the slave mode states and required processing.

Example: When the received slave address matches the SBI's own address and the direction bit is "1" in the slave receiver mode.

INTSBIx interrupt

```
if TRX = 0
Then go to other processing.
if AL = 0
Then go to other processing.
if AAS = 0
Then go to other processing.
SBIxCR1    ←    X    X    X    1    0    X    X    X        Sets the number of bits to be transmitted.
SBIxDBR    ←    X    X    X    X    X    X    X    X        Sets the transmit data.

    Note:X; Don't care
```

Table 14-2 Processing in Slave Mode

<TRX>	<AL>	<AAS>	<ADO>	State	Processing
1	1	1	0	Arbitration Lost is detected while the slave address was being transmitted and the SBI received a slave address with the direction bit "1" transmitted by another master.	Set the number of bits in a data word to <BC[2:0]> and write the transmit data into SBIxDBR.
	0	1	0	In the slave receiver mode, the SBI received a slave address with the direction bit "1" transmitted by the master.	
		0	0	In the slave transmitter mode, the SBI has completed a transmission of one data word.	Test LRB. If it has been set to "1", that means the receiver does not require further data. Set <PIN> to 1 and reset <TRX> to 0 to release the bus. If <LRB> has been reset to "0", that means the receiver requires further data. Set the number of bits in the data word to <BC[2:0]> and write the transmit data to the SBIxDBR.
0	1	1	1/0	Arbitration Lost is detected while a slave address is being transmitted, and the SBI receives either a slave address with the direction bit "0" or a general-call address transmitted by another master.	Read the SBIxDBR (a dummy read) to set <PIN> to 1, or write "1" to <PIN>.
		0	0	Arbitration Lost is detected while a slave address or a data word is being transmitted, and the transfer is terminated.	
	0	1	1/0	In the slave receiver mode, the SBI received either a slave address with the direction bit "0" or a general-call address transmitted by the master.	
		0	1/0	In the slave receiver mode, the SBI has completed a reception of a data word.	Set the number of bits in the data word to <BC[2:0].> and read the received data from SBIxDBR.

14.6.4   Generating the Stop Condition

When SBIxSR<BB> is "1", writing "1" to SBIxCR2<MST, TRX, PIN> and "0" to <BB> causes the SBI to start a sequence for generating the stop condition on the bus.

Do not alter the contents of <MST, TRX, BB, PIN> until the stop condition appears on the bus.

If another device is holding down the SCL bus line, the SBI waits until the SCL line is released.

After that, the SDA pin goes "High", causing the stop condition to be generated.
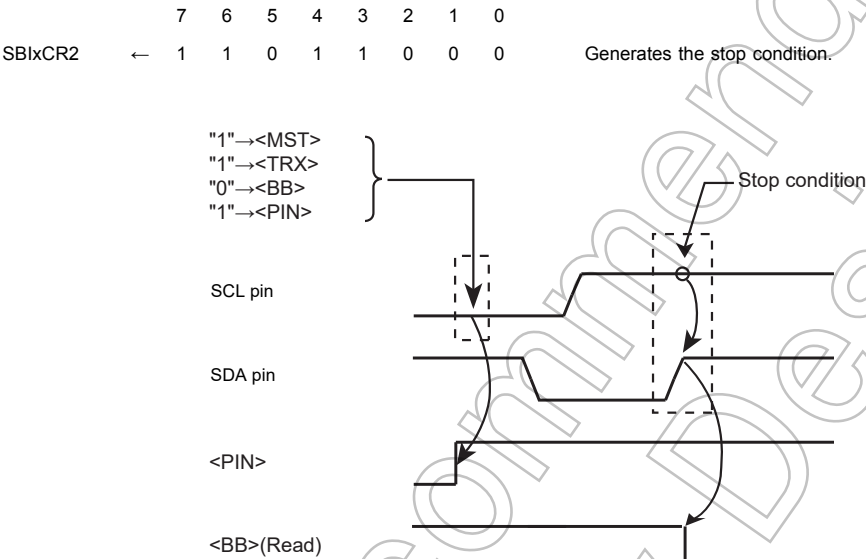
Figure 14-13 Generating the Stop Condition

14.6.5   Restart Procedure

Restart is used when a master device changes the data transfer direction without terminating the transfer to a slave device.The procedure of generating a restart in the master mode is described below.

First, write SBIxCR2<MST, TRX, BB> to "0" and write "1" to <PIN> to release the bus. At this time, the SDAx pin is held at the "High" level and the SCLx pin is released. Because no stop condition is generated on the bus, other devices recognize that the bus is busy.

Then, test SBIxSR<BB> and wait until it becomes "0" to ensure that the SCLx pin is released.

Next, test <LRB> and wait until it becomes "1" to ensure that no other device is pulling the SCLx bus line to the "Low" level.

Once the bus is determined to be free by following the above procedures, follow the procedures described in "14.6.2 Generating the Start Condition and a Slave Address"to generate the start condition.

To satisfy the setup time of restart, at least 4.7μs wait period (in the standard mode) must be created by the software after the bus is determined to be free.

Note 1: **Do not write <MST> to "0" when it is "0". (Restart cannot be initiated.)**

Note 2: **When the master device is acting as a receiver, data transmission from the slave device which serves as a transmitter must be completed before generating a restart. To complete data transfer, slave device must receive a "High" level acknowledge signal. For this reason, <LBR> before generating a restart becomes "1", the rising edge of the SCL line is not detected even <LBR>=**

"1" is confirmed by following the restart procedure. To check the status of the SCL line, read the port.

```
                    7   6   5   4   3   2   1   0
SBIxCR2     ←   0   0   0   1   1   0   0   0        Releases the bus.

if SBIxSR<BB> ≠ 0                                    Checks that the SCL pin is released.
Then
if SBIxSR<LRB> ≠ 1                                   Checks that no other device is pulling the SCL pin to the "Low".
Then
4.7 µs Wait
SBIxCR1     ←   X   X   X   1   0   X   X   X        Selects the acknowledgment mode.
SBIxDBR     ←   X   X   X   X   X   X   X   X        Sets the desired slave address and direction.
SBIxCR2     ←   1   1   1   1   1   0   0   0        Generates the start condition.
```

Note:X; Don't care
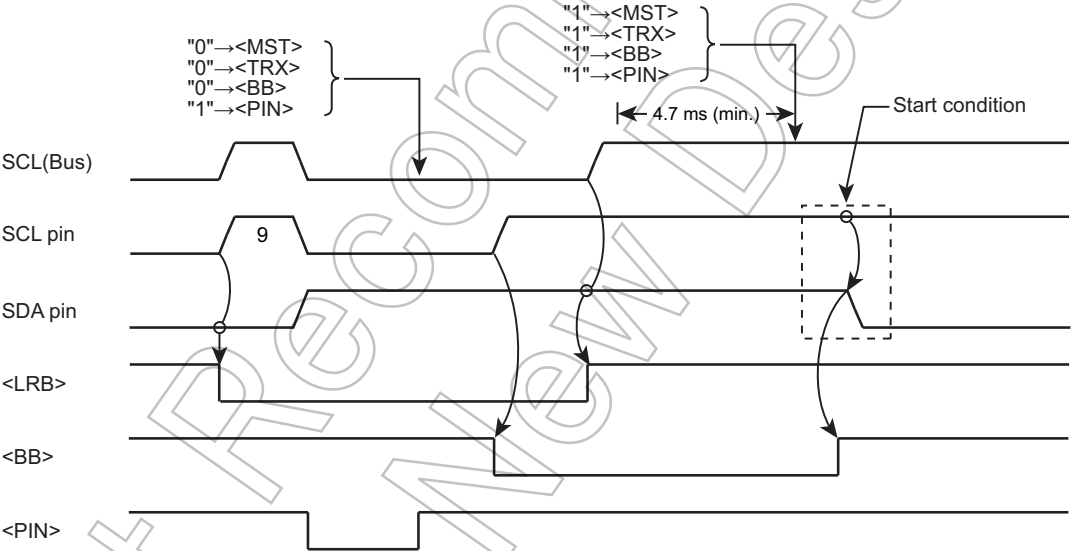
Figure 14-14 Timing Chart of Generating a Restart

14.7 Control register of SIO mode

The following registers control the serial bus interface in the clock-synchronous 8-bit SIO mode and provide its status information for monitoring.

14.7.1 SBIxCR0(control register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SBIEN	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	SBIEN	R/W	Serial bus interface operation. 0:Disable 1: Enable Enable this bit before using the serial bus interface. If this bit is disabled, power consumption can be reduced because all clocks except SBIxCR0 stop. If the serial bus interface operation is enabled and then disabled, the settings will be maintained in each register.
6-0	-	R	Read as 0.

14.7.2   SBIxCR1(Control register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SIOS	SIOINH	SIOM		-		SCK	
After reset	0	0	0	0	1	0	0	0(Note 1)

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	SIOS	R/W	Transfer Start/Stop 0: Stop 1: Start
6	SIOINH	R/W	Transfer 0: Continue 1: Forced termination
5-4	SIOM[1:0]	R/W	Select transfer mode 00: Transmit mode 01: Reserved 10:Transmit/receive mode 11:Receive mode
3	-	R	Read as 1.
2-0	SCK[2:0]	R/W	On writing <SCK[2:0]>: Select serial clock frequency. (Note 1)

$$
\begin{array}{lll}
000 & n = 3 & 4\ \text{MHz} \\
001 & n = 4 & 2\ \text{MHz} \\
010 & n = 5 & 1\ \text{MHz} \\
011 & n = 6 & 500\ \text{kHz} \\
100 & n = 7 & 250\ \text{kHz} \\
101 & n = 8 & 125\ \text{kHz} \\
110 & n = 9 & 62.5\ \text{kHz} \\
111 & - & \text{External clock}
\end{array}
$$

System clock: fsys (= 64MHz)
Clock gear: fc/1
$\text{Frequency} = \dfrac{\text{fsys}/2}{2^{n}}$ [Hz]

Note 1: **After a reset, the <SCK[0]> bit is read as "1". However, if the SIO mode is selected at the SBIxCR2 register, the initial value is read as "0". In this document, the value written in the column "after reset" is the value after setting the SIO mode in the initial state. The descriptions of the SBIxCR2 register and the SBIxSR register are the same.**

Note 2: Set <SIOS> to "0" and <SIOINH> to "1" before programming the transfer mode and the serial clock.

14.7.3 SBIxDBR (Data buffer register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DB							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-0	DB[7:0]	R	Receive data
		W	Transmit data

Note 1: **The transmission data must be written in to the register from the MSB (bit 7). The received data is stored in the LSB.**

Note 2: **Since SBIxDBR has independent buffers for writing and reading, a written data cannot be read. Thus, read-modify-write instructions, such as bit manipulation, cannot be used.**

14.7.4 SBIxCR2(Control register 2)

This register serves as SBIxSR register by writing to it.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	SBIM		-	-
After reset	1(Note 1)	1(Note 1)	1(Note 1)	1(Note 1)	0	0	1(Note 1)	1(Note 1)

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7-4	-	R	Read as 1. (Note 1)
3-2	SBIM[1:0]	W	Select serial bus interface operating mode (Note 2) 00: Port mode 01: SIO mode 10: I2Cbus mode 11: Reserved
1-0	-	R	Read as 1. (Note 1)

Note 1: **In this document, the value written in the column "after reset" is the value after setting the SIO mode in the initial state.**

Note 2: **Make sure that modes are not changed during a communication session.**

14.7.5 SBIxSR (Status Register)

This register serves as SBIxCR2 by writing to it.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	SIOF	SEF	-	-
After reset	1(Note 1)	1(Note 1)	1(Note 1)	1(Note 1)	0	0	1(Note 1)	1(Note 1)

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-4	-	R	Read as 1.(Note 1)
3	SIOF	R	Serial transfer status monitor. 0: Completed 1: In progress
2	SEF	R	Shift operation status monitor 0: Completed. 1: In progress
1-0	-	R	Read as 1. (Note 1)

Note: **In this document, the value written in the column "after reset" is the value after setting the SIO mode in the initial state.**

14.7.6   SBIxBR0 (Baud rate register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	I2SBI	-	-	-	-	-	-
After reset	1	0	1	1	1	1	1	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	-	R	Read as 1.
6	I2SBI	R/W	Operation in IDLE mode. 0: Stop 1: Operate
5-1	-	R	Read as 1.
0	-	R/W	Make sure to write "0".

14.8 Control in SIO mode

14.8.1 Serial Clock

14.8.1.1 Clock source

Internal or external clocks can be selected by programming SBIxCR1<SCK[2:0]>.

(1) Internal clocks

In the internal clock mode, one of the seven frequencies can be selected as a serial clock, which is output to the outside through the SCKx pin.

At the beginning of a transfer, the SCKx pin output becomes the "High" level.

If the program cannot keep up with this serial clock rate in writing the transmit data or reading the received data, the SBI automatically enters a wait period. During this period, the serial clock is stopped automatically and the next shift operation is suspended until the processing is completed.
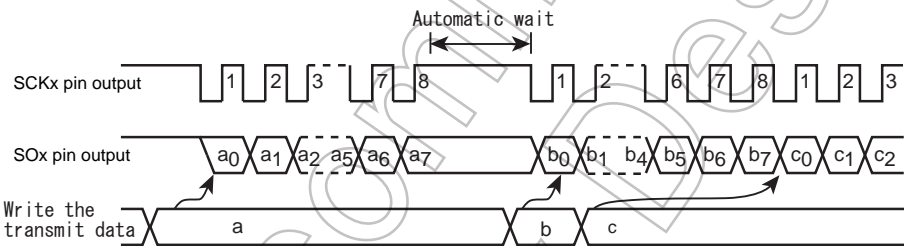
Figure 14-15 Automatic Wait

(2) External clock (<SCK[2:0]> = "111")

The SBI uses an external clock supplied from the outside to the SCKx pin as a serial clock.

For proper shift operations, the serial clock at the "High" and "Low" levels must have the pulse widths as shown below.

$f_{SCKL}, f_{SCKH} > 4/f_{sys}$
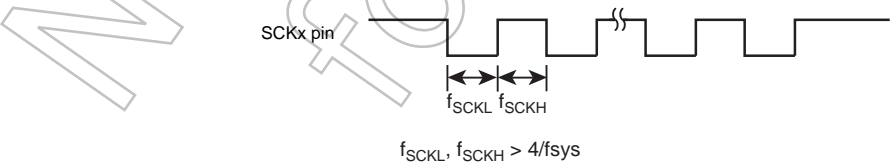
Figure 14-16 Maximum Transfer Frequency of External Clock Input

14.8.1.2   Shift Edge

Leading-edge shift is used in transmission. Trailing-edge shift is used in reception.

- Leading-edge shift

    Data is shifted at the leading edge of the serial clock (or the falling edge of the SCKx pin input/output).

- Trailing-edge shift

    Data is shifted at the trailing edge of the serial clock (or the rising edge of the SCKx pin input/output).

SCKx pin	
SOx pin	bit 0, bit 1, bit 2, bit 3, bit 4, bit 5, bit 6, bit 7
Shift register	76543210, *7654321, **765432, ***76543, ****7654, *****765, ******76, *******7

(a) Leading-edge

SCKx pin	
SIx pin	bit 0, bit 1, bit 2, bit 3, bit 4, bit 5, bit 6, bit 7
Shift register	********, 0*******, 10******, 210*****, 3210****, 43210***, 543210**, 6543210*, 76543210

(b) Trailing-edge
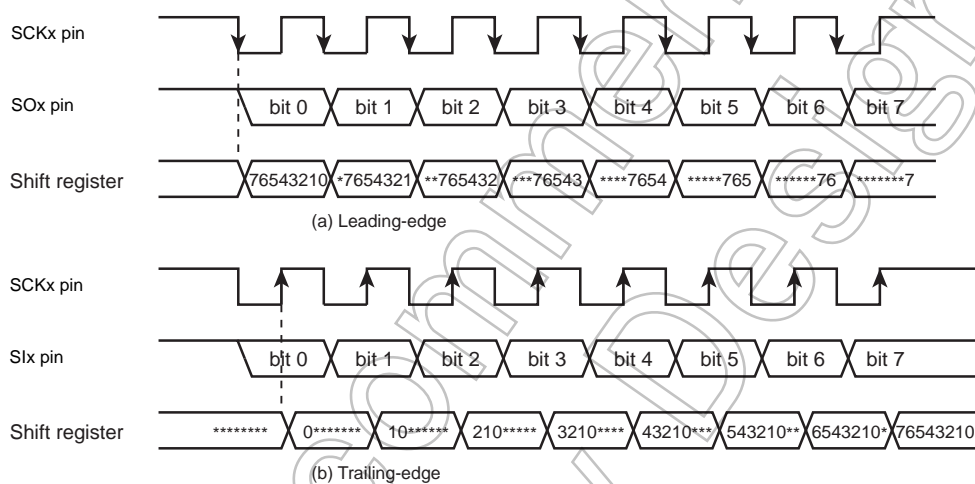
Figure 14-17 Shift Edge

14.8.2 Transfer Modes

The transmit mode, the receive mode or the transmit/receive mode can be selected by programming SBIxCR1<SIOM[1:0]>.

14.8.2.1 8-bit transmit mode

Set the control register to the transmit mode and write the transmit data to SBIxDBR.

After writing the transmit data, writing "1" to SBIxCR1<SIOS> starts the transmission. The transmit data is moved from SBIxDBR to a shift register and output to the SO pin, with the least-significant bit (LSB) first, in synchronization with the serial clock. Once the transmit data is transferred to the shift register, SBIxDBR becomes empty, and the INTSBIx (buffer-empty) interrupt is generated, requesting the next transmit data.

In the internal clock mode, the serial clock will be stopped and automatically enter the wait state, if next data is not loaded after the 8-bit data has been fully transmitted. The wait state will be cleared when SBIxDBR is loaded with the next transmit data.

In the external clock mode, SBIxDBR must be loaded with data before the next data shift operation is started. Therefore, the data transfer rate varies depending on the maximum latency between when the interrupt request is generated and when SBIxDBR is loaded with data in the interrupt service program.

At the beginning of transmission, the same value as in the last bit of the previously transmitted data is output in a period from setting SBIxSR<SIOF> to "1" to the falling edge of SCK.

Transmission can be terminated by clearing <SIOS> to "0" or setting <SIOINH> to "1" in the INTSBIx interrupt service program. If <SIOS> is cleared, remaining data is output before transmission ends. The program checks SBIxSR<SIOF> to determine whether transmission has come to an end. <SIOF> is cleared to "0" at the end of transmission. If <SIOINH> is set to "1", the transmission is aborted immediately and <SIOF> is cleared to "0".

When in the external clock mode, <SIOS> must be cleared to "0" before next data shifting. If <SIOS> does not be cleared to "0" before next data shifting, SBI output dummy data and stopped.

```
                    7   6   5   4   3   2   1   0
SBIxCR1    ←    0   1   0   0   0   X   X   X      Selects the transmit mode.


SBIxDBR    ←    X   X   X   X   X   X   X   X      Writes the transmit data.
SBIxCR1    ←    1   0   0   0   0   X   X   X      Starts transmission.

INTSBIx interrupt

SBIxDBR    ←    X   X   X   X   X   X   X   X      Writes the transmit data.
```

<SIOS> is cleared.

<SIOS>

<SIOF>

<SEF>

SCKx pin(output)

SOx pin

INTSBIx interrupt request

SBIxDBR

Write the transmit data

(a) Internal clock

<SIOS> is cleared.

<SIOS>

<SIOF>

<SEF>

SCKx pin(input)

SOx pin

INTSBIx interrupt request

SBIxDBR

Write the transmit data

(b)External clock

Figure 14-18 Transmit Mode

Example: Example of programming (external clock) to terminate transmission by <SIO>

```
              7   6   5   4   3   2   1   0
if SBIxSR<SIOF> ≠ 0                              Recognizes the completion of the transmission.

    Then


   if SCK ≠ 1                                    Recognizes "1" is set to the SCK pin by monitoring the
                                                 port.
    Then

 SBIxCR1    ←   0   0   0   0   0   1   1   1    Completes the transmission by setting <SIOS> = 0.
```
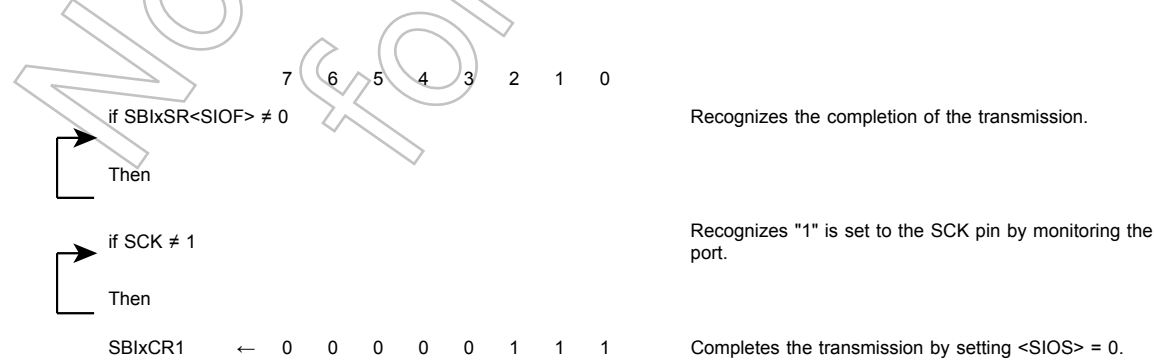
14.8.2.2   8-bit receive mode

Set the control register to the receive mode. Then writing "1" to SBIxCR1<SIOS> enables reception.Data is taken into the shift register from the SI pin, with the least-significant bit (LSB) first, in synchronization with the serial clock. Once the shift register is loaded with the 8-bit data, it transfers the received data to SBIxDBR and the INTSBIx (buffer-full) interrupt request is generated to request reading the received data. The interrupt service program then reads the received data from SBIxDBR.

In the internal clock mode, the serial clock will be stopped and automatically be in the wait state until the received data is read from SBIxDBR.

In the external clock mode, shift operations are executed in synchronization with the external clock. The maximum data transfer rate varies, depending on the maximum latency between generating the interrupt request and reading the received data

Reception can be terminated by clearing <SIOS> to "0" or setting <SIOINH> to "1" in the INTSBIx interrupt service program. If <SIOS> is cleared, reception continues until all the bits of received data are written to SBIxDBR. The program checks SBIxSR<SIOF> to determine whether reception has come to an end.<SIOF> is cleared to "0" at the end of reception. After confirming the completion of the reception, last received data is read. If <SIOINH> is set to "1", the reception is aborted immediately and <SIOF> is cleared to "0". (The received data becomes invalid, and there is no need to read it out.)

Note:**The contents of SBIxDBR will not be retained after the transfer mode is changed. The ongoing reception must be completed by clearing <SIOS> to "0" and the last received data must be read before the transfer mode is changed.**

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	1	1	0	X	X	X	Selects the receive mode.
SBIxCR1	←	1	0	1	1	0	X	X	X	Starts reception.

INTSBIx interrupt

Reg.	←	SBIxDBR		Reads the received data.
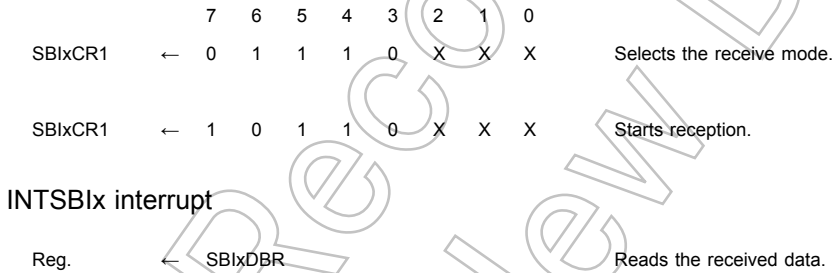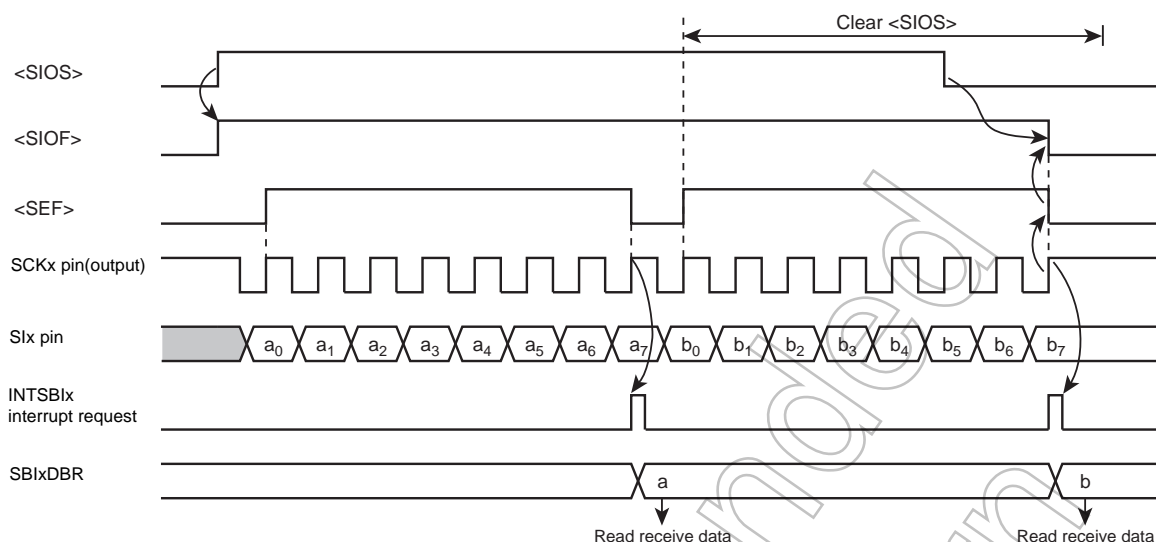
Figure 14-19 Receive Mode (Example: Internal Clock)

14.8.2.3   8-bit transmit/receive mode

Set the control register to the transfer/receive mode. Then writing the transmit data to SBIxDBR and setting SBIxCR1<SIOS> to "1" enables transmission and reception.The transmit data is output through the SOx pin at the falling of the serial clock, and the received data is taken in through the SI pin at the rising of the serial clock, with the least-significant bit (LSB) first. Once the shift register is loaded with the 8-bit data, it transfers the received data to SBIxDBR and the INTSBIx interrupt request is generated.The interrupt service program reads the received data from the data buffer register and writes the next transmit data. Because SBIxDBR is shared between transmit and receive operations, the received data must be read before the next transmit data is written.

In the internal clock operation, the serial clock will be automatically in the wait state until the received data is read and the next transmit data is written.

In the external clock mode, shift operations are executed in synchronization with the external serial clock. Therefore, the received data must be read and the next transmit data must be written before the next shift operation is started.The maximum data transfer rate for the external clock operation varies depending on the maximum latency between when the interrupt request is generated and when the transmit data is written.

At the beginning of transmission, the same value as in the last bit of the previously transmitted data is output in a period from setting <SIOF> to "1" to the falling edge of SCK.

Transmission and reception can be terminated by clearing <SIOS> to "0" or setting SBIxCR1<SIOINH> to "1" in the INTSBIx interrupt service program. If <SIOS> is cleared, transmission and reception continue until the received data is fully transferred to SBIxDBR. The program checks SBIxSR<SIOF> to determine whether transmission and reception have come to an end. <SIOF> is cleared to "0" at the end of transmission and reception.If <SIOINH> is set to "1", the transmission and reception is aborted immediately and <SIOF> is cleared to "0".

Note:**The contents of SBIxDBR will not be retained after the transfer mode is changed. The ongoing transmission and reception must be completed by clearing <SIOS> to "0" and the last received data must be read before the transfer mode is changed.**
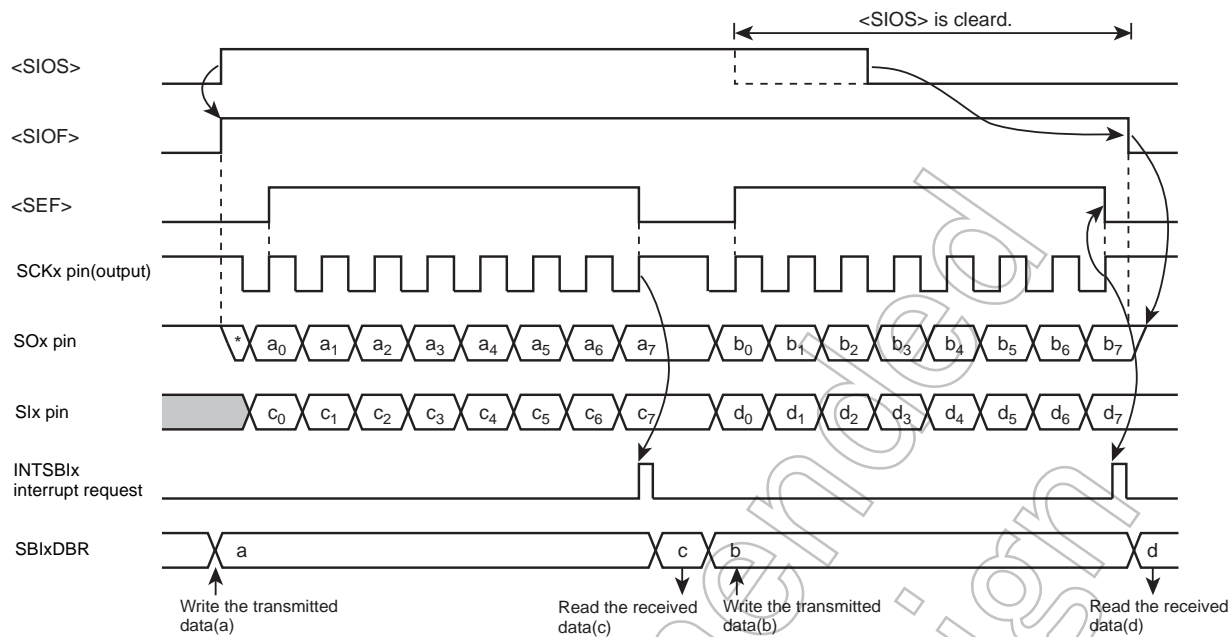
Figure 14-20 Transmit/Receive Mode (Example: Internal Clock)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	1	0	0	X	X	X	Selects the transmit mode.
SBIxDBR	←	X	X	X	X	X	X	X	X	Writes the transmit data.
SBIxCR1	←	1	0	1	0	0	X	X	X	Starts reception/transmission.

INTSBIx interrupt

Reg.	←	SBIxDBR	Reads the received data.
SBIxDBR	←	X X X X X X X X	Writes the transmit data.

14.8.2.4 Data retention time of the last bit at the end of transmission

Under the condition SBIxCR1<SIOS>= "0", the last bit of the transmitted data retains the data of SCK rising edge as shown below. Transmit mode and transmit/receive mode are the same.
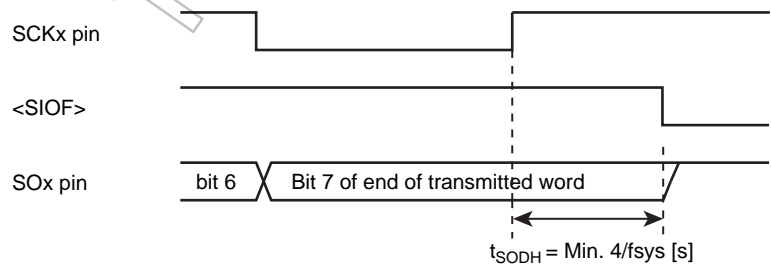
Figure 14-21 Data retention time of the last bit at the end of transmission

15. Consumer Electronics Control (CEC)

15.1 Outline

The CEC function transmits and receives data that conforms to Consumer Electronics Control (hereafter referred to as CEC) protocol.

It can operate conformably to HDMI 1.3a specifications.

15.1.1 Reception

- Clock sampling at fs clock or TBAOUT which is output of 16bit Timer/Event counters
    - Adjustable noise canceling time
- Data reception per 1byte
    - Flexible data sampling point
    - Data reception is available even when an address discrepancy is detected.
- Error detection
    - Cycle error (min./max.)
    - ACK collision
    - Waveform error

15.1.2 Transmission

- Data transmission per 1byte
    - Triggered by auto-detection of bus free state
- Flexible waveform
    - Adjustable rising edge and cycle
- Error detection
    - Arbitration lost
    - ACK response error

15.1.3 Precautions

When data reception at logical address discrepancy is enabled(CECRCR1<CECOTH> = "1"), if the initiator sends a new message beginning with the start bit without having sent the last block with EOM="1", a maximum cycle error is determined for the ACK bit and an interrupt is generated. Then, the receive operation is performed in the usual way.

15.2 Block Diagram

Figure 15-1 shows the Block Diagram of CEC
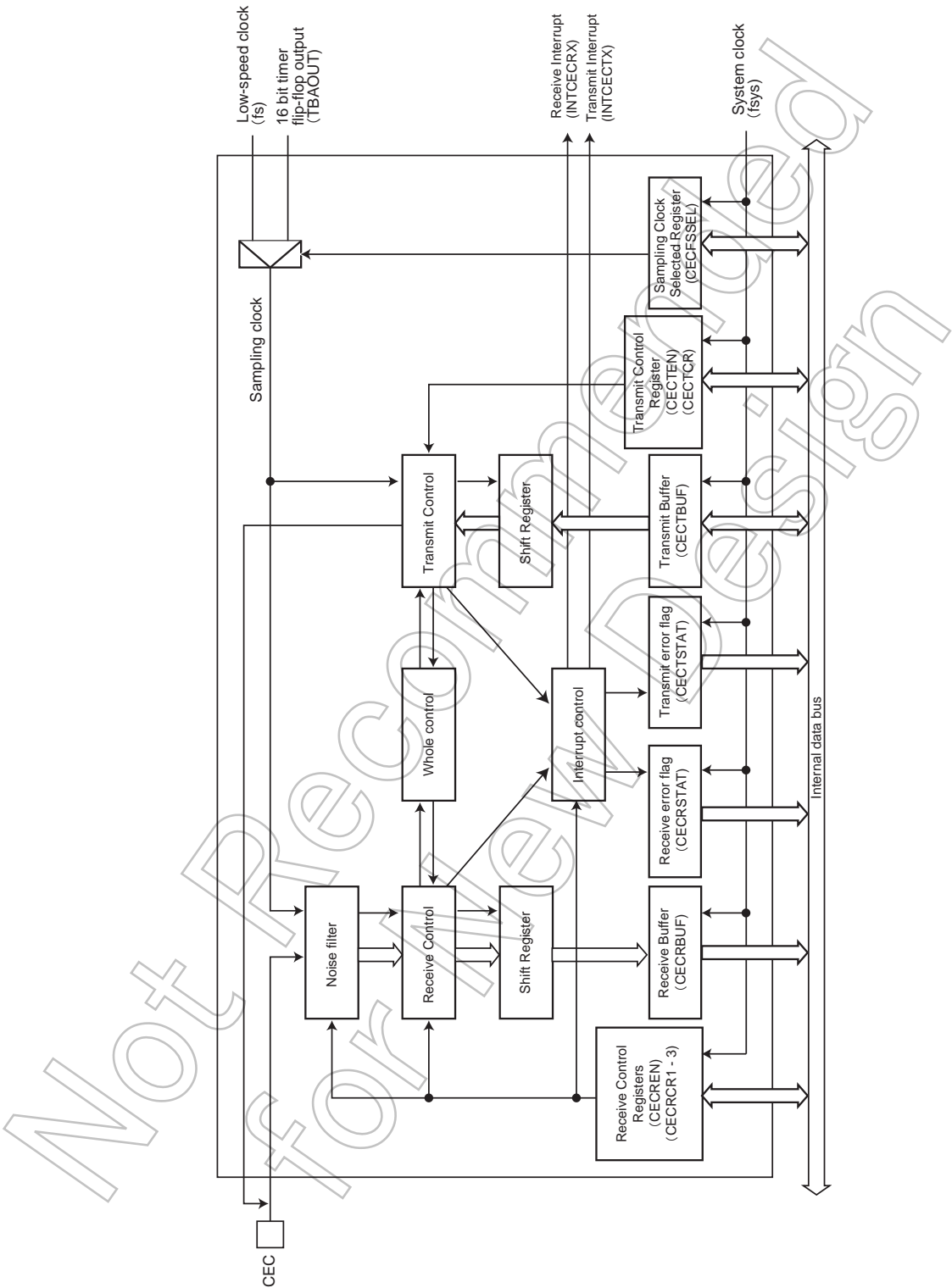
Figure 15-1 Block Diagram of CEC

15.3   Registers

15.3.1   Register List

The control registers and address for CEC are as follows.

Base Address = 0x400E_2000

Registers		Address (Base+)
CEC Enable Register	CECEN	0x0000
Logical Address Register	CECADD	0x0004
Software Reset Register	CECRESET	0x0008
Receive Enable Register	CECREN	0x000C
Receive Buffer Register	CECRBUF	0x0010
Receive Control Register 1	CECRCR1	0x0014
Receive Control Register 2	CECRCR2	0x0018
Receive Control Register 3	CECRCR3	0x001C
Transmit Enable Register	CECTEN	0x0020
Transmit Buffer Register	CECTBUF	0x0024
Transmit Control Register	CECTCR	0x0028
Receive Interrupt Status Register	CECRSTAT	0x002C
Transmit Interrupt Status Register	CECTSTAT	0x0030
CEC Sampling Clock Selected Register	CECFSSEL	0x0034

15.3.2   CECEN (CEC Enable Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	CECEN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as 0.
2	−	R/W	Write "0".
1	−	R/W	Write as "1".
0	CECEN	R/W	CEC operation 0 : Disabled 1 : Enabled Specifies the CEC operation. Enable CEC before using. When the CEC operation is disabled, no clocks are supplied to the CEC module except for the CECEN register. Thus power consumption can be reduced. When CEC is disabled after it was enabled, each register setting is maintained.

15.3.3 CECADD (Logical Address Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	CECADD[15:8]							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CECADD[7:0]							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-0	CECADD[15:0]	R/W	Logical address 15 to 0 Specifies the logical address assigned to CEC. Multiple addresses can be set simultaneously since each bit corresponds with each address.

Note: A broadcast message is received regardless of the register setting. By allocating a logical address of a device to 15, logical "0" is sent as an ACK response to the broadcast message.

15.3.4　CECRESET (Software Reset Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	CECRESET
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	FUnction
31-1	−	R	Read as 0
0	CECRESET	W	Software reset 0: Disabled 1: Enabled Stops all the CEC operation and initializes the register. Setting this bit to "1" affects as follows: Reception: Stops immediately. The received data is discarded. Transmission (including the CEC line): Stops immediately. Register: All the registers other than CECEN are initialized. Read as 0.

15.3.5   CECREN (Receive Enable Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	CECREN
After reset	0	0	0	0	0	0	0	Undefined

Bit	Bit Symbol	Type	Function
31-1	–	R	Read as 0
0	CECREN	R/W	Reception control [Write] 0 : Disabled 1 : Enabled [Read] 0 : Stopped 1 : In operation Controls the reception operation of CEC. Writing "0" or "1" to this bit enables or disables data reception. This bit becomes ready for data reception by writing "1". The state of the reception circuit is monitored by reading this bit. It enables you to check if what you set has properly been reflected.

Note 1: Enable the <CECREN> bit after setting the CECRCR1, CECRCR2 and CECRCR3.

Note 2: It takes a little time to reflect the setting of the <CECREN> bit to the circuit. Make sure that the register is under suspension when you try to change settings or to enable disableed-settings.

15.3.6   CECRBUF (Receive Buffer Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	CECACK	CECEOM
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CECRBUF							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-10	−	R	Read as 0.
9	CECACK	R	ACK bit Reads the received ACK bit.
8	CECEOM	R	EOM bit Reads the received EOM bit.
7-0	CECRBUF[7:0]	R	Received data Reads one byte of data received. The bit 7 is the MSB.

Note 1: Writing to this register is ignored.

Note 2: Read this register as soon as a receive interrupt is generated. The subsequent reading data may not be ensured.

15.3.7 CECRCR1 (Receive Control Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	CECACKDIS
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	CECHNC		-	CECLNC		
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	CECMIN			-	CECMAX		
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	CECDAT			CECTOUT		CECRIHLD	CECOTH
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-25	−	R	Read as 0.
24	CECACKDIS	R/W	Logical "0" as ACK response 0: send 1: not send Specifies if logical "0" is sent or not as an ACK response to the data block when destination address corresponds with the address set in the logical address register. (Logical "0" is sent to the header block as an ACK response regardless of the bit setting when detecting the addresses corresponding)
23-22	−	R	Read as 0.
21-20	CECHNC[1:0]	R/W	The number of "High" samplings for noise cancellation. 　00:　None　(one time of fs clock observed.) 　01:　1/fs　(two consecutive fs clocks observed.) 　10:　2/fs　(three consecutive fs clocks observed.) 　11:　3/fs　(four consecutive fs clocks observed.) Specifies the time of the noise cancellation for each 1/fs when detecting "High". It is considered as noise if "High"s of the same number as the specified cycles are not sampled.
19	−	R	Read as 0.
18-16	CECLNC[2:0]	R/W	The number of "Low" samplings for noise cancellation. 　000:　None (one time of fs clock observed.)　　100:　− (Reserved) 　001:　1/fs (two consecutive fs clocks observed)　　101:　− (Reserved) 　010:　2/fs (three consecutive fs clocks observed)　　110:　− (Reserved) 　011:　3/fs (four consecutive fs clocks observed.)　　111:　− (Reserved) Specifies the time of the noise cancellation for each 1/fs when detecting "Low". It is considered as noise if "Low"s of the same number as the specified cycles are not sampled.
15	−	R	Read as 0.
14-12	CECMIN[2:0]	R/W	Time to identify as minimum cycle error 　000:　67/fs (approx.2.045ms)　　　　100:　67/fs − 1/fs 　001:　67/fs + 1/fs　　　　　　　　　101:　67/fs − 2/fs 　010:　67/fs + 2/fs　　　　　　　　　110:　67/fs − 3/fs 　011:　67/fs + 3/fs　　　　　　　　　111:　67/fs − 4/fs Specifies the minimum time to identify a valid bit. Base time is 67/fs (approx.2.045) ms. Enables to specify it between the ranges −4/fs to +3/fs by the unit of 1/fs. An interrupt is generated and "Low" is output to CEC for approx. 3.63 ms when one bit cycle is shorter than the specified time.
11	−	R	Read as 0.

Bit	Bit Symbol	Type	Function
10-8	CECMAX[2:0]	R/W	Time to identify as maximum cycle error 000: 90/fs (approx. 2.747ms) 100: 90/fs − 1/fs 001: 90/fs + 1/fs 101: 90/fs − 2/fs 010: 90/fs + 2/fs 110: 90/fs − 3/fs 011: 90/fs + 3/fs 111: 90/fs − 4/fs Specifies the maximum time to identify as a valid bit. Base time is 90/fs (approx.2.747 ms). Enables to specify it between the ranges −4/fs to +3/fs by the unit of 1/fs. An interrupt is generated when one bit cycle is longer than the specified time.
7	−	R	Read as 0.
6-4	CECDAT[2:0]	R/W	Point of determining the data as 0 or 1. 000: 34/fs (approx. 1.038ms) 100: 34/fs − 2/fs 001: 34/fs + 2/fs 101: 34/fs − 4/fs 010: 34/fs + 4/fs 110: 34/fs − 6/fs 011: 34/fs + 6/fs 111: Reserved Specifies the point of determining the data as logical "0" or logical "1". Base time is 34/fs (approx.1.038 ms). Enables to specify it within ±6/fs by the unit of 2/fs.
3-2	CECTOUT[1:0]	R/W	Cycle to identify timeout 00: 1 bit cycle 01: 2 bit cycle 10: 3 bit cycle 11: Reserved Specifies the time to determine a timeout. Enables to specify it between 1 bit and 3 bits for each bit cycle. This setting is used to detect a timeout when the <CECRIHLD> bit is valid.
1	CECRIHLD	R/W	Error interrupt suspend 0: Not suspended 1: Suspended Specifies whether to suspend a receive error interrupt (maximum cycle error, buffer overrun and waveform error). Setting "1" generates no interrupt at the error detection. If data continues to an ACK bit, an ACK response is executed by a reversed logic. If the subsequent bits are interrupted, it is determined as a timeout, based on the setting in <CECTOUT>. After the ACK response or the timeout determination, an interrupt is generated.
0	CECOTH	R/W	Data reception at logical address discrepancy 0: Not received 1: Received Specifies whether to receive data when the destination address does not correspond with the address set in the CECADD register.

Note 1:   The settings in <CECHNC>, <CECLNC> and <CECDAT> are also used in receiving an ACK response at transmission.

Note 2:   Changing the configurations during transmission or reception may harm its proper operation. Before the change, set the CECREN <CECREN> bit to disable the reception and read the <CECREN> bit and the CECTEN <CECTRANS> bit to ensure that the operation is stopped.

Note 3:   A broadcast message is received regardless of the <CECOTH> register setting.

Note 4:   <CECLNC> must be used under the same setting as CECTCR<CECDTRS>.

15.3.8   CECRCR2 (Receive Control Register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	CECSWAV3			-	CECSWAV2		
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	CECSWAV1			-	CECSWAV0		
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-15	–	R	Read as 0.
14-12	CECSWAV3[2:0]	R/W	Max. cycle to detect start bit. 000: 154/fs (approx. 4.700 ms) 100: 154/fs + 4/fs 001: 154/fs + 1/fs 101: 154/fs + 5/fs 001: 154/fs + 2/fs 110: 154/fs + 6/fs 011: 154/fs + 3/fs 111: 154/fs + 7/fs
11	–	R	Read as 0.
10-8	CECSWAV2[2:0]	R/W	Min. cycle to detect start bit. 000: 141/fs (approx. 4.303 ms) 100: 141/fs − 4/fs 001: 141/fs − 1/fs 101: 141/fs − 5/fs 001: 141/fs − 2/fs 110: 141/fs − 6/fs 011: 141/fs − 3/fs 111: 141/fs − 7/fs
7	–	R	Read as 0.
6-4	CECSWAV1[2:0]	R/W	Max. time of start bit rising timing. 000: 128/fs (approx. 3.906 ms) 100: 128/fs + 4/fs 001: 128/fs + 1/fs 101: 128/fs + 5/fs 001: 128/fs + 2/fs 110: 128/fs + 6/fs 011: 128/fs + 3/fs 111: 128/fs + 7/fs
3	–	R	Read as 0.
2-0	CECSWAV0[2:0]	R/W	Min. time of start bit rising timing. 000: 115/fs (approx. 3.510 ms) 100: 115/fs − 4/fs 001: 115/fs − 1/fs 101: 115/fs − 5/fs 001: 115/fs − 2/fs 110: 115/fs − 6/fs 011: 115/fs − 3/fs 111: 115/fs − 7/fs

<CECSWAV3>:   Specifies the cycles to detect a start bit.

<CECSWAV2>:   <CECSWAV3> is for the maximum cycles. The base time is 154/fs (approx.4.700 ms). Enables to specify it between the ranges 0 to +7/fs by the unit of 1/fs.

<CECSWAV2> is for the minimum cycles. The base time is 141/fs (approx.4.303 ms). Enables to specify it between the ranges 0 to -7/fs by the unit of 1/fs.

<CECSWAV1>:   Specifies the rising timing of a start bit in its detection.

<CECSWAV0>:   <CECSWAV1> is for the maximum time of the rising timing. The base time is 128/fs (approx.3.906 ms). Enables to specify it between the ranges 0 to +7/fs by the unit of 1/fs.

<CECSWAV0> is for the minimum time of the rising timing. The base time is 115/fs (approx.3.510 ms). Enables to specify it between the ranges 0 to -7/fs by the unit of 1/fs.

Note: Changing the configurations during reception may harm its proper operation. Before the change, set CECREN <CECREN> to disable the reception and read the <CECREN> bit to ensure that the operation is stopped.

15.3.9 CECRCR3 (Receive Control Register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-		CECWAV3		-		CECWAV2	
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-		CECWAV1		-		CECWAV0	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	CECWAVEN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-23	−	R	Read as 0.
22-20	CECWAV3[2:0]	R/W	The latest rising timing of logical "0" determined as proper waveform. 000: 56/fs (approx. 1.709 ms)　　　100: 56/fs + 4/fs 001: 56/fs + 1/fs　　　101: 56/fs + 5/fs 001: 56/fs + 2/fs　　　110: 56/fs + 6/fs 011: 56/fs + 3/fs　　　111: 56/fs + 7/fs
19	−	R	Read as 0.
18-16	CECWAV2[2:0]	R/W	The fastest rising timing of logical "0" determined as proper. 000: 43/fs (approx.1.312 ms)　　　100: 43/fs − 4/fs 001: 43/fs − 1/fs　　　101: 43/fs − 5/fs 010: 43/fs − 2/fs　　　110: 43/fs − 6/fs 011: 43/fs − 3/fs　　　111: 43/fs − 7/fs
15	−	R	Read as 0.
14-12	CECWAV1[2:0]	R/W	The latest rising timing of logical "1" determined as proper waveform. 000: 26/fs (approx. 0.793 ms)　　　100: 26/fs + 4/fs 001: 26/fs + 1/fs　　　101: 26/fs + 5/fs 001: 26/fs + 2/fs　　　110: 26/fs + 6/fs 011: 26/fs + 3/fs　　　111: 26/fs + 7/fs
11	−	R	Read as 0
10-8	CECWAV0[2:0]	R/W	The fastest rising timing of logical "1" determined as proper. 000: 13/fs (approx. 0.396 ms)　　　100: 13/fs − 4/fs 001: 13/fs − 1/fs　　　101: 13/fs − 5/fs 010: 13/fs − 2/fs　　　110: 13/fs − 6/fs 011: 13/fs − 3/fs　　　111: 13/fs − 7/fs
7-2	−	R	Read as 0.
1	CECRSTAEN	R/W	Start bit detection 1: Enable 0: Disable Detects a reception of start bit and generates interrupt.
0	CECWAVEN	R/W	Waveform error detection 1: Enable 0: Disable Detects a received waveform does not identical to the one defined and generates waveform error interrupt. If enabled, an error is detected according to the setting of <CECWAV0> <CECWAV1> <CECWAV2> <CEC WAV3>.

<CECWAV3>: This setting is enabled when the <CECWAVEN> bit is set to "1".

By setting these bits, an error is detected if rising edge of the received waveform comes later than that of proper logical "0".

Base time is 56/fs (approx. 1.709ms). Enables to specify it between the ranges 0 to +7/fs by the unit of 1/fs.

The received waveform is considered to be an error if a rising edge is not detected from the start point of the bit to the value specified in <CECWAV3>.

<CECWAV2>: This setting is enabled when the <CECWAVEN> bit is set to "1".

<CECWAV1>: By setting these bits, an error is detected if rising edge of the received waveform comes faster than logical "0" and later than that of proper logical "1".

Base time for <CECWAV1> bit is 26/fs (approx. 0.793ms). Enables to specify it between the ranges 0 to +7/fs by the unit of 1/fs.

Base time for <CECWAV2> bit is 43/fs (approx.1.312ms). Enables to specify it between the ranges 0 to −7/fs by the unit of 1/fs.

If a rising edge is detected during <CECWAV2> bit and <CECWAV1> bit setting, an error occurs.

<CECWAV0>: This setting is enabled when the <CECWAVEN> bit is set to "1".

By setting these bits, an error is detected if rising edge of the received waveform comes faster than that of proper logical "1".

Base time is 13/fs (approx. 0.396ms). Enables to specify it between the ranges 0 to −7/fs by the unit of 1/fs.

The received waveform is considered to be an error if a rising edge is not detected from a start point of the bit to the value specified in <CECWAV0>.

Note: Changing the configurations during reception may harm its proper operation. Before the change, set CECREN <CECREN> to disable the reception and read the <CECREN> bit to ensure that the operation is stopped.

15.3.10 CECTEN (Transmit Enable Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	CECTRANS	CECTEN
After reset	0	0	0	0	0	0	0	Undefined

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as 0.
1	CECTRANS	R	Transmission state 0: not in progress 1:in progress Indicates whether the transmission is in progress or not. It indicates "1" upon starting the transmission of the start bit. It indicates "0" if transmission is completed or an interrupt is generated. Writing to this bit is ignored.
0	CECTEN	W	Transmission control 0: Disable 1: Enable Controls the CEC transmission. Writing this bit enables or disables the transmission. Writing "1" to this bit initiates the transmission. This bit is automatically cleared by a transmit completion interrupt or an error interrupt.

Note 1: Set <CECTEN> after setting the CECTBUF and CECTCR register.

Note 2: Stop transmission and reception before changing the settings or enabling the transmission and reception.

15.3.11 CECTBUF (Transmit Buffer Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	CECTEOM
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CECTBUF							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-9	−	R	Read as 0.
8	CECTEOM	R/W	EOM bit Specifies the EOM bit to transmit.
7-0	CECTBUF[7:0]	R/W	Transmitted data Specifies a byte of data to transmit. The bit 7 is the MSB.

15.3.12   CECTCR (Transmit Control Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	CECSTRS			-	CECSPRD		
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	CECDTRS			CECDPRD			
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	CECBRD	CECFREE			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-23	–	R	Read as 0.
22-20	CECSTRS[2:0]	R/W	Rising timing of start bit. 000: Base time 100: Base time− 4/fs 001: Base time− 1/fs 101: Base time− 5/fs 010: Base time− 2/fs 110: Base time− 6/fs 011: Base time− 3/fs 111: Base time− 7/fs Specifies the rising timing of a start bit. Base time is 121/fs (approx. 3.693 ms). Enables to specify it between the ranges 0 to −7/fs by the unit of 1/fs.
19	–	R	Read as 0.
18-16	CECSPRD[2:0]	R/W	Start bit cycle 000: Base time 100: Base time− 4/fs 001: Base time− 1/fs 101: Base time− 5/fs 010: Base time− 2/fs 110: Base time− 6/fs 011: Base time− 3/fs 111: Base time− 7/fs Specifies a cycle of a start bit. Base time is 147/fs (approx. 4.486 ms). Enables to specify it between the ranges 0 to −7/fs by the unit of 1/fs.
15	–	R	Read as 0.
14-12	CECDTRS[2:0]	R/W	Rising timing of data bit. 000: Base time 100: Reserved 001: Base time− 1/fs 101: Reserved 010: Base time− 2/fs 110: Reserved 011: Base time− 3/fs 111: Reserved Specifies the rising timing of a data bit Base time is 20/fs (approx. 0.610 ms, when logical "1") or 49/fs (approx. 1.495 ms, when logical "0"). Enables to specify it between the ranges 0 to −3/fs by the unit of 1/fs.
11-8	CECDPRD[2:0]	R/W	Data bit cycle 0000: Base time 1000: Base time− 8/fs 0001: Base time− 1/fs 1001: Base time− 9/fs 0010: Base time− 2/fs 1010: Base time− 10/fs 0011: Base time− 3/fs 1011: Base time− 11/fs 0100: Base time− 4/fs 1100: Base time− 12/fs 0101: Base time− 5/fs 1101: Base time− 13/fs 0110: Base time− 6/fs 1110: Base time− 14/fs 0111: Base time− 7/fs 1111: Base time− 15/fs Specifies a cycle of a data bit. Base time is 79/fs (approx. 2.411 ms). Enables to specify it between the ranges 0 to −15/fs by the unit of 1/fs.

Bit	Bit Symbol	Type	Function
7-5	–	R	Read as 0.
4	CECBRD	R/W	Broadcast transmission 0: Not broadcast transmission 1: Broadcast transmission Set this bit to "1" when transmitting a broadcast message.
3-0	CECFREE[3:0]	R/W	Time of bus to be free 0000: 1bit cycle 1000: 9bit cycle 0001: 2bit cycle 1001: 10bit cycle 0010: 3bit cycle 1010: 11bit cycle 0011: 4bit cycle 1011: 12bit cycle 0100: 5bit cycle 1100: 13bit cycle 0101: 6bit cycle 1101: 14bit cycle 0110: 7bit cycle 1110: 15bit cycle 0111: 8bit cycle 1111: 16bit cycle Specifies time of a bus to be free that checked before transmission. Start transmission after checking the CEC line kept inactive during the specified cycles.

Note:<CECDTRS> must be used under the same setting as CECRCR1<CECLNC>.

15.3.13 CECRSTAT (Receive Interrupt Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	CECRIWAV	CECRIOR	CECRIACK	CECRIMIN	CECRIMAX	CECRISTA	CECRIEND
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as 0.
6	CECRIWAV	R	Interrupt flag 0: No wave form error 1: Wave form error Indicates that waveform error is detected. The error occurs when waveform error detection is enabled in CECRCR3 <CECWAVEN>.
5	CECRIOR	R	Interrupt flag 0: No receive buffer overrun 1:Receive buffer overrun Indicates the receive buffer receives next data before reading the data that had already been set.
4	CECRIACK	R	Interrupt flag 0: No ACK collision 1: ACK collision Indicates "0" is detected after the specified time to output ACK bit "0".
3	CECRIMIN	R	Interrupt flag 0: No minimum cycle error 1:Minimum cycle error Indicates one bit cycle is shorter than the minimum cycle error detection time specified in CECRCR1<CECMIN>.
2	CECRIMAX	R	Interrupt flag 0: No maximum cycle error 1: Maximum cycle error Indicates one bit cycle is longer than the maximum cycle error detection time specified in CECRCR1<CECMAX>.
1	CECRISTA	R	Interrupt flag 0: No start bit detection 1: Start bit detection Indicates a start bit is detected.
0	CECRIEND	R	Interrupt flag 0: Not one byte data reception completed 1: Completion of 1 byte data reception Indicates 1 byte of data reception is completed.

Note: Writing to this bit is ignored.

15.3.14   CECTSTAT (Transmit Interrupt Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	CECTIUR	CECTIACK	CECTIAL	CECTIEND	CECTISTA
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as 0.
4	CECTIUR	R	Interrupt flag 0: No transmit buffer underrun 1: Transmit buffer underrun Indicates next data has not set to the transmission buffer within a byte of data transmission.
3	CECTIACK	R	Interrupt flag 0: No ACK error detection 1: ACK error detection Indicates one of the following conditions occurs. ・When logical "0" is not detected in transmission to the specific address. ・When logical "1" is not detected in transmission of a broadcast message.
2	CECTIAL	R	Interrupt flag 0: No arbitration lost 1: Arbitration lost occurs Indicates "Low" is detected while outputting "High".
1	CECTIEND	R	Interrupt flag 0: No data transmission completion 1: data transmission is completed Indicates data transmission including the EOM bit is completed.
0	CECTISTA	R	Interrupt flag 0: No start transmission 1: Start transmission Indicates 1 byte of data transmission is started.

Note: Writing to this bit is ignored.

15.3.15   CECFSSEL(CEC Sampling Clock Select Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	CECCLK
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	−	R	Read as 0.
0	CECCLK	R/W	CEC sampling clock 0: Low-speed clock (fs) 1: TBAOUT Sets the sampling clock for CEC function.Enables to select either low-speed clock (fs) or timer output as of CEC sampling clock.Timer output range is 30kHz to 34kHz by setting TBAOUT.

Note: When changing sampling clock by CECFSSEL register, stop (prohibit) CEC operation by CECEN<CE-CEN> register once. Then set CECFSSEL register first prior to other CEC related registers after starting (permitting) the CEC operation again. And also in the case of software reset by CECRESET register, set CECFSSEL register first prior to other CEC related registers when changing sampling clock.

15.4 Operations

15.4.1 Sampling clock

CEC lines are sampled by a 32.768kHz of low speed clock (fs) or TBAOUT which is output of 16bit Timer/ Event counters.

The sampling clock is configurable with the <CECCLKC> bits of the CECFSSEL register.

15.4.2 Reception

15.4.2.1 Basic Operation

If a start bit is detected, a start bit interruption generates. By generating start bit interruption, CECRSTAT<CECRISTA> is set. The start bit interrupt is generated when the CECRCR3<CECRSTA> is set to "1".

If one byte data, EOM bit and ACK bit are received, the received data is stored in CECRBUF register, and a received interruption generates. By generating the received interruption, CECRSTAT<CECRIEND> is set.

In the CECRBUF register, 8 bit data, EOM bit and ACK bit are stored. The ACK bit is not generated in the CEC circuit internally. This bit is generated from a observation of CEC signal same as other data.

After one data block is received, receiving operation continues until detecting the last block of data with EOM bit set to"1". Detecting the end of last block, CEC becomes the start bit waiting mode.

Detecting an error during data reception causes an error interrupt, and CEC waits for the next start bit. The received data is discarded.

Note: Regarding data reception, please carefully read "15.1.3 Precautions".
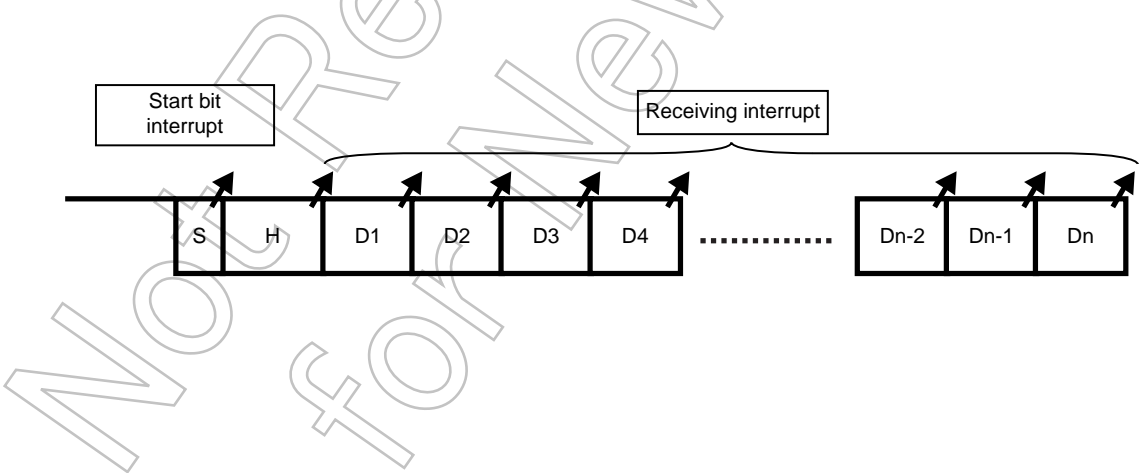
15.4.2.2   Preconfiguration

Before receiving data, reception settings to the Logical Address Register <CECADD>, the Receive Control Register 1 <CECRCR1>, the Receive Control Register 2 <CECRCR2> and the Receive Control Register 3 <CECRCR3> are required.

(1)   Logical Address Configuration

Configure logical address assigned to this product to the CECADD register. Multiple addresses can be set simultaneously since every bit in this register corresponds with each address.

> Note: A broadcast message is received regardless of the CECADD register setting. By allocating a logical address of a device to 15, logical "0" is sent as an ACK response to the broadcast message.

(2)   Noise Cancellation Time

The noise cancellation time is configurable with the <CECHNC> and <CECLNC> bits of the CECRCR1 register. It is considered as noise if "High"or "Low"of the same number as the specified value are not sampled.You can configure the time to detect "High" and "Low" respectively.

A CEC line is monitored at each rising edge of a sampling clock. In the case that the CEC line is changed from "High" to "Low", the change is fully recognized if "Low"s of the same number as specified in the <CECLNC> bit are monitored. In the case that the CEC line is changed from "Low" to "High", the change is fully recognized if "High" of the same number as specified in the <CECHNC> bit are sampled.

> Note: Use <CECLNC> in the same settings used for CECTCR<CECDTRS>.

The following illustrates the operation of a case that a noise cancelling is configured as <CECHNC [1:0]> = "10" (3 samplings) and <CECLNC[2:0]> = "011" (4 samplings). By cancelling the noise, a signal "1" shifts to "0" after "0" is sampled four times. The signal "0" shifts to "1" after "1" is sampled three times.

<CECHNC[1:0]> = 10   (3 samplings)
<CECLNC[2:0]> = 011 (4 samplings)
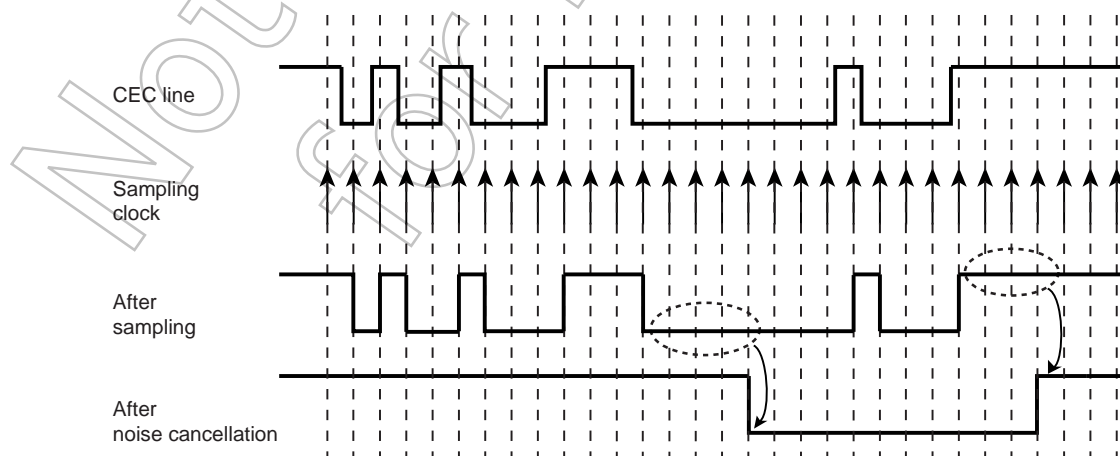
(3) Cycle error

Configure CECRCR1<CECMIN> and <CECMAX> bits to detect a cycle error.

A cycle error can be detected from each sampling clock cycle between the ranges −4/fs to +3/fs by the unit of 1/fs from the minimum value (67/fs, approx. 2.045ms) or the maximum value (90/fs approx. 2.747ms).

Detecting an error during data reception causes an error interrupt, and CEC waits for the next start bit. The received data is discarded.

(4) Point of Determining Data

Configure the CECRCR1 <CECDAT> bit for the point of determining the data as "0" or "1".

Base time is 34/fs (approx.1.038ms) from the start point and also configurable ±6/fs by the unit of 2/fs.

Data sampring timing that specification recommends
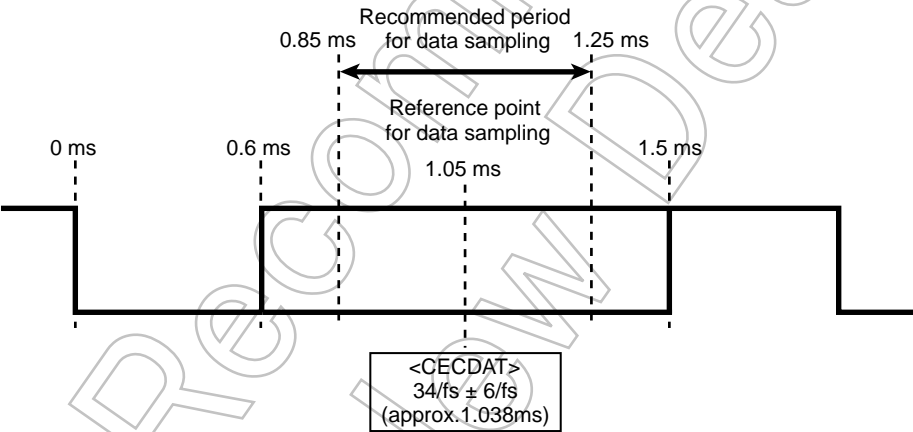
(5) ACK Response

Configuring the CECRCR1 <CECACKDIS> bit enables you to specify if logical "0" is sent or not as an ACK response to the data block when destination address corresponds with the address set in the logical address register.

Logical "0" issent to the header block as an ACK response regardless of the bit setting of <CECACKDIS>.

The following lists the ACK responses.

"Yes" indicates that CEC outputs "0" as a response to the ACK signal from a transmission device (ACK bit: logical "0"). "No" indicates that CEC does not output "0" as a response to the ACK signal from a transmission device (ACK bit: logical "1").

Register setting		Header block address		Data block address	
		Conformity	Discrepancy	Conformity	Discrepancy
CECRCR1 <CECACKDIS>	"0" (responding logical "0")	Yes	No	Yes	No
	"1" (not responding logical "0")			No	No

The following describes the ACK response timing.

When the falling edge of the ACK bit from the initiator is detected, this IP outputs "Low" for approximately 1.526 ms. The start time of outputting "Low" is specified with CECRCR1<CECLNC> bit that sets the noise cancelling time.

Note: Use <CECLNC> in the same settings used for CECTCR<CECDTRS>.
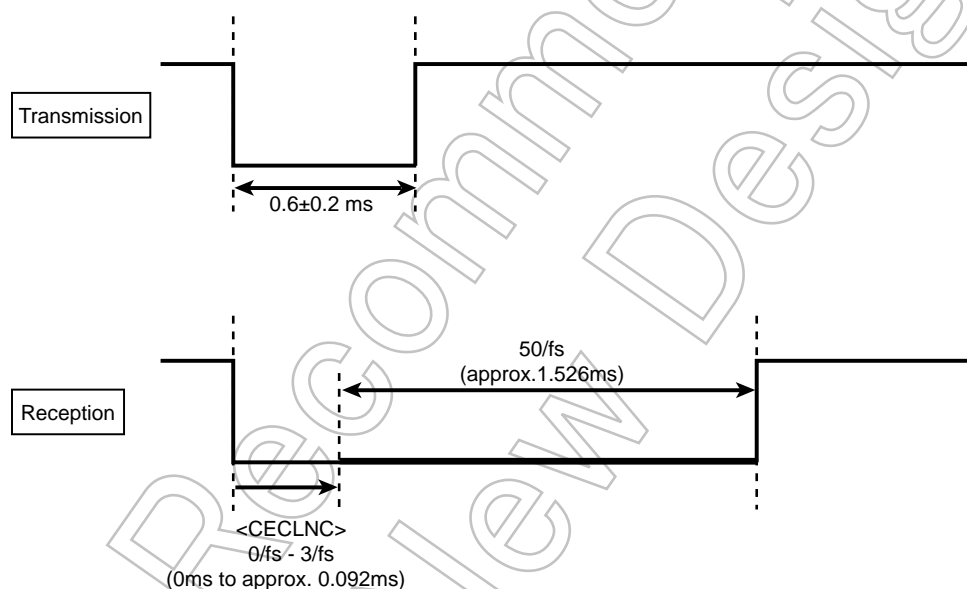
(6)   Receive Error Interrupt Suspend

Configure the CECRCR1 <CECRIHLD> bit to specify if a receive error interrupt (maximum cycle error, buffer overrun and waveform error) is suspended or not. Setting "1" generates no interrupt at the error detection.

If data continues to the ACK bit, an ACK response is executed by a reversed logic. If the subsequent bits are interrupted, it is determined as a timeout, based on the setting in <CECTOUT> of the CECRCR1 register.

After the ACK response or the timeout determination, an interrupt is generated.

(7)   Cycles to Identify Timeout

Configure the CECRCR1<CECTOUT> bit to specify the time to determine a timeout.

This is used when the setting of a receive error interrupt suspension, which is specified in CECRCR1 <CECRIHLD>, is valid.

(8)   Data Reception at Logical Address Discrepancy

By setting CECRCR1 <CECOTH>, you can specify if data is received or not when destination address does not correspond with the address set in the CECADD register.

In this case, an ordinary data reception is performed and an interrupt is generated by detecting an error. An ACK response is , however, not performed, neither the header block nor the data block.

Note 1:   A broadcast message is received regardless of the <CECOTH> register setting.

Note 2:   If the initiator sends a new message beginning with the start bit without having sent the last block with EOM="1", a maximum cycle error is determined for the ACK bit and an interrupt is generated. Then, the receive operation is performed in the usual way.

(9)   Start Bit Detection

Configuring the CECRCR2 register allows you to specify the rising timing and a cycle of the start bit detection respectively.

<CECSWAV0> is to specify the fastest start bit rising timing. <CECSWAV1> is to specify the latest start bit rising timing (the period that 1. indicates in the figure shown below).

<CECSWAV2> is to specify the minimum cycle of a start bit. <CECSWAV3> is to specify the maximum cycle of a start bit (the period that 2. indicates in the figure shown below).

If a rising edge during the period 1. and a falling edge during the period 2. are detected, the start bit is considered to be valid.

Permissible value of signal transition timing on specification (Start bit)
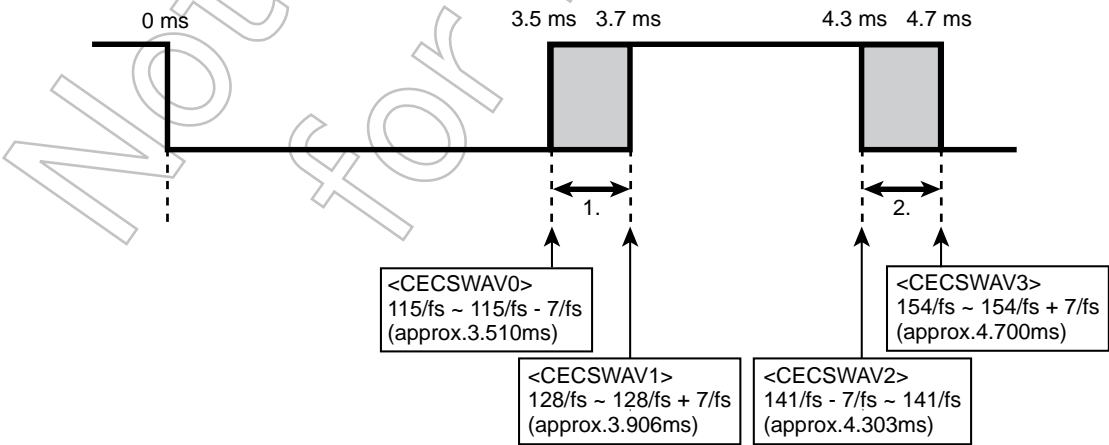
(10)   Waveform Error Detection

To detect an error when a received waveform is out of the defined tolerance range, configure the CECRCR3 register.

