TOSHIBA

32 Bit RISC Microcontroller

TX04 Series

TMPM46BF10FG

TOSHIBA ELECTRONIC DEVICES & STORAGE CORPORATION

TOSHIBA

Dear customers

Sep. 01, 2021

Toshiba Electronic Devices & Storage Corporation
Toshiba Electronic Device Solutions Corporation
5801-1, Horikawatyou, Saiwai Ward, Kawasaki City, Kanagawa prefecture, 212-8520
Tel: +81-44-548-2200
Fax: +81-44-548-8965

Error correction for technical datasheet of Universal Asynchronous Receiver-Transmitter

Thank you for using Toshiba microcontrollers.

We have found the mistakes about occurring transmission interrupt timing of the Universal Asynchronous Receiver-Transmitter (UART and FUART) and the Universal Asynchronous Receiver-Transmitter Circuit with 50% duty mode (UART) in our technical datasheet and reference manual. We will inform you about the mistakes in this document.

We apologize for any inconvenience, but we ask that you review the content.

If you have any questions, please contact our sales representative.

1. Applicable products

TMPM342FYXBG	TMPM440FEXBG	TMPA900CMXBG
TMPM343F10XBG	TMPM440F10XBG	TMPA901CMXBG
TMPM343FDXBG	TMPM461F10FG	TMPA910CRAXBG
TMPM366F20AFG	TMPM461F15FG	TMPA910CRBXBG
TMPM366FWFG	TMPM462F10FG	TMPA911CRXBG
TMPM366FYFG	TMPM462F15FG	TMPA912CMXBG
TMPM366FDFG	TMPM46BF10FG	TMPA913CHXBG
TMPM366FWXBG	TMPM4G6FDFG	
TMPM366FYXBG	TMPM4G6FEFG	
TMPM366FDXBG	TMPM4G6F10FG	
TMPM367FDFG	TMPM4G7FDFG	
TMPM367FDXBG	TMPM4G7FEFG	
TMPM368FDFG	TMPM4G7F10FG	
TMPM368FDXBG	TMPM4G8FDFG	
TMPM369FDFG	TMPM4G8FDXBG	
TMPM369FDXBG	TMPM4G8FEFG	
TMPM36BF10FG	TMPM4G8FEXBG	
TMPM36BFYFG	TMPM4G8F10FG	
TMPM381FWDFG	TMPM4G8F10XBG	
TMPM381FWFG	TMPM4G8F15FG	
TMPM383FSEFG	TMPM4G8F15XBG	
TMPM383FSUG	TMPM4G9FDFG	
TMPM383FWEFG	TMPM4G9FDXBG	
TMPM383FWUG	TMPM4G9FEFG	
TMPM3V4FSEFG	TMPM4G9FEXBG	
TMPM3V4FSUG	TMPM4G9F10FG	
TMPM3V4FWEFG	TMPM4G9F10XBG	
TMPM3V4FWUG	TMPM4G9F15FG	
TMPM3V6FWDFG	TMPM4G9F15XBG	
TMPM3V6FWFG		

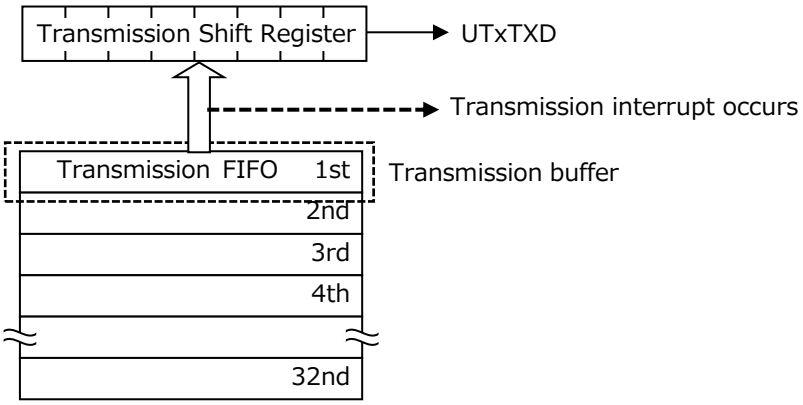
2. Details

The timing of occurring transmission interrupt is shown as below.

There is the mistake in the timing of occurring transmission interrupt when the transmission FIFO is not used only, and it will be corrected as below. There is no mistake in the transmission interrupt timing when using the transmission FIFO.

2.1. When the transmission FIFO is unused

Transmission interrupt occurs when a transmission data moves from the transmission buffer (the 1st level of transmission FIFO) to transmission shift register. (When the transmission buffer becomes empty.)



2.1.1. The timing of occurring transmission interrupt

The transmission interrupt when the transmission FIFO is not used occurs when the transmission buffer becomes empty because it notifies the timing of writing to the transmission buffer for the next data. The transmission interrupt is automatically cleared when the next data is written to the transmission buffer. Therefore, it is not necessary to clear the transmission interrupt by software when continuously transmitting data (set UARTxICR<TXIC> to "1").

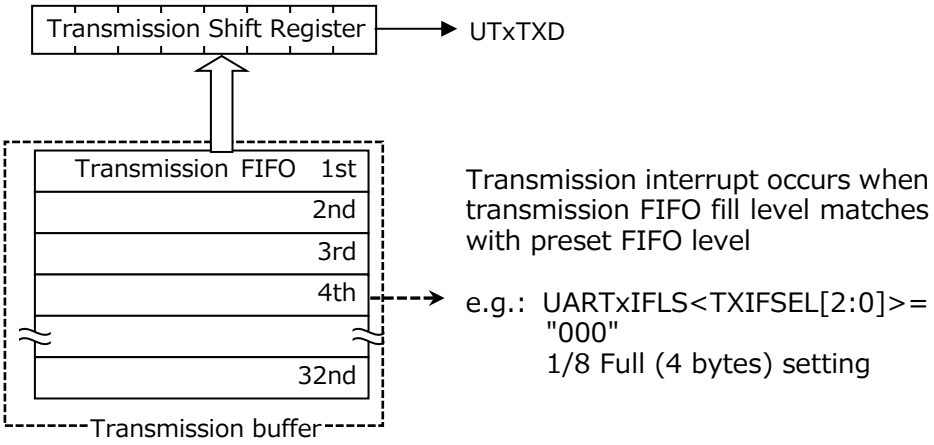
When the transmission is terminated, the final transmission data is transferred to the shift register, and the final transmission interrupt occurs when the transmission buffer becomes empty. If the next data is not written to the transmission buffer, the transmission interrupt can be intentionally cleared by executing clear by software in the interrupt handler (set UARTxICR <TXIC> to "1").

If you execute the transmission interrupt clear by software during data transmission (set UARTxICR <TXIC> to "1"), the transmission interrupt does not occur if you write the data to the transmission buffer at the same time as the STOP bit is generated. In order to generate the transmission interrupt reliably, do not clear the transmission interrupt by software, write data to the transmission buffer during data transmission, or write the data to the transmission buffer while transmission is stopped (when UARTxFR<BUSY>= "0").

When transmitting data continuously, it is recommended to transfer the data by using the transmission FIFO in the next section.

2.2. When transmission FIFO is used

Transmission interrupt occurs when transmission FIFO level matches with preset FIFO level which is specified by UARTxIFLS<TXIFSEL[2:0]>.



2.2.1. The timing of occurring transmission interrupt

When using the transmission FIFO, the transmission interrupt occurs when transmission FIFO level matches with preset FIFO level.

For example, in case of UARTxIFLS<TXIFSEL[2:0]> = "000" (1/8 full 4 bytes setting), the transmission interrupt occurs when the transmission FIFO level matches with 4th level.

The transmission interrupt is cleared when data whose FIFO level is above the specified FIFO level is stored in the transmission FIFO and occurs again when the specified FIFO level is reached.

3. Description

The description about occurring transmission interrupt is different from each product. The chapter number of placement for each product are shown as below.

There is the mistake in the timing of occurring transmission interrupt when the transmission FIFO is not used only, and it will be corrected as below. There is no mistake in the transmission interrupt timing when using the transmission FIFO.

The details of revised description for the mistake will be explained in "4. Revised description" below, and the revised description is all target products in common.

3.1. Description Type A

3.1.1. Applicable products and chapter of the description

Product name	Chapter of the description
TMPM342FYXBG	16.4.7
TMPM366F20AFG (Note)	15.4.7
TMPM366FWFG, TMPM366FYFG, TMPM366FDFG, TMPM366FWXBG, TMPM366FYXBG, TMPM366FDXBG	16.4.7
TMPM367FDFG, TMPM367FDXBG, TMPM368FDFG, TMPM368FDXBG, TMPM369FDFG, TMPM369FDXBG,	13.4.7
TMPM36BFYFG, TMPM36BF10FG	13.4.7
TMPA900CMXBG, TMPA901CMXBG, TMPA910CRAXBG, TMPA910CRBXBG, TMPA911CRXBG, TMPA912CMXBG, TMPA913CHXBG	3.13.1.1 (7)

Note: The chapter in a section of the Universal Asynchronous Receiver-Transmitter (UART).

Type A

Original description (Red box)

Interrupt type	Interrupt timing
Overrun error	After receiving the stop bit of Overflow data
Break error	After receiving STOP bit
Parity error	After receiving parity data
Frame error	After receiving frame over bit
Receive time out error	After 511 clocks(Baud16) from Receive FIFO data storage
Transmit interrupt	After transmitting the last data (MSB data)
Receive interrupt	After receiving STOP bit

3.2. Description Type B(1)

3.2.1. Applicable products and chapter of the description

Product name	Chapter of the description
TMPM461F10FG, TMPM461F15FG, TMPM462F10FG, TMPM462F15FG	14.4.6.2

Type B(1)

Original description (Red box)

Interrupt source	Interrupt generation timing
Overrun error generation	After a stop bit is received when FIFO is full.
Break error interrupt	After a stop bit is received.
Parity error generation	After a parity data is received.
Framing error generation	After bit data that generates frame over is received.
Reception timeout interrupt	After data is received in receive FIFO, then 511 clocks of Baud16 has elapsed.
Transmission interrupt	After MSB of last data is transmitted.
Reception interrupt	After a stop bit is received.

3.3. Description Type B(2)

3.3.1. Applicable products and chapter of the description

Product name	Chapter of the description
TMPM343FDXBG, TMPM343F10XBG, TMPM366F20AFG (Note)	16.4.6.2
TMPM381FWFG, TMPM381FWDFG, TMPM383FSUG, TMPM383FSEFG, TMPM383FWUG, TMPM383FWEFG, TMPM3V4FSUG, TMPM3V4FSEFG, TMPM3V4FWUG, TMPM3V4FWEFG, TMPM3V6FWFG, TMPM3V6FWDFG	11.4.6.2
TMPM440FEXBG, TMPM440F10XBG	26.4.6.2

Note: The chapter in a section of the Universal Asynchronous Receiver-Transmitter Circuit with 50% duty mode (UART).

Type B(2)

Original description (Red box)

Interrupt source	Interrupt generation timing	
Overrun error generation	After a stop bit is received when FIFO is full.	
Break error interrupt	After a stop bit is received.	
Parity error generation	After a parity data is received.	
Framing error generation	After bit data that generates frame over is received.	
Reception timeout interrupt	After data is received in receive FIFO, then 511 clocks of Baud16 has elapsed.	
Transmission interrupt	When the FIFO is unused:	After the transmission is enabled, when a START bit and STOP bit in the first byte of the transmission data are sent, a transmit interrupt occurs. In the second byte and the following byte, a transmit interrupt occurs only when a STOP bit is sent. (In this case, each interrupt is cleared after the transmit data is written.)
	When the FIFO is used:	When a STOP bit is sent (after the MSB data is transmitted), if the amount of data in the FIFO is the same level as the specified level of FIFO, a transmit interrupt occurs.
Reception interrupt	When the FIFO is unused:	A receive interrupt occurs when the FUART receives a STOP bit.
	When the FIFO is used:	A receive interrupt occurs when the FUART receives a STOP bit included in the data that fills the FIFO to the specified level.

3.4. Description Type B(3)

3.4.1. Applicable products and chapter of the description

Product name	Chapter of the description
TMPM4G6FDFG, TMPM4G6FEFG, TMPM4G6F10FG, TMPM4G7FDFG, TMPM4G7FEFG, TMPM4G7F10FG, TMPM4G8FDFG, TMPM4G8FDXBG, TMPM4G8FEFG, TMPM4G8FEXBG, TMPM4G8F10FG, TMPM4G8F10XBG, TMPM4G8F15FG, TMPM4G8F15XBG, TMPM4G9FDFG, TMPM4G9FDXBG, TMPM4G9FEFG, TMPM4G9FEXBG, TMPM4G9F10FG, TMPM4G9F10XBG, TMPM4G9F15FG, TMPM4G9F15XB	Reference Manual (Note) Full Universal Asynchronous Receiver Transmitter Circuit (FUART-B) 3.8.2

Note: In this reference manual, read UARTxIFLS with ***[FURTxIFLS]***, UARTxICR with ***[FURTxICR]***, UARTxFR with ***[FURTxFR]***.