An error is detected when the <CECWAVEN> bit of the CECRCR3 register is enabled. You can specify the detection time in the <CECWAV0>, <CECWAV1>, <CECWAV2> and <CECWAV3> bits.

If the rising edge is detected during the period 1. or 2. shown below, or not detected in the timing described in 3., a waveform error interrupt is generated.

1.   A period between the beginning of a bit and the fastest logical "1" rising timing

2.   A period between the latest logical "1" rising timing and the fastest logical "0" rising timing.

3.   The latest logical "0" rising timing.

Permissible value of signal transition timing on specification (Data bit)
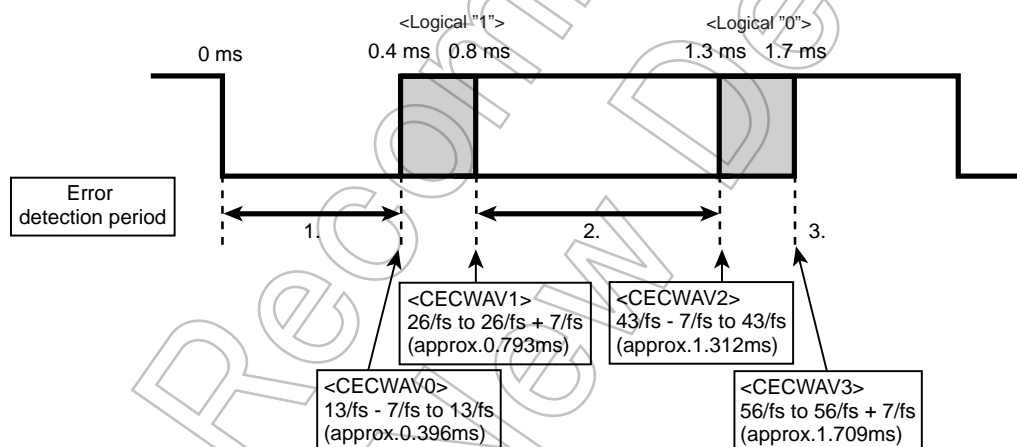
15.4.2.3 Enabling Reception

After configuring the CECADD, CECRCR1, CECRCR2 and CECRCR3 registers, CEC is ready for reception by enabling the CECREN <CECREN> bit. Detecting a start bit initiates the reception.

Note: Changing the configurations of the CECADD, CECRCR1, CECRCR2 and CECRCR3 registers during reception may harm its proper operation. Before the change of the registers shown below, set the CECREN <CECREN> bit to disable the reception and read the <CECREN> bit and the CECTEN <CECTRANS> bit to ensure that the operation is stopped.

Register name	Bit Symbol	Setting item
CECADD	<CECADD[15:0]>	Logical address
CECRCR1	<CECHNC><CECLNC>	Noise cancellation time
	<CECMIN><CECMAX>	Time to identify cycle error
	<CECOTH>	Data reception at logical address discrepancy
CECRCR2	<CECSWAV0><CECSWAV1> <CECSWAV2><CECSWAV3>	Start bit detection
CECRCR3	<CECWAV0><CECWAV1> <CECWAV2><CECWAV3>	Waveform error detection (when enabled)

15.4.2.4 Detecting Error Interrupt

Detecting an error during data reception causes an error interrupt, and CEC waits for the next start bit. The received data is discarded.

It is possible to suspend a receive error interrupt (maximum cycle error, receive buffer overrun and waveform error), continue reception and send the reversed ACK response.

You can check the interrupt factor by monitoring the bit of the CECRSTAT register corresponding to interrupts.

15.4.2.5 Details of reception error

(1) Cycle error

Period between the falling edges of the two sequential bits is measured during reception. If the period does not comply with the specified minimum or maximum value, a cycle error interrupt is generated.

A setting of maximum cycle and minimum cycle time is specified by CECRCR1<CECMIN> and <CECMAX> bits. Maximum value is 90/fs (approx.2.747ms) and minimum value is 67/fs (approx. 2.045ms). It can be specified between the ranges −4/fs to +3/fs by the unit of 1/fs to detect cycle errors.

The CECRSTAT <CECRIMIN> bit or the <CECRIMAX> bit is set if a cycle error interrupt is generated.

The minimum cycle error causes CEC to output "Low" for approx. 3.63 ms.

Note 1: When minimum cycle error is detected, "Low" is output after "Low" detecting noise cancellation time.

Note 2: If the initiator sends a new message beginning with the start bit without having sent the last block with EOM="1", a maximum cycle error is determined for the ACK bit and an interrupt is generated. For detaled information, refer to "15.1.3 Precautions".

(2)   ACK Collision

At an ACK response, detecting "Low" after the specified period to output generates an ACK collision interrupt or a minimum cycle error interrupt.

The ACK collision interrupt sets the CECRSTAT <CECRIACK> bit. The minimum cycle error interrupt sets the CECRSTAT <CECRIMIN> bit.

The following describes the period and method of detection.

Detection starts approx. 0.3 ms after the end of the period of outputting "Low" and ends approx 2.0 ms from the starting point (the falling edge) of the ACK bit.

At 0.3 ms from the end of the period of outputting "Low", CEC checks if the CEC line is "0" or not. If it is "Low", an ACK collision interrupt is generated. If it is "High", and "Low" is detected during the detection period, the minimum cycle error interrupt is generated. The minimum cycle error causes CEC to output "Low" for approx. 3.63ms.
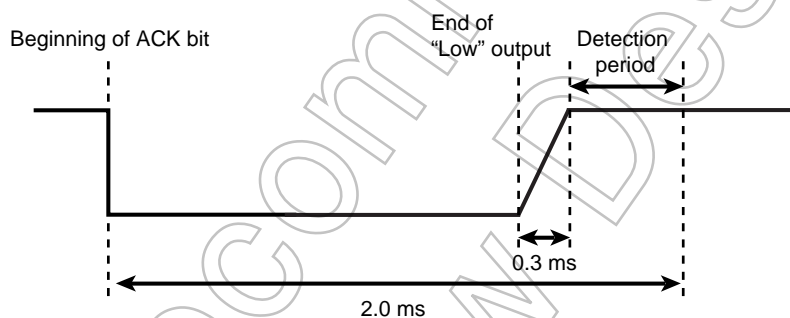
(3)   Receive Buffer Overrun

A receive buffer overrun interrupt is generated when the next data reception is completed before reading the data stored in the receive buffer.

The interrupt sets the CECRSTAT <CECRIOR> bit.

(4)   Waveform Error

A waveform error occurs when waveform error detection is enabled in CECRCR3.

Detecting a waveform, which does not identical to the defined, results in the waveform error. The interrupt is generated.

The interrupt sets the CECRSTAT <CECRIWAV> bit.

(5) Suspending Receive Error Interrupt

You can specify if a maximum cycle error, a buffer overrun and a waveform error to be suspended without generating an interrupt when an erro is detected. This can be set in the CECRCR1 <CECRIHLD> bit. To enable the setting, a timeout setting with the CECRCR1 <CECTOUT> bit is required.

Under suspend-enable condition, if CEC keeps receiving the next bit and the entire reception including the ACK bit is completed, CEC generates an interrupt after a reversed ACK response is executed. "1" is set to the bits of the CECRSTAT register: the <CECRIEND> bit that indicates the reception completion, and the bits corresponding to the detected errors.

If the reception of the next bit is interrupted, CEC starts to measure the timeout period, and an interrupt is generated after the timeout. "1" is set to the bits of the CECRSTAT register corresponding to the detected error.

The timeout is measured from the end of the last bit received as is the case with wait time of a bus to be free in transmission.

The information that the interrupts are suspended is held until the EOM bit is received or the timeout occurs. Thus, an interrupt is generated in each reception of a byte of data if multiple bytes are received while interrupts are suspended. "1" is set to the bits of the CECRSTAT register: the <CECRIEND> bit that indicates the reception completion, and the bits corresponding to the detected errors. The flags of the suspended interrupts and the reception completion are set to the bits of the CECRSTAT register.

Note 1: A minimum cycle error interrupt is generated upon detecting a minimum cycle error in the next received bit while interrupts are suspended. "Low" is output to CEC for approx. 3.63 ms. The flags of the suspended interrupts and the minimum cycle error are set to the bits of the CECRSTAT register.

Note 2: If an interrupt other than a minimum cycle error interrupt is generated while interrupts are suspended, CEC continues reception until the ACK response or the timeout. All the flags of the detected interrupts are set to the bits of the CECRSTAT register.

15.4.2.6 Stopping Reception

Writing "0" to the CECREN <CECREN> bit disables data reception. If the data reception is disabled during data reception, receiving operation stops and the received data is discarded.

Note: If the reception is disabled while "Low" is sent as a signal of minimum cycle error, the "Low" output is stopped as well.

15.4.3　Transmission

15.4.3.1　Basic Operation

In the transmission setting, the CEC firstly confirms the bus free wait status; it checks whether a CEC falling edge signal does not exit for specified bit cycles, and then sends a start bit. The confirmation of bus free wait is performed all the time. Thus once bus free wait condition is satisfied, a transmission will start soon when transmission setting is done.

After transmitting a start bit, CEC transmits one byte data and EOM data that are stored in the transmit buffer to the shift register. When the transmission of the first bit of the one byte data begins, transmission interrupt is generates, and CECTSTAT<CECTISTA> is set. After transmission interrupt generation, next one byte data is prepared to the transmit data buffer.

One byte data transmission completes in order of transmission of 8 bits data, EOM bit, ACK bit transmission and ACK bit response confirmation.

Data transmission continues until EOM is set to "1".

If EOM is set to "1", the end of transmission interrupt generates after confirmation of data, EOM, ACK bit transmission and ACK bit response. By the end of transmission interrupt generates, CECTSTAT<CECTIEND> is set.

Interrupt generation ends a series of transmission process, and CECTEN<CECTEN> is cleared.

If an error is generated during transmission, an error interrupt is generates to stop transmission.

Even if reception is enabled, no reception is executed during transmission.
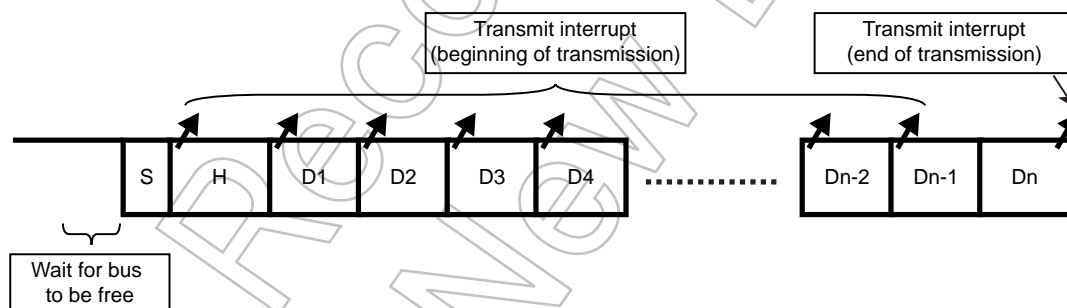
15.4.3.2　Preconfiguration

Before transmitting data, transmission settings to the Transmit Control Register (CECTCR) and the Transmit Buffer Register (CECTBUF) are required.

(1)　Bus Free Wait Time

Specify the bus free wait time in the CECTCR<CECFREE> bits. It can be specified in a range of 1 to 16 bit cycles.

Counting of the bus free wait time begins one bit cycle after the falling edge of the final bit. If the signal stays high for the specified number of bit cycles, transmission starts.
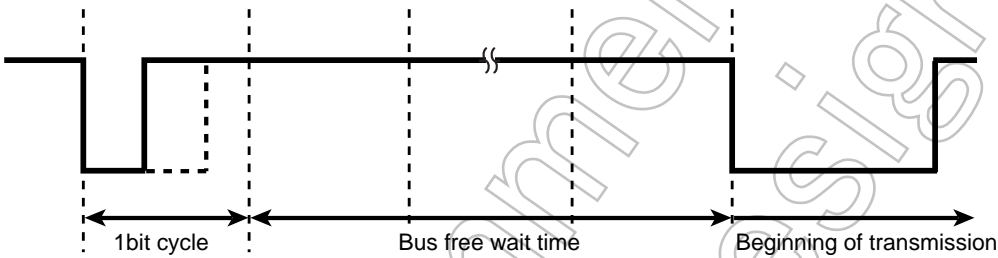
(2)　Transmitting Broadcast Message

Set the CECTCR <CECBRD> bit when transmitting a broadcast message.If this bit is set, logical "0" response during an ACK cycle results in an error.If not, logical "1" response during an ACK cycle results in an error.

(3)　Adjusting Transmission Waveform

Both start bit and data bit are capable of adjusting the rising timing and cycle. With the CECTCR <CECSTRS> <CECSPRD> <CECDTRS> <CECDPRD> bits, the timing can be specified between the defined fastest rising/cycle timing and the reference value.

The following figures show how the waveforms differ according to the configurations of the start bit, logical "0" and logical "1".

Note:Use <CECDTRS> in the same settings used for CECRCR1<CECLNC>.

Start bit	<CECSTRS> 121/fs - 7/fs to 121/fs (approx.3.693 ms) <CECSPRD> 147/fs - 7/fs to 147/fs (approx.4.486 ms)
Logical "0"	<CECDTRS> 49/fs - 3/fs to 49/fs (approx.1.495 ms) <CECDPRD> 79/fs - 15/fs to 79/fs (approx.2.411 ms)
Logical "1"	<CECDTRS> 20/fs - 3/fs to 20/fs (approx.0.610 ms) <CECDPRD> 79/fs - 15/fs to 79/fs (approx.2.411 ms)
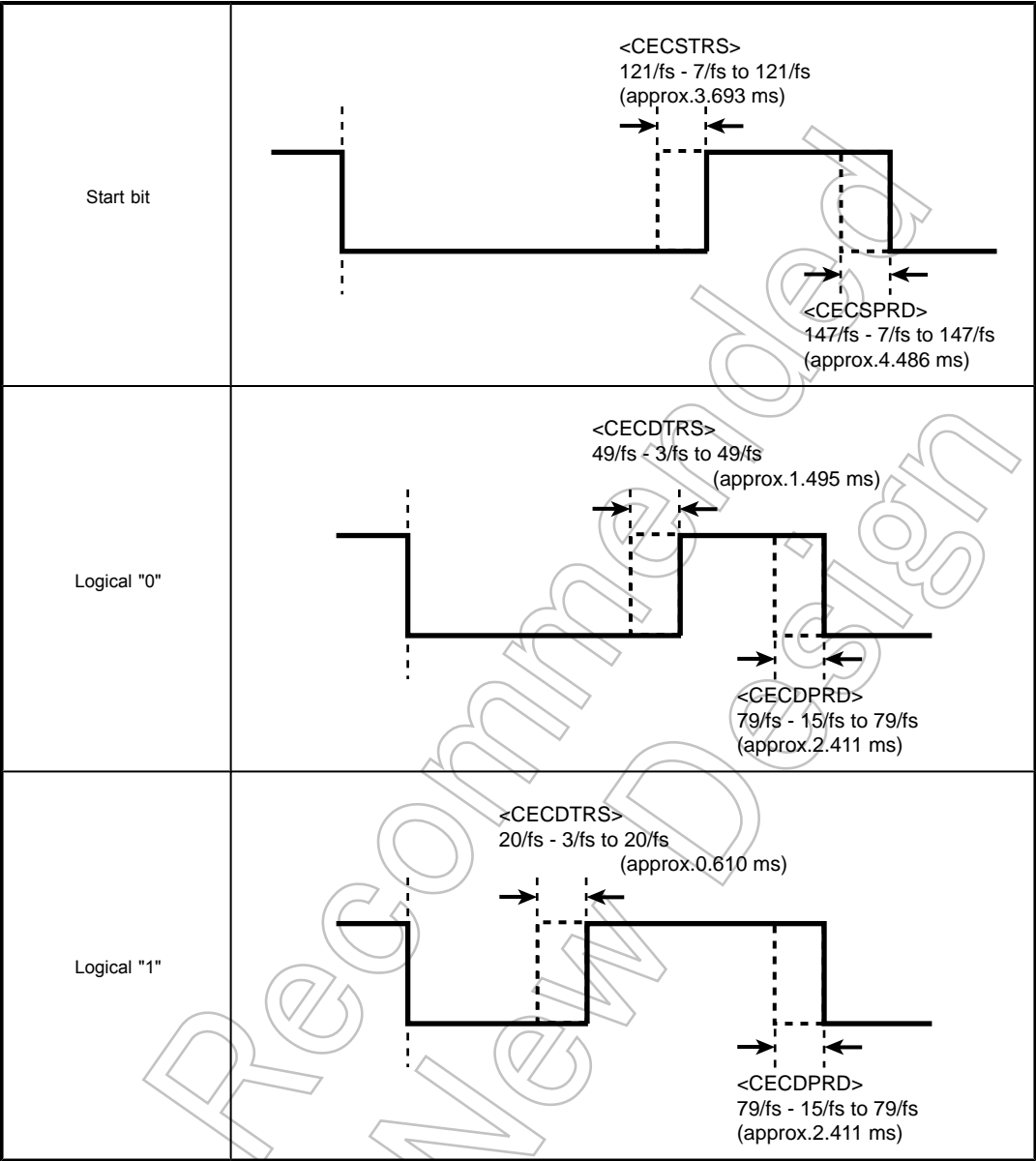
(4)   Preparing Transmission Data

Configure a byte of transmission data and EOM data with the CECTBUF register.

15.4.3.3   Detecting Transmission Error

Error detection during transmission generates an interrupt and stops transmission. It clears the CECTEN <CECTEN> bit.

To identify an error factor, the CECTSTAT register has bits that correspond with each interrupt. You can identify the interrupt factor by checking these bits.

Note: An attempt to stop transmission by an error may cause an improper waveform output to CEC. This is because output is stopped immediately after the error occurs.

15.4.3.4   Details of Transmission Error

(1)   Arbitration Lost

An arbitration lost error occurs when CEC detects "Low" on completion of appropriate low duration.

Detecting an arbitration lost error sets the CECTSTAT <CECTIAL> bit.

Two types of the arbitration lost detection periods are shown below.
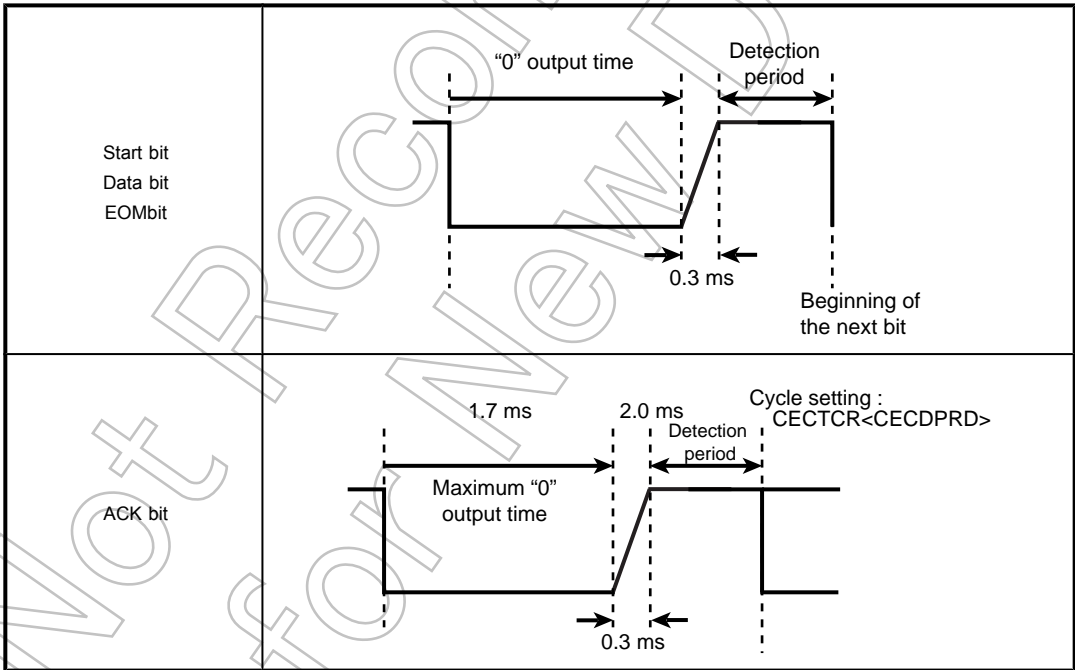
(2)   ACK error

An ACK error interrupt occurs when an ACK response does not conform to the configuration speci-
fied in the CECTCR <CECBRD> bit.

When the ACK error interrupt occurs, the CECTSTAT <CECTIACK> bit is set.

The ACK error is detected in the following cases.

Configuration	Determined as an ACK error when
<CECBRD> = 0 Broadcast transmission?: No	ACK response is logical "1"
<CECBRD> = 1 Broadcast transmission?: Yes	ACK response is logical "0"

(3)   Transmit Buffer Underrun

A transmit buffer underrun error is caused by the following sequence.

1. Data in the transmit buffer is transmit to the shift register.

2. An interrupt occurs.

3. A byte of data is transmitted.

4. No data is set to the transmit buffer before starting transmission of a byte of subsequent data.

When an underrun error occurs, the CECTSTAT <CECTIUR> bit is set.

(4)   Order of ACK Error and Transmit Buffer Overrun

If interrupt factors of the ACK error and transmit buffer underrun are detected at the end of trans-
mission of a byte of data, the transmit buffer underrun has priority.

The transmit buffer underrun interrupt occurs first and then the ACK error interrupt occurs.

15.4.3.5   Stopping Transmission

To stop transmission, send data including the EOM bit that indicates "1". This generates a transmit com-
pletion interrupt.

Please note that proper operation is not ensured if the start bit of transmission is set to "0" during trans-
mission.

15.4.3.6   Retransmission

Transmission is stopped by error detection. To retry the transmission, configure the condition and data
of starting the transmission.

15.4.4   Software Reset

The entire CEC function can be initialized by software.

Setting "1" to the CECRESET <CECRESET> bit causes the following operations.

- ・ Reception : Immediately stops. The received data is discarded.
- ・ Transmission : Immediately stops including output to the CEC line.
- ・ Register : All the registers other than CECEN are initialized.

Please note that software reset during transmission may cause the CEC line waveform that does not identical to the defined.

16. CAN Controller (CAN)

This product includes one channel of CAN controller.

16.1 Overview

- · Compliant with CAN version 2.0 B (active)
- · Standard and extended formats supported
- · Data frames and remote frames supported for each format
- · 32 Mailboxes (31 receive and transmit, 1 receive only)
- · CAN bus baud rate up to 1 Mbps (with a system clock of at least 48 MHz)
- · Bit timing parameter equivalent to Intel 82527 ™
- · Baud rate prescaler built in
- · The order in which messages are transmitted can be selected from the following two types of internal arbitrations :
    - The mailbox with the lower number will be sent first
    - The mailbox with the higher priority identifier will be sent first
- · Time stamp function for receive and transmit messages
- · Operation modes

Normal operation mode	
Configuration mode	
Sleep mode	CAN walk-up with CAN bus active state detection (at CANMCR<WUBA> ="1") or a write access to the master control register MCR
Suspend mode	Inactive state on the CAN bus
Test loop back mode	Self acknowledge
Test error mode	Writable error counters

- · Message receive mask function for two systems
    - Programmable global receive mask (common to mailboxes 0 to 31)
    - Programmable local receive mask (for mailbox 31 only)
- · Receive mask bit for ID extension bit
- · Interrupt signal

INTCANRX	: CAN receive completion interrupt
INTCANTX	: CAN transmit completion interrupt
INTCANGB	: CAN global interrupt Interrupt from eight causes including warning level, error passive and bus-off interrupts)

16.2 Block Diagram

Figure 16-1 shown the block diagram for the CAN controller.
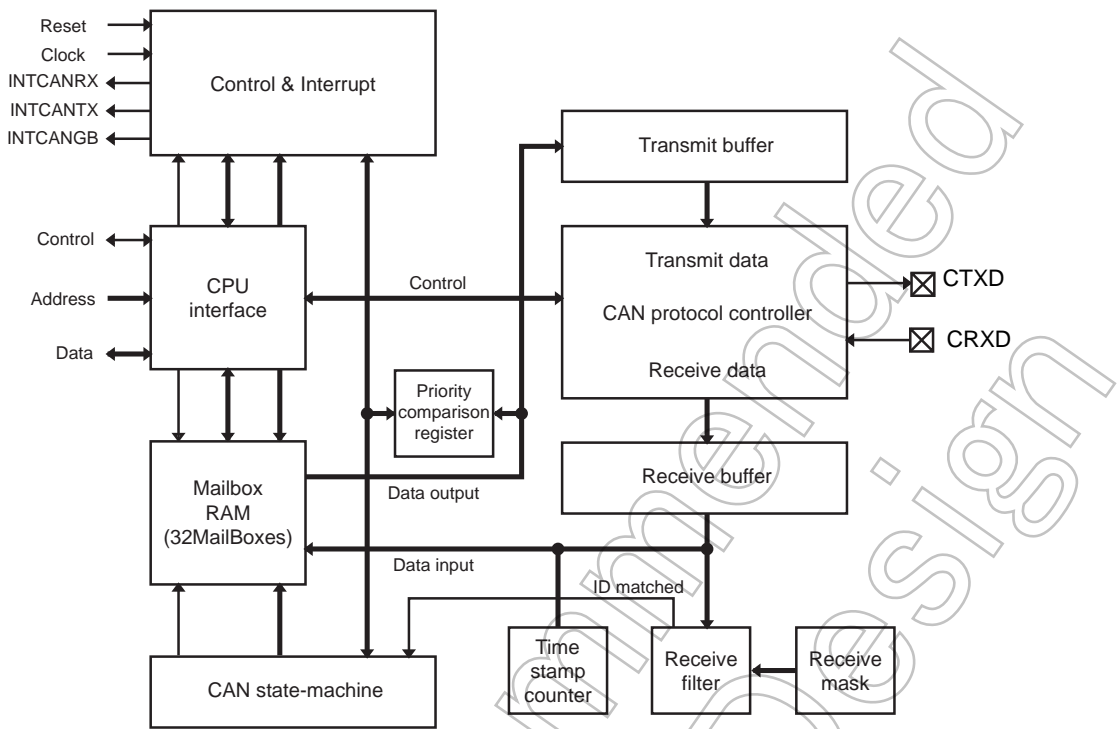
Figure 16-1 Block Diagram of CAN controller

16.3 CAN Interface

The interface to the CAN bus is an input pin CRXD and an output pin CTXD. Connect these pins via the CAN bus transceiver (ISO / DIS 11898 compliant).

High speed and low speed transceivers are differentiated. When this IP care must be taken that the electrical characteristics (e.g. , 3.3 V to 5 V) of these pins at chip level satisfy the needs of the transceiver.

16.4 Register

16.4.1 Register list

Mail Box x	Base Address
Channel0	0x4000_2000
Channel1	0x4000_2020
Channel2	0x4000_2040
Channel3	0x4000_2060
Channel4	0x4000_2080
Channel5	0x4000_20A0
Channel6	0x4000_20C0
Channel7	0x4000_20E0
Channel8	0x4000_2100
Channel9	0x4000_2120
Channel10	0x4000_2140
Channel11	0x4000_2160
Channel12	0x4000_2180
Channel13	0x4000_21A0
Channel14	0x4000_21C0
Channel15	0x4000_21E0
Channel16	0x4000_2200
Channel17	0x4000_2220
Channel18	0x4000_2240
Channel19	0x4000_2260
Channel20	0x4000_2280
Channel21	0x4000_22A0
Channel22	0x4000_22C0
Channel23	0x4000_22E0
Channel24	0x4000_2300
Channel25	0x4000_2320
Channel26	0x4000_2340
Channel27	0x4000_2360
Channel28	0x4000_2380
Channel29	0x4000_23A0
Channel30	0x4000_23C0
Channel31	0x4000_23E0

Register name (x=0 to 31)		Address(Base+)
Message ID Field Register	CANMBxID	0x0000
Time Stamp Values / Message Control Field Register	CANMBxTSVMCF	0x0008
Data Field Register	CANMBxDL	0x0010
Data Field Register	CANMBxDH	0x0018

Base Address = 0x4000_2400

Register name		Address(Base+)
Mailbox Configuration Register	CANMC	0x0000
Mailbox Direction Register	CANMD	0x0008
Transmission Request Set Register	CANTRS	0x0010
Transmission Request Reset Register	CANTRR	0x0018
Transmission Acknowledge Register	CANTA	0x0020
Abort Acknowledge Register	CANAA	0x0028
Receive Message Pending Register	CANRMP	0x0030
Receive Message Lost Register	CANRML	0x0038
Local Acceptance Mask Register	CANLAM	0x0040
Global Acceptance Mask Register	CANGAM	0x0048
Master Control Register	CANMCR	0x0050
Global Status Register	CANGSR	0x0058
Bit Configuration Register 1	CANBCR1	0x0060
Bit Configuration Register 2	CANBCR2	0x0068
Global Interrupt Flag Register	CANGIF	0x0070
Global Interrupt Mask Register	CANGIM	0x0078
Mailbox Transmit Interrupt Flag Register	CANMBTIF	0x0080
Mailbox Receive Interrupt Flag Register	CANMBRIF	0x0088
Mailbox Interrupt Mask Register	CANMBIM	0x0090
Change Data Request Register	CANCDR	0x0098
Remote Frame Pending Register	CANRFP	0x00A0
CAN Error Counter Register	CANCEC	0x00A8
Time Stamp Counter Prescaler Register	CANTSP	0x00B0
Time Stamp Counter Register	CANTSC	0x00B8

16.4.2 CANMBxID (Message ID Field Register)

	31	30	29	28	27	26	25	24
bit symbol	IDE	GAME/LAME	RFH			ID		
After reset								
	23	22	21	20	19	18	17	16
bit symbol				ID				
After reset								
	15	14	13	12	11	10	9	8
bit symbol				ID				
After reset								
	7	6	5	4	3	2	1	0
bit symbol				ID				
After reset								

Bit	Bit Symbol	Type	Function
31	IDE	R/W	ID Extension bit 0: Standard format (11-bit ID) from <ID28> to <ID18> used 1: Extended format (29-bit ID) from <ID28> to <ID0> used Sets the mailbox by selecting whether to receive or transmit the extended format (<IDE>="1") or the standard format (<IDE>="0").
30	GAME / LAME	R/W	Global (GAME) / Local (LAME) acceptance mask enable bit 0: Receive mask is not used for receive filtering. 1: Receive mask is used for receive filtering. <GAME> is the enable bit for the global acceptance mask GAM shared in mailboxes 0 to 30, and <LAME> is the enable bit for the local acceptance mask LAM used only for mailbox 31. When <GAME>=0 or <LAME>=0, the received message are stored in the mailbox only when the receive message ID is the same as the mailbox ID. For transmit mailboxes, the acceptance mask function is not applied. In such case, always set <GAME> to "0".
29	RFH	R/W	Remote frame handling bit (only for transmit mailboxes) 0:Transmit mailboxes do not respond to remote frames. Software must handle remote frames. 1:Transmit mailboxes respond to remote frames. (The <TRS> bit is set.) <RFH> determines whether a mailbox configured as a transmit mailbox will automatically respond to remote frame reception. When the ID of the received remote frame matches the ID of the transmit mailbox where <RFH>="1" and <GAME>="1", this mailbox ID is overwritten with the remote ID, and the mailbox automatically responds the remote frame using the overwritten ID. Handled as data frames in the case of receive mailboxes.(The <RMP> bit and the <RFP> bit are set.)
28-0	ID[28:0]	R/W	Message ID Standard format (11-bit ID) : From <ID28> to <ID18> are used. Extended format (29-bit ID) :From <ID28> to <ID0> are used. For the priority of message IDs, the message ID having most "0"s consecutively starting from the ID's highest bit (<ID28> bit) has the higher priority.

Register the mailbox IDs at the time of initial setup. To change the message ID field or a mailbox after the mailbox is enabled, clear the <MCx> bit in the CANMC register corresponding to the mailbox to "0", and then disable the mailbox for the CAN controller before writing a new ID.

16.4.3 CANMBxTSVMCF (Time Stamp Values / Message Control Field Register)

	31	30	29	28	27	26	25	24
bit symbol				TSV				
After reset								
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset								
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	RTR		DLC		
After reset								

Bit	Bit Symbol	Type	Function
31-16	TSV[15:0]	R/W	Time stamp counter value The 16-bit time stamp counter values read when message have been successfully received or transmitted are stored. No value is set when message reception or transmission fails. For the details of the entire time stamp counter function, Refer to "16.5.6 Time Stamp Function".
15-5	–	R	Read undefined. Write as "0".
4	RTR	R/W	Remote frame transmit request bit. 0:Data frame 1:Remote frame
3-0	DLC[3:0]	R/W	Data length code Sets the data length (number of bytes) of messages

<DLC[3:0]>	Number of bytes	Corresponding data
0000	0 byte	None
0001	1 byte	D0
0010	2 bytes	D0,D1
0011	3 bytes	D0,D1,D2
0100	4 bytes	D0,D1,D2,D3
0101	5 bytes	D0,D1,D2,D3,D4
0110	6 bytes	D0,D1,D2,D3,D4,D5
0111	7 bytes	D0,D1,D2,D3,D4,D5,D6
1000	8 bytes	D0,D1,D2,D3,D4,D5,D6,D7

When <DLC3:0>="1001" or more is set, data length is processed as 8 bytes.

The time stamp values do not need to be initially set.

The message control field needs no initial programming in the case of receive mailboxes. When a received message is stored in the mailbox, <RTR> and <DLC[3:0]> are also stored in the message control field at the same time. The transmit mailboxes need initial setting.

To change the message control field of a transmit mailbox (which is set to <RFH>="1") after enabling the mailbox, clear the CANMC<MCx> bit to "0" and then disable the mailbox for the CAN controller before writing a new <RTR> and <DLC[3:0]>. The message control field of the transmit mailbox set to <RFH>="0" can be changed irrespective of the CANMC<MCx> bit setting, but the user needs to check that the CANTRS<TRSx> bit is "0" before writing a new <RTR> and <DLC[3:0]>.

16.4.4 CANMBxDH/CANMBxDL (Data fields Register)

For transmission, data is transmitted according to the data byte count set in the <DLC[3:0]> of the mailbox.

For reception, the data length code in the received message is copied to the <DLC[3:0]> of the mailbox, and the data byte count only set in the <DLC[3:0]> is made valid.

Mailboxes are readable and writable, but do not write data fields for receive mailboxes. If data fields are written, a mismatch may occur in received data.

To update the data field of a transmit mailbox set to <RFH>="1", set "1" in CANCDR<CDR> and suspend transmit requests temporarily before writing new data. To update the data field of a transmit mailbox set to <RFH>="0", check that the CANTRS<TRS> bit is "0" before writing new data.

CANMBxDH

	31	30	29	28	27	26	25	24
bit symbol					D7			
After reset								
	23	22	21	20	19	18	17	16
bit symbol					D6			
After reset								
	15	14	13	12	11	10	9	8
bit symbol					D5			
After reset								
	7	6	5	4	3	2	1	0
bit symbol					D4			
After reset								

Bit	Bit Symbol	Type	Function
31-24	D7[7:0]	R/W	Transmitted and received data is stored.
23-16	D6[7:0]	R/W	Transmitted and received data is stored.
15-8	D5[7:0]	R/W	Transmitted and received data is stored.
7-0	D4[7:0]	R/W	Transmitted and received data is stored.

CANMBxDL

	31	30	29	28	27	26	25	24
bit symbol				D3				
After reset								
	23	22	21	20	19	18	17	16
bit symbol				D2				
After reset								
	15	14	13	12	11	10	9	8
bit symbol				D1				
After reset								
	7	6	5	4	3	2	1	0
bit symbol				D0				
After reset								

Bit	Bit Symbol	Type	Function
31-24	D3[7:0]	R/W	Transmitted and received data is stored.
23-16	D2[7:0]	R/W	Transmitted and received data is stored.
15-8	D1[7:0]	R/W	Transmitted and received data is stored.
7-0	D0[7:0]	R/W	Transmitted and received data is stored.

16.4.5   CANMC (Mailbox Configuration Register)

	31	30	29	28	27	26	25	24
bit symbol	MC31	MC30	MC29	MC28	MC27	MC26	MC25	MC24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	MC23	MC22	MC21	MC20	MC19	MC18	MC17	MC16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	MC15	MC14	MC13	MC12	MC11	MC10	MC9	MC8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MC7	MC6	MC5	MC4	MC3	MC2	MC1	MC0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	MC31 to MC0	R/W	Access configuration to mailbox (Each bit corresponds with mailboxes 31 to 0) 0:The corresponding mailbox MBx is disabled for the CAN controller. 1:The corresponding mailbox MBx is enabled for the CAN controller. Write access from CPU _see sub-table below_

	ID field	Transmit mailbox with <RFH>="1"	Data field	Control field
<MCx>=0	Enabled	Enabled	Enabled	Enabled
<MCx>=1	Disabled	Disabled	Enabled	Enabled

Note: Following care is required during reprogramming of a CANMC in operation.

Receive: For a receive mailbox it needs to be ensured that the mailbox is not being disabled while reception for this mailbox is ongoing. If a mailbox is disabled or reconfigured during an ongoing reception, the current frame might be received.

Transmit: When the CAN controller is transmitting data (CANTRS<TRSx>="1"), Clear <MCx> to "0" after the transmission is completed (CANTRS<TRSx>="0").

16.4.6 CANMD (Mailbox Direction Register)

	31	30	29	28	27	26	25	24
bit symbol	MD31	MD30	MD29	MD28	MD27	MD26	MD25	MD24
After reset	1	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	MD23	MD22	MD21	MD20	MD19	MD18	MD17	MD16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	MD15	MD14	MD13	MD12	MD11	MD10	MD9	MD8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MD7	MD6	MD5	MD4	MD3	MD2	MD1	MD0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	MD31	R	Mailbox direction : Mailbox 31 Mailbox 31 is the receive-only mailbox. This is always set to "1" and can not be changed.
30-0	MD30 to MD0	R/W	Mailbox direction : Mailboxes 30 to 0 (Each bit corresponds with mailboxes 30 to 0.) 0:Set as a transmit mailbox. 1:Set as a receive mailbox. Each mailbox can be set as a transmit or receive mailbox.

Set the CANMD register at the initial setup. The directions of mailboxes cannot be changed when operation is ongoing. To change CANMD register settings, set the corresponding CANMC<MCx> bit to "0" before making changes.

16.4.7   CANTRS (Transmission Request Register)

	31	30	29	28	27	26	25	24
bit symbol	-	TRS30	TRS29	TRS28	TRS27	TRS26	TRS25	TRS24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	TRS23	TRS22	TRS21	TRS20	TRS19	TRS18	TRS17	TRS16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TRS15	TRS14	TRS13	TRS12	TRS11	TRS10	TRS9	TRS8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TRS7	TRS6	TRS5	TRS4	TRS3	TRS2	TRS1	TRS0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	−	R	Read : Read as "0". Write : Write as "0".
30-0	TRS30 to TRS0	R/W	Transmit request set (Each bit corresponds with mailboxes 30 to 0.) Set <TRSx> requests the message transmission of corresponding mailbox x. When transmission is requested for multiple mailboxes, the message are transmitted in accordance with the priority corresponding to the MCR<MTOS> bit. A write of "1" from the CPU to mailbox x configured as transmit mailbox can set the bit. A write of "0" from the CPU is invalid.

Note: Mailbox 31 is receive-only mailbox.

The transmission request set register can be set by a write of "1" from the CPU to only the CANTRS<TRSx> bits of the mailboxes configured for transmission. The CANTRS<TRSx> bits of the mailboxes configured for reception cannot be set.

The CANTRS<TRSx> bit is cleared to "0" when the message has been successfully transmitted or the transmit request is reset by setting the CANTRR<TRRx> bit to "1."

When transmission fails, the transmission process is repeated until it succeeds or the transmit request is reset by setting the CANTRR<TRRx> bit to "1."

When the CANTRS<TRSx> bit is "1", do not write to mailbox x.

16.4.8 CANTRR (Transmission Request Register)

	31	30	29	28	27	26	25	24
bit symbol	-	TRR30	TRR29	TRR28	TRR27	TRR26	TRR25	TRR24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	TRR23	TRR22	TRR21	TRR20	TRR19	TRR18	TRR17	TRR16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TRR15	TRR14	TRR13	TRR12	TRR11	TRR10	TRR9	TRR8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TRR7	TRR6	TRR5	TRR4	TRR3	TRR2	TRR1	TRR0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	–	R	Read : Read as "0". Write : Write as "0".
30-0	TRR30 to TRR0	R/W	Transmit request reset (Each bit corresponds with mailboxes 30 to 0.) Setting <TRRx> cancels the message transmission of corresponding mailbox x. A write of "1" from the CPU to mailbox x configured as transmit mailbox can set the bit. Write of "0" from the CPU is invalid.

Note: Mailbox 31 is receive-only mailbox.

The transmission request reset register can be set by a write of "1" from the CPU to only the CANTRR<TRRx> bits of the mailboxes configured for transmission. The CANTRR<TRRx> bits of the mailboxes configured for reception cannot be set.

The CANTRR<TRRx> bit is cleared to "0" by the internal logic when the message has been successfully transmitted or the transmission is aborted. A write of "0" from the CPU is invalid.

When the CANTRR<TRRx> bit is "1," do not write to mailbox x.

Setting the CANTRR<TRRx> bit cancels the message transmission of mailbox x set by the CANTRS<TRSx> bit, where the operation executed will be any of the following three sequences:

a. A transmission request of a message has not yet been transmitted.

A transmission request of a message will be cleared immediately.

(CANTRS<TRSx> = 0, CANTRR<TRRx> = 0, CANAA<AAx> = 1)

b. A transmission request of a message is currently being transmitted and an arbitration lost error occurs or an error is detected on the CAN bus.

A transmission request of a message will be cleared and the transmission will be canceled.

(CANTRS<TRSx> = 0, CANTRR<TRRx> = 0, CANAA<AAx> = 1)

c. A transmission request of a message is currently being transmitted and no arbitration lost error occurs or no error is detected on the CAN bus.

A transmission request of a message will not be cleared and the transmission will be completed.

(CANTRS<TRSx> = 0, CANTRR<TRRx> = 0, CANTA<TAx> = 1)

16.4.9   CANTA (Transmission Acknowledge Register)

	31	30	29	28	27	26	25	24
bit symbol	-	TA30	TA29	TA28	TA27	TA26	TA25	TA24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	TA23	TA22	TA21	TA20	TA19	TA18	TA17	TA16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TA15	TA14	TA13	TA12	TA11	TA10	TA9	TA8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TA0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	−	R	Read : Read as "0". Write : Write as "0".
30-0	TA30 to TA0	R/W	Transmission acknowledge (Each bit corresponds with mailboxes 30 to 0) When the message in mailbox x has been successfully transmitted, the <TAx> bit is set to "1". The <TAx> bit can be cleared by a write of "1" from the CPU to the <TAx> bit or the TRS<TRSx> bit.

Note: Mailbox 31 is receive-only mailbox.

The CANTA<TAx> bit is set to "1" when a message in mailbox x has been successfully transmitted. When the mailbox interrupt is enabled by setting the corresponding <MBIMx> bit in the mailbox interrupt mask register CANMBIM to "1", the <MBTIFx> bit of the mailbox transmit interrupt flag register CAN-MBTIF is set to "1" and the CAN transmit completion interrupt INTCANTX occurs.

A write of "1" to the <TAx> bit or the CANTRS<TRSx> bit from the CPU can clear the <TAx> bit. A write of "0" to the <TAx> bit or the CANTRS<TRSx> bit from the CPU is invalid.

16.4.10 CANAA (Abort Acknowledge Register)

	31	30	29	28	27	26	25	24
bit symbol	-	AA30	AA29	AA28	AA27	AA26	AA25	AA24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	AA23	AA22	AA21	AA20	AA19	AA18	AA17	AA16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	AA15	AA14	AA13	AA12	AA11	AA10	AA9	AA8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	AA7	AA6	AA5	AA4	AA3	AA2	AA1	AA0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	–	R	Read : Read as "0". Write : Write as "0".
30-0	AA30 to AA0	R/W	Abort acknowledge (Each bit corresponds with mailboxes 30 to 0.) When the message in mailbox x has not been successfully transmitted, the <AAx> bit is set to "1". The <AAx> bit can be cleared by a write of "1" from CPU to the <AAx> bit or the CANTRS<TRSx> bit.

Note: Mailbox 31 is receive-only mailbox.

The CANAA<AAx> bit is set to "1" when a message in mailbox x has not been successfully transmitted. When CANGIF<TRMABF> bit in the global interrupt flag register is also set to "1", and the transmit abort interrupt is enabled by setting the CANGIM<TRAMABM> bit in the global interrupt mask register to "1", the CAN global interrupt INTCANGB occurs.

A write of "1" to the <AAx> bit or the CANTRS<TRSx> bit from the CPU can clear the <AAx> bit. A write of "0" to the <AAx> bit or the CANTRS<TRSx> bit from the CPU is invalid.

16.4.11   CANCDR (Change Data Request Register)

	31	30	29	28	27	26	25	24
bit symbol	-	CDR30	CDR29	CDR28	CDR27	CDR26	CDR25	CDR24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	CDR23	CDR22	CDR21	CDR20	CDR19	CDR18	CDR17	CDR16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	CDR15	CDR14	CDR13	CDR12	CDR11	CDR10	CDR9	CDR8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CDR7	CDR6	CDR5	CDR4	CDR3	CDR2	CDR1	CDR0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	–	R	Read : Read as "0". Write : Write as "0".
30-0	CDR30 to CDR0	R/W	Change data request (Each bit corresponds with mailboxes 30 to 0.) When the <CDRx> bit of transmit mailbox x is set to "1", the transmit request of this mailbox x is ignored. It means mailbox x for which the CANTRS<TRSx> bit and the <CDRx> bit are set will be excluded from the internal arbitration range and will not be transmitted if transmission has not started. After the <CDRx> bit is cleared to "0", mailbox x is back to be included in the internal arbitration range.

Note: Mailbox 31 is receive-only mailbox.

The change data request register CABCDR is effective when updating the data field of transmit mailbox x where auto acknowledgement of remote frames is enabled (CANMBnID<RFH>="1"). Mailbox x enabling automatic acknowledgement starts message transmission automatically responding to received remote frames and so may update the data field during message transmission (In such cases, updated data is output midway through transmission). The update of the data field can be avoided by setting the <CDRx> bit to "1" and temporarily suspending data transmission.

16.4.12   CANRMP (Receive Message Pending Register)

	31	30	29	28	27	26	25	24
bit symbol	RMP31	RMP30	RMP29	RMP28	RMP27	RMP26	RMP25	RMP24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	RMP23	RMP22	RMP21	RMP20	RMP19	RMP18	RMP17	RMP16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	RMP15	RMP14	RMP13	RMP12	RMP11	RMP10	RMP9	RMP8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RMP7	RMP6	RMP5	RMP4	RMP3	RMP2	RMP1	RMP0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	RMP31 to RMP0	R/W	Receive message pending (Each bit corresponds with mailboxes 31 to 0.) After a message is received and the content of the received message is written in mailbox x, the <RMPx> bits set to "1". After received data is read, a write of "1" to the <RMPx> bit can clear the <RMPx> bit.

The CANRMP<RMPx> bit is set to "1" when a message in mailbox x has been successfully received. When the mailbox interrupt is enabled by setting the corresponding <MBIMx> bit in the mailbox interrupt mask register CANMBIM to "1", the <MBRIFx> bit of the mailbox receive interrupt flag register CAN-MBRIF is set to "1" and the CAN receive completion interrupt INTCANRX occurs.

To clear the <RMPx> bit, write "1" to the <RMPx> bit from the CPU. A write of "0" to the <RMPx> bit from the CPU is invalid.

16.4.13   CANRML (Receive Message Lost Register)

	31	30	29	28	27	26	25	24
bit symbol	RML31	RML30	RML29	RML28	RML27	RML26	RML25	RML24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	RML23	RML22	RML21	RML20	RML19	RML18	RML17	RML16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	RML15	RML14	RML13	RML12	RML11	RML10	RML9	RML8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RML7	RML6	RML5	RML4	RML3	RML2	RML1	RML0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	RML31 to RML0	R/W	Receive message lost (Each bit corresponds with mailboxes 31 to 0.) When mailbox x for which the <RMPx> bit is set to "1" receives the next message, the content of the received message is overwritten to the mailbox x, and the <RMLx> bit is set to "1". A write of "1" to the <RMPx> bit can clear the <RMLx> bit.

The CANRML<RMLx> bit is set by the internal logic and can be cleared with a write of "1" to the CANRMP<RMPx> bit from the CPU. The <RMPx> bit is also cleared at the same time. A write of "1" or "0" to the <RMLx> bit from the CPU is invalid.

With the CANRMP<RMPx> bit set to "1", if mailbox x receives the next message, the corresponding <RMLx> bit in the receive message lost register CANRML is set to "1". In this case, mailbox x is overwritten with the new received message.

When the <TRMABF> bit in the global interrupt flag register CANGIF is also set to "1", and the transmit abort interrupt is enabled by setting the <TRMABM> bit in the global interrupt mask register CANGIM to "1", the CAN global interrupt INTCANGB occurs.

When the receive message lost interrupt is enabled by setting the <RMLIM> bit in the global interrupt mask register CANGIM to "1", the CAN global interrupt INTCANGB occurs.

Table 16-1 shows the changes of the CANRMP and CANRML registers before and after a message is received.

Table 16-1 Change of RMP and RML Registers Before / After a Message i Received

ID	Before Reception		After reception		Operation
	<RMPx>	<RMLx>	<RMPx>	<RMLx>	
No match	Don't care	Don't care	Don't care	Don't care	Received message are not stored in any mailboxes.
Match	0	0	1	0	The received message is stored in mailbox x with a matching ID.
	1	0	1	1	The received message is overwritten in mailbox x with a matching ID. This shows that the previous message was lost.
	1	1	1	1	

16.4.14 CANRFP (Remote Frame Pending Register)

	31	30	29	28	27	26	25	24
bit symbol	RFP31	RFP30	RFP29	RFP28	RFP27	RFP26	RFP25	RFP24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	RFP23	RFP22	RFP21	RFP20	RFP19	RFP18	RFP17	RFP16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	RFP15	RFP14	RFP13	RFP12	RFP11	RFP10	RFP9	RFP8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RFP7	RFP6	RFP5	RFP4	RFP3	RFP2	RFP1	RFP0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	RFP31 to RFP0	R/W	Remote frame pending (Each bit corresponds with mailboxes 31 to 0.) When mailbox x configured as receive mailbox receives a remote frame, the <RFPx> bit and CANRMP<RMPx> bit are set to "1". The <RFPx> bit can be cleared by a write of "1" to the CANRMP<RMPx> bit.

The CANRFP<RFPx> bit is set by the internal logic and can be cleared with a write of "1" to the CANRMP<RMPx> bit from the CPU. The <RMPx> bit is also cleared at the same time. A write of "0" to the <RMPx> bit and a write of "1" or "0" to the <RFPx> bit from the CPU are invalid.

Even when mailbox x with <RFPx>="1" is overwritten by data frame reception, the <RFPx> bit is cleared.

When the remote frame pending interrupt is enabled by setting the <RFPM> bit in the global interrupt mask register CANGIM to "1", the CAN global interrupt INTCANGB occurs.

16.4.15   CANLAM (Local Acceptance Mask Register)

The local acceptance mask register CANLAM will only be used for filtering of the receiving message ID for mailbox 31. This feature allows locally masking to any ID bits of the receiving message for mailbox 31.

	31	30	29	28	27	26	25	24
bit symbol	LAMI	-	-	LAM				
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	LAM							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	LAM							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	LAM							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	LAMI	R/W	Mask of the \<IDE> bit of mailbox 31 0:Not masked 1:Masked In case of \<LAMI>="0", the message in the standard or the extended format is received, according to the \<IDE> bit of the mailbox 31. In case of \<LAMI>="1", the message in the standard and the extended format is received, regardless of the \<IDE> bit of the mailbox 31.
30-29	−	R	Read : Read as "0". Write : Write as "0".
28-0	LAM[28:0]	R/W	Mask of receive message ID 0:Not masked The reception message is received when the corresponding bit of reception message ID is the same as mailbox ID. 1:Masked The reception message is received regardless of the value of the corresponding bit of reception message.

In the extended format, < ID[28:0] > and < LAM[28:0] > are used to filtering.

In the standard format, < ID[28:18] > and < LAM[28:18] > are used to filtering.

When the message in a standard format is received, the part of the extended ID (<ID[17:0]>) will become an undefined value. Therefore, the standard and the extended format cannot be recommended to be received in alternately the same mailbox.

Please set CANLAM when initialization (At the configuration mode) and do not change the setting while operating. When the setting is changed while receiving the message, the CANLAM value on the way of the setting change is used to filtering of reception message ID.

16.4.16 CANGAM (Global Acceptance Mask Register)

The global acceptance mask register CANGAM will be used for filtering of the receiving message ID for mailbox 0 to 30. This feature allows to globally masking any ID bits of the receiving message for mailbox 0 to 30.

	31	30	29	28	27	26	25	24
bit symbol	GAMI	-	-	GAM				
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	GAM							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	GAM							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	GAM							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	GAMI	R/W	Mask of the <IDE> bit of mailboxes 0 to 30 0:Not masked 1:masked In case if <GAMI> =" 0", the message of the standard or the extended format is received, according to the <IDE> bit of the mailboxes 0 to 30. In case of <GAMI> = "1", the message of the standard and the extended format is received, regardless of the <IDE> bit of the mailboxes 0 to 30.
30-29	–	R	Read : Read as "0". Write : Write as "0".
28-0	GAM[28:0]	R/W	Mask of receive message ID 0:Not masked The reception message is received when the corresponding bit of reception message ID is the same as mailbox ID. 1:Masked The reception message is received regardless of the value of the corresponding bit of reception message.

In the extended format, < ID[28:0] > and < GAM[28:0] > are used for filtering.

In the standard format, < ID[28:18] > and < GAM[28:18] > are used for filtering.

When the message in the standard format is received, the part of the extended ID (<ID[17:0]>) will become an undefined value. Therefore, the standard and the extended format cannot be recommended to be received in alternately the same mailbox.

Please set CANGAM during the initialization (At the configuration mode) and do not change the setting during the operation. When the setting is changed while receiving the message, the CANGAM value on the way of the setting change is used for filtering of reception message ID.

16.4.17   CANMCR (Master Control Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	SUR	-	TSTLB	TSTERR
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CCR	SMR	-	WUBA	MTOS	-	TSCC	SRES
After reset	1	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-12	−	R	Read : Read as "0". Write : Write as "0".
11	SUR	R/W	Suspend mode request 0: Cancels suspend mode (normal operation) 1: Request suspend mode
10	−	R	Read : Read as "0". Write : Write as "0".
9	TSTLB	R/W	Test loop back 0:Cancels test loop back mode (normal operation) 1:Request test loop back mode (This mode supports stand-alone operation.)
8	TSTERR	R/W	Test error 0:Cancels test error mode (normal operation) 1:Request test error mode (In this mode, it is possible to write the CAN error counter register (CANCEC)).
7	CCR	R/W	Change configuration request 0:Cancels configuration mode (normal operation) 1:Request configuration mode (In this mode, it possible to write the bit configuration registers, CANBCR1 and CANBCR2.)
6	SMR	R/W	Sleep mode request 0:Cancels sleep mode (normal operation) 1:Request sleep mode (In this mode, the clock of the CAN controller stops and the error counters and transmit requests are reset.)
5	−	R	Read : Read as "0". Write : Write as "0".
4	WUBA	R/W	Walk-up on bus activity 0: Wakes up only by a write access to the CAMCR register. 1:Wakes up by detecting a bus active state or a write access to the CAMCR.
3	MTOS	R/W	Mailbox transmission order select 0:Messages are transmitted in ascending order of mailbox number. 1:Messages in mailboxes are transmitted in descending order of message ID priority.
2	−	R	Read : Read as "0". Write : Write as "0".
1	TSCC	R/W	Time stamp counter clear 0: Disable 1:Clears the time stamp counter to "0". (note1) This bit is for write only and is read as always "0".

Bit	Bit Symbol	Type	Function
0	SRES(Note2)	R/W	Software reset 0:Disable 1:Resets the CAN controller by software. This bit is for write only and is read as always "0".

Note1:The time stamp counter is also cleared by a write to the CANTSP register and a write of "0" to the CANTSC register.

Note2:After software reset,all registers in CAN must be access after the following time.
- (1) When communication by CAN bus is not performed,please wait more than 16 CPU clocks.
- (2) When communication by CAN bus is performed,please wait more than 88 CPU clocks.

16.4.18 CANBCR1 (Bit Configuration Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	BRP	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	BRP							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-10	−	R	Read : Read as "0". Write : Write as "0".
9-0	BRP[9:0]	R/W	Baud rate prescaler Setting value : 0 to 1023

16.4.19 CANBCR2 (Bit Configuration Register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	SJW	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SAM	TSEG2			TSEG1			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-10	−	R	Read : Read as "0". Write : Write as "0".
9-8	SJW[1:0]	R/W	Resynchronization jump width. 00 : 1 × TQ 01 : 2 × TQ 10 : 3 × TQ 11 : 4 × TQ
7	SAM	R/W	Setting sampling count 0:Single sampling 1 Triple sampling
6-4	TSEG2[2:0]	R/W	Setting of bit time after sample point 000 : Reserved 100 : 5 × TQ 001 : 2 × TQ 101 : 6 × TQ 010 : 3 × TQ 110 : 7 × TQ 011 : 4 × TQ 111 : 8 × TQ
3-0	TSEG1[3:0]	R/W	Setting of bit time before sample point (except SYNCSEG). 0000 : Reserved 1000 : 9 × TQ 0001 : 2 × TQ 1001 : 10 × TQ 0010 : 3 × TQ 1010 : 11 × TQ 0011 : 4 × TQ 1011 : 12 × TQ 0100 : 5 × TQ 1100 : 13 × TQ 0101 : 6 × TQ 1101 : 14 × TQ 0110 : 7 × TQ 1110 : 15 × TQ 0111 : 8 × TQ 1111 : 16 × TQ

16.4.20 CANTSC (Time Stamp Counter Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TSC							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TSC							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read : Read as "0". Write : Write as "0".
15-0	TSC[15:0]	R/W	Time stamp counter Free-running 16-bit counter

Overflow of the CANTSC can be detected by the time stamp counter overflow interrupt flag <TSOIF> of the global interrupt flag register (CANGIF), and the time stamp counter overflow flag <TSO> of the global status register (CANGSR). Both flags can be cleared by writing "1" to <TSOIF> in the CANGIF register.

There is a 4-bit prescaler for the CANTSC. After power-up the time stamp counter is driven directly from the bit clock (<TSP[3:0]>="0"). The period T_{TSC} for the time stamp counter will be calculated with the following formula :

$$T_{TSC} = TBIT \times (TSP + 1)$$

16.4.21   CANTSP (Time Stamp Counter Prescaler Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	TSP			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	R	Read : Read as "0". Write : Write as "0".
3-0	TSP[3:0]	R/W	Time stamp counter prescaler Sets the value to be loaded to the prescaler for the 4-bit TSC.

To ensure that the value of the CANTSC will not change during the write cycle to the mailbox, a hold register is implemented. The value of the CANTSC will be copied to the hold register and then written to the mailbox from the hold register if a message has been received or transmitted successfully. The reception is successful for the receiver, if there is no error but the last one bit of End-of-frame. Transmission is successful for the transmitter if there is no error until the last bit of End-of-frame. (Refer to the CAN specification 2.0B).

16.4.22 CANGSR (Global Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	MIS
After reset	0	0	0	0	0	0	0	1
	15	14	13	12	11	10	9	8
bit symbol	MIS				RM	TM	-	SUA
After reset	1	1	1	1	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CCE	SMA	-	-	TSO	BO	EP	EW
After reset	1	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-17	–	R	Read : Read as "0". Write : Write as "0".
16-12	MIS[4:0]	R	Message in slot. Indicates the mailbox number of a message located in the transmit buffer. 00000 :Message for mailbox 0 ... 11110 :Message for mailbox 30, 11111 : There is no message in the transmit buffer.
11	RM	R	Receive mode 0:The CAN controller is not receiving a message. 1:The CAN controller is receiving a message.
10	TM	R	Transmit mode 0:The CAN controller is not transmitting a message. 1:The CAN controller is transmitting a message.
9	–	R	Read : Read as "0". Write : Write as "0".
8	SUA	R	Suspend mode acknowledge 0:The CAN controller is not in suspend mode. 1:The CAN controller is in suspend mode.
7	CCE	R	Change configuration enable 0:The CAN controller is not in configuration mode. 1:The CAN controller is in configuration mode. In this mode, it is possible to write the bit configuration registers, CANBCR1 and CANBCR2.
6	SMA	R	Sleep mode acknowledge 0:The CAN controller is not in sleep mode. 1:The CAN controller is in sleep mode. In this mode, the clock of the CAN controller stops and the error counters and transmit request are reset.
5-4	–	R	Read : Read as "0". Write : Write as "0".
3	TSO	R	Time stamp overflow 0:The time stamp counter is not overflow. 1:The time stamp counter has overflow at least once after this bit was last cleared to "0". To clear this bit, clear <TSOIF> bit in the CANGIF register to "0".

Bit	Bit Symbol	Type	Function
2	BO	R	Bus off status 0:In bus on state (normal operation) 1:In bus off state When CAN bus errors occur abnormally often and the transmit error counter <TEC> reaches its limit of 256, the CAN controller enters bus off state. No messages can be transmitted and received, The error counter is un-defined. After the bus off recovery sequence, the CAN controller automatically enters bus on state.
1	EP	R	Error passive status 0:The CAN controller is not in error passive mode. 1:The CAN controller is in error passive mode.
0	EW	R	Warning status 0:Both <TEC> and <REC> values are 96 or less. 1:At least one of the <TEC> and <REC> values is greater than 96 and has reached the warning level.

16.4.23 CANCEC (Error Counter Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TEC							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	REC							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read : Read as "0". Write : Write as "0".
15-8	TEC[7:0]	R	8-bit transit error counter (After reset release)
		R/W	8-bit transit error counter (CANMCR<TSTERR>="1")
7-0	REC[7:0]	R	8-bit receive error counter (After reset release)
		R/W	8-bit receive error counter (CANMCR<TSTERR>="1")

The CAN controller contains two error counters : the receive error counter <REC> and the transmit error counter <TEC>. The value of both counters can be read from the CPU. A write access to the error counters is only possible in test error mode (The <TSTERR> bit in the CANMCR register is "1"). In the case of a write to the CANCEC register, the write data to the lower 8 bits <REC> is written also to the higher 8 bits (TEC).

The CAN error counters count up or down according to the CAN Specification 2.0B.

The <REC> is not increased after exceeding the error passive limit (128).When <REC>=128, after the correct reception of a message, the <REC> is set to a value between 119 and 127. After reaching the "bus off" status, the error counters are undefined.

If the status "bus off" is reached, the receive error counter is incremented after 11 consecutive recessive bits on the bus. If the counter reaches the count 128, the module changes automatically to the status error active. All internal flags are reset and the error counters will be cleared to "0". The configuration registers keep the programmed values. The values of the counters are undefined during "bus off" status.

When CAN enters configuration mode, the error counters will be cleared.

16.4.24   CANGIF (Global Interrupt Flag Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RFPF	WUIF	RMLIF	TRMABF	TSOIF	BOIF	EPIF	WLIF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read : Read as "0". Write : Write as "0".
7	RFPF	R/W	Remote frame pending flag 0:No remote frame has been received. 1:Remote frames have been received. (in the receive mailbox) This bit will not be set when matching with the transmit mailbox for which the <TFH> bit is "1".
6	WUIF	R/W	Walk-up interrupt flag 0:In sleep mode or normal operation mode 1:Sleep mode has been canceled.
5	RMLIF	R/W	Receive message lost interrupt flag 0:No receive message lost error has occurred. 1:A receive message lost error has occurred in at least one mailbox configured as a receive mailbox.
4	TRMABF	R/W	Transmission abort flag 0:No transport abort has occurred. 1:Transport abort has occurred. (At least one bit in the CANAA register is set.)
3	TSOIF	R/W	Time stamp counter overflow interrupt flag 0:No overflow has occurred in the time stamp counter after this bit was last cleared. 1:There was at least one overflow of the time stamp counter after this bit was last cleared.
2	BOIF	R/W	Bus off interrupt flag 0: The CAN controller is in bus on mode. 1: The CAN controller is in bus off mode.
1	EPIF	R/W	Error passive interrupt flag 0: The CAN controller is in error active mode. 1: The CAN controller is in error passive mode.
0	WLIF	R/W	Warning level interrupt flag 0:None of the error counters have reached the warning level. 1: At least one of the error counters has reached the warning level.

Each interrupt flag of the global interrupt flag register (CANGIF) will be set to "1" if the corresponding global interrupt condition has met. When the global interrupt flag is set to "1", if the corresponding bit in the global interrupt mask register (CANGIM) is "1" (interrupt enabled), the CAN global interrupt (INTCANGB) will be "High".

The CANGIF register can be cleared by writing "1" to the corresponding bit in the CANGIF register. A write of "0" is invalid.

16.4.25 CANGIM (Global Interrupt Mask Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RFPFM	WUIM	RMLIM	TRMABF	TSOIM	BOIM	EPIM	WLIM
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read : Read as "0". Write : Write as "0".
7	RFPM	R/W	Remote frame pending interrupt mask 0:Interrupt disable 1:Interrupt enable
6	WUIM	R/W	Walk-up interrupt mask 0:Interrupt disable 1:Interrupt enable
5	RMLIM	R/W	Receive message lost interrupt mask 0:Interrupt disable 1:Interrupt enable
4	TRMABF	R/W	Transmit abort interrupt mask 0:Interrupt disable 1:Interrupt enable
3	TSOIM	R/W	Time stamp counter overflow interrupt mask 0:Interrupt disable 1:Interrupt enable
2	BOIM	R/W	Bus off interrupt mask 0:Interrupt disable 1:Interrupt enable
1	EPIM	R/W	Error passive interrupt mask 0:Interrupt disable 1:Interrupt enable
0	WLIM	R/W	Warning level interrupt mask 0:Interrupt disable 1:Interrupt enable

The global interrupt mask register (CANGIM) controls whether to enable or disable a global interrupt correspondingly to each interrupt condition of the CANGIF register. When the bit in the CANGIF register is "0", the corresponding CAN global interrupt (INTCANGB) is disabled. When the bit in the CANGIF register is "1", the corresponding CAN global interrupt (INTCANGB) is enabled.

Reset operation clears all bits in the CANGIM register to "0", disabling global interrupts.

16.4.26 CANMBIM (Mailbox Interrupt Mask Register)

	31	30	29	28	27	26	25	24
bit symbol	MBIM31	MBIM30	MBIM29	MBIM28	MBIM27	MBIM26	MBIM25	MBIM24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	MBIM23	MBIM22	MBIM21	MBIM20	MBIM19	MBIM18	MBIM17	MBIM16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	MBIM15	MBIM14	MBIM13	MBIM12	MBIM11	MBIM10	MBIM9	MBIM8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MBIM7	MBIM6	MBIM5	MBIM4	MBIM3	MBIM2	MBIM1	MBIM0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	MBIM31 to MBIM0	R/W	Mailbox interrupt mask 0:Interrupt disabled for corresponding mailbox 1:Interrupt enabled for corresponding mailbox

The settings in CANMBIM determine, for which mailbox the interrupt generation is enabled or disabled. If a bit in CANMBIM is "0", the interrupt generation for the corresponding mailbox is disabled and if it is "1", the interrupt generation is enabled. Reset value of CANMBIM is "0".

16.4.27 CANMBTIF (Mailbox Transmit Interrupt Flag Register)

	31	30	29	28	27	26	25	24
bit symbol	-	MBTIF30	MBTIF29	MBTIF28	MBTIF27	MBTIF26	MBTIF25	MBTIF24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	MBTIF23	MBTIF22	MBTIF21	MBTIF20	MBTIF19	MBTIF18	MBTIF17	MBTIF16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	MBTIF15	MBTIF14	MBTIF13	MBTIF12	MBTIF11	MBTIF10	MBTIF9	MBTIF8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MBTIF7	MBTIF6	MBTIF5	MBTIF4	MBTIF3	MBTIF2	MBTIF1	MBTIF0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	−	R	Read : Read as "0". Write : Write as "0".
30-0	MBTIF30 to MBTIF0	R/W	Mailbox transmit interrupt flag (Each bit corresponds with mailboxes 30 to 0.) When the message in mailbox x has been successfully transmitted and the interrupt mask of the CAN-MBIM register is enabled (<MBIMx>="1"), the <MBTIFx> bit is set to "1" and the transmit completion interrupt (INTCANTX) becomes the "High" level. When CANMBIM<MBIMx> bit is "0", the <MBTIFx> bit is not set and INTCANTX stays at the "Low" level. Transmission completion is checked by reading the CANTA register. If even one bit in the CANMBTIF register is "1", INTCANTX is the "High" level. The <MBTIFx> bit is cleared by a write of "1" to the <MBTIFx> bit from the CPU. A write of "0" is invalid.

When the mailbox is set to receive, the corresponding bit in the CANMBTIF register is read as "0". When the mailbox is set to transmit, the corresponding bit in the CANMBRIF register is read as "0".

16.4.28 CANMBRIF (Mailbox Receive Interrupt Flag Register)

	31	30	29	28	27	26	25	24
bit symbol	MBRIF31	MBRIF30	MBRIF29	MBRIF28	MBRIF27	MBRIF26	MBRIF25	MBRIF24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	MBRIF23	MBRIF22	MBRIF21	MBRIF20	MBRIF19	MBRIF18	MBRIF17	MBRIF16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	MBRIF15	MBRIF14	MBRIF13	MBRIF12	MBRIF11	MBRIF10	MBRIF9	MBRIF8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MBRIF7	MBRIF6	MBRIF5	MBRIF4	MBRIF3	MBRIF2	MBRIF1	MBRIF0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	MBRIF31 to MBRIF0	R/W	Mailbox receive interrupt flag (Each bit corresponds with mailboxes 31 to 0.) When mailbox x has successfully received the message and the interrupt mask of the CANMBIM register is enabled (<MBIMx> = "1"), the <MBRIFx> bit is set to "1" and the receive completion interrupt (INTRX) becomes the "High" level. When the <MBIMx> bit in the MBIM register is "0", the <MBRIFx> bit is not set and INTRX stays at the "Low" level. Receive completion is checked by reading the CANRMP register. If even one bit in the CANMBRIF register is "1", INTCANRX is the "High" level. The <MBRIFx> bit is cleared by a write of "1" to the <MBRIFx> bit from the CPU. A write of "0" is invalid.

16.5   Operation explain of each circuit

16.5.1   Mailbox

The mailboxes consist of a single port RAM (accessible from the internal CAN core and the CPU). The CPU controls the CAN controller by changing the settings of the mailboxes and control registers. The settings of the mailboxes and control registers are used for such processes as reception filtering, message transmission, and interrupt processing.

To start transmission, set the transmit request bit corresponding to the mailbox to transmit messages to. After the bit has been set, all transmission procedures and error processes (when errors occur) are executed without CPU involvement. When the mailbox is set to receive, the CPU reads the mailbox data using read instructions. The user can also set it so that an interrupt will be issued to the CPU every time a message has been successfully received or transmitted

In total, 32 mailboxes are provided, each of which consists of 8 byte data, 29 bit IDs, and several control bits. The mailboxes (except mailbox 31) can be set to either transmission or reception. Mailbox 31 is the receive-only mailbox. Mailbox 31 is designed so that it can receive different message ID groups using other receive masks than mailboxes 0 to 30.

Figure 16-2 shows the configuration of the mailboxes.
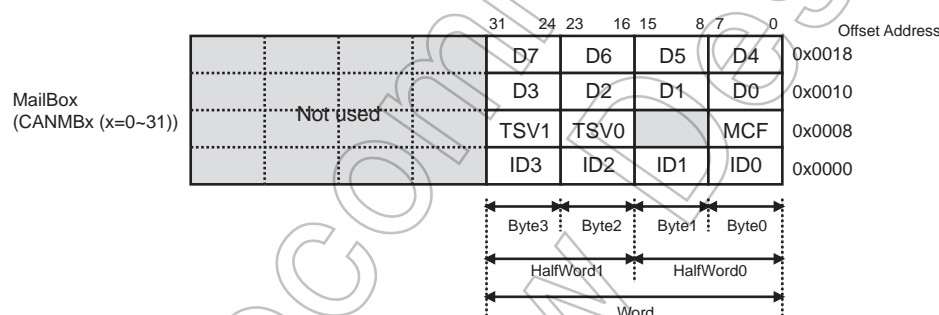
Figure 16-2 Configuration of Mailboxes

1.   Message ID field (ID3 to ID0)
   - ID extension bit <IDE>
   - Global / local acceptance mask enable bit <GAME / LAME>
   - Remote frame handling bit <RFH>
   - 29-bit message ID <ID[28:0]>

2.   Message control field (MCF)
   - Remote frame transmit request bit <RTR>
   - Data length of 4 bits <DLC[3:0]>

3.   Time stamp value (TSV1, TSV0)

      Stores time stamp counter value during receiving / transmitting message. (TSV[15:0]>

4.   Data field (D7 to D0)

      Data of 8 bytes <D7[7:0]> to <D0[7:0]>

16.5.2   Transmit Control Register

Transmission control consists of two registers. One is the transmission request set register CANTRS, and the other is the transmission request reset register CANTRR. Therefore it is possible to clear the transmission request without generating a conflict in the handling of the transmit mailboxes in the state-machine. This mechanism also prevents clearing the transmission request of a mailbox to which transmission is already in progress.

When a write of data and the ID to mailbox x configured as a transmit mailbox (CANMD<MDx>="0") is performed and access to mailbox x is enabled (CANMC<MCx>="1"), setting the CANTRS<TRSx> bit to "1" causes the messages in mailbox x to be transmitted.

If there is more than one mailbox configured as a transmit mailbox and more than one corresponding TRS bit is set, then the messages will be sent in the selected order. The order of transmission depends on the <MTOS> bit in the master control register CANMCR.

If the CANMCR<MTOS> bit is "0", the mailbox with the lower number has the higher priority. For example, if the mailboxes CANMB0, CANMB2, and CANMB5 are configured as transmit mailboxes and the corresponding CANTRS<TRSx> bits are set to "1", then the messages will be transmitted in the following order: CANMB0, CANMB2, and CANMB5. If a new transmission request is set for CANMB0 during processing of the CANMB2 message, then in the next internal arbitration-run, CANMB0 is selected for the next transmit message and transmission of the CANMB0 message starts after CANMB2 transmission is completed. This will also happen when an arbitration lost error occurs when the CANMB2 message is being transmitted. The CANMB0 message will be sent instead of the CANMB2 lost in arbitration.

If the CANMCR<MTOS> bit is "1", the mailbox with the highest priority ID among those mailboxes for which transmission is requested will be transmitted. In a transmission after an arbitration lost error occurred also, the message in the mailbox with the highest priority ID among those mailboxes for which transmission is requested at the time will be transmitted.

16.5.3   Receive Control Register

The ID of a received message is compared to the ID of the mailbox set as the receive mailbox. The comparison of the IDs depends on the <GAME> / <LAME> values of the global/local acceptance mask enable bit MBnID3 in the mailbox and the data held in the global/local acceptance mask registers GAM / LAM.

When a match is detected, the ID of the received message, the control bits, and data bytes are written in the matching mailbox. At the same time, when the corresponding receive message pending bit CANRMP<RMPx> is set to "1" and the mailbox interrupt is enabled (CANMBIM<MBIMx>="1"), the CAN receive completion interrupt INTCANRX occurs. After a match is detected, no further ID comparison takes place.

If the ID of the received message does not match with any of the mailboxes 0 to 30, the ID is compared to the ID of the receive-only mailbox 31. When a match is detected, the settings of the received message are written in receive-only mailbox 31.

If no match is detected, the received message will not be stored in the mailbox and no change occurs in the mailbox.

The <RMPx> bit must be cleared by the CPU after data is read. With the <RMPx> bit set to "1", if the next message to this mailbox x is received, the corresponding receive message lost bit <RMLx> is set to "1". In this case, mailbox x is overwritten with the new message.

Figure 16-3 shows timing when a receive message lost occurs.
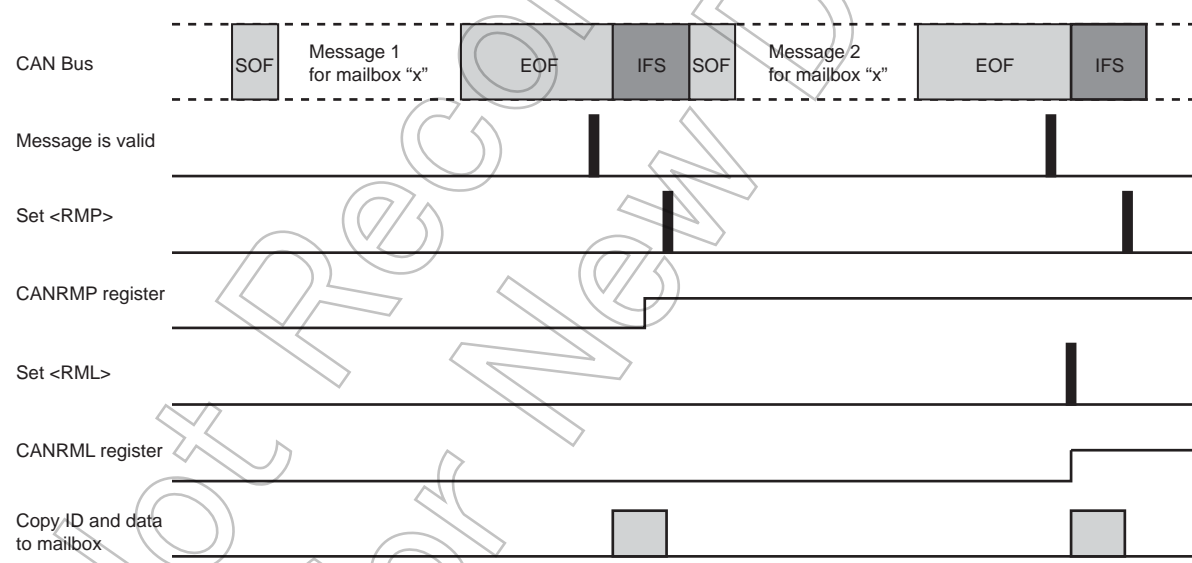
Figure 16-3 Timing when a Receive Message Lost Occurs

16.5.4   Remote Frame Control Register

After a remote frame is received, the remote frame ID is compared to the mailbox ID. The comparison of the IDs depends on the <GAME> / <LAME> values of the global / local acceptance mask enable bit CAN-MBxID in the mailbox and the data held in the global/local acceptance mask registers GAM/LAM.

After an ID match is detected, no further comparison takes place.

When the remote frame ID matches with the ID of transmit mailbox n where the remote frame handling bit CANMBxID<RFH> is set to "1", the CANTRS<TRSx> bit will be set to "1" so that the message is transmitted responding to the remote frame. For a transmit mailbox where the CANMBxID<RFH> bit is set to "0", the mailbox does not respond to the remote frame even when the ID matches.

If the ID matches with the ID of receive mailbox n, the received message is handled same as a data frame, and this sets the CANRMP<RMPx> bit and the CANRFP<RFPx> bit to "1".

When the remote frame ID matches with the ID of mailbox n where both the CANMBxID<RFH> bit and the <GAME> bit are set to "1", the ID of mailbox x is overwritten with the remote frame ID, and the mailbox will automatically respond to the remote frame using the ID (The <TRSx> bit is set to transmit a data frame). Therefore, when the global acceptance mask register CANGAM is used, one mailbox x may respond to multiple remote frame IDs depending on the mask value.

16.5.5   Receive Filtering

For mailboxes 0 to 30, the global acceptance mask register CANGAM will be used if the bit <GAME> in the mailbox is set. The receiving message will be stored in the first mailbox with a matching ID. Only if there is no matching ID in the mailboxes 0 to 30, the receiving message will be compared to the receive-only mailbox (mailbox 31). If the <LAME> bit in mailbox 31 is set, the local acceptance mask register CAN-LAM will be used.

Figure 16-4 shows receive filtering.
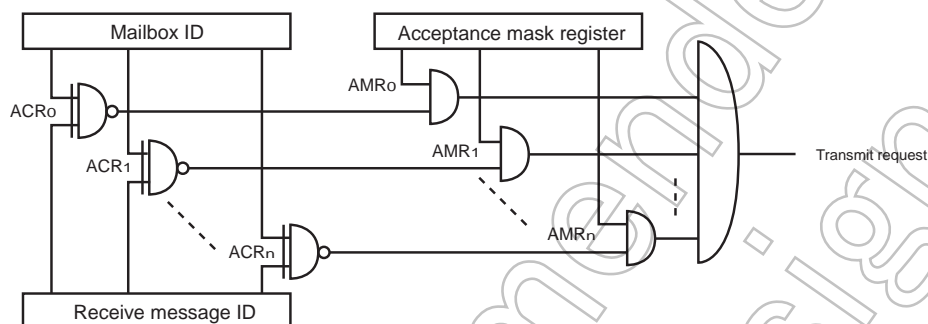
Figure 16-4 Receive filtering

16.5.6   Time Stamp Function

There is a free-running 16-bit time stamp counter (CANTSC) implemented in the CAN controller to show the time of message reception and transmission. The content of the CANTSC is written into the time stamp value (TSV) of the corresponding mailbox when a received message has been stored or a message has been transmitted.

The CANTSC is driven by the bit clock of the CAN bus line. When the operation mode of the CAN is in configuration mode or in sleep mode, the CANTSC will be stopped. After power-up reset, a write to the time stamp counter prescaler register (CANTSP) clears the CANTSC to "0". The CANTSC is readable and writable from the CPU both in configuration mode and normal operation mode.

Figure 16-5 shows the structure of the time stamp counter.
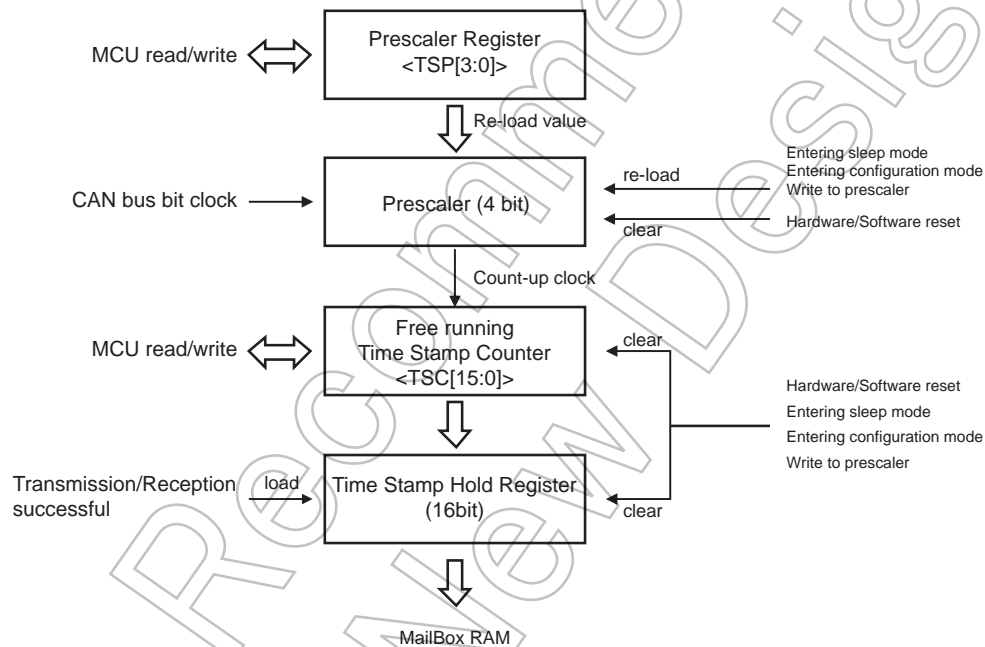
Figure 16-5 Timer Stamp Counter

The free running time stamp counter and the time stamp hold register will be cleared in the following cases:

- · After reset (Power on reset or software reset)
- · When the controller enters into configuration mode
- · When the controller enters into sleep mode
- · When a write access is performed to the CANTSP register.

16.5.7 Interrupt Control

The CAN controller has the following interrupt sources. And these interrupt sources are divided into three groups and each group has one interrupt output.

- ・ CAN transmit completion interrupt (INTCANTX)

    It occurs at the completion of transmission

- ・ CAN receive completion interrupt (INTCANRX)

    It occurs at the completion of reception

- ・ CAN Global interrupt (INTCANGB)

    It occurs by eight sources other than those above.

	Sources	Group
Transmit interrupt	:a message has been transmitted successfully.	INTCANTX
Receive interrupt	:a message has been received successfully.	INTCANRX
Warning level interrupt	: at least one of the two error counters is greater than or equal to 97.	INTCANGB
Error passive interrupt	:CAN enters the error passive mode.	
Bus off interrupt	:CAN enters the bus off mode.	
Time stamp overflow interrupt		
Transmit abort interrupt		
Receive message lost interrupt		
Walk-up interrupt	:after walk-up from sleep mode, this interrupt will be generated.	
Remote frame receive interrupt		

For mailbox interrupts, there are two interrupt output lines separated from global interrupts. These are the mailbox receive completion interrupt (INTCANRX) and the mailbox transmit completion interrupt (INTCANTX), which are dependent on mailbox settings.

There are two interrupt flag registers and one interrupt mask register. One interrupt flag register is for the mailbox receive interrupt flag register (CANMBRIF) and one for the mailbox transmit interrupt flag register (CANMBTIF). In addition, there is the mailbox interrupt mask register (CANMBIM) for setting whether to enable or disable each mailbox interrupt. The CANMBIM register is used both for transmit and receive mailboxes.

Figure 16-6 shows the block diagram of CAN interrupt signal.
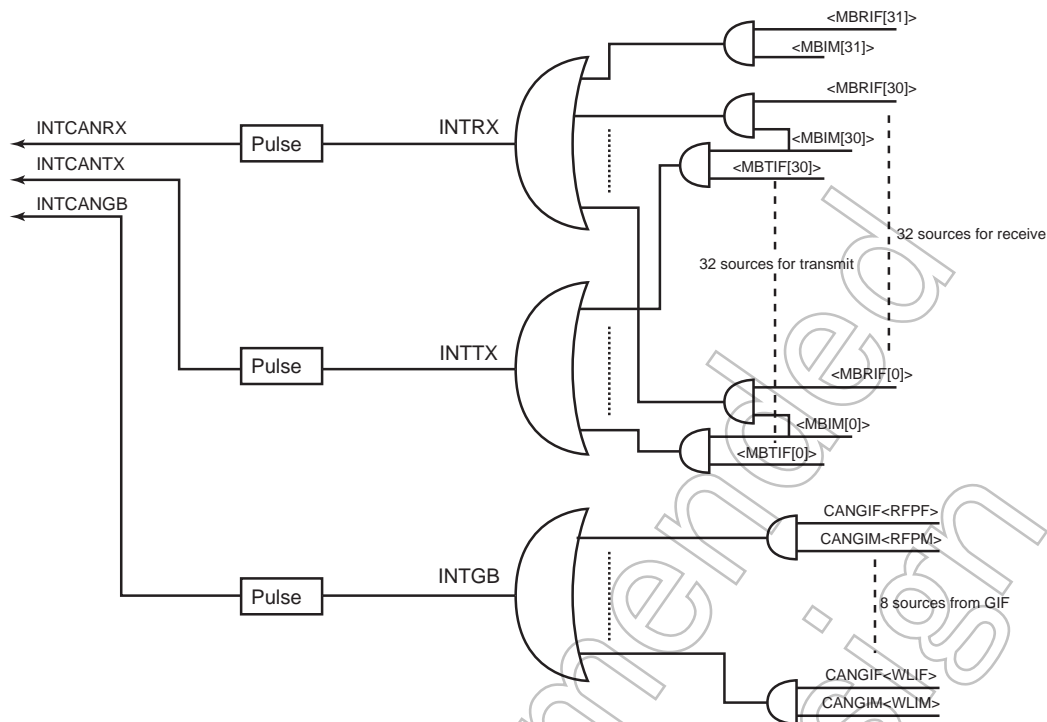
Figure 16-6 Block Diagram of CAN interrupt signals

The CAN receive completion interrupt signal INTRX is the OR of the signal for which the 32 sources issued by the mailbox receive interrupt flag register CANMBRIF that are ANDed with each bit of the mailbox interrupt mask register CANMBIM.

The CAN transmit completion interrupt signal INTTX is the OR of the signal for which the 31 sources issued by the mailbox transmit interrupt flag register CANMBTIF that are ANDed with each bit of the mailbox interrupt mask register CANMBIM.

The CAN global interrupt signal INTGB is the OR of the signal for which the 8 sources issued by the global interrupt flag register CANGIF that are ANDed with each bit of the global interrupt mask register CANGIM.

16.6 Operation Mode

16.6.1 Configuration Mode

The CAN controller needs initial setup before starting operation (setting of the bit configuration registers, CANBCR1 and CANBCR2). Writes to the CANBCR1 and CANBCR2 are possible only when the CAN controller is in configuration mode.

After reset, the CANMCR<CCR> and the CANGSR<CCE> are set to "1" and the configuration mode is set. A write of "0" to the <CCR> bit sets the CAN controller to normal operation mode. After leaving configuration mode, the <CCE> bit is cleared to "0" and the power-up sequence starts. The power-up sequence detects 11 consecutive recessive bits on the CAN bus line. After detection, the CAN controller is bus on and ready for operation.

A write of "1" to the <CCR> bit sets the CAN controller to enter configuration mode from normal operation mode. After the CAN controller has entered configuration mode, the <CCE> bit is set to "1".

Figure 16-7 shows the flowchart of the initial setup of the CAN controller.

When the CAN controller enters into configuration mode, the CAN error counter (CANCEC), the time stamp counter (CANTSC), and the time stamp hold registers will be cleared.
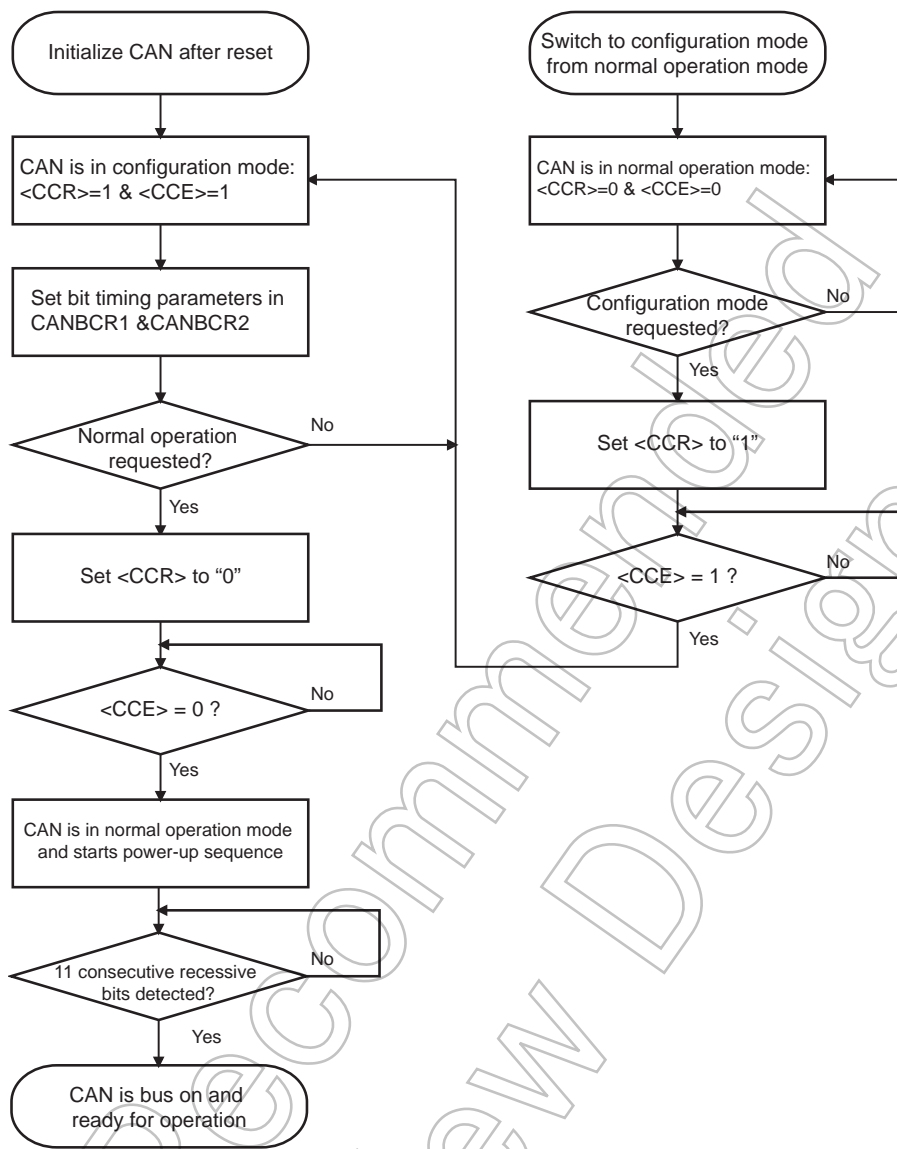
Figure 16-7 Flowchart of Initial Setup of CAN Controller

16.6.2　Sleep Mode

Sleep mode is requested by a write of "1" to the <SMR> bit in the CANMCR register. After the CAN controller has entered into sleep mode, the CANGSR<SMA> bit is set to "1".

The read value of the CANGSR register is 0xF040. This means that there is no message in the transmit buffer and sleep mode is active where the <SMA> bit is "1". Read values to all other registers deliver the value 0x0000. Write accesses to all registers, except the CANMCR register, will be denied.

The CAN controller cancels sleep mode (wakes up) and starts the power-up sequence if a write access to the CANMCR register is detected, or there is any bus activity detected on the CAN bus with the CANMCR <WUBA> bit set to "1". The CAN controller waits until detecting 11 consecutive recessive bits on the CANRX input terminal, after it goes into bus active state. The walk-up message is invalid.

In sleep mode, the CAN error counters and all transmission request set CANTRS<TRSx> bits and transmission request reset CANTRR<TRRx> bits are cleared. The <SMR> bit and the <SMA> bit are cleared after the CAN controller leaves sleep mode.

If sleep mode is requested while the CAN controller is transmitting a message (CANMCR<SMR>="1"), the CAN controller enters sleep mode after any of the following occurs:

- ・　The message has been successfully transmitted.
- ・　The message has been successfully transmitted after an arbitration lost error.
- ・　The message has been successfully received after an arbitration lost error.

16.6.3　Suspend Mode

The suspend mode is requested by writing "1" to the CANMCR<SUR> bit. If the CAN bus line is not idle, the current message transmission/reception is completed before suspend mode is activated. After the CAN controller has entered suspend mode, the CANGSR<SUA> bit is set to "1".

In suspend mode, the CAN controller is not active on the CAN bus line. That means neither error frames nor acknowledgement will be sent. The error counters and the CANGSR<EP> bit will not be cleared either.

If suspend mode is requested during the bus off recovery sequence execution, the CAN controller enters suspend mode after the bus off recovery sequence is finished.

To restart the CAN controller, the <SUR> bit needs to be programmed to "0". After leaving the bus off state or the inactive state, the CAN controller restarts the bus off recovery sequence.

The CAN controller cancels suspend mode with a write of "0" to the <SUR> bit.

16.6.4 Test Loop Back Mode

In test loop back mode, the CAN controller can receive its own transmitted message and generates its own acknowledge bit. No other CAN node is necessary for the operation.

The test loop back mode can be enabled or disabled only when the CAN controller is in suspend mode. In test loop back mode, the CAN controller can transmit a message from a mailbox and receive it in another mailbox. The setup for mailboxes is the same as in normal operation mode.

16.6.5 Test Error Mode

In test error mode, writes to the CAN error counter register (CANCEC) are possible. The values of the lower 8 bits are concurrently written to both the transmit error counter (CANTEC) and the receive error counter (CANREC). The maximum value that can be written into the error counters is 255. The error counter value of 256 which forces the CAN controller into bus off mode can not be written.

The test error mode can be enabled or disabled only when the CAN controller is in suspend mode.

Figure 16-8 shows the flowchart of the setup of test loop back mode and test error mode.
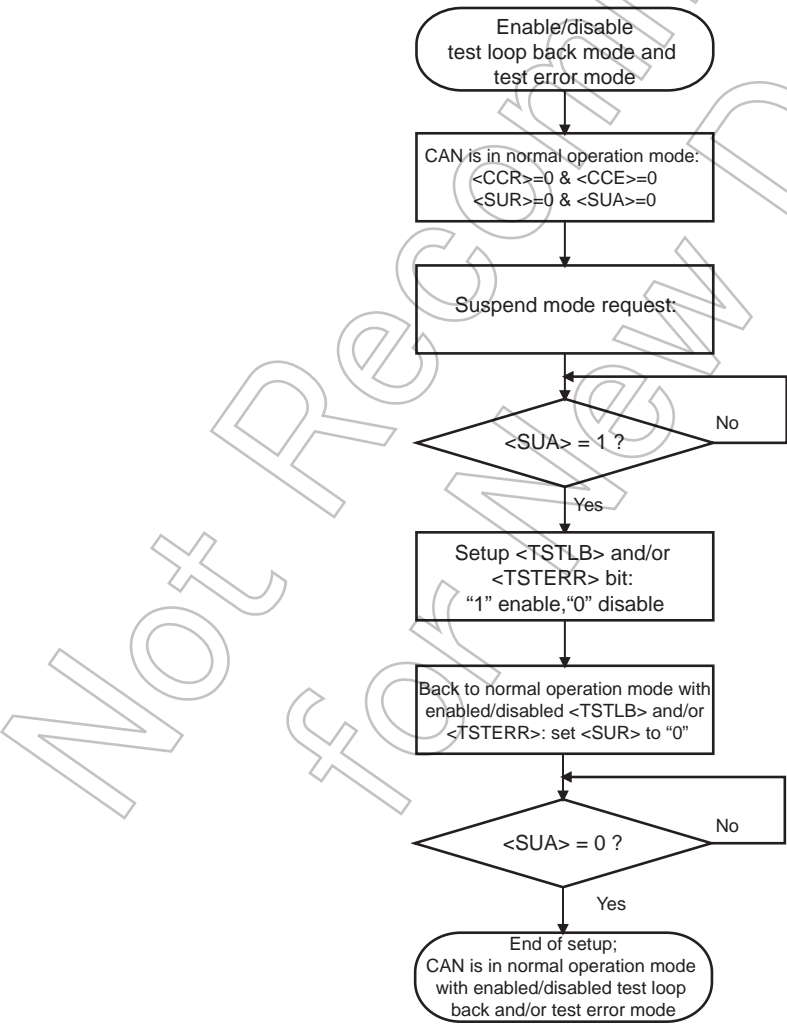
Figure 16-8 Flowchart of Setup of Test Loop Back Mode and Test Error Mode

16.7   Description of Operation

16.7.1   Receive Messages

Figure 16-9 shows an example flowchart of message reception using the CAN receive completion interrupt (INTCANRX).
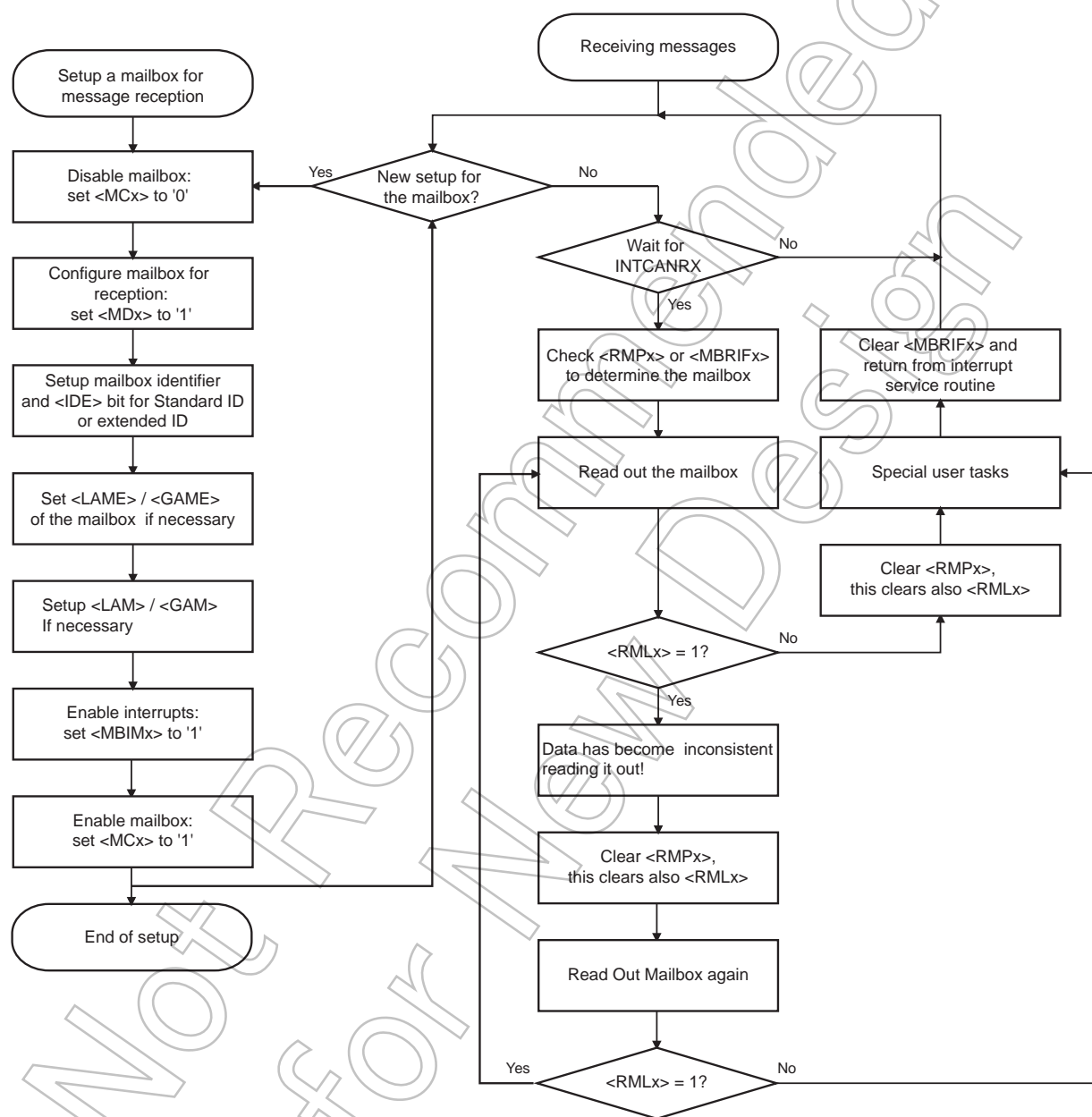
Figure 16-9 Flowchart of Message Reception

It is also possible to use polling instead of receive interrupts. In this case, the "waiting for INTCANRX" in above flowchart must be replaced by polling CANRMP. Further, enabling interrupts and clearing CANMBRIF must be removed from the flow.

16.7.2 Transmitting Message

Figure 16-10 shows an example flowchart of message transmission using the CAN transmit completion interrupt (INTCANTX).
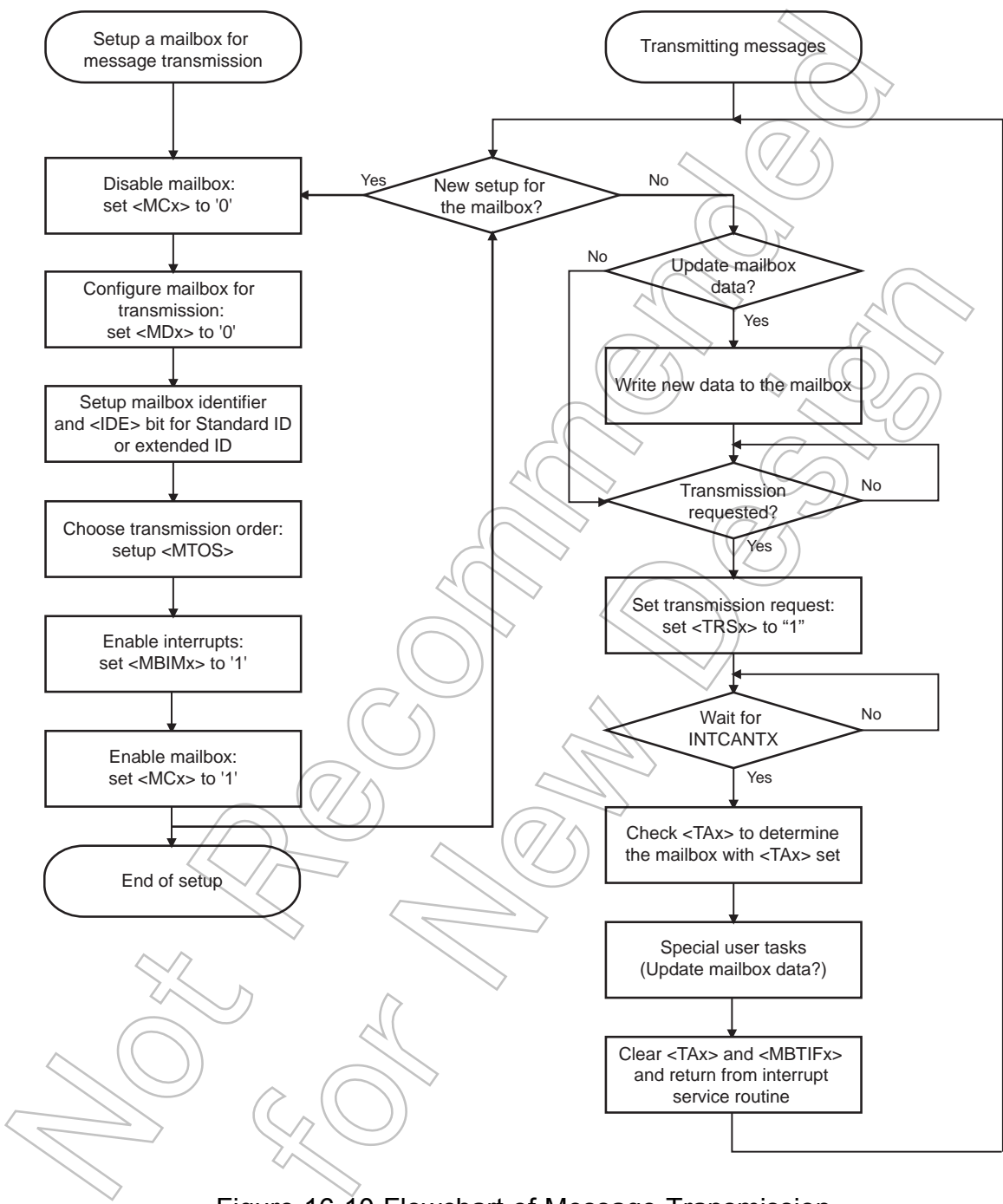
Figure 16-10 Flowchart of Message Transmission

It is also possible to use polling instead of transmit interrupts. In this case, the "waiting for INTCANTX" in above flowchart must be replaced by polling TA. Further, enabling interrupts and clearing CANMBTIF must be removed from the flow.

16.7.3   Remote Frame Handling

Figure 16-11 shows an example flowchart of remote frame handling by using the automatic reply feature. This feature is available when the <RFH> bit of the transmit mailbox is set to "1". To avoid data inconsistency when updating the mailbox data, the CANCDR register controls transmission during data update of the mailbox.
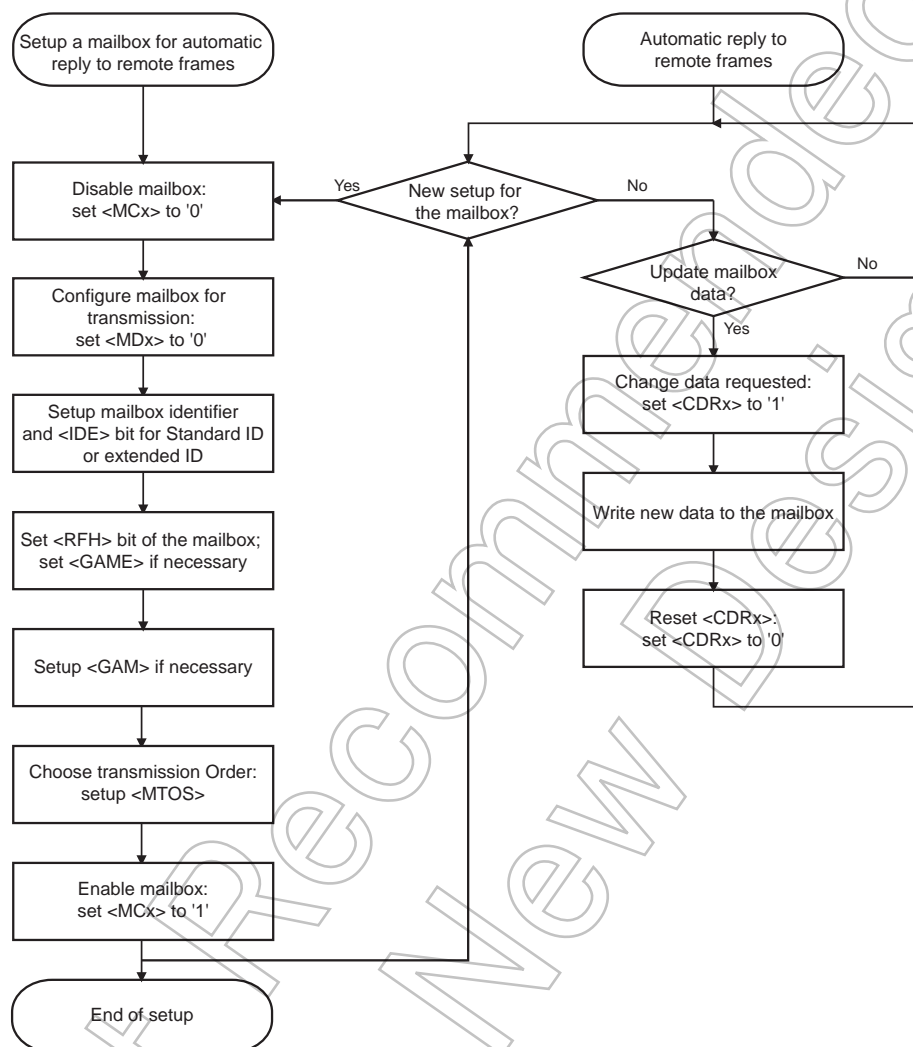
Figure 16-11 Flowchart of Remote Frame Handling Using the Automatic Reply Feature

16.8   Bit Configuration

The length of a bit is determined by the parameters TSEG1, TSEG2, and BRP. All controllers on the CAN bus must have the same baud rate and bit length. At different clock frequencies of the individual controllers, the baud rate has to be adjusted by the above-mentioned parameters. In the bit timing logic, the conversion of the parameters to the required bit timing is implemented. The configuration registers CANBCR1 and CANBCR2 contain the data about bit timing. Its definition corresponds to the CAN specification 2 (equivalent to Intel 82527).

Figure 16-12 shows CAN bit timing.
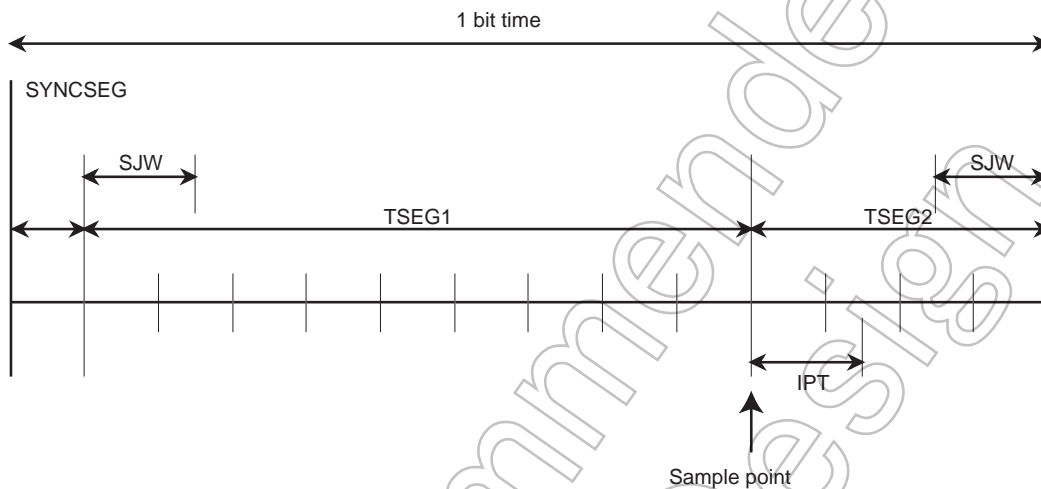
Figure 16-12 CAN Bit Timing

T_{SCL} (CAN system clock) is defined by :

$$T_{SCL} = \frac{<BRP[9:0]>+1}{f_{CANOSC}}$$

$1 \times T_{SCL} = 1 \times T_Q$ (T_Q : time quantum)

f_{CANOSC} is the clock for CAN baud rate generation. The clock obtained by dividing the system clock f_{SYS} by 4 is supplied as the clock for CAN baud rate generation. If f_{SYS} = 48MHz then f_{CANOSC} = 12MHz.

The synchronization segment SYNCSEG always has the length of one T_Q.

The baud rate is defined by :

$$BR = \frac{1}{((<TSEG1[13:0]>+1)+(<TSEG2[2:0]>+1)+1) \times T_{SCL}}$$

Note:<TSEG1[3:0]> and <TSEG2[2:0]> are values of the CANBCR2 register. It is not $T_Q$unit value.

Information processing time (IPT) is the time segment starting with the sample point reserved for processing of the sampled bit level. The information processing time is equal to three CAN system clock cycles.

<SJW[1:0]> indicates how much the time quantum (T_Q) value in bit length is allowed to be lengthened or shortened when resynchronizing. Values between "1" (<SJW[1:0]> 00) and "4" (<SJW[1:0]> 11) are adjustable. The bus line is sampled and synchronization is performed at each falling edge of the bus signal within a bit grid. For <SJW[1:0]>, set a value equal to or smaller than <TSEG2[2:0]>.

Setting the <SAM> bit enables the multiple sampling of the bus line. The level is determined by the result from the majority decision of three sampling values. Sampling is taken at the sample point and the previous last two CAN system clock points. When <BRP[9:0]> is smaller than 4, the sampling performed is always once regardless of the value set in the <SAM> bit.

Table 16-2 shows the restrictions when the baud rate is set.

Table 16-2 Restrictions when Setting the Baud Rate

<BRP[9:0]>	T_Q length (number of CAN clock cycles)	IPT length (number of CAN clock cycles)	Minimum TSEG2 length (T_Q unit)
0	1	3	3
1	2	3	2
> 1	<BRP[9:0]>+1	3	2

- Restrictions for TSEG1

    TSEG1 ≥ TSEG2 : The length of TSEG1 should be equal to or greater than the length of TSEG2.

- Restrictions for SJW

    SJW ≤ TSEG2 : For the synchronization jump width, set a value equal to or smaller than TSEG2.

- Restrictions for SAM

    The three-time sampling is not allowed under the condition that <BRP[9:0]> is smaller than 4. For the condition that <BRP[9:0]> < 4, a one-time sampling will always be performed regardless of the value of SAM.

Example : For 500 Kbit/s

A bit has a length of 2μs. If f_{OSC} = 12 MHz,. the baud rate prescaler is set to "1". That means a bit for this data transmission rate has to be programmed with a length of $12T_Q$. According to the above formula, the values to be programmed always are smaller by one than the calculated values:

<BRP[9:0]> = 0y00_0000_0001

<TSEG1[3:0]> = 0y0110 (7 TQ)

<TSEG2[2:0]> = 0y011 (4 TQ)

In this case, the sample point is 8/12 66%.

Other combinations for TSEG1 / TSEG2 are possible; with TSEG2 3 the full range for SJW.

SJW should always be set to the highest value possible. SJW is not allowed to be greater than TSEG2.

The three-time sampling of the bus cannot be set because of the condition that <BRP[9:0]> is smaller than 4. Thus, SAM="0" should be set.

17. Remote control signal preprocessor(RMC)

17.1 Basic operation

Remote control signal preprocessor (hereafter referred to as RMC) receives a remote control signal of which carrier is removed.

17.1.1 Reception of Remote Control Signal

- A sampling clock can be selected from either low frequency clock (32.768kHz) or Timer output.
- Noise canceling time can be adjusted.
- Leader detection
- Batch reception up to 72bit of data

17.2 Block Diagram

Figure 17-1 shows the block diagram of RMC.
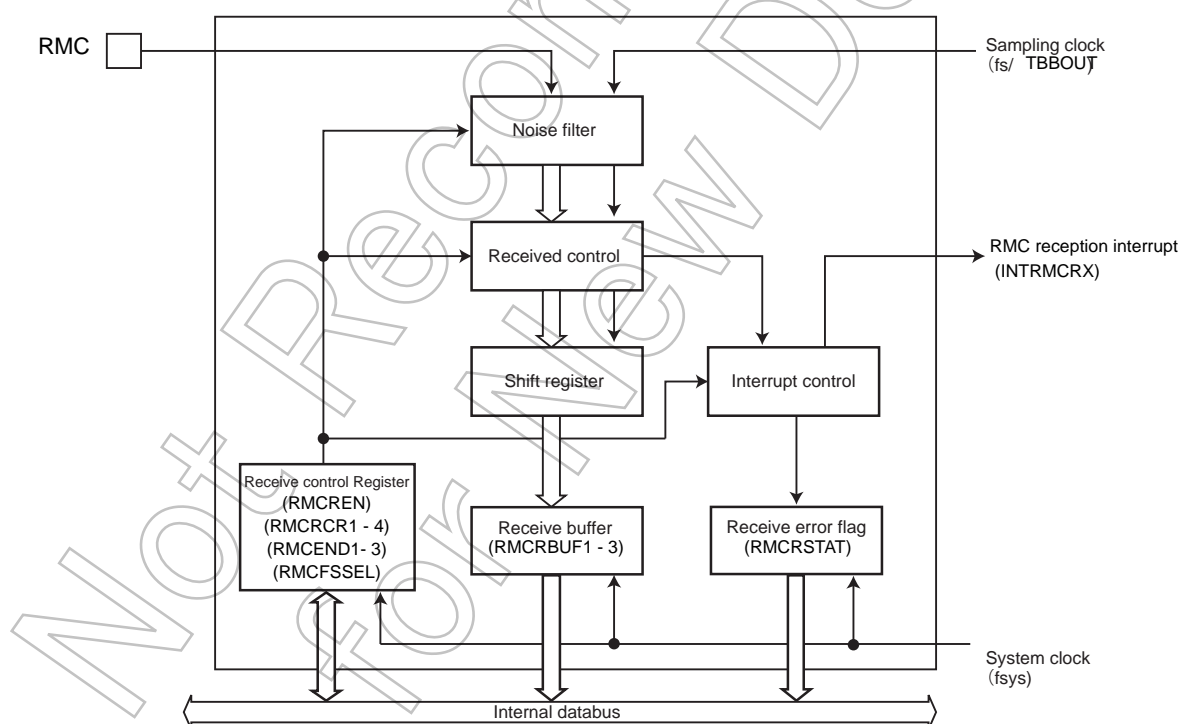
Figure 17-1 Block diagram of RMC

17.3 Registers

17.3.1 Register List

Addresses and names of RMC control registers are shown below.

Base Address = 0x400E_3000

Register		Address(Base+)
Enable Register	RMCEN	0x0000
Receive Enable Register	RMCREN	0x0004
Receive Data Buffer Register 1	RMCRBUF1	0x0008
Receive Data Buffer Register 2	RMCRBUF2	0x000C
Receive Data Buffer Register 3	RMCRBUF3	0x0010
Receive Control Register 1	RMCRCR1	0x0014
Receive Control Register 2	RMCRCR2	0x0018
Receive Control Register 3	RMCRCR3	0x001C
Receive Control Register 4	RMCRCR4	0x0020
Receive Status Register	RMCRSTAT	0x0024
Receive End bit Number Register 1	RMCEND1	0x0028
Receive End bit Number Register 2	RMCEND2	0x002C
Receive End bit Number Register 3	RMCEND3	0x0030
Source Clock selection Register	RMCFSSEL	0x0034

17.3.2   RMCEN(Enable Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	RMCEN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as 0.
1	−	R/W	Write as "1".
0	RMCEN	R/W	Controls RMC operation. 0: Disabled 1:Enabled To allow RMC to function, enable the RMCEN bit first. If the operation is disabled, all the clocks for RMC except for the enable register are stopped, and it can reduce power consumption. If RMC is enabled and then disabled, the settings in each register remain intact.

17.3.3 RMCREN(Receive Enable Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	RMCREN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	−	R	Read as 0.
0	RMCREN	R/W	Reception 0: Disabled 1: Enabled Controls reception of RMC. Setting this bit to "1" enables reception.

Note: Enable the <RMCREN> bit after setting the RMCRCR1, RMCRCR2, and RMCRCR3.

17.3.4   RMCRBUF1(Receive Data Buffer Register 1)

	31	30	29	28	27	26	25	24
bit symbol	RMCRBUF(Received data 31 to 24 bit)							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	RMCRBUF(Received data 23 to 16 bit)							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	RMCRBUF(Received data 15 to 8bit)							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RMCRBUF(Received data 7 to 0 bit)							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	RMCRBUF[31:0]	R	Received data (31 to 0 bit) Reads 4 bytes of received data. (31 to 0 bit)

17.3.5   RMCRBUF2(Receive Data Buffer Register 2)

	31	30	29	28	27	26	25	24
bit symbol	RMCRBUF(Received data 63 to 54 bit)							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	RMCRBUF(Received data 55 to 48 bit)							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	RMCRBUF(Received data 47 to 40 bit)							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RMCRBUF(Received data 39 to 32 bit)							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-0	RMCRBUF[63:32]	R	Received data (63 to 32 bit) Reads 4 bytes of received data. (63 to 32 bit)

17.3.6   RMCRBUF3(Receive Data Buffer Register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RMCRBUF(Received data 71 to 64 bit)							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-0	RMCRBUF[71:64]	R	Received data (71 to 64 bit). Reads 1 byte of received data. (71 to 64 bit).

Note: The received bit is stored in the data buffer register in MSB-first order, and the last received bit is stored in the LSB (bit 0). If the remote control signal is received in the LSB first algorithm, the received data is stored in reverse sequence.

17.3.7   RMCRCR1(Receive Control Register 1)

	31	30	29	28	27	26	25	24
bit symbol	RMCLCMAX							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	RMCLCMIN							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	RMCLLMAX							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RMCLLMIN							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-24	RMCLCMAX[7:0]	R/W	Specifies a maximum cycle of leader detection. Calculating formula of the maximum cycle: <RMCLCMAX> × 4/fs [s].
23-16	RMCLCMIN[7:0]	R/W	Specifies a minimum cycle of leader detection. Calculating formula of the minimum cycle: <RMCLCMIN> × 4/fs [s].
15-8	RMCLLMAX[7:0]	R/W	Specifies a maximum low width of leader detection. Calculating formula of the maximum low width: <RMCLLMAX> × 4/fs [s]
7-0	RMCLLMIN[7:0]	R/W	Specifies a minimum low width of leader detection. Calculating formula for the minimum low width: <RMCLLMIN> × 4/fs [s] When RMCRCR2<RMCLD> = 1, a value of the low-pulse width is less than the specified value, it is defined as data bit.

Note:When you configure the register, you must follow the rule shown below.

Leader	Rules
Low width + High width	<RMCLCMAX[7:0]> > <RMCLCMIN[7:0]> <RMCLLMAX[7:0]> > <RMCLLMIN[7:0]> <RMCLCMIN[7:0]> > <RMCLLMAX[7:0]>
Only high width	<RMCLCMAX[7:0]> > <RMCLCMIN[7:0]> <RMCLLMAX[7:0]> = 0x00 <RMCLLMIN[7:0]> = don't care
No Leader	<RMCLCMAX[7:0]> = 0x00 <RMCLCMIN[7:0]> = don't care <RMCLLMAX[7:0]> = don't care <RMCLLMIN[[7:0]> = don't care

## 17.3.8	RMCRCR2(Receive Control Register 2)

	31	30	29	28	27	26	25	24
bit symbol	RMCLIEN	RMCEDIEN	-	-	-	-	RMCLD	RMCPHM
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	RMCLL							
After reset	1	1	1	1	1	1	1	1
	7	6	5	4	3	2	1	0
bit symbol	RMCDMAX							
After reset	1	1	1	1	1	1	1	1

Bit	Bit Symbol	Type	Function
31	RMCLIEN	R/W	Leader detection interrupt 0: Not generated 1: Generated
30	RMCEDIEN	R/W	Remote control input falling edge interrupt 0: Not generated 1: Generated
29-26	−	R	Read as 0.
25	RMCLD	R/W	Receiving remote control signal with or without leader 0: Disabled 1: Enabled
24	RMCPHM	R/W	Receiving a remote control signal by a phase modulation 0: Not receiving a remote control signal by a phase modulation. (receive by a cycle modulation) 1: Receive remote control signal by a fixed-frequency pulse modulation. To receive a fixed-frequency remote control signal by a pulse modulation, set this bit to "1".
23-16	−	R	Read as 0.
15-8	RMCLL[7:0]	R/W	Excess low width that triggers reception completion and interrupt generation. 0000_0000 to 1111_1110: Reception completion and interrupt generation at <RMCLL> × 1/fs [s]. 1111_1111: not to use as the trigger
7-0	RMCDMAX[7:0]	R/W	Maximum data bit cycle that triggers reception completion and interrupt generation. 0000_0000 to 1111_1110: Reception completion and interrupt generation at <RMCDMAX> × 1/fs [s]. 1111_1111: not to use as the trigger

17.3.9   RMCRCR3(Receive Control Register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	RMCDATH						
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	RMCDATL						
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-15	-	R	Read as 0.
14-8	RMCDATH[6:0]	R/W	Larger threshold to determine a signal pattern in a phase method Calculating formula of the threshold: <RMCDATH> × 1/fs [s] Specifies a larger threshold (within a range of 1.5T and 2T) to determine a pattern of remote control signal in a phase method. If the measured cycle exceeds the threshold, the bit is determined as "10". If not, the bit is determined as "01".
7	-	R	Read as 0.
6-0	RMCDATL[6:0]	R/W	Threshold to determine 0 or 1 smaller threshold to determine a signal pattern in a phase method. Calculating formula of the threshold: <RMCDATL> × 1/fs [s] Specifies two kinds of thresholds: a threshold to determine whether a data bit is 0 or 1; a smaller threshold (within a range of 1T and 1.5T) to determine a pattern of remote control signal in a phase method. As for the determination of data bit, if the measured cycle exceeds the threshold, the bit is determined as "1". If not, the bit is determined as "0". Calculating formula of the threshold: <RMCDATL> × 1/fs [s]. As for the determination of a remote control signal pattern in a phase method, if the measured cycle exceeds the threshold, the bit is determined as "01". If not, the bit is determined as "00".

Note: If the <RMCPHM> bit of the Receive Control Register 2 is "0", <RMCDATH[6:0]> are not enabled.
The bits are enabled when <RMCPHM> is "1".

17.3.10 RMCRCR4(Receive Control Register 4)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RMCPO	-	-	-	RMCNC			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	RMCPO	R/W	Remote control input signal 0: Not reversed 1: Reversed
6-4	−	R	Read as 0.
3-0	RMCNC[3:0]	R/W	Specifies noise cancellation time. 0000: No cancellation 0001 to 1111: cancellation Calculating formula of noise cancellation time: <RMCNC> × 1/fs [s]

17.3.11   RMCRSTAT(Receive Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	RMCRLIF	RMCLOIF	RMCDMAXIF	RMCEDIF	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RMCRLDR	RMCRNUM						
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15	RMCRLIF	R	Interrupt source flag 0: No leader detection interrupt generated. 1: Leader detection interrupt generated.
14	RMCLOIF	R	Interrupt source flag 0: No low width detection interrupt generated. 1: Low width detection interrupt generated.
13	RMCDMAXIF	R	Interrupt source flag 0: No maximum data bit cycle interrupt generated. 1: Maximum data bit cycle interrupt generated.
12	RMCEDIF	R	Interrupt source flag 0: No falling edge interrupt generated. 1: Falling edge interrupt generated.
11-8	−	R	Read as 0.
7	RMCRLDR	R	Leader detection. 0: Disable leader detection. 1: Enable leader detection.
6-0	RMCRNUM[6:0]	R	The number of received data bit 000_0000:no data bit (only with leader) 000_0001 to 100_1000: 1 to 72bit 100_1001 to 111_1111: 73bit and more Indicates the number of bits received as remote control signal data. The number cannot be monitored during reception. On completion of reception, the number is stored.

Note 1: This register is updated every time an interrupt is generated.Writing to this register is ignored.

Note 2: RMC keeps receiving 73 bit or more data unless reception is completed by detecting the maximum data bit cycle or the excess low width. In this case, the received data in the data buffer may not be ensured.

17.3.12 RMCEND1(Receive End bit Number Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	RMCEND1						
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	-	R	Read as 0.
6-0	RMCEND1[6:0]	R/W	Specifies that the number of receive data bit 000_0000 : No specifically the receive data bit 000_0001 to 100_1000 : Specifies that the number of receive data bit(1 to 72bit) 100_1001 to 111_1111 : Don't set the value

17.3.13 RMCEND2(Receive End bit Number Register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	RMCEND2						
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	-	R	Read as 0.
6-0	RMCEND2[6:0]	R/W	Specifies that the number of receive data bit 000_0000 : No specifically the receive data bit 000_0001 to 100_1000 : Specifies that the number of receive data bit(1 to 72bit) 100_1001 to 111_1111 : Don't set the value

17.3.14   RMCEND3(Receive End bit Number Register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	RMCEND3						
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	-	R	Read as 0.
6-0	RMCEND3[6:0]	R/W	Specifies the number of receive data bit 000_0000 : No specifically the receive data bit 000_0001 to 100_1000 : Specifies that the number of receive data bit(1 to 72bit) 100_1001 to 111_1111 : Don't set the value

Note 1: As specified to RMCEND1, RMCEND2 and RMCEND3, it is able to set three kinds of the receive data bit.

Note 2: To use the RMCEND1, RMCEND2 and RMCEND3 is in combination with the maximum data bit cycle.

17.3.15   RMCFSSEL(Source Clock selection Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	RMCCLK
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	-	R	Read as 0.
0	RMCCLK	R/W	Specifies that Sampling clock of RMC function 0 : Low frequency Clock (32.768kHz) 1 : Timer output(TBBOUT) For the Sampling of RMC function, It is able to set the Low Frequency Clock (32.768kHz) or Timer output (TBBOUT). The Setting range of Timer output by TBBOUT is from 30 to 34kHz.

Note: To Change the sampling clock by using the RMCFSSEL, disable the RMC operation first by using the RMCEN<RMCEN>. Then, enable it again, and set the RMCFSSEL before setting other RMC registers.

17.4 Operation Description

17.4.1 Reception of Remote Control Signal

17.4.1.1 Sampling clock

A remote control signal is sampled by using low-speed 32.768kHz clock (fs).

17.4.1.2 Basic operation

RMC set RMCRSTAT<RMCRLDR> bit when a leader is detected.

At this time, if you set the RMCRCR2<RMCLIEN> bit, leader detection will generate a leader detection interrupt. When a leader detection interrupt occurs, RMCRSTAT<RMCRLIF> bit is set.

After the leader detecting, each data bit is determined as "0" or "1" in sequence. The results are stored in RMCRBUF1, RMCRBUF2 and RMCRBUF3 registers up to 72 bits. By setting RMCRCR2< RMCEDIEN> bit, a remote control signal input falling edge interrupt can be generated in each falling edge of data bit. When a remote control signal input falling edge interrupt is generated, RMCRSTAT< RMCEDIF > bit is set.

Data reception stops when the maximum data bit cycle is detected annd low-width matches the setting value, and then, an interrupt occurs. If <RMCEND1>, <RMCEND2> nad <RMCEND3> of the register RMCxEND1, RMCxEND2 and RMCEND3 have been configured, data reception stops and an interrupt occurs only in the case that the number of bits received before maximum data bit cycle is detected. The condition of RMC can be checked by reading the remote control receive status register.

To check the status of RMC if reception is completed, read the remote control receive status register.

On completion of reception, RMC is waiting for the next leader.

By setting RMC to receive a signal without a leader, RMC recognizes the received as data and starts reception without detecting a leader.

If the next data reception is completed before reading the preceding received data, the preceding data is overwritten by the next one.
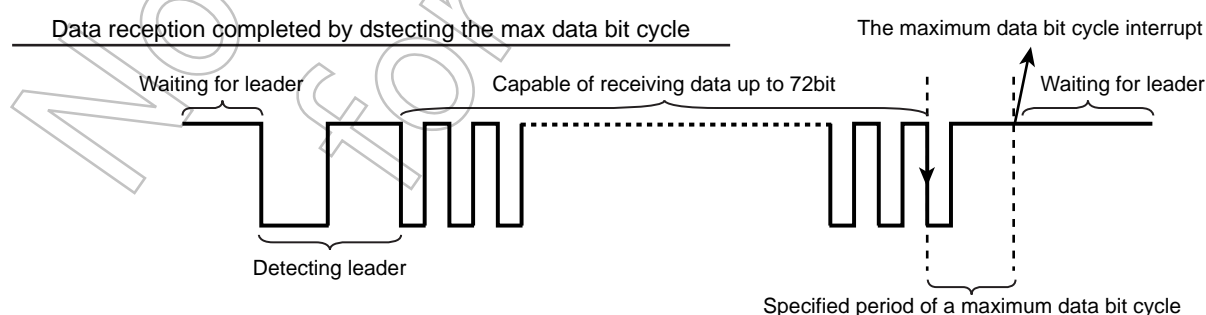
17.4.1.3   Preparation

Before starting receiving process, configure how to receive remote control signal using the Remote Control Signal Receive Control Registers (RMCRCR1, RMCRCR2 and RMCRCR3, RMCRCR4).

(1)   Settings of Noise Cancelling Time

Configure noise cancelling time with the RMCRCR4 <RMCNC[3:0]> bit.

Noise canceling is applied to remote control signals sampled by the sampling clock.

RMC monitors a sampled remote control signal in each rising edge of a sampling clock. If "High" is monitored, RMC recognizes that the signal was changed to "Low" after monitoring cycles of "Low"s specified in <RMCNC>. If "Low" is monitored, RMC recognizes that the signal was changed to "High" after monitoring cycles of "High" specified in <RMCNC>.

The following figure shows how RMC operates according to the noise cancel setting of <RMCNC [3:0]> = "0011" (3 cycles). Subsequent to noise cancellation, the signal is changed from "High" to "Low" upon monitoring 3 cycles of "Low", and the signal is changed from "Low" to "High" upon monitoring 3 cycles of "High".

<RMCNC [3:0]> = 0011 (3 cycle)
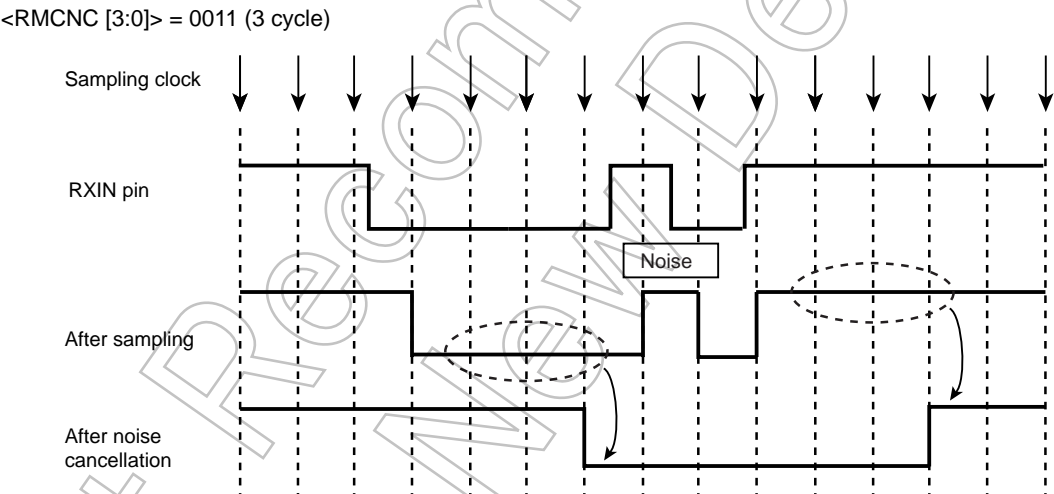
(2)   Settings of Detecting Leader

Set the leader cycle and a low width of the leader to RMCRCR1 <RMCLLMIN[7:0]>
<RMCLLMAX[7:0]> <RMCLCMIN[7:0]> <RMCLCMAX[7:0]> bits. When you configure those
above, follow the rule shown below.

Leader	Rules
Low width + High Width	<RMCLCMAX[7:0]> > <RMCLCMIN[7:0]> <RMCLLMAX[7:0]> > <RMCLLMIN[7:0]> <RMCLCMIN[7:0]> > <RMCLLMAX[7:0]>
Only high width	<RMCLCMAX[7:0]> > <RMCLCMIN[7:0]> <RMCLLMAX[7:0]> = 0y00000000 <RMCLLMIN[7:0]> = don't care
No leader	<RMCLCMAX[7:0]> = 0y00000000 <RMCLCMIN[7:0]> = don't care <RMCLLMAX[7:0]> = don't care <RMCLLMIN[7:0]> = don't care

The following shows a leader waveform and the RMCRCR1 register settings.
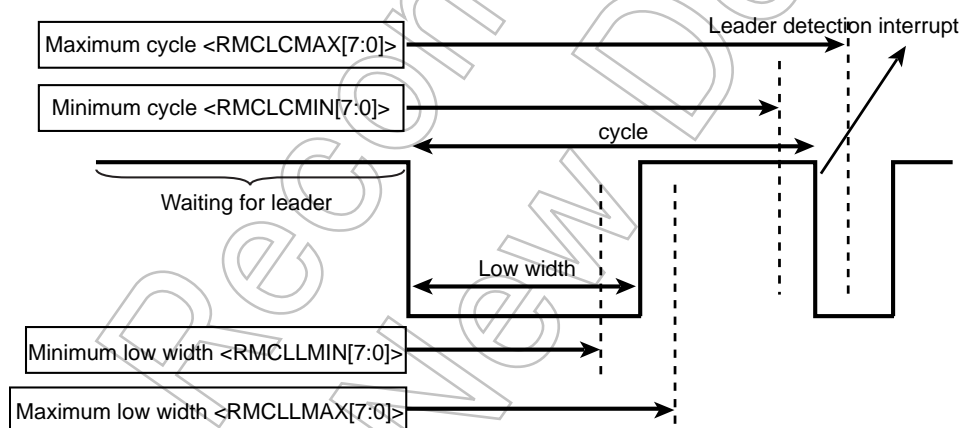
If you want to generate an interrupt when detecting a leader, configure the RMCRCR2
<RMCLIEN> bit.

A remote control signal without a leader cannot generate a leader detection interrupt.

(3)   Setting of 0/1 determination data bit

Based on a falling edge cycle, the data bit is determined as 0 or 1.

There are two kinds of determinations:

1.   Determination by threshold.

Configure a threshold value to RMCRCR3<RMCDATL[6:0]> bit which determines data bit as "0" or "1." If the determination value is equal to threshold value or more, it is determined as "1." If the determination value is less than threshold value, it is determined as "0."

2.   Determination by falling edge interrupt inputs.

By setting "1" to the RMCRCR2<RMCEDIEN> bit, a remote control signal input falling edge interrupt can be generated in each falling edge of the data bit. Using this interrupt together with a timer enables the determination to be done by software.

The followings shows the determination model of data bit.

Threshold of 0/1 detemination is set to 2.5T with the <RMCDATL[6:0]>bit.
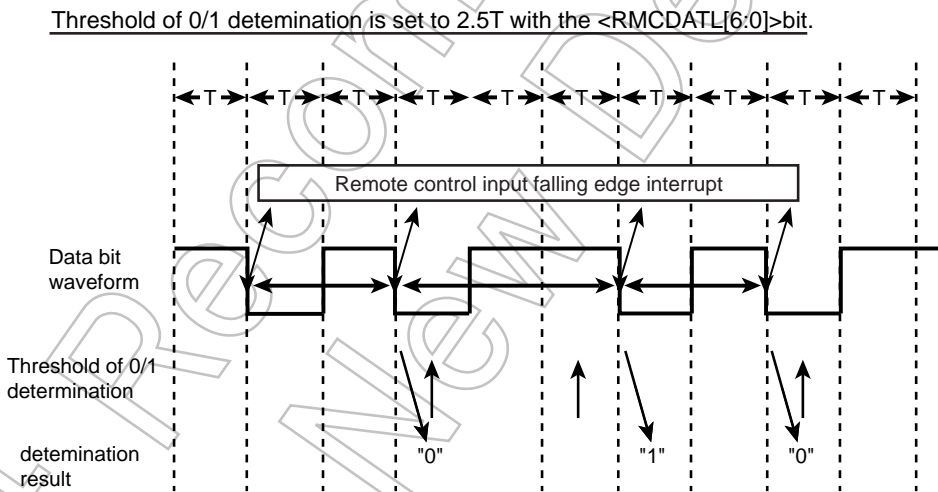
As for data bit determination of a remote control signal in a phase method, see"17.4.1.8 Receiving a Remote Control Signal in a Phase Method".

(4)   Settings of Reception Completion

To complete data reception, settings of detecting the maximum data bit cycle and excess low width are required. If multiple factors are specified, reception is completed by the factor detected first. Make sure to configure the reception completion settings.

1.   Completetion by the maximum data bit cycle

To complete reception by detecting a maximum data bit cycle, you need to configure the RMCRCR2 <RMCDMAX[7:0]> bits.

If the falling edge of the data bit cycle isn't monitored after time specified as threshold in the <RMCDMAX[7:0]> bits, a maximum data bit cycle is detected. The detection completes reception and generates an interrupt.After interrupt inputs generated, RMCRSTAT< RMCDMAXIF > bit is set to "1".

To complete reception by setting the number of receive data is set a RMCEND 1 to 3 register of each <RMCEND1>, <RMCEND2>, <RMCEND3>.In this case when the number of set reception bit agreed with the number of bit which received at the time of the outbreak of MAX on the number of receive data is set a RMCEND 1 to 3 register of each <RMCEND1>, <RMCEND2>, <RMCEND3>, it occurs by an MAX interrupt in data bit period.

As specified to RMCEND3 to 1, it is able to set three kinds of the receive data bit.

When it can receive the Maximum Data bit , the number of bit is not match the setting value in <RMCEND1>, <RMCEND2>, <RMCEND3>, it wait for Leader Reception.

The maximum data bit cycle interrupt

Threshold:<RMCDMAX[7:0]>
If the falling edge of the data bit cycle is not monitored after time specified as threshold,a maximum data bit cycle is detected.
The detection completes reception and generates an interrupt.
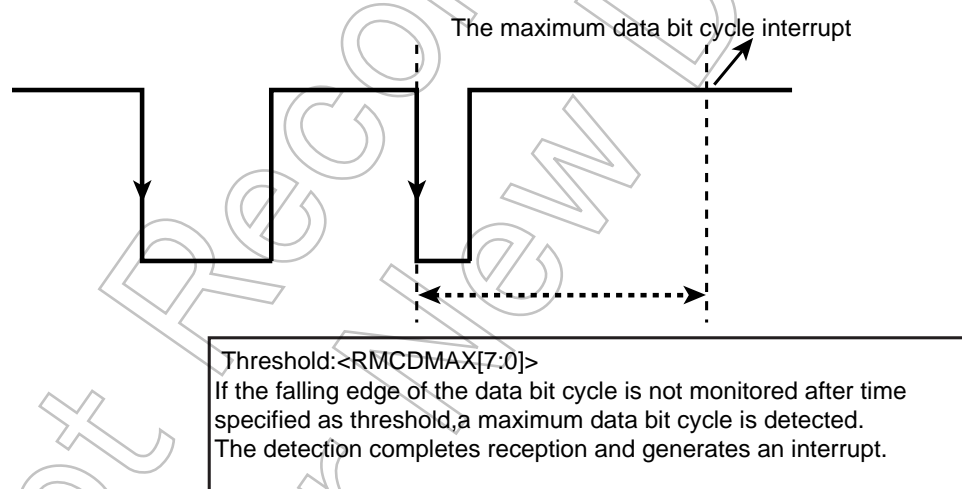
2.   Completetion by detecting low width

To complete reception by detecting the low width, you need to configure the RMCRCR2 <RMCLL[7:0]> bits.

After the falling edge of the data bit is detected, if the signal stays low longer than specified, excess low width is detected. The detection completes reception and generates an interrupt.

After interrupt inputs generated, RMCRSTAT<RMCLOIF> bit is set to "1."

Excess low width
detection interrupt.

Threshold:<RMCLL[7:0]>
Excess low width is detected when signal
stay low longer than specified.
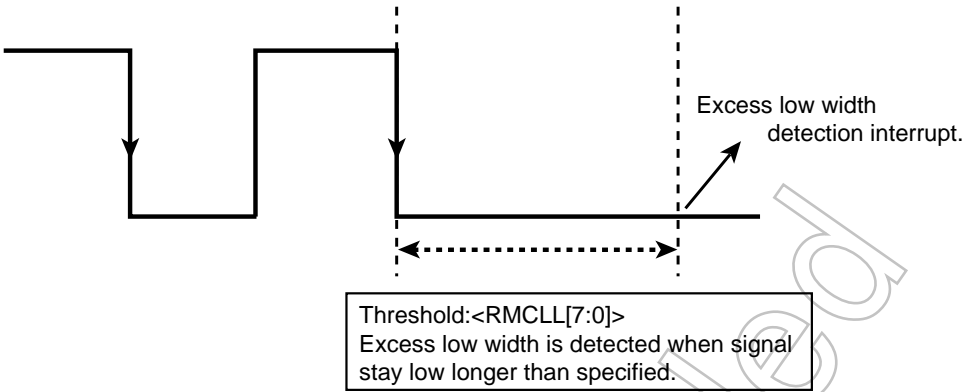
17.4.1.4 Enabling Reception

By enabling the RMCREN <RMCREN> bit after configuring the RMCRCR1, RMCRCR2, RMCRCR3 and RMCRCR4 registers, RMC is ready for reception. Detecting a leader initiates reception.

Note: Changing the configurations of the RMCRCR1, RMCRCR2, RMCRCR3 and RMCRCR4 registers during reception may harm their proper operation. Be careful if you change them during reception.

17.4.1.5 Stopping Reception

RMC stops reception by clearing the RMCREN <RMCREN> bit to "0" (reception disabled).

Clearing this bit during reception stops reception immediately and the received data is discarded.

17.4.1.6 Receiving Remote Control Signal without Leader in Waiting Leader

Setting RMCRCR2 <RMCLD> enables RMC to receive signals with or without a leader.