Type B(3)

Original description (Red box)

Interrupt source	Interrupt generation timing
Overrun error generation	After a STOP bit is received when FIFO is full.
Break error interrupt	After a STOP bit is received.
Parity error generation	After a parity data is received.
Framing error generation	After Bit data that generates frame over is received.
Reception timeout interrupt	After data is received in receive FIFO, then 511 clocks of the transfer clock have elapsed.
Transmission interrupt	1 byte hold register (FIFO is unused): After transmission has been enabled. For the first Byte, when START bit starts to transmit and when STOP bit starts to transmit. For the second Byte or later, when STOP bit starts to transmit (after each interrupt has been generated and the interrupt is cleared by each data write).
	FIFO is enabled: When the data count in FIFO becomes a set level at the start of STOP bit transmission (after MSB data is transmitted).
Reception interrupt	1 byte hold register (FIFO is unused): After STOP bit is received.
	FIFO is enabled: After STOP bit is received when the data count in FIFO becomes a set level.

3.5. Description Type C

3.5.1. Applicable products and chapter of the description.

Product name	Chapter of the description
TMPM46BF10FG	19.4.6.2

Type C

Original description (Red box)

Interrupt source	Interrupt generation timing
Overrun error generation	After a stop bit is received when FIFO is full.
Break error interrupt	After a stop bit is received.
Parity error generation	After a parity data is received.
Framing error generation	After bit data that generates frame over is received.
Reception timeout interrupt	After data is received in receive FIFO, then 511 clocks of Baud16 has elapsed.
Transmission interrupt	After MSB of last data is transmitted.
Reception interrupt	After a stop bit is received.

4. Revised description

The description of the transmission interrupt occurrence timing differs depending on the products, but the correct description is as follows in common.

4.1. The timing of occurring transmission interrupt

The transmission interrupt when the transmission FIFO is not used occurs when the transmission buffer becomes empty because it notifies the timing of writing to the transmission buffer for the next data. The transmission interrupt is automatically cleared when the next data is written to the transmission buffer. Therefore, it is not necessary to clear the transmission interrupt by software when continuously transmitting data (set UARTxICR<TXIC> to "1").

When the transmission is terminated, the final transmission data is transferred to the shift register, and the final transmission interrupt occurs when the transmission buffer becomes empty. If the next data is not written to the transmission buffer, the transmission interrupt can be intentionally cleared by executing clear by software in the interrupt handler (set UARTxICR <TXIC> to "1").

If you execute the transmission interrupt clear by software during data transmission (set UARTxICR <TXIC> to "1"), the transmission interrupt does not occur if you write the data to the transmission buffer at the same time as the STOP bit is generated. In order to generate the transmission interrupt reliably, do not clear the transmission interrupt by software, write data to the transmission buffer during data transmission, or write the data to the transmission buffer while UART transmission is stopped (when UARTxFR<BUSY> = "0").

End of document

©2026
Toshiba Electronic Devices & Storage Corporation

****
Arm, Cortex and Thumb are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved.
****

General precautions on the use of Toshiba MCUs

This Page explains general precautions on the use of Toshiba MCUs.

Note that if there is a difference between the general precautions and the description in the body of the document, the description in the body of document has higher priority.

1. The MCUs' operation at power-on

   At power-on, internal state of the MCUs is unstable. Therefore, state of the pins is undefined until reset operation is completed.

   When a reset is performed by an external reset pin, pins of the MCUs that use the reset pin are undefined until reset operation by the external pin is completed.

   Also, when a reset is performed by the internal power-on reset, pins of the MCUs that use the internal power-on reset are undefined until power supply voltage reaches the voltage at which power-on reset is valid.

2. Unused pins

   Unused input/output ports of the MCUs are prohibited to use. The pins are high-impedance.

   Generally, if MCUs operate while the high-impedance pins left open, electrostatic damage or latch-up may occur in the internal LSI due to induced voltage influenced from external noise.

   Toshiba recommend that each unused pin should be connected to the power supply pins or GND pins via resistors.

3. Clock oscillation stability

   A reset state must be released after the clock oscillation becomes stable. If the clock is changed to another clock while the program is in progress, wait until the clock is stable.

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
2015/02/03	Tentative 1	First Release
2015/07/21	Tentative 3	First Release
2015/11/12	1	First Release
2019/03/29	2	Contents Revised
2022/03/31	3	Contents Revised
2022/09/30	4	Contents Revised
2023/07/14	5	Contents Revised
2023/07/31	6	Contents Revised

TMPM46BF10FG

The TMPM46BF10FG is a 32-bit RISC microprocessor with an Arm® Cortex®-M4 processor with FPU core.

Features of the TMPM46BF10FG are shown as follows:

1.1 Features

1. Arm Cortex-M4 processor with FPU 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]

      A 32-bit multiplication (32 × 32 = 32 bits) and multiply-accumulate operation (32 + 32 × 32 = 32 bits) can be executed with one clock.

      SIMD (Single Instruction Multiple Data) operation can be executed with one clock.

      A division takes within 2 to 12 cycles

      [Low power consumption]

      Optimized design using a low power consumption library

      Standby function that stops the operation of the microcontroller core
   c. High-speed interrupt response suitable for real-time control

      An interruptible long instruction.

      Stack push automatically handled by hardware.

2. Single-precision floating-point execution unit (FPU)
   - Conformity to IEEE754 standard
   - Addition/subtraction/multiplication can be executed with one clock. Multiply-accumulate operation can be executed with 3 clocks.
   - Dedicated data register can perform parallel processing aside from CPU.

3. On-chip program and data memory
   - On-chip RAM : 514 K bytes

     Back-up RAM 2 K bytes is included.
   - On-chip Flash ROM :

     TMPM46BF10FG : 1024Kbyte

4. External bus interface (EBIF)
   - Expandable to 16MB (shared with program and data)
   - Supports multiplex bus : 8-bit/16-bit width
   - Chip select/wait controller: 4 channels

5. SLC NAND Flash Controller (SNFC)
   - Memory size: 1 G bit to 4 G bit
   - Data bus: 8-bit width
   - ECC: Incorporates the error detection/correction circuit based on BCH encoding/decoding
   - Automatic load function: Achieves high-speed data transfer using the µDMA controller.

6. AES processor (AES)
   - Key lengths are described in FIPS-197. Supports 128-bit,192-bit, and 256-bit lengths
   - Supports ECB, CBC, and CTR algorithm.

7. Secure hash processor (SHA)
   - Conformity with FIPS PUB 180-3 Secure Hash standard Algorithm (SHA-224 and SHA256).
   - A message length is up to 260 bytes. Calculates in units of 512 bit.
   - Message automatic padding

8. Entropy seed generator (ESG)
   - Generates a 512 bit random seed using the ring oscillator. - 1 Channel

9. Multiple length arithmetic coprocessor (MLA)
   - Montgomery reduction, multiple length arithmetic (addition and subtraction) are implemented.
   - Supports multiple length arithmetic equivalent to the key length of 256 bits of elliptic curve cryptography.

10. µDMA controller (µDMAC) : 32 channels / 3 units

    Transfer mode : Built-in memory, peripheral function and external memory

11. Clock controller (CG)
    - External high-speed oscillation : 8 to 16MHz (Oscillator)
      : 8 to 40MHz (Input clock)
    - Internal high-speed oscillation : 10MHz
    - Installed 1 unit of Built-in PLL. (2x, 3x, 4x, 5x, 6x, 8x, 10x, 12x)
    - Clock gear function: divides high-speed clock into 1/1, 1/2, 1/4, 1/8 or 1/16.

12. Low power consumption function
    - IDLE, STOP1, STOP2

13. External interrupt source
    - External interrupt pin 16 pins : The order of precedence can be set over 7 levels.

14. Input/ output ports (PORT) : 72 pins
    - Input/Output pin : 71 pins
    - Output pin : 1 pins

15. 16-bit timer (TMRB) : 8 channels
    - 16-bit interval timer mode
    - 16-bit event counter mode

- 16-bit PPG output (4-channels can be started synchronously)
- Input capture function

16. Real time clock (RTC): 1 channel
    - Clock (hour, minute and second)
    - Calendar (month, week, date and leap year)
    - Clock adjustment function
    - +/-30s adjustment function
    - 1Hz clock output

17. Multi purpose timer (MPT) : 4channels
    - IGBT control
    - 16-bit timer

18. Watchdog timer (WDT) : 1 channel

    Reset or non-maskable interrupt (NMI) are occurred.

19. Serial channel (SIO/UART) : 4 channels
    - Selectable either UART or synchronous mode
    - Transmit FIFO: 4-stage 8-bit width, receive FIFO: 4-stage 8-bit width

20. Asynchronous serial communication interface (UART): 2 channels
    - Data length: 5, 6, 7, 8 bits
    - Transmit FIFO: 32-stage 8-bit width, transmit FIFO: 32-stage 12-bit width

21. I2C bus interface (I2C) : 3 channels
    - Communication rate 100kbps / 400kbps

22. Synchronous Serial interface (SSP) : 3 channels
    - Communication protocol that includes SPI: 3 types (SPI/SSI/Microwire)
    - Communication speed

      Channel 0 :

      Max 20MHz @ fsys=120MHz when master mode

      Max 6.6MHz @ fsys=120MHz when slave mode

      Channel 1/2 :

      Max 10MHz @ fsys=120MHz when master mode

      Max 3.3MHz @ fsys=120MHz when slave mode

23. 12-bit AD converter (ADC): 1 unit (8channels)
    - Fixed channel / Channel scan mode
    - Single / repeat mode
    - External trigger start and internal timer trigger start are possible.
    - Repeat conversion is capable.
    - AD monitoring
    - Minimum conversion time: 1 μs (ADC conversion clock 40MHz)

24. LVD function : 1 unit

25. Maximum operating frequency : 120MHz

26. Endian
    - Little-endian

27. Debug interface
    - JTAG/SWD/SWV/DATA TRACE (4 bits) are supported.

28. Operating voltage range
    - 2.7 to 3.6V

29. Temperature range
    - −40°C to 85°C

30. Package

    LQFP100 (14mm x 14mm, 0.5mm pitch)

1.2 Block Diagram

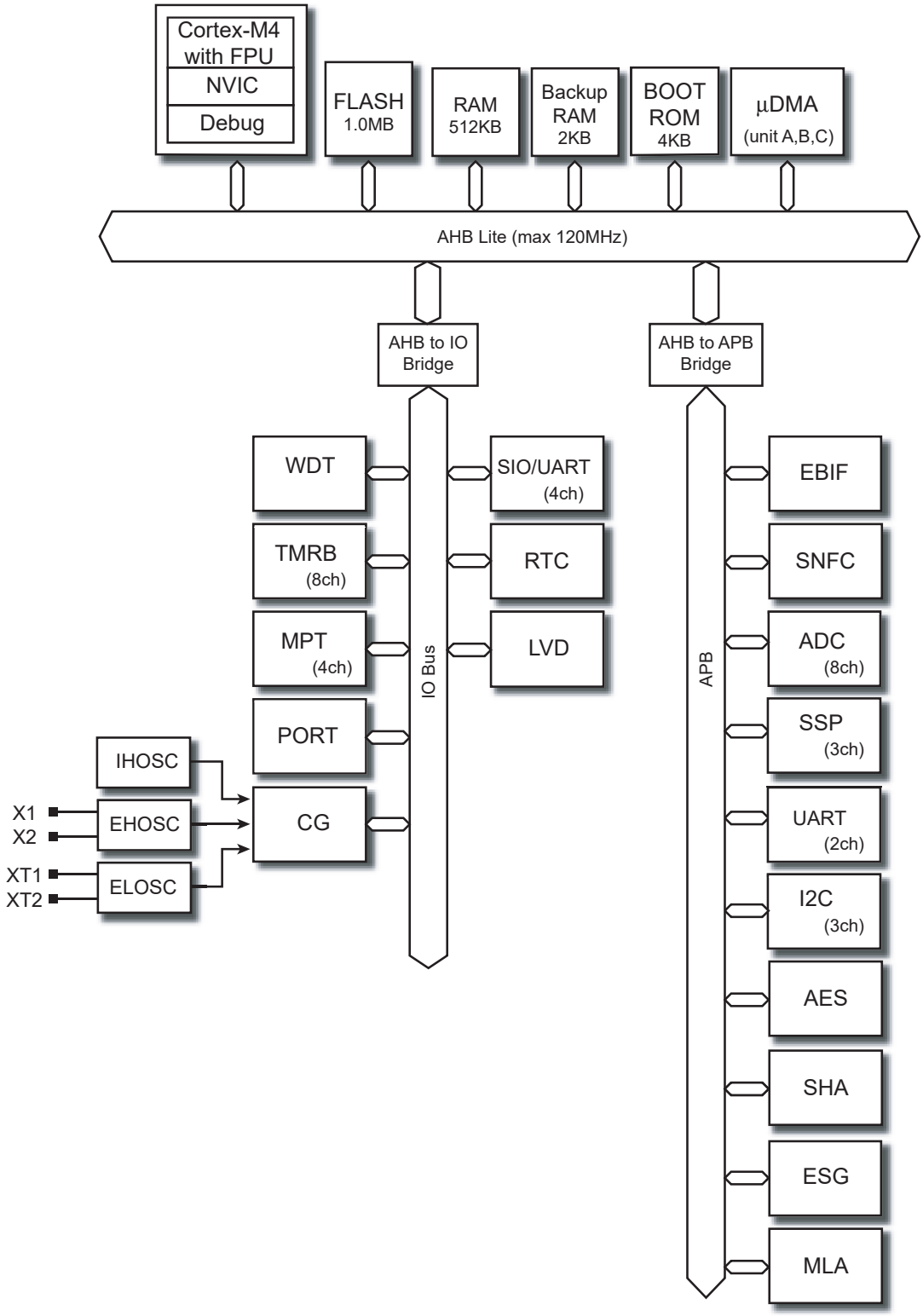


Figure 1-1 Block Diagram

1.3 Pin Layout (Top view)

The bellowed figure shows the pin layout of TMPM46BF10FG.