By setting RMCRCR2 <RMCLD>, RMC starts receiving data if it recognizes a signal of which low width is shorter than a maximum low width of leader detection specified in the RMCRCR1 <RMCLLMAX [7:0]> bits. RMC keeps receiving data until the final data bit is received.

If RMCRCR2 <RMCLD> is enabled, the same settings of error detection, reception completion and data bit determination of 0 or 1 are applied regardless of whether a signal has a leader or not.

Thus receivable remote control signals are limited.
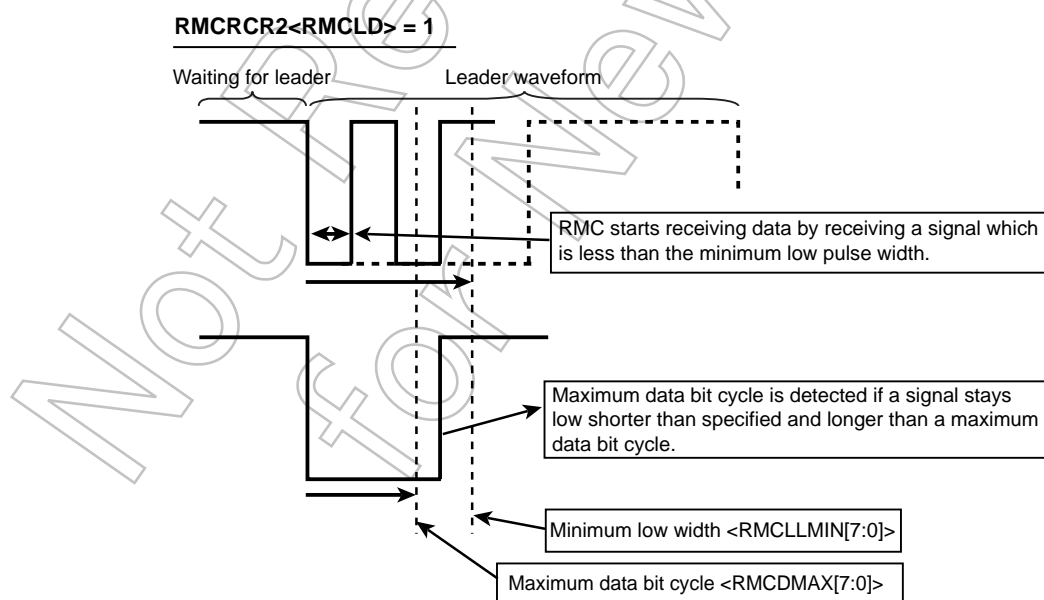
17.4.1.7 A Leader only with Low Width

The figure shown below illustrates a remote control signal that starts with a leader of which waveform only has low width.

This signal starts with a leader that only has low width and a data bit cycle starts from the rising edge. To enable the signal, it must be sent after being reversed by setting the RMCRCR4 <RMCPO> bit to "1".

This is because RMC is configured to detect a data bit cycle from the falling edge

To detect a leader, configure only low-pulse width of the leader with the <RMCLLMAX[7ÅF0] >=0y0000 _ 0000,<RMCLCMAX[7:0]>><RMCLCMIN[7:0]>.

In this case, the value of <RMCLLMIN[7:0]> is set as "don't care".

To detect whether data "0" or data "1", configure the threshold of 0/1 detection with the RMCRCR3 <RMCDATL[6:0]>.

The maximum data bit cycle is configured with the <RMCDMAX[7:0]> of the RMCRCR2.

To complete data reception, configure the maximum data bit cycle with <RMCDMAX[7:0]> of the RMCRCR2, and configure the low-pulse width detection with <RMCLL[7:0]>.

After detecting the maximum data bit cycle and confirming the low-pulse with which is specified after receiving the last bit, receiving data is completed.

The RMC generates an interrupt and waits for the next leader.
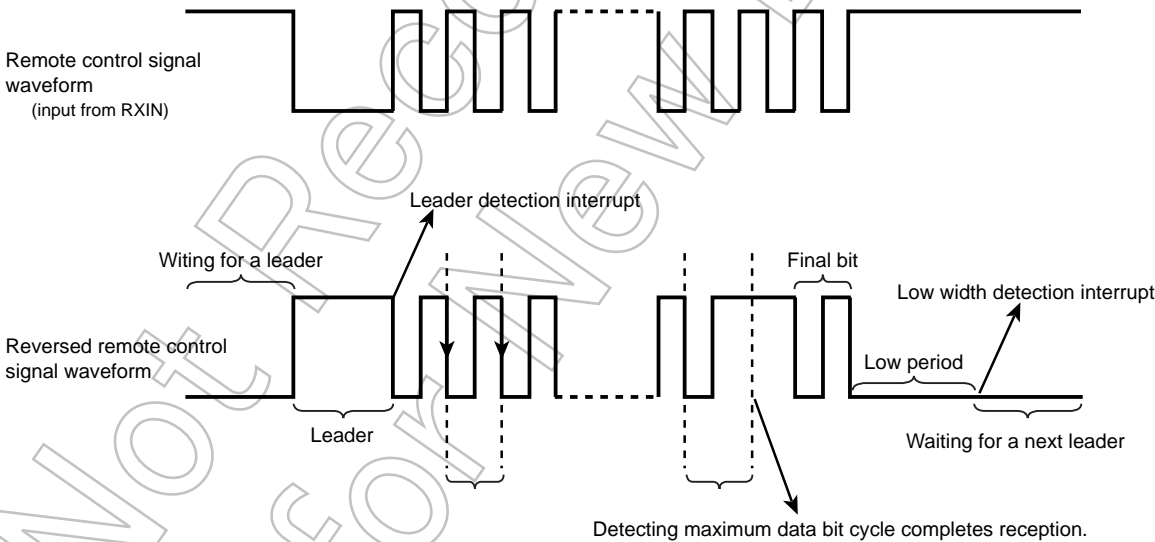
17.4.1.8 Receiving a Remote Control Signal in a Phase Method

RMC is capable of receiving a remote control signal in a phase method of which signal cycle is fixed. A signal in the phase method has three waveform patterns (see the figure shown below).

By setting two thresholds a remote control signal pattern is determined. RMC converts the signal into data "0" or "1". On completion of reception, received data "0" and "1" are stored in the RMCRBUF1, RMCRBUF 2 and RMCRBUF3.

By setting RMCRCR2<RMCPHM> = "1", RMC enables to receive a remote control signal in the phase method. Each threshold can be configured with the RMCRCR3 <RMCDATL[6:0]> bits and <RMCDATH[6:0]> bits.

Two thresholds are used to distinguish three waveform patterns. On condition that a cycle between two falling edges is "T", three patterns show cycles of 1T, 1.5T and 2T. Details of the two thresholds are shown below.

	Determined by	Threshold	Register bits to set
Threshold 1	Pattern 1 & pattern 2	1T to 1.5T	RMCRCR3<RMCDATL[6:0]>
Threshold 2	Pattern 2 & pattern 3	1.5T to 2T	RMCRCR3<RMCDATH[6:0]>

To determine a remote control signal in the phase method, three patterns of data waveform and preceding data are required. In addition, the signal needs to start from data "11".

Waveform pattern in phase method
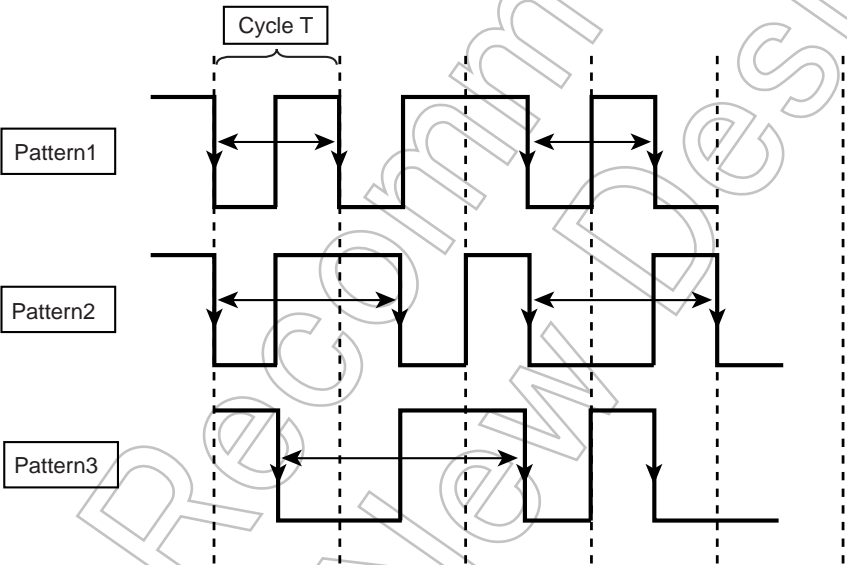
Remote control signal data in phase method

A pulse shape in cycle indicates whether it is data "0" or data "1".

<u>Remote control signal in phase method</u>

Remote control
signal

The first two bits of data
need to be "11".
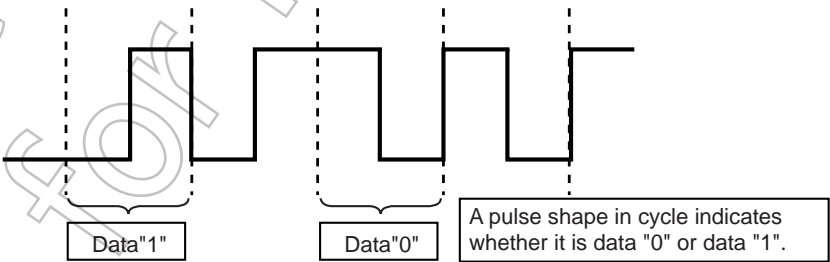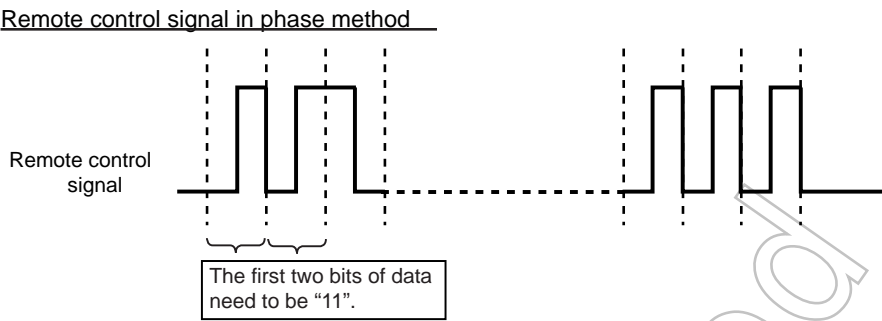
18.   USB Host Controller

The USB Host Controller (USBHC) is compliant with the USB Specification Revision 2.0 and the Open HCI Specification Release 1.0a, and supports USB transfers at 12 Mbps (full-speed). The USBHC is connected to the CPU via bus bridge logic.

The USCHC is subject to some restrictions. For details, See "18.8 Restrictions on Using the USB Host Controller".

18.1   System Overview

The key features of the USBHC are as follows :

1. Supports full-speed (12 Mbps) USB devices.

    Low-speed (1.5 Mbps) USB devices are not supported.

2. Supports control, bulk, interrupt and isochronous transfers (There are some restrictions. Refer to "18.8 Restrictions on Using the USB Host Controller").

3. Contains two 16-byte FIFO buffers (IN and OUT) in the bus bridge logic for connecting with the CPU, allowing up to 16 bytes of burst transfers.

4. Supports data transfers between the FIFO buffers in the bus bridge logic and the on-chip SRAM. Accessible SRAM is RAM0(0x2000_0000 to 0x2000_1FFF) and RAM1(0x2000_2000 to 0x2000_3FFF).

Note: **While USBHC accesses to RAM0 and RAM1, if CPU or DMA accesses to RAM0 and RAM1, CPU and DMA have priority. For detail, refer to "18.7.5 Competing access to the RAM0 and the RAM1".**

18.2 System Configuration

The USBHC consists of the following three blocks :

1. USBHC core (OHCI)

2. USB transceiver

3. CPU bus bridge logic
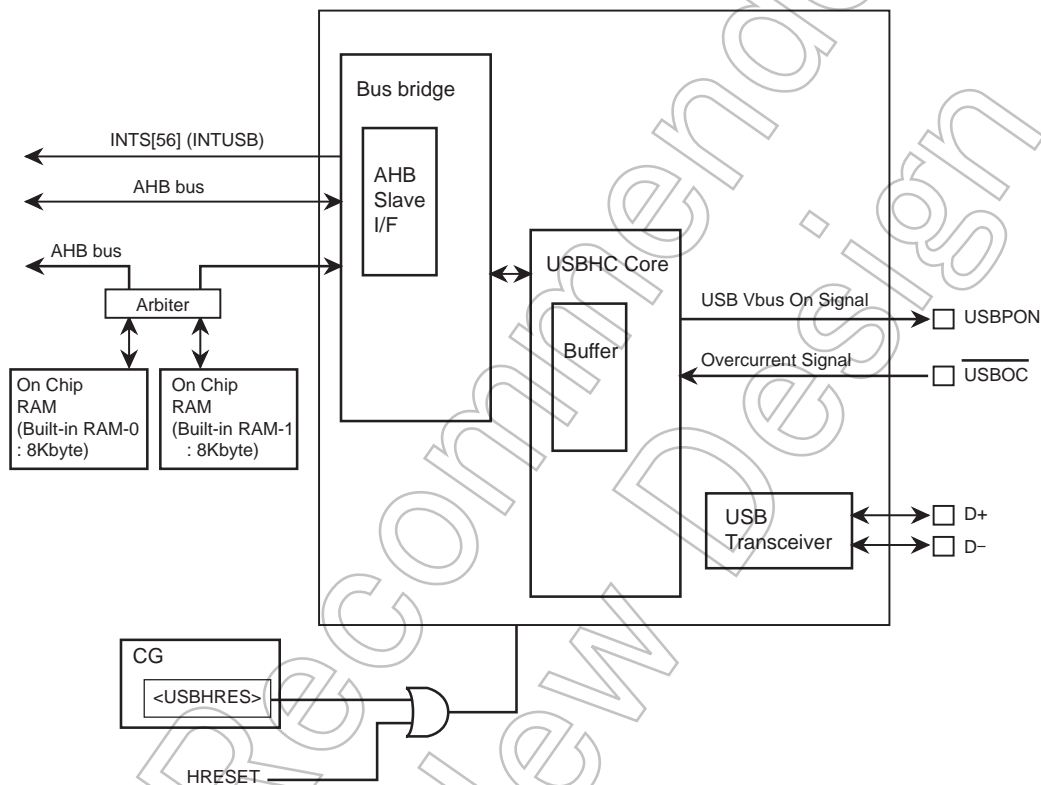
Figure 18-1 USB Host Controller Block Diagram

18.3   Interrupt

The USBHC generates the following interrupts :

- ・ Scheduling Overrun
- ・ HcDoneHead Write back
- ・ Start of Frame
- ・ Resume Detect
- ・ Unrecoverable Error
- ・ Frame Number Overflow
- ・ Root Hub Status Change
- ・ Ownership Change

When an event that causes an interrupt occurs, the USBHC sets the corresponding bit in the HcInterruptStatus register. At this time, if the MasterInterruptEnable (MIE) bit is enabled and the corresponding bit in the HcInterruptEnable register is enabled, a USB interrupt (INTUSB) is generated.

The USBHS driver software can clear each bit in the HcInterruptStatus register by writing a 1 to it. (The driver software cannot set these bits, and the USBHC cannot clear these bits.)

18.4 Reset

The USBHC is initialized by a hardware or software reset.

18.4.1 Hardware reset

A hardware reset is generated by an external reset pin or internal events (WDT reset, USBHC reset or Back-up mode reset).

- All registers are initialized.
- The reset signal is output on the bus by an external pull-down resister. (D+ = D- = 0)

    (The USB transceiver is in the SUSPEND state.)
- The USB state changes to the USBRESET state.
- List processing and SOF token generation are disabled.
- The FrameNumber field of the HcFmNumber register is not increased.

18.4.2 Software reset

A software reset is generated when a "1" is written to the HostControllerReset bit in the HcCommandStatus register.

- All OHCI registers are initialized except the following :
    - - The RemoteWakeupConnected and InterruptRouting bits in the HcControl register remain the same.
    - - The HcBCR0 register is not initialized.
- The USBHC outputs the reset signal on the USB bus (D+ =D- =0)
- The USB state changes to the USBSUSPEND state.

    (The FunctionalState bit in the HcControl register is set to 0x03 to transition to the USBSUS-PEND state.)

18.5 Bus Power Control

The USBHC has a control signal for the external Vbus power IC. This signal is controlled by the USBPON (PG6) pin.

To use PG6 as the USBPON pin, the port G control register (PGFR2) must be set appropriately. Then, setting the LPSC bit of the HcRhStatus register in the OCI register to "1" makes the USBPON pin output "High" level.

The $\overline{\text{USBOC}}$ (PG7) pin is used to detect overcurrent conditions. When low level is detected on this pin, the USBHC sets the OCI bit in the HcRhStatus register to "1". (To use PG7 as the $\overline{\text{USBOC}}$ pin, the port G control register (PGFR2) must be set appropriately.)

18.6   Register

The USBHC contains a set of control registers compliant with the Open HCI Specification which are mapped into the memory space. The bus bridge logic for connecting with the CPU also includes control registers.

These registers are directly accessible from the CPU via a 32-bit bus.

Base Address = 0x4000_3000

Register name		Address(Base+)
Hc Revision Register	HcRevision	0x0000
Hc Control Register	HcControl	0x0004
Hc Command Status Register	HcCommandStatus	0x0008
Hc Interrupt Status Register	HcInterruptStatus	0x000C
Hc Interrupt Enable Register	HcInterruptEnable	0x0010
Hc Interrupt Disable Register	HcInterruptDisable	0x0014
Hc Host Controller Communication Area Register	HcHCCA	0x0018
Hc Period Current Endpoint Descriptor Register	HcPeriodCurrentED	0x001C
Hc Control Head Endpoint Descriptor Register	HcControlHeadED	0x0020
Hc Control Current Endpoint Descriptor Register	HcControlCurrentED	0x0024
Hc Bulk Head Endpoint Descriptor Register	HcBulkHeadED	0x0028
Hc Bulk Current Endpoint Descriptor Register	HcBulkCurrentED	0x002C
Hc Done Head Register	HcDoneHead	0x0030
Hc Frame Interval Register	HcFmInterval	0x0034
Hc Frame Remaining Register	HcFmRemaining	0x0038
Hc Frame Number Register	HcFmNumber	0x003C
Hc Period Start Register	HcPeriodStart	0x0040
Hc Low Speed Threshold Register	HcLSThreshold	0x0044
Hc Root hub Descriptor A Register	HcRhDescriptorA	0x0048
Hc Root hub Descriptor B Register	HcRhDescripterB	0x004C
Hc Root hub Status Register	HcRhStatus	0x0050
Hc Root hub Port Status Register	HcRhPortStatus1	0x0054
Hc BCR0 Resister	HcBCR0	0x0080

Note 1: **The Open HCI Specification Release 1.0a specifies the FrameRemaining (FR) and FrameRemainingToggle (FRT) bits in the HcFmRemaining register and the FramNumber (FN) bit in the HcFmNumber register as read-only to the Host Control Driver (HDC). However, the USBHC allows write accesses to these registers by the HCD for debug purposes. If the HCD writes to these registers, undefined results will occur. These bits must not be written by the HCD.**

Note 2: **The explanation of "18.6.1 HcRevision Register" to "18.6.23 HcBCR0 Register" are references. About the explanation based on OHCI, refer to "Open HCI Specification Release 1.0a" specification.**

18.6.1   HcRevision Register

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset								
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset								
	7	6	5	4	3	2	1	0
bit symbol	REV							
After reset	0	0	0	1	0	0	0	0

Bit	Bit Symbol	Type (HDC)	Type (HC)	Function
31-8	–	–	–	Reserved
7-0	REV[7:0]	R	R	Filed name:Revision This read-only field contains the BCD representation of the version of the HCI specification that is implemented by this HC. For example, a value of 0x11 corresponds to version 1.1. All of the HC implementations that are compliant with this specification will have a value of 0x10.

18.6.2   HcControl Register

The HcControl register defines the operating modes for the Host Controller. Most of the fields in this register are modified only by the Host Controller Driver, except HostControllerFunctionalState and RemoteWakeup-Connected.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset								
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	RWE	RWC	IR
After reset						0	0	0
	7	6	5	4	3	2	1	0
bit symbol	HCFS		BLE	CLE	IE	PLE	CBSR	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-11	–	–	–	Reserved
10	RWE	R/W	R	Filed name:Remote Walk-up Enable This bit is used by HCD to enable or disable the remote walk-up feature upon the detection of upstream resume signaling. When this bit is set and the ResumeDetected bit in HcInterruptStatus is set, a remote walk-up is signaled to the host system. Setting this bit has no impact on the generation of hardware interrupt.
9	RWC	R/W	R/W	Filed name:Remote Walk-up Connected This bit indicates whether HC supports remote walk-up signaling. If remote walk-up is supported and used by the system, it is the responsibility of system firmware to set this bit during POST. HC clears the bit upon a hardware reset but does not alter it upon a software reset.
8	IR	R/W	R	Filed name:Interrupt Routing This bit determines the routing of interrupts generated by events registered in HcInterruptStatus. If clear, all interrupts are routed to the normal host bus interrupt mechanism. If set, interrupts are routed to the System Management Interrupt. HCD clears this bit upon a hardware reset, but it does not alter this bit upon a software reset. HCD uses this bit as a tag to indicate the ownership of HC.
7-6	HCFS[1:0]	R/W	R/W	Filed name:Host Controller Functional State For USB 00 : USBRESET 01 : USBRESUME 10 : USBOPERATIONAL 11 : USBSUSPEND A transition to USBOPERATIONAL from another state causes SOF generation to begin 1 ms later. HCD may determine whether HC has begun sending SOFs by reading the StartofFrame field of HcInterruptStatus. This field may be changed by HC only when in the USBSUSPEND state. HC may move from the USBSUSPEND state to the USBRESUME state after detecting the resume signaling from a downstream port. HC enters USBSUSPEND after a software reset, whereas it enters USBRESET after a hardware reset. The latter also resets the Root Hub and asserts subsequent reset signaling to downstream ports.
5	BLE	R/W	R	Filed name:Bulk List Enable This bit is set to enable the processing of the Bulk list in the next Frame. If cleared by HCD, processing of the Bulk list does not occur after the next SOF. HC checks this bit whenever it determines to process the list. When disabled, HCD may modify the list. If HcBulkCurrentED is pointing to an ED to be removed, HCD must advance the pointer by updating HcBulkCurrentED before re-enabling processing of the list.

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
4	CLE	R/W	R	Filed name:Control List Enable This bit is set to enable the processing of the Control list in the next Frame. If cleared by HCD, processing of the Control list does not occur after the next SOF. HC must check this bit whenever it determines to process the list. When disabled, HCD may modify the list. If HcControlCurrentED is pointing to an ED to be removed, HCD must advance the pointer by updating HcControlCurrentED before re-enabling processing of the list.
3	IE	R/W	R	Filed name:Isochronous Enable This bit is used by HCD to enable/disable processing of isochronous EDs. While processing the periodic list in a Frame, HC checks the status of this bit when it finds an Isochronous ED (F=1). If set (enabled), HC continues processing the EDs. If cleared (disabled), HC halts processing of the periodic list (which now contains only isochronous EDs) and begins processing the Bulk/Control lists. Setting this bit is guaranteed to take effect in the next Frame (not the current Frame). * This product has some restrictions on isochronous transfers.
2	PLE	R/W	R	Filed name:Periodic List Enable This bit is set to enable the processing of the periodic list in the next Frame. If cleared by HCD, processing of the periodic list does not occur after the next SOF. HC must check this bit before it starts processing the list.
1-0	CBSR[1:0]	R/W	R	Filed name:Control Bulk Service Ratio This specifies the service ratio between Control and Bulk EDs. Before processing any of the nonperiodic lists, HC must compare the ratio specified with its internal count on how many nonempty Control EDs have been processed, in determining whether to continue serving another Control ED or switching to Bulk EDs. The internal count will be retained when crossing the frame boundary. In case of reset, HCD is responsible for restoring this value. <table><tr><td><CBSR[1:0]></td><td>No. of Control EDs over Bulk EDs served</td></tr><tr><td>00</td><td>1 : 1</td></tr><tr><td>01</td><td>2 : 1</td></tr><tr><td>10</td><td>3 : 1</td></tr><tr><td>11</td><td>4 : 1</td></tr></table>

18.6.3   HcCommandStatus Register

The HcCommandStatus register is used by the Host Controller to receive commands issued by the Host Controller Driver, as well as reflecting the current status of the Host Controller. To the Host Controller Driver, it appears to be a "write to set" register. The Host Controller must ensure that bits written as "1" are set in the register while bits written as "0" remain unchanged in the register. The Host Controller Driver may issue multiple distinct commands to the Host Controller without concern for corrupting previously issued commands. The Host Controller Driver has normal read access to all bits.

The SchedulingOverrunCount field indicates the number of frames with which the Host Controller has detected the scheduling overrun error. This occurs when the Periodic list does not complete before EOF. When a scheduling overrun error is detected, the Host Controller increments the counter and sets the SchedulingOverrun field in the HcInterruptStatus register.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset								
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	SOC	
After reset							0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset								
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	OCR	BLF	CLF	HCR
After reset					0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-8	–	–	–	Reserved
17-16	SOC[1:0]	R	R/W	Filed name:Scheduling Overrun Count These bits are incremented on each scheduling overrun error. It is initialized to 00 and wraps around at 11. This will be incremented when a scheduling overrun is detected even if SchedulingOverrun in HcInterruptStatus has already been set. This is used by HCD to monitor any persistent scheduling problems.
15-4	–	–	–	Reserved
3	OCR	R/W	R/W	Filed name:Ownership Change Request This bit is set by an OS HCD to request a change of control of the HC. When set HC will set the OwnershipChange field in HcInterruptStatus. After the changeover, this bit is cleared and remains so until the next request from OS HCD.
2	BLF	R/W	R/W	Filed name:Bulk List Filled This bit is used to indicate whether there are any TDs on the Bulk list. It is set by HCD whenever it adds a TD to an ED in the Bulk list. When HC begins to process the head of the Bulk list, it checks BF. As long as BulkListFilled is 0, HC will not start processing the Bulk list. If BulkListFilled is 1, HC will start processing the Bulk list and will set BF to 0. If HC finds a TD on the list, then HC will set BulkListFilled to 1 causing the Bulk list processing to continue. If no TD is found on the Bulk list, and if HCD does not set BulkListFilled, then BulkListFilled will still be 0 when HC completes processing the Bulk list and Bulk list processing will stop.
1	CLF	R/W	R/W	Filed name:Control List Filled This bit is used to indicate whether there are any TDs on the Control list. It is set by HCD whenever it adds a TD to an ED in the Control list. When HC begins to process the head of the Control list, it checks CLF. As long as ControlListFilled is 0, HC will not start processing the Control list. If CF is 1, HC will start processing the Control list and will set ControlListFilled to 0. If HC finds a TD on the list, then HC will set ControlListFilled to 1 causing the Control list processing to continue. If no TD is found on the Control list, and if the HCD does not set ControlListFilled, then ControlListFilled will still be 0 when HC completes processing the Control list and Control list processing will stop.
0	HCR	R/W	R/W	Filed name:Host Controller Reset This bit is set by HCD to initiate a software reset of HC. Regardless of the functional state of HC, it moves to the USBSUSPEND state in which most of the operational registers are reset except those stated otherwise; e.g., the InterruptRouting field of HcControl, and no Host bus accesses are allowed. This bit is cleared by HC upon the completion of the reset operation. The reset operation must be completed within 10 s. This bit, when set, should not cause a reset to the Root Hub and no subsequent reset signaling should be asserted to its downstream ports.

18.6.4   HcInterruptStatus Register

This register provides status on various events that cause hardware interrupts. When an event occurs, the Host Controller sets the corresponding bit in this register. When the bit is set, a hardware interrupt is generated if the interrupt is enabled in the HcInterruptEnable register and the MasterInterruptEnable bit is set. The Host Controller Driver may clear specific bits in this register by writing "1" to bit positions to be cleared. The Host Controller Driver may not set any of these bits. The Host Controller will never clear the bit.

	31	30	29	28	27	26	25	24
bit symbol	-	OC	-	-	-	-	-	-
After reset		0						
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset								
	7	6	5	4	3	2	1	0
bit symbol	-	RHSC	FNO	UE	RD	SF	WDH	SO
After reset		0	0	0	0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31	–	–	–	Reserved
30	OC	R/W	R/W	Filed name:Ownership Change This bit is set by HC when HCD sets the OwnershipChangeRequest field in HcCommandStatus. This event, when unmasked, will always generate an System Management Interrupt (SMI) immediately. This bit is tied to 0 when the SMI pin is not implemented.
29-7	–	–	–	Reserved
6	RHSC	R/W	R/W	Filed name:Root Hub Status Change This bit is set when the content of HcRhStatus or the content of any of HcRhPortStatus[NumberofDownstreamPort] has changed.
5	FNO	R/W	R/W	Filed name:Frame Number Overflow This bit is set when the MSB of HcFmNumber (bit 15) changes value, from 0 to 1 or from 1 to 0, and after HccaFrameNumber has been updated.
4	UE	R/W	R/W	Filed name:Unrecoverable Error This bit is set when HC detects a system error not related to USB. HC should not proceed with any processing nor signaling before the system error has been corrected. HCD clears this bit after HC has been reset.
3	RD	R/W	R/W	Filed name:Resume Detected This bit is set when HC detects that a device on the USB is asserting resume signaling. It is the transition from no resume signaling to resume signaling causing this bit to be set. This bit is not set when HCD sets the USBRESUME state.
2	SF	R/W	R/W	Filed name:Startof Frame This bit is set by HC at each start of a frame and after the update of HccaFrameNumber. HC also generates a SOF token at the same time
1	WDH	R/W	R/W	Filed name:Writeback Done Head This bit is set immediately after HC has written HcDoneHead to HccaDoneHead. Further updates of the HccaDoneHead will not occur until this bit has been cleared. HCD should only clear this bit after it has saved the content of HccaDoneHead.
0	SO	R/W	R/W	Filed name:Scheduling Overrun This bit is set when the USB schedule for the current Frame overruns and after the update of HccaFrameNumber. A scheduling overrun will also cause the SchedulingOverrunCount of HcCommandStatus to be incremented.

18.6.5 HcInterruptEnable Register

Each enable bit in the HcInterruptEnable register corresponds to an associated interrupt bit in the HcInterrupt-Status register. The HcInterruptEnable register is used to control which events generate a hardware interrupt. When a bit is set in the HcInterruptStatus register and the corresponding bit in the HcInterruptEnable register is set and the MasterInterruptEnable bit is set, then a hardware interrupt is requested on the host bus.

Writing a "1" to a bit in this register sets the corresponding bit, whereas writing a "0" to a bit in this register leaves the corresponding bit unchanged. On read, the current value of this register is returned.

	31	30	29	28	27	26	25	24
bit symbol	MIE	OC	-	-	-	-	-	-
After reset	0	0						
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset								
	7	6	5	4	3	2	1	0
bit symbol	-	RHSC	FNO	UE	RD	SF	WDH	SO
After reset		0	0	0	0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31	MIE	R/W	R	Filed name:Master Interrupt Enable A '0' written to this field is ignored by HC. A '1' written to this field enables interrupt generation due to events specified in the other bits of this register. This is used by HCD as a Master Interrupt Enable.
30	OC	R/W	R	Filed name:Ownership Change 0: Ignore 1: Enable interrupt generation due to Ownership Change.
29-7	–	–	–	Reserved
6	RHSC	R/W	R	Filed name:Root Hub Status Change 0: Ignore 1:Enable interrupt generation due to Root Hub Status Change.
5	FNO	R/W	R	Filed name:Frame Number Overflow 0: Ignore 1: Enable interrupt generation due to Frame Number Overflow.
4	UE	R/W	R	Filed name:Unrecoverable Error 0: Ignore 1: Enable interrupt generation due to Unrecoverable Error.
3	RD	R/W	R	Filed name:Resume Detected 0: Ignore 1: Enable interrupt generation due to Resume Detected.
2	SF	R/W	R	Filed name:Startof Frame 0: Ignore 1: Enable interrupt generation due to Start of Frame.
1	WDH	R/W	R	Filed name:Writeback Done Head 0: Ignore 1: Enable interrupt generation due to HcDoneHeadWriteback.
0	SO	R/W	R	Filed name:Scheduling Overrun 0: Ignore 1: Enable interrupt generation due to Scheduling Overrun.

18.6.6   HcInterruptDisable Register

Each disable bit in the HcInterruptDisable register corresponds to an associated interrupt bit in the HcInterruptStatus register. The HcInterruptDisable register is coupled with the HcInterruptEnable register. Thus, writing a "1" to a bit in this register clears the corresponding bit in the HcInterruptEnable register, whereas writing a "0" to a bit in this register leaves the corresponding bit in the HcInterruptEnable register unchanged. On read, the current value of the HcInterruptEnable register is returned.

	31	30	29	28	27	26	25	24
bit symbol	MIE	OC	-	-	-	-	-	-
After reset	0	0						
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset								
	7	6	5	4	3	2	1	0
bit symbol	-	RHSC	FNO	UE	RD	SF	WDH	SO
After reset		0	0	0	0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31	MIE	R/W	R	Filed name:Master Interrupt Enable A '0' written to this field is ignored by HC. A '1' written to this field disables interrupt generation due to events specified in the other bits of this register. This field is set after a hardware or software reset.
30	OC	R/W	R	Filed name:Ownership Change 0: Ignore 1: Disable interrupt generation due to Ownership Change.
29-7	–	–	–	Reserved
6	RHSC	R/W	R	Filed name:Root Hub Status Change 0: Ignore 1: Disable interrupt generation due to Root Hub Status Change.
5	FNO	R/W	R	Filed name:Frame Number Overflow 0: Ignore 1: Disable interrupt generation due to Frame Number Overflow.
4	UE	R/W	R	Filed name:Unrecoverable Error 0: Ignore 1: Disable interrupt generation due to Unrecoverable Error.
3	RD	R/W	R	Filed name:Resume Detected 0: Ignore 1: Disable interrupt generation due to Resume Detected.
2	SF	R/W	R	Filed name:Startof Frame 0: Ignore 1: Disable interrupt generation due to Start of Frame.
1	WDH	R/W	R	Filed name:Writeback Done Head 0: Ignore 1: Disable interrupt generation due to HcDoneHeadWriteback.
0	SO	R/W	R	Filed name:Scheduling Overrun 0: Ignore 1: Disable interrupt generation due to SchedulingOverrun.

18.6.7   HcHCCA Register

The HcHCCA register contains the physical address of the Host Controller Communication Area. The Host Controller Driver determines the alignment restrictions by writing all 1s to HcHCCA and reading the content of HcHCCA. The alignment is evaluated by examining the number of zeroes in the lower order bits. The minimum alignment is 256 bytes; therefore, bits 0 through 7 must always return "0" when read. This area is used to hold the control structures and the Interrupt table that are accessed by both the Host Controller and the Host Controller Driver.

	31	30	29	28	27	26	25	24
bit symbol	HCCA							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	HCCA							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	HCCA							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	-
After reset								

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-8	HCCA[23:0]	R/W	R	Filed name:Host Controller Communication Area This is the base address of the Host Controller Communication Area.
7-0	–	–	–	Reserved

18.6.8   HcPeriodCurrentED Register

The HcPeriodCurrentED register contains the physical address of the current Isochronous or Interrupt End-point Descriptor.

	31	30	29	28	27	26	25	24
bit symbol	PCED							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	PCED							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	PCED							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PCED				-	-	-	-
After reset	0	0	0	0				

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-4	PCED[27:0]	R	R/W	Filed name:Period Current ED This is used by HC to point to the head of one of the Periodic lists which will be processed in the current Frame. The content of this register is updated by HC after a periodic ED has been processed.
3-0	–	–	–	Reserved

18.6.9 HcControlHeadED Register

The HcControlHeadED register contains the physical address of the first Endpoint Descriptor of the Control list.

	31	30	29	28	27	26	25	24
bit symbol	CHED							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	CHED							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	CHED							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CHED				-	-	-	-
After reset	0	0	0	0				

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-4	CHED[27:0]	R/W	R	Filed name:Control Head ED HC traverses the Control list starting with the HcControlHeadED pointer. The content is loaded from HCCA during the initialization of HC.
3-0	–	–	–	Reserved

18.6.10　HcControlCurrentED Register

The HcControlCurrentED register contains the physical address of the current Endpoint Descriptor of the Control list.

	31	30	29	28	27	26	25	24
bit symbol	CCED							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	CCED							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	CCED							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CCED				-	-	-	-
After reset	0	0	0	0				

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-4	CCED[27:0]	R/W	R/W	Filed name:Control Current ED This pointer is advanced to the next ED after serving the present one. HC will continue processing the list from where it left off in the last Frame. When it reaches the end of the Control list, HC checks the ControlListFilled field of HcCommandStatus. If set, HC copies the content of HcControlHeadED to HcControlCurrentED and clears the bit. If not set, HC does nothing. HCD is allowed to modify this register only when the ControlListEnable field of HcControl is cleared. When set, HCD only reads the instantaneous value of this register. Initially, this is set to zero to indicate the end of the Control list.
3-0	–	–	–	Reserved

18.6.11 HcBulkHeadED Register

The HcBulkHeadED register contains the physical address of the first Endpoint Descriptor of the Bulk list.

	31	30	29	28	27	26	25	24
bit symbol	BHED							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	BHED							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	BHED							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	BHED				-	-	-	-
After reset	0	0	0	0				

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-4	BHED[27:0]	R/W	R	Filed name:Bulk Head ED HC traverses the Bulk list starting with the HcBulkHeadED pointer. The content is loaded from HCCA during the initialization of HC.
3-0	–	–	–	Reserved

18.6.12   HcBulkCurrentED Register

The HcBulkCurrentED register contains the physical address of the current endpoint of the Bulk list. As the Bulk list will be served in a round-robin fashion, the endpoints will be ordered according to their insertion to the list.

	31	30	29	28	27	26	25	24
bit symbol	BCED							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	BCED							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	BCED							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	BCED				-	-	-	-
After reset	0	0	0	0				

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-4	BCED[27:0]	R/W	R/W	Filed name:Bulk Current ED This is advanced to the next ED after the HC has served the present one. HC continues processing the list from where it left off in the last Frame. When it reaches the end of the Bulk list, HC checks the ControlListFilled field of HcControl. If set, HC copies the content of HcBulkHeadED to HcBulkCurrentED and clears the bit. If it is not set, HC does nothing. HCD is only allowed to modify this register when the BulkListEnable of HcControl is cleared. When set, the HCD only reads the instantaneous value of this register. This is initially set to zero to indicate the end of the Bulk list.
3-0	–	–	–	Reserved

18.6.13   HcDoneHead Register

The HcDoneHead register contains the physical address of the last completed Transfer Descriptor that was added to the Done queue. In normal operation, the Host Controller Driver should not need to read this register as its content is periodically written to the HCCA.

	31	30	29	28	27	26	25	24
bit symbol				DH				
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol				DH				
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol				DH				
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol		DH			-	-	-	-
After reset	0	0	0	0				

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-4	DH[27:0]	R	R/W	Filed name:Done Head When a TD is completed, HC writes the content of HcDoneHead to the NextTD field of the TD. HC then overwrites the content of HcDoneHead with the address of this TD. This is set to zero whenever HC writes the content of this register to HCCA. It also sets the Write-backDoneHead of HcInterruptStatus.
3-0	−	−	−	Reserved

18.6.14   HcFmInterval Register

The HcFmInterval register contains a 14-bit value which indicates the bit time interval in a Frame, (i.e., between two consecutive SOFs), and a 15-bit value indicating the Full Speed maximum packet size that the Host Controller may transmit or receive without causing scheduling overrun. The Host Controller Driver may carry out minor adjustment on the FrameInterval by writing a new value over the present one at each SOF. This provides the programmability necessary for the Host Controller to synchronize with an external clocking resource and to adjust any unknown local clock offset.

	31	30	29	28	27	26	25	24
bit symbol	FIT	FSMPS						
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	FSMPS							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	FI					
After reset	0	0	1	0	1	1	1	0
	7	6	5	4	3	2	1	0
bit symbol	FI							
After reset	1	1	0	1	1	1	1	1

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31	FIT	R/W	R	Filed name:Frame Interval Toggle HCD toggles this bit whenever it loads a new value to FrameInterval.
30-16	FSMPS[14:0]	R/W	R	Filed name:FS Largest data Packet This field specifies a value which is loaded into the Largest Data Packet Counter at the beginning of each frame. The counter value represents the largest amount of data in bits which can be sent or received by the HC in a single transaction at any given time without causing scheduling over- run. The field value is calculated by the HCD.
15-14	−	−	−	Reserved
13-0	FI[13:0]	R/W	R	Filed name:Frame Interval This specifies the interval between two consecutive SOFs in bit times. The nominal value is set to be 11,999. HCD should store the current value of this field before resetting HC. By setting the HostController- Reset field of HcCommandStatus as this will cause the HC to reset this field to its nominal value. HCD may choose to restore the stored value upon the completion of the Reset sequence.

18.6.15 HcFmRemaining Register

The HcFmRemaining register is a 14-bit down counter showing the bit time remaining in the current Frame.

	31	30	29	28	27	26	25	24
bit symbol	FRT	-	-	-	-	-	-	-
After reset	0							
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	FR					
After reset			0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	FR							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31	FRT	R	R/W	Filed name:Frame Remaining Toggle This bit is loaded from the FrameIntervalToggle field of HcFmInterval whenever FrameRemaining reaches 0. This bit is used by HCD for the synchronization between FrameInterval and FrameRemaining.
30-14	–	–	–	Reserved
13-0	FR[13:0]	R	R/W	Filed name:Frame Remaining This counter is decremented at each bit time. When it reaches zero, it is reset by loading the FrameInterval value specified in HcFmInterval at the next bit time boundary. When entering the USBOPERATIONAL state, HC re-loads the content with the FrameInterval of HcFmInterval and uses the updated value from the next SOF.

18.6.16   HcFmNumber Register

The HcFmNumber register is a 16-bit counter. It provides a timing reference among events happening in the Host Controller and the Host Controller Driver. The Host Controller Driver may use the 16-bit value specified in this register and generate a 32-bit frame number without requiring frequent access to the register.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset								
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	FN							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	FN							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-16	−	−	−	Reserved
15-0	FN[15:0]	R	R/W	Filed name:Frame Number This is increased when HcFmRemaining is re-loaded. It will be rolled over to 0x0000 after 0xffff. When entering the USBOPERATIONAL state, this will be increased automatically. The content will be written to HCCA after HC has increased the FrameNumber at each frame boundary and sent a SOF but before HC reads the first ED in that Frame. After writing to HCCA, HC will set the StartofFrame in HcInterruptStatus.

18.6.17   HcPeriodicStart Register

The HcPeriodicStart register has a 14-bit programmable value which determines the earliest time when HC should start processing the periodic list

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset								
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	PS					
After reset			0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PS							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-14	–	–	–	Reserved
13-0	PS[13:0]	R/W	R	Filed name:Periodic Start After a hardware reset, this field is cleared. This is then set by HCD during the HC initialization. The value is calculated roughly as 10% off from HcFmInterval. A typical value will be 3E67h. When HcFmRemaining reaches to the value specified, processing of the periodic lists will have priority over Control/Bulk processing. HC will therefore start processing the Interrupt list after completing the current Control or Bulk transaction that is in progress.

18.6.18   HcLSThreshold Register

The HcLSThreshold register contains an 12-bit value used by the Host Controller to determine whether to commit to the transfer of a maximum of 8-byte LS packet before EOF. Neither the Host Controller nor the Host Controller Driver are allowed to change this value.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset								
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	LST			
After reset					0	1	1	0
	7	6	5	4	3	2	1	0
bit symbol	LST							
After reset	0	0	1	0	1	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-12	–	–	–	Reserved
11-0	LST[11:0]	R/W	R	Filed name:LS Threshold This field contains a value which is compared to the FrameRemaining field prior to initiating a Low Speed transaction. The transaction is started only if FrameRemaining this field. The value is calculated by HCD with the consideration of transmission and setup overhead.

18.6.19 HcRhDescriptorA Register

The HcRhDescriptorA register is one of two registers describing the characteristics of the Root Hub. Reset values are implementation-specific. The descriptor length, descriptor type, and hub controller current fields of the hub Class Descriptor are emulated by the HCD. All other fields are located in the HcRhDescriptorA and HcRhDescriptorB registers.

	31	30	29	28	27	26	25	24
bit symbol	POTPGT							
After reset	0	0	0	0	0	0	1	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset								
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	NOCP	OCPM	DT	NPS	PSM
After reset				0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	NDP							
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-24	POTPGT[7:0]	R/W	R	Filed name:Power On To Power Good Time This byte specifies the duration HCD has to wait before accessing a powered-on port of the Root Hub. It is implementation-specific. The unit of time is 2 ms. The duration is calculated as POTPGT×2 ms.
23-13	–	–	–	Reserved
12	NOCP	R/W	R	Filed name:No Over Current Protection This bit describes how the overcurrent status for the Root Hub ports are reported. When this bit is cleared, the OverCurrentProtectionMode field specifies global or per-port reporting. 0: Over-current status is reported collectively for all downstream ports. 1: No overcurrent protection supported.
11	OCPM	R/W	R	Filed name:Over Current Protection Mode This bit describes how the overcurrent status for the Root Hub ports are reported. At reset, this fields should reflect the same mode as PowerSwitchingMode. This field is valid only if the NoOver-CurrentProtection field is cleared. 0: Over-current status is reported collectively for all downstream ports. 1: Over-current status is reported on a per-port basis.
10	DT	R	R	Device Type This bit specifies that the Root Hub is not a compound device. The Root Hub is not permitted to be a compound device. This field should always read/write 0.
9	NPS	R/W	R	Filed name:No Power Switching These bits are used to specify whether power switching is supported or ports are always powered. It is implementation- specific. When this bit is cleared, the PowerSwitchingMode specifies global or per-port switching. 0: Ports are power switched. 1: Ports are always powered on when the HC is powered on.
8	PSM	R/W	R	Filed name:Power Switching Mode This bit is used to specify how the power switching of the Root Hub ports is controlled. It is implementation-specific. This field is only valid if the NoPowerSwitching field is cleared. 0: All ports are powered at the same time. 1: Each port is powered individually. This mode allows port power to be controlled by either the global switch or per-port switching. If the PortPowerControlMask bit is set, the port responds only to port power commands (Set/ClearPortPower). If the port mask is cleared, then the port is controlled only by the global power switch (Set/ClearGlobalPower).
7-0	NDP[7:0]	R	R	Filed name:Number Downstream Ports These bits specify the number of downstream ports supported by the Root Hub. It is implementation-specific. This module has one port, so "0x01" is read.

18.6.20   HcRhDescriptorB Register

The HcRhDescriptorB register is one of two registers describing the characteristics of the Root Hub. These fields are written during initialization to correspond with the system implementation. Reset values are implementation-specific.

	31	30	29	28	27	26	25	24
bit symbol	PPCM							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	PPCM							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	DR							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DR							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-16	PPCM[15:0]	R/W	R	Filed name:Port Power Control Mask Each bit indicates if a port is affected by a global power control command when PowerSwitching-Mode is set. When set, the port's power state is only affected by per-port power control (Set/Clear-PortPower). When cleared, the port is controlled by the global power switch (Set/ClearGlobalPower-er). If the device is configured to global switching mode (PowerSwitchingMode=0), this field is not valid. bit0: Reserved bit1: Ganged-power mask on Port#1 bit2: Ganged-power mask on Port#2 (Note) bit15: Ganged-power mask on Port#15
15-0	DR[15:0]	R/W	R	Filed name:Device Removable Each bit is dedicated to a port of the Root Hub. When this bit is "0", the attached device is removable. When this bit is "1", the attached device is not removable. bit0: Reserved bit1: Device attached to Port#1 bit2: Device attached to Port#2 (Note) bit15: Device attached to Port#15

Note:**Since this host controller does not have Port#2 to Port#15, write "0" to the corresponding bit.**

18.6.21   HcRhStatus Register

The HcRhStatus register is divided into two parts. The lower word of a Dword represents the Hub Status field and the upper word represents the Hub Status Change field. Reserved bits should always be written "0".

	31	30	29	28	27	26	25	24
bit symbol	CRWE	-	-	-	-	-	-	-
After reset	Undefined							
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	OCIC	LPSC
After reset							0	0
	15	14	13	12	11	10	9	8
bit symbol	DRWE	-	-	-	-	-	-	-
After reset	0							
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	OCI	LPS
After reset							0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31	CRWE	W	R	Filed name:Clear Remote Wakeup Enable Writing a "1" clears DeviceRemoveWakeupEnable. Writing a "0" has no effect.
30-18	–	–	–	Reserved
17	OCIC	R/W	R/W	Filed name:Over Current Indicator Change This bit is set by hardware when a change has occurred to the OCI field of this register. The HCD clears this bit by writing a "1". Writing a "0" has no effect.
16	LPSC	R/W	R	Filed name:Local Power Status Change (read)LocalPowerStatusChange The Root Hub does not support the local power status feature; thus, this bit is always read as "0". (write)SetGlobalPower In global power mode (PowerSwitchingMode="0"), This bit is written to "1" to turn on power to all ports (set PortPowerStatus). In per-port power mode, it sets PortPowerStatus only on ports whose PortPowerControlMask bit is not set. Writing a "0" has no effect.
15	DRWE	R/W	R	Filed name:Device Remote Walk-up Enable (read)DeviceRemoteWakeupEnable This bit enables a ConnectStatusChange bit as a resume event, causing a USBSUSPEND to USBRESUME state transition and setting the ResumeDetected interrupt. 0: ConnectStatusChange is not a remote wakeup event. 1: ConnectStatusChange is a remote wakeup event. (write) Writing a "1" sets DeviceRemoveWakeupEnable. Writing a "0" has no effect. (Refer to "18.8 Restrictions on Using the USB Host Controller")
14-2	–	–	–	Reserved
1	OCI	R	R/W	Filed name:Over Current Indicator This bit reports overcurrent conditions when the global reporting is implemented. When set, an overcurrent condition exists. When cleared, all power operations are normal. If per-port overcurrent protection is implemented, this bit is always "0".
0	LPS	R/W	R	Filed name:Local Power Status (read)LocalPowerStatus The Root Hub does not support the local power status feature; thus, this bit is always read as "0". (write)ClearGlobalPower In global power mode (PowerSwitchingMode=0), this bit is written to "1" to turn off power to all ports (clear PortPowerStatus). In per-port power mode, it clears PortPowerStatus only on ports whose PortPowerControlMask bit is not set. Writing a "0" has no effect.

18.6.22 HcRhPortStatus1 Register

The HcRhPortStatus1 register is used to control and report port events on a per-port basis. NumberDownstreamPorts represents the number of HcRhPortStatus registers that are implemented in hardware. The lower word is used to reflect the port status, whereas the upper word reflects the status change bits. Some status bits are implemented with special write behavior (see below). If a transaction (token through handshake) is in progress when a write to change port status occurs, the resulting port status change must be postponed until the transaction is complete. Reserved bits should always be written "0".

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset								
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	PRSC	OCIC	PSSC	PESC	CSC
After reset				0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	LSDA	PPS
After reset							Undefined	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PRS	POCI	PSS	PES	CCS
After reset				0	0	0	0	0

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
31-21	–	–	–	Reserved
20	PRSC	R/W	R/W	Filed name:Port Reset Status Change This bit is set at the end of the 10 ms port reset signal. The HCD writes a "1" to clear this bit. Writing a "0" has no effect. 0: port reset is not complete. 1: port reset is complete.
19	OCIC	R/W	R/W	Filed name:Port Over Current Indicator Change This bit is valid only if overcurrent conditions are reported on a per-port basis. This bit is set when Root Hub changes the PortOverCurrentIndicator bit. The HCD writes a "1" to clear this bit. Writing a "0" has no effect. 0: No change in PortOverCurrentIndicator 1: PortOverCurrentIndicator has changed.
18	PSSC	R/W	R/W	Filed name:Port Suspend Status Change This bit is set when the full resume sequence has been completed. This sequence includes the 20 s resume pulse, LS EOP, and 3 ms resynchronization delay. The HCD writes a "1" to clear this bit. Writing a "0" has no effect. This bit is also cleared when ResetStatusChange is set. 0: Resume is not complete. 1: Resume is complete.
17	PESC	R/W	R/W	Filed name:Port Enable Status Change This bit is set when hardware events cause the PortEnableStatus bit to be cleared. Changes from HCD writing do not set this bit. The HCD writes a "1" to clear this bit. Writing a "0" has no effect. 0: No change in PortEnableStatus. 1: Change in PortEnableStatus.
16	CSC	R/W	R/W	Filed name:Connect Status Change This bit is set whenever a connect or disconnect event occurs. The HCD writes a "1" to clear this bit. Writing a "0" has no effect. If CurrentConnectStatus is cleared when a SetPortReset, SetPortEnable, or SetPortSuspend write occurs, this bit is set to force the driver to re-evaluate the connection status since these writes should not occur if the port is disconnected. 0: No change in CurrentConnectStatus. 1: Change in CurrentConnectStatus (Note) If the DeviceRemovable[NDP] bit is set, this bit informs to set only a Root Hub reset while the device is is attached to the system.
15-10	–	–	–	Reserved

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
9	LSDA	R/W	R/W	(read)Low Speed Device Attached
				This bit indicates the speed of the device attached to this port. When set, a Low Speed device is attached to this port. When clear, a Full Speed device is attached to this port. This field is valid only when the CurrentConnectStatus is set.
				0: Full speed device attached.
				1: Low speed device attached.
				(write)Clear Port Power
				The HCD clears the PortPowerStatus bit by writing a "1" to this bit. Writing a "0" has no effect.
8	PPS	R/W	R/W	(read)Port Power Status
				This bit reflects the port's power status, regardless of the type of power switching implemented. This bit is cleared if an overcurrent condition is detected. HCD sets this bit by writing SetPortPower or SetGlobalPower. HCD clears this bit by writing ClearPortPower or ClearGlobalPower. Which power control switches are enabled is determined by PowerSwitchingMode and PortPortControlMask[NDP]. In global switching mode (PowerSwitchingMode="0"), only Set/ClearGlobalPower controls this bit. In per-port power switching (PowerSwitchingMode="1"), if the PortPowerControlMask [NDP] bit for the port is set, only Set/ClearPortPower commands are enabled. If the mask is not set, only Set/ClearGlobalPower commands are enabled. When port power is disabled, CurrentConnectStatus, PortEnableStatus, PortSuspendStatus, and PortResetStatus should be reset.
				0: Port power off
				1: Port power on
				(write)Set Port Power
				The HCD writes a '1' to set the PortPowerStatus bit. Writing a '0' has no effect.
				(Note)
				This bit is always reads "1" if power switching is not supported.
7-5	–	–	–	Reserved
4	PRS	R/W	R/W	(read)Port Reset Status
				When this bit is set by a write to SetPortReset, port reset signaling is asserted. When reset is completed, this bit is cleared when PortResetStatusChange is set. This bit cannot be set if CurrentConnectStatus is cleared.
				0: Port reset signal is not active
				1: Port reset signal is active
				(write)Set Port Reset
				The HCD sets the port reset signaling by writing a "1" to this bit. Writing a "0" has no effect. If CurrentConnectStatus is cleared, this write does not set PortResetStatus, but instead sets ConnectStatusChange. This informs the driver that it attempted to reset a disconnected port.
3	POCI	R/W	R/W	(read)Port Over Current Indicator
				This bit is only valid when the Root Hub is configured in such a way that overcurrent conditions are reported on a per-port basis. If per-port overcurrent reporting is not supported, this bit is set to "0". If cleared, all power operations are normal for this port. If set, an overcurrent condition exists on this port. This bit always reflects the overcurrent input signal
				0: No overcurrent condition
				1: Overcurrent condition detected
				(write)ClearSuspendStatus
				The HCD writes a "1" to initiate a resume. Writing a "0" has no effect. A resume is initiated only if PortSuspendStatus is set.
2	PSS	R/W	R/W	(read)Port Suspend Status
				This bit indicates the port is suspended or in the resume sequence. It is set by a SetSuspendState write and cleared when PortSuspendStatusChange is set at the end of the resume interval. This bit cannot be set if CurrentConnectStatus is cleared. This bit is also cleared when PortResetStatusChange is set at the end of the port reset or when the HC is placed in the USBRESUME state. If an upstream resume is in progress, it should propagate to the HC.
				0: Port is not suspended.
				1: Port is suspended.
				(write)Set Port Suspend
				The HCD sets the PortSuspendStatus bit by writing a "1" to this bit. Writing a "0" has no effect. If CurrentConnectStatus is cleared, this write does not set PortSuspendStatus; instead it sets ConnectStatusChange. This informs the driver that it attempted to suspend a disconnected port.

Bit	Bit Symbol	Type (HCD)	Type (HC)	Function
1	PES	R/W	R/W	(read)Port Enable Status
				This bit indicates whether the port is enabled or disabled. The Root Hub may clear this bit when an overcurrent condition, disconnect event, switched-off power, or operational bus error such as babble is detected. This change also causes PortEnabledStatusChange to be set. HCD sets this bit by writing SetPortEnable and clears it by writing ClearPortEnable. This bit cannot be set when CurrentConnectStatus is cleared. This bit is also set, if not already, at the completion of a port reset when ResetStatusChange is set or port suspend when SuspendStatusChange is set.
				0: Port is disabled.
				1: Port is enabled.
				(write)Set Port Enable
				The HCD sets PortEnableStatus by writing a "1". Writing a "0" has no effect. If CurrentConnectStatus is cleared, this write does not set PortEnableStatus, but instead sets ConnectStatusChange. This informs the driver that it attempted to enable a disconnected port.
0	CCS	R/W	R/W	(read)Current Connect Status
				This bit reflects the current state of the downstream port.
				0: No device connected
				1: Device connected
				(write)Clear Port Enable
				The HCD writes a "1" to this bit to clear the PortEnableStatus bit. Writing a "0" has no effect. The CurrentConnectStatus is not affected by any write.
				(Note)
				This bit is always read as '1' when the attached device is nonremovable (DeviceRemoveable[NDP]).

18.6.23   HcBCR0 Register

The HcBCR0 register controls over current input of enable or disable to the USBHC and the SUSPEND state of the USB transceiver.

The USBHC is not active in SLOW mode or Low Power Consumption Modes. Therefore, before entering SLOW mode or Low Power Consumption Modes, place the USBHC and the USB transceiver in the SUSPEND state.

	31	30	29	28	27	26	25	24
bit symbol	-	TRNS_SUSP	OVCE	-	-	-	-	-
After reset	0	1	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	−	R	Reserved
30	TRNS_SUSP	R/W	Filed name:Transceiver Suspend This bit controls the SUSPEND state of the USB transceiver. To enter SLOW mode or Low power Consumption Modes with the USB transceiver in the SUSPEND state, set this bit to "1". 0: - (Controlled by the USB Host Controller) 1: Suspend
29	OVCE	R/W	Filed name:USB Host Over Current input Enable 0: Enable 1: Disable
28-0	−	R	Reserved

18.7   Notes on Using the USB Host Controller

18.7.1   Setting the USB Clock

The PLL is not active after reset release. Therefore, before accessing the USB Host, the CGPLLSEL <PLLSEL> must be set to "1".

18.7.2   Oscillator Recommendation

To generate a clock for the USBHC, we recommend using a 12-MHz crystal oscillator with an accuracy of ±50ppm or less to comply with the USB specification.

Note that a USB clock generated by the on-chip PLL may not satisfy the requirements of the USB specification depending on the implementation environment, conditions and variations.

18.7.3   Entering SLOW Mode and Low Power Consumption Modes

Before entering SLOW Mode and Low Power Consumption Modes, the USBHC must be placed in the SUSPEND state. Then turn off Vbus.

(Example Software setting)

1	Disable interrupt				
2	HcCommandStatus	<HCR>	= 1	(Write)	: Software reset of the USB host controller
3	HcControl	<HCFS>	= 1 1	(Read)	: Check the transition to the SUSPEND state.
4	HcBCR0	<TRANS_SUSP>	= 1	(Write)	: Change the state of the D+ and D- pin to the SUSPEND state
5	Output "0" to PG6				: Vbus OFF
6	Setup to shifting low power consumption and others.				: Stop of peripheral function, port setting, warm up setting and so on. About shifting BACKUP mode, refer to that section.
7	Stop PLL				
8	Execute WFI instruction				

18.7.4   When not using USB

The D+ and D- pins must be pulled-down.

18.7.5   Competing access to the RAM0 and the RAM1

If the CPU/DMA accesses the RAM0 or the RAM1 when the USBHC is already accessing it, the USBHC connection is discontinued and the CPU/DMA has higher priority for access. The USBHC is reconnected when the CPU/DMA access is completed. Note that if the disconnection of the USBHC caused by the CPU/DMA takes long, it may cause problems in USB transfer. Specifically, in case IN transfer, the CPU/DMA must complete to access in the period where the 64-byte FIFO contained in the USBHC becomes full (within 42.66μs).

18.8   Restrictions on Using the USB Host Controller

1. For an isochronous transfer, a Frame number to be transferred is defined in an Isochronous Transfer Descriptor (ITD). However, Frame numbers are not synchronized between the Host and software. If a descriptor to be executed in a previous Frame is scheduled later, the Host determines that a time error has occurred and writes back DATAOVERRUN to the CC field of the ITD. At this time, if the following conditions are met, the Host will write back inappropriate status (NOERROR).

   <Conditions>

   The above problem occurs if transfers are scheduled in a way the following two conditions both true:

   1 ITD.FC[2:0] = R[2:0]
   2 ITD.FC[2:0] < R[15:0]

   Where ITD.FC indicates the number of times an ITD is executed and R = HcFmNumber (current Frame number) - ITD.SF (transfer start Frame number).

   Make sure that each ITD is synchronized to the current Frame number. If not, this ITD should not be linked.

2. If a fatal error occurs on the USB system and the Host detects this error, the OHCI core sets HcInterrutStatus.UE[4].

   At this time, if the HcInterruptEnable.UE[4] register has been set additionally, a hardware interrupt is generated. After this interrupt is detected, a software reset (HcCommandStatus.HCR[0] = 0y1) is required to recover from the Unrecoverable Error state and the Host then moves to the SUSPEND state.

3. After the software reset, OHCI registers are initialized. If a remote wakeup from the device occurs, the Host remains in the SUSPEND state. When the remote wakeup function is used, a program for recovering from the SUSPEND state must be prepared.

4. When an overcurrent error occurs, if the HcRhDescriptorA.NPS[9] register has been set to "1", the HcRhPortStatus1.PRS[4] and HcRhPortStatus1.PSS[2] bits are not cleared. Therefore, do not set the HcRhDescriptorA.NPS[9] to "1" and the HcRhDescriptorB.DR[Port No] to "1".

5. When the HcRhStatus.DRWE [15] is set to "1", a remote wakeup event does not cause a USBSUSPEND to USBRESUME transition. When remote wakeup events are used, causing state transition by monitoring to the HcInterruptStatus.RD[3] by the HCD. And when remote wakeup events are not used, do not set the HcRhStatus.DRWE [15] to "1".

6. In a system supporting overcurrent conditions, set the HcRhDescriptorA.NOCP [12] to "0".

7. When Port Reset is executed while generating Babble, Port keep disable state. Port Reset sequence is below.

(Example)

1	HcRhPortStatus1	<PRS>	=	1	(Write)	: Excute port reset
2	Detect of HW interrupt					
3	HcRhPortStatus1	<PRSC>	=	1	(Read)	: Port is an invalid state.
	HcRhPortStatus1	<PES>	=	0		
	HcRhPortStatus1	<CCS>	=	1		
4	Clear of HW interrupt					
5	HcRhPortStatus1	<PRS>	=	1	(Write)	: Retry port reset
6	Detect if HW interrupt					
7	HcRhPortStatus1	<PRSC>	=	1	(Read)	: Port is a valid state.
	HcRhPortStatus1	<PES>	=	1		
	HcRhPortStatus1	<CCS>	=	1		

8. When Scheduling Overrun is occurred while executing Full-Speed Isochronous transfer, It may be generated the Unrecoverable Error. When Unrecoverable Error was generated, recover from Unrecoverable Error by software reset.

9. The data received from the USB device is written to a specified internal RAM through the BUS-I/F as receive buffer; however, data is not written to the internal RAM under the conditions shown below. Align the starting address to be multiple of 4, do not specify the address starting from 4n+1 of receiving buffer.

4n indicates the address aligned to 4 bytes.

<Condition ignored write accesses>

Write accesses are ignored when the conditions below are all satisfied.
1. The received data is MPS (Max Packet Size)×n+2
2. The last 2 bytes of the write buffer address are 4n+1 and 4n+2.

18.9   Connection Example
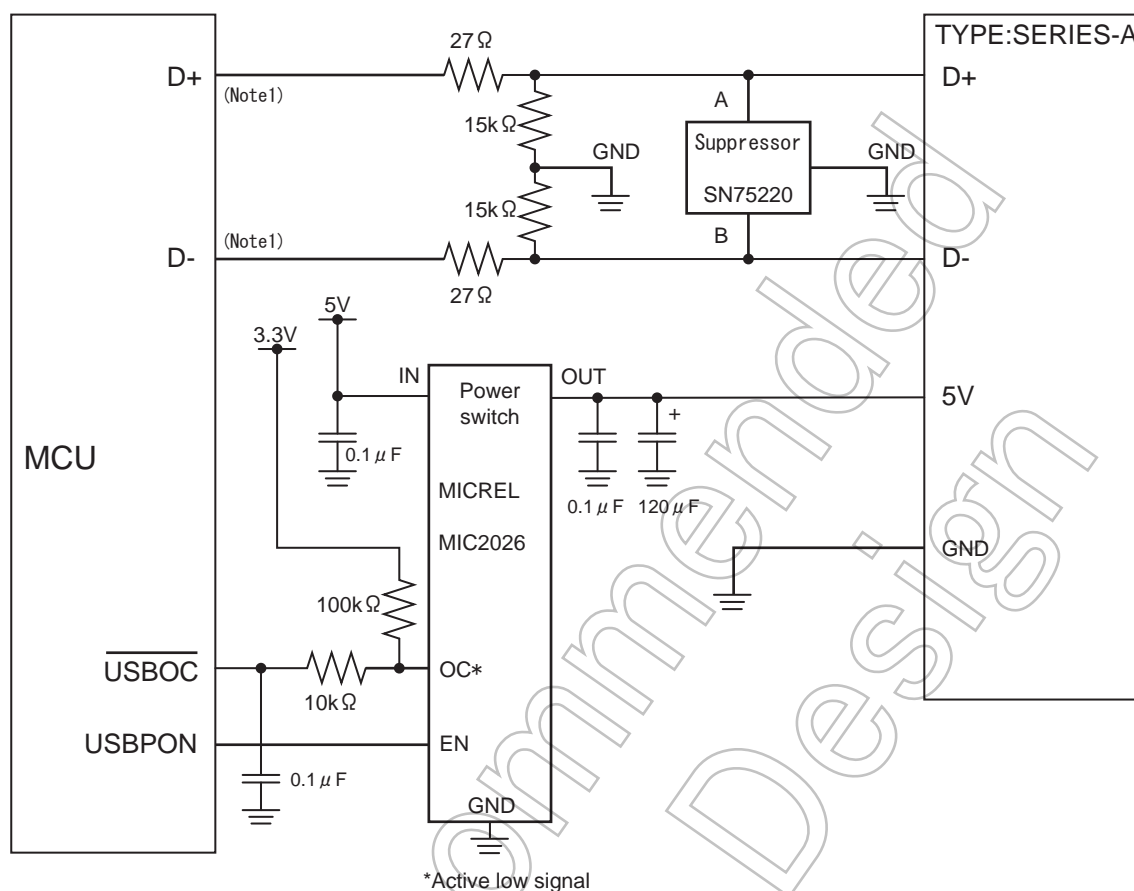
Bus power switch device	: TPS2052 (Texas Instruments)
	: MIC2026-1BM (MICREL)
	: MIC2026-1BN (MICREL)
Transient voltage suppressor device	: SN75240 (Texas Instruments)
Resistance precision	: 5%
Resistance rating	: 1/2W for 27É¹ recommended
Capacitor	: Low ESR type 120µF capacitor (OS-CON, etc.) recommended

Note 1: **Do not apply voltages exceeding the absolute maximum rating.**

Note 2: **When designing your board, make sure that the D+ and D- pins are placed at the equal distance from the USB A receptacle.**

Note 3: **No suppressor device is required in the USB specification.**

Note 4: **After reset release, the USBPON and $\overline{\text{USBOC}}$ pins are initialized as input ports. Process the pin so as not to influence the external control circuit.**

19. Watchdog Timer(WDT)

The watchdog timer (WDT) is for detecting malfunctions (runaways) of the CPU caused by noises or other disturbances and remedying them to return the CPU to normal operation.

If the watchdog timer detects a runaway, it generates a INTWDT interrupt or reset.

Note:INTWDT interrupt is a factor of the non-maskable interrupts (NMI).

Also, the watchdog timer notifies of the detecting malfunction to the external peripheral devices from the watchdog timer pin ($\overline{\text{WDTOUT}}$) by outputting "Low".

19.1   Configuration

Figure 19-1shows the block diagram of the watchdog timer.
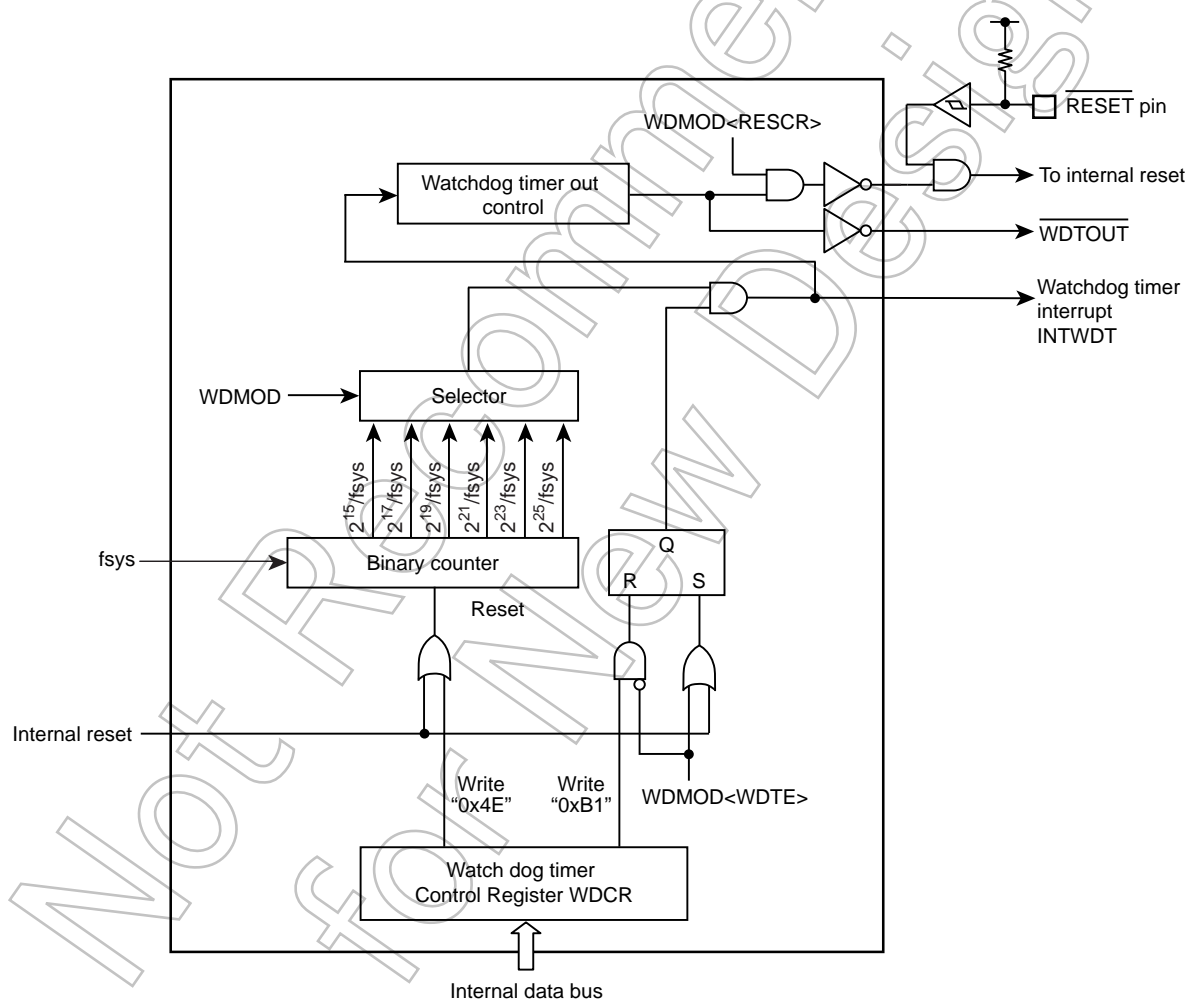
Figure 19-1 Block Diagram of the Watchdog Timer

19.2   Register

The followings are the watchdog timer control registers and addresses.

Base Address = 0x400F_2000

Register name		Address(Base+)
Watchdog Timer Mode Register	WDMOD	0x0000
Watchdog Timer Control Register	WDCR	0x0004

19.2.1   WDMOD(Watchdog Timer Mode Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	WDTE	WDTP			-	I2WDT	RESCR	-
After reset	1	0	0	0	0	0	1	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	WDTE	R/W	Enable/Disable control 0:Disable 1:Enable
6-4	WDTP[2:0]	R/W	Selects WDT detection time(Refer toTable 19-1) 000: 2^{15}/fsys 100: 2^{23}/fsys 001: 2^{17}/fsys 101: 2^{25}/fsys 010: 2^{19}/fsys 110:Setting prohibited. 011: 2^{21}/fsys 111:Setting prohibited.
3	−	R	Read as 0.
2	I2WDT	R/W	Operation when IDLE mode 0: Stop 1:In operation
1	RESCR	R/W	Operation after detecting malfunction 0: INTWDT interrupt request generates. (Note) 1: Reset
0	−	R/W	Write 0.

Note:INTWDT interrupt is a factor of the non-maskable interrupts (NMI).

Table 19-1 Detection time of watchdog timer (fc = 64MHz)

Clock gear value CGSYSCR<GEAR[2:0]>	WDMOD<WDTP[2:0]>					
	000	001	010	011	100	101
000 (fc)	0.51 ms	2.05 ms	8.19 ms	32.77 ms	131.07 ms	524.29 ms
100 (fc/2)	1.02 ms	4.10 ms	16.38 ms	65.54 ms	262.14 ms	1.05 s
101 (fc/4)	2.05 ms	8.19 ms	32.77 ms	131.07 ms	524.29 ms	2.10 s
110 (fc/8)	4.10 ms	16.38 ms	65.54 ms	262.14 ms	1.05 s	4.19 s

19.2.2 WDCR (Watchdog Timer Control Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	WDCR							
After reset	-	-	-	-	-	-	-	-

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as 0.
7-0	WDCR	W	Disable/Clear code 0xB1:Disable code 0x4E: Clear code Others:Reserved

19.3 Operations

19.3.1 Basic Operation

The Watchdog timer is consists of the binary counters that work using the system clock (fsys) as an input. Detecting time can be selected between 2^{15}, 2^{17}, 2^{19} , 2^{21}, 2^{23} and 2^{25} by the WDMOD<WDTP[2:0]>. The detecting time as specified is elapsed, the watchdog timer interrupt (INTWDT) generates, and the watchdog timer out pin ($\overline{\text{WDTOUT}}$) output "Low".

To detect malfunctions (runaways) of the CPU caused by noise or other disturbances, the binary counter of the watchdog timer should be cleared by software instruction before INTWDT interrupt generates. If the binary counter is not cleared, the non-maskable interrupt generates by INTWDT. Thus CPU detects malfunction (runway), malfunction countermeasure program is performed to return to the normal operation.

Additionally, it is possible to resolve the problem of a malfunction (runaway) of the CPU by connecting the watchdog timer out pin to reset pins of peripheral devices.

19.3.2 Operation Mode and Status

The watchdog timer begins operation immediately after a reset is cleared.

If not using the watchdog timer, it should be disabled.

The watchdog timer cannot be used as the high-speed frequency clock is stopped. Before transition to below modes, the watchdog timer should be disabled.In IDLE2 mode, its operation depends on the WDMOD<I2WDT> setting.

- STOP mode
- SLEEP mode
- SLOW mode
- BACKUP STOP mode
- BACKUP SLEEP mode

Also, the binary counter is automatically stopped during debug mode.

19.4 Operation when malfunction (runaway) is detected

19.4.1 INTWDT interrupt generation

In the Figure 19-2 shows the case that INTWDT interrupt generates (WDMOD<RESCR>="0").

When an overflow of the binary counter occurs, INTWDT interrupt generates. It is a factor of non-maskable interrupt (NMI). Thus CPU detects non-maskable interrupt and performs the countermeasure program.

The factor of non-maskable interrupt is the plural. CGNMIFLG identifies the factor of non-maskable interrupts. In the case of INTWDT interrupt, CGNMIFLG<NMIFLG0> is set.

When INTWDT interrupt generates, simultaneously the watchdog timer out ($\overline{\text{WDTOUT}}$) output "Low". $\overline{\text{WDTOUT}}$ becomes "High" by the watchdog timer clearing that is writing clear code 0x4E to the WDCR register.
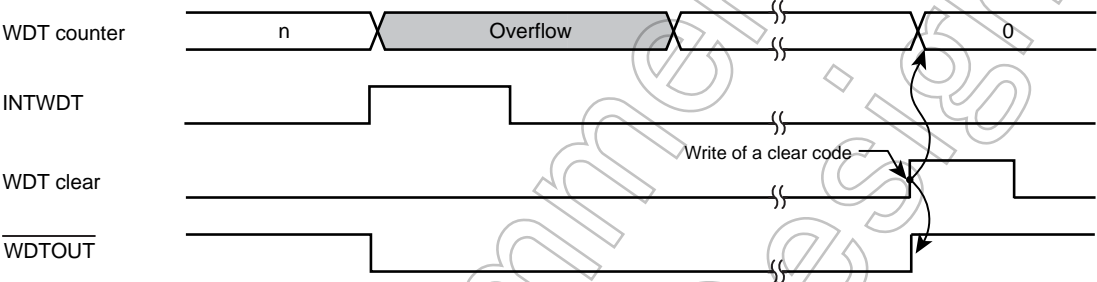
Figure 19-2 INTWDT interrupt generation

19.4.2   Internal reset generation

Figure 19-3 shows the internal reset generation (WDMOD<RESCR>="1").

MCU is reset by the overflow of the binary counter. In this case, reset status continues for 32 states. A clock is initialized so that input clock (fsys) is the same as a high-speed frequency clock (fosc). This means fsys = fosc.
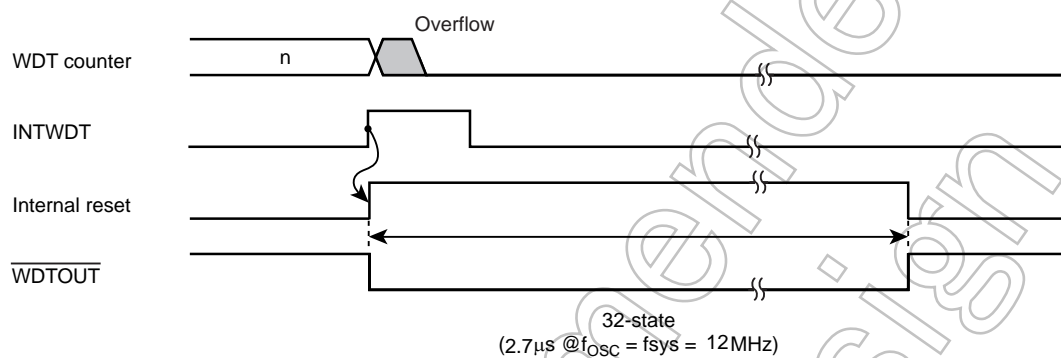
Figure 19-3 Internal reset generation

19.5   Control register

The watchdog timer (WDT) is controlled by two control registers WDMOD and WDCR.

19.5.1   Watchdog Timer Mode Register (WDMOD)

1. Specifying the detection time of the watchdog timer <WDTP[2:0]>.

    Set the watchdog timer detecting time to WDMOD<WDTP[2:0]>. After reset, it is initialized to WDMOD<WDTP[2:0]> = "000".

2. Enabling/disabling the watchdog timer <WDTE>.

    When resetting, WDMOD <WDTE> is initialized to "1" and the watchdog timer is enabled.

    To disable the watchdog timer to protect from the error writing by the malfunction, first <WDTE> bit is set to "0", and then the disable code (0xB1) must be written to WDCR register.

    To change the status of the watchdog timer from "disable" to "enable," set the <WDTE> bit to "1".

3. Watchdog timer out reset connection <RESCR>

    This register specifies whether WDTOUT is used for internal reset or interrupt. After reset, WDMOD<RESCR> is initialized to "1", the internal reset is generated by the overflow of binary counter.

19.5.2   Watchdog Timer Control Register(WDCR)

This is a register for disabling the watchdog timer function and controlling the clearing function of the binary counter.

19.5.3　Setting example

19.5.3.1　Disabling control

By writing the disable code (0xB1) to this WDCR register after setting WDMOD <WDTE> to "0," the watchdog timer can be disabled and the binary counter can be cleared.

```
             7   6   5   4   3   2   1   0
WDMOD    ←   0   –   –   –   –   –   –   –    Set <WDTE> to "0".
WDCR     ←   1   0   1   1   0   0   0   1    Writes the disable code (0xB1).
```

19.5.3.2　Enabling control

Set WDMOD <WDTE> to "1".

```
             7   6   5   4   3   2   1   0
WDMOD    ←   1   –   –   –   –   –   –   –    Set <WDTE> to "1".
```

19.5.3.3　Watchdog timer clearing control

Writing the clear code (0x4E) to the WDCR register clears the binary counter and it restarts counting.

```
             7   6   5   4   3   2   1   0
WDCR     ←   0   1   0   0   1   1   1   0    Writes the clear code (0x4E).
```

19.5.3.4　Detection time of watchdog timer

In the case that 2^{21}/fsys is used, set "011" to WDMOD<WDTP[2:0]>.

```
             7   6   5   4   3   2   1   0
WDMOD    ←   1   0   1   1   –   –   –   –
```

20. Key-on Wakeup

20.1 Outline

- TMPM363F10FG has 4 key input, KWUP0 to KWUP3, which can be used for releasing STOP mode or for external interrupts. Note that interrupt processing is executed with one interrupt factor for 4 inputs. (This is programmed in the CG block) Each key input can be configured to be used or not, by programming (KWUPCRn<KEYnEN>).

- The active state of each input can be configured to the rising edge, the falling edge, both edge, the high level or the low level, by programming KWUPCRn<KEYn>.

- An interrupt request is cleared by programming the key interrupt request clear register KWUPCLR in the interrupt processing.

- The KWUP input pins have pull-up functions, which can be switched between static pull-up and dynamic pull-up by programming the KWUPCRn<DPEn>. This programming is needed for each of 4 inputs.

20.2 Block Diagram
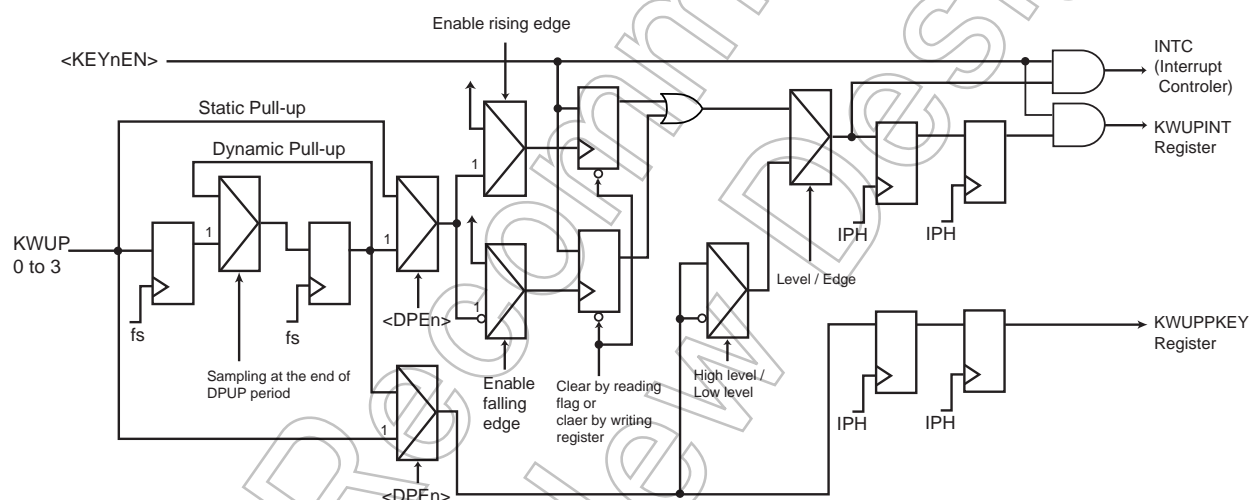
Figure 20-1 Key-on Wakeup Block Diagram

20.3   Register in detail

20.3.1   Register list

Base Address = 0x400F_1000

Register name		Address(Base+)
Control register 0	KWUPCR0	0x0000
Control register 1	KWUPCR1	0x0004
Control register 2	KWUPCR2	0x0008
Control register 3	KWUPCR3	0x000C
Port monitor register	KWUPPKEY	0x0080
Pull-up cycle register	KWUPCNT	0x0084
Interrupt all clear register	KWUPCLR	0x0088
Interrupt monitor register	KWUPINT	0x008C

20.3.2   KWUPCR0 (Control register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DPE0	KEY0			-	-	-	KEY0EN
After reset	0	0	1	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	DPE0	R/W	Selected static pull-up or dynamic pull-up. 0: Static pull-up 1: Dynamic pull-up
6-4	KEY0[2:0]	R/W	Selected the input active status of KWUP0. 000:"Low" level 001:"High" level 010: falling edge 011: rising edge 100: Both edge Except above: Reserved
3-1	-	R	Read as "0".
0	KEY0EN	R/W	Selected enable or disable KWUP interrupt of KWUP0. 0: Disable 1: Enable

20.3.3 KWUPCR1 (Control register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DPE1	KEY1			-	-	-	KEY1EN
After reset	0	0	1	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7	DPE1	R/W	Selected static pull-up or dynamic pull-up. 0: Static pull-up 1: Dynamic pull-up
6-4	KEY1[2:0]	R/W	Selected the input active status of KWUP1. 000:"Low" level 001:"High" level 010: falling edge 011: rising edge 100: Both edge Except above: Reserved
3-1	–	R	Read as "0".
0	KEY1EN	R/W	Selected enable or disable KWUP interrupt of KWUP1. 0: Disable 1: Enable

20.3.4 KWUPCR2 (Control register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DPE2	KEY2			-	-	-	KEY2EN
After reset	0	0	1	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	DPE2	R/W	Selected static pull-up or dynamic pull-up. 0: Static pull-up 1: Dynamic pull-up
6-4	KEY2[2:0]	R/W	Selected the input active status of KWUP2. 000:"Low" level 001:"High" level 010: falling edge 011: rising edge 100: Both edge Except above: Reserved
3-1	−	R	Read as "0".
0	KEY2EN	R/W	Selected enable or disable KWUP interrupt of KWUP2. 0: Disable 1: Enable

20.3.5   KWUPCR3 (Control register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DPE3	KEY3			-	-	-	KEY3EN
After reset	0	0	1	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	DPE3	R/W	Selected static pull-up or dynamic pull-up. 0: Static pull-up 1: Dynamic pull-up
6-4	KEY3[2:0]	R/W	Selected the input active status of KWUP3. 000:"Low" level 001:"High" level 010: falling edge 011: rising edge 100: Both edge Except above: Reserved
3-1	−	R	Read as "0".
0	KEY3EN	R/W	Selected enable or disable KWUP interrupt of KWUP3. 0: Disable 1: Enable

20.3.6 KWUPPKEY (Port monitor register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PKEY3	PKEY2	PKEY1	PKEY0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-0	PKEY3 to PKEY0	R	PORT status 0:"Low" 1:"High" For port status, it can be monitored the external status with KWUPPKEY<PKEYn>. The monitoring is sampled in dynamic pull-up period.

20.3.7 KWUPCNT (Pull-up cycle register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	T2S		T1S		-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5-4	T2S[1:0]	R/W	Dynamic pull-up cycle 00: 256/fs (7.8 ms @ fs = 32.768 kHz) 01: 512/fs (15.6 ms @ fs = 32.768 kHz) 10: 1024/fs (31.3 ms @ fs = 32.768 kHz) 11: 2048/fs (62.5 ms @ fs = 32.768 kHz) Repeats dynamic pull-up operation for the T2 cycle by <T2S[1:0]>.
3-2	T1S[1:0]	R/W	Dynamic pull-up period 00: 2/fs (61.0 μs @fs = 32.768 kHz) 01: 4/fs (122.1 μs @fs = 32.768 kHz) 10: 8/fs (244.1 μs @fs = 32.768 kHz) 11: 16/fs (488.3 μs @fs = 32.768 kHz) Activate the pull-up during T1 period by <TS[1:0]>, the remaining period is not activated.
1-0	-	R	Read as "0".

Dynamic pull-up operation is as following.
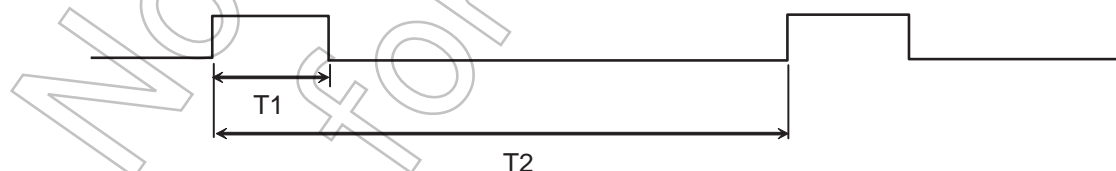
Figure 20-2 Dynamic pull-up operation

Note 1: Activate fs during the dynamic pull-up used.

Note 2: After changed dynamic pull-up setting, wait key input for a T1 period.

20.3.8 KWUPCLR (All interrupt request clear register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	KEYCLR			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	R	Read as "0".
3-0	KEYCLR[3:0]	W	All interrupt request of KWUP is cleared by writing "1010". Read as "0".

20.3.9   KWUPINT (Interrupt monitor register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	KEYINT3	KEYINT2	KEYINT1	KEYINT0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	R	Read as "0".
3-0	KEYINT3 to KEYINT0	R	Interrupt 0:No 1:Yes

When KWUPCRn<KEYnEN>="1" and activated signal into the key-on wakeup port, the <KEYINTn> corresponding channel of KWUPINT will be set to "1" for the interrupt execution.

KWUPINT is read only register, due to the reading this register corresponding bit set to "1" and interrupt request will becleared. That one all can be cleared at once by the KWUPCLR register.

When setting the active status to "Level" input by KWUPCRn<KEYn>, KWUPINT corresponding bit for a key-on wakeup is kept "1" in reading without changing a external input setting function to nothing.

20.4 Key-on Wakeup Operation

TMPM363F10FG has 4 key input pins, KEY0 to KEY3. Program the CGIMCGF<INTLEN> register in the CG to determine whether to use the key inputs for releasing the STOP mode or for normal interrupts. Setting <IN-TLEN> to "1" causes all the key inputs, KEY0 to KEY3, to be used for interrupts for releasing the STOP mode.

Program KWUPCRn<KEYnEN> to enable or disable interrupt inputs for each key input pin. Also, program KWUPCRn<KEYn> to define the active state of each key input pin to be used.

Detection of key inputs is carried out in the KWUP block, and the detection results are notified to the CGIMCGF in the CG as the active high level. Therefore, program CGIMCGF<EMCGL[2:0]> to "001" to determine the detection level to the high level.

Setting CGINCGF<INTLEN> to "0" (default) configures all the input pins, KEY0 to KEY3 to the normal interrupts. In this case, to be detected interrupt request by the CPU, "High" pulse or "High" level signal must be input.

Program KWUPCRn in the same way to enable or disable each key input and define their active states. Writing "1010" to KWUPCLR<KEYCLR[3:0]> during interrupt processing clears all the key interrupt requests.

Note: If two or more key inputs are generated, all the key input requests will be cleared by clearing interrupt requests.

20.5　Pull-up Function

Each key input has the pull-up function and can be programmed by setting the register in the port.

When a static pull-up is set, it can not be depend on KWUPCRn<KEYnEN> and the pull-up be used.

Regarding to dynamic pull-up, refer to "20.3.7 KWUPCNT (Pull-up cycle register)".

20.5.1　In case of using KWUP inputs with pull-up enabled

a.　When you make the first setting after turning the power on (Example : port J7 withe interrupt at both edge)

PJFR2<PJ7F2>	=	1				: The function is set to KWUP3
PJPUP<PJ7UP>	=	1				: Pull-up on control
PJIE<PJ7IE>	=	1				: Enable input function
KWUPCR3<KEY3EN>	=	0				: Disable interrupt
KWUPCR3<KEY3[2:0]>	=	1	0	0		: Change active status (both edge)
Wait completing pull-up						
KWUPCLR<KEYCLR[3:0]>	=	1	0	1	0	: Clear interrupt request
KWUPCR3<KEY3EN>	=	1				: Enable interrupt
CGIMCGF<EMCGL[2:0]>	=	0	0	1		: Change active level ("High" level)
CGIMCGF<INTLEN>	=	1				: Enable INTKWUP

b.　When changing the active state of KWUP input during operation

Interrupt enable clear register 2 [1]	=	1			: Disable INTKWUP
KWUPCR3<KEY3EN>	=	0			: Disable interrupt
KWUPCR3<KEY3[2:0]>	=	0	0	0	: Change active status ("Low" level)
KWUPCLR<KEYCLR[3:0]>	=	1	0	1	0	: Clear interrupt request
KWUPCR3<KEY3EN>	=	1			: Enable interrupt
Interrupt priority setting register <PRI_33>	=	*	*	*	: Set interrupt priority (***= 000 to 111)
Interrupt enable set register 2 [1]	=	1			: Enable INTKWUP

c.　When enabling KWUP input during operation

Interrupt enable clear register 2 [1]	=	1			: Disable INTKWUP
KWUPCR3<KEY3EN>	=	0			: Disable interrupt
KWUPCR3<KEY3[2:0]>	=	*	*	*	: Set active status (***= 000 to 100)
KWUPCLR<KEYCLR[3:0]>	=	1	0	1	0	: Clear interrupt request
KWUPCR3<KEY3EN>	=	1			: Enable interrupt
Interrupt priority setting register <PRI_33>	=	*	*	*	: Set interrupt priority (***= 000 to 111)
Interrupt enable set register 2 [1]	=	1			: Enable INTKWUP

20.5.2　In case of using KWUP inputs with pull-up disabled

a. When you make the first setting after turning the power on

PJFR2<PJ7F2>	=	1			: The function is set to KWUP3
PJPUP<PJ7UP>	=	0			: Pull-up off control
PJIE<PJ7IE>	=	1			: Enable input function
KWUPCR3<KEY3EN>	=	0			: Disable interrupt
KWUPCR3<KEY3[2:0]>	=	0	0	0	: Set active status ("Low" level)
KWUPCLR<KEYCLR[3:0]>	=	1	0	1 0	: Clear interrupt request
KWUPCR3<KEY3EN>	=	1			: Interrupt enable
CGIMCGF<EMCGL[2:0]>	=	0	0	1	: Change active level ("High" level)
CGIMCGF<INTLEN>	=	1			: Enable INTKWUP

b. When changing the active sate of KWUP input during operation

Interrupt enable clear register 2 [1]	=	1			: Disable INTKWUP
KWUPCR3<KEY3EN>	=	0			: Disable interrupt
KWUPCR3<KEY3[2:0]>	=	*	*	*	: Change active status
					(***= 000 to 100)
KWUPCLR<KEYCLR[3:0]>	=	1	0	1 0	: Clear interrupt request
KWUPCR3<KEY3EN>	=	1			: Enable interrupt
Interrupt priority setting register <PRI_33>	=	*	*	*	: Set interrupt priority
					(***= 000 to 111)
Interrupt enable set register 2 [1]	=	1			:Enable INTKWUP

c. When enabling KWUP input during operation

Interrupt enable clear register 2 [1]	=	1			: Disable INTKWUP
KWUPCR3<KEY3EN>	=	0			: Disable interrupt
KWUPCR3<KEY3[2:0]>	=	*	*	*	: Change active status
					(***= 000 to 100)
KWUPCLR<KEYCLR[3:0]>	=	1	0	1 0	: Clear interrupt request
KWUPCR3<KEY3EN>	=	1			: Enable interrupt
Interrupt priority setting register <PRI_33>	=	*	*	*	: Set interrupt priority
					(***= 000 to 111)
Interrupt enable set register 2 [1]	=	1			: Enable INTKWUP

20.6   KWUP input Detection Timing

1. PJPUP<PJnUP>="1", KWUPCRn<DPEn>="0" with always pull-up

    The active state of each key input can be defined to the high or low level or to the rising or falling edges by setting KWUPCRn<KEYn>. The active state of key inputs are continuously detected.

2. PJPUP<PJnUP>="1", KWUPCRn<DPEn>="1" with dynamic pull-up

    Detection of the active state of each key input (interrupt detection) is carried out at the edge one-clock before fs at the end of the T1 period. Therefore, a key input not shorter than the T2 period is needed.There is a delay up to the T2 period before key input detection. The figure below shows an example of defining the active state to the falling edge.
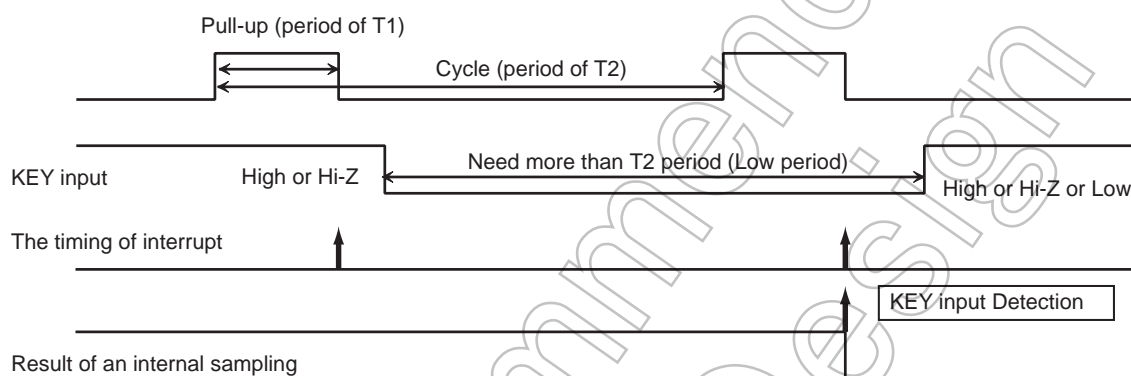
21. Backup module

21.1 Features

The BACKUP mode, one of the system operation modes, enable the MCU to operate in the low power consumption. By cutting electricity to the entire block, such as CPU or other peripheral IPs, other than the backup module, this mode significantly reduces power consumption.

The BACKUP mode contains two modes :

- BACKUP SLEEP mode (enabling low frequency oscillator)
- BACKIUP STOP mode (disabling low frequency oscillator)

21.2 Block Diagram
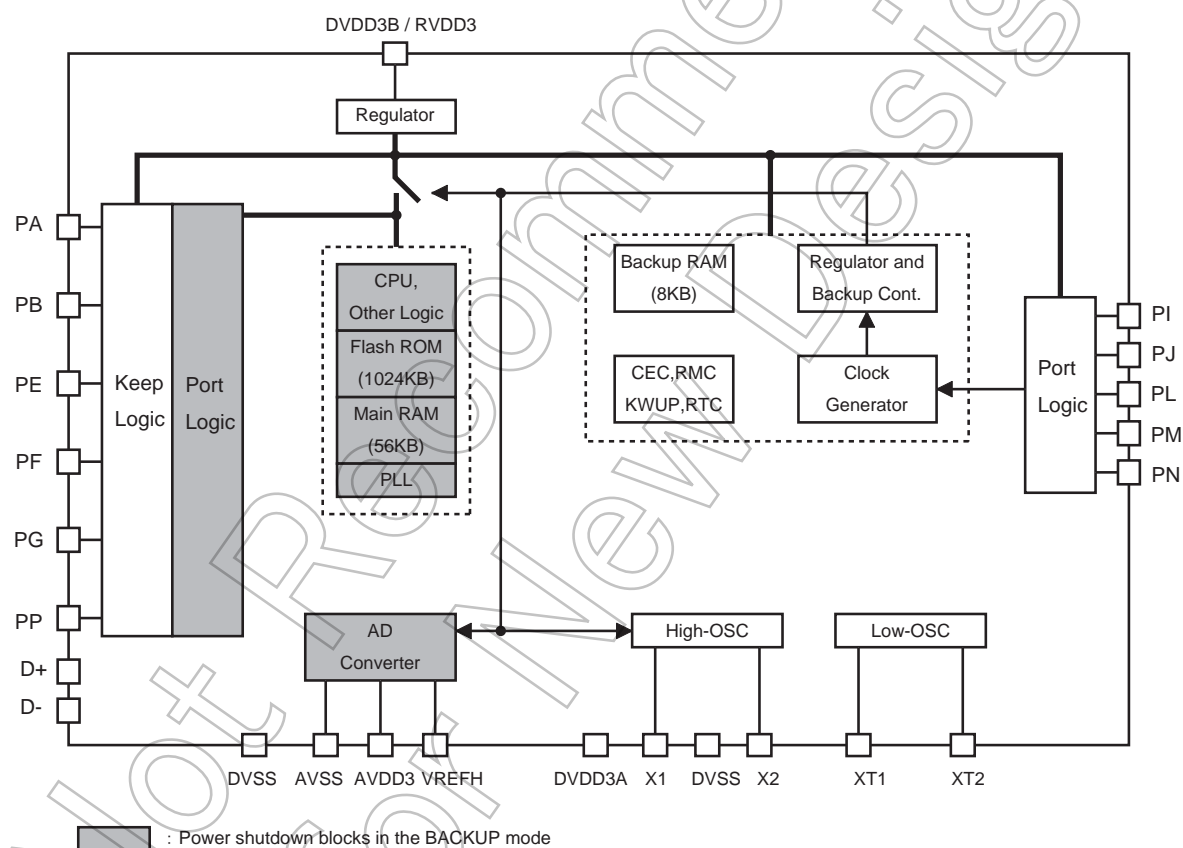
Figure 21-1 Power shutdown blocks in the backup mode

21.3 BACKUP Mode Operation

The BACKUP mode only corresponds to the single chip mode (MCU starts from the built-in flash memory after reset). In the BACKUP mode, single boot mode (MCU starts from built-in boot ROM after reset) is not supported.

21.3.1 Operable peripherals in the BACKUP mode

- In the BACKUP SLEEP mode

  Port output, Key-on-wakeup (KWUP), CEC, remote control circuit (RMC), real-time clock (RTC), low speed oscillator, data in BACKUP RAM (8KB)
- In the BACKUP STOP mode

  Port output, Key-on-wakeup (KWUP), data in BACKUP RAM (8KB)

21.3.1.1 Transition to the BACKUP mode

Figure 21-2 shows the state transition between NORMAL mode, SLOW mode and BACKUP mode (BACKUP SLEEP and BACKUP STOP). The BACKUP mode (BACKUP SLEEP and BACKUP STOP) will return the preceding mode of transition by release source.

In addition, each mode changes to the reset processing routine if the reset operation occurs.
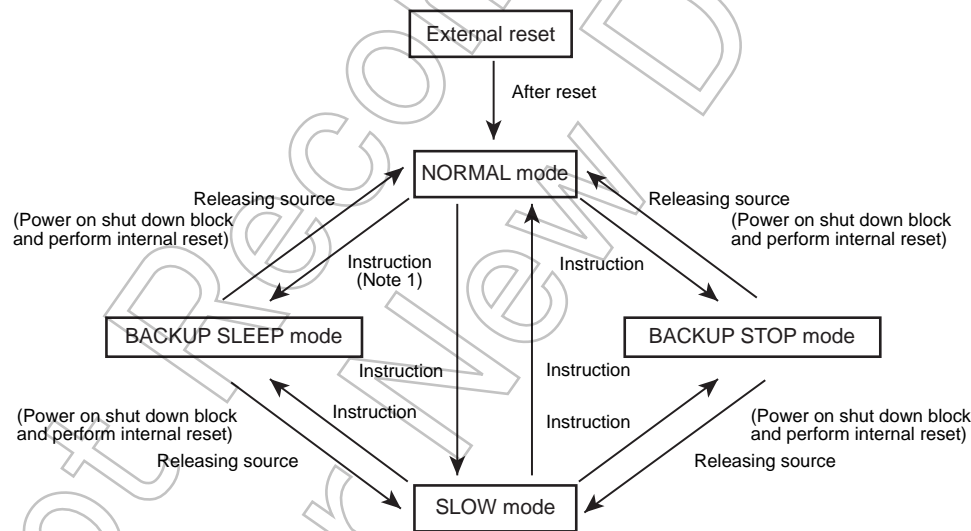
Figure 21-2 BACKUP Mode Transition Diagram

Note 1: In case that low-speed oscillator is stopped in the NORMAL mode, make sure to start the low-speed oscillation and confirm the stable oscillation before changing to BACKUP SLEEP mode.

Note 2: The program for changing to the BACKUP mode must be executed in the built-in flash ROM or built-in RAM.

Note 3: Do not release BACKUP mode by reset.

21.3.1.2   Backup Transition Flow

(1)   Preparing for BACKUP mode

The preparation program for changing to the backup mode must be executed in the built-in flash ROM or built-in RAM.

1. Stopping peripherals and saving data

    Both in the NORMAL mode and SLOW mode stop peripheral function including DMAC, SMC and WDT. In case of transition to BACKUP SLEEP mode, no need to stop peripheral functions (CEC, RMC, RTC and KWUP) which are used in BACKUP SLEEP mode. It is necessary to save data to preserved in BACKUP RAM. BACKUP RAM is used only 8KB data from 0x2000_E000 to 0x2000_FFFF.

2. Prohibit the interrupt

    To prevent from obstruction a transition to BACKUP mode, interrupt request set to disable if needed. It is note that $\overline{\text{NMI}}$ interrupt and INTRTC interrupt request cannot be disabled so that these interrupt requests must be avoided in advance.

3. Setting of port keep function (CGSTBYCR<PTKEEP>)

    Port keep function retains the port status of the momentary when CGSTBCR<PTKEEP> is set to "1". Object ports are A to H, K, O, P, SWDIO, $\overline{\text{NMI}}$. Port keep function is capable of retaining input enable / disable, port 0 / 1 output status and on / off status of pull-up / pull-down register.

    By these settings made before the transition to the BACKUP mode, port keep function can hold the port status. When using the port keep function, port register of each port must be set properly.

    The input / output status of port I, J, L, M and N are depend on the port register regardless the port keep function. The interrupt of BACKUP mode is set by using these ports.

    All unnecessary ports must be set to disable by the input enable control register.

4. Clock related setting and warm up time

    Stop PLL circuit by setting CGOSCCR<PLLON>="0". Set high-speed clock to fc (1/1) by CGSYSCR<GEAR[2:0]>="000". Using BACKUP SLEEP is needed for starting low-speed oscillator by CGOSCCR<XTEN>.

    In addition, it is necessary to setting warm up time returning from BACKUP mode by CGOSCCR<WUPT[11:0]><WUPTL[2:0]>. The warm up time is referred to the section "Clock / Mode control".

(2) Transition to BACKUP mode

1. Setting modes and clearing release source of BACKUP mode

   By the CGSTBYCR<STBY> register, set to the BACKUP STOP mode or BACKUP SLEEP mode.

2. Transition to the BACKUP mode

   Clear the interrupt which releases from BACKUP mode, then execute WFI instruction

Precautions for the use of the BACKUP mode (about debug tool)
The communication with debug tool is disconnected, if MCU changes to the BACKUP mode. In this case, it is necessary to reconnect to debug tool.

(3) Returning from backup mode (Releasing)

1. Releasing source of BACKUP mode

   Releasing source of BACKUP STOP and BACKUP SLEEP shown as below.

BACKUP mode	Releasing source of BACKUP mode
BACKUP STOP	INT0 to 4, INTKWUP (Static)
BACKUP SLEEP	INT0 to 4, INTKWUP (Dynamic / Static), INTRTC, INTCECRX, INTRMCRX0

2. Releasing operation by releasing source of BACKUP mode

   If the event of releasing source are received, regulator starts to supply power to the shut down block. Depending on the returned modes, high-speed oscillator and low-speed oscillator will start operation.

   The warm up timer will starts when the oscillation becomes stable. During warm up time, internal reset signal of power shut down block which returned from BACKUP mode is continuing active level. Internal reset is cleared after warm up time has elapsed, and then MCU returns to the preceding mode of BACKUP mode.

Precaution after BACKUP mode released
· By reading CGRSTFLG register, it can be found which reset are occurred.
· Make sure to perform the port A, B, E, F, G, and P setting before releasing port keep function by (CGSTBYCR<PTKEEP>="0").

TOSHIBA

21.3.1.3   Transition Flowchart
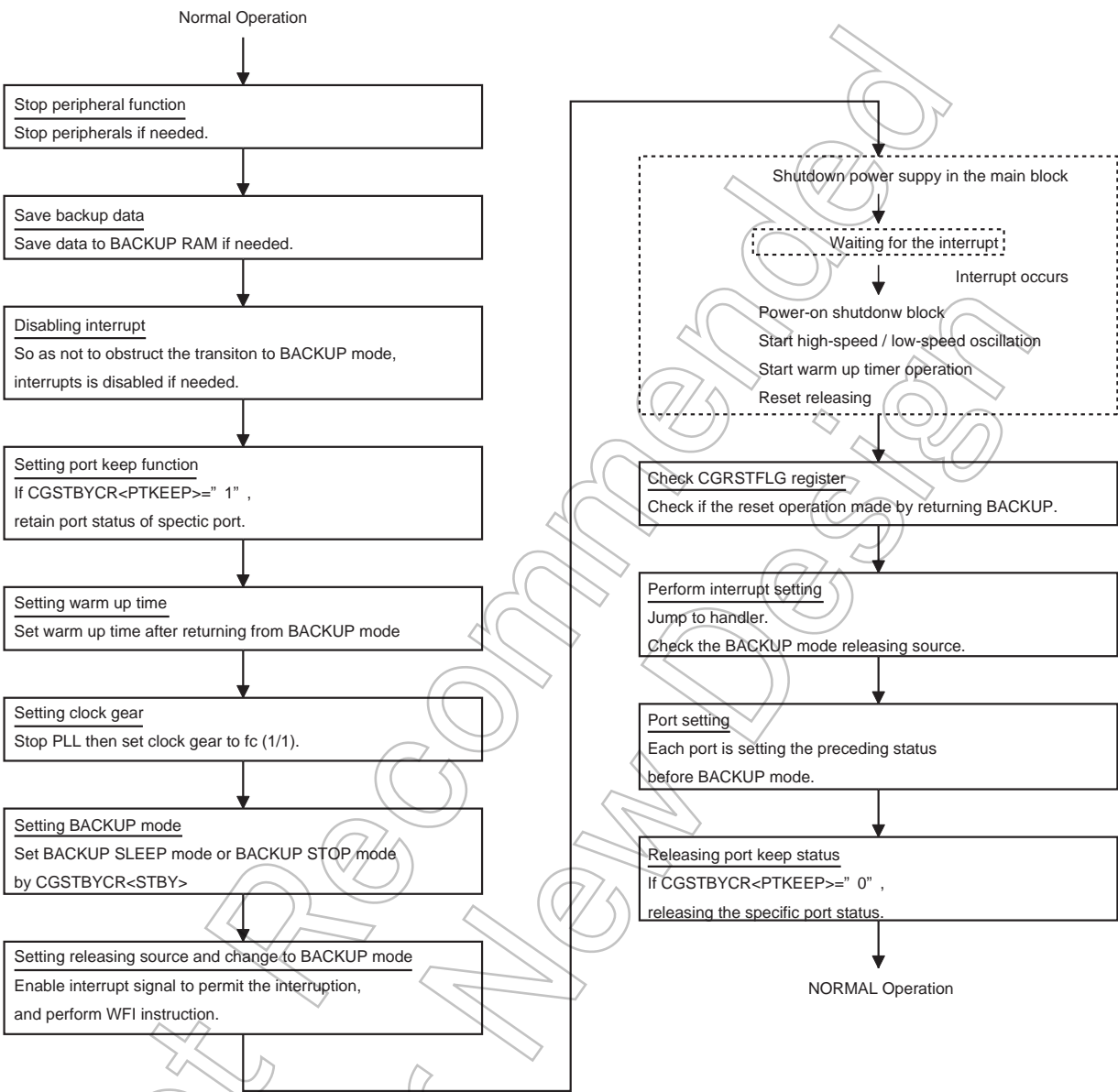
Figure 21-3 Stare transition flowchart

21.3.1.4   BACKUP Mode Timing Chart
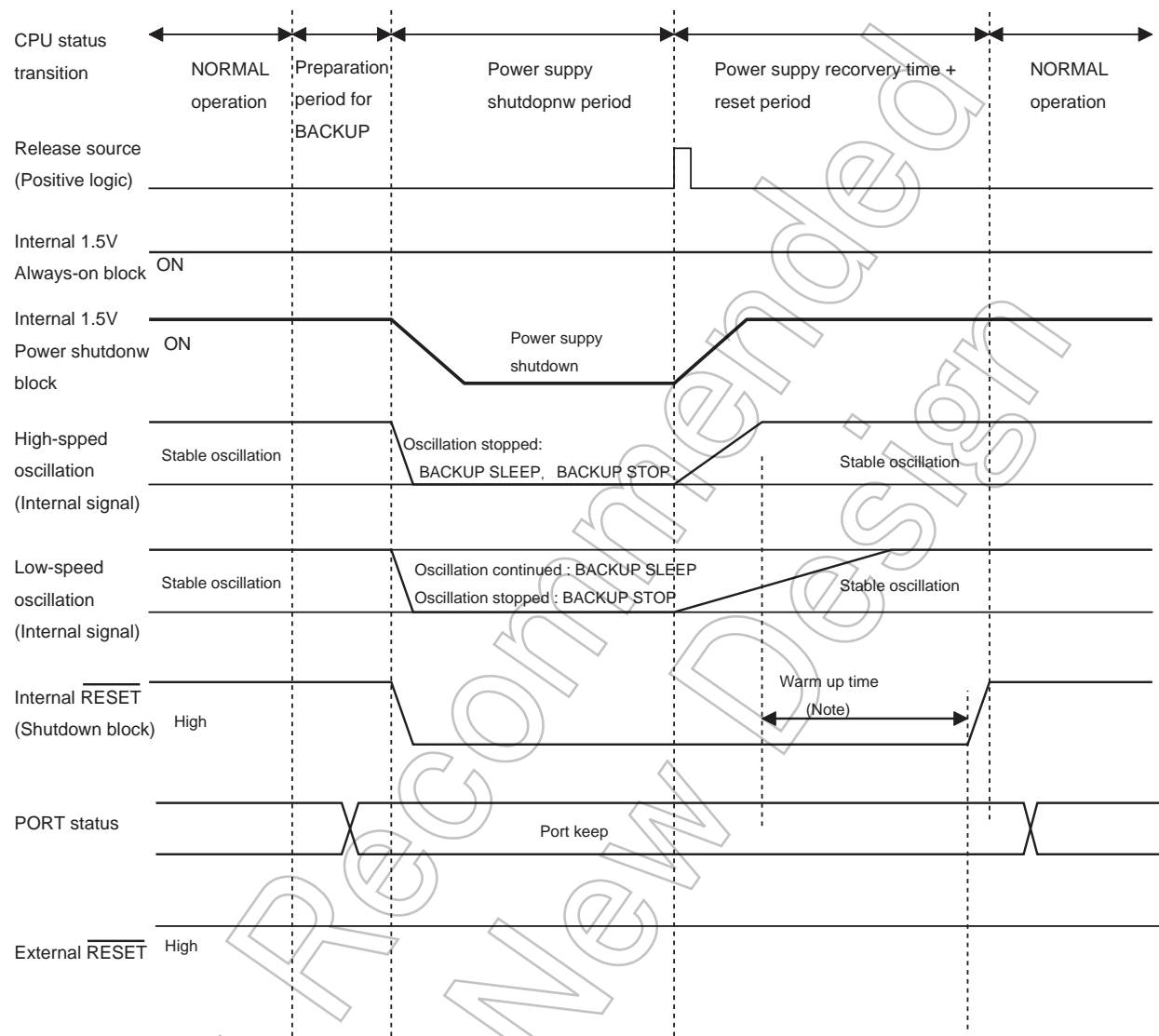
Figure 21-4 BACKUP mode sequence

Note:"Figure 21-4 BACKUP mode sequence" shows the transition modes ; NORMAL→BACKUP
       STOP→NORMAL or NORMAL→BACKUP SLEEP→NORMAL. In this transition, a clock for warm
       up counter is used high-speed oscillator. The warm up time must be set 500Éþs or more.

22. Analog / Digital Converter (ADC)

22.1 Outline

TMPM363F10FG has a 10-bit, sequential-conversion analog / digital converter (AD converter). This AD convert-er is equipped with 8 analog input channel.

These 8 analog input channels (pins AIN0~AIN7) are also used as input / output ports.

Note 1: To assure conversion accuracy, the specified value must be set to the ADCBAS register.

Note 2: If it is necessary to reduce a power current by operating the TMPM363F10FG in IDLE or STOP mode and if either case shown below is applicable, you must first stop the AD converter and then execute the instruction to put the TMPM363F10FG into standby mode.

    1.   TheTMPM363F10FG must be put into IDLE mode when ADMOD1<I2AD> is "0".

    2.   The TMPM363F10FG must be put into STOP mode.

22.2 Configuration

Figure 22-1 shows the block diagram of this AD converter.
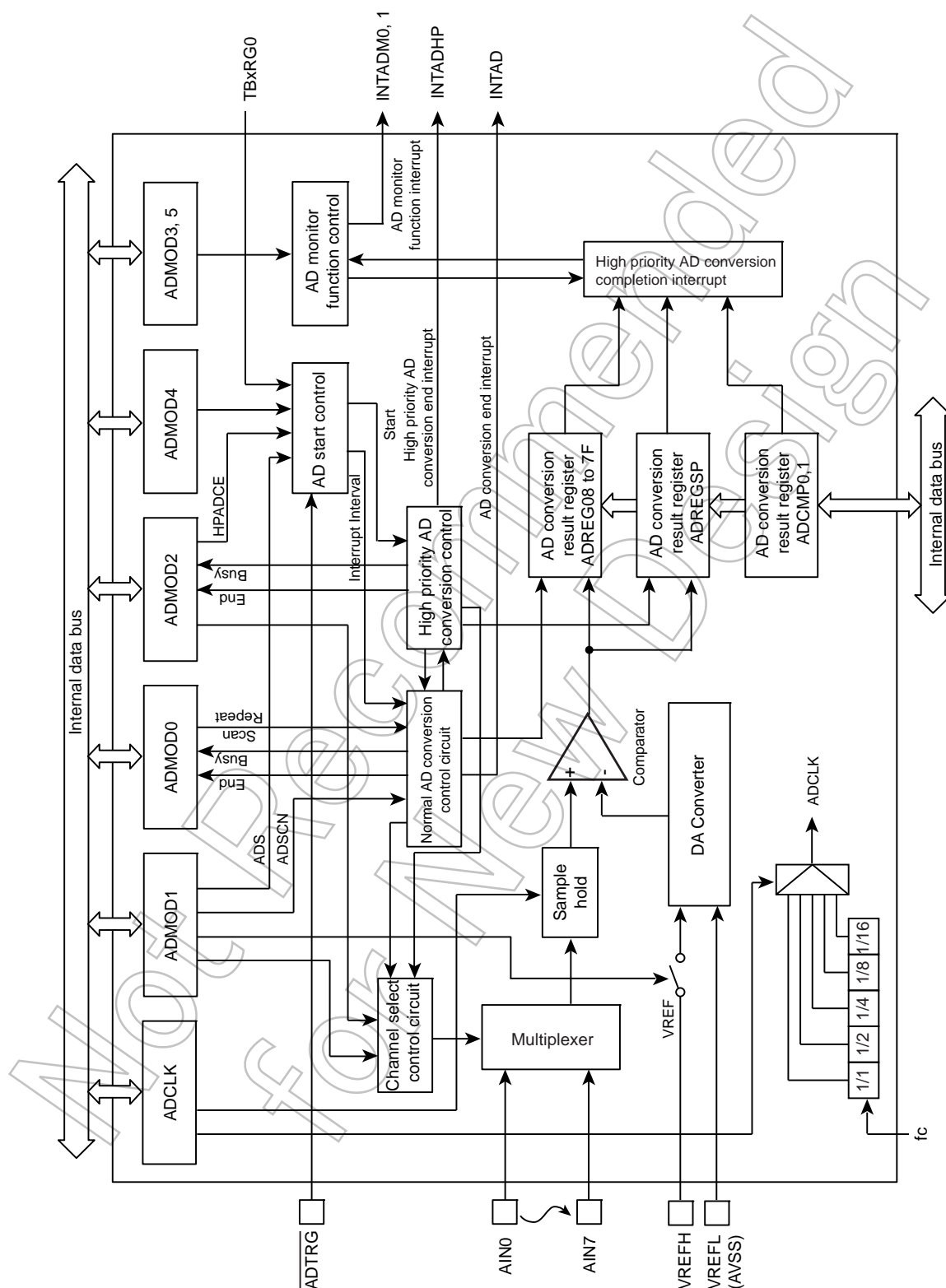
Figure 22-1 AD Converter Block Diagram

22.3　Registers

22.3.1　Register list

The control registers and addresses of the AD converter are as follows.

The AD converter is controlled by the AD mode control registers (ADMOD0 through ADMOD5). The result of AD conversion is stored in the eight AD conversion result registers, ADREG08 through ADREG7F. The highest-priority conversion result is stored in the register ADREGSP.

To assure conversion accuracy, the specified value must be set to the ADCBAS register.

Base Address = 0x400F_0000

Register name		Address (Base+)
Conversion Clock Setting Register	ADCLK	0x0000
Mode Control Register 0	ADMOD0	0x0004
Mode Control Register 1	ADMOD1	0x0008
Mode Control Register 2	ADMOD2	0x000C
Mode Control Register 3	ADMOD3	0x0010
Mode Control Register 4	ADMOD4	0x0014
Mode Control Register 5	ADMOD5	0x0018
Conversion Accuracy Setting Register	ADCBAS	0x0020
Reserved	-	0x0024
Reserved	-	0x0028
Conversion Result Register 08	ADREG08	0x0030
Conversion Result Register 19	ADREG19	0x0034
Conversion Result Register 2A	ADREG2A	0x0038
Conversion Result Register 3B	ADREG3B	0x003C
Conversion Result Register 4C	ADREG4C	0x0040
Conversion Result Register 5D	ADREG5D	0x0044
Conversion Result Register 6E	ADREG6E	0x0048
Conversion Result Register 7F	ADREG7F	0x004C
Conversion Result Register SP	ADREGSP	0x0050
Conversion Result Comparison Register 0	ADCMP0	0x0054
Conversion Result Comparison Register 1	ADCMP1	0x0058

Note:Access to the "Reserved" address is prohibited.

22.3.2 ADCBAS (Conversion Accuracy Setting Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADCBAS							
After reset	0	0	1	1	1	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7-0	ADCBAS[7:0]	R/W	Write as "0x58".

Note: To assure conversion accuracy, the specified value (0x0000_0058) must be set to the ADCBAS register.

22.3.3 ADCLK (Conversion Clock Setting Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TSH				-	ADCLK		
After reset	1	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7-4	TSH[3:0]	R/W	Select the AD sample hold time. 1000: 8 conversion clock 1001: 16 conversion clock 1010: 24 conversion clock 1011: 32 conversion clock 0011: 64 conversion clock 1100: 128 conversion clock 1101: 512 conversion clock Others : Reserved
3	–	R	Read as "0".
2-0	ADCLK[2:0]	R/W	Select the AD conversion clock 000: fc 001: fc/2 010: fc/4 011: fc/8 100: fc/16 Others : Reserved

A clock count required for conversion is 46 clocks at the minimum.

Examples of sample hold time and conversion time as shown as below.

(Example : If fc = 64MHz)

<TSH[3:0]>	Sample hold time	Conversion time(<ADCLK[2:0]> setting)				
		000 (fc)	001 (fc/2)	010 (fc/4)	011 (fc/8)	100 (fc/16)
1000 (8 conversion clock)	0.125 μs	Reserved	1.44μs	2.88μs	5.75μs	11.5μs
1001 (16 conversion clock)	0.25 μs	Reserved	1.69μs	3.38μs	6.75μs	13.5μs
1010 (24 conversion clock)	0.375 μs	Reserved	1.94μs	3.88μs	7.75μs	15.5μs
1011 (32 conversion clock)	0.5 μs	Reserved	2.19μs	4.38μs	8.75μs	17.5μs
0011 (64 conversion clock)	1.0 μs	Reserved	3.19μs	6.38μs	12.75μs	25.5μs
1100 (128 conversion clock)	2.0 μs	Reserved	5.19μs	10.38μs	20.75μs	41.5μs
1101 (512 conversion clock)	8.0 μs	Reserved	17.19μs	34.38μs	68.75μs	137.5μs

(Example : If fc = 40MHz)

<TSH[3:0]>	Sample hold time	Conversion time(<ADCLK[2:0]> setting)				
		000 (fc)	001 (fc/2)	010 (fc/4)	011 (fc/8)	100 (fc/16)
1000 (8 conversion clock)	0.2 µs	1.15µs	2.3µs	4.6µs	9.2µs	18.4µs
1001 (16 conversion clock)	0.4 µs	1.35µs	2.7µs	5.4µs	10.8µs	21.6µs
1010 (24 conversion clock)	0.6 µs	1.55µs	3.1µs	6.2µs	12.4µs	24.8µs
1011 (32 conversion clock)	0.8 µs	1.75µs	3.5µs	7.0µs	14.0µs	28.0µs
0011 (64 conversion clock)	1.6 µs	2.55µs	5.1µs	10.2µs	20.4µs	40.8µs
1100 (128 conversion clock)	3.2 µs	4.15µs	8.3µs	16.6µs	33.2µs	66.4µs
1101 (512 conversion clock)	12.8 µs	13.75µs	27.5µs	55.0µs	110.0µs	220.0µs

Note 1: Do not change the setting of the AD conversion clock during AD conversion.

Note 2: Please use at ADCLK≤40MHz. When fosc is 8MHz and PLL is 8 times, fc is 64MHz, in this case, specify except ADCLK<ADCLK>="000".

22.3.4   ADMOD0 (Mode Control Register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	EOCFN	ADBFN	-	ITM		REPEAT	SCAN	ADS
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	EOCFN	R	Normal AD conversion completion flag (note1) 0: Before or during conversion 1: Completion
6	ADBFN	R	Normal AD conversion busy flag 0: Conversion stop 1: During conversion
5	−	R	Read as "0".
4-3	ITM[1:0]	R/W	Specify interrupt in fixed channel repeat conversion mode (refer to the table below and note 2)
2	REPEAT	R/W	Specify repeat mode 0: Single conversion mode 1: Repeat conversion mode
1	SCAN	R/W	Specify scan mode 0: Fixed channel mode 1: Channel scan mode
0	ADS	R/W	Start AD conversion start (note3) 0: Don't care 1: Start conversion Always read as "0".

Specify AD conversion interrupt in fixed channel repeat conversion mode

<ITM[1:0]>	Fixed channel repeat conversion mode <SCAN> = "0", <REPEAT> = "1"
00	Generate interrupt once every single conversion.
01	Generate interrupt once every 4 conversions.
10	Generate interrupt once every 8 conversions.
11	Setting prohibited

Note 1: This bit is "0" cleared when it is read.

Note 2: It is valid only when it's specified in the fixed channel repeat mode (<REPEAT> ="1", <SCAN> = "0").

Note 3: Conversion must be started after setting the mode.

Note 4: When DMA transfer is executed by utilizing AD conversion completion interrupts, perform software reset first. Then startup a DMA operation (DMA request wait mode), and start ADC setting.

22.3.5 ADMOD1 (Mode Control Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	VREFON	I2AD	ADSCN	-	ADCH			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7	VREFON	R/W	VREF application control(Note1 and Note2) 0: OFF 1: ON
6	I2AD	R/W	Specify operation mode in IDLE mode 0: STOP 1: Operation
5	ADSCN	R/W	Specify operation mode in channel scan mode 0: 4 channel scan 1: 8 channel scan
4	–	R/W	Write as "0".
3-0	ADCH[3:0]	R/W	Select analog input channel (Refer to the below table.)

Select Analog Input Channel

ADMOD0<SCAN> ADMOD1<ADCH[3:0]>	0 Fixed channel	1 Channel scan (<ADSCN> = 0)	1 Channel scan (<ADSCN> = 1)
0000	AIN0	AIN0	AIN0
0001	AIN1	AIN0 to AIN1	AIN0 to AIN1
0010	AIN2	AIN0 to AIN2	AIN0 to AIN2
0011	AIN3	AIN0 to AIN3	AIN0 to AIN3
0100	AIN4	AIN4	AIN0 to AIN4
0101	AIN5	AIN4 to AIN5	AIN0 to AIN5
0110	AIN6	AIN4 to AIN6	AIN0 to AIN6
0111	AIN7	AIN4 to AIN7	AIN0 to AIN7
1000	Reserved		
1001			
1010			
1011			
1100			
1101			
1110			
1111			

Note 1: Before starting AD conversion, write "1" to the <VREFON> bit, wait for 3µs during which time the internal reference voltage should stabilize, and then write "1" to the ADMOD0<ADS>.

Note 2: Set <VREFON> to "0" to go into standby mode upon completion of AD conversion.

22.3.6   ADMOD2 (Mode Control Register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	EOCFHP	ADBFHP	HPADCE	-	HPADCH			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	EOCFHP	R	Top-priority AD conversion completion flag (Note1) 0: Before or during conversion 1: Completion
6	ADBFHP	R	Top-priority AD conversion BUSY flag 0: During conversion halts 1: During conversion
5	HPADCE	R/W	Activate top-priority conversion 0: Don't care 1: Start conversion "0" is always read.
4	−	R/W	Write as "0".
3-0	HPADCH[3:0]	R/W	Select analog input channel when activating top-priority conversion. (See the table below)

<HPADCH[3:0]>	Analog input channel when executing top-priority conversion
0000	AIN0
0001	AIN1
0010	AIN2
0011	AIN3
0100	AIN4
0101	AIN5
0110	AIN6
0111	AIN7
1000	Reserved
1001	
1010	
1011	
1100	
1101	
1110	
1111	

Note 1: This bit is "0" cleared when it is read.

Note 2: Specify <HDADCE> after selecting channel.

22.3.7   ADMOD3 (Mode Control Register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	ADOBIC0	ADREGS0				ADOBSV0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7	–	R/W	Write as "0".
6	–	R	Read as "0".
5	ADOBIC0	R/W	Set the AD monitor function interrupt 0 0 : If the value of the conversion result is smaller than the comparison register 0, an interrupt is generated. 1 : If the value of the conversion result is bigger than the comparison register 0, an interrupt is generated.
4 to 1	ADREGS0[3:0]	R/W	Select a target conversion result register when using the AD monitor function 0 (See the below table).
0	ADOBSV0	R/W	AD monitor function 0 0: Disable 1: Enable

<ADREGS0[3:0]>	Conversion result register to be compared	<ADREGS0[3:0]>	Conversion result register to be compared
0000	ADREG08	0100	ADREG4C
0001	ADREG19	0101	ADREG5D
0010	ADREG2A	0110	ADREG6E
0011	ADREG3B	0111	ADREG7F
-	-	1xxx	ADREGSP

22.3.8 ADMOD4 (Mode Control Register 4)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	HADHS	HADHTG	ADHS	ADHTG	-	-	ADRST	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7	HADHS	R/W	H/W source for activating top-priority AD conversion 0: External trigger 1: Match with timer register 0 (TB5RG0)
6	HADHTG	R/W	H/W for activating top-priority AD conversion 0: Disable 1: Enable
5	ADHS	R/W	H/W source for activating normal AD conversion (note1) 0: External trigger 1: Match with timer register (TB6RG0)
4	ADHTG	R/W	H/W for activating normal AD conversion 0: Disable 1: Enable
3-2	–	R	Read as "0".
1-0	ADRST[1:0]	W	Overwriting "10" with "01" allows ADC to be software reset.(note 2)

Note 1: The external trigger cannot be used for H/W activation of AD conversion when it is used for H/W activation of top priority AD conversion.

Note 2: A software reset initializes all the registers except for ADCLK<ADCLK>.

Note 3: The disables the external trigger used for H/W activation. Therefore "0" cannot be set to <HADHS> and <ADHS>.

22.3.9   ADMOD5 (Mode Control Register 5)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	ADOBIC1	ADREGS1				ADOBSV1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5	ADOBIC1	R/W	Set the AD monitor function interrupt 1. 0: If the value of the conversion result is smaller than the comparison register 1, an interrupt is generated. 1: If the value of the conversion result is bigger than the comparison register 1, an interrupt is generated.
4-1	ADREGS1[3:0]	R/W	Select a target conversion result register when using the AD monitor function 1 (See the below table).
0	ADOBSV1	R/W	AD monitor function1 0: Disable 1: Enable

<ADREGS1[3:0]>	Conversion result register to be compared	<ADREGS1[3:0]>	Conversion result register to be compared
0000	ADREG08	0100	ADREG4C
0001	ADREG19	0101	ADREG5D
0010	ADREG2A	0110	ADREG6E
0011	ADREG3B	0111	ADREG7F
−	−	1xxx	ADREGSP

22.3.10 ADREG08 (Conversion Result Register 08)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADR0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADR0		-	-	-	-	OVR0	ADR0RF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as "0".
15-6	ADR0[9:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to the Table 22-2 in 22.4.5.7.
5-2	–	R	Read as "0".
1	OVR0	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADR0>, "1" is set. This bit is "0" cleared when it is read.
0	ADR0RF	R	AD conversion result storage flag 0:Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

22.3.11 ADREG19 (AD Conversion Result Register 19)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADR1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADR1		-	-	-	-	OVR1	ADR1RF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-6	ADR1[9:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to the Table 22-2 in 22.4.5.7.
5-2	−	R	Read as "0".
1	OVR1	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADR1>, "1" is set. This bit is "0" cleared when it is read.
0	ADR1RF	R	AD conversion result storage flag 0:Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note:Access to this register must be a half word or a word access.

22.3.12   ADREG2A (AD Conversion Result Register 2A)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADR2							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADR2		-	-	-	-	OVR2	ADR2RF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-6	ADR2[9:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to the Table 22-2 in 22.4.5.7.
5-2	−	R	Read as "0".
1	OVR2	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADR2>, "1" is set. This bit is "0" cleared when it is read.
0	ADR2RF	R	AD conversion result storage flag 0:Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

22.3.13　ADREG3B (AD Conversion Result Register 3B)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol				ADR3				
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADR3		-	-	-	-	OVR3	ADR3RF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-6	ADR3[9:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to the Table 22-2 in 22.4.5.7.
5-2	−	R	Read as "0".
1	OVR3	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADR3>, "1" is set. This bit is "0" cleared when it is read.
0	ADR3RF	R	AD conversion result storage flag 0:Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

22.3.14 ADREG4C (AD Conversion Result Register 4C)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADR4							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADR4		-	-	-	-	OVR4	ADR4RF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as "0".
15-6	ADR4[9:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to the Table 22-2 in 22.4.5.7.
5-2	–	R	Read as "0".
1	OVR4	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADR4>, "1" is set. This bit is "0" cleared when it is read.
0	ADR4RF	R	AD conversion result storage flag 0:Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

22.3.15   ADREG5D (AD Conversion Result Register 5D)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADR5							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADR5		-	-	-	-	OVR5	ADR5RF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-6	ADR5[9:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to the Table 22-2 in 22.4.5.7.
5-2	−	R	Read as "0".
1	OVR5	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADR5>, "1" is set. This bit is "0" cleared when it is read.
0	ADR5RF	R	AD conversion result storage flag 0:Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

22.3.16 ADREG6E (AD Conversion Result Register 6E)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADR6							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADR6		-	-	-	-	OVR6	ADR6RF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-6	ADR6[9:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to the Table 22-2 in 22.4.5.7.
5-2	−	R	Read as "0".
1	OVR6	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADR6>, "1" is set. This bit is "0" cleared when it is read.
0	ADR6RF	R	AD conversion result storage flag 0:Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

22.3.17   ADREG7F (AD Conversion Result Register 7F)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADR7							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADR7		-	-	-	-	OVR7	ADR7RF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-6	ADR7[9:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to the Table 22-2 in 22.4.5.7.
5-2	−	R	Read as "0".
1	OVR7	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADR7>, "1" is set. This bit is "0" cleared when it is read.
0	ADR7RF	R	AD conversion result storage flag 0:Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

22.3.18 ADREGSP (AD Conversion Result Register SP)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADRSP							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADRSP		-	-	-	-	OVRSP	ADRSPRF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as "0".
15-6	ADRSP[9:0]	R	AD conversion result Top-priority AD conversion result is stored.
5-2	–	R	Read as "0".
1	OVRSP	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADRSP>, "1" is set. This bit is "0" cleared when it is read.
0	ADRSPRF	R	AD conversion result storage flag 0:Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

22.3.19   ADCMP0 (AD Conversion Result Comparison Register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADCOM0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADCOM0		-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-6	ADCOM0[9:0]	R/W	When AD monitor function 0 is enabled, it sets a value to be compared with the value of the conversion result register specified by ADMOD3<ADREGS0>.
5-0	−	R	Read as "0".

Note: To write values into this register, the AD monitor function 0 must be disabled (AD-MOD3<ADOBSV0> ="0").

22.3.20   ADCMP1 (AD Conversion Result Comparison Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ADCOM1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADCOM1		-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-6	ADCOM1[9:0]	R/W	When AD monitor function 0 is enabled, it sets a value to be compared with the value of the conversion result register specified by ADMOD5<ADREGS1>.
5-0	−	R	Read as "0".

Note: To write values into this register, the AD monitor function 1 must be disabled (AD-MOD5<ADOBSV1> ="0").

22.4 Description of Operations

22.4.1 Analog Reference Voltage

The "High" level of the analog reference voltage shall be applied to the VRFEH pin, and the "Low" shall be applied to the VREFL pin.

To start AD conversion, make sure that you first write "1" to the <VREFON> bit, wait for 3 μs during which time the internal reference voltage should stabilize, and then write "1" to the ADMOD0<ADS> bit.

By writing "0" to the ADMOD1<VREFON> bit, a switched-on state of VREFH - VREFL can be turned into a switched -off state. To switch to the power-consumption mode, set "0" to the <VREFON> bit after conversion.

Note:VREFL and AVSS are shared by TMPM363F10FG.

22.4.2 AD Conversion Mode

Two types of AD conversion are supported: normal AD conversion and top-priority AD conversion.

For normal AD conversion, the following four operation modes are supported.

22.4.2.1 Normal AD conversion

For normal AD conversion, the following four operation modes are supported and the operation mode is selected with the ADMOD0<REPEAT, SCAN>.

- Fixed channel single conversion mode
- Channel scan single conversion mode
- Fixed channel repeat conversion mode
- Channel scan repeat conversion mode

(1) Fixed channel single conversion mode

If ADMOD0<REPEAT, SCAN> is set to "00", "AD conversion is performed in the fixed channel single conversion mode.

In this mode, AD conversion is performed once for one channel selected. After AD conversion is completed, ADMOD0<EOCFN> is set to "1", ADMOD0<ADBFN> is cleared to "0", and the AD conversion completion interrupt request (INTAD) is generated. <EOCFN> is cleared to "0" upon read.

(2) Channel scan single conversion mode

If ADMOD0<REPEAT, SCAN> is set to "01", "AD conversion is performed in the channel scan single conversion mode.

In this mode, AD conversion is performed once for each scan channel selected. After AD scan conversion is completed, ADMOD0<EOCFN> is set to "1", ADMOD0<ADBFN> is cleared to "0", and the conversion completion interrupt request (INTAD) is generated. <EOCFN> is cleared to "0".

(3)   Fixed channel repeat conversion mode

If ADMOD0<REPEAT, SCAN> is set to "10", AD conversion is performed in fixed channel re-peat conversation mode.

In this mode, AD conversion is performed repeatedly for one channel selected. After AD conversion is completed, ADMOD0<EOCFN> is set to "1". ADMOD0<ADBFN> is not cleared to "0". It remains at "1". The timing with which the conversion completion interrupt request (INTAD) is generated can be selected by setting ADMOD0<ITM> to an appropriate setting. <EOCFN> is set with the same timing as this interrupt INTAD is generated.

By reading <EOCFN>, it is cleared to "0".

(4)   Channel scan repeat conversion mode

If ADMOD0<REPEAT, SCAN> is set to "11", AD conversion is performed in the channel scan repeat conversion mode.

In this mode, AD conversion is performed repeatedly for a scan channel selected. Each time one AD scan conversion is completed, ADMOD0<EOCFN> is set to "1", and the conversion completion interrupt request (INTAD) is generated. ADMOD0<ADBFN> is cleared to "0". It remains at "1". <EOCFN> is cleared to "0" upon read.

22.4.2.2   Top-priority AD conversion

By interrupting ongoing normal AD conversion, top-priority AD conversion can be performed.

The fixed-channel single conversion is automatically selected, irrespective of the ADMOD0<RE-PEAT,SCAN> setting. When conditions to start operation are met, a conversion is performed just once for a channel designated by ADMOD2<HPADCH>. When conversion is completed, the top-priority AD conversion completion interrupt (INTADHP) is generated, and ADMOD2<EOCFHP> showing the completion of AD conversion is set to "1". <ADBFHP> returns to "0". EOCFHP flag is cleared to "0" upon read.

Top-priority AD conversion activated while top-priority AD conversion is under way is ignored.

Note: Top-priority A/D conversion interrupt cannot generate DMA transfer request. To generate DMA trans-fer request, please use A/D conversion completion interrupt (INTAD).

22.4.3   AD Monitor Function

There are two channels of AD monitor function.

If ADMOD3<ADOBSV0> and ADMOD5<ADOBSV1> are set to "1", the AD monitor function is ena-bled. If the value of the conversion result register specified by ADMOD3<ADREGS0> and AD-MOD5<ADREGS1> becomes larger or smaller ("Larger" or "Smaller" to be designated by ADMOD3<ADO-BIC0> and ADMOD5<ADBIC1>) than the value of a comparison register, the AD monitor function inter-rupt (INTADM0,INTADM1) is generated. This comparison operation is performed each time a result is stor-ed in a corresponding conversion result register.

If the conversion result register assigned to perform the AD monitor function is continuously used without reading the conversion result, the conversion result is overwritten. The conversion result storage flag <ADRxRF> and the overrun flag <OVRx> remain being set.

22.4.4 Selecting the Input Channel

After a reset, ADMOD0<REPEAT,SCAN> is initialized to "00" and ADMOD1<ADCH[3:0]> is initialized to "0000".

The channels to be converted are selected according to the operation mode of the AD converter as shown below.

1. Normal AD conversion mode

   · If the analog input channel is used in a fixed state (ADMOD0<SCAN> = "0")

      One channel is selected from analog input pins AIN0 through AIN7 by setting ADMOD1<ADCH> to an appropriate setting.

   · If the analog input channel is used in a scan state (ADMOD0<SCAN> = "1")

      One scan mode is selected from the scan modes by setting ADMOD1 <ADCH> and ADSCN to an appropriate setting.

2. Top-priority AD conversion mode

   One channel is selected from analog input pins from AIN0 through AIN7 by setting ADMOD2<HPADCH> to an appropriate setting.

22.4.5 AD Conversion Details

22.4.5.1 Starting AD Conversion

Normal AD conversion is activated by setting ADMOD0<ADS> to "1". Top-priority AD conversion is activated by setting ADMOD2<HPADCE> to "1".

Four operation modes are made available to normal AD conversion. In performing normal AD conversion, one of these operation modes must be selected by setting ADMOD0<REPEAT,SCAN> to an appropriate setting. For top-priority AD conversion, only one operation mode can be used: fixed channel single conversion mode.

Normal AD conversion can be activated using the H/W activation source selected by ADMOD4<ADHS>, and top-priority AD conversion can be activated using the HW activation source selected by ADMOD4<HADHS>. If bits of <ADHS> and <HADHS> are "0", normal and top-priority AD conversions are activated in response to the input of a falling edge through the $\overline{\text{ADTRG}}$ pin. If these bits are "1", normal AD conversion is activated in response to TB6RG0 generated by the 16-bit timer 6, and top-priority AD conversion is activated in response to TB5RG0 generated by the 16-bit timer 5.

To permit H/W activation, set ADMOD4<ADHTG> to "1" for normal AD conversion and set ADMOD4<HADHTG> to "1" for top-priority AD conversion.

Software activation is still valid even after H/W activation has been permitted.

Note 1: When an external trigger is used for the HW activation source of a top-priority AD conversion, an external trigger cannot be set for activating normal AD conversion H/W.

Note 2: TMPM363F10FG disables the external trigger used for H/W activation. Therefore "0" cannot be set to <HADHS> and <ADHS>.

22.4.5.2   AD Conversion

When normal AD conversion starts, the AD conversion Busy flag (ADMOD0<ADBFN>) showing that AD conversion is under way is set to "1".

When top-priority AD conversion starts, the top-priority AD conversion Busy flag (AD-MOD2<ADBFHP>) showing that AD conversion is underway is set to "1". At that time, the value of the Busy flag ADMOD0<ADBFN> for normal AD conversion before the start of top-priority AD conversions are retained. The value of the conversion completion flag ADMOD0<EOCFN> for normal AD conversion before the start of top-priority AD conversion is retained. .

Note: Normal AD conversion must not be activated when top-priority AD conversion is under way.

22.4.5.3   Top-priority AD conversion during normal AD conversion

If top-priority AD conversion has been activated during normal AD conversion, ongoing normal AD conversion is suspended, and restarts normal AD conversion after top-priority AD conversion is completed.

If ADMOD2<HPADCE> is set to "1" during normal AD conversion, ongoing normal AD conversion is suspended, and the top-priority AD conversion starts; specifically, AD conversion (fixed-channel single conversion) is executed for a channel designated by ADMOD2<HPADCH>. After the result of this top-priority AD conversion is stored in the storage register ADREGSP, normal AD conversion is resumed.

If H/W activation of top-priority AD conversion is authorized during normal AD conversion, ongoing AD conversion is discontinued when requirements for activation using a H/W activation resource are met, and top-priority AD conversion (fixed-channel single conversion) starts for a channel designated by ADMOD2<HPADCH>. After the result of this top-priority AD conversion is stored in the storage register ADREGSP, normal AD conversion is resumed.

For example, if channel repeat conversion is activated for channels AIN0 through AIN3 and if <HPADCE> is set to "1" during AIN2 conversion, AIN2 conversion is suspended, and conversion is performed for a channel designated by <HPADCH> (AIN7 in the case shown below). After the result of conversion is stored in ADREGSP, channel repeat conversion is resumed, starting from AIN2.