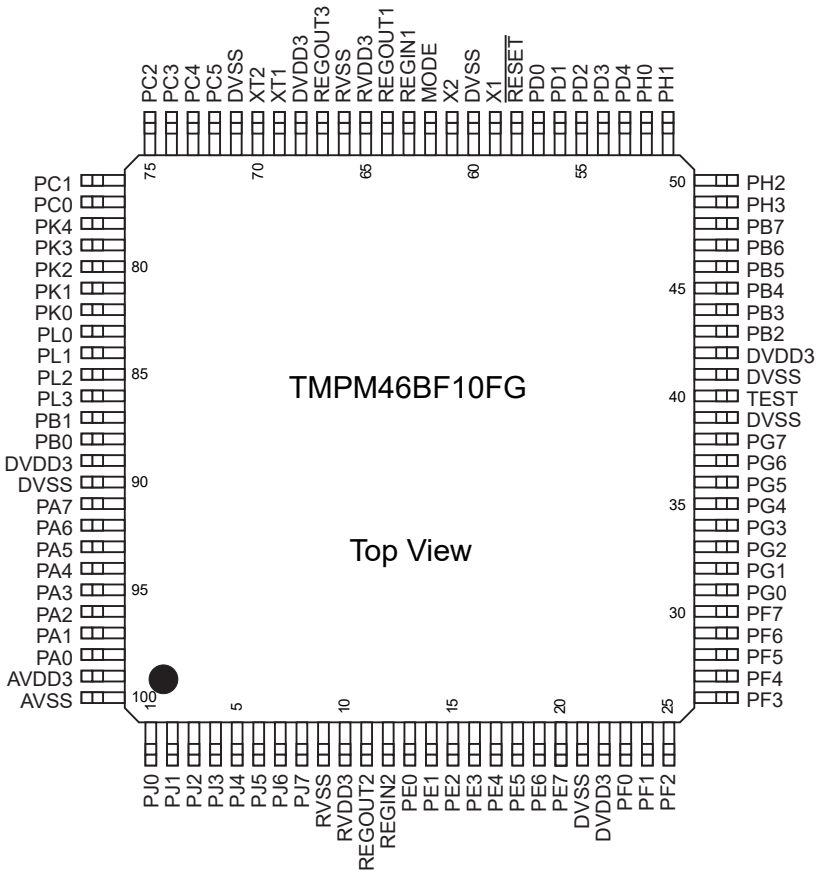


Figure 1-2 Pin Layout (LQFP100 TOP VIEW)

1.4 Pin names and Functions

1.4.1 Pin names and Functions for each peripheral function, control pin and power supply pin

1.4.1.1 Peripheral functions

Table 1-1 The number of pins and Pin names

Peripheral function	Pin name	Input or Output	Function
Clock / Mode control	SCOUT	Output	System clock output
External interrupt	INTx	Input	External interrupt input pin x External interrupt input pin x has a noise filter (Filter width 30ns typ.).
μDMA	$\overline{\text{DMAREQx}}$	Input	DMA request input pin
External bus interface	An	Output	Address bus output
	ADn	I/O	Address and data bus input / output
	$\overline{\text{RD}}$	Output	Read strobe output pin
	$\overline{\text{WR}}$	Output	Write strobe output pin
	ALE	Output	Address latch enable output pin
	$\overline{\text{BELL}}$	Output	Byte enable output pin
	$\overline{\text{BELH}}$	Output	Byte enable output pin
	$\overline{\text{CSn}}$	Output	Chip select output pin
SLC NAND Flash controller	$\overline{\text{SNFCCE}}$	output	Chip enable output pin
	SNFCCLE	output	Command latch enable output pin
	SNFCALE	output	Address latch enable output pin
	$\overline{\text{SNFCRE}}$	output	Read enable output pin
	$\overline{\text{SNFCWE}}$	output	Write enable output pin
	SNFCDn	I/O	Data bus input / output
	SNFCRB	Input	Ready/Busy input pin
16 bit timer / even counter	TBxIN0	Input	Input capture input pin 0
	TBxOUT	Output	output pin
16 bit multi purpose timer	MTxTBIN	Input	Timer mode input pin
	MTxTBOUT	Output	Timer mode output pin
	MTxIN	Input	IGBT mode input pin
	$\overline{\text{GEMGx}}$	Input	IGBT mode emergency stop input pin
	MTxOUT0	Output	IGBT mode output pin 0
	MTxOUT1	Output	IGBT mode output pin 1
SIO/UART	SCxTXD	Output	Data output pin
	SCxRXD	Input	Data input pin
	SCxSCK	I/O	Clock input / output pin
	$\overline{\text{SCxCTS}}$	Input	Hand shake input pin

Table 1-1 The number of pins and Pin names

Peripheral function	Pin name	Input or Output	Function
UART	UTxTXD	Output	Data output pin
	UTxIROUT	Output	IrDA 1.0 data output pin
	UTxRXD	Input	Data input pin
	UTxIRIN	Input	IrDA 1.0 data input pin
	UTxDCD	Input	Modem status (DCD) input pin
	UTxDSR	Input	Modem status (DSR) input pin
	UTxRIN	Input	Modem status (RIN) input pin
	$\overline{\text{UTxCTS}}$	Input	Possible transferring input pin
	UTxDTR	Output	Modem control (DTR) output pin
	$\overline{\text{UTxRTS}}$	Output	Transfer request output pin
I2C	I2CxSDA	I/O	Data input / output pin
	I2xCSCL	I/O	Clock input / output pin
SSP	SPxDO	Output	Data output pin
	SPxDI	Input	Data input pin
	SPxCLK	I/O	Clock input / output pin
	SPxFSS	I/O	Frame / slave select input pin
Analog digital convertor	AINx	Input	Analog input pin
	$\overline{\text{ADTRG}}$	Input	External trigger input pin
Real time clock	RTCOUT	Output	1Hz clock output pin

1.4.1.2 Debug function

Table 1-2 Pin name and functions

Pin name	Input or Output	Function
TMS	Input	JTAG test mode select input pin
TCK	Input	JTAG serial data input pin
TDO	Output	JTAG serial data output pin
TDI	Input	JTAG test reset input pin
$\overline{\text{TRST}}$	Input	Serial wire reset input pin
SWDIO	I/O	Serial wire data input / output pin
SWCLK	Input	Serial wire clock input pin
SWV	Output	Serial wire viewer output pin
TRACECLK	Output	Trace clock output pin
TRACEDATA0	Output	Trace data output pin 0
TRACEDATA1	Output	Trace data output pin 1
TRACEDATA2	Output	Trace data output pin 2
TRACEDATA3	Output	Trace data output pin 3

1.4.1.3 Control function

Table 1-3 Pin name and functions

Pin name	Input or Output	Function
X1	Input	High frequency resonator connection pin
X2	Output	High frequency resonator connection pin
XT1	Input	Low frequency resonator connection pin
XT2	Output	Low frequency resonator connection pin
MODE	Input	MODE pin This pin must be fixed to Low level.
TEST	Input	TEST pin This pin must be open.
$\overline{\text{RESET}}$	Input	Reset signal input pin
$\overline{\text{BOOT}}$	Input	BOOT mode control pin BOOT mode control pin is sampled at the rising edge of reset signal input pin. TMPM46BF10FG changes Single Boot Mode when BOOT mode control pin is "Low". TMPM46BF10FG changes Single Chip Mode when BOOT mode control pin is "High". Refer to "Flash memory" section for a detail.

1.4.1.4 Power supply pins

Table 1-4 Pin name and functions

Power supply pin name	Function
REGIN1	Pin connected with the capacitor (1.0μF) for the regulator
REGOUT1	Pin connected with the capacitor (1.0μF) for the regulator
REGIN2	Pin connected with the capacitor (1.0μF) for the regulator
REGOUT2	Pin connected with the capacitor (1.0μF) for the regulator
REGOUT3	Pin connected with the capacitor (1.0μF) for the regulator
RVDD3	Power supply pin for the regulator
RVSS	GND pin for the regulator
DVDD3	Power supply pin for the digital circuit DVDD3 supplies the following pins. PA to PH, PK, PL, X1, X2, XT1, XT2, MODE, TEST, $\overline{\text{RESET}}$, $\overline{\text{BOOT}}$
DVSS	GND pin for the digital circuit
AVDD3	Power supply pin for the analog circuit AVDD3 supplies the following pins. PJ
AVSS	GND pin for ADC

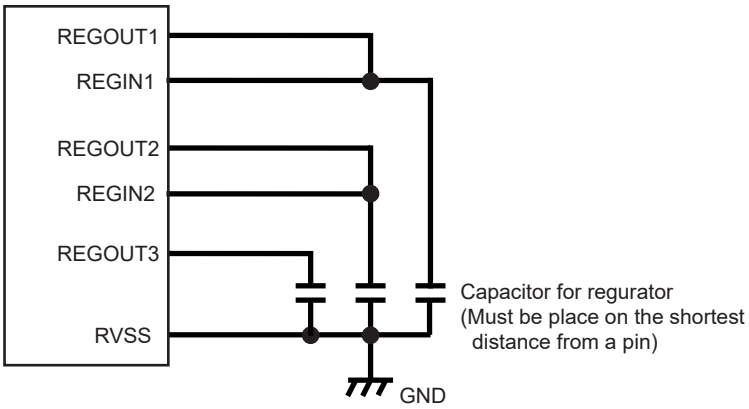


Figure 1-3 Capacitor for a regulator connection circuit

1.4.2 Pin names and Function of TMPM46BF10FG

1.4.2.1 The detail for pin names and function list

The mean of the symbol in the table is shown bellow.

1. Function A

   The function which is specified without setting of function register is shown in this cell.

2. Function B

   The function which is specified with setting of function register is shown in this cell. The number in this cell is corresponded with the number of function register.

3. Pin specification

   The mean of the symbol in the table is shown bellow.

   - SMT/CMOS : Type of input gate

     SMT : Schmitt input

     CMOS : CMOS input

   - 5V_T : 5V tolerant support

     Yes : supported

     N/A : Not supported

   - OD : Programmable open drain output support

     Yes : supported

     N/A : Not supported

   - PU/PD : Programmable Pull-Up / Pull-Down

     PU : Programmable Pull-Up supported

     PD : Programmable Pull-Down supported

1.4.2.2 PORT / Debug pin

Table 1-5 Pin names and functions <Sorted by PORT>

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/PD	OD	5V_T	SMT/ CMOS
PORTA												
98	PA0		TDO/SWV	UT1DTR					PU	Yes	N/A	SMT
97	PA1		TMS/SWDIO	UT1DSR					PU	Yes	N/A	SMT
96	PA2		TCK/SWCLK	UT1RIN					PD	Yes	N/A	SMT
95	PA3	INT3	TDI	UT1DCD					PU	Yes	N/A	SMT
94	PA4		$\overline{TRST}$	$\overline{UT1RTS}$					PU	Yes	N/A	SMT
93	PA5		TRACECLK	UT1RXD	UT1IRIN				PU	Yes	N/A	SMT
92	PA6		TRACEDATA0	UT1TXD	UT1IROUT				PU	Yes	N/A	SMT
91	PA7		TRACEDATA1	$\overline{UT1CTS}$	SC3SCK	$\overline{SC3CTS}$	TB7OUT		PU	Yes	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTB												
88	PB0		TRACEDATA2		SC3TXD				PU	Yes	N/A	SMT
87	PB1		TRACEDATA3		SC3RXD				PU	Yes	N/A	SMT
43	PB2		$\overline{WR}$		MT3OUT0	MT3TBOUT	$\overline{SNFCWE}$		PU	Yes	N/A	SMT
44	PB3		$\overline{RD}$		MT3OUT1	MT3TBIN	$\overline{SNFCRE}$		PU	Yes	N/A	SMT
45	PB4		$\overline{CS0}$		$\overline{GEMG3}$		SNFCCLE		PU	Yes	N/A	SMT
46	PB5		ALE		MT3IN		SNFCALE		PU	Yes	N/A	SMT
47	PB6	$\overline{BOOT}$	$\overline{BELL}$	SCOUT		TB3OUT	$\overline{SNFCCE}$		PU	Yes	N/A	SMT
48	PB7						SNFCRB		PU	Yes	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTC												
77	PC0	INTE							PU	Yes	N/A	SMT
76	PC1	INTF							PU	Yes	N/A	SMT
75	PC2		TB3IN0						PU	Yes	N/A	SMT
74	PC3		TB4IN0						PU	Yes	N/A	SMT
73	PC4	INT1	TB6IN0						PU	Yes	N/A	SMT
72	PC5		TB7IN0	RTCOUT					PU	Yes	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTD												
57	PD0		SP2FSS						PU	Yes	N/A	SMT
56	PD1		SP2DI						PU	Yes	N/A	SMT
55	PD2		SP2DO						PU	Yes	N/A	SMT
54	PD3		SP2CLK						PU	Yes	N/A	SMT
53	PD4	INT7	TB5IN0						PU	Yes	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTE												
13	PE0	INT4			A16		TB0IN0		PU	Yes	N/A	SMT
14	PE1	INT5	SC0RXD		A17		TB1IN0		PU	Yes	N/A	SMT
15	PE2		SC0TXD		A18		TB1OUT		PU	Yes	N/A	SMT
16	PE3		SC0SCK		A19	$\overline{\text{SC0CTS}}$	TB0OUT		PU	Yes	N/A	SMT
17	PE4		SC1SCK		A20	$\overline{\text{SC1CTS}}$	TB2OUT		PU	Yes	N/A	SMT
18	PE5		SC1TXD		A21				PU	Yes	N/A	SMT
19	PE6		SC1RXD		A22				PU	Yes	N/A	SMT
20	PE7	INT6			A23		TB2IN0		PU	Yes	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTF												
23	PF0		AD0		$\overline{\text{UT0CTS}}$				PU	Yes	N/A	SMT
24	PF1		AD1		UT0TXD	UT0IROUT			PU	Yes	N/A	SMT
25	PF2		AD2		UT0RXD	UT0IRIN			PU	Yes	N/A	SMT
26	PF3		AD3		$\overline{\text{UT0RTS}}$		SP1CLK		PU	Yes	N/A	SMT
27	PF4	INT0	AD4		UT0DCD		SP1DO		PU	Yes	N/A	SMT
28	PF5		AD5		UT0RIN		SP1DI		PU	Yes	N/A	SMT
29	PF6		AD6		UT0DSR	I2C1SCL	SP1FSS		PU	Yes	N/A	SMT
30	PF7		AD7		UT0DTR	I2C1SDA			PU	Yes	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTG												
31	PG0		AD8		MT0IN		SNFCD0		PU	Yes	N/A	SMT
32	PG1		AD9		$\overline{\text{GEMG0}}$		SNFCD1		PU	Yes	N/A	SMT
33	PG2		AD10		MT0OUT1	MT0TBIN	SNFCD2		PU	Yes	N/A	SMT
34	PG3		AD11		MT0OUT0	MT0TBOUT	SNFCD3		PU	Yes	N/A	SMT
35	PG4		AD12				SNFCD4		PU	Yes	N/A	SMT
36	PG5		AD13				SNFCD5		PU	Yes	N/A	SMT
37	PG6		AD14				SNFCD6		PU	Yes	N/A	SMT
38	PG7		AD15				SNFCD7		PU	Yes	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTH												
52	PH0		$\overline{BELH}$	TB5OUT	MT2IN	I2C2SDA			PU	N/A	N/A	SMT
51	PH1		$\overline{CS1}$	TB4OUT	$\overline{GEMG2}$	I2C2SCL			PU	N/A	N/A	SMT
50	PH2		$\overline{CS2}$		MT2OUT1	MT2TBIN			PU	N/A	N/A	SMT
49	PH3		$\overline{CS3}$		MT2OUT0	MT2TBOUT			PU	N/A	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTJ												
1	PJ0	AIN0							PU	N/A	N/A	SMT
2	PJ1	AIN1							PU	N/A	N/A	SMT
3	PJ2	AIN2							PU	N/A	N/A	SMT
4	PJ3	AIN3 INT9							PU	N/A	N/A	SMT
5	PJ4	AIN4 INTA							PU	N/A	N/A	SMT
6	PJ5	AIN5 INTB							PU	N/A	N/A	SMT
7	PJ6	AIN6 INTC							PU	N/A	N/A	SMT
8	PJ7	AIN7	$\overline{DMAREQ0}$	$\overline{DMAREQ1}$	$\overline{DMAREQ2}$				PU	N/A	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTK												
82	PK0	INTD							PU	Yes	N/A	SMT
81	PK1	INT8		SP0FSS		TB6OUT			PU	Yes	N/A	SMT
80	PK2			SP0DI	I2C0SDA				PU	Yes	N/A	SMT
79	PK3			SP0DO	I2C0SCL				PU	Yes	N/A	SMT
78	PK4			SP0CLK					PU	Yes	N/A	SMT

Pin No.	PORT	Function A	Function B						Port Specification			
			1	2	3	4	5	6	PU/ PD	OD	5V_T	SMT/ CMOS
PORTL												
83	PL0	INT2			MT1IN	$\overline{ADTRG}$			PU	Yes	N/A	SMT
84	PL1				$\overline{GEMG1}$		SC2RXD		PU	Yes	N/A	SMT
85	PL2				MT1OUT1	MT1TBIN	SC2TXD		PU	Yes	N/A	SMT
86	PL3				MT1OUT0	MT1TBOUT	SC2SCK	$\overline{SC2CTS}$	PU	Yes	N/A	SMT

1.4.2.3 Control pin

Table 1-6 The number of pin and pin names

Pin No.	Contorol function Pin name	Pin No.	Control function Pin name
59	X1	62	MODE
61	X2	40	TEST
69	XT1	58	$\overline{\text{RESET}}$
70	XT2	47	$\overline{\text{BOOT}}$

1.4.2.4 Power Supply pin

Table 1-7 The number of pin and pin names

Pin No.	Power supply Pin name
63	REGIN1
64	REGOUT1
12	REGIN2
11	REGOUT2
67	REGOUT3
10, 65	RVDD3
9, 66	RVSS
22, 42, 68, 89	DVDD3
21, 39, 41, 60, 71, 90	DVSS
99	AVDD3
100	AVSS

2. Product Information

This chapter describes peripheral function-related channels or number of units, information of pins and product-specific function information. Use this chapter in conjunction with Chapter Peripheral Function.

- "2.1.1 DMA Controller (DMAC)"
- "2.1.2 External Bus Interface (EBIF)"
- "2.1.3 SLC NAND Flash controller (SNFC)"
- "2.1.4 16-bit Timer/Event Counter (TMRB)"
- "2.1.5 16-bit Multi-Purpose Timer (MPT)"
- "2.1.6 Serial Channel (SIO/UART)"
- "2.1.7 Universal Asynchronous Serial Communication Circuit (UART)"
- "2.1.8 I2C Bus (I2C)"
- "2.1.9 Synchronous Serial Interface (SSP)"
- "2.1.10 Analog/Digital Converter (ADC)"
- "2.1.11 Flash Memory (FLASH)"
- "2.1.12 Debug Interface"

2.1 Information of Each Peripheral Function

2.1.1 DMA Controller (DMAC)

TMPM46BF10FG incorporates 3 units of built-in DMA controller.

Table 2-1 Pin Specifications

Unit	DMAREQx
DMAA	PJ7
DMAB	
DMAC	

Note: to use DMAC function, refer to "Bus Matrix " of "Memory Map" chapter and "DMA Controller(DMAC)" chapter .

Table 2-2 DMA Request Table

Channel	Unit A		Unit B		Unit C	
	Burst	Single	Burst	Single	Burst	Single
0	SNFC_PRD11(Note1)	–	SNFC_GIE1(Note1)	–	SNFC_RD1(Note1)	–
1	SNFC_PRD12(Note1)	–	SNFC_GIE2(Note1)	–	SNFC_RD2(Note1)	–
2	SNFC_PRD21(Note1)	–	SNFC_GIE3(Note1)	–	SNFC_RD3(Note1)	–
3	SNFC_PRD22(Note1)	–	SNFC_GIE4(Note1)	–	SNFC_RD4(Note1)	–
4	ADC conversion completion	–	SNFC_GIE5(Note1)	–	SNFC_RD5(Note1)	–
5	UART0 reception	UART0 reception	SNFC_GIE6(Note1)	–	SNFC_RD6(Note1)	–
6	UART0 transmission	UART0 transmission	SNFC_GIE7(Note1)	–	SNFC_RD7(Note1)	–
7	UART1 reception	UART1 reception	SNFC_GIE8(Note1)	–	SNFC_RD8(Note1)	–
8	UART1 transmission	UART1 transmission	SNFC_GID11(Note1)	–	AES read	–
9	SIO/UART0 reception	–	SNFC_GID12(Note1)	–	AES write	–
10	SIO/UART0 transmission	–	SNFC_GID13(Note1)	–	SHA write	–
11	SIO/UART1 reception	–	SNFC_GID14(Note1)	–	DMA transfer completion (ch10 of Unit C)	–
12	SIO/UART1 transmission	–	SNFC_GID15(Note1)	–	I2C0 transmission/reception	–
13	SIO/UART2 reception	–	SNFC_GID16(Note1)	–	I2C1 transmission/reception	–
14	SIO/UART2 transmission	–	SNFC_GID17(Note1)	–	I2C2 transmission/reception	–
15	SIO/UART3 reception	–	SNFC_GID18(Note1)	–	MPT0 compare match 0	–
16	SIO/UART3 transmission	–	SNFC_GID21(Note1)	–	MPT0 compare match 1	–
17	TMRB0 compare match (Note2)	–	SNFC_GID22(Note1)	–	MPT1 compare match 0	–
18	TMRB1 compare match (Note2)	–	SNFC_GID23(Note1)	–	MPT1 compare match 1	–
19	TMRB2 compare match (Note2)	–	SNFC_GID24(Note1)	–	MPT2 compare match 0	–
20	TMRB3 compare match (Note2)	–	SNFC_GID25(Note1)	–	MPT2 compare match 1	–
21	TMRB4 compare match (Note2)	–	SNFC_GID26(Note1)	–	MPT3 compare match 0	–
22	TMRB5 compare match (Note2)	–	SNFC_GID27(Note1)	–	MPT3 compare match 1	–

Table 2-2 DMA Request Table

23	TMRB6 compare match (Note2)	–	SNFC_GID28(Note1)	–	TMRB3 input capture 0	–
24	TMRB7 compare match (Note2)	–	ADC conversion completion	–	TMRB3 input capture 1	–
25	TMRB0 input capture 0	–	SSP0 reception	SSP0 reception	TMRB4 input capture 0	–
26	TMRB0 input capture 1	–	SSP0 transmission	SSP0 transmission	TMRB4 input capture 1	–
27	TMRB1 input capture 0	–	SSP1 reception	SSP1 reception	TMRB5 input capture 0	–
28	TMRB1 input capture 1	–	SSP1 transmission	SSP1 transmission	TMRB5 input capture 1	–
29	TMRB2 input capture 0	–	SSP2 reception	SSP2 reception	TMRB6 input capture 0	–
30	TMRB2 input capture 1	–	SSP2 transmission	SSP2 transmission	TMRB6 input capture 1	–
31	$\overline{\text{DMAREQA}}$	–	$\overline{\text{DMAREQB}}$	–	$\overline{\text{DMAREQC}}$	–

Note 1: See the chapter "SNFC" for details..

Note 2: A DMA transfer request of TMRB occurs under the same conditions as a TMRB interrupt. A TMRB interrupt occurs when the up-counter matches timer register 0/1, or the up-counter overflows. Mask unnecessary factors with interrupt mask register (TBxIM) if required.

2.1.2 External Bus Interface (EBIF)

TMPM46BF10FG incorporates the external interface function and the multiplex bus modes can be used.

Table 2-3 Pin specifications

Multiplex bus	Port
A16 to A23	PE0 to PE7
AD0 to AD7	PF0 to PF7
AD8 to AD15	PG0 to PG7
ALE	PB5
$\overline{RD}$	$\overline{PB3}$
$\overline{WR}$	PB2
$\overline{BELL}$	PB6
$\overline{BELH}$	PH0
$\overline{CS0}$	PB4
$\overline{CS1}$	PH1
$\overline{CS2}$	PH2
$\overline{CS3}$	PH3

2.1.3 SLC NAND Flash controller (SNFC)

TMPM46BF10FG incorporates 1 unit of SNFC.

Table 2-4 Pin specifications

Function	Port
$\overline{SNFCCE}$	PB6
SNFCCLE	PB4
SNFCALE	PB5
$\overline{SNFCRE}$	PB3
$\overline{SNFCWE}$	PB2
SNFCD[7:0]	PG7 to PG0
SNFCRB	PB7

2.1.4 16-bit Timer/Event Counter (TMRB)

TMPM46BF10FG incorporates 8 channels of TMRB.

Table 2-5 Pin specifications

Channel	TBxOUT	TBxIN0
TMRB0	PE3	PE0
TMRB1	PE2	PE1
TMRB2	PE4	PE7
TMRB3	PB6	PC2
TMRB4	PH1	PC3
TMRB5	PH0	PD4
TMRB6	PK1	PC4
TMRB7	PA7	PC5

Table 2-6 Synchronous start specifications

Master channel	Slave channel
TMRB0	TMRB1, TMRB2, TMRB3
TMRB4	TMRB5, TMRB6, TMRB7

Table 2-7 Capture trigger specifications

Trigger input channel	Trigger output
TMRB0 TMRB1 TMRB2	TB7OUT
TMRB3 TMRB4 TMRB5	TB2OUT
TMRB6 TMRB7	TB5OUT

2.1.5 16-bit Multi-Purpose Timer (MPT)

TMPM46BF10FG contains 4 channels of MPT.