Top-priority AD has been activated							
Conversion channel	Ch0	Ch1	Ch2	Ch7	Ch2	Ch3	Ch0
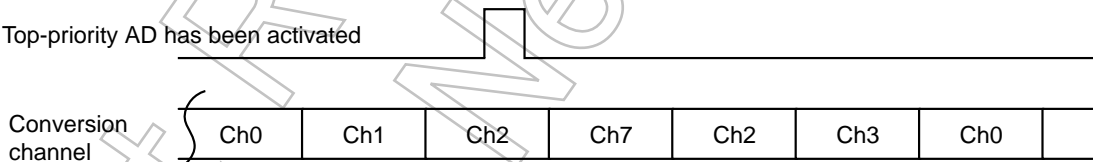
22.4.5.4   Stopping Repeat Conversion Mode

To stop the AD conversion operation in the repeat conversion mode (fixed-channel repeat conversion mode or channel scan repeat conversion mode), write "0" to ADMOD0<REPEAT>. When ongoing AD conversion is completed, the repeat conversion mode terminates, and ADMOD0<ADBFN> is set to "0".

22.4.5.5 Reactivating normal AD conversion

To reactivate normal AD conversion while the conversion is underway, a software reset (AD-MOD3<ADRST>) must be performed before starting AD conversion. The H/W activation method must not be used to reactivate normal AD conversion.

22.4.5.6 Conversion completion

(1) Normal AD conversion completion

When normal AD conversion is completed, the AD conversion completion interrupt (INTAD) is generated. The result of AD conversion is stored in the storage register is the storage register, and two registers change: the register ADMOD0<EOCFN> which indicates the completion of AD conversion and the register ADMOD0<ADBFN>.Interrupt request, conversion register storage register and <EOCFN><ADBFN> change with a different timing according to a mode selected.

In mode other than fixed-channel repeat conversion mode, conversion results are stored in AD conversion result registers (ADREG08 through ADRG7F) corresponding to a channel.

In fixed-channel repeat conversion mode, the conversion results are sequentially stored in storage registers ADREG08 through ADREG7F. However, if interrupt setting on <ITM> is set to be generated each time one AD conversion is completed, the conversion result is stored only in ADREG08. If interrupt setting on <ITM> is set to be generated each time four AD conversions are completed, the conversion results are sequentially stored in ADREG08H through ADREG3B.If interrupt setting on <ITM> is set to be generated each time eight AD conversions are completed, the conversion results are sequentially stored in ADREG08H through ADREG7F.

Interrupt requests, flag changes and conversion result registers in each mode are as shown below.

- Fixed-channel single conversion mode

  After AD conversion completed, ADMOD0<EOCFN> is set to "1", AD-MOD0<ADBFN> is cleared to "0", and the interrupt request is generated.

  Conversion results are stored a conversion result register correspond to a channel.

- Channel scan single conversion mode

  After the channel scan conversion is completed, ADMOD0<EOCFN> is set to "1", AD-MOD0<ADBFN> is set to "0", and the interrupt request INTAD is generated.

  Conversion results are stored a conversion result register correspond to a channel.

- Fixed-channel repeat conversion mode

  ADMOD0<ADBFN> is not cleared to "0". It remains at "1". The timing with which the interrupt request INTAD is generated can be selected by setting ADMOD0<ITM> to an appropriate setting. ADMOD0<EOCFN> is set with the same timing as this interrupt INTAD is generated.

  a. One conversion

  With <ITM[1:0]> set to "00", an interrupt request is generated each time one AD conversion is completed. In this case, the conversion results are always stored in the storage register ADREG08. After the conversion result is stored, <EOCFN> changes to "1".

  b. Four conversions

With <ITM[1:0]> set to "01", an interrupt request is generated each time four AD conversions are completed. In this case, the conversion results are sequentially stored in the storage register ADREG08 through ADREG3B. After the conversion result is stored in ADREG3B, <EOCFN> is set to "1", and the storage of subsequent conversion results starts from ADREG08.

c. 8 conversions

With <ITM[1:0]> set to "10", an interrupt request is generated each time eight AD conversions are completed. In this case, the conversion results are sequentially stored in the storage register ADREG08 through ADREG7F. After the conversion result is stored in ADREG7F, <EOCFN> is set to "1", and the storage of subsequent conversion results starts from ADREG08.

· Channel scan repeat conversion mode

Each time one AD conversion is completed, ADMOD0<EOCF> is set to "1" and interrupt request INTAD is generated. ADMOD0<ADBFN> is not cleared to "0". It remains at "1".

AD conversion results are stored in a AD conversion result register corresponding to a channel.

(2) Top-priority AD conversion completion

After the AD conversion is completed, the top-priority AD conversion completion interrupt (INTADHP) is generated, and ADMOD2<EOCFHP> which indicates the completion of top-priority AD conversion is set to "1".

AD conversion results are stored in the AD conversion result register SP.

(3) Data polling

To confirm the completion of AD conversion without using interrupts, data polling can be used. When AD conversion is completed, ADMOD0<EOCFN> is set to "1". To confirm the completion of AD conversion and to obtain the results, poll this bit.

AD conversion result storage register must be read by half word or word access. If <OVRx> = "0" and <ADRxRF> = "1", a correct conversion result has been obtained.

22.4.5.7 Interrupt generation timings and AD conversion result storage register

Table 22-1shows a relation in the following three items: AD conversion modes, interrupt generation timings and flag operations.Table 22-2 shows a relation between analog channel inputs and AD conversion result registers.

Table 22-1 Relations in conversion modes, interrupt generation timings and flag operations

Conversion mode		Scan / repeat mode setting (ADMOD0)			Interrupt generation timing	<EOCFN>/ <EOCFHP> set timing (note)	ADMOD0 <ADBFN> (After the interrupt is generated)	ADMOD2 <ADBFHP>
		<REPEAT>	<SCAN>	<ITM[1:0]>				
Normal conversion	Fixed-channel single conversion	0	0	–	After generation is completed.	After conversion is completed.	0	–
	Fixed-channel repeat conversion	1	0	00	Each time one conversion is completed.	After one conversion is completed.	1	–
				01	Each time four conversion is completed.	After four conversions are completed.	1	–
				10	Each time eight conversion is completed.	After eight conversions are completed.	1	–
	Channel scan single conversion	0	1	–	After scan conversion is completed.	After scan conversion is completed.	0	–
	Channel scan repeat conversion	1	1	–	After one scan conversion is completed.	After one scan conversion is completed.	1	–
Top-priority conversion		–	–	–	After completion is completed.	Conversion completion	–	0

Note:ADMOD0<EOCFN> and ADMOD2<EOCFHP> are cleared upon read.

Table 22-2 Relation between analog channels input and AD conversion result registers

Analog input channels	Normal AD conversion				Top-priority AD conversion
	Other conversion mode than those shown on the right side	Fixed channel repeat conversion mode (every one conversion)	Fixed channel repeat conversion mode (every four conversions)	Fixed channel repeat conversion mode (every eight conversions)	
AIN0	ADREG08	ADREG08 fixed	ADREG08 ←┐ ↓ ↓ ADREG3B ─┘	ADREG08 ←┐ ↓ ↓ ↓ ADREG7F ─┘	ADREGSP
AIN1	ADREG19				
AIN2	ADREG2A				
AIN3	ADREG3B				
AIN4	ADREG4C				
AIN5	ADREG5D				
AIN6	ADREG6E				
AIN7	ADREG7F				

Note:To access the conversion result register, use a half-word or a word access.

Cautions
The result value of AD conversion may vary depending on the fluctuation of the supply voltage, or may be affected by noise.
When using analog input pins and ports alternately, do not read and write ports during conversion because the conversion accuracy may be reduced. Also the conversion accuracy may be reduced if the output ports current fluctuate during AD conversion.
Please take counteractive measures with the program such as averaging the AD conversion results.

23. Real Time Clock (RTC)

23.1 Function

1. Clock (hour, minute and second)

2. Calendar (month, week, date and leap year)

3. Selectable 12 (am/ pm) and 24 hour display

4. Time adjustment + or − 30 seconds (by software)

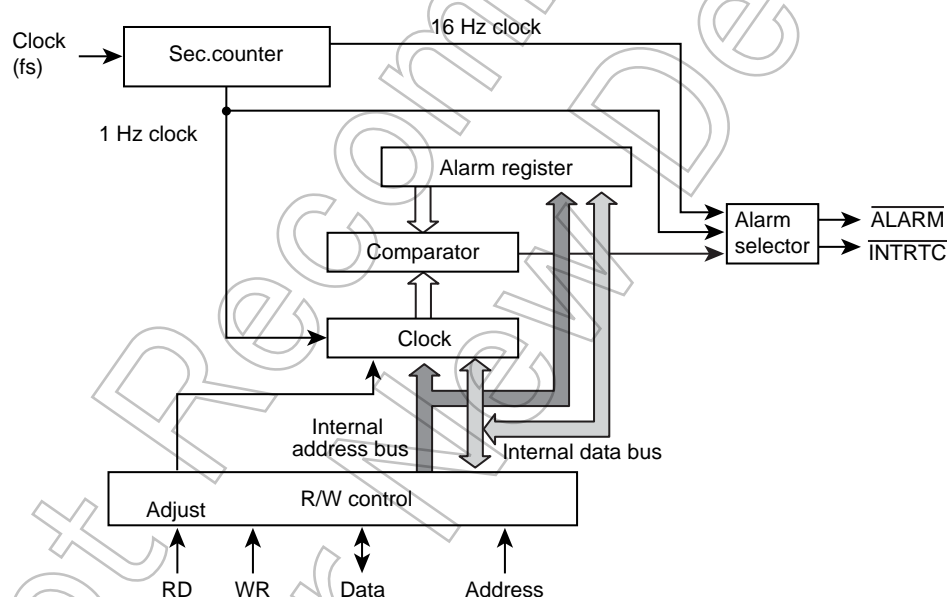5. Alarm (alarm output)

6. Alarm interrupt

23.2 Block Diagram

Figure 23-1 Block Diagram

Note 1: Western calendar year column:This product uses only the final two digits of the year. The year following 99 is 00 years. Please take into account the first two digits when handling years in the western calendar.

Note 2: Leap year:A leap year is divisible by 4 excluding a year divisible by 100; the year divisible by 100 is not considered to be a leap year. Any year divisible by 400 is a leap year. This product is considered the year divisible by 4 to be a leap year and does not take into account the above exceptions. It needs adjustments for the exceptions.

23.3 Detailed Description Register

23.3.1 Register List

The registers and the addresses related to RTC are shown as below.

RTC has two functions, PAGE0 (clock) and PAGE1 (alarm), which share some parts of registers.

The PAGE can be selected by setting RTCPAGER<PAGE >.

Base Address = 0x4004_0100

Register name		Address(Base+)
Second column register (only PAGE0)	RTCSECR	0x0000
Minute column register	RTCMINR	0x0001
Hour column register	RTCHOURR	0x0002
- (note 1)	-	0x0003
Day of the week column register	RTCDAYR	0x0004
Day column register	RTCDATER	0x0005
Month column register (PAGE0)	RTCMONTHR	0x0006
Selection register of 24-hour,12-hour (PAGE1)		
Year column register (PAGE0)	RTCYEARR	0x0007
Leap year register (PAGE1)		
PAGE register	RTCPAGER	0x0008
- (note 1)	-	0x0009
- (note 1)	-	0x000A
- (note 1)	-	0x000B
Reset register	RTCRESTR	0x000C
Reserved	-	0x000D
- (note 1)	-	0x000E
- (note 1)	-	0x000F

Note 1: "0" is read by reading the address. Writing is disregarded.
Note 2: Access to the "Reserved" areas is prohibited.

23.3.2 Control Register

Reset operation initializes the following registers.

- RTCPAGER<PAGE>, <ADJUST>, <INTENA>
- RTCRESTR<RSTALM>, <RSTTMR>, <DIS16HZ>, <DIS1HZ>

Other clock-related registers are not initialized by reset operation.

Before using the RTC, set the time, month, day, day of the week, year and leap year in the relevant registers.

Caution is required in setting clock data, adjusting seconds or resetting the clock.

Refer to "23.4.3 Entering the Low Power Consumption Mode" for more information.

Table 23-1 PAGE0 (clock function) register

Symbol	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function
RTCSECR	–	40sec.	20sec.	10sec.	8sec.	4sec.	2sec.	1sec.	Second column
RTCMINR	–	40min.	20min.	10min.	8min.	4min.	2min.	1min.	Minute column
RTCHOURR	–	–	20hours PM/AM	10hour	8hour	4hour	2hour	1hours	Hour column
RTCDAYR	–	–	–	–	–	Day of the week			Day of the week column
RTCDATER	–	–	Day20	Day10	Day8	Day4	Day2	Day1	Day column
RTCMONTHR	–	–	–	Oct.	Aug.	Apr.	Feb.	Jan.	Month column
RTCYEARR	year 80	year 40	year20	year 10	year 8	year 4	year 2	year 1	Year column (lower two columns)
RTCPAGER	Interrupt enable	–	–	Adjustment function	Clock enable	Alarm enable	–	PAGE setting	PAGE register
RTCRESTR	1 Hz enable	16 Hz enable	Clock reset	Alarm reset	–	Always write "1"			Reset register

Note: Reading RTCSECR, RTCMINR, RTCHOURR, RTCDAYR, RTCMONTHR, RTCYEARR of PAGE0 captures the current state.

Table 23-2 PAGE1 (alarm function) registers

Symbol	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function
RTCSECR	–	–	–	–	–	–	–	–	–
RTCMINR	–	40min.	20min.	10min.	8min.	4min.	2min.	1min.	Minute column
RTCHOURR	–	–	20hours PM/AM	10hour	8hour	4hour	2hour	1hour	Hour column
RTCDAYR	–	–	–	–	–	Day of the week			Day of the week column
RTCDATER	–	–	Day20	Day10	Day8	Day4	Day2	Day1	Day column
RTCMONTHR	–	–	–	–	–	–	–	24/12	24-hour clock mode
RTCYEARR	–	–	–	–	–	–	Leap-year setting		Leap-year mode
RTCPAGER	Interrupt enable	–	–	Adjustment function	Clock enable	Alarm enable	–	PAGE setting	PAGE register
RTCRESTR	1 Hz Enable	16 Hz Enable	Clock reset	Alarm reset	–	Always write "1"			Reset register

Note 1: Reading RTCMINR, RTCHOURR, RTCDAYR, RTCMONTHR, RTCYEARR of PAGE1 captures the current state.

Note 2: RTCSECR, RTCMINR, RTCHOURR, RTCDAYR, RTCDATER, RTCMONTHR, RTCYEARR of PAGE0 and RTCYEARR of PAGE1 (for leap year) must be read twice and compare the data captured.

23.3.3 Detailed Description of Control Register

23.3.3.1 RTCSECR (Second column register (for PAGE0 only))

	7	6	5	4	3	2	1	0
bit symbol	-	SE						
After reset	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Functon
7	–	R	Read as 0.
6-0	SE	R/W	Setting digit register of second 000_0000 : 00sec.　　001_0000 : 10sec.　　010_0000 : 20sec. 000_0001 : 01sec.　　001_0001 : 11sec.　　・ 000_0010 : 02sec.　　001_0010 : 12sec.　　011_0000 : 30sec. 000_0011 : 03sec.　　001_0011 : 13sec.　　・ 000_0100 : 04sec.　　001_0100 : 14sec.　　100_0000 : 40sec. 000_0101 : 05sec.　　001_0101 : 15sec.　　・ 000_0110 : 06sec.　　001_0110 : 16sec.　　101_0000 : 50sec. 000_0111 : 07sec.　　001_0111 : 17sec.　　・ 000_1000 : 08sec.　　001_1000 : 18sec.　　・ 000_1001 : 09sec.　　001_1001 : 19sec.　　101_1001 : 59sec.

Note: The setting other than listed above is prohibited.

23.3.3.2 RTCMINR (Minute column register (PAGE0/1))

	7	6	5	4	3	2	1	0
Bit symbol	-	MI						
After reset	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Functon
7	–	R	Read as 0.
6-0	MI	R/W	Setting digit register of Minutes. 000_0000 : 00min.　　001_0000 : 10min.　　010_0000 : 20min. 000_0001 : 01min.　　001_0001 : 11min. 000_0010 : 02min.　　001_0010 : 12min.　　011_0000 : 30min. 000_0011 : 03min.　　001_0011 : 13min.　　・ 000_0100 : 04min.　　001_0100 : 14min.　　100_0000 : 40min. 000_0101 : 05min.　　001_0101 : 15min.　　・ 000_0110 : 06min.　　001_0110 : 16min.　　101_0000 : 50min. 000_0111 : 07min.　　001_0111 : 17min.　　・ 000_1000 : 08min.　　001_1000 : 18min.　　・ 000_1001 : 09min.　　001_1001 : 19min.　　101_1001 : 59min.

Note: The setting other than listed above is prohibited.

23.3.3.3 RTCHOURR (Hour column register(PAGE0/1))

(1) 24-hour clock mode (RTCMONTHR<MO0>= "1")

	7	6	5	4	3	2	1	0
Bit symbol	-	-	HO					
After reset	0	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Functon
7-6	–	R	Read as 0.
5-0	HO	R/W	Setting digit register of Hour. 00_0000 : 0 o'clock 01_0000 : 10 o'clock 10_0000 : 20 o'clock 00_0001 : 1 o'clock 01_0001 : 11 o'clock 10_0001 : 21 o'clock 00_0010 : 2 o'clock 01_0010 : 12 o'clock 10_0010 : 22 o'clock 00_0011 : 3 o'clock 01_0011 : 13 o'clock 10_0011 : 23 o'clock 00_0100 : 4 o'clock 01_0100 : 14 o'clock 00_0101 : 5 o'clock 01_0101 : 15 o'clock 00_0110 : 6 o'clock 01_0110 : 16 o'clock 00_0111 : 7 o'clock 01_0111 : 17 o'clock 00_1000 : 8 o'clock 01_1000 : 18 o'clock 00_1001 : 9 o'clock 01_1001 : 19 o'clock

Note: The setting other than listed above is prohibited.

(2) 12-hour clock mode (RTCMONTHR<MO0> = "0")

	7	6	5	4	3	2	1	0
Bit symbol	-	-	HO					
After reset	0	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Functon
7-6	–	R	Read as 0.
5-0	HO	R/W	Setting digit register of Hour. (AM) (PM) 00_0000 : 0 o'clock 10_0000 : 0 o'clock 00_0001 : 1 o'clock 10_0001 : 1 o'clock 00_0010 : 2 o'clock 10_0010 : 2 o'clock 00_0011 : 3 o'clock 10_0011 : 3 o'clock 00_0100 : 4 o'clock 10_0100 : 4 o'clock 00_0101 : 5 o'clock 10_0101 : 5 o'clock 00_0110 : 6 o'clock 10_0110 : 6 o'clock 00_0111 : 7 o'clock 10_0111 : 7 o'clock 00_1000 : 8 o'clock 10_1000 : 8 o'clock 00_1001 : 9 o'clock 10_1001 : 9 o'clock 01_0000 : 10 o'clock 11_0000 : 10 o'clock 01_0001 : 11 o'clock 11_0001 : 11 o'clock

Note: The setting other than listed above is prohibited.

23.3.3.4 RTCDAYR (Day of the week column register(PAGE0/1))

	7	6	5	4	3	2	1	0
Bit symbol	-	-	-	-	-		WE	
After reset	0	0	0	0	0	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
7-3	−	R	Read as 0.
2-0	WE	R/W	Setting digit register of day of the week. 000: Sunday 001: Monday 010: Tuesday 011: Wednesday 100: Thursday 101: Friday 110: Saturday

Note: The setting other than listed above is prohibited.

23.3.3.5 RTCDATER (Day column register (for PAGE0/1 only))

	7	6	5	4	3	2	1	0
Bit symbol	-	-				DA		
After reset	0	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Functon
7-6	−	R	Read as 0.

5-0	DA	R/W	Setting digit register of day.
			
			                         01_0000 : 10th day 10_0000 : 20th day 11_0000 : 30th day
			          00_0001 : 1st day 01_0001 : 11th day 10_0001 : 21th day 11_0001 : 31th day
			          00_0010 : 2nd day 01_0010 : 12th day 10_0010 : 22th day
			          00_0011 : 3rd day 01_0011 : 13th day 10_0011 : 23th day
			          00_0100 : 4th day 01_0100 : 14th day 10_0100 : 24th day
			          00_0101 : 5th day 01_0101 : 15th day 10_0101 : 25th day
			          00_0110 : 6th day 01_0110 : 16th day 10_0110 : 26th day
			          00_0111 : 7th day 01_0111 : 17th day 10_0111 : 27th day
			          00_1000 : 8th day 01_1000 : 18th day 10_1000 : 28th day
			          00_1001 : 9th day 01_1001 : 19th day 10_1001 : 29th day

Note 1: The setting other than listed above is prohibited.

Note 2: Do not set for non-existent days (e.g. 30th Feb.).

23.3.3.6   RTCMONTHR (Month column register (for PAGE0 only))

	7	6	5	4	3	2	1	0
Bit symbol	-	-	-	MO				
After reset	0	0	0	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Functon
7-5	−	R	Read as 0.
4-0	MO	R/W	Setting digit register of Month. 0_0001 : January 0_0111 : July 0_0010 : February 0_1000 : August 0_0011 : March 0_1001 : September 0_0100 : April 1_0000 : October 0_0101 : May 1_0001 : November 0_0110 : June 1_0010 : December

Note: The setting other than listed above is prohibited.

23.3.3.7   RTCMONTHR (Selection of 24-hour clock or 12-hour clock (for PAGE1 only))

	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	MO0
After reset	0	0	0	0	0	0	0	Undefined

Bit	Bit Symbol	Type	Function
7-1	−	R	Read as 0.
0	MO0	R/W	0: 12-hour 1: 24-hour

Note: Do not change the RTCMONTHR<MO0> while the RTC is in operation.

23.3.3.8 RTCYEARR (Year column register (for PAGE0 only))

	7	6	5	4	3	2	1	0
bit symbol				YE				
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
7-0	YE	R/W	Setting digit register of Year.

			0000_0000 : 00 year	0001_0000 : 10 years	0110_0000 : 60 years
			0000_0001 : 01 years	·	·
			0000_0010 : 02 years	0010_0000 : 20 years	0111_0000 : 70 years
			0000_0011 : 03 years	·	·
			0000_0100 : 04 years	0011_0000 : 30 years	1000_0000 : 80 years
			0000_0101 : 05 years	·	·
			0000_0110 : 06 years	0100_0000 : 40 years	1001_0000 : 90 years
			0000_0111 : 07 years	·	·
			0000_1000 : 08 years	01001_0000 : 50 years	·
			0000_1001 : 09 years	·	1001_1001 : 99 years

Note: The setting other than listed above is prohibited.

23.3.3.9 RTCYEARR (Leap year register (for PAGE1 only))

	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	LEAP	
After reset	0	0	0	0	0	0	Undefined	Undefined

Bit	Bit Symbol	Type	Functon
7-2	−	R	Read as 0.
1-0	LEAP	R/W	00 : A leap year
			01 : one year after a leap year
			10 : two years after a leap year
			11 : three years after a leap year

23.3.3.10 RTCPAGER(PAGE register(PAGE0/1))

	7	6	5	4	3	2	1	0
Bit symbol	INTENA	-	-	ADJUST	ENATMR	ENAALM	-	PAGE
After reset	0	0	0	0	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
7	INTENA	R/W	INTRTC 0:Disable 1:Enable
6-5	–	R	Read as 0.
4	ADJUST	R/W	[Write] 0: Don't care 1: Sets ADJUST request Adjusts seconds. The request is sampled when the sec. counter counts up. If the time elapsed is between 0 and 29 seconds, the sec. counter is cleared to "0". If the time elapsed is between 30 and 59 seconds, the min. counter is carried and sec. counter is cleared to "0". [Read] 0: ADJUST no request 1: ADJUST requested If "1" is read, it indicates that ADJUST is being executed. If "0" is read, it indicates that the execution is finished.
3	ENATMR	R/W	Clock 0: Disable 1: Enable
2	ENAALM	R/W	ALARM 0: Disable 1: Enable
1	–	R	Read as 0.
0	PAGE	R/W	PAGE selection 0:Selects Page0 1:Selects Page1

Note 1: A read-modify-write operation cannot be porfomed.

Note 2: To set interrupt enable bits to <ENATMR>, <ENAALM> and <INTENA>, you must follow the order specified here. Make sure not to set them at the same time (make sure that there is time lag between interrupt enable and clock/alarm enable).To change the setting of <ENATMR> and <ENAALM>, <INTENA> must be disabled first.

Example: Clock setting/Alarm setting

```
              7  6  5  4  3  2  1  0
RTCPAGER  ←   0  0  0  0  1  1  0  0    Enables Clock and alarm
RTCPAGER  ←   1  0  0  0  1  1  0  0    Enables interrupt
```

23.3.3.11 RTCRESTR (Reset register (for PAGE0/1))

	7	6	5	4	3	2	1	0
Bit symbol	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	-	-	-	-
After reset	1	1	0	0	0	1	1	1

Bit	Bit Symbol	Type	Function
7	DIS1HZ	R/W	1 Hz 0:Enable 1: Disable
6	DIS16HZ	R/W	16 Hz 0: Enable 1: Disable
5	RSTTMR	R/W	[Write] 0: Don't care 1: Sec.counter reset Resets the sec counter. The equest is sampled using low-speed clock. [Read] 0: No reset request 1: RESET requested If "1" is read, it indicates that RESET is being executed. If "0" is read, it indicates that the execution is finished.
4	RSTALM	R/W	0:Don't care 1: Alarm reset Initializes alarm registers (Minute column, hour column, day column and day of the week column) as follows. MInute:00, Hour:00, Day:01, Day of the week:Sunday
3	–	R	Read as 0.
2-0	–	R/W	Write "1".

Note 1: A read-modify-write operation cannot be performed.

The setting of <DIS1HZ> and <DIS16MHZ>, RTCPAGER<ENAALM> used for alarm, 1Hz interrupt and 16Hz interrupt is shown as below.

<DIS1HZ>	<DIS16HZ>	RTCPAGER <ENAALM>	Interrupt source signal
1	1	1	Alarm
0	1	0	1 Hz
1	0	0	16 Hz
Others			Interrupt not generated.

23.4   Operational Description

The RTC incorporates a second counter that generates a 1Hz signal from a 32.768 kHz signal.

The second counter operation must be taken into account when using the RTC.

23.4.1   Reading clock data

1.   Using 1Hz interrupt

The 1Hz interrupt is generated being synchronized with counting up of the second counter.

Data can be read correctly if reading data after 1Hz interrupt occurred.

2.   Using pair reading

There is a possibility that the clock data may be read incorrectly if the internal counter operates carry during reading. To ensure correct data reading, read the clock data twice as shown below. A pair of data read successively needs to match.
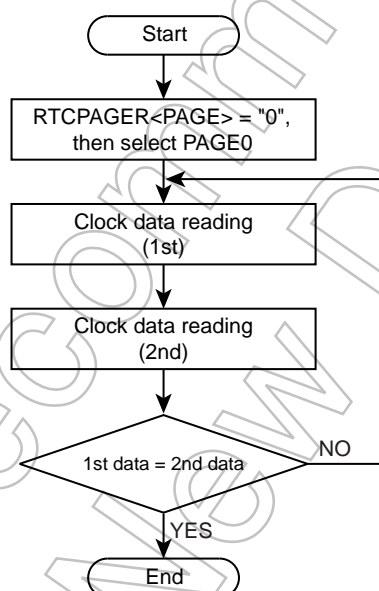
Figure 23-2 Flowchart of the clock data reading

23.4.2   Writing clock data

A carry during writing ruins correct data writing. The following procedure ensures the correct data writing.

1.   Using 1 Hz interrupt

The 1Hz interrupt is generated by being synchronized with counting up of the second counter. If data is written in the time between 1Hz interrupt and subsequent one second count, it completes correctly.

2.   Resetting counter

Write data after resetting the second counter.

The 1Hz-interrupt is generated one second after enabling the interrupt subsequent to counter reset.

The time must be set within one second after the interrupt.

```
                        ┌───────────┐
                        │   Strat   │
                        └─────┬─────┘
                              ▼
              ┌───────────────────────────────┐
              │ RTCPAGER<PAGE> = "0" then      │
              │        select PAGE0            │
              └───────────────┬───────────────┘
                              ▼
              ┌───────────────────────────────┐
              │ RTCRESTR<RSTTMR> = "1" then    │
              │        reset counter           │
              └───────────────┬───────────────┘
                              ▼
              ┌───────────────────────────────┐
              │ RTCRESTR<DIS1HZ> = "0" then    │
              │      enable 1Hz interrupt      │
              └───────────────┬───────────────┘
                              ▼
                        ◇─────────────◇        NO
                       ╱ First interrupt ╲ ────────┐
                       ╲   (After 1s)    ╱         │
                        ◇─────────────◇            │
                              │ YES                │
                              ▼                    │
              ┌───────────────────────────────┐    │
              │        Timer setting           │    │
              └───────────────┬───────────────┘    │
                              ▼                    │
                        ┌───────────┐              │
                        │   End     │              │
                        └───────────┘              │
```
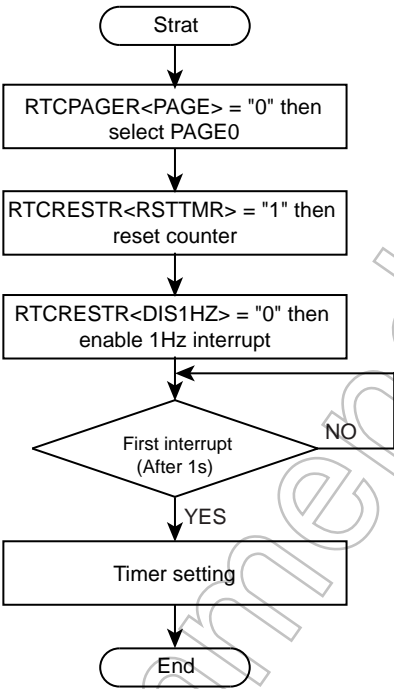
Figure 23-3 Flowchart of the clock data writing

3. Disabling the clock

Writing "0" to RTCPAGER<ENATMR> disables clock operation including a carry.

Stop the clock after the 1Hz-interrupt. The second counter keeps counting.

Set the clock again and enable the clock within one second before next 1Hz-interrupt

```
                        ┌───────────┐
                        │   Start   │
                        └─────┬─────┘
                              ▼
              ┌───────────────────────────────┐
              │        Disabling clock         │
              └───────────────┬───────────────┘
                              ▼
              ┌───────────────────────────────┐
              │     Writing the clock data     │
              └───────────────┬───────────────┘
                              ▼
              ┌───────────────────────────────┐
              │       Enabling the clock       │
              └───────────────┬───────────────┘
                              ▼
                        ┌───────────┐
                        │   End     │
                        └───────────┘
```
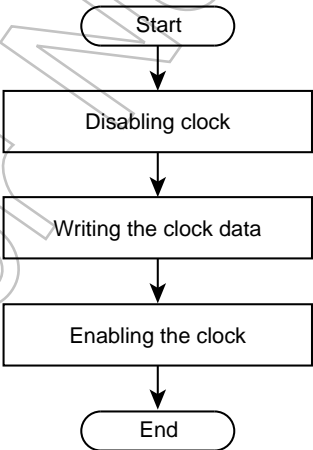
Figure 23-4 Flowchart of the disabling clock

23.4.3   Entering the Low Power Consumption Mode

To enter SLEEP mode, in which the system clock stops, after changing clock data, adjusting seconds or re-setting the clock, be sure to observe one of the following procedures

1. After changing the clock setting registers, setting the RTCPAGER<ADJUST> bit or setting the RTCRESTR<RSTTMR> bit, wait for one second for an interrupt to be generated.

2. After changing the clock setting registers, setting the RTCPAGER<ADJUST> bit or setting the RTCRESTR<RSTTMR> bit, read the corresponding clock register values, <ADJUST> or <RSTTMR> to make sure that the setting you have made is reflected.

23.5 Alarm function

By writing "1" to RTCPAGER<PAGE>, the alarm function of the PAGE1 registers is enabled. One of the following three signals is output to the $\overline{\text{ALARM}}$ pin.

1. "Low" pulse (when the alarm register corresponds with the clock)

2. 1Hz cycle "Low" pulse

3. 16Hz cycle "Low" pulse

In any cases shown above, the INTRTC outputs one cycle pulse of low-speed clock. It outputs the INTRTC interrupt request simultaneously.

The INTRTC interrupt signal is falling edge triggered. Specify the falling edge as the active state in the CG Interrupt Mode Control Register

23.5.1 "Low" pulse (when the alarm register corresponds with the clock)

"Low" pulse is output to the $\overline{\text{ALARM}}$ pin when the values of the PAGE0 clock register and the PAGE1 alarm register correspond. The INTRTC interrupt is generated and the alarm is triggered.

The alarm settings

Initialize the alarm with alarm prohibited. Write "1" to RTCRESTR<RSTALM>.

It makes the alarm setting to be 00 minute, 00 hour, 01 day and Sunday.

Setting alarm for min., hour, date and day is done by writing data to the relevant PAGE1 register.

Enable the alarm with the RTCPAGER <ENAALM> bit. Enable the interrupt with the RTCPAGER <INTENA> bit.

The following is an example program for outputting an alarm from the ALARM pin at noon (12:00) on Monday 5th.

		7	6	5	4	3	2	1	0	
RTCPAGER	←	0	0	0	0	1	0	0	1	Disables alarm,sets PAGE1
RTCRESTR	←	1	1	0	1	0	0	0	0	Initializes alarm
RTCDAYR	←	0	0	0	0	0	0	0	1	Monday
RTCDATER	←	0	0	0	0	0	1	0	1	5th day
RTCHOURR	←	0	0	0	1	0	0	1	0	Sets 12 o'clock
RTCMINR	←	0	0	0	0	0	0	0	0	Sets 00 min
RTCPAGER	←	0	0	0	0	1	1	0	0	Enables alarm
RTCPAGER	←	1	0	0	0	1	1	0	0	Enables interrupts

The above alarm works in synchronization with the low-speed clock. When the CPU is operating at high frequency oscillation, a maximum of one clock delay at fs (about 30μs) may occur for the time register setting to become valid.

Note:To make the alarm work repeatedly (e.g. every Wednesday at 12:00), next alarm must be set during the INTRTC interrupt routine that is generated when the time set for the alarm matches the RTC count.

23.5.2   1Hz cycle "Low" pulse

The RTC outputs a "Low" pulse cycle of low-speed 1Hz clock to the $\overline{\text{ALARM}}$ pin by setting RTCPAGER<INTENA>="1" after setting RTCPAGER<ENAALM>= "0", RTCRESTR<DIS1HZ>= "0" and <DIS16HZ>= "1". It generates an INTRTC interrupt simultaneously.

23.5.3   16Hz cycle "Low" pulse

The RTC outputs a "Low" pulse cycle of low-speed 16Hz clock to the $\overline{\text{ALARM}}$ pin by setting RTCPAGER<INTENA>="1" after setting RTCPAGER<ENAALM>= "0", RTCRESTR<DIS1HZ>= "1" and <DIS16HZ>= "0". It generates an INTRTC interrupt simultaneously.

TOSHIBA

24. Flash

This section describes the hardware configuration and operation of the flash memory.

24.1 Flash Memory

24.1.1 Features

1. Memory capacity

   TMPM363F10FG contains flash memory. The memory sizes and configurations are shown in the table below.

   Independent write access to each block is available. When the CPU is to access the internal flash memory, 32-bit data bus width is used.

2. Write / erase time

   Writing is executed per page. TMPM363F10FG contains 128 words.

   Page writing requires 1.25ms (typical) regardless of number of words.

   A block erase requires 0.1 sec. (typical).

   The following table shows write and erase time per chip.

Product Name	Memory size	Block Configuration			Number of words	Write time	Erase time
		128 KB	64 KB	32 KB			
TMPM363F10FG	1024 KB	7	1	2	128	2.56 sec	1.0 sec

Note: **The above values are theoretical values not including data transfer time. The write time per chip depends on the write method to be used by users.**

3. Programming method

   There are two types of the onboard programming mode for users to program (rewrite) the device while it is mounted on the user's board:

   a. User boot mode

      The use's original rewriting method can be supported.

   b. Single boot mode

      The rewriting method to use serial data transfer (Toshiba's unique method) can be supported.

4. Rewriting method

The flash memory included in this device is generally compliant with the applicable JEDEC standards except for some specific functions. Therefore, if a user is currently using an external flash memory device, it is easy to implement the functions into this device. Furthermore, the user is not required to build his/her own programs to realize complicated write and erase functions because such functions are automatically performed using the circuits already built-in the flash memory chip.

JEDEC compliant functions	Modified, added, or deleted functions
・ Automatic programming ・ Automatic chip erase ・ Automatic block erase ・ Data polling / toggle bit	\<Modified\> Block protect (only software protection is supported) \<Deleted\> Erase resume - suspend function

5. Protect/ Security Function

This device is also implemented with a read-protect function to inhibit reading flash memory data from any external writer device. On the other hand, rewrite protection is available only through command-based software programming; any hardware setting method to apply +12VDC is not supported. See the chapter "ROM protection" for details of ROM protection and security function.

Note: **If a password is set to 0xFF (erased data), it is difficult to protect data securely due to an easy-to-guess password. Even if Single Boot mode is not used, it is recommended to set a unique value as a password.**

24.1.2　Block Diagram of the Flash Memory Section
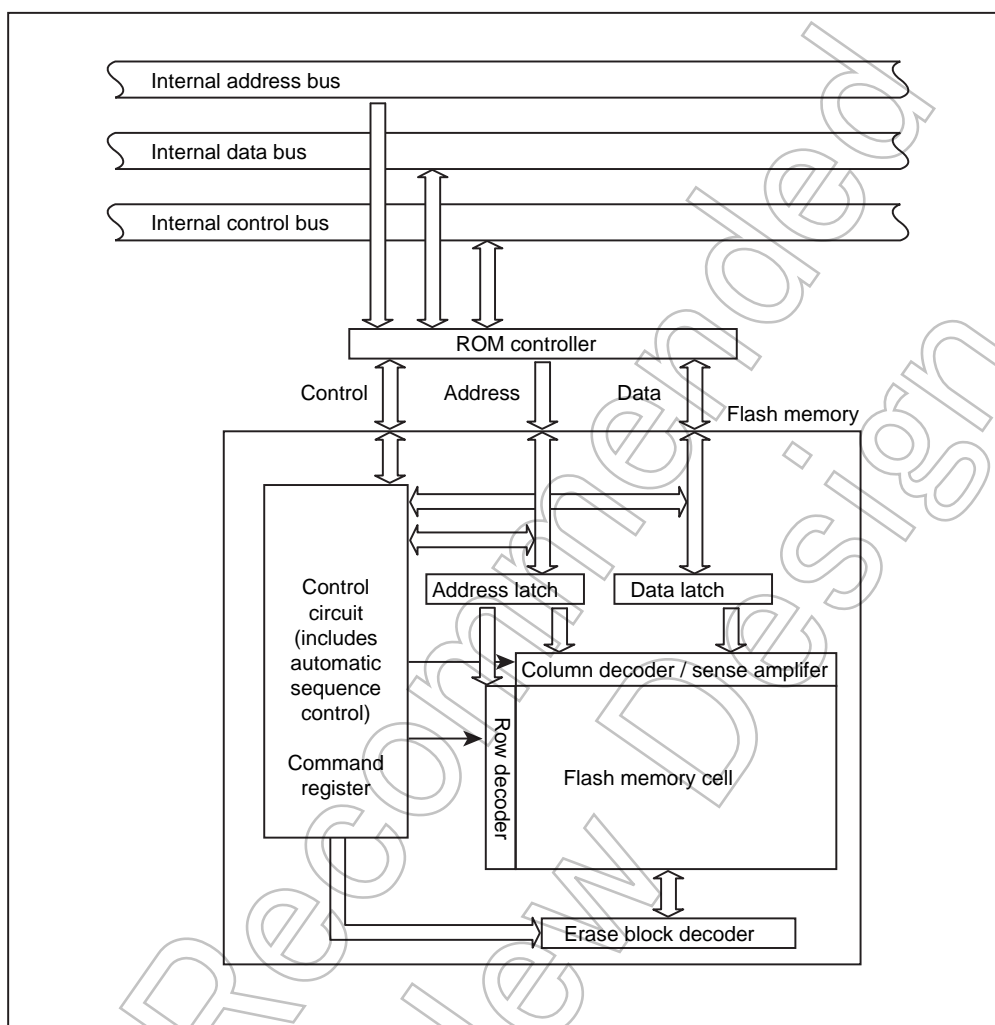
Figure 24-1 Block Diagram of the Flash Memory Section

24.2 Operation Mode

This device has three operation modes including the mode not to use the internal flash memory.

Table 24-1 Operation modes

Operation mode		Operation details
Single chip mode		After reset is cleared, it starts up from the internal flash memory.
	Normal mode	In this operation mode, two different modes, i.e., the mode to execute user application programs and the mode to rewrite the flash memory onboard the user's set, are defined.The former is referred to as "normal mode" and the latter "user boot mode".
	User boot mode	A user can uniquely configure the system to switch between these two modes. For example, a user can freely design the system such that the normal mode is selected when the port "A0" is set to "1" and the user boot mode is selected when it is set to "0". A user should prepare a routine as part of the application program to make the decision on the selection of the modes.
Single boot mode		After reset is cleared, it starts up from the internal Boot ROM (Mask ROM). In the Boot ROM, an algorithm to enable flash memory rewriting on the user's set through the serial port of this device is programmed. By connecting to an external host computer through the serial port, the internal flash memory can be programmed by transferring data in accordance with predefined protocols.

Among the flash memory operation modes listed in the above table, the User Boot mode and the Single Boot mode are the programmable modes. These two modes, the User Boot mode and the Single Boot mode, are referred to as "Onboard Programming" modes where onboard rewriting of internal flash memory can be made on the user's set.

Either the Single Chip or Single Boot operation mode can be selected by externally setting the level of the $\overline{\text{BOOT}}$ (PI0) pin while the device is in reset status.

Table 24-2 Operating Mode Setting

Operation mode	Pin	
	$\overline{\text{RESET}}$	$\overline{\text{BOOT}}$ (PI0)
Single chip mode	$0 \rightarrow 1$	1
Single boot mode	$0 \rightarrow 1$	0
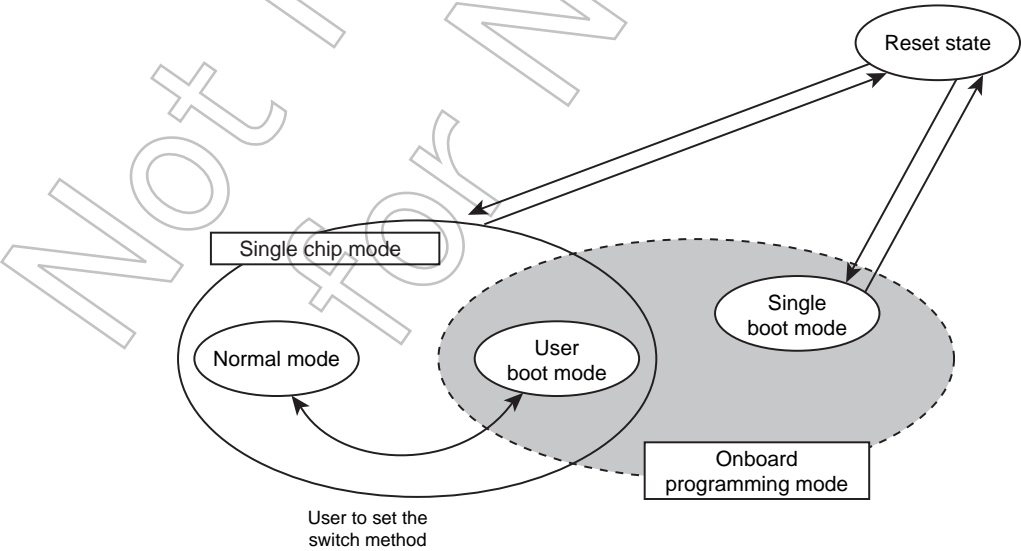
Figure 24-2 Mode Transition Diagram

24.2.1   Reset Operation

To reset the device, ensure that the power supply voltage is within the operating voltage range, that the internal oscillator has been stabilized, and that the $\overline{\text{RESET}}$ input is held at "0" for a minimum duration of 12 system clocks (0.19μs with 64MHz operation; the "1/1" clock gear mode is applied after reset).

Note 1: **It is necessary to apply "0" to the RESET inputs upon power on for a minimum duration of 700 μs regardless of the operating frequency.**

Note 2: **While flash auto programming or erasing is in progress, at least 0.5 μs of reset period is required regardless of the system clock frequency. In this condition, it takes approx. 2 ms to enable reading after reset.**

24.2.2   User Boot Mode (Single chip mode)

User Boot mode is to use flash memory programming routine defined by users. It is used when the data transfer buses for flash memory program code on the old application and for serial I/O are different. It operates at the single chip mode; therefore, a switch from normal mode in which user application is activated at the single chip mode to User Boot Mode for programming flash is required. Specifically, add a mode judgment routine to a reset program in the user application.

The condition to switch the modes needs to be set by using the I/O of TMPM363F10FG in conformity with the user's system setup condition. Also, flash memory programming routine that the user uniquely makes up needs to be set in the new application. This routine is used for programming after being switched to User Boot Mode. The execution of the programming routine must take place while it is stored in the area other than the flash memory since the data in the internal flash memory cannot be read out during delete / writing mode. Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental corruption during subsequent Single-Chip (Normal mode) operations. Be sure not to cause any exceptions including a non-maskable while User Boot Mode.

(1-A) and (1-B) are the examples of programming with routines in the internal flash memory and in the external memory. For a detailed description of the erase and program sequence, refer to "24.3 On-board Programming of Flash Memory (Rewrite/Erase)".

24.2.2.1 (1-A) Method 1: Storing a Programming Routine in the Flash Memory

(1) Step-1

Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMPM363F10FG on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

|	
(a) Mode judgment routine:	Code to determine whether or not to switch to User Boot mode
(b) Programming routine:	Code to download new program code from a host controller and re-program the flash memory
(c) Copy routine:	Code to copy the data described in (b) from the TMPM363F10FG flash memory to either the TMPM363F10FG on-chip RAM or external memory device.
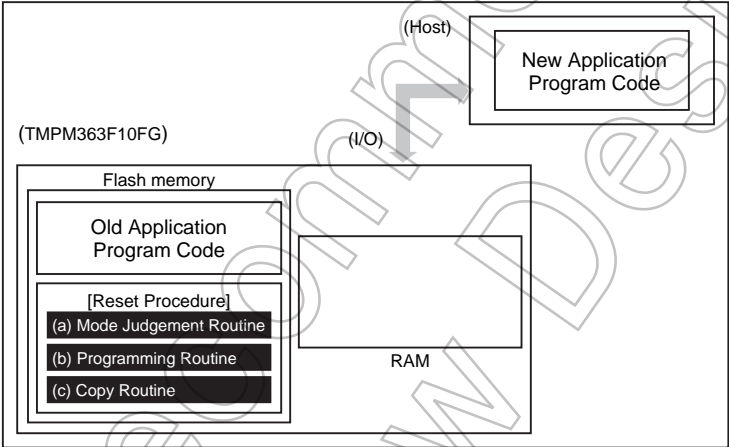
(2)   Step-2

The following description is the case that programming routines are installed in the reset processing program. After $\overline{\text{RESET}}$ pin is released, the reset procedure determines whether to put the TMPM363F10FG flash memory in User Boot mode. If mode switching conditions are met, the flash memory enters User Boot mode. (All interrupts including NMI must be not used while in User Boot mode.)
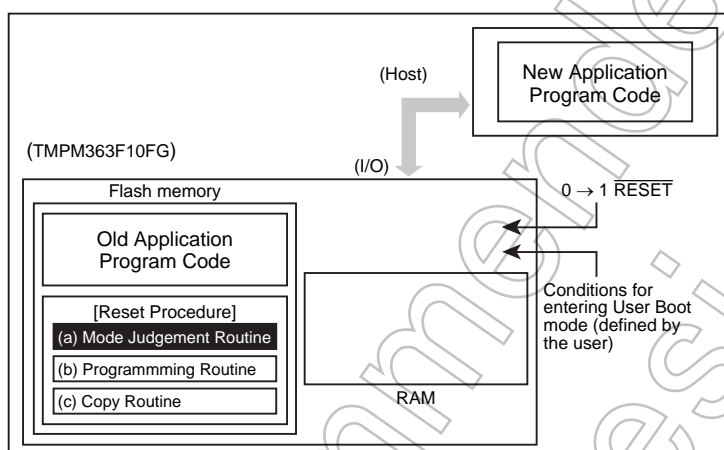

(3)   Step-3

Once transition to User Boot mode is occurred, execute the copy routine (c) to copy the flash programming routine (b) to the TMPM363F10FG on-chip RAM.

(4)   Step-4

Jump program execution to the flash programming routine in the on-chip RAM to clear write or erase protection and erase a flash block containing the old application program code.

(5)   Step-5

Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. When the programming is completed, the writing or erase protection of that flash block in the user's program area must be set.

TOSHIBA
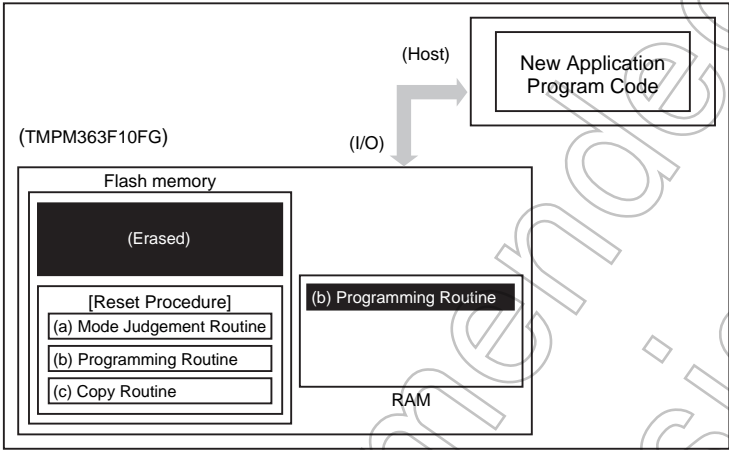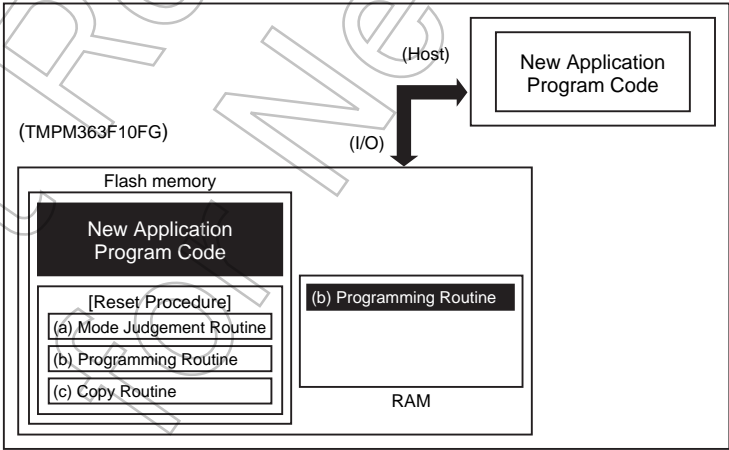
(6)   Step-6

Set $\overline{\text{RESET}}$ to "0" to reset the TMPM363F10FG. Upon reset, the on-chip flash memory is set to Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.

24.2.2.2 (1-B) Method 2: Transferring a Programming Routine from an External Host

(1) Step-1

Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMPM363F10FG on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

(a) Mode judgment routine:	Code to determine whether or not to switch to User Boot mode
(b) Transfer routine:	Code to download new program code from a host controller

Also, prepare a programming routine shown below on the host controller:

(c) Programming routine:	Code to download new program code from an external host controller and re-program the flash memory
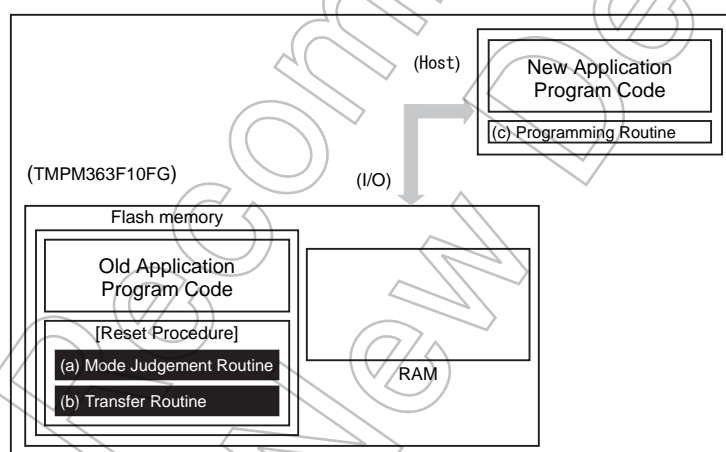
(2)   Step-2

   The following description is the case that programming routines are installed in the reset process-
ing program. After RESET is released, the reset procedure determines whether to put the
TMPM363F10FG flash memory in User Boot mode. If mode switching conditions are met, the
flash memory enters User Boot mode. (All interrupts including NMI must be not used while in
User Boot mode).
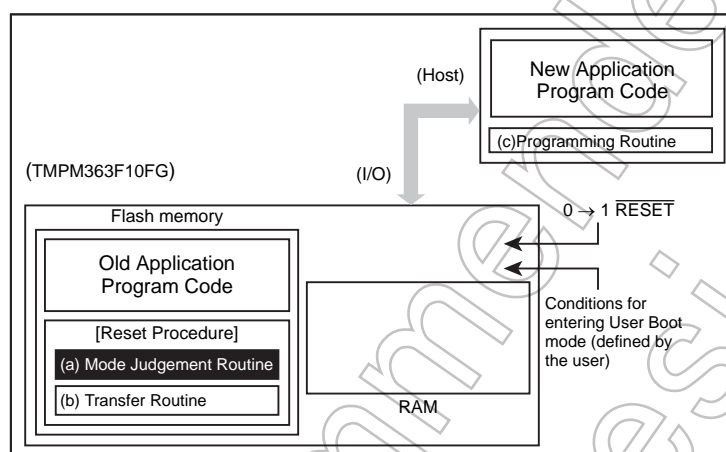
(3)   Step-3

   Once User Boot mode is entered, execute the transfer routine (b) to download the flash program-
ming routine (c) from the host controller to the TMPM363F10FG on-chip RAM.
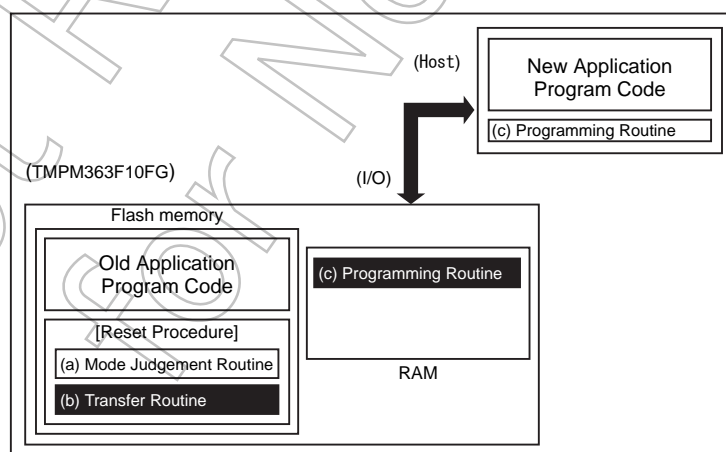
(4) Step-4

Jump program execution to the flash programming routine in the on-chip RAM to clear write or erase protection and erase a flash block containing the old application program code.
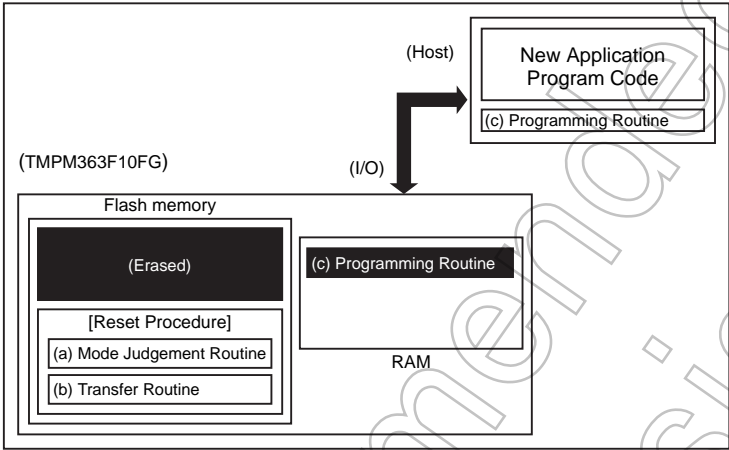
(5) Step-5

Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. When the programming is completed, the writing or erase protection of that flash block in the user program area must be set.
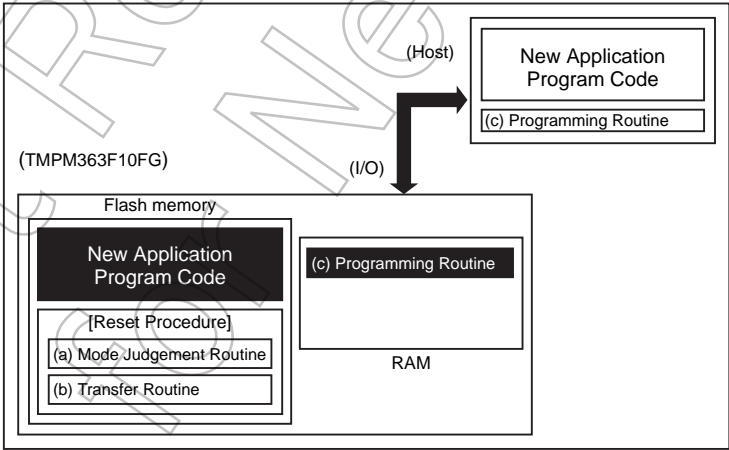
(6)   Step-6

Set $\overline{\text{RESET}}$ to "0" low to reset the TMPM363F10FG. Upon reset, the on-chip flash memory is set to Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.
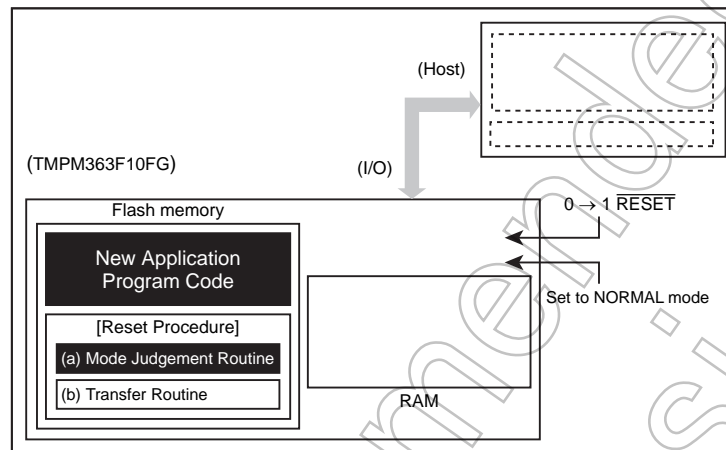
24.2.3 Single Boot Mode

In Single Boot mode, the flash memory can be re-programmed by using a program contained in the TMPM363F10FG on-chip boot ROM. This boot ROM is a masked ROM. When Single Boot mode is selected upon reset, the boot ROM is mapped to the address region including the interrupt vector table while the flash memory is mapped to an address region different from it.

Single Boot mode allows for serial programming of the flash memory. Channel 4 of the SIO (SIO4) of the TMPM363F10FG is connected to an external host controller. Via this serial link, a programming routine is downloaded from the host controller to the TMPM363F10FG on-chip RAM. Then, the flash memory is re-programmed by executing the programming routine. The host sends out both commands and programming data to re-program the flash memory. Communications between the SIO4 and the host must follow the protocol described later. To secure the contents of the flash memory, the validity of the application's password is verified before a programming routine is downloaded into the on-chip RAM. If password matching fails, the transfer of a programming routine itself is aborted. As in the case of User Boot mode, all interrupts including the non-maskable interrupt (NMI) must be disabled in Single Boot mode while the flash memory is being erased or programmed. In Single Boot mode, the boot-ROM programs 33are executed in Normal mode.

Once re-programming is complete, it is recommended to set the write/erase protection to the relevant flash blocks from accidental corruption during subsequent Single-Chip (Normal mode) operations.

### 24.2.3.1 (2-A) Using the Program in the On-Chip Boot ROM

(1) Step-1

The flash block containing the old version of the program code does not need to be erased before executing the programming routine. Since a programming routine and programming data are transferred via the SIO (SIO4), the SIO4 must be connected to a host controller. Prepare a programming routine (a) on the host controller.
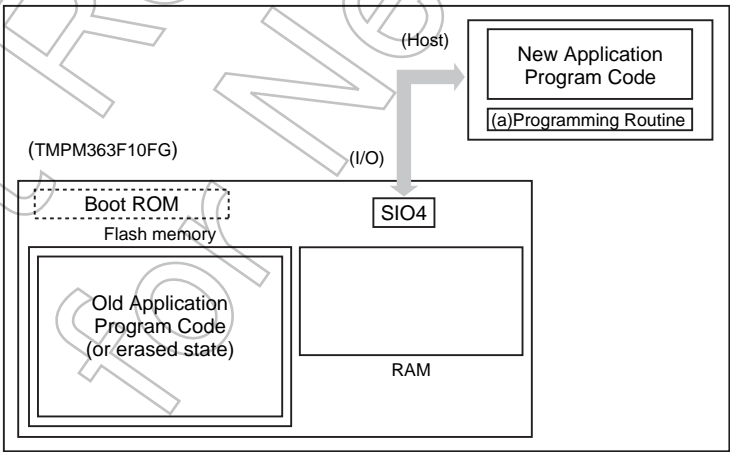
(2)   Step-2

Set the $\overline{\text{RESET}}$ pin to "1" to cancel the reset of the TMPM363F10FG when the $\overline{\text{BOOT}}$ pin has al-
ready been set to "0". After reset, CPU reboots from the on-chip boot ROM. The 12-byte password
transferred from the host controller via SIO4 is firstly compared to the contents of the special flash
memory locations. (If the flash block has already been erased, the password is 0xFF).
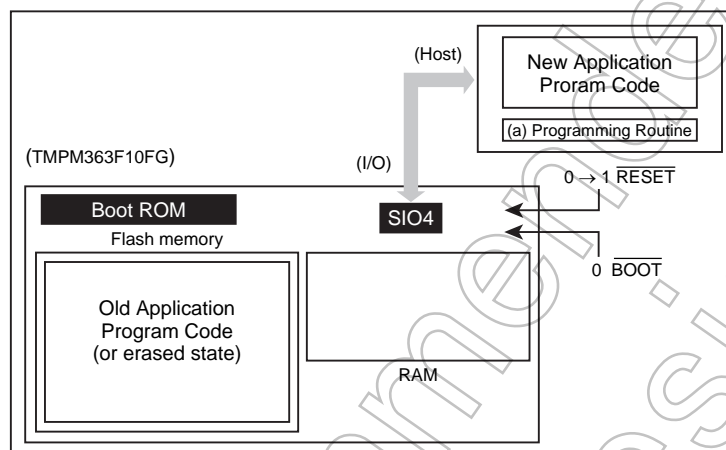
(3)   Step-3

If the password is correct, the boot program downloads the programming routine (a) from the
host controller into the on-chip RAM of the TMPM363F10FG. The programming routine must be stor-
ed in the range from 0x2000_0400 to the end address of RAM.
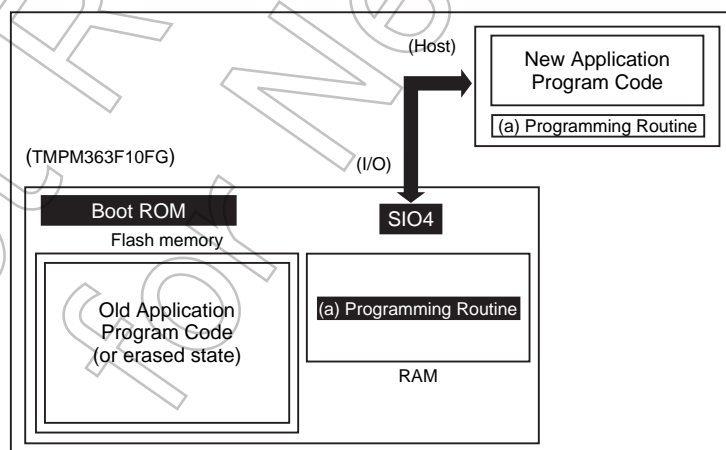
(4)   Step-4

The CPU jumps to the programming routine (a) in the on-chip RAM to erase the flash block containing the old application program code. The Block Erase or Chip Erase command may be used.
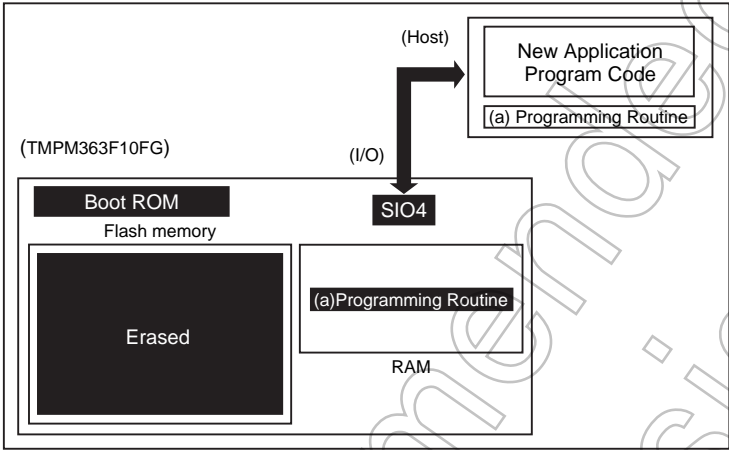

(5)   Step-5

Next, the programming routine (a) downloads new application program code from the host controller and programs it into the erased flash block. When the programming is completed, the writing or erase protection of that flash block in the user's program area must be set.

In the example below, new program code comes from the same host controller via the same SIO4 channel as for the programming routine. However, once the programming routine has begun to execute in the on-chip RAM, it is free to change the transfer path and the source of the transfer. Create board hardware and a programming routine to suit your particular needs.
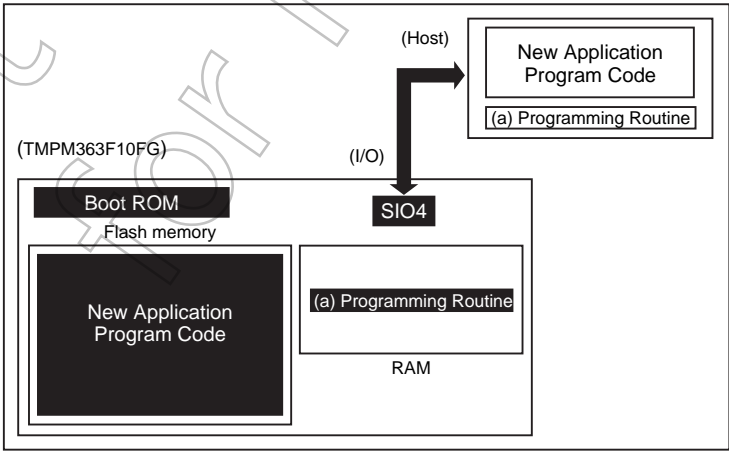
(6)   Step-6

When programming of the flash memory is complete, power off the board and disconnect the cable between the host and the target board. Turn on the power again so that the TMPM363F10FG reboots in Single-Chip (Normal) mode to execute the new program.
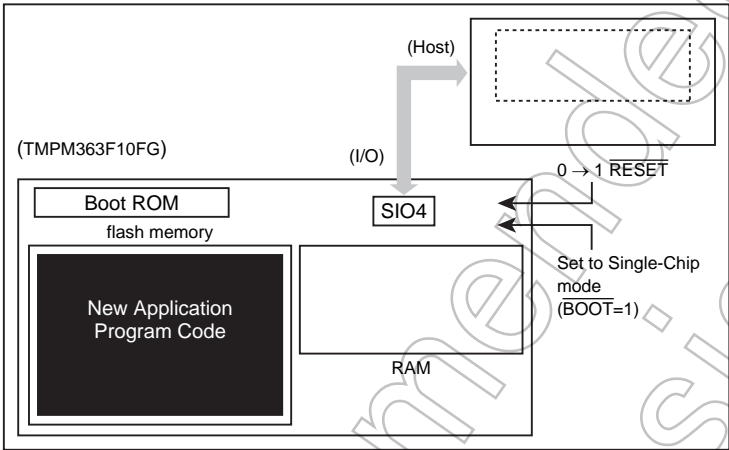
24.2.4   Configuration for Single Boot Mode

To execute the on-board programming, boot the TMPM363F10FG with Single Boot mode following the configuration shown below.

$$\overline{BOOT}(PI0) = 0$$
$$\overline{RESET} = 0 \rightarrow 1$$

Set the $\overline{RESET}$ input to "0", and set the each $\overline{BOOT}$ (PI0) pins to values shown above, and then release $\overline{RE\text{-}SET}$ pin (high).

24.2.5   Memory Map

Figure 24-3 shows a comparison of the memory maps in Normal and Single Boot modes. In Single Boot mode, the internal flash memory is mapped to 0x3F80_0000 and later addresses, and the Internal boot ROM (Mask ROM) is mapped to 0x0000_0000 through 0x0000_0FFF.

The internal flash memory and RAM addresses of each device are shown below.

Product Name	Flash Size	RAM Size	Flash Address (Single Chip / Single Boot Mode)	RAM Address
TMPM363F10FG	1024 KB	64 KB	0x0000_0000 to 0x000F_FFFF 0x3F80_0000 to 0x3F8F_FFFF	0x2000_0000 to 0x2000_FFFF

Single Chip Mode

0xFFFF_FFFF

Internal I/O   0x4000_7FFF

                    0x4000_0000

Internal RAM   0x2000_FFFF
(64 KB)
                    0x2000_0000

Internal Flash ROM 0x000F_FFFF
(1024 KB)
                    0x0000_0000

Single Boot Mode

0xFFFF_FFFF

Internal I/O   0x4000_7FFF

                    0x4000_0000

Internal Flash ROM 0x3F8F_FFFF
(1024 KB)
                    0x3F80_0000
Reserved   0x3F7F_FFFF

                    0x3F7F_F000

Internal RAM   0x2000_FFFF
(64 KB)
                    0x2000_0000

Internal BOOT ROM   0x0000_0FFF
(4 KB)
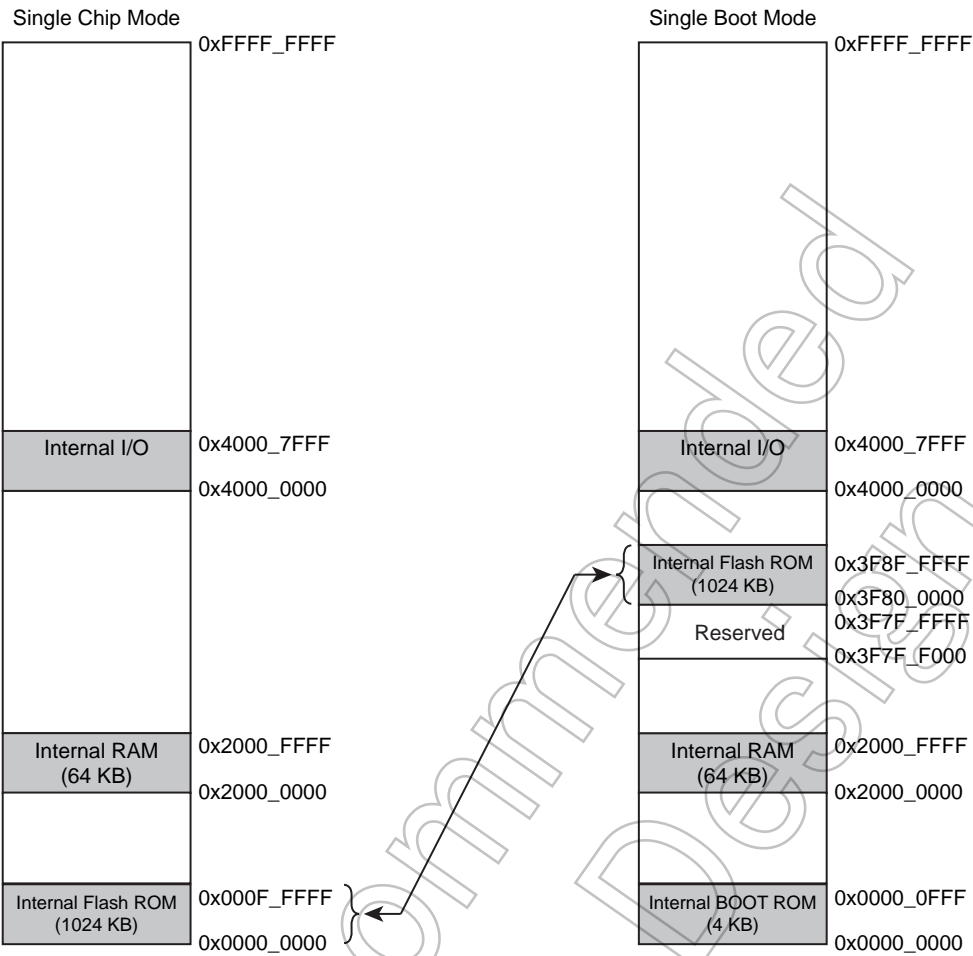   0x0000_0000

Figure 24-3 Memory Maps for TMPM363F10FG

24.2.6   Interface specification

In Single Boot mode, an SIO channel is used for communications with a programming controller. The same configuration is applied to a communication format on a programming controller to execute the on-board programming. Both UART (asynchronous) and I/O Interface (synchronous) modes are supported. The communication formats are shown below.