Table 2-8 Pin specifications

Channel	MTxTBOUT MTxOUT0	MTxTBIN MTxOUT1	$\overline{\text{GEMGx}}$	MTxIN
MPT0	PG3	PG2	PG1	PG0
MPT1	PL3	PL2	PL1	PL0
MPT2	PH3	PH2	PH1	PH0
MPT3	PB2	PB3	PB4	PB5

Table 2-9 Synchronous start/clear specifications

Master channel	Slave channel
MPT0	MPT1, MPT2, MPT3

2.1.6 Serial Channel (SIO/UART)

TMPM46BF10FG incorporates 4 channels of SIO.

Table 2-10 Pin specifications

Channel	$\overline{\text{SCxCTS}}$ SCxSCK	SCxTXD	SCxRXD
SC0	PE3	PE2	PE1
SC1	PE4	PE5	PE6
SC2	PL3	PL2	PL1
SC3	PA7	PB0	PB1

Table 2-11 Transfer clock specifications

Clock input channel	Clock output
SC0 SC1	TB4OUT
SC2 SC3	TB7OUT

2.1.7 Universal Asynchronous Serial Communication Circuit (UART)

TMPM46BF10FG incorporates 2 channels of UART.

Table 2-12 Pin specifications

Channel	UTxDTR	UTxDSR	UTxRIN	UTxDCD	$\overline{\text{UTxRTS}}$	UTxRXD UTxIRIN	UTxTXD UTxIROUT	$\overline{\text{UTxCTS}}$
UART0	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
UART1	PA0	PA1	PA2	PA3	PA4	PA5	PA6	PA7

2.1.8 I2C Bus (I2C)

TMPM46BF10FG incorporates 3 channels of I2C.

Table 2-13 Pin specifications

Channel	I2CxSDA	I2CxSCL
I2C0	PK2	PK3
I2C1	PF7	PF6
I2C2	PH0	PH1

2.1.9 Synchronous Serial Interface (SSP)

TMPM46BF10FG incorporates 3 channels of SSP.

Table 2-14 Pin specifications

Channel	SPxCLK	SPxDO	SPxDI	SPxFSS
SSP0	PK4	PK3	PK2	PK1
SSP1	PF3	PF4	PF5	PF6
SSP2	PD3	PD2	PD1	PD0

Table 2-15 SPxCLK cycle

Channel	Master (min.)	Slave (min.)
SSP0	50ns	150ns
SSP1 SSP2	100ns	300ns

2.1.10 Analog/Digital Converter (ADC)

TMPM46BF10FG incorporates 1 unit of ADC.

Table 2-16 Pin specifications

Unit	AIN0 to 7	$\overline{\text{ADTRG}}$
ADC	PJ0 to 7	PL0

Table 2-17 Internal startup trigger selection (ADILVTRGSEL)

Type	Bit Symbol	Internal trigger
Highest priority AD conversion startup trigger	HPTRGSEL[3:0]	0000 : $\overline{\text{TRG0}}$ (A match with TB2RG1 of TMRB2) 0001 : $\overline{\text{TRG1}}$ (A match with TB3RG1 of TMRB3) 0010 : $\overline{\text{TRG2}}$ (A match with TB4RG1 of TMRB4) 00 11: $\overline{\text{TRG3}}$ (A match with TB5RG1 of TMRB5) 0100 : $\overline{\text{TRG4}}$ (A match with TB6RG1 of TMRB6) 0101 : $\overline{\text{TRG5}}$ (A match with TB7RG1 of TMRB7) 0110 : $\overline{\text{TRG6}}$ (A match with MT0IGTRG of MPT0 (IGBT)) 0111 : $\overline{\text{TRG7}}$ (A match with MT1IGTRG of MPT1 (IGBT)) 1000 : $\overline{\text{TRG8}}$ (A match with MT2IGTRG of MPT2 (IGBT)) 1001 : $\overline{\text{TRG9}}$ (A match with MT3IGTRG of MPT3 (IGBT)) 1010 to 1111: Reserved
Normal AD conversion start-up trigger	TRGSEL[3:0]	0000 : $\overline{\text{TRG0}}$ (A match with TB2RG1 of TMRB2) 0001 : $\overline{\text{TRG1}}$ (A match with TB3RG1 of TMRB3) 0010 : $\overline{\text{TRG2}}$ (A match with TB4RG1 of TMRB4) 00 11: $\overline{\text{TRG3}}$ (A match with TB5RG1 of TMRB5) 0100 : $\overline{\text{TRG4}}$ (A match with TB6RG1 of TMRB6) 0101 : $\overline{\text{TRG5}}$ (A match with TB7RG1 of TMRB7) 0110 : $\overline{\text{TRG6}}$ (A match with MT0IGTRG of MPT0 (IGBT)) 0111 : $\overline{\text{TRG7}}$ (A match with MT1IGTRG of MPT1 (IGBT)) 1000 : $\overline{\text{TRG8}}$ (A match with MT2IGTRG of MPT2 (IGBT)) 1001 : $\overline{\text{TRG9}}$ (A match with MT3IGTRG of MPT3 (IGBT)) 1010 to 1111: Reserved

2.1.11 Flash Memory (FLASH)

TMPM46BF10FG incorporates 1 unit of FLASH.

Table 2-18 Pin connection

Pin		Port
Mode setting pin	$\overline{BOOT}$	PB6
Reset pin	$\overline{RESET}$	-
Communicationpin	TXD0	PE2
	RXD0	PE1

2.1.12 Debug Interface

TMPM46BF10FG supports serial wire debug ports, JTAG debug ports and trace outputs.

Table 2-19 Pin specifications

	TMS SWDIO	TCK SWCLK	TDO SWV	TDI	$\overline{TRST}$
JTAG Serial wire	PA1	PA2	PA0	PA3	PA4

	TRACECLK	TRACEDATA0	TRACEDATA1	TRACEDATA2	TRACEDATA3
Trace output	PA5	PA6	PA7	PB0	PB1

2.2 Usage Note for TMPM46BF10FG

1.Cautions in case DMAC function is unused in TMPM46BF10FG

It should be set the all DMAC units as DMAxCfg =0x00000001,

DMAxChnlReqMaskSet =0xFFFFFFFF, DMAxChnlEnableSet =0xFFFFFFFF.

(x=A,B,C)

3. Processor Core

The TX04 series has a high-performance 32-bit processor core (the Arm Cortex-M4 processor with FPU core). For information on the operations of this processor core, please refer to the documentation set issued by Arm Limited. This chapter describes the functions unique to the TX04 series that are not explained in that document.

3.1 Information on the processor core

The following table shows the revision of the processor core in the TMPM46BF10FG.

Refer to the detailed information about the CPU core and architecture, refer to the Arm documentation set for "the Cortex-M4 series processors" in the following URL:

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

Product Name	Core Revision
TMPM46BF10FG	r0p1

3.2 Configurable Options

The Cortex-M4 processor with FPU core has optional blocks.

The following tables shows the configurable options in the TMPM46BF10FG.

Configurable options	Implementation
MPU (Memory Protection Unit)	Absent
FPB (Flash Patch and Breakpoint)	Two literal comparators Six instruction comparators
DWT (Data Watchpoint and Trace)	Four comparators
ITM (Instrumentation Trace Macrocell)	Present
ETM (Embedded Trace Macrocell)	Present
AHB-AP (AHB Access Port)	Present
HTM Interface (AHB Trace Macrocell Interface)	Absent
TPIU (Trace Port Interface Unit)	Present
WIC (Wake-up Interrupt Controller)	Absent
Debug Port (Serial-Wire or JTAG Debug Port)	Present
FPU (Floating Point Unit)	Present
Bit banding	Present
Constant AHB control	Disable

3.3 Exceptions/ Interruptions

Exceptions and interruptions are described in the following section.

3.3.1 Number of Interrupt Inputs

The number of interrupt inputs can optionally be defined from 1 to 240 in the Cortex-M4 processor with FPU core.

TMPM46BF10FG has 103 interrupt inputs. The number of interrupt inputs is reflected in <INTLINESNUM [3:0]> bit of NVIC register. In this product, if read <INTLINESNUM[3:0]> bit, "0x03" is read out.

3.3.2 Number of Priority Level Interrupt Bits

The Cortex-M4 processor with FPU core can optionally configure the number of priority level interrupt bits from 3 bits to 8 bits.

TMPM46BF10FG 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.

3.3.3 SysTick

The Cortex-M4 processor with FPU 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.

3.3.4 SYSRESETREQ

The Cortex-M4 processor with FPU core outputs SYSRESETREQ signal when <SYSRESETREQ> bit of Application Interrupt and Reset Control Register are set.

TMPM46BF10FG provides as same as warm reset operation when SYSRESETREQ signal are output.

3.3.5 LOCKUP

When irreparable exception generates, the Cortex-M4 processor with FPU core outputs LOCKUP signal to show a serious error included in software.

TMPM46BF10FG does not use this signal. To return from LOCKUP status, it is necessary to use non-maskable interrupt (NMI) or reset.

3.3.6 Auxiliary Fault Status register

The Cortex-M4 processor with FPU core provides auxiliary fault status registers to supply additional system fault information to software.

However, TMPM46BF10FG is not defined this function. If auxiliary fault status register is read, always "0x0000_0000" is read out.

3.4 Events

The Cortex-M4 processor with FPU core has event output signals and event input signals. An event output signal is output by SEV instruction execution. If an event is input, the core returns from low-power consumption mode caused by WFE instruction.

TMPM46BF10FG does not use event output signals and event input signals. Please do not use SEV instruction and WFE instruction.

3.5 Power Management

The Cortex-M4 processor with FPU core provides power management system which uses SLEEPING signal and SLEEPDEEP signal. 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 Register is set.

TMPM46BF10FG 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."

3.6 Exclusive access

In Cortex-M4 processor with FPU core, the DCode bus system supports exclusive access. However TMPM46BF10FG does not use this function.

3.7 Floating Point Unit (FPU)

This product implements the Cortex-M4 processor with FPU that is the single precision variant of the ARMv7-M Floating point Extension (FPv4-SP). It provides floating-point computation functionality that is compliant with the ANSI/IEEE Std 754-2008, IEEE Standard for Binary Floating-Point Arithmetic, referred to as the IEEE 754 standard.

The FPU shares address bus and data bus with Cortex-M4 processor with FPU core, and operates in cooperation. It performs add, sub-tract, and multiply in 1 clock, and multiply and accumulate in 3 clocks. The parallel processing that is different from CPU is possible with using the exclusive data register.

The FPU supports all single-precision data-processing instructions and data types described in the Arm Architecture Reference Manual.

4. Memory Map

4.1 Memory Map

The memory maps for TMPM46BF10FG are based on the Arm Cortex-M4 processor with FPU core memory map. The internal ROM, internal RAM and special function registers (SFR) of TMPM46BF10FG are mapped to the Code, SRAM and peripheral regions of the Cortex-M4 processor with FPU respectively. The special function register (SFR) means the control registers of all input/output ports and peripheral functions.

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

For more information on each region, see the "Arm documentation set for the Arm Cortex-M4".

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

A memory map of TMPM46BF10FG is shown bellows.

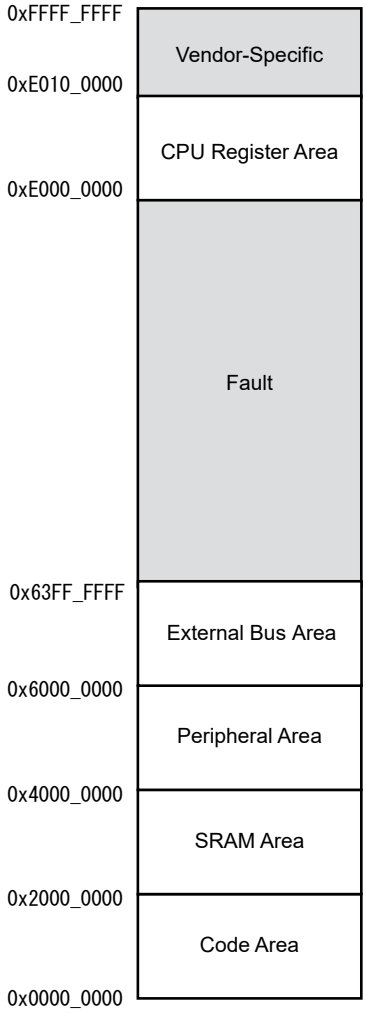


Figure 4-1 MemoryMap

4.2 Bus Matrix

This MCU contains two bus masters such as a CPU core and µDMA controllers .

Bus masters connect to slave ports (S0 to S5) of Bus Matrix. In the bus matrix, master ports (M0 to M15) connect to peripheral functions via connections described as (o) or (•) in the following figure. (•) shows a connection to a mirror area.

While multiple slaves are connected on the same bus master line in the Bus Matrix, if multiple slave accesses are generated at the same time, a priority is given to access from a master with the smallest slave number.

4.2.1 Structure

4.2.1.1 Single chip mode

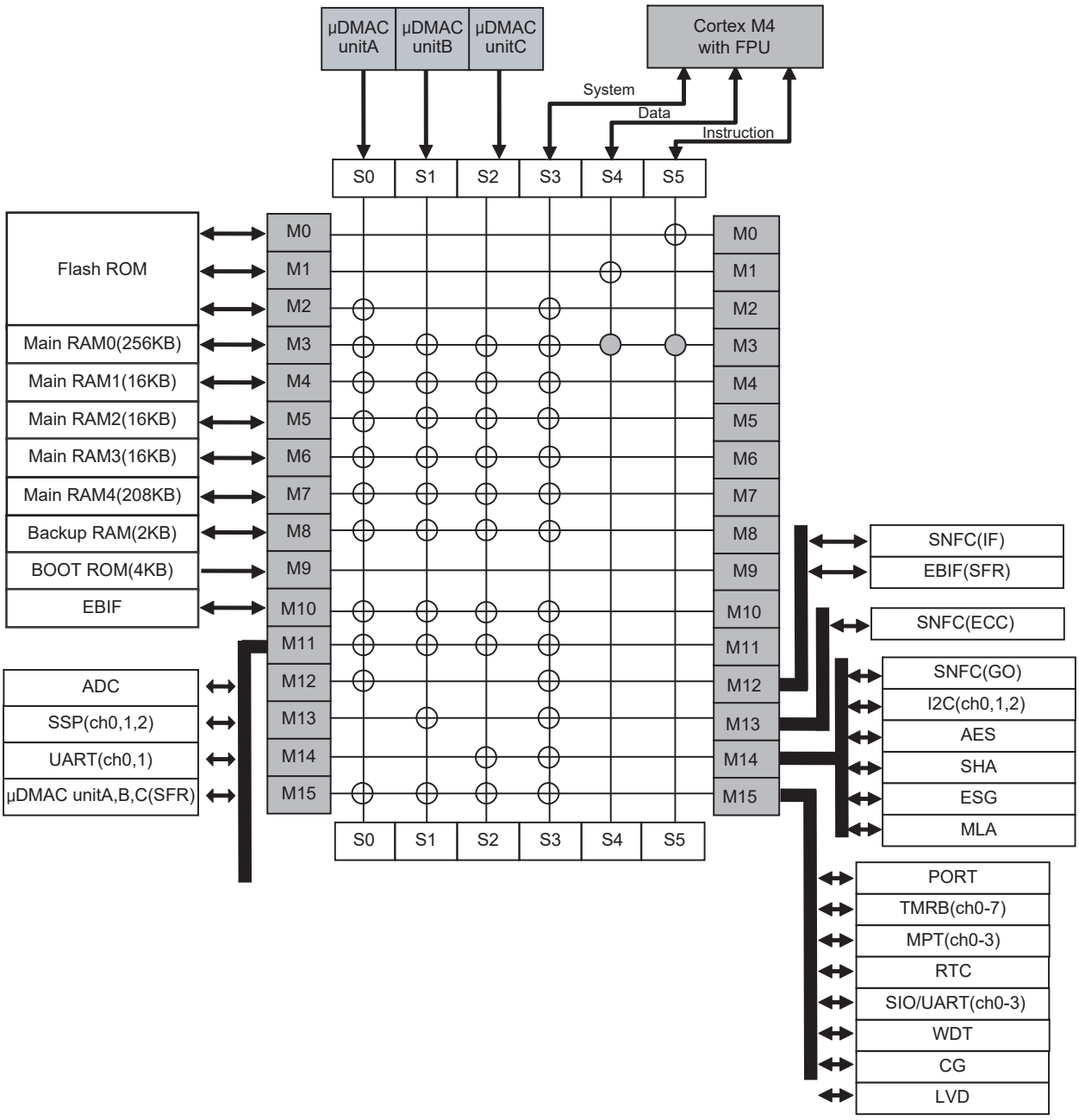


Figure 4-2 Bus Matrix of TMPM46BF10FG

4.2.1.2 Single boot mode

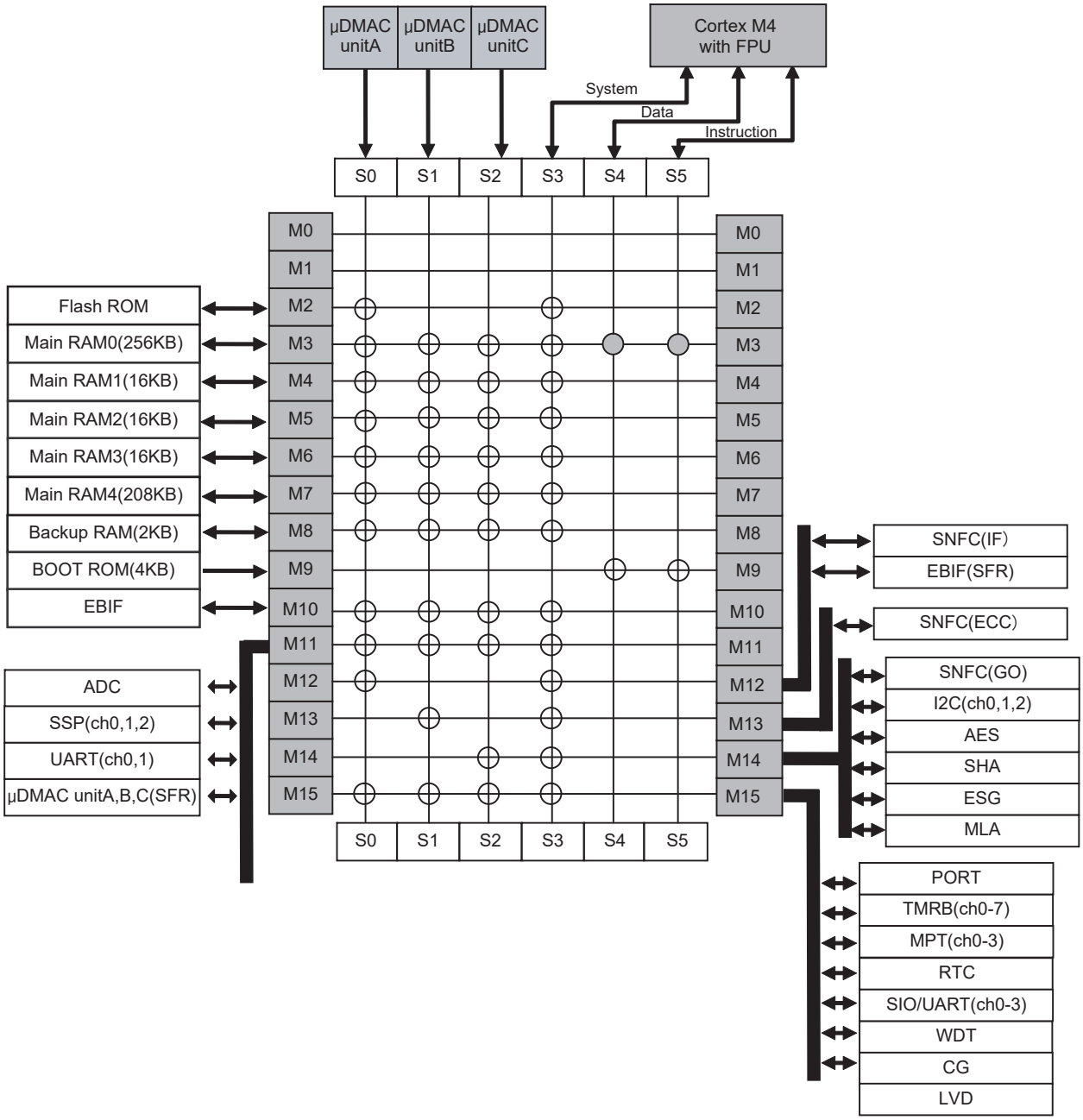


Figure 4-3 Bus Matrix of TMPM46BF10FG

4.2.2 Connection table

4.2.2.1 Code area / SRAM area

(1) Single chip mode

Start Address	Slave	Master	µDMAC unit A	µDMAC unit B	µDMAC unit C	Core S-Bus	Core D-Bus	Core I-Bus
			S0	S1	S2	S3	S4	S5
0x0000_0000	Flash ROM	M0 M1	Fault	Fault	Fault	Fault	M1	M0
0x0010_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x1000_0000	RAM0 (mirror)	M3	Fault	Fault	Fault	Fault	o	o
0x1004_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x2000_0000	Main RAM0	M3	o	o	o	o	Fault	Fault
0x2004_0000	Main RAM1	M4	o	o	o	o	Fault	Fault
0x2004_4000	Main RAM2	M5	o	o	o	o	Fault	Fault
0x2004_8000	Main RAM3	M6	o	o	o	o	Fault	Fault
0x2004_C000	Main RAM4	M7	o	o	o	o	Fault	Fault
0x2008_0000	Backup RAM	M8	o	o	o	o	Fault	Fault
0x2008_0800	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x2200_0000	Bit band alias	-	Fault	Fault	Fault	o	Fault	Fault
0x2301_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault

(2) Single boot mode

Start Address	Slave	Master	µDMAC unit A	µDMAC unit B	µDMAC unit C	Core S-Bus	Core D-Bus	Core I-Bus
			S0	S1	S2	S3	S4	S5
0x0000_0000	Boot ROM	M9	Fault	Fault	Fault	Fault	o	o
0x0000_1000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x1000_0000	Main RAM0 (mirror)	M3	Fault	Fault	Fault	Fault	o	o
0x1004_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x2000_0000	Main RAM0	M3	o	o	o	o	Fault	Fault
0x2004_0000	Main RAM1	M4	o	o	o	o	Fault	Fault
0x2004_4000	Main RAM2	M5	o	o	o	o	Fault	Fault
0x2004_8000	Main RAM3	M6	o	o	o	o	Fault	Fault
0x2004_C000	Main RAM4	M7	o	o	o	o	Fault	Fault
0x2008_0000	Backup RAM	M8	o	o	o	o	Fault	Fault
0x2008_0800	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x2200_0000	Bit band alias	-	Fault	Fault	Fault	o	Fault	Fault
0x2301_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x3F7F_F000	Reserved	-	Fault	Fault	Fault	Reserved	Fault	Fault
0x3F80_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault

Note: Please do not access the address range given in Reserved.

4.2.2.2 Peripheral area / External bus area

Start Address	Slave	Master	µDMAC unit A	µDMAC unit B	µDMAC unit C	Core S-Bus	Core D-Bus	Core I-Bus
			S0	S1	S2	S3	S4	S5
0x4000_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x4004_0000	SSP(ch0-2)	M11	o	o	o	o	Fault	Fault
0x4004_8000	UART(ch0-1)		o	o	o	o	Fault	Fault
0x4004_C000	µDMAC unitA(SFR)		o	o	o	o	Fault	Fault
0x4004_D000	µDMAC unitB(SFR)		o	o	o	o	Fault	Fault
0x4004_E000	µDMAC unitC(SFR)		o	o	o	o	Fault	Fault
0x4005_0000	ADC		o	o	o	o	Fault	Fault
0x4005_C000	EBIF(SFR)	M12	o	-	-	o	Fault	Fault
0x4005_C400	SNFC(IF)		o	-	-	o	Fault	Fault
0x4005_C800	SNFC(ECC)	M13	-	o	-	o	Fault	Fault
0x4005_CC00	SNFC(GO)	M14	-	-	o	o	Fault	Fault
0x4005_F000	DMACR		-	-	o	o	Fault	Fault
0x4006_6000	ADILV		-	-	o	o	Fault	Fault
0x400A_0000	I2C(ch0-2)		-	-	o	o	Fault	Fault
0x400B_8200	AES		-	-	o	o	Fault	Fault
0x400B_8300	SHA		-	-	o	o	Fault	Fault
0x400B_8400	ESG		-	-	o	o	Fault	Fault
0x400B_8500	SRST		-	-	o	o	Fault	Fault
0x400B_9000	MLA		-	-	o	o	Fault	Fault
0x400C_0000	PORT	M15	o	o	o	o	Fault	Fault
0x400C_4000	TMRB		o	o	o	o	Fault	Fault
0x400C_7000	MPT		o	o	o	o	Fault	Fault
0x400C_C000	RTC		o	o	o	o	Fault	Fault
0x400E_1000	SIO/UART		o	o	o	o	Fault	Fault
0x400F_2000	WDT		o	o	o	o	Fault	Fault
0x400F_3000	CG		o	o	o	o	Fault	Fault
0x400F_4000	LVD		o	o	o	o	Fault	Fault
0x4010_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x4280_0000	Bit band alias	-	Fault	Fault	Fault	o	Fault	Fault
0x4400_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x5DFF_0000	Flash(SFR)	M15	o	o	o	o	Fault	Fault
0x5E00_0000	Flash(Mirror)	M2	o	Fault	Fault	o	Fault	Fault
0x5E10_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault
0x6000_0000	EBIF	M10	o	o	o	o	Fault	Fault
0x6400_0000	Fault	-	Fault	Fault	Fault	Fault	Fault	Fault

4.2.3 Address lists of peripheral functions

Do not access to addresses in the peripheral area except control registers. For details of control registers, refer to Chapter of each peripheral functions.