- UART communication

    Communication channel : SIO channel 4

    Serial transfer mode : UART (asynchronous), half -duplex, LSB first

    Data length : 8 bits

    Parity bit : None

    STOP bit : 1 bit

    Baud rate : Arbitrary baud rate

- I/O Interface mode

    Communication channel : SIO channel 4

    Serial transfer mode : I/O interface mode, full -duplex, LSB first

    Synchronization clock (SCLK4) : Input mode

    Handshaking signal : PN3 configured as an output mode

    Baud rate : Arbitrary baud rate

Table 24-3 Required Pin Connections

Pin		Interface	
		UART	I/O Interface Mode
Power supply pins	RVDD3	o	o
	AVDD3	o	o
	DVDD3B	o	o
	DVDD3A	o	o
	AVSS	o	o
	DVSS	o	o
Mode-setting pin	$\overline{BOOT}$ (PI0)	o	o
Reset pin	$\overline{RESET}$	o	o
Communication pin	TXD4 (PN0)	o	o
	RXD4 (PN1)	o	o
	SCLK4 (PN2)	×	o (Input mode)
	PN3	×	o (Output mode)

24.2.7 Data Transfer Format

Table 24-4, Table 24-6 to Table 24-9 illustrate the operation commands and data transfer formats at each operation mode. In conjunction with this section, refer to "24.2.10 Operation of Boot Program".

Table 24-4 Single Boot Mode Commands

Code	Command
0x10	RAM transfer
0x20	Show Flash Memory SUM
0x30	Show Product Information
0x40	Chip and protection bit erase

24.2.8 Restrictions on internal memories

Single Boot Mode places restrictions on the internal RAM and ROM as shown in Table 24-5.

Table 24-5 Restrictions in Single Boot Mode

Memory	Details
Internal RAM	A program contained in the BOOT ROM uses the area, through 0x2000_0000 to 0x2000_03FF, as a work area. Store the RAM transfer program from 0x2000_0400 through the end address of RAM.
Internal ROM	The following addresses are assigned for storing software ID information and passwords. Storing program in these addresses is not recommendable. 0x3F8F_FFF0 to 0x3F8F_FFFF

24.2.9 Transfer Format for Boot Program

The following tables shows the transfer format for each Boot program command. Use this section in conjunction with Chapter "24.2.10 Operation of Boot Program".

24.2.9.1   RAM Transfer

Table 24-6 Transfer Format for the RAM Transfer Command

	Byte	Data Transferred from the Controller to the TMPM363F10FG	Baud rate	Data Transferred from the TMPM363F10FG to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode : 0x86 For I/O Interface mode : 0x30	Desired baud rate (Note 1)	–
	2 byte	–		ACK for the serial operation mode byte ・For UART mode - Normal acknowledge : 0x86 (The boot program aborts if the baud rate can not be set correctly.) ・For I/O Interface mode - Normal acknowledge :0x30
	3 byte	Command code (0x10)		–
	4 byte	–		ACK for the command code byte (Note 2) - Normal acknowledge : 0x10 - Negative acknowledge : 0xX1 - Communication error : 0xX8
	5 byte to 16 byte	Password sequence (12 bytes)) 0x3F8F_FFF4 to 0x3F8F_FFFF		–
	17 byte	Check SUM value for bytes 5 to 16		–
	18 byte	–		ACK for the checksum byte (Note 2) - Normal acknowledge : 0x10 - Negative acknowledge : 0xX1 - Communication error : 0xX8
	19 byte	RAM storage start address 31 to 24		–
	20 byte	RAM storage start address 23 to 16		–
	21 byte	RAM storage start address 15 to 8		–
	22 byte	RAM storage start address 7 to 0		–
	23 byte	RAM storage start address 15 to 8		–
	24 byte	RAM storage start address 7 to 0		–
	25 byte	Check SUM value for bytes 19 to 24		–
	26 byte	–		ACK for the checksum byte (Note 2) - Normal acknowledge : 0x10 - Negative acknowledge : 0xX1 - Communication error : 0xX8
	27 byte to mbyte	RAM storage data		–
	m+ 1 byte	Checksum value for bytes 27 to m		–
	m+ 2 byte	–		ACK for the checksum byte (Note 2) - Normal acknowledge : 0x10 - Negative acknowledge : 0xX1 - Communication error : 0xX8
RAM	m+ 3 byte	–		Jump to RAM storage start address

Note 1: **In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.**

Note 2: **In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.**

Note 3: **The 19th to 25th bytes must be within the RAM address range from 0x2000_0400 through the end address of RAM.**

24.2.9.2 Show Flash Memory SUM

Table 24-7 Transfer Format for the Show Flash Memory SUM Command

	Byte	Data Transferred from the Controller to the TMPM363F10FG	Baud rate	Data Transferred from the TMPM363F10FG to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode : 0x86 For I/O Interface mode : 0x30	Desired baud rate (Note 1)	–
	2 byte	–		ACK for the serial operation mode byte ・For UART mode - Normal acknowledge : 0x86 (The boot program aborts if the baud rate can not be set correctly.) ・For I/O Interface mode - Normal acknowledge :0x30
	3 byte	Command code (0x20)		–
	4 byte	–		ACK for the command code byte (Note 2) - Normal acknowledge : 0x10 - Negative acknowledge : 0xX1 - Communication error : 0xX8
	5 byte	–		SUM (upper byte)
	6 byte	–		SUM (lower byte)
	7 byte	–		Checksum value for byte 5 and 6
	8 byte	(Wait for the next command code.)		–

Note 1: **In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.**

Note 2: **In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.**

24.2.9.3   Transfer Format for the Show Product Information

Table 24-8 Transfer Format for the Show Product Information Command

	Byte	Data Transferred from the Controller to the TMPM363F10FG	Baud rate	Data Transferred from the TMPM363F10FG to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode : 0x86 For I/O Interface mode : 0x30	Desired baud rate (Note 1)	–
	2 byte	–		ACK for the serial operation mode byte ・For UART mode - Normal acknowledge : 0x86 (The boot program aborts if the baud rate can not be set correctly.) ・For I/O Interface mode - Normal acknowledge :0x30
	3 byte	Command code (0x30)		–
	4 byte	–		ACK for the command code byte (Note 2) - Normal acknowledge : 0x10 - Negative acknowledge : 0xX1 - Communication error : 0xX8
	5 byte	–		Flash memory data at address 0x3F8F_FFF0
	6 byte	–		Flash memory data at address 0x3F8F_FFF1
	7 byte	–		Flash memory data at address 0x3F8F_FFF2î'ín
	8 byte	–		Flash memory data at address 0x3F8F_FFF3
	9 byte to 20 byte	–		Product name (12-byte ACCII code) From 9th byte : 'TMPM360F1_ _'
	21 byte to 24 byte	–		Password comparison start address (4 bytes) From 21st byte : 0xF4, 0xFF, 0x8F, 0x3F
	25 byte to 28 byte	–		RAM start address (4 bytes) From 25th byte : 0x00, 0x00, 0x00, 0x20
	29 byte to 32 byte	–		Dummy data (4 bytes) From 29th byte : 0x00, 0x00, 0x00, 0x00
	33 byte to 36 byte	–		RAM end address (4bytes) From 33th byte : 0xFF, 0xFF, 0x00, 0x20
	37 byte to 40 byte	–		Dummy date (4bytes) From 37th byte : 0x00, 0x00, 0x00, 0x00
	41 byte to 44 byte	–		Dummy date (4bytes) From 41st byte 0x00, 0x00, 0x00, 0x00

Table 24-8 Transfer Format for the Show Product Information Command

	Byte	Data Transferred from the Controller to the TMPM363F10FG	Baud rate	Data Transferred from the TMPM363F10FG to the Controller
	45 byte to 46 byte	–		Dummy data (2 bytes) From 45th byte : 0x00, 0x00
	47 byte to 50 byte	–		Flash memory start address (4 bytes) From 47th byte : 0x00, 0x00, 0x80, 0x3F
	51 byte to 54 byte	–		Flash memory end address (4 bytes) From 51st byte : 0xFF, 0xFF, 0x8F, 0x3F
	55 byte to 56 byte	–		Flash memory block count (2 bytes) From 55th byte : 0x0A, 0x00
	57 byte to 60 byte	–		Start address of a group of the same-size (16K) flash blocks (4 bytes) From 57th byte : 0x00, 0x00, 0x00, 0x00 TMPM363F10FG does not have 16KB block.
	61 byte to 64 byte	–		Size (in halfwords) of the same-size (16K) flash blocks (4 bytes) From 61st byte : 0x00, 0x00, 0x00, 0x00 TMPM363F10FG does not have 16KB block.
	65 byte	–		Number of flash blocks of the same size (16K) (1 byte) 0x00 TMPM363F10FG does not have 16KB block.
	66 byte to 69 byte	–		Start address of a group of the same-size (32K) flash blocks (4 bytes) From 66th byte : 0x00, 0x00, 0x8F, 0x3F
	70 byte to 73 byte	–		Size (in halfwords) of the same-size (32K) flash blocks (4 bytes) From 70th byte : 0x00, 0x40, 0x00, 0x00
	74 byte	–		Number of flash blocks of the same size (32K) (1 byte) 0x02
	75 byte to 78 byte	–		Start address of a group of the same-size (32K) flash blocks (4 bytes) From 75th byte : 0x00, 0x00, 0x81, 0x3F
	79 byte to 82 byte	–		Size (in halfwords) of the same-size (32K) flash blocks (4 bytes) From 79th byte : 0x00, 0x80, 0x00, 0x00

Table 24-8 Transfer Format for the Show Product Information Command

	Byte	Data Transferred from the Controller to the TMPM363F10FG	Baud rate	Data Transferred from the TMPM363F10FG to the Controller
	83 byte	–		Number of flash blocks of the same size (64K) (1 byte) 0x01
	84 byte to 87 byte	–		Start address of a group of the same-size (128K) flash blocks (4 bytes) From 84th byte : 0x00, 0x00, 0x82, 0x3F
	88 byte to 91 byte	–		Size (in halfwords) of the same-size (128K) flash blocks (4 bytes) From 88th byte : 0x00, 0x00, 0x01, 0x00
	92 byte	–		Number of flash blocks of the same size (128K) (1 byte) 0x07
	93 byte	–		Checksum value for bytes from 5 to 92
	94 byte	(Wait for the next command code.)		–

Note 1: **In I/O Interface mode, the baud rate for the transfers of the first and second byte must be 1/16 of the desired baud rate.**

Note 2: **In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.**

24.2.9.4　Chip Erase and Protect Bit Erase

Table 24-9 Transfer Format for the Chip and Protection Bit Erase Command

	Byte	Data Transferred from the Controller to the TMPM363F10FG	Baud rate	Data Transferred from the TMPM363F10FG to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode : 0x86 For I/O Interface mode : 0x30	Desired baud rate (Note 1)	–
	2 byte	–		ACK for the serial operation mode byte ・For UART mode - Normal acknowledge : 0x86 (The boot program aborts if the baud rate can not be set correctly.) ・For I/O Interface mode - Normal acknowledge :0x30
	3 byte	Command code (0x40)		–
	4 byte	–		ACK for the command code byte (Note 2) - Normal acknowledge : 0x10 - Negative acknowledge : 0xX1 - Communication error : 0xX8
	5 byte	Chip erase command code (0x54)		–
	6 byte	–		ACK for the command code byte (Note 2) - Normal acknowledge : 0x10 - Negative acknowledge : 0xX1 - Communication error : 0xX8
	7 byte	–		ACK for the chip erase command code byte - Normal acknowledge : 0x4F - Negative acknowledge : 0x4C
	8 byte	(Wait for the next command code.)		–

Note 1: **In I/O Interface mode, the baud rate for the transfers of the first and second byte must be 1/16 of the desired baud rate.**

Note 2: **In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.**

24.2.10 Operation of Boot Program

When Single Boot mode is selected, the boot program is automatically executed on startup. The boot program offers these four commands, of which the details are provided on the following subsections.

1. RAM Transfer command

   The RAM Transfer command stores program code transferred from the host controller to the on-chip RAM and executes the program once the transfer is successfully completed. The user program RAM space can be assigned to the range from 0x2000_0400 to the end address of RAM, whereas the boot program area (0x2000_0000 to 0x2000_03FF) is unavailable. The user program starts at the assigned RAM address.

   The RAM Transfer command can be used to download a flash programming routine of your own; this provides the ability to control on-board programming of the flash memory in a unique manner. The programming routine must utilize the flash memory command sequences described in Section 24.3. Before initiating a transfer, the RAM Transfer command verifies a password sequence coming from the controller against that stored in the flash memory.

Note: **If a password is set to 0xFF (erased data), it is difficult to protect data securely due to an easy-to-guess password. Even if Single Boot mode is not used, it is recommended to set a unique value as a password.**

2. Show Flash Memory SUM command

   The Show Flash Memory SUM command adds the entire contents of the flash memory together. The boot program does not provide a command to read out the contents of the flash memory. Instead, the Flash Memory SUM command can be used for software revision management.

3. Show Product Information command

   The Show Product Information command provides the product name, on-chip memory configuration and the like. This command also reads out the contents of the flash memory locations at addresses shown below. In addition to the Show Flash Memory Sum command, these locations can be used for software revision management.

Product name	Area
TMPM363F10FG	0x3F8F_FFF0 to 0x3F8F_FFF3

4. Flash Memory Chip Erase and Protection Bit Erase command

   This command erases the entire area of the flash memory automatically. All the blocks in the memory cell and their protection conditions are erased even when any of the blocks are prohibited from writing and erasing. When the command is completed, the FCSECBIT <SECBIT> bit is set to "1". This command serves to recover boot programming operation when a user forgets the password. Therefore password verification is not executed.

24.2.10.1 RAM Transfer Command

See Table 24-6 for the transfer format of this command.

1. The 1st byte specifies which one of the two serial operation modes is used. For a detailed description of how the serial operation mode is determined, see "24.2.10.6 Determination of a Serial Operation Mode" described later. If the mode is determined as UART mode, the boot program checks if the baud rate setting can be performed. During the first-byte processing, receiving operation is prohibited. (SC4MOD0<RXE>=0)

   - To communicate in UART mode

     The 1st byte is set to "0x86" and is transmitted from the controller to the target board at the specified baud rate by setting UART. If the serial operation mode is determined as UART, then the boot program checks if the baud rate setting can be performed. If that baud rate cannot be set, the boot program aborts and any subsequent communications cannot be done. Please refer to "Baud rate setting" for the method of judging whether the setting of the baud rate is possible.

   - To communicate in I/O Interface mode

     The 1st byte is set to "0x30" and is transmitted from the controller to the target board at 1/16 of the desired baud rate by the synchronous setting. Same as the 1st byte, a 1/16 of the specified baud rate is used in the 2nd transmission. From the 3rd byte (operation command data), users can transmit data at specified baud rate.

     In I/O interface mode, CPU considers the reception terminal to be an input port and monitors the level of I/O port. If the baud rate is high or operation frequency is high, CPU may not distinguish the level of I/O port. To avoid this situation, the baud rate is set at the 1/16 of desired baud rate in the I/O interface. When the serial operation mode is determined as I/O Interface mode, SCLK Input mode is set. The controller must ensure that its AC timing restrictions are satisfied at the selected baud rate. In the case of I/O Interface mode, the boot program does not check the receive error flag; thus there is no error acknowledge responce (bit 3, 0x08).

2. The 2nd byte, transmitted from the target board to the controller, is an acknowledge response to the 1st byte where the serial operation mode is set. When 1st byte is determined as UART and can be set at the specified baud rate, data "0x86" is transmitted. When 1st byte is determined as I/O interface, data "0x30" is transmitted.

   - UART mode

     The 2nd byte is used for distinguishing whether the baud rate can be set. If the baud rate can be set, a value of SC4BRCE is renewed and data "0x86" is sent to the controller. If the baud rate cannot be set, transmit operation is stopped and no data is transmitted. After transmission of 1st byte completed, the controller allows for five seconds of time-out. If it does not receive 0x86 within the allowed time-out period, the controller should give up the communication. Receiving operation is permitted by setting SC4MOD0<RXE>=1, before loading 0x86 to the SIO transmit buffer.

   - I/O Interface mode

     The boot program sets a value of the SC4MOD0 and SC4CR registers to configure the the I/O Interface mode and writes 0x30 to the SC4BUF. Then, the SIO4 waits for the SCLK4 signal to come from the controller. After the transmission of the 1st byte completed, the controller should send the SCLK clock to the target board after a certain idle time (several microseconds). This must be done at 1/16 of the desired baud rate. If the 2nd byte, which is from the target board to the controller, is 0x30, then the controller regards it as communication possible. From the 3rd byte, users can transmit data at specified baud rate. Receiving operation is permitted by setting SC4MOD0<RXE>=1, before loading 0x86 to the SIO.

3. The 3rd byte transmitted from the controller to the target board is a command. The code for the RAM Transfer command is 0x10.

4. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there is a receive error, the boot program transmits 0xX8 (bit 3) and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

    If the 3rd byte is equal to any of the command codes listed in Table 24-4, the boot program echoes it back to the controller. When the RAM Transfer command is received, the boot program echoes back a value of 0x10 and then branches to the RAM Transfer routine. Once this branch is taken, password verification is done. Password verification is detailed in the later Section "Password". If the 3rd byte is not a valid command, the boot program sends back 0xX1 (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

5. The 5th to 16th bytes transmitted from the controller to the target board, are a 12-byte password. Each byte is compared to the contents of following addresses in the flash memory. The verification is started with the 5th byte. If the password verification fails, the RAM Transfer routine sets the password error flag.

Product name	Area
TMPM363F10FG	0x3F8F_FFF4 to 0x3F8F_FFFF

6. The 17th byte is a checksum value for the password sequence (5th to 16th bytes). To calculate the checksum value for the 12-byte password, add the 12 bytes together, ignore the carries and caluculate the 8-bit two's complement by using lower 8 bits then transmit this checksum value from the controller. The checksum calculation is described in details in the later Section "Checksum Calculation".

7. The 18th byte, transmitted from the target board to the controller, is an acknowledge response to the 5th to 17th bytes. First, the RAM Transfer routine checks for a receive error in the 5th to 17th byte. If there is a receive error, the boot program sends back 0x18 (bit 3) and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., 1). When the SIO4 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

    Next, the RAM Transfer routine performs the checksum operation to ensure 17th byte data integrity. Adding the series of the 5th to 16th bytes must result in 0x00 (with the carry dropped). In case of a checksum error, the RAM Transfer routine sends back 0x11 to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

    Finally, the password verification result is checked. If the following case is generated, the boot program transmits an acknowledge response (bit 0, 0x11) as a password error and waits for next operation command (3rd byte).

    · Irrespective of the result of the password comparison, all the 12 bytes of a password in the flash memory are the same value other than 0xFF.

    · Not the entire password bytes transmitted from the controller matched those contained in the flash memory.

    When all the above verification has been successful, the RAM Transfer routine returns a normal acknowledge response (0x10) to the controller.

8.   The 19th to 22nd bytes, transmitted from the controller the target board, indicate the start address of the RAM region where subsequent data (e.g., a flash programming routine) should be stored. The 19th byte corresponds to bits 31 to 24 of the address and the 22nd byte corresponds to bits 7 to 0 of the address. The start address of the stored RAM must be even address.

9.   The 23rd and 24th bytes, transmitted from the controller to the target board, indicate the number of bytes that will be transferred from the controller to be stored in the RAM. The 23rd byte corresponds to bits 15 to 8 of the number of bytes to be transferred, and the 24th byte corresponds to bits 7 to 0 of the number of bytes.

10. The 25th byte is a checksum value for the 19th to 24th bytes. To calculate the checksum value, add all these bytes together, ignore the carries and caluculate the 8-bit two's complement by using lower 8 bits then transmit this checksum value from the controller. The checksum calculation is described in detail in the later Section "24.2.10.9 Checksum Calculation".

11. The 26th byte, transmitted from the target board to the controller, is an acknowledge response to the 19th to 25th bytes of data. First, the RAM Transfer routine checks for a receive error in the 19th to 25th bytes. If there is a receive error, the RAM Transfer routine sends back 0x18 and returns to the command wait state (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., 1). When the SIO4 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

   Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 19th to 24th bytes must result in 0x00 (with the carry dropped). In case of a checksum error, the RAM Transfer routine sends back 0x11 to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

   ·   The 19th to 25th bytes data must be within the range of 0x2000_0400 to the end address of RAM.

   When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (0x10) to the controller.

12. The 27th to mth bytes from the controller are stored in the on-chip RAM of the TMPM363F10FG. Storage begins at the address specified by the 19th to 22nd bytes and continues for the number of bytes specified by the 23rd to 24th bytes.

13. The (m+1) th byte is a checksum value. To calculate the checksum value, add the 27th to mth bytes together, ignore the carries and calculate the 8-bit two's complement by using lower 8 bits then transmit this checksum value from the controller. The checksum calculation is described in detail in later Section "24.2.10.9 Checksum Calculation".

14. The (m+2) th byte is a acknowledge response to the 27th to (m+1) th bytes. First, the RAM Transfer routine checks for a receive error in the 27th to (m+1) th bytes. If there is a receive error, the RAM Transfer routine sends back 0x18 (bit 3) and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., 1). When the SIO4 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

   Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 27th to (m+1) th bytes must result in 0x00 (with the carry dropped). In case of a checksum error, the RAM Transfer routine sends back 0x11 (bit 0) to the controller and returns to the command wait state (i.e., the 3rd byte) again. When the above checks have been completed successfully, the RAM Transfer routine returns a normal acknowledge response (0x10) to the controller.

15. If the (m+2) th byte was a normal acknowledge response, a branch is made to the address specified by the 19th to 22nd bytes.

24.2.10.2 Show Flash Memory SUM Command

See Table 24-7 for the transfer format of this command.

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Flash Memory Sum command is 0x20.

3. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there is a receive error, the boot program transmits 0xX8 (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined- they hold the same values as the upper four bits of the previously issued command. When the SIO4 is configured for I/O Interface mode, the boot program does not check for a receive error.

   If the 3rd byte is equal to any of the command codes listed in Table 24-4, the boot program echoes it back to the controller. When the Show Flash Memory Sum command is received, the boot program echoes back a value of 0x20 and then branches to the Show Flash Memory Sum routine. If the 3rd byte is not a valid command, the boot program sends back 0xX1 (bit 0) to the controller and returns to the command wait state (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

4. The Show Flash Memory Sum routine adds all the bytes of the flash memory together. The 5th and 6th bytes, transmitted from the target board to the controller, indicate the upper and lower bytes of this total sum, respectively. For details on sum calculation, see Section "24.2.10.8 Calculation of the Show Flash Memory Sum Command".

5. The 7th byte is a checksum value for the 5th and 6th bytes. To calculate the checksum value, add the 5th and 6th bytes together, ignore the carry and calculate the 8-bit two's complement by using lower 8 bits then transmit this checksum value from the controller.

6. The 8th byte is the next command code.

24.2.10.3 Show Product Information Command

See Table 24-8 for the transfer format of this command.

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Product Information command is 0x30.

3. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there is a receive error, the boot program transmits 0xX8 (bit 3) and returns to

the command wait state again. In this case, the upper four bits of the acknowledge response are undefined- they hold the same values as the upper four bits of the previously issued command. When the SIO4 is configured for I/O Interface mode, the boot program does not check for a receive error.

If the 3rd byte is equal to any of the command codes listed in Table 24-4, the boot program echoes it back to the controller. When the Show Flash Memory Sum command is received, the boot program echoes back a value of 0x30 and then branches to the Show Flash Memory Sum routine. If the 3rd byte is not a valid command, the boot program sends back 0xX1 (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

4. The 5th to 8th bytes, transmitted from the target board to the controller, are the data read from addresses shown below in the flash memory. Software version management is possible by storing a software ID in these locations.

Product name	Area
TMPM363F10FG	0x3F8F_FFF0 to 0x3F8F_FFF3

5. The 9th to 20th bytes, transmitted from the target board to the controller, indicate the product name as shown below (where [ ] is a space) in ASCII code.

Product name	Core
TMPM363F10FG	T, M, P, M, 3, 6, 0, F, 1, _, [ ], _

6. The 21st to 24th bytes, transmitted from the target board to the controller, indicate the start address of the flash memory area contained the password. Each product has own start address shown below. Starting from the 21st byte, the following values are transmitted.

Product name	Address
TMPM363F10FG	0xF4, 0xFF, 0x8F, 0x3F

7. The 25th to 28th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip RAM. TMPM363F10FG has own start address shown below. Starting from the 25th byte, the following values are transmitted.

Product name	Address
TMPM363F10FG	0x00, 0x00, 0x00, 0x20

8. The 29th to 32nd bytes, transmitted from the target board to the controller, are dummy data. Starting from the 29th byte, 0x00, 0x00, 0x00, 0x00 are transmitted.

9. The 33rd to 36th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip RAM. TMPM363F10FG has own end address shown below. Starting from the 33th byte, the following values are transmitted.

Product name	Address
TMPM363F10FG	0xFF, 0xFF, 0x00, 0x20

10. The 37th to 40th bytes, transmitted from the target board to the controller, are 0x00, 0x00, 0x00 and 0x00. The 41st to 44th bytes, transmitted from the target board to the controller, are 0x00, 0x00, 0x00 and 0x00.

11. The 45th and 46th bytes transmitted are 0x00, 0x00.

12. The 47th to 50th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip flash memory, are 0x00, 0x00, 0x80, and 0x3F.

Product name	Address
TMPM363F10FG	0x00, 0x00,0x80, 0x3F

13. The 51st to 54th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip flash memory. Each product has own end address shown below.Starting from the 51th byte, the following values are transmitted.

Product name	Address
TMPM363F10FG	0xFF, 0xFF, 0x8F, 0x3F

14. The 55th to 56th bytes, transmitted from the target board to the controller, indicate the number of flash blocks available. Each product transmits own number shown below. Starting from the 55th byte, the following values are transmitted.

Product	Number of flash blocks
TMPM363F10FG	0x0A, 0x00

15. The 57th to 92nd bytes, transmitted from the target board to the controller, contain information about the flash blocks. Flash blocks of the same size are treated as a group. Information about the flash blocks indicate the start address of a group, the size of the blocks in that group (in halfwords) and the number of the blocks in that group. The 57th to 65th bytes are the information about the 16-kbyte blocks. The 66th to 74th bytes are the information about the 32-kbyte blocks. The 75th to 83rd bytes are the information about the 64-kbyte blocks. The 84th to 92nd bytes are the information about the 128-kbyte blocks. See Table 24-8 for the values of bytes transmitted.

16. The 93rd byte, transmitted from the target board to the controller, is a checksum value for the 5th to 92nd bytes. To calculate the checksum value, add all these bytes together, ignore the carries and calculate the 8-bit two's complement by using lower 8 bits.

17. The 94th byte is the next command code.

24.2.10.4 Chip and Protection Bit Erase Command

See Table 24-9 for the transfer format of this command.

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. From the Controller to the TMPM363F10FG

   The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Product Information command is 0x40.

3. From TMPM363F10FG to the Controller

   The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte.

   Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits 0xX8 (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

   If the 3rd byte is equal to any of the command codes listed in Table 24-4, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 0x40. If the 3rd byte is not a valid command, the boot program sends back 0xX1 (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

4. From the controller to the TMPM363F10FG

   The 5th byte, transmitted from the target board to the controller, is the Chip Erase Enable command code (0x54).

5. From TMPM363F10FG to the Controller

   The 6th byte, transmitted from the target board to the controller, is an acknowledge response to the 5th byte.

   Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits 0xX8 (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

   If the 5th byte is equal to any of the command codes to enable erasing, the boot program echoes it back to the controller. When the Chip and Protection Erase command was received, the boot program echoes back a value of 0x54 and then branches to the Chip Erase routine. If the 5th byte is not a valid command, the boot program sends back 0xX1 (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

6. From TMPM363F10FG to the Controller

   The 7th byte indicates whether the Chip Erase command is normally completed or not.

   At normal completion, completion code (0x4F) is sent.

   When an error was detected, error code (0x4C) is sent.

7. The 9th byte is the next command code.

24.2.10.5   Acknowledge Responses

The boot program represents processing states with specific codes. Table 24-10 to show the values of possible acknowledge responses to the received data. The upper four bits of the acknowledge response are equal to those of the command being executed. The 3rd bit indicates a receive error. The 0th bit indicates an invalid command error, a checksum error or a password error. The 1st bit and 2nd bit are always "0". Receive error checking is not done in I/O Interface mode.

Table 24-10 ACK Response to the Serial Operation Mode Byte

Return Value	Meaning
0x86	The SIO can be configured to operate in UART mode. (See Note)
0x30	The SIO can be configured to operate in I/O Interface mode.

Note:**In the UART mode, if the baud rate setting cannot be set, the communication is stopped without any response.**

Table 24-11 ACK Response to the Command Byte

Return Value	Meaning
0x?8 (See note)	A receive error occurred while receiving a command code.
0x?1 (See note)	An undefined command code was received. (Reception was completed normally.)
0x10	The RAM Transfer command was received.
0x20	The Show Flash Memory Sum command was received.
0x30	The Show Product Information command was received.
0x40	The Chip Erase command was received.

Note:**The upper four bits of the ACK response are the same as those of the previous command code.**

Table 24-12 ACK Response to the Checksum Byte

Return Value	Meaning
0xN8 (See note)	A receive error occurred.
0xN1 (See note)	A checksum or password error occurred.
0xN0 (See note)	The checksum was correct.

Note:**The upper four bits of the ACK response are the same as those of the operation command code. For example, it is 1 (N ; RAM transfer command data [7:4]) when password error occurs.**

Table 24-13 ACK Response to Chip and Protection Bit Erase Byte

Return Value	Meaning
0x54	The Chip Erase enabling command was received.
0x4F	The Chip Erase command was completed.
0x4C	The Chip Erase command was abnormally completed.

24.2.10.6 Determination of a Serial Operation Mode

The first byte from the controller determines the serial operation mode. To use UART mode for communications between the controller and the target board, the controller must firstly send a value of 0x86 at a desired baud rate to the target board. To use I/O Interface mode, the controller must send a value of 0x30 at 1/16 of the desired baud rate. Figure 24-4 shows the waveforms for the first byte in each mode.
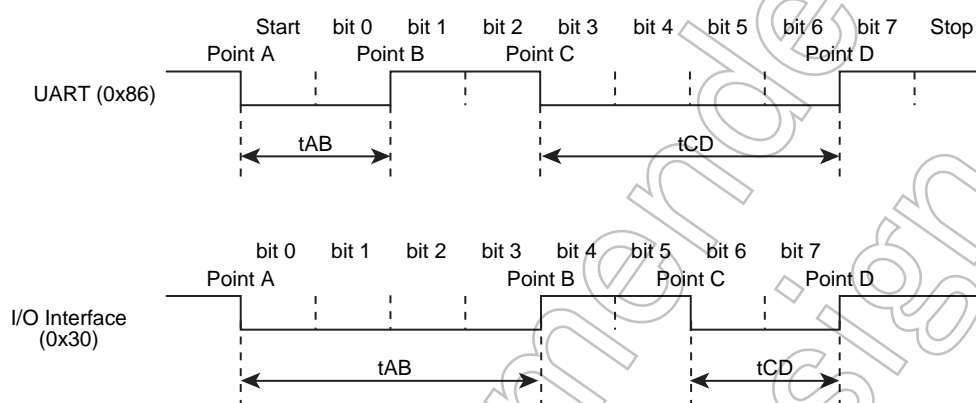

Figure 24-4 Serial Operation Mode Byte

After $\overline{\text{RESET}}$ is released, the boot program monitors the first serial byte from the controller, with the SIO reception disabled, and calculates the intervals of tAB, tAC and tAD. Figure 24-5 shows a flowchart describing the steps to determine the intervals of tAB, tAC and tAD. As shown in the flowchart, the boot program captures timer counts when each time the logic transition occurs in the first serial byte. Consequently, the calculated tAB, tAC and tAD intervals tend to have slight errors. If the transfer goes at a high baud rate, the CPU might not be able to keep up with the speed of logic transitions at the serial receive pin. In particular, I/O Interface mode may have this problem since its baud rate is generally much higher than that for UART mode. To avoid such a situation, the controller should send the first serial byte at 1/16 of the desired baud rate.

The flowchart in Figure 24-5 shows how the boot program distinguishes between UART and I/O Interface modes. If the length of tAB is equal to or less than the length of tCD, the serial operation mode is determined as UART mode. If the length of tAB is greater than the length of tCD, the serial operation mode is determined as I/O Interface mode. Note that if the baud rate is too high or the timer operating frequency is too low, each timer value becomes small. It causes an unintentional behavior of the controller. To prevent this problem, reset UART mode within the programming routine.

For example, the serial operation mode may be determined to be I/O Interface mode when the intended mode is UART mode. To avoid such a situation, when UART mode is utilized, the controller should allow for a time-out period within which it expects to receive an echo-back (0x86) from the target board. The controller should give up the communication if it fails to get that echo-back within the allowed time. When I/O Interface mode is utilized, once the first serial byte has been transmitted, the controller should send the SCLK clock after a certain idle time to get an acknowledge response. If the received acknowledge response is not 0x30, the controller should give up further communications.

When the intended mode is I/O interface mode, the first byte does not have to be 0x30 as long as tAB is greater than tCD as shown above. 0x91, 0xA1 or 0xB1 can be sent as the first byte code to determine the falling edges of Point A and Point C and the rising edges of Point B and Point D. If tAB is greater than tCD and SIO is selected by the resolution of the operation mode determination, the second byte code is 0x30 even though the transmitted code on the first byte is not 0x30 (The first byte code to determine I/O interface mode is described as 0x30).
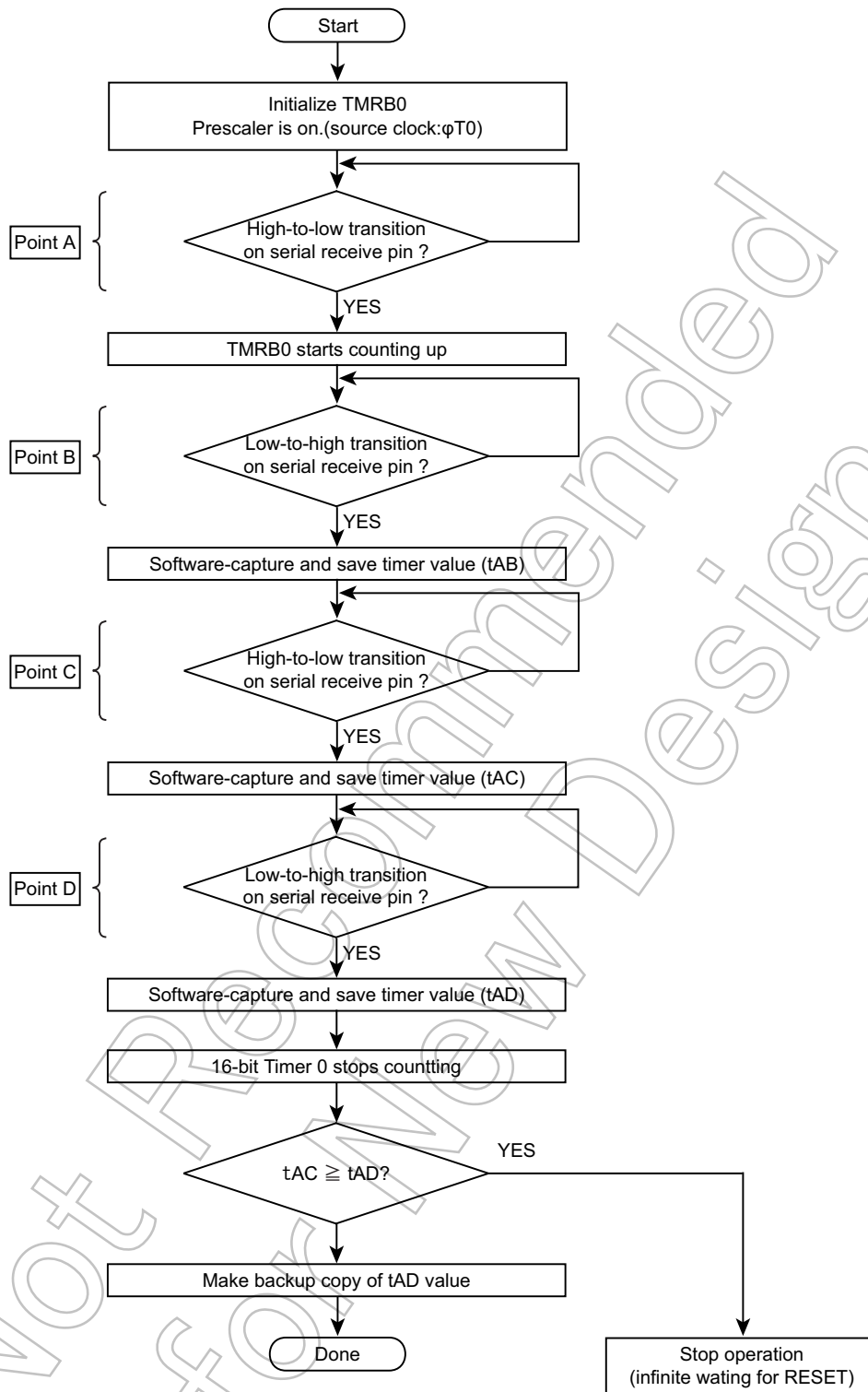
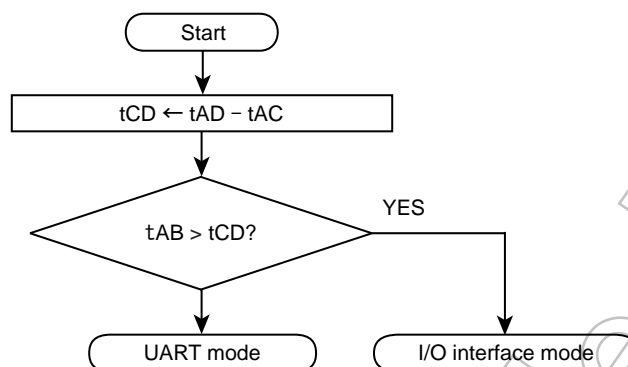Figure 24-5 Serial Operation Mode Byte Reception Flowchart

```
                            ┌──────────────┐
                            │    Start     │
                            └──────┬───────┘
                                   │
                        ┌──────────▼──────────┐
                        │   tCD ← tAD − tAC    │
                        └──────────┬──────────┘
                                   │
                              ◇─────────◇        YES
                         ◇   tAB > tCD?   ◇ ─────────┐
                              ◇─────────◇            │
                                   │                 │
                                   │                 │
                          ┌────────▼───────┐  ┌──────▼─────────────┐
                          │   UART mode    │  │  I/O interface mode │
                          └────────────────┘  └────────────────────┘
```

Figure 24-6 Serial Operation Mode Determination Flowchart

24.2.10.7   Password

The RAM Transfer command (0x10) causes the boot program to perform password verification. Following an echo-back of the command code, the boot program verifies the contents of the 12-byte password area within the flash memory. The following table shows the password area of each product.

Product name	Area
TMPM363F10FG	0x3F8F_FFF4 to 0x3F8F_FFFF

Note: **If a password is set to 0xFF (erased data area), it is difficult to protect data securely due to an easy-to-guess password. Even if Single Boot mode is not used, it is recommended to set a unique value as a password.**

If all these address locations contain the same bytes of data other than 0xFF, a password area error occurs as shown in Figure 24-7. In this case, the boot program returns an error acknowledge (0x11) in response to the checksum byte (the 17th byte), regardless of whether the password sequence sent from the controller is all 0xFFs.

Receiving data (5th to 16th bytes) from the controller is compared to the password stored in the flash memory. All of the 12 bytes must match to pass the password verification. Otherwise, a password error occurs, which causes the boot program to reply an error acknowledge in response to the checksum byte (the 17th byte).

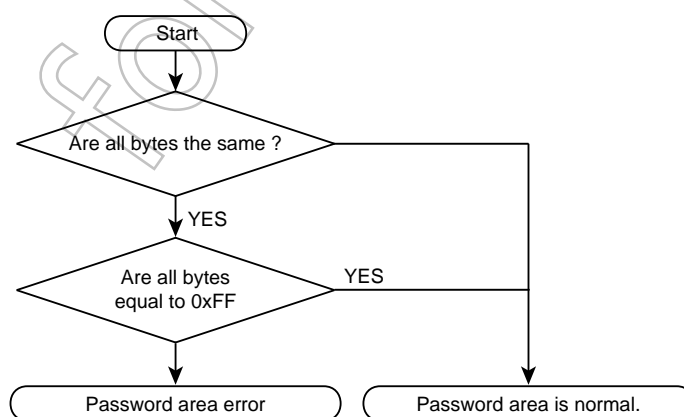The password verification is performed even if the security function is enabled.

```
                          ┌──────────────┐
                          │    Start     │
                          └──────┬───────┘
                                 │
                            ◇─────────◇
                       ◇  Are all bytes the same ?  ◇ ───────────────┐
                            ◇─────────◇                               │
                                 │ YES                                │
                                 │                                    │
                            ◇─────────◇            YES                │
                       ◇   Are all bytes   ◇ ──────────┐              │
                       ◇   equal to 0xFF   ◇           │              │
                            ◇─────────◇                │              │
                                 │                     │              │
                       ┌─────────▼────────┐   ┌────────▼──────────────▼──┐
                       │ Password area error │  │  Password area is normal. │
                       └──────────────────┘   └──────────────────────────┘
```

Figure 24-7 Password Area Verification Flowchart

24.2.10.8   Calculation of the Show Flash Memory Sum Command

The result of the sum calculation ("byte + byte + byte + ・・・") is responded by a half-word quantity. The Show Flash Memory Sum command adds all 512 Kbytes of the flash memory together and provides the total sum as a halfword quantity. The sum is sent to the controller, with the upper eight bits first, followed by the lower eight bits.

Example)

0xA1
0xB2
0xC3
0xD4

For the interest of simplicity, assume the depth of the flash memory is four location. Then the sum of the four bytes is calculated as :
   0xA1 + 0xB2 + 0xC3 + 0xD4 = 0x02EA
Hence, 0x02 is first sent to the controller, followed by 0xEA.

24.2.10.9   Checksum Calculation

The checksum byte for a series of bytes of data is calculated by adding the bytes together with ignoring the carries and calculating the 8-bit two's complement by using lower 8 bits. The Show Flash Memory Sum command and the Show Product Information command perform the checksum calculation. The controller must perform the same checksum operation in transmitting checksum bytes.

Example) Assume the Show Flash Memory Sum command provides the upper and lower bytes of the sum as 0xE5 and 0xF6. To calculate the checksum for a series of 0xE5 and 0xF6:

Add the bytes together

0xE5 + 0xF6 = 0x1DB

Calculate the two's complement by using lower 8 bits, and that is the checksum byte. Then send 0x25 to the controller.

0 - 0xDB = 0x25

24.2.11   General Boot Program Flowchart

Figure 24-8 shows an overall flowchart of the boot program.
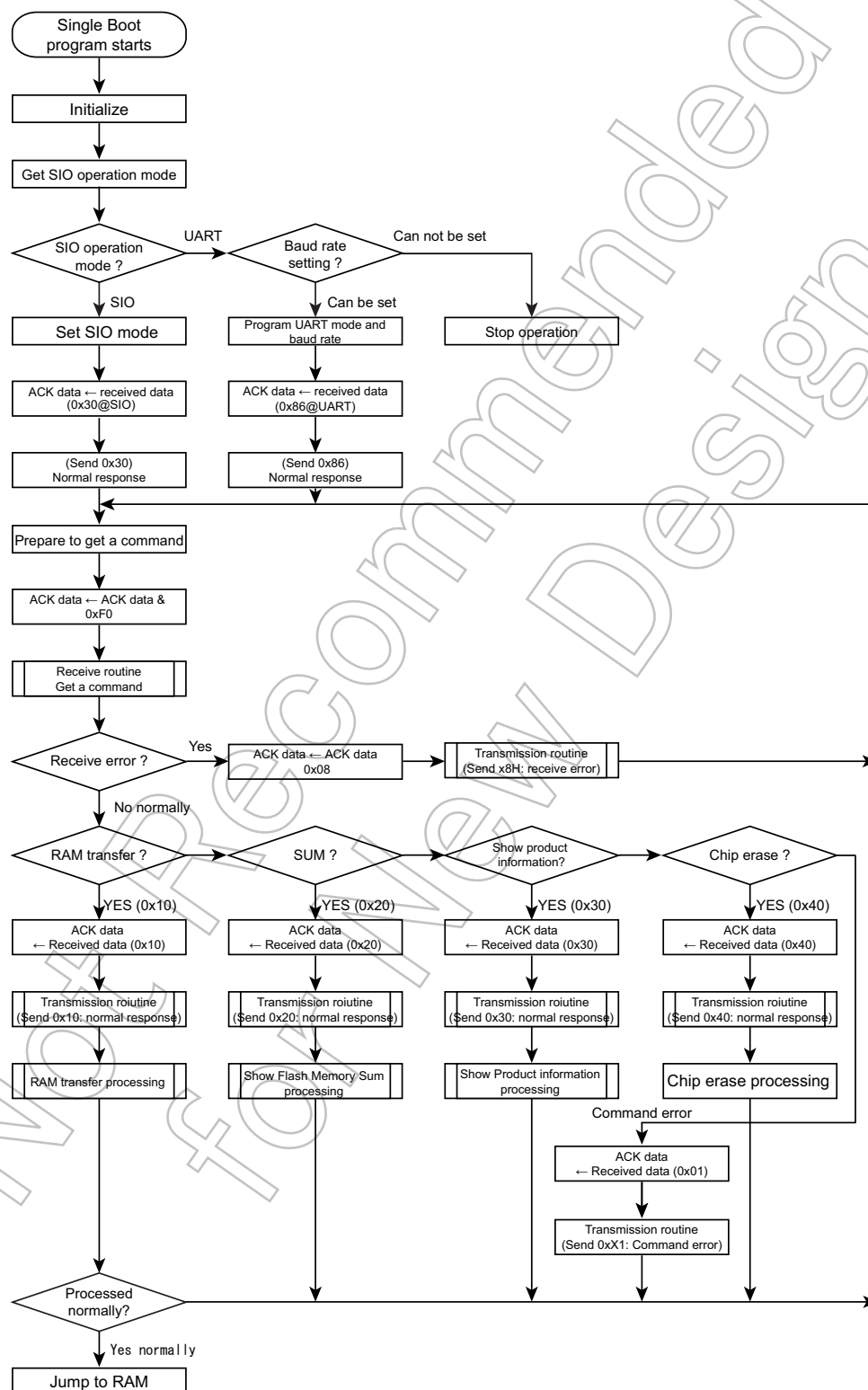

Figure 24-8 Overall Boot Program Flowchart

24.3 On-board Programming of Flash Memory (Rewrite/Erase)

In on-board programming, the CPU is to execute software commands for rewriting or erasing the flash memory. The rewrite/erase control program should be prepared by the user beforehand. Because the flash memory content cannot be read while it is being written or erased, it is necessary to run the rewrite/erase program from the internal RAM after shifting to the user boot mode.

24.3.1 Flash Memory

Except for some functions, writing and erasing flash memory data are in accordance with the standard JEDEC commands.

In writing or erasing, use 32-bit data transfer command of the CPU to enter commands to the flash memory. Once the command is entered, the actual write or erase operation is automatically performed internally.

Table 24-14 Flash Memory Functions

Major functions	Description
Automatic page program	Writes data automatically per page.
Automatic chip erase	Erase the entire area of the flash memory automatically.
Automatic block erase	Erases a selected block automatically.
Protect function	The write or erase operation can be individually inhibited for each block.

24.3.1.1   Block Configuration

(1)   TMPM363F10FG

User Boot Mode	Single Boot Mode		Page Configuration
0x000F_FFFF	0x3F8F_FFFF		
		128K bytes (BLOCK0)	128 words × 256
0x000E_0000	0x3F8E_0000		
		128K bytes (BLOCK1)	128 words × 256
0x000C_0000	0x3F8C_0000		
		128K bytes (BLOCK2)	128 words × 256
0x000A_0000	0x3F8A_0000		
		128K bytes (BLOCK3)	128 words × 256
0x0008_0000	0x3F88_0000		
		128K bytes (BLOCK4)	128 words × 256
0x0006_0000	0x3F86_0000		
		128K bytes (BLOCK5)	128 words × 256
0x0004_0000	0x3F84_0000		
		128K bytes (BLOCK6)	128 words × 256
0x0002_0000	0x3F82_0000		
		64K bytes (BLOCK7)	128 words × 128
0x0001_0000	0x3F81_0000	32K bytes (BLOCK9)	128 words × 64
0x0000_8000	0x3F80_8000		
		32K bytes (BLOCK8)	128 words × 64
0x0000_0000	0x3F80_0000		

Figure 24-9 Block Configuration of Flash Memory (TMPM363F10FG)

24.3.1.2 Basic Operation

This flash memory device has the following two operation modes:

- The mode to read memory data (Read mode)
- The mode to automatically erase or rewrite memory data (Automatic operation)

Transition to the automatic mode is made by executing a command sequence while it is in the memory read mode. In the automatic operation mode, flash memory data cannot be read and any commands stored in the flash memory cannot be executed. In the automatic operation mode, any interrupt or exception generation cannot set the device to the read mode except when a hardware reset is generated. During automatic operation, be sure not to cause any exception other than reset and debug exceptions while a debug port is connected. Any exception generation cannot set the device to the read mode except when a hardware reset is generated.

(1) Read

When data is to be read, the flash memory must be set to the read mode. The flash memory will be set to the read mode immediately after power is applied, when CPU reset is removed, or when an automatic operation is normally terminated. In order to return to the read mode from other modes or after an automatic operation has been abnormally terminated, either the Read/reset command (a software command to be described later) or a hardware reset is used. The device must also be in the read mode when any command written on the flash memory is to be executed.

- Read / reset command and Read command (software reset)

    When ID-Read command is used, the reading operation is terminated instead of automatically returning to the read mode. In this case, the Read/reset command can be used to return the flash memory to the read mode. Also, when a command that has not been completely written has to be canceled, the Read/reset command must be used. The Read command is used to return to the read mode after executing 32-bit data transfer command to write the data "0x0000_00F0" to an arbitrary address of the flash memory.
- With the Read/reset command, the device is returned to the read mode after completing the third bus write cycle.

(2) Command write

This flash memory uses the command control method. Commands are executed by executing a command sequence to the flash memory. The flash memory executes automatic operation commands according to the address and data combinations applied (refer to Command Sequence).

If it is desired to cancel a command write operation already in progress or when any incorrect command sequence has been entered, the Read/reset command is to be executed. Then, the flash memory will terminate the command execution and return to the read.

While commands are generally comprised of several bus cycles and the operation applying to the 32-bit (word) data transmission command to the flash memory is called "bus write cycle". The bus write cycles have a specific sequential order and the flash memory will perform an automatic operation when the sequence of the bus write cycle data and address of command write is operated in accordance with a predefined specific order. If any bus write cycle does not follow a predefined command write sequence, the flash memory will terminate the command execution and return to the read mode.

Note 1: **Command sequences are executed from outside the flash memory area.**

Note 2: **Each bus write cycle must be sequentially executed by 32-bit data transmit command. While a command sequence is being executed, access to the flash memory is prohibited. Also, do not generate any interrupt (except debug exceptions when a debug port is connected). If such an operation is made, it may result in an unexpected read access to the flash memory, and the command sequencer may not be able to correctly recognize**

the command. While it may cause an abnormal termination of the command sequence, it also may cause an incorrect recognition of the command.

Note 3:   **For the command sequencer to recognize a command, the device must be in the read mode prior to executing the command. Be sure to check before the first bus write cycle where FCFLCS <RDY / BSY> is set to "1". It is recommended to subsequently execute a Read command.**

Note 4:   **Upon issuing a command, if any address or data is incorrectly written, be sure to perform a software reset to return to the read mode again.**

24.3.1.3   Reset (Hardware reset)

A hardware reset is used to cancel the operational mode set by the command write operation when forcibly terminated during auto programming/erasing or abnormal termination in the automatic operation.

The flash memory has a reset input as the memory block and it is connected to the CPU reset signal. Therefore, when the $\overline{RESET}$ input pin of this device is set to VIL or when the CPU is reset due to any overflow of the watch dog timer, the flash memory will return to the read mode terminating any automatic operation that may be in progress. It should also be noted that applying a hardware reset during an automatic operation can result in incorrect rewriting of data. In such a case, be sure to perform the rewriting again.

Refer to Section "1.2.1 Reset Operation" for CPU reset operations. After a given reset input, the CPU will read the reset vector data from the flash memory and starts operation after the reset is removed.

24.3.1.4   Commands

(1)   Automatic Page Program

Writing to a flash memory device is to change "1" data cells to "0" data cells. Any "0" data cell cannot be changed to a "1" data cell. For changing "0" data cells to "1" data cells, it is necessary to perform an erase operation.

The automatic page programming function of this device writes data of each page. The TMPM363F10FG contains 128 words in a page. A 128 word block is defined by the same [31:9] address. It starts from the address [8:0] = 0x00 and ends at the address [8:0] = 0x1FF. This programming unit is hereafter referred to as a "page".

Writing to data cells is automatically performed by an internal sequencer and no external control by the CPU is required. The state of automatic page programming (whether it is in writing operation or not) can be checked by FCFLCS [0] <RDY/BSY>.

Also, any new command sequence is not accepted while it is in the automatic page programming mode. If it is desired to interrupt the automatic page programming, use the hardware reset function. If the operation is stopped by a hardware reset operation, it is necessary to once erase the page and then perform the automatic page programming again because writing to the page has not been normally terminated.

The automatic page programming operation is allowed only once for a page already erased. No programming can be performed twice or more. Note that rewriting to a page that has been once written requires execution of the automatic block erase or automatic chip erase command before executing the automatic page programming command again. Note that an attempt to rewrite a page two or more times without erasing the content may cause damages to the device.

No automatic verify operation is performed internally to the device. So, be sure to read the data programmed to confirm that it has been correctly written.

The automatic page programming operation starts when the third bus write cycle of the command cycle is completed. After the fifth bus write cycle, data will be written sequentially starting from the next address of the address specified in the fourth bus write cycle (in the fourth bus write cycle, the page top address will be command written) (32 bits of data is input at one time). Be sure to use

the 32-bit data transfer command in writing commands after the fourth bus cycle. At this time, any 32-bit data transfer commands shall not be placed across word boundary. After the fifth bus write cycle, data is command written to the same page area. Even if it is desired to write the page only partially, it is required to perform the automatic page programming for the entire page. In this case, the address input for the fourth bus write cycle shall be set to the top address of the page. Be sure to perform command write operation with the input data set to "1" for the data cells not to be set to "0". For example, if the top address of a page is not to be written, set the input data in the fourth bus write cycle to 0xFFFFFFFF as a command write.

Once the third bus cycle is executed, the automatic page programming is in operation. This condition can be checked by monitoring FCFLCS<RDY / BSY>. Any new command sequence is not accepted while it is in automatic page programming mode. If it is desired to stop operation, use the hardware reset function. Be careful in doing so because data cannot be written normally if the operation is interrupted. When a single page has been command written with normally terminating the automatic page writing process, FCFLCS<RDY / BSY> is set to "1" then it returns to the read mode.

When multiple pages are to be written, it is necessary to execute the page programming command for each page because the number of pages to be written by a single execution of the automatic page program command is limited to only one page. It is not allowed for automatic page programming to process input data across pages.

Data cannot be written to a protected block. When automatic programming is finished, it automatically returns to the read mode. This condition can be checked by monitoring FCFLCS<RDY/BSY>. If automatic programming has failed, the flash memory is locked in the current mode and will not return to the read mode. For returning to the read mode, it is necessary to execute hardware reset to reset the flash memory or the device. In this case, while writing to the address has failed, it is recommended not to use the device or not to use the block that includes the failed address.

Note: **Software reset becomes ineffective after the fourth bus write cycle of the automatic page programming command.**

(2) Automatic chip erase

The automatic chip erase operation starts when the sixth bus write cycle of the command cycle is completed.

This condition can be checked by monitoring FCFLCS<RDY / BSY>. While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that data has been correctly erased. Any new command sequence is not accepted while it is in an automatic chip erase operation. If it is desired to stop operation, use the hardware reset function. If the operation is forced to stop, it is necessary to perform the automatic chip erase operation again because the data erasing operation has not been normally terminated.

Also, any protected block cannot be erased. If all the blocks are protected, the automatic chip erase operation will not be performed and it returns to the read mode after completing the sixth bus read cycle of the command sequence. When an automatic chip erase operation is normally terminated, it automatically returns to the read mode. If an automatic chip erase operation has failed, the flash memory is locked in the current mode and will not return to the read mode.

For returning to the read mode, it is necessary to execute hardware reset to reset the device. In this case, the failed block cannot be detected. It is recommended not to use the device anymore or to identify the failed block by using the block erase function for not to use the identified block anymore.

(3) Automatic block erase (for each block)

The automatic block erase operation starts when the sixth bus write cycle of the command cycle is completed.

This status of the automatic block erase operation can be checked by monitoring FCFLCS <RDY / BSY>. While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that data has been correctly erased. Any new command sequence is not ac-

cepted while it is in an automatic block erase operation. If it is desired to stop operation, use the hardware reset function. In this case, it is necessary to perform the automatic block erase operation again because the data erasing operation has not been normally terminated.

Also, any protected block cannot be erased. If an automatic block erase operation has failed, the flash memory is locked in the mode and will not return to the read mode. In this case, execute hardware reset to reset the device.

(4)   Automatic programming of protection bits (for each block)

This device is implemented with protection bits. This protection can be set for each block. See Table 24-18 for table of protection bit addresses. This device assigns 1 bit to 1 block as a protection bit. The applicable protection bit is specified by PBA in the seventh bus write cycle. By automatically programming the protection bits, write and/or erase functions can be inhibited (for protection) individually for each block. The protection status of each block can be checked by FCFLCS <BLPRO> to be described later. This status of the automatic programming operation to set protection bits can be checked by monitoring FCFLCS <RDY/BSY>. Any new command sequence is not accepted while automatic programming is in progress to program the protection bits. If it is desired to stop the programming operation, use the hardware reset function. In this case, it is necessary to perform the programming operation again because the protection bits may not have been correctly programmed. If all the protection bits have been programmed, all FCFLCS <BLPRO> are set to "1" indicating that it is in the protected state. This disables subsequent writing and erasing of all blocks.

> Note:**Software reset is ineffective in the seventh bus write cycle of the automatic protection bit programming command. FCFLCS <RDY/BSY> turns to "0" after entering the seventh bus write cycle.**

(5)   Automatic erasing of protection bits

Different results will be obtained when the automatic protection bit erase command is executed depending on the status of the protection bits and the security bits. It depends on whether all <BLPRO> in the FCFLCS register are set to "1" or not, when FCSECBIT<FCSECBIT> is set to "1". Be sure to check the value of FCFLCS <BLPRO> before executing the automatic protection bit erase command. See Chapter "Protect/security function" for details.

- When all the FCFLCS <BLPRO> are set to "1" (all the protection bits are programmed):

   When the automatic protection bit erase command is command written, the flash memory is automatically initialized within the device. When the seventh bus write cycle is completed, the entire area of the flash memory data cells is erased and then the protection bits are erased. This operation can be checked by monitoring FCFLCS <RDY/BSY>. If the automatic operation to erase protection bits is normally terminated, FCFLCS will be set to "0x00000001". Since no automatic verify operation is performed internally to the device, be sure to read the data to confirm that it has been correctly erased. For returning to the read mode while the automatic operation after the seventh bus cycle is in progress, it is necessary to use the hardware reset to reset the device. If this is done, it is necessary to check the status of protection bits by FCFLCS <BLPRO> after retuning to the read mode and perform either the automatic protection bit erase, automatic chip erase, or automatic block erase operation, as required.

- When FCFLCS <BLPRO> include "0" (not all the protection bits are programmed):

   If the automatic protection bit is cleared to "0", the protection condition is canceled. With this device, protection bits can be programmed to an individual block and performed bit-erase operation in the four bits unit as shown in Table 24-18. The target bits are specified in the seventh bus write cycle.The protection status of each block can be checked by FCFLCS <BLPRO> to be described later. This status of the programming operation for automatic protection bits can be checked by monitoring FCFLCS <RDY/BSY>. When the automatic operation to erase protection bits is normally terminated, the protection bits of FCFLCS <BLPRO> selected for erasure are set to "0".

In any case, any new command sequence is not accepted while it is in an automatic operation to erase protection bits. If it is desired to stop the operation, use the hardware reset function. When the automatic operation to erase protection bits is normally terminated, it returns to the read mode.

Note: **The FCFLCS <RDY / BSY> bit is "0" while in automatic operation and it turns to "1" when the automatic operation is terminated.**

(6) ID-Read

Using the ID-Read command, you can obtain the type and other information on the flash memory contained in the device. The data to be loaded will be different depending on the address [15:14] of the fourth and subsequent bus write cycles (recommended input data is 0x00). After the fourth bus write cycle, when an arbitrary flash memory area is read, the ID value will be loaded. Once the fourth bus write cycle of an ID-Read command has passed, the device will not automatically return to the read mode. In this condition, the set of the fourth bus write cycle and ID-Read commands can be repeatedly executed. For returning to the read mode, use the Read/reset command or hardware reset command.

24.3.1.5   Flash control / status register

Base Address = 0x41FF_F000

Register name		Address (Base+)
Reserved	-	0x0000
Reserved	-	0x0004
Security bit register	FCSECBIT	0x0010
Flash control register	FCFLCS	0x0020
Reserved	-	0x0024
Reserved	-	0x0028

Note: **Do not access to the reserved address.**

(1)   FCFLCS (Flash control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	BLPRO9	BLPRO8
After reset	0	0	0	0	0	0	(Note 2)	(Note 2)
	23	22	21	20	19	18	17	16
bit symbol	BLPRO7	BLPRO6	BLPRO5	BLPRO4	BLPRO3	BLPRO2	BLPRO1	BLPRO0
After reset	(Note 2)	(Note 2)	(Note 2)	(Note 2)	(Note 2)	(Note 2)	(Note 2)	(Note 2)
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	RDY / BSY
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31 to 26	−	R	Read as 0.
25 to 16	BLPRO9 to BLPRO0	R	Protection for Block 9 to 0 0: disabled 1: enabled Each of the protection bits represents the protection status of the corresponding block. When a bit is set to "1", it indicates that the block corresponding to the bit is protected. When the block is protected, data cannot be written to it.
15 to 1	−	R	Read as 0.
0	RDY/BSY	R	Ready / Busy (Note 1) 0:Auto operating 1:Auto operation terminated. Ready/Busy flag bit The RDY/BSY output is provided as a means to monitor the status of automatic operation. This bit is a function bit for the CPU to monitor the function. When the flash memory is in automatic operation, it outputs "0" to indicate that it is busy. When the automatic operation is terminated, it returns to the ready state and outputs "1" to accept the next command. If the automatic operation has failed, this bit maintains the "0" output. By applying a hardware reset, it returns to "1".

Note 1: **This command must be issued in the ready state. Issuing the command in the busy state may disable both correct command transmission and further command input. To exit from the condition, execute system reset. System reset requires at least 0.5 µs regardless of the system clock frequency. In this condition, it takes approx. 2 ms to enable reading after reset.**

Note 2: **The value varies depending on protection applied.**

(2)   FCSECBIT (Security bit register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	SECBIT
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-1	–	R	Read as 0.
0	SECBIT	R/W	Security bits 0:disabled 1:enabled

Note: **This register is initialized by cold reset.**

24.3.1.6   List of Command Sequences

Table 24-15 shows the address and the data of each command of flash memory.

Bus cycles are "bus write cycles" except for the second bus cycle of the Read command, the fourth bus cycle of the Read/reset command, and the fifth bus cycle of the ID-Read command. Bus write cycles are executed by 32-bit (word) data transfer commands. (In the following table, only lower 8 bits data are shown.)

See Table 24-16 for the detail of the address bit configuration. Use a value of "Addr." in the Table 24-15 for the address [15:8] of the normal command in the Table 24-16.

Note: **Always set "0" to the address bits [1:0] in the entire bus cycle.**

Table 24-15 Flash Memory Access from the Internal CPU

Command sequence	First bus cycle	Second bus cycle	Third bus cycle	Fourth bus cycle	Fifth bus cycle	Sixth bus cycle	Seventh bus cycle
	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.
	Data	Data	Data	Data	Data	Data	Data
Read	0xXX	–	–	–	–	–	–
	0xF0	–	–	–	–	–	–
Read / Reset	0x54XX	0xAAXX	0x54XX	RA	–	–	–
	0xAA	0x55	0xF0	RD	–	–	–
ID-Read	0x54XX	0xAAXX	0x54XX	IA	0xXX	–	–
	0xAA	0x55	0x90	0x00	ID	–	–
Automatic page programming	0x54XX	0xAAXX	0x54XX	PA	PA	PA	PA
	0xAA	0x55	0xA0	PD0	PD1	PD2	PD3
Automatic chip erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	–
	0xAA	0x55	0x80	0xAA	0x55	0x10	–
Auto block erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	BA	–
	0xAA	0x55	0x80	0xAA	0x55	0x30	–
Protection bit programming	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x9A	0xAA	0x55	0x9A	0x9A
Protection bit erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x6A	0xAA	0x55	0x6A	0x6A

Supplementary explanation

- RA: Read address
- RD: Read data
- IA: ID address
- ID: ID data
- PA: Program page address
  PD: Program data (32 bit data)

  After fourth bus cycle, enter data in the order of the address for a page.
- BA: Block address
- PBA: Protection bit address

24.3.2 Address bit configuration for bus write cycles

Table 24-16 is used in conjunction with "Table 24-15 Flash Memory Access from the Internal CPU".

Address setting can be performed according to the normal bus write cycle address configuration from the first bus cycle. "0" is recommended" in the Table 24-16 Address Bit Configuration for Bus Write Cycles can be changed as necessary.

Address	Addr [31:20]	Addr [19]	Addr [18]	Addr [17]	Addr [16]	Addr [15]	Addr [14]	Addr [13:11]	Addr [10]	Addr [9]	Addr [8]	Addr [7:0]
Normal commands	Normal bus write cycle address configuration											
	Flash area	"0" is recommended.				Command						Addr[1:0]="0" (fixed) Others:0 (recommended)
ID-READ	IA: ID address (Set the fourth bus write cycle address for ID-Read operation)											
	Flash area	"0" is recommended.			ID address			Addr[1:0]="0" (fixed), Others:0 (recommended)				
Block erase	BA: Block address (Set the sixth bus write cycle address for block erase operation)											
	Block selection (Table 24-16)							Addr[1:0]="0" (fixed), Others:0 (recommended)				
Auto page program-ming	PA: Program page address (Set the fourth bus write cycle address for page programming operation)											
	Page selection											Addr[1:0]="0" (fixed) Others:0 (recommended)
Protection bit pro-gramming	PBA: Protection bit address (Set the seventh bus write cycle address for protection bit programming)											
	Flash area	Fixed to "0".	Protection bit selection (Table 24-17)		Fixed to "0".				Protect bit se-lection (Table 24-17)			Addr[1:0]="0" (fixed) Others:0 (recommended)
Protection bit erase	PBA: Protection bit address (Set the seventh bus erase cycle address for protection bit erasure)											
	Flash area	Fixed to "0".	Protection bit selection (Table 24-18)		Fixed to "0".							Addr[1:0]="0" (fixed) Others:0 (recommended)

As block address, specify any address in the block to be erased.

Table 24-16 Block Address Table

Block	Address (User boot mode)	Address (Single boot mode)	Size (Kbyte)
8	0x0000_0000 to 0x0000_7FFF	0x3F80_0000 to 0x3F80_7FFF	32
9	0x0000_8000 to 0x0000_FFFF	0x3F80_8000 to 0x3F80_FFFF	32
7	0x0001_0000 to 0x0001_FFFF	0x3F81_0000 to 0x3F81_FFFF	64
6	0x0002_0000 to 0x0003_FFFF	0x3F82_0000 to 0x3F83_FFFF	128
5	0x0004_0000 to 0x0005_FFFF	0x3F84_0000 to 0x3F85_FFFF	128
4	0x0006_0000 to 0x0007_FFFF	0x3F86_0000 to 0x3F87_FFFF	128
3	0x0008_0000 to 0x0009_FFFF	0x3F88_0000 to 0x3F89_FFFF	128
2	0x000A_0000 to 0x000B_FFFF	0x3F8A_0000 to 0x3F8B_FFFF	128
1	0x000C_0000 to 0x000D_FFFF	0x3F8C_0000 to 0x3F8D_FFFF	128
0	0x000E_0000 to 0x000F_FFFF	0x3F8E_0000 to 0x3F8F_FFFF	128

Note:**As for the addresses from the first to the fifth bus cycles, specify the upper addresses of the blocks to be erased.**

Table 24-17 Protection Bit Programming Address Table

Block	Protection bit	The seventh bus write cycle address				
		Address [18]	Address [17]	Address [16:11]	Address [10]	Address [9]
Block0	<BLPRO[0]>	0	0		0	0
Block1	<BLPRO[1]>	0	0		0	1
Block2	<BLPRO[2]>	0	0		1	0
Block3	<BLPRO[3]>	0	0		1	1
Block4	<BLPRO[4]>	0	1	Fixed to "0".	0	0
Block5	<BLPRO[5]>	0	1		0	1
Block6	<BLPRO[6]>	0	1		1	0
Block7	<BLPRO[7]>	0	1		1	1
Block9	<BLPRO[9]>	1	0		0	1
Block8	<BLPRO[8]>	1	0		0	0

Table 24-18 Protection Bit Erase Address Table

Block	Protection bit	The seventh bus write cycle address [18:17]	
		Address [18]	Address [17]
Block0 to 3	<BLPRO[0:3]>	0	0
Block4 to 7	<BLPRO[4:7]>	0	1
Block8 to 9	<BLPRO[8:9]>	1	0

Note: **The protection bit erase command cannot erase by individual block.**

Table 24-19 The ID-Read command's fourth bus write cycle ID address (IA) and the data to be read by the following 32-bit data transfer command (ID)

IA[15:14]	ID[7:0]	Code
0y00	0x98	Manufacturer code
0y01	0x5A	Device code
0y10	Reserved	–
0y11	0x10	Macro code

24.3.2.1 Flowchart

```
                                                    ┌──────────┐
                                                    │  Start   │
                                                    └────┬─────┘
                                                         │
                                                         ▼
                        ┌───────────────────────────────────────────────────┐
              ┌────────▶│  Automatic page programming command suquence       │
              │         │  (see the flowchart shown below)                   │
              │         └───────────────────────┬───────────────────────────┘
              │                                 │
              │                                 ▼
  ┌───────────┴─────────────────┐      ╱───────────────────╲
  │  Address = Address + 0x200   │  NO ╱  The address of the  ╲
  │  (set by a page)             │◀────   last page ?          
  └──────────────────────────────┘      ╲───────────────────╱
                                                 │
                                                 │ YES
                                                 ▼
                                     ┌─────────────────────────────┐
                                     │  Automatic page programming  │
                                     └─────────────────────────────┘
```

Automatic Page Programming Command Sequence (Address / Command)

```
          ┌───────────────────────────────────────┐
          │             0x54xx/0xAA                │
          └───────────────────┬───────────────────┘
                              ▼
          ┌───────────────────────────────────────┐
          │             0xAAxx/0x55                │
          └───────────────────┬───────────────────┘
                              ▼
          ┌───────────────────────────────────────┐
          │             0x54xx/0xA0                │
          └───────────────────┬───────────────────┘
                              ▼
          ┌───────────────────────────────────────┐
          │  Programming address (page address)/   │
          │  Programming data (32 bit data)        │
          └───────────────────────────────────────┘
```
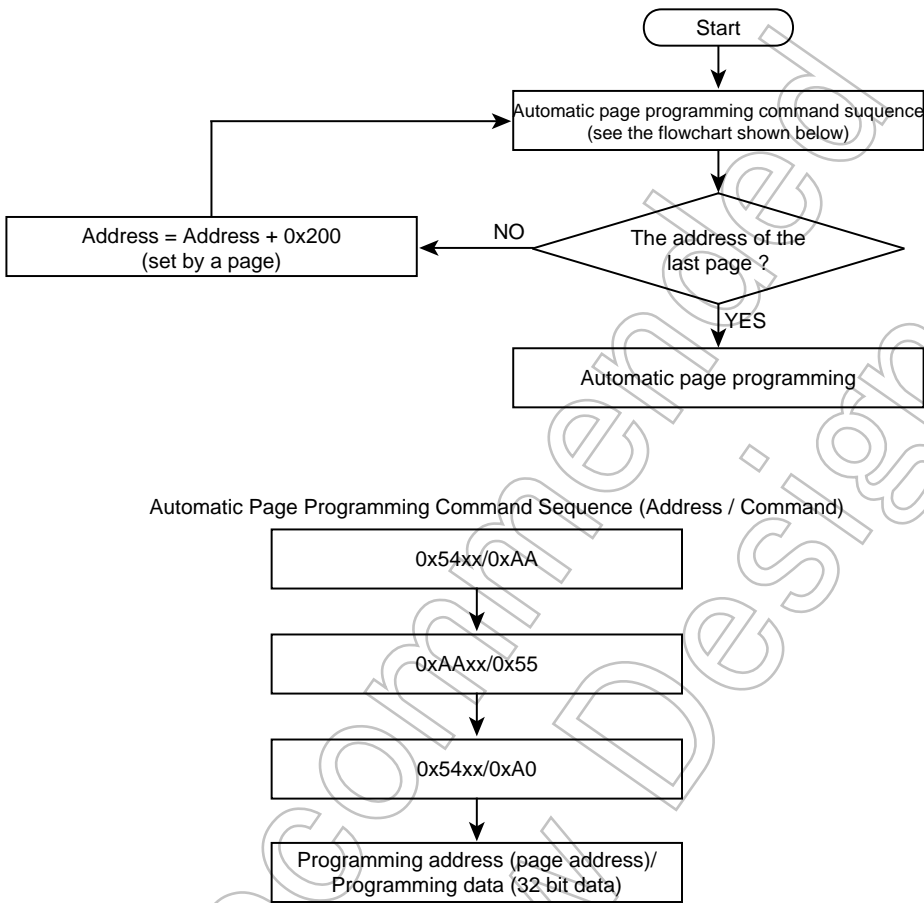
Figure 24-10 Automatic Programming

Note: **Command sequence is executed by 0x54xx or 0x55xx.**
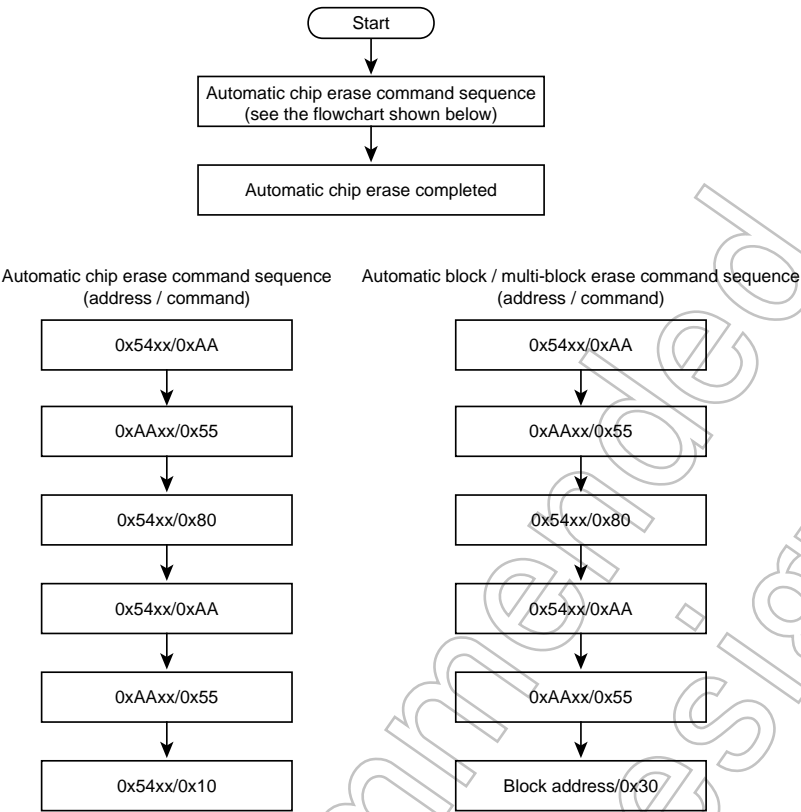
Figure 24-11 Automatic Erase

Note: **Command sequence is executed by 0x54xx or 0x55xx.**

25. ROM protection

25.1 Outline

The TMPM363F10FG offers two kinds of ROM protection/ security functions.