Peripheral Function		Base Address
Synchronous Serial Port (SSP)	ch0	0x4004_0000
	ch1	0x4004_1000
	ch2	0x4004_2000
Asynchronous Serial Channel (UART)	ch0	0x4004_8000
	ch1	0x4004_9000
µDMA Controller (µDMAC)	unitA	0x4004_C000
	unitB	0x4004_D000
	unitC	0x4004_E000
	DMAIF	0x4005_F000
Analog / Digital Converter (ADC)	AD	0x4005_0000
	ADILV	0x4006_6000
External bus interface (EBIF)		0x4005_C000
SLC NAND Flash controller (SNFC)	SNFC_IF	0x4005_C400
	SNFC_ECC	0x4005_C800
	SNFC_GO	0x4005_CC00
I2C Bus interface (I2C)	ch0	0x400A_0000
	ch1	0x400A_1000
	ch2	0x400A_2000
Advanced Encryption standard (AES)		0x400B_8200
Secure Hash Algorithm (SHA)		0x400B_8300
Entropy Seed Generator (ESG)		0x400B_8400
IP Soft Reset (SRST)		0x400B_8500
Multiple Length Arithmetic (MLA)		0x400B_9000
Input / Output port	Port A	0x400C_0000
	Port B	0x400C_0100
	Port C	0x400C_0200
	Port D	0x400C_0300
	Port E	0x400C_0400
	Port F	0x400C_0500
	Port G	0x400C_0600
	Port H	0x400C_0700
	Port J	0x400C_0800
	Port K	0x400C_0900
	Port L	0x400C_0A00
16-bit Timer / Event Counters (TMRB)	ch0	0x400C_4000
	ch1	0x400C_4100
	ch2	0x400C_4200
	ch3	0x400C_4300
	ch4	0x400C_4400
	ch5	0x400C_4500
	ch6	0x400C_4600
	ch7	0x400C_4700

Peripheral Function		Base Address
16-bit Multi-Purpose Timer (MPT)	ch0	0x400C_7000
	ch1	0x400C_7100
	ch2	0x400C_7200
	ch3	0x400C_7300
Real Time Clock (RTC)		0x400C_C000
Serial Channel (SIO/UART)	ch0	0x400E_1000
	ch1	0x400E_1100
	ch2	0x400E_1200
	ch3	0x400E_1300
Watchdog Timer(WDT)		0x400F_2000
Clock/Mode control(CG)		0x400F_3000
Low Voltage Detection Circuit (LVD)		0x400F_4000
Flash Control(Flash SFR)		0x5DFF_0000

5. Reset Operation

The following are sources of reset operation.

- RESET pin ($\overline{\text{RESET}}$)
- Releasing the STOP2 mode
- Low voltage detection circuit (LVD)
- Watchdog timer (WDT)
- Application interrupt by CPU and a signal from the reset register bit <SYSRESETREQ>

To recognize a source of reset, check CGRSTFG in the clock generator register described in chapter of "Exception".

A reset by releasing the STOP2 mode is refer to the "Clock/Mode control".

A reset by low voltage detection circuit is refer to the "Low voltage detection circuit".

A reset by WDT is refer to the chapter on the "Watchdog timer".

A reset by <SYSRESETREQ> is referred to "Cortex-M4 Technical Reference Manual".

Note: Once reset operation is done, internal RAM data is not assured.

5.1 Cold Reset

When turning-on power, $\overline{\text{RESET}}$ pin must be kept "Low".

When turning-on power, it is necessary to take a stable time of built-in regulator into consideration. In the TMPM46BF10FG, the internal regulator requires at least approximately 2ms to be stable. At cold reset, $\overline{\text{RESET}}$ pin must be kept "Low" for a duration of time sufficiently long enough for the internal regulator to be stable. Approximately 0.251ms after $\overline{\text{RESET}}$ pin becomes "High", internal reset will be released.

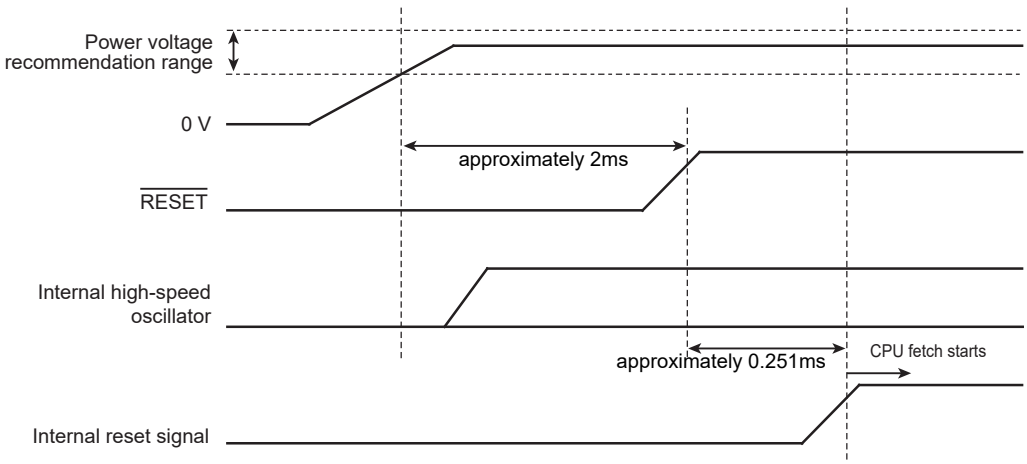


Figure 5-1 Cold Reset Operation Sequence

Note: The above sequence is applied as well when restoring power.

5.2 Warm Reset

To do reset TMPM46BF10FG, the following conditions are required; power supply voltage is in the operational range ; $\overline{\text{RESET}}$ pin is kept "Low" at least for 2ms. Approximately 0.251ms after $\overline{\text{RESET}}$ pin becomes "High", internal reset will be released.

In case of WDT reset or <SYSRESETREQ> reset, internal reset will be released approximately 30 internal high-speed clocks after reset.

5.3 After reset

All of the control register of the internal core and the peripheral function control register (SFR) are initialized by reset.

When reset is released, TMPM46BF10FG starts operation by a clock of internal high-speed oscillator. External clock and PLL multiple circuit should be set if necessary.

6. Clock/Mode control

6.1 Outline

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

6.2 Registers

6.2.1 Register List

The following table shows the Clock/Mode control related registers and addresses.

For the Base Address, refer to the "Peripheral Base address list "of Chapter "Memory Map".

Register name		Address (Base+)
System control register	CGSYSCR	0x0000
Oscillation control register	CGOSCCR	0x0004
Standby control register	CGSTBYCR	0x0008
PLL selection register	CGPLLSEL	0x000C
Clock stop register A for peripheral	CGFSYSMSKA	0x0020
Clock stop register B for peripheral	CGFSYSMSKB	0x0024
Protect register	CGPROTECT	0x003C

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	-	-	-	FCSTOP	-	-	SCOSEL	
After reset	0	0	0	0	0	0	0	1
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	FPSEL	-	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
31-21	–	R	Read as "0".
20	FCSTOP	R/W	ADC clock 0: Active 1: Stop Enables to stop providing ADC clock. ADC clock is provided after reset. Confirming that ADC is stopped or finished in advance is required when setting "1"(stop). Enables to confirm that ADC is stopped or finished.
19-18	–	R	Read as "0".
17-16	SCOSEL	R/W	SCOUT out 00: fs 01: fsys/8 10: fsys/4 11: fosc Enables to output the specified clock from SCOUT pin.
15-13	–	R	Read as "0".
12	FPSEL	–	fperiph source clock 0: fgear 1: fc Specifies the source clock to fperiph. Selecting fc fixes fperiph regardless of the clock gear mode.
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 circuit.
7-3	–	R	Read as "0".
2-0	GEAR[2:0]	R/W	High-speed clock (fc) gear 000: fc 100: fc/2 001: Reserved 101: fc/4 010: Reserved 110: fc/8 011: Reserved 111: fc/16

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				WUPTL		WUPSEL2	WUPSEL1
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	WUEF	WUEON	-	-	-	HOSCON	OSCF	OSCSEL
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	DRVOSCL	-	-	-	XTEN	-	XEN2	XEN1
After reset	0	0	0	0	1	0	1	0

Bit	Bit Symbol	Type	Function
31-20	WUPT[11:0]	R/W	Warm-up timer setup value. Setup the 16-bit timer for warm-up timer of upper 12-bit counter value.
19-18	WUPTL[1:0]	R/W	Warm-up timer setup value. Setup the 16-bit timer for warm-up timer of lower 2-bit counter value, This is used for low-speed clock.
17	WUPSEL2	R/W	Select High-Speed warm-up clock 0: Internal OSC 1: External OSC Select the OSC clock for warm-up timer. A warm-up timer is counting by the selected clock.
16	WUPSEL1	R/W	Select warm-up timer 0: High-speed 1: Low-speed
15	WUEF	R	Status of warm-up timer 0: WUP finish 1: WUP active Can be monitored the status of the warm-up timer.
14	WUEON	W	Operation of warm-up timer 0: don't care 1: warm-up timer start Enables to start the warm-up timer. Read as "0".
13	–	R	Read as "0".
12	–	R/W	Write as "0".
11	–	R	Read as "0".
10	HOSCON	R/W	Select external OSC source 0: external clock input 1: external oscillator
9	OSCF	R	Status of Selected High-speed oscillator 0: internal high-speed oscillator 1: external high-speed oscillator
8	OSCSEL	R/W	High-speed oscillator (Note6) 0: internal high-speed oscillator 1: external high-speed oscillator
7	DRVOSCL	R/W	Drive current control for low-speed oscillator (Note7)(Note8) 0: High 1: Normal

Bit	Bit Symbol	Type	Function
6-4	–	R	Read as "0".
3	XTEN	R/W	External low-speed oscillator operation 0: Stop 1: Oscillation
2	–	R	Read as "0".
1	XEN2	R/W	Internal high-speed oscillator operation (for SYS) (Note9) 0: Stop 1: Oscillation
0	XEN1	R/W	External OSC (Note10) 0: Not used 1: Used

Note 1: Refer to 6.6.8.1 about the Warm-up setup.

Note 2: When selecting external oscillator (input external clock), select <OSCSEL> after setting <HOSCON>. (Do not select simultaneously)

Note 3: Refer to "6.3.5 Clock Multiplication Circuit (PLL)" about setting PLL.

Note 4: Returning from the STOP1/STOP2 mode, related bits CGOSCCR<WUPSEL2>,<WUPSEL1>, <OSCSEL>, <XEN3>, <XEN2>, <XEN1> of the register CGOSCCR and CGPLLSEL<PLLSEL>,<PLLON> are initialized because of internal high-speed oscillator starts up.

Note 5: When using internal high-speed oscillator (IHOSC), do not use it as system clock which high accuracy assurance is required.

Note 6: When changed the <OSCSEL>, next operation will be done after confirming the <OSCF> which is changed correctly.

Note 7: This bit is initialized to "0" by Reset, So Set to "1" only after confirming stable oscillation

Note 8: If the MCU did not enter the STOP1/2 mode, CGOSCCR<DRVOSCL> may be initialized.

Note 9: When disable the Internal OSC after changed from STOP2 to Normal mode, confirm the CGRTSFLG<OSCFLG> which is "1".

Note 10: Set to "1" only when using External High-speed clock (oscillator /clock input)

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	-
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	-	-	-	-	-	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	Keeps I/O control signal in STOP2 mode 0: Control by port 1: Keep status when setting 0->1 (This register must be set before entering STOP2 mode)
16-3	–	R	Read as "0".
2-0	STBY[2:0]	R/W	Low power consumption mode 000: Reserved 001: STOP1 010: Reserved 011: IDLE 100: Reserved 101: STOP2 110: Reserved 111: Reserved

Note: Access to the Reserved is prohibited.

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	-	-	-	-	-	PLLST	PLLSEL	PLLON
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	PLLSET							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PLLSET							-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-19	–	R	Read as "0".
18	PLLST	R	Status of selected clock in PLL 0: f_{osc} 1: f_{PLL}
17	PLLSEL	R/W	Use of PLL 0: f_{osc} use 1: f_{PLL} use
16	PLLON	R/W	PLL (multiplying circuit) operation 0: Stop 1: Oscillation
15-1	PLLSET[14:0]	R/W	PLL multiplying value (Do not use except below) 0x692E: Input clock 8MHz to 10MHz, output clock 96MHz to 120MHz (12 multiplying) 0x6A26: Input clock 8MHz to 12MHz, output clock 80MHz to 120MHz (10 multiplying) 0x6A1E: Input clock 10MHz to 12MHz, output clock 80MHz to 96MHz (8 multiplying) 0x6917: Input clock 8MHz to 10MHz, output clock 48MHz to 60MHz (6 multiplying) 0x6A13: Input clock 8MHz to 12MHz, output clock 40MHz to 60MHz (5 multiplying) 0x6A0F: Input clock 8MHz to 16MHz, output clock 32MHz to 64MHz (4 multiplying) 0x6A0A: Input clock 30MHz to 40MHz, output clock 90MHz to 120MHz (3 multiplying) 0x6A06: Input clock 30MHz to 40MHz, output clock 60MHz to 80MHz (2 multiplying)
0	–	R	Read as "0".

Note 1: Select PLL multiplying value which is shown in Table 6-2.

Note 2: Refer to "6.3.5 Clock Multiplication Circuit (PLL)" about setting PLL.

Note 3: Returning from the STOP1/STOP2 mode, related bits CGOSCCR<WUPSEL2>,<WUPSEL1>, <OSCSEL>, <XEN2>, <XEN1>, and CGPLLSEL<PLLON>,<PLLSEL> are initialized because of internal high-speed oscillator starts up.