One is a write/ erase-protection function for the internal flash ROM data.

The other is a security function that restricts internal flash ROM data readout and debugging.

25.2 Features

25.2.1 Write/ erase-protection function

The write/ erase-protection function enables the internal flash to prohibit the writing and erasing operation for each block.

To activate the function, write "1" to the corresponding bits to a block to protect. Writing "0" to the bits cancels the protection.

The protection settings of the bits can be monitored by the FCFLCS <BLPRO[9:0]> bit. See the chapter "Flash" for programming details.

25.2.2 Security function

The security function restricts flash ROM data readout and debugging.

This function is available under the conditions shown below.

1. The FCSECBIT <SECBIT> bit is set to"1".

2. All the protection bits (the FCFLCS<BLPRO> bits) used for the write/erase-protection function are set to "1".

Note:**The FCSECBIT <SECBIT> bit is set to "1" at a power-on reset right after power-on.**

Table 25-1 shows details of the restrictions by the security function.

Table 25-1 Restrictions by the security function

Item	Details
1) ROM data readout	Data can be read from CPU.
2) Debug port	Communication of SW and trace are prohibited
3) Command for flash memory	Writing a command to the flash memory is prohibited. An attempt to erase the contents in the bits used for the write/ erase-protection erases all the protection bits.

25.3 Register

Base Address = 0x41FF_F000

Register name		Address(Base+)
Reserved	-	0x0000,0x0004
Security bit register	FCSECBIT	0x0010
Flash control register	FCFLCS	0x0020
Reserved	-	0x0024,0x0028

Note: **Access to the "Reserved" area is prohibited.**

25.3.1 FCFLCS (Flash control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	BLPRO9	BLPRO8
After reset	0	0	0	0	0	0	(Note2)	(Note2)
	23	22	21	20	19	18	17	16
bit symbol	BLPRO7	BLPRO6	BLPRO5	BLPRO4	BLPRO3	BLPRO2	BLPRO1	BLPRO0
After reset	(Note2)	(Note2)	(Note2)	(Note2)	(Note2)	(Note2)	(Note2)	(Note2)
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	RDY/BSY
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-26	–	R	Read as 0.
25-16	BLPRO9 to BLPRO0	R	Protection for Block9 to 0 0: disabled 1: enabled Protection status bits Each of the protection bits represents the protection status of the corresponding block. When a bit is set to "1," it indicates that the block corresponding to the bit is protected. When the block is protected, data cannot be written to it.
15-1	–	R	Read as 0.
0	RDY/BSY	R	Ready/Busy (Note 1) 0: Auto operating 1:Auto operation terminated Ready/Busy flag bit The RDY/BSY output is provided as a means to monitor the status of automatic operation. This bit is a function bit for the CPU to monitor the function. When the flash memory is in automatic operation, it outputs "0" to indicate that it is busy. When the automatic operation is terminated, it returns to the ready state and outputs "1" to accept the next command. If the automatic operation has failed, this bit maintains the "0" output. By applying a hardware reset, it returns to "1."

Note 1: **This command must be issued in the ready state. Issuing the command in the busy state may disable both correct command transmission and further command input. To exit from the condition, execute system reset. System reset requires at least 0.5 ms regardless of the system clock frequency. In this condition, it takes approx. 2 ms to enable reading after reset.**

Note 2: **The value varies depending on protection applied.**

25.3.2 FCSECBIT(Security bit register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	SECBIT
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-1	–	R	Read as 0.
0	SECBIT	R/W	Security bit 0: Disabled 1: Enabled

Note: This register is initialized by cold reset.

25.4　Writing and erasing

Writing and erasing protection bits are available with a single chip mode, single boot mode and writer mode.

25.4.1　Protection bits

Writing to the protection bits is done on block-by-block basis.

When the settings for all the blocks are "1", erasing must be done after setting the FCSECBIT <SECBIT> bit to "0". Setting "1" at that situation erases all the protection bits. To write and erase the protection bits, command sequence is used.

See the capter "Flash" for details

25.4.2　Security bit

The FCSECBIT <SECBIT> bit that activates security function is set to "1" at a power-on reset right after power-on.

The bit is rewritten by the following procedure.

1. Write the code 0xa74a9d23 to FCSECBIT register.

2. Write data within 16 clocks from the above.1.

Note: The above procedure is enabled only when using 32-bit data transfer command.

TOSHIBA

26. RAM Interface

After releasing reset, then wait time of RAM (from 0x2000_4000 to 0x2000_BFFF) is set one wait. It is possible to use zero wait.

26.1 Register List

The control register and it's address are shwon as below.

Base Address = 0x41FF_F058

Register name		Address(Base+)
RAM Interface register	RCWAIT	0x0000

26.1.1 RCWAIT(RAM Interface Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	RAM1WAIT
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	ā@î\
31-1	–	R	Read as "0"
0	RAM1WAIT	R/W	Specify RAMWAIT 0 : 0WAIT 1 : 1WAIT

TOSHIBA

27. Port Section Equivalent Circuit Schematic

Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

The input protection resistance ranges from several tens of Ω to several hundreds of Ω. Damping resistors X2 and XT2 are shown with a typical value.

27.1   PA0 to 7, PB0 to 7, PP1, PP3 to 5
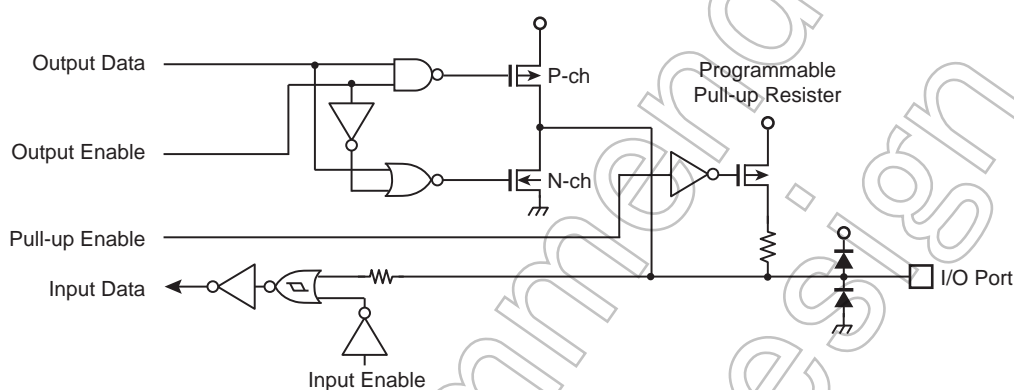
27.2   PE0 to 7, PF0 to 4, PG0 to 7, PI0, PL0 to 7, PM0 to 7, PN0 to 3, PP0, PP2, PP6

2023/07/21
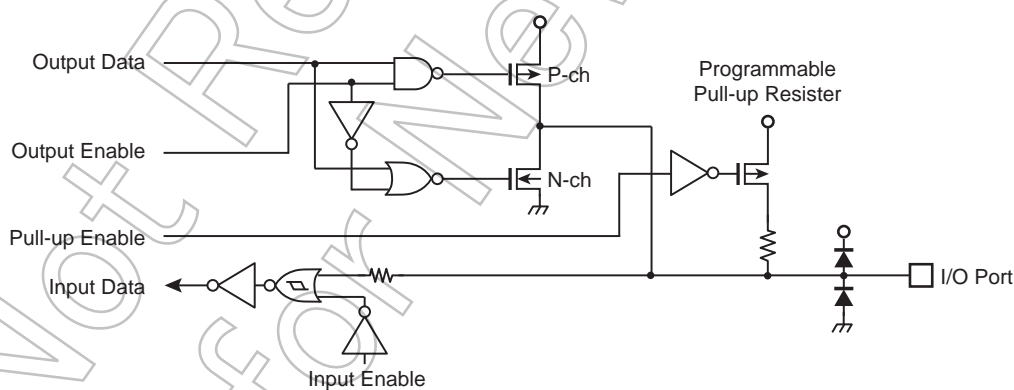
27.3   PI1
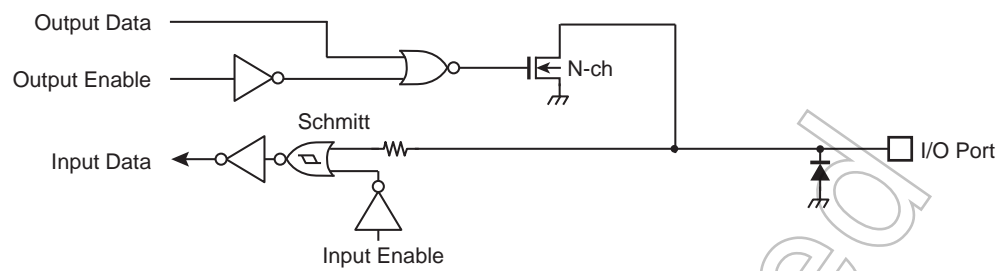
27.4   PJ0 to 7
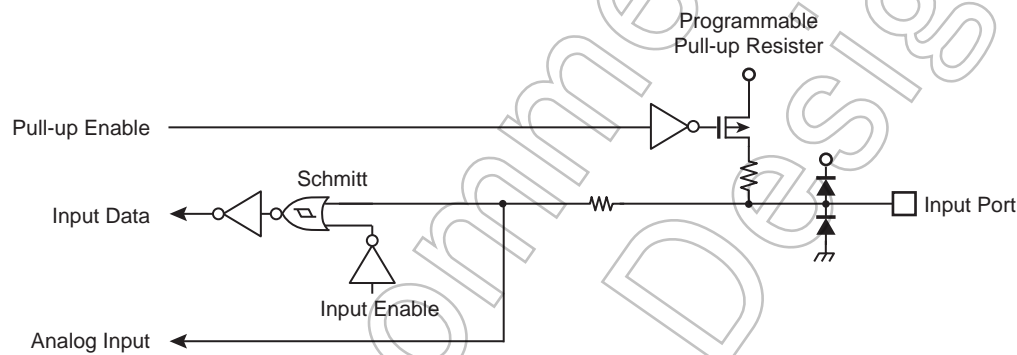
27.5 $\overline{\text{RESET}}$, $\overline{\text{NMI}}$
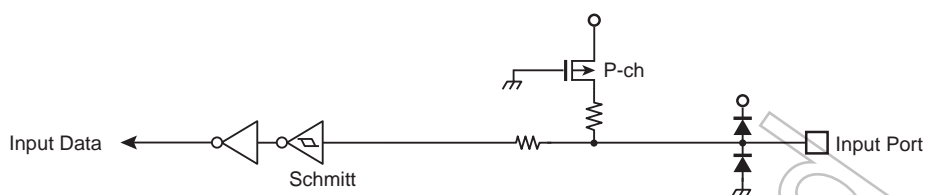
27.6 MODE, SWCLK
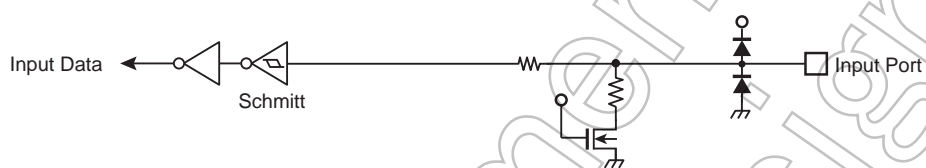
27.7 SWDIO
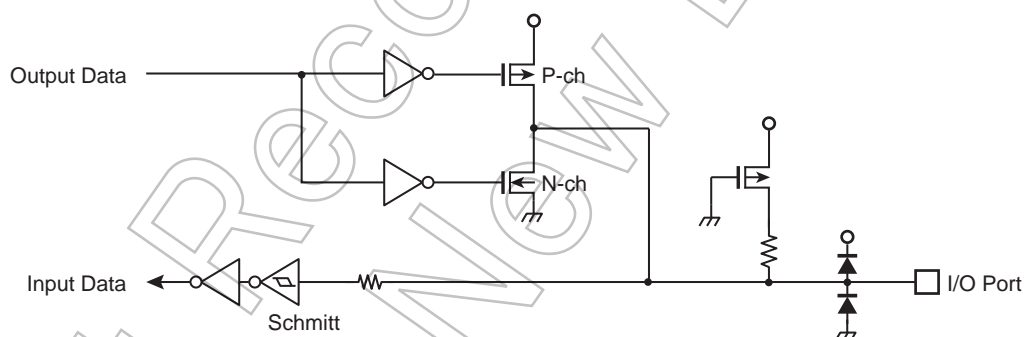
27.8　X1, X2
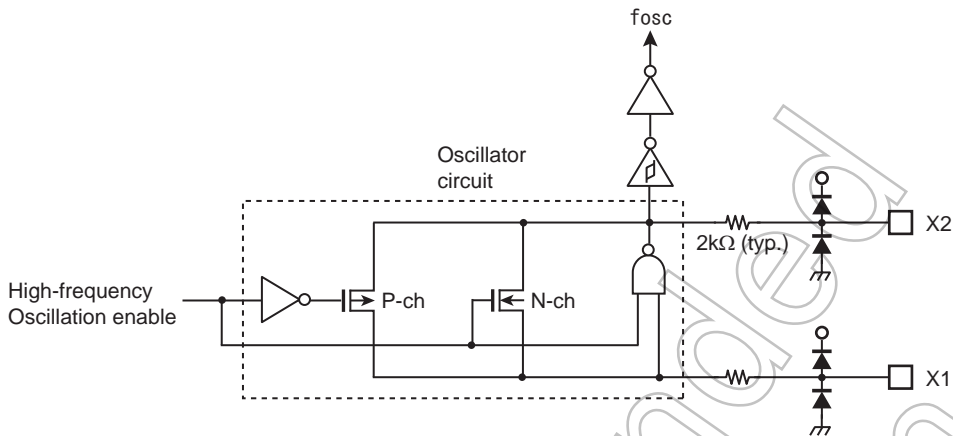
27.9　XT1, XT2
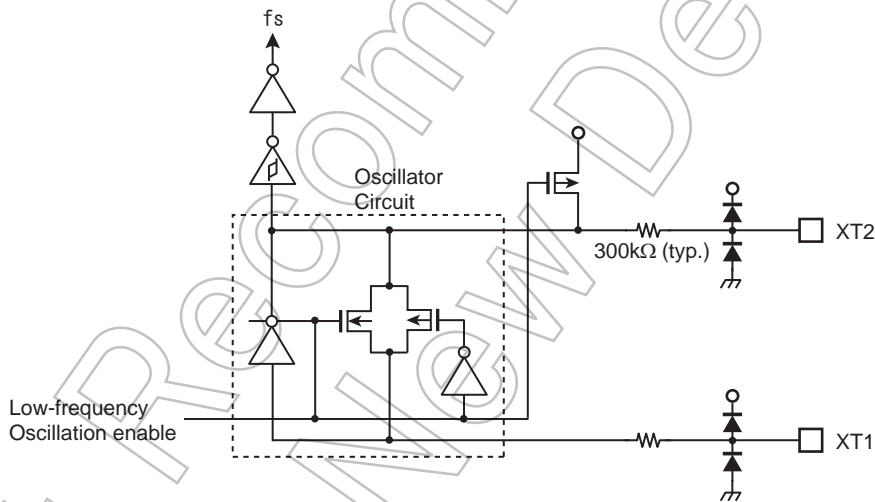
27.10　VREFH, AVSS
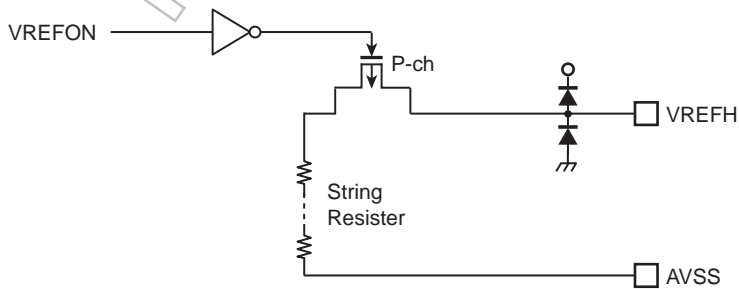
28. Electrical Characteristics

28.1 Absolute Maximum Ratings

Parameter		Symbol	Rating	Unit
Supply voltage		DVDD3 (Note2)	−0.3 to 3.9	V
		AVDD3	−0.3 to 3.9	
		RVDD3	−0.3 to 3.9	
Input voltage		V_{IN}	−0.3 to DVDD3 + 0.3	V
Low-level output current	Per pin	I_{OL}	5	mA
	Total	ΣI_{OL}	50	
High-level output current	Per pin	I_{OH}	−5	
	Total	ΣI_{OH}	−50	
Power consumption (Ta = 85 °C)		PD	600	mW
Soldering temperature (10 s)		T_{SOLDER}	260	°C
Storage temperature		T_{STG}	−40 to 125	°C
Operating Temperature	Except during Flash W/E	T_{OPR}	−40 to 85	°C
	During Flash W/E		0 to 70	

Note 1: **Absolute maximum ratings are limiting values of operating and environmental conditions which should not be exceeded under the worst possible conditions. The equipment manufacturer should design so that no Absolute maximum rating value is exceeded with respect to current, voltage, power consumption, temperature, etc. Exposure to conditions beyond those listed above may cause permanent damage to the device or affect device reliability, which could increase potential risks of personal injury due to IC blowup and/or burning.**

Note 2: DVDD3 = DVDD3A = DVDD3B

28.2 DC Electrical Characteristics (1/3)

Ta = −40 to 85 °C

Parameter		Symbol	Condition		Min.	Typ. (Note1)	Max.	Unit
Supply voltage	DVDD3A = AVDD3=DVDD3B= RVDD3 (Note3) DVSS = AVSS = 0V	DVDD3A AVDD3 DVDD3B RVDD3	f_{OSC} = 8 to 16.0 MHz fsys = 1to 64 MHz fs = 30 to 34 kHz		2.7	–	3.6	V
Low-level Input voltage	PJ0 to 7 (Note2)	V_{IL1}	2.7 V ≤ AVDD3 ≤ 3.6 V		−0.3	–	0.25 AVDD3	V
	PA0 to 7, PB0 to 7, PP1, PP3 to 5	V_{IL2}	2.7 V ≤ DVDD3 ≤ 3.6 V				0.3 DVDD3	
	PE0 to 7, PF0 to 4, PG0 to 7, PI0 to 1, PL0 to 7, PM0 to 7, PN0 to 3, PP0/2/6 RESET, NMI, MODE, SWDIO, SWCLK	V_{IL3}					0.25 DVDD3	
	X1	V_{IL4}	2.7 V ≤ DVDD3A ≤ 3.6 V				0.1 DVDD3A	
HIgh-level Input voltage	PJ0 to 7 (Note2)	V_{IH1}	2.7 V ≤ AVDD3 ≤ 3.6 V		0.75 AVDD3	–	AVDD3 + 0.3	V
	PA0 to 7, PB0 to 7, PP1, PP3 to 5	V_{IH2}	2.7 V ≤ DVDD3 ≤ 3.6 V		0.7 DVDD3		DVDD3 + 0.3	
	PE0 to 7, PF0 to 4, PG0 to 7, PI0 to 1, PL0 to 7, PM0 to 7, PN0 to 3, PP0/2/6 RESET, NMI, MODE, SWDIO, SWCLK	V_{IH3}			0.75 DVDD3			
	X1	V_{IH4}	2.7 V ≤ DVDD3A ≤ 3.6 V		0.9 DVDD3A		DVDD3A + 0.3	
Low-level output voltage	Except below	V_{OL1}	I_{OL} = 2 mA	DVDD3 ≥ 2.7 V	–	–	0.4	V
	PL0/1/4/5, PG0/1/4/5	V_{OL2}	I_{OL} = 3 mA		–	–		
High-level output voltage		V_{OH}	I_{OH} = −2 mA	DVDD3 ≥ 2.7 V	2.4	–	–	V
Input leakage current		I_{LI}	0.0 ≤ V_{IN} ≤ DVDD3 0.0 ≤ V_{IN} ≤ AVDD3		–	0.02	±5	μA
Output leakage current		I_{LO}	0.2 ≤ V_{IN} ≤ DVDD3 − 0.2 0.2 ≤ V_{IN} ≤ AVDD3 − 0.2		–	0.05	±10	
Pull-up resister at RESET		RRST	DVDD3 = 2.7 V to 3.6 V		–	50	150	kΩ
Hysteresis voltage		VTH	2.7 V ≤ DVDD3 ≤ 3.6 V		0.3	0.6	–	V
Programmable pull-up / pull-down resister		PKH	DVDD3 = 2.7 V to 3.6 V		–	50	150	kΩ
Pin capacitance (Except power supply pins)		C_{IO}	fc = 1 MHz		–	–	10	pF

Note 1: Ta = 25 °C, DVDD3 = RVDD3 = AVDD3 = 3.3 V, unless otherwise noted.
Note 2: **When PJ port is used for input port.**
Note 3: **The same voltage must be supplied to DVDD3A, DVDD3B, AVDD3, RVDD3.**

28.3 DC Electrical Characteristics (2/3)

DVDD3A = DVDD3B = AVDD3 = RVDD3 = 2.7 V to 3.6 V, Ta =−40 to 85 °C

Parameter	Symbol	Condition	Min.	Typ. (Note1)	Max.	Unit
Low-level output current	I_{OL1}	Except below (per pin)	–	–	2	mA
	I_{OL2}	PL0/1/4/5,PG0/1/4/5 (per pin)	–	–	3	mA
	ΣI_{OL1}	Per group (PORT L/I)	–	–	20	mA
	ΣI_{OL2}	Per group (PORT M/N/P)	–	–	27	mA
	ΣI_{OL3}	Per group (PORT A/B/E)	–	–	27	mA
	ΣI_{OL4}	Per group (PORT F/G)	–	–	27	mA
	ΣI_{OL5}	Total (All ports)	–	–	35	mA
High-level output current	I_{OH}	Per pin	–	–	−2	mA
	ΣI_{OH1}	Per group (PORT I/L/M/N/P)	–	–	−13	mA
	ΣI_{OH2}	Per group (PORT A/B//E)	–	–	−13	mA
	ΣI_{OH3}	Per group (PORT F/G)	–	–	−13	mA
	ΣI_{OH4}	Total (All ports)	–	–	−35	mA

Note 1: Ta = 25 °C, DVDD3 = RVDD3 = AVDD3 = 3.3 V, unless otherwise noted.

Note 2: **The same voltage must be supplied to DVDD3, DVDD3A, DVDD3B, AVDD3, RVDD3.**

Note 3: **High-level output current (ΣI_{OH}) is total per power supply pin.**

28.4   DC Electrical Characteristics (3/3)

DVDD3A = DVDD3B = AVDD3 = RVDD3 = 2.7 V ~ 3.6 V, Ta =−40 to 85 °C

Parameter	Symbol	Condition	Min.	Typ. (Note1)	Max.	Unit
NORMAL (Note2) Gear 1/1	I_{DD}	fsys = 64 MHz (f_{OSC} = 8 MHz)	–	85	95	mA
IDLE2 (Note3)			–	40	48	
IDLE1 (Note4)		fsys = 1 MHz (f_{OSC} = 8 MHz, PLL= OFF,CG = 1/8)	–	1.3	5	
SLOW		fs = 32.768 kHz	–	1	6	
SLEEP(Note5)			–	260	2450	µA
STOP		–	–	250	2400	
BACKUP SLEEP (Note6)		fs=32.768 kHz	–	35	210	
BACKUP STOP (Note7)		–	–	25	200	

Note 1: Ta = 25 °C, DVDD3 = RVDD3 = AVDD3 = 3.3 V, unless otherwise noted.

Note 2: I_{DD} NORMAL: Measured with:

   Operation program : The dhrystone ver. 2.1 operated in Flash.

   Operation peripheral : All function operates excluding ADC.

Note 3: **I_{DD} IDLE2: Measured with:**

   Operation peripheral : All peripherals are operated.

Note 4: **I_{DD} IDLE1: Measured with:**

   Operation peripheral : Some peripherals are operated.

Note 5: **I_{DD} SLEEP: Measured with:**

   Operation peripheral : CEC, RMC and RTC operated.

Note 6: **I_{DD} BACKUP SLEEP: Measured with:**

   Operation peripheral : CEC, RMC and RTC operated, keep contents of BACKUP RAM,

   other peripherals are shut down power supply.

Note 7: **I_{DD} BACKUP STOP: Measured with:**

   Operation peripheral : Keep contents of BACKUP RAM,

   other peripherals are shut down power supply.

28.5 10-bit ADC Electrical Characteristics

DVDD3A = DVDD3B = AVDD3 = RVDD3 = 2.7 V to 3.6 V

AVSS = DVSS, Ta = −40 to 85 °C

Parameter		Symbol	Condition	Min	Typ.	Max.	Unit
Analog reference voltage(+)		VREFH	−	2.7	3.3	3.6	V
Analog input voltage		VAIN	−	AVSS	−	VREFH	V
Power supply current of analog reference voltage	AD conversion	IREF	DVSS = AVSS	−	2.5	5.5	mA
	Non-AD conversion			−	0.02	5	µA
Supply current	AD conversion	−	Except IREF	−	−	3	mA
INL error		−	AIN resistance ≤ 1.3 kΩ AIN load capacitance ≥ 0.1 Fµ Conversion time ≥ 1.5 µs	−	±2	±3	LSB
DNL error				−	±1	±2	
Offset error				−	±2	±4	
Full-scale error				−	±2	±4	

Note 1: 1LSB = (VREFH − AVSS) / 1024 [V]

Note 2: **Peripheral functions are disabled.**

28.6 AC Electrical Characteristics

28.6.1 AC measurement condition

The AC characteristics data of this chapter is measured under the following conditions unless otherwise noted.

- Output levels : High = 0.8 x DVDD3
- Output levels : Low = 0.2 x DVDD3
- Input levels : Refer to low-level input voltage and high-level input voltage in "DC Electrical Characteristcs".
- Load capacity : CL = 30pF

Note:**" Equation " condition is DVDD3 =2.7V to 3.6 V.**

28.6.2 Static memory controller (SMC)

"T" is 1/2 cycles of an internal bus frequency (fsys) in the Equation of the table.

AC measurement condition

- Output levels : High = 0.7 x DVDD3, Low = 0.3 x DVDD3
- Input levels : High = 0.7 x DVDD3, Low = 0.3 x DVDD3
- Load capacity : CL = 40pF

28.6.2.1 Basic Bus cycle (Read)

| Parameter | Symbol | Equation | | fsys = 64 MHz T=31.25 N = 4 M = 1 K = 5 L = 2 P = 2 Q = 2 | Unit |
		Min.	Max.		
SMCCLK	t_{CYC}	31.25	2000	31.3	
A1 to A23 Valid → D0 to D15 Input (Multiplex bus mode)	t_{ADL}	−	$(N)T - 35.0$	90.0	
$\overline{OE}$ falling edge → D0 to D15 Input	t_{OED}	−	$(N - M)T - 25.0$	68.8	
$\overline{OE}$ Low-level pulse width	t_{OEW}	$(N - M)T - 13.0$	−	80.8	
A1 ~ A16 Valid → $\overline{OE}$ falling edge (Multiplex bus mode)	t_{AOEL}	$(M)T - 15.0$	−	16.3	ns
$\overline{OE}$ rising edge → D0 to D15 Hold	t_{HR}	0.00	−	0.00	
A1 to A23 Valid → D0 to D15 Hold	t_{HA}	0.00	−	0.00	
$\overline{OE}$ High-level pulse width	t_{OEHW}	$(M)T - 13.0$	−	18.3	
$\overline{ALE}$ Low-level pulse width	t_{LL}	$T - 13.0$	−	18.3	
A1 to A16 Valid → $\overline{ALE}$ rising edge	t_{AL}	$T - 15.0$	−	16.3	
$\overline{ALE}$ rising edge → A1to A16 Hold	t_{LA}	$T - 10.0$	−	21.3	
$\overline{OE}$ rising edge → $\overline{ALE}$ falling edge	t_{CLR}	$(P)T - 13.0$	−	49.5	
$\overline{OE}$ rising edge → A1to A16 Hold	t_{CAR}	$(P)T - 13.0$	−	49.5	
$\overline{OE}$ rising edge → A1to A16 Output	t_{RAE}	$(P)T - 13.0$	−	49.5	

Note:" **Equation Measurement condition:**

N	=	t_{RC} Cycle	≥ 3	,	M	=	t_{CEOE} Cycle	≥ 1
K	=	t_{WC} Cycle	≥ 3	,	L	=	t_{WP} Cycle	≥ 1
P	=	t_{TR} Cycle	≥ 1	,	Q	=	t_{PC} Cycle	≥ 1

28.6.2.2   BASIC Bus Cycle (Write)

Parameter	Symbol	Equation		fsys = 64 MHz N = 4 M = 1 K = 5 L = 2 P = 2 Q = 2	Unit
		Min.	Max.		
D0 to D15 Valid → $\overline{WE}$ rising edge (Multiplex bus mode)	t_{DW1}	(L)T − 23.0	−	39.5	ns
$\overline{WE}$ Low-level pulse width (Multiplex bus mode)	t_{WW1}	(L + 1)T − 13.0	−	80.8	
A1 to A23 Valid → $\overline{WE}$ falling edge	t_{AW}	T− 15.0	−	16.3	
$\overline{WE}$ rising edge → A1 to A23 Hold	t_{WA}	(K − L)T − 13.0	−	80.8	
$\overline{WE}$ rising edge → D0 to D15 Hold (Multiplex bus mode)	t_{WD1}	(K − L − 2)T − 10.0	−	21.3	
$\overline{WE}$ rising edge → A1to A16 Hold	t_{CLW}	(K − L − 2 + P)T − 13.0	−	80.8	
$\overline{WE}$ rising edge → $\overline{ALE}$ falling edge	t_{CAW}	(K − L − 2 + P)T − 13.0	−	80.8	

Note:" **Equation Measurement condition:**

N	=	t_{RC} Cycle	≥ 3	,	M	=	t_{CEOE} Cycle	≥ 1
K	=	t_{WC} Cycle	≥ 3	,	L	=	t_{WP} Cycle	≥ 1
P	=	t_{TR} Cycle	≥ 1	,	Q	=	t_{PC} Cycle	≥ 1

28.6.2.3   Example of Read / Write cycle

When accessing to read from the external bus, dummy read cycle besides a valid one is occurred. Therefore, the external memory can not be used as FIFO.

The bollowing operation chart shows only valid read cycle except the dummy cycle.

(1)   Memory Read Cycle (Multiplex Bus) (t_{RC}=7, t_{CEOE}=3)
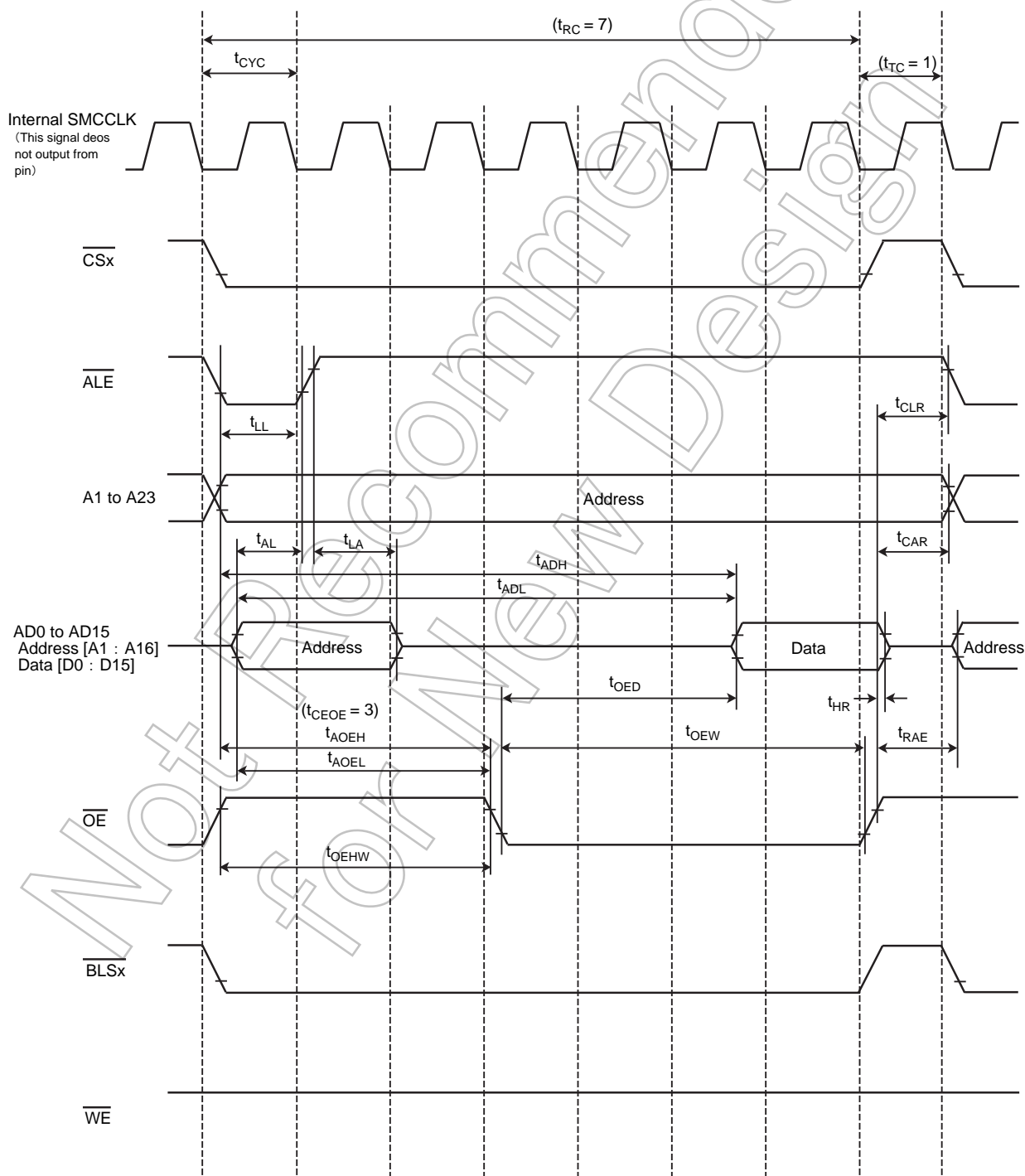
(2)   Memory Write Cycle (Multiplex Bus) (t_{WC}=7, t_{WP}=4)
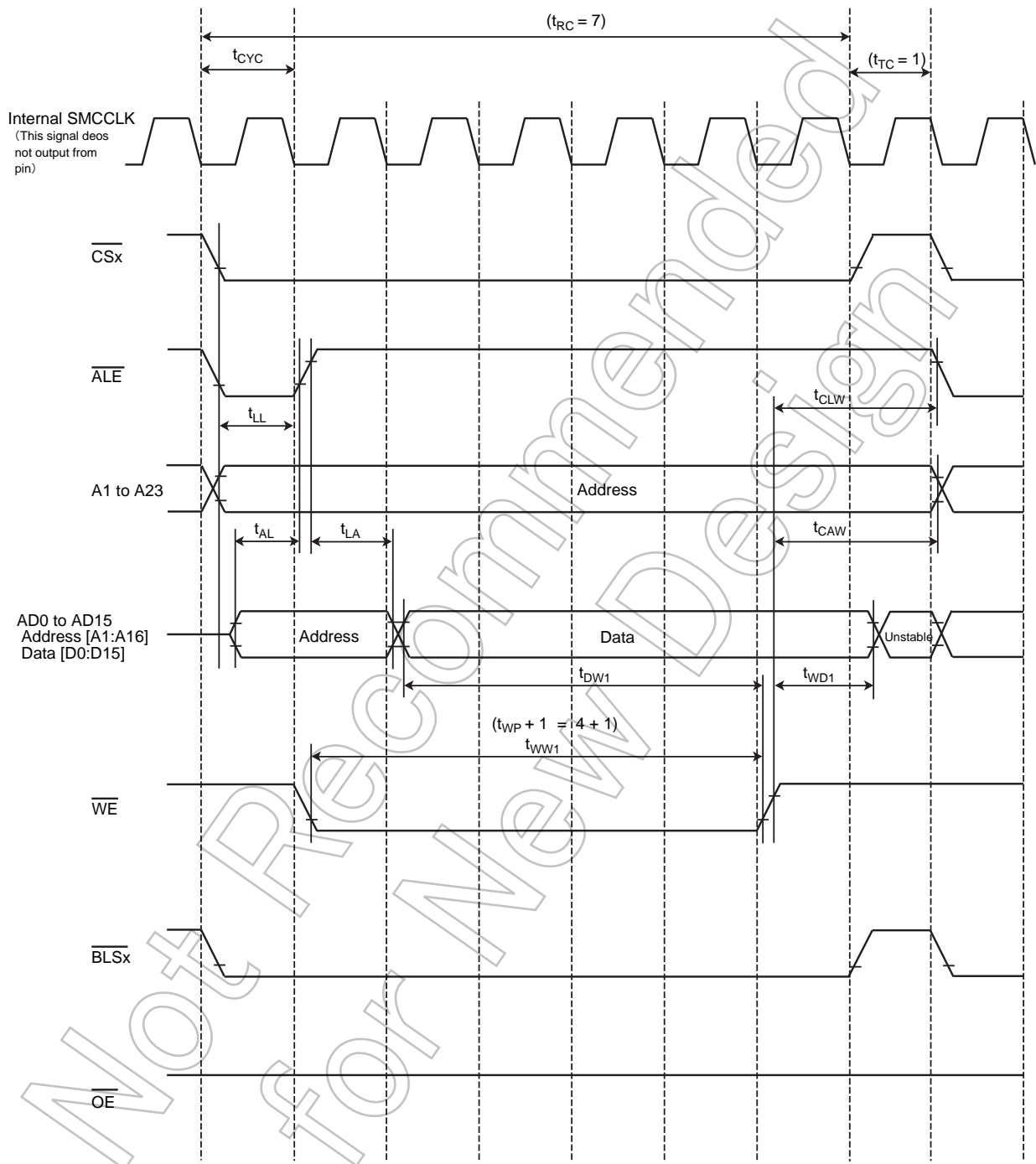
28.6.3   Serial Interface (SIO/UART)

28.6.3.1   I/O Interface mode

In the table below, the letter x represents the SIO operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

(1)   SCLK Input mode

[Data input]

Parameter	Symbol	Equation		fsys = 64 MHz		Unit
		Min.	Max.	Min.	Max.	
SCLK Clock High width (input)	t_{SCH}	4x	–	62.5	–	ns
SCLK Clock Low width (input)	t_{SCL}	4x	–	62.5	–	
SCLK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	–	125	–	
Valid Data input → SCLK rise / fall (Note1)	t_{SRD}	30	–	30.0	–	
SCLK rise / fall (Note1) → Input Data hold	t_{HSR}	x + 30	–	45.6	–	

[Data output]

Parameter	Symbol	Equation		fsys = 64 MHz		Unit
		Min.	Max.	Min.	Max.	
SCLK Clock High width (input)	t_{SCH}	4x	–	91.9 (Note3)	–	ns
SCLK Clock Low width (input)	t_{SCL}	4x	–	91.9 (Note3)	–	
SCLK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	–	184	–	
Output Data → SCLK rise / fall (Note1)	t_{OSS}	$t_{SCY}/2 - 3x - 45$	–	0.00 (Note2)	–	
SCLK rise / fall (Note1) → Output Data hold	t_{OHS}	$t_{SCY}/2$	–	91.9	–	

Note 1: SCLK rise/fall : SCLK rise mode uses the rise timing of SCLK. SCLK fall mode uses the fall timing of SCLK.

Note 2: **Use the frequency of SCLK in a range where the calculation value keeps positive.**

Note 3: **The value indicates a minimum value that enables t_{OSS} to be zero or more.**

(2)   SCLK output mode

Parameter	Symbol	Equation		fsys = 64 MHz		Unit
		Min.	Max.	Min.	Max.	
SCLK cycle (programmable)	t_{SCY}	4x	–	62.5	–	ns
Output Data → SCLK rise	t_{OSS}	$t_{SCY}/2 - 20$	–	11.3	–	
SCLK rise → Output Data hold	t_{OHS}	$t_{SCY}/2 - 20$	–	11.3	–	
Valid Data input → SCLK rise	t_{SRD}	x + 45	–	45	–	
SCLK rise → Input Data hold	t_{HSR}	0	–	0	–	
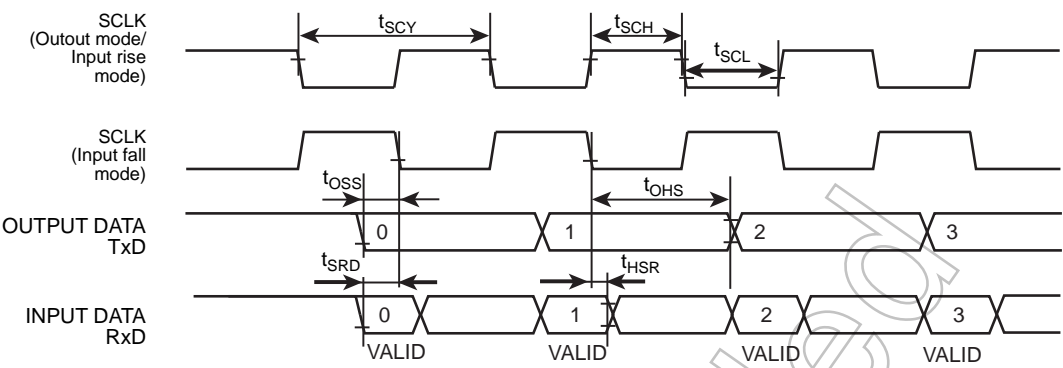
28.6.4   Serial Bus Interface (I2C/SIO)

28.6.4.1   I2C Mode

In the table below, the letter x represents the I2C operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

n denotes the value of n programmed into the SCK (SCL output frequency select) field in the SBIxCR.

Parameter	Symbol	Equation		Standard Mode		Fast Mode		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
SCL clock frequency	t_{SCL}	0	–	0	100	0	400	kHz
Hold time for a START condition	$t_{HD;STA}$	–	–	4.0	–	0.6	–	µs
SCL Low width (input) (Note1)	t_{LOW}	–	–	4.7	–	1.3	–	µs
SCL high width (input) (Note2)	t_{HIGH}	–	–	4.0	–	0.6	–	µs
Setup time for a repeated START condition	$t_{SU;STA}$	(Note5)	–	4.7	–	0.6	–	µs
Data hold time (input) (Note3, 4)	$t_{HD;DAT}$	–	–	0.0	–	0.0	–	µs
Data setup time	$t_{SU;DAT}$	–	–	250	–	100	–	ns
Setup time for a STOP condition	$t_{SU;STO}$	–	–	4.0	–	0.6	–	µs
Bus free time between STOP condition and START condition	t_{BUF}	(Note5)	–	4.7	–	1.3	–	µs

Note 1: SCL clock Low width (output) : $(2^{n-1} + 58)/x$

Note 2: SCL clock High width (output) : $(2^{n-1} + 14)/x$

On I2C-bus specification, Maximum Speed of Standard Mode is 100kHz, Fast mode is 400khz. Internal SCL frequency setting should comply with Note1 & Note2 shown above.

Note 3: **The output data hold time is equal to 4x of internal SCL.**

Note 4: **The Philips I2C-bus specification states that a device must internally provide a hold time of at least 300 ns for the SDA signal to bridge the undefined region of the falling edge of SCL. However, this SBI does not satisfy this requirement. Also, the output buffer for SCL does not incorporate slope control of the falling edges; therefore, the equipment manufacturer should design so that the input data hold time shown in the table is satisfied, including tr/tf of the SCL and SDA lines.**

Note 5: **Software dependent**

Note 6: The Philips I2C-bus specification instructs that if the power supply to a Fast-mode device is switched off, the SDA and SCL I/O pins must be floating so that they don't obstruct the bus lines. However, this SBI does not satisfy this requirement.
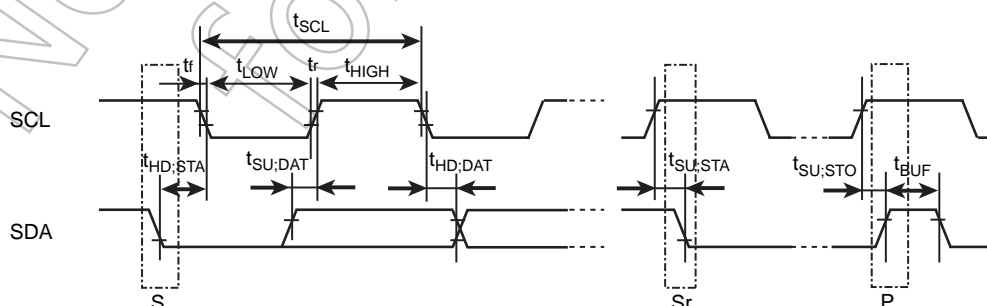
S: START condition
Sr: RESTART condition
P: STOP condition

28.6.4.2   Clock-Synchronous 8-Bit SIO mode

In the table below, the letter x represents the SBI operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

(1)   SCK Input mode(The electrical specifications below are for an SCK signal with a 50% duty cycle.)

[Data input]

Parameter	Symbol	Equation		fsys = 64 MHz		Unit
		Min.	Max.	Min.	Max.	
SCK Clock High width (input)	t_{SCH}	4x	–	62.5	–	
SCK Clock Low width (input)	t_{SCL}	4x	–	62.5	–	
SCK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	–	125	–	ns
SCK rise→ Valid Data input	t_{SRD}	30 − x	–	14.4	–	
SCK rise → Input Data hold	t_{HSR}	2x + 30	–	61.3	–	

[Data output]

Parameter	Symbol	Equation		fsys = 64 MHz		Unit
		Min.	Max.	Min.	Max.	
SCK Clock High width (input)	t_{SCH}	4x	–	91.9 (Note2)	–	
SCK Clock Low width (input)	t_{SCL}	4x	–	91.9 (Note1)	–	
SCK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	–	184	–	ns
SCK rise → Output Data	t_{OSS}	$t_{SCY}/2 − 3x − 45$	–	0 (Note1)	–	
SCK rise → Output Data hold	t_{OHS}	$t_{SCY}/2 + x$	–	107.6	–	

Note 1: **Use the frequency of SCLK in a range where the calculation value keeps positive.**
Note 2: **The value indicates a minimum value that enables t_{OSS} to be zero or more.**

(2)   SCK Output Mode (The electrical specifications below are for an SCK signal with a 50% duty cycle.)

Parameter	Symbol	Equation		fsys = 64 MHz		Unit
		Min.	Max.	Min.	Max.	
SCK cycle (programmable)	t_{SCY}	16x	–	250	–	
Output Data → SCK rise	t_{OSS}	$t_{SCY}/2 − 20$	–	105	–	
SCK rise → Output Data hold	t_{OHS}	$t_{SCY}/2 − 20$	–	105	–	ns
Valid Data input → SCK rise	t_{SRD}	x + 45	–	60.6	–	
SCK rise → Input Data hold	t_{HSR}	0	–	0	–	

Note 1: SCK cycle after automatic wait becomes 14x.
Note 2: t_{OSS} after automatic wait may be $t_{SCY}/2$-x-20.
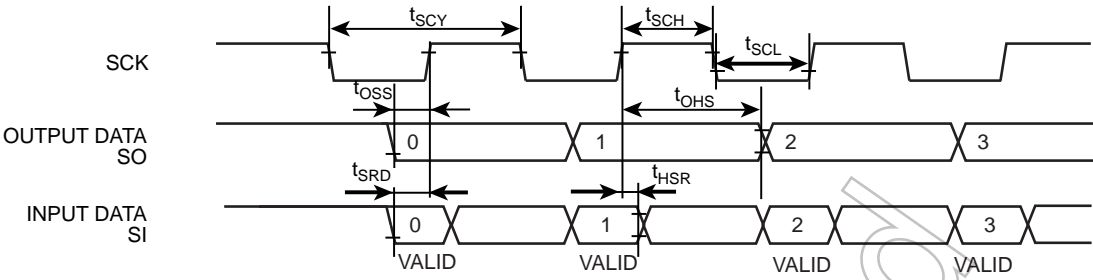
28.6.5   SSP Controller (SSP)

"T" is 1/2 cycles of an internal bus frequency (fsys) in the Equation of the table.

AC measurement condition

- Output level : High = 0.8 x DVDD3, Low = 0.2 x DVDD3
- Input level : Refer low-level input voltage and high-level input voltage in DC Electrical Characteristics.
- Load capacitance : CL = 30pF

Note: The "Equation" column in the table shows the specifications under the conditions DVDD3 2.7 to 3.6 V.

Parameter	Symbol	Equation		fsys = 64 MHz m = 4 n = 12	Unit
		Min.	Max.		
SPCLK period (Master)	T_m	$(m)T$ However more than 50 ns	–	62.5 (16MHz)	ns
SPCLK period (Slave)	T_s	$(n)T$	–	187.5 (8MHz)	
SPCLK rise up time	t_r	–	10.0	10.0	
SPCLK fall down time	t_f	–	10.0	10.0	
Master mode : SPCLK low-level pulse width	t_{WLM}	$(m)T / 2 - 10.0$	–	21.3	
Master mode : SPCLK high-level pulse width	t_{WHM}	$(m)T / 2 - 10.0$	–	21.3	
Slave mode : SPCLK low-level pulse width	t_{WLS}	$(n)T / 2 - 10.0$	–	83.8	
Slave mode : SPCLK high-level pulse width	t_{WHS}	$(n)T / 2 - 10.0$	–	83.8	
Master mode : SPCLK rise / fall → output valid	t_{ODSM}	–	15.0	15.0	
Master mode : SPCLK rise / fall → output data hold	t_{ODHM}	$(m)T / 2 - 10.0$	–	21.3	
Master mode : input data valid → SPCLK rise / fall	t_{IDSM}	15.0	–	15.0	
Master mode : SPCLK rise / fall → input data hold	t_{IDHM}	5.00	–	5.00	
Master mode : SPFSS valid → SPCLK rise / fall	t_{OFSM}	$(m)T - 10.0$	$(m)T + 10.0$	$52.5 - 72.5$	
Slave mode : SPCLK rise / fall → output data valid	t_{ODSS}	–	$(3T) + 22.0$	68.9	
Slave mode : SPCLK rise / fall → output data hold	t_{ODHS}	$(n)T / 2 + (2T)$	–	125	
Slave mode : input data valid → SPCLK rise / fall	t_{IDSS}	0.00	–	0.00	
Slave mode : SPCLK rise / fall → input data hold	t_{IDHS}	$(3T) + 10.0$	–	56.9	
Slave mode : SPFSS valid → SPCLK rise / fall	t_{OFSS}	$(n)T - 15.0$	–	172.5	

Note: Baud rate clock is set under below condition

Master mode :

$\quad$ m=(<CPSDVSR>×(1+<SCR>))= f_{sys} /f_{SPCLK} $\qquad$ <CPSDVR> is set only even number and "m" must set within 65024≥ m≥4

Slave mode :

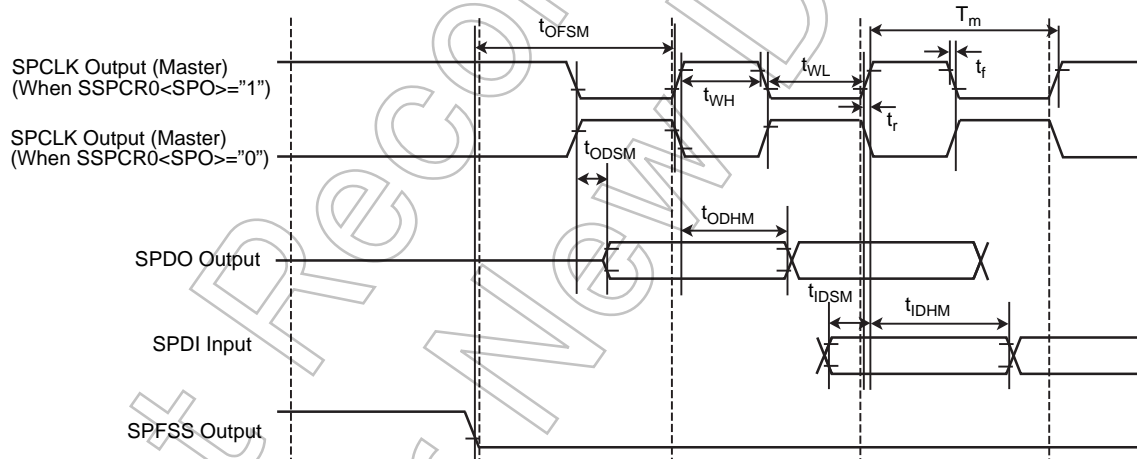$\quad$ n=(<CPSDVSR>×(1+<SCR>))= f_{sys} /f_{SPCLK} $\qquad$ "n" must set within 65024≥ n≥12

28.6.5.1   SSP SPI mode (Master)

- $f_{sys} / 4 \geq f_{SPCLK} \geq f_{sys} / 65024$

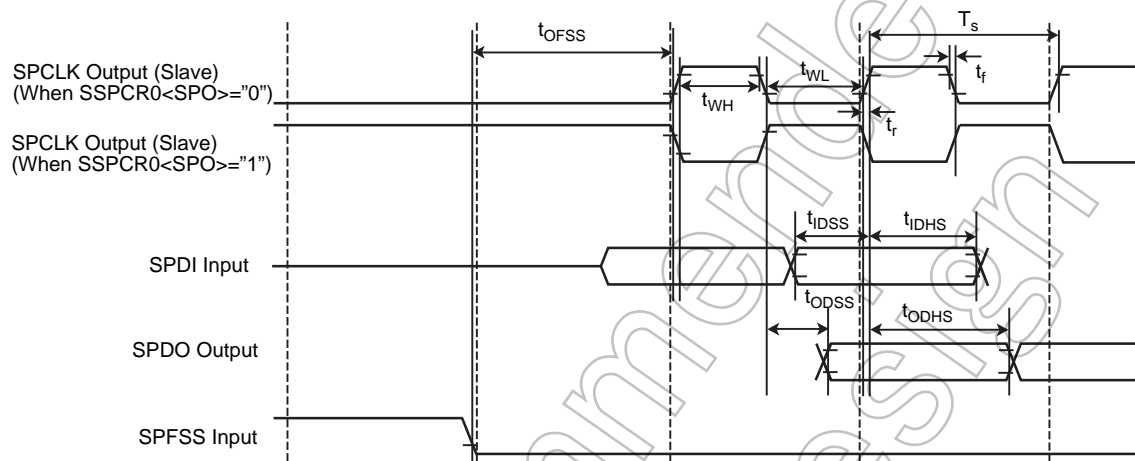(1)   Master SSPCR0<SPH> = "0" (Data is latched on the first edge)
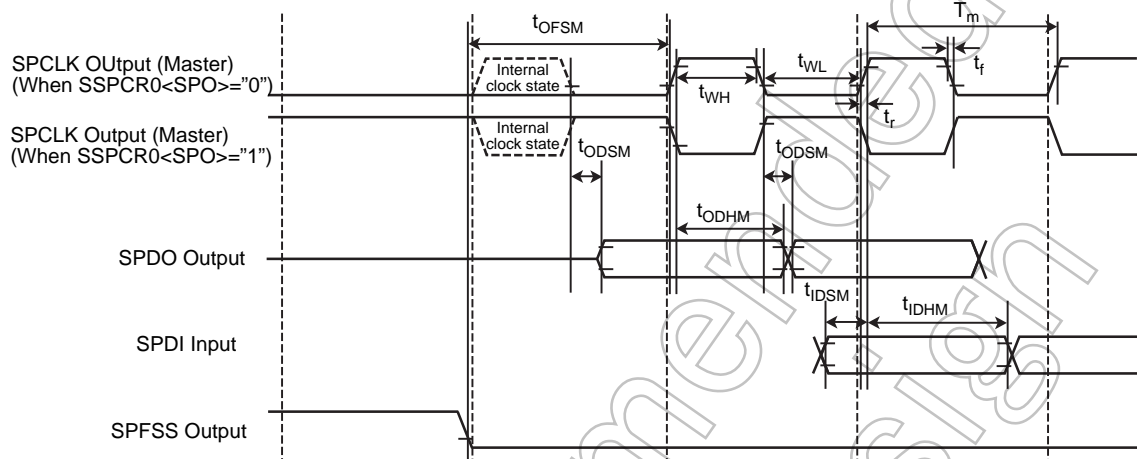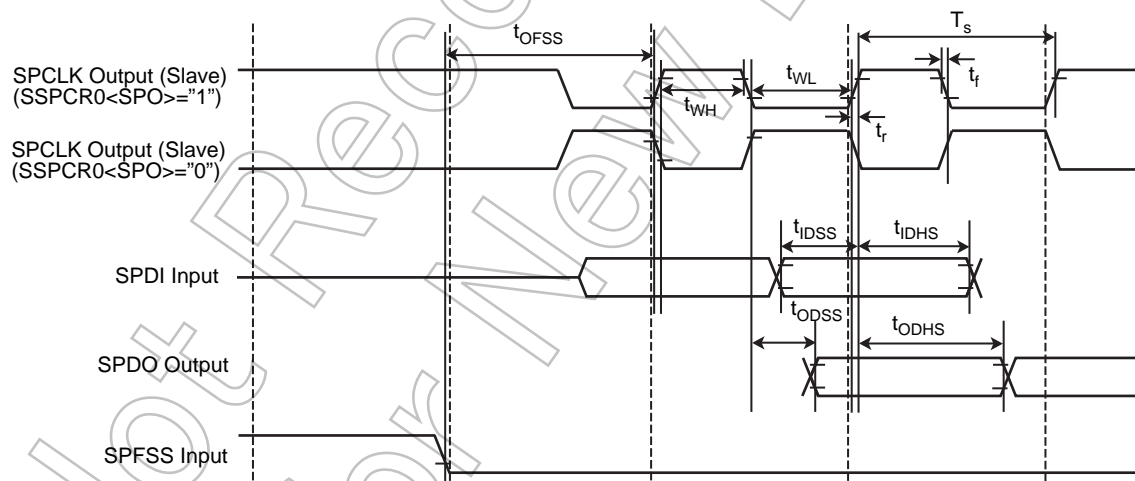

(2)   Master SSPCR0<SPH> = "1" (Data is latched on the second edge)

28.6.5.2   SSP SPI mode (Slave)

- $f_{sys} / 12 \geq f_{SPCLK} \geq f_{sys} / 65024$

(1)   Slave SSPCR0<SPH> = "0" (Data is latched on the first edge)

(2)   Slave SSPCR0<SPH> = "1" (Data is latched on the second edge)

28.6.6   USB Host Controller

The clock of USB host controller is as same as system clock fsys.

DVDD3A = 3.3±0.3V

Parameter	Symbol	Min.	Max.	Unit
D+,D− rise time	t_R	4	20	ns
D+,D− fall time	t_F	4	20	
Differential common mode range	VDI	0.2	–	V
Cross over signal voltage	V_{CRS}	1.3	2.0	
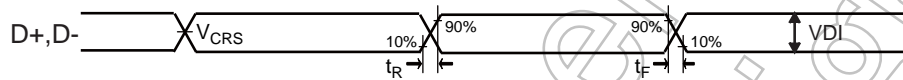
28.6.7   16-bit timer / even counter

28.6.7.1   Event counter

In the table below, the letter x represents the 16-bit timer / event counter operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		fsys = 64 MHz		Unit
		Min.	Max.	Min.	Max.	
Clock low-level pulse width	t_{VCKL}	2x + 100	–	131.3	–	ns
Clock high-level pulse width	t_{VCKH}	2x + 100	–	131.3	–	ns

28.6.7.2   Capture

In the table below, the letter x represents the 16-bit timer / event counter operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		fsys = 64 MHz		Unit
		Min.	Max.	Min.	Max.	
Clock low-level pulse width	t_{CPL}	2x + 100	–	131.3	–	ns
Clock high-level pulse width	t_{CPH}	2x + 100	–	131.3	–	ns

28.6.8   External Interrupt

In the table below, the letter x represents the fsys cycle time.

1.   Except STOP release interrupt

Parameter	Symbol	Equation		fsys = 64 MHz		Unit
		Min.	Max.	Min.	Max.	
INT0 to 7 Low-level pulse width	t_{INTAL}	x + 100	–	115.6	–	ns
INT0 to 7 High-level pulse width	t_{INTAH}	x + 100	–	115.6	–	ns

2.   STOP release interrupt

Parameter	Symbol	Min.	Max.	Unit
INT0 to 7 Low-level pulse width	t_{INTBL}	100	–	ns
INT0 to 7 High-level pulse width	t_{INTBH}	100	–	ns

28.6.9 $\overline{\text{NMI}}$

Parameter	Symbol	Min.	Max.	Unit
$\overline{\text{NMI}}$ Low-level pulse width	t_{INTCL}	100	–	ns

28.6.10   SCOUT Pin AC Characteristic

Parameter	Symbol	Equation		fsys = 48 MHz		Unit
		Min.	Max.	Min.	Max.	
High-level pulse width	t_{SCH}	0.5T − 5	–	5.5	–	ns
Low-level pulse width	t_{SCL}	0.5T − 5	–	5.5	–	ns

Note:**In the above table, the letter T represents the cycle time of the SCOUT output clock.**
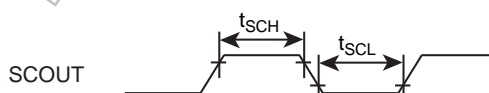
28.6.11 Debug communication

Parameter	Symbol	Min.	Max.	Unit
CLK cycle	T_{dck}	100	–	ns
CLK rise → Output data hold	T_{d1}	4	–	ns
CLK rise → Output data valid	T_{d2}	–	30	ns
Input data valid ← CLK rise	T_{ds}	20	–	ns
CLK rise → Input data hold	T_{dh}	15	–	ns
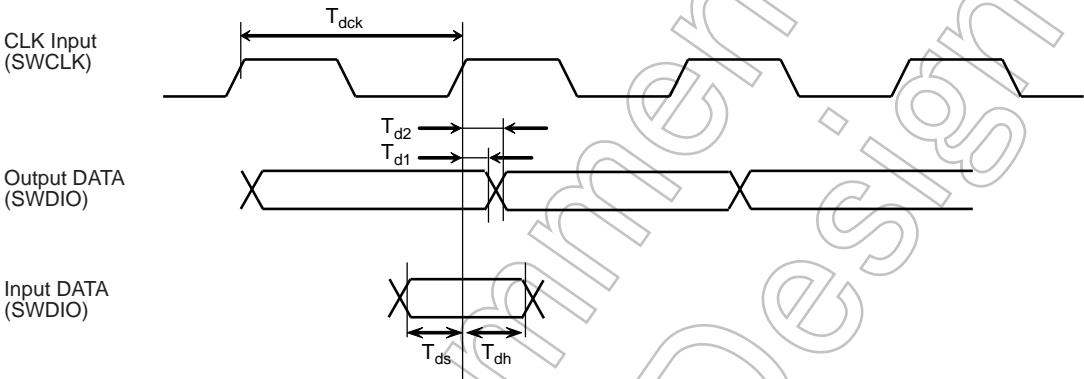
28.6.12 ETM Trace

Parameter	Symbol	Min.	Max.	Unit
TRACECLK cycle	t_{tclk}	31.25	–	ns
TRACEDATA valid ← TRACECLK rise	t_{setupr}	2	–	ns
TRACECLK rise → TRACEDATA hold	t_{holdr}	1	–	ns
TRACEDATA valid ← TRACECLK rise	t_{setupf}	2	–	ns
TRACECLK fall → TRACEDATA hold	t_{holdf}	1	–	ns
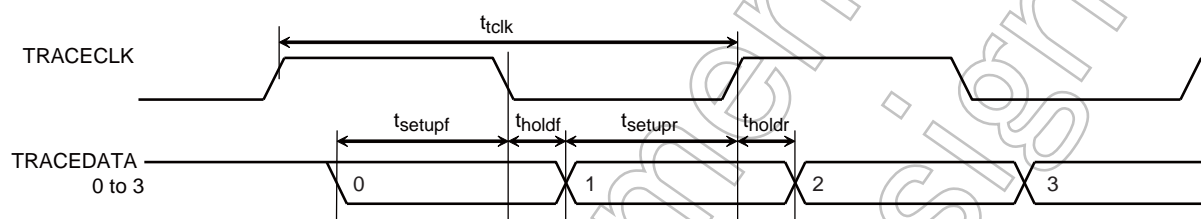
28.7 Flash Characteristics

28.7.1 Erase / Write Characteristics

Parameter	Condition	Min.	Typ.	Max.	Unit
The number of E / W cycle	DVDD3 = AVDD3 = RVDD3 = 2.7 V to 3.6 V Ta = 0 to 70 °C	–	–	100	cycle

28.8   Oscillation Circuit

Oscillation circuit is shown as below ;
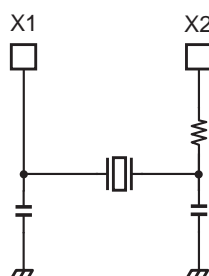
Figure 28-1 High-frequency oscillation connection
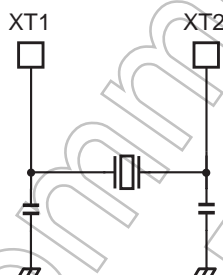
Figure 28-2 Low-frequency oscillation connection

Note:**To obtain a table oscillation, load capacity and the position of the oscillator must be configured properly. Since these factors are strongly affectd by substrate pattern, please evaluate oscillation stability using the substrate you use.**

The TMPM363F10FG has been evaluated by the oscillator vender below. Please refer this information when selecting exter-nal parts.

28.8.1   Ceramic oscillator

The TMPM363F10FG recommends the high-frequency oscillator by Murata

Manufacturing Co., Ltd.

Please refer to the Murata Website for details.

28.8.2   Crystal oscillator

The TMPM363F10FG recommends the high-frequency oscillator by KYOCERA Corporation.

Please refer to the KYOCERA Website for details.

28.8.3   Precaution for designing printed circuit boad

Be sure to design printed circuit board patterns that connect a crystal unit with other oscillation elements so that the lenghts of such patterns become shortest possible to prevent deterioration of characteristics due to stray capacitances and wiring inductance. For multi-layer circuit boards, it is importtant not to wire the ground and other signal patterns right beneath the oscillation circuit.

For more information ,please refer to the URL of the oscillator vendor.

28.9   Handling Precaution
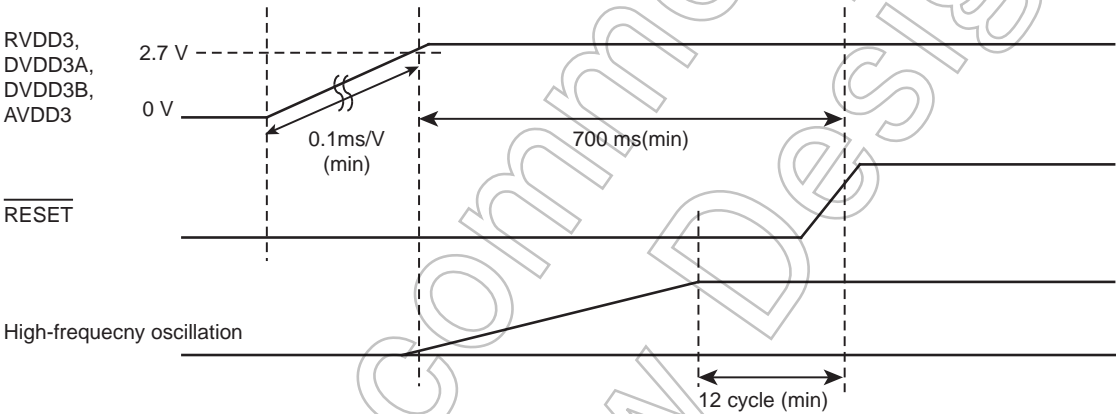
28.9.1   Notice of Power Supply

28.9.1.1   Notice of Power on

The power supply must be raised (from 0V to 2.7V) at a speed of 0.1ms/V or slower.

The power-on sequence must consider the time for the internal regulator and oscillator to be stable. In the TX03, the internal regulator requires at least 700 µs to be stable.

The time required to achieve stable oscillation varies with system. At cold reset, the external reset pin must be kept "Low" for a duration of time sufficiently long enough for the internal regulator and oscillator to be stable.

The power-on sequence is shown as below ;

28.9.1.2   Notice of Power on again

When power on again after power off, the voltage of power supply is equal or less than 0.5V and keeps this level until more than 2ms. After this, the power supply must be raised at a speed of 0.1ms/V or slower. Regarding to reset, refer to 28.9.1.1.

29. Package Dimensions

LQFP100−P−1414−0.50H   "Unit : mm"
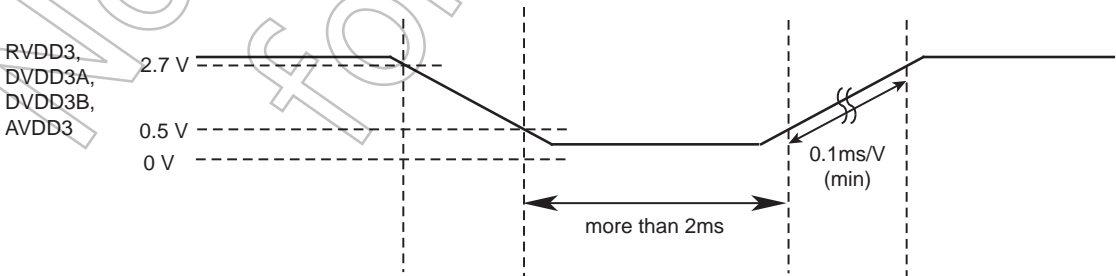
- ## RESTRICTIONS ON PRODUCT USE

Toshiba Corporation and its subsidiaries and affiliates are collectively referred to as "TOSHIBA".
Hardware, software and systems described in this document are collectively referred to as "Product".

- TOSHIBA reserves the right to make changes to the information in this document and related Product without notice.

- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.

- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating a   operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS' PRODUCT DESIGN OR APPLICATIONS.**

- **PRODUCT IS NEITHER INTENDED NOR WARRANTED FOR USE IN EQUIPMENTS OR SYSTEMS THAT REQUIRE EXTRAORDINARILY HIGH LEVELS OF QUALITY AND/OR RELIABILITY, AND/OR A MALFUNCTION OR FAILURE OF WHICH MAY CAUSE LOSS OF HUMAN LIFE, BODILY INJURY, SERIOUS PROPERTY DAMAGE AND/OR SERIOUS PUBLIC IMPACT ("UNINTENDED USE").** Except for specific applications as expressly stated in this document, Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, lifesaving and/or life supporting medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, and devices related to power plant. **IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT.** For details, please contact your TOSHIBA sales representative or contact us via our website.

- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.

- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.

- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.

- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**

- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.

- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. **TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.**

TOSHIBA ELECTRONIC DEVICES & STORAGE CORPORATION

https://toshiba.semicon-storage.com/