Note 4: When changed the <PLLSEL>, Next operation will be done after confirming the <PLLSEL> which is changed correctly

6.2.6 CGFSYSMSKA (Clock stop register A for peripheral)

	31	30	29	28	27	26	25	24
bit symbol	TRACECLK	MPT3	MPT2	MPT1	MPT0	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	TMRB7	TMRB6	TMRB5	TMRB4	TMRB3
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TMRB2	TMRB1	TMRB0	-	-	PORTL	PORK	PORTJ
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PORTH	PORTG	PORTF	PORTE	PORTD	PORTC	PORTB	PORTA
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	TARCECLK	R/W	Clock control for TRACE 0: operation 1: clock stop
30	MPT3	R/W	Clock control for MPT3 0: operation 1: clock stop
29	MPT2	R/W	Clock control for MPT2 0: operation 1: clock stop
28	MPT1	R/W	Clock control for MPT1 0: operation 1: clock stop
27	MPT0	R/W	Clock control for MPT0 0: operation 1: clock stop
26-21	-	R	Read as "0".
20	TMRB7	R/W	Clock control for TMRB7 0: operation 1: clock stop
19	TMRB6	R/W	Clock control for TMRB6 0: operation 1: clock stop
18	TMRB5	R/W	Clock control for TMRB5 0: operation 1: clock stop
17	TMRB4	R/W	Clock control for TMRB4 0: operation 1: clock stop
16	TMRB3	R/W	Clock control for TMRB3 0: operation 1: clock stop
15	TMRB2	R/W	Clock control for TMRB2 0: operation 1: clock stop
14	TMRB1	R/W	Clock control for TMRB1 0: operation 1: clock stop

Bit	Bit Symbol	Type	Function
13	TMRB0	R/W	Clock control for TMRB0 0: operation 1: clock stop
12-11	-	R	Read as "0".
10	PORTL	R/W	Clock control for PORT L 0: operation 1: clock stop
9	PORTK	R/W	Clock control for PORT K 0: operation 1: clock stop
8	PORTJ	R/W	Clock control for PORT J 0: operation 1: clock stop
7	PORTH	R/W	Clock control for PORT H 0: operation 1: clock stop
6	PORTG	R/W	Clock control for PORT G 0: operation 1: clock stop
5	PORTF	R/W	Clock control for PORT F 0: operation 1: clock stop
4	PORTE	R/W	Clock control for PORT E 0: operation 1: clock stop
3	PORTD	R/W	Clock control for PORT D 0: operation 1: clock stop
2	PORTC	R/W	Clock control for PORT C 0: operation 1: clock stop
1	PORTB	R/W	Clock control for PORT B 0: operation 1: clock stop
0	PORTA	R/W	Clock control for PORT A 0: operation 1: clock stop

Note:Set register to "1" after confirming stop of a Peripheral circuit operation. after setting to "1",do not set to "0".

6.2.7 CGFSYSMSKB (Clock stop register B for peripheral)

	31	30	29	28	27	26	25	24
bit symbol	MLA	ESG	SHA	AES	-	WDT	ADC	DMAIF
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	DMAC	DMAB	DMAA	EBIF	SSP2	SSP1	SSP0	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	I2C2	I2C1	I2C0	UART1	UART0	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	SIO3	SIO2	SIO1	SIO0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	MLA	R/W	Clock control for MLA 0: operation 1: clock stop
30	ESG	R/W	Clock control for ESG 0: operation 1: clock stop
29	SHA	R/W	Clock control for SHA 0: operation 1: clock stop
28	AES	R/W	Clock control for AES 0: operation 1: clock stop
27	-	R	read as "0".
26	WDT	R/W	Clock control for WDT 0: operation 1: clock stop
25	ADC	R/W	Clock control for ADC 0: operation 1: clock stop
24	DMAIF	R/W	Clock control for DMAIF 0: operation 1: clock stop
23	DMAC	R/W	Clock control for DMAC C 0: operation 1: clock stop
22	DMAB	R/W	Clock control for DMAC B 0: operation 1: clock stop
21	DMAA	R/W	Clock control for DMAC A 0: operation 1: clock stop
20	EBIF	R/W	Clock control for EBIF 0: operation 1: clock stop
19	SSP2	R/W	Clock control for SSP2 0: operation 1: clock stop

Bit	Bit Symbol	Type	Function
18	SSP1	R/W	Clock control for SSP1 0: operation 1: clock stop
17	SSP0	R/W	Clock control for SSP0 0: operation 1: clock stop
16-15	-	R	read as "0".
14	I2C2	R/W	Clock control for I2C2 0: operation 1: clock stop
13	I2C1	R/W	Clock control for I2C1 0: operation 1: clock stop
12	I2C0	R/W	Clock control for I2C0 0: operation 1: clock stop
11	UART1	R/W	Clock control for UART1 0: operation 1: clock stop
10	UART0	R/W	Clock control for UART0 0: operation 1: clock stop
9-4	-	R	read as "0".
3	SIO3	R/W	Clock control for SIO/UART3 0: operation 1: clock stop
2	SIO2	R/W	Clock control for SIO/UART2 0: operation 1: clock stop
1	SIO1	R/W	Clock control for SIO/UART1 0: operation 1: clock stop
0	SIO0	R/W	Clock control for SIO/UART0 0: operation 1: clock stop

Note:Set register to "1" after confirming stop of a circuit operation. after setting to "1",do not set to "0".

6.2.8 CGPROTECT (Protect 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	CGPROTECT							
After reset	1	1	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-6	–	R	Read as "0".
7-0	CGPROTECT	R/W	Register protection control 0xC1: Register write enable Except 0xC1: Register write disable Initial value is "0xC1" as writing enable to each register and when writing except "0xC1", each register except CGPROTECT register can not be written.

6.3 Clock control

6.3.1 Clock Type

Each clock is defined as follows:

EHCLKIN	: Clock input from the X1 pins.
EHOSC	: Clock input from the external high-speed oscillator.
ELOSC	: Clock input from the external Low-speed oscillator.
IHOSC	: Clock input from the internal high-speed oscillator.(for SYS)
FOSCHI	: Clock specified by CGOSCCR<HOSCON>.
fosc	: Clock specified by CGOSCCR<OSCSEL>.
fpll	: Clock multiplied by PLL.
fc	: Clock specified by CGPLLSEL<PLLSEL> (high-speed clock)
fgear	: Clock specified by CGSYSCR<GEAR[2:0]>
fsys	: Clock specified by CGSYSCR<GEAR[2:0]>.(system clock)
fperiph	: Clock specified by CGSYSCR<FPSEL[2:0]>
φT0	: Clock specified by CGSYSCR<PRCK[2:0]> (prescaler clock)

The high-speed clock fc and the prescaler clock φT0 are dividable as follows.

High-speed clock	: fc, fc/2, fc/4, fc/8, fc/16
Prescaler clock	: 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.

internal high-speed oscillator	: oscillating
external high-speed oscillator	: stop
external 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 to be the same as fosc.

For example, system clock fsys is as same as the frequency of an internal high-speed oscillator after releasing reset signal when an internal oscillator is started.

fc = fosc
fsys = fosc
φT0 = fosc

6.3.3 Clock system Diagram

Figure 6-1 shows the clock system diagram.

The input clocks to selector shown with an arrow are set as default after reset.

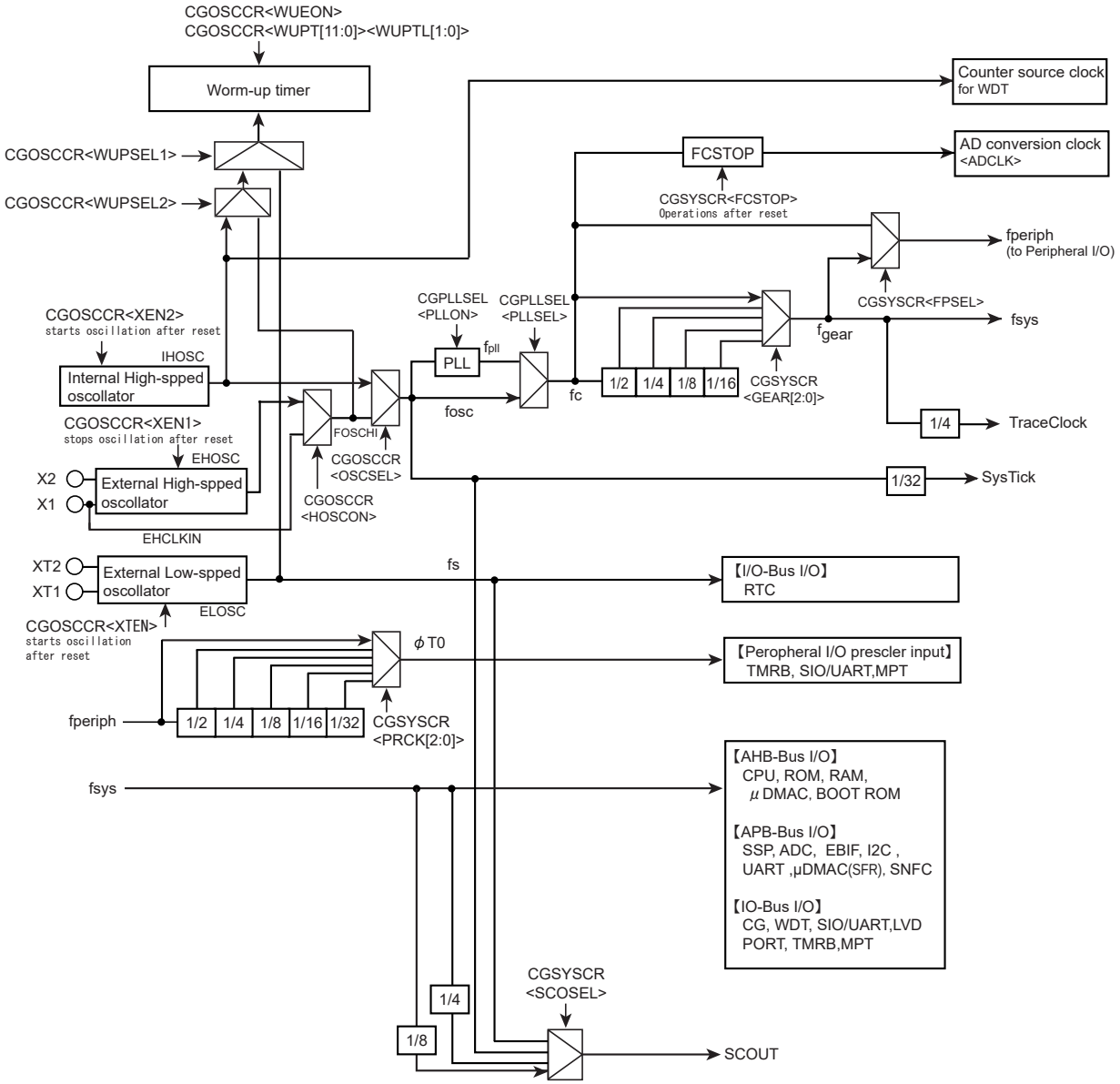


Figure 6-1 Clock Block Diagram

6.3.4 Warm-up function

The warm-up function secures the stability time for the oscillator of fs and the PLL with the warm-up timer when releasing STOP1 and STOP2. Refer to "6.6.7 Warm-up" for a detail.

Note: Transition to the power consumption mode while the warm-up timer is operating is prohibited.

How to configure the warm-up function.

1. Specify the count up clock

   Specify the count up clock for the warm-up timer in the CGOSCCR<WUPSEL2>, <WUPSEL1>.

2. Specify the warm-up timer value

   The warm-up time can be selected by setting the CGOSCCR<WUPT[11:0]><WUPTL[1:0]>. The value can be calculated by following formula with round lower 4 bit off, set to the bit of <WUPT[11:0]> for high-speed oscillation and set to the bit of <WUPT[11:0]><WUPTL[1:0]> for low-speed oscillation.

Note: Setting warm-up timer value to CGOSCCR<WUPT[11:0]>, wait until this value is reflected, then transit to the low power consumption mode by executing a command "WFI"

Warm-up time equation and setup example are shown below.

$$\text{number of warm-up cycle} = \frac{\text{warm-up time to set}}{\text{input frequency cycle (s)}}$$

Note: 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.

<example> When using high-speed oscillator 8MHz, and set warm-up time 5ms.

$$\frac{\text{warm-up time to set}}{\text{input frequency cycle (s)}} = \frac{5\text{ms}}{1/8\text{MHz}} = 40000 \text{ cycle} = 0\text{x}9\text{C}40$$

Round lower 4 bit off, set 0x9C4 to CGOSCCR<WUPT[11:0]>

3. confirm the start and completion of warm-up

   The CGOSCCR<WUEON><WUEF> is used to confirm the start and completion of warm-up through software (instruction). When CGOSCCR<WUEON> is set to "1", the warm-up start a count up. The completion of warm-up can be confirmed with CGOSCCR<WUEF>.

The example of warm-up function setup.

Table 6-1 The example of warm-up setting

	CGOSCCR<WUODR[11:0]> = "0x9C4"	: Specify the warm-up time
↻	Read CGOSCCR<WUODR[11:0]>	: Confirm warm-up time reflecting Repeat until reading "0x9C4"
	CGOSCCR<XEN2> = "1"	: high-speed oscillator (fosc) enable
	CGOSCCR<WUEON> = "1"	: Start the warm-up timer (WUP)
↻	CGOSCCR<WUEF> read	: Wait until the state becomes "0" (warm-up is finished)

Note 1: The warm-up function is not necessary when using stable external clock.

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: Setting warm-up timer value to CGOSCCR<WUPT[11:0]><WUPTL[1:0]>, wait until this value is reflected, then transit to standby mode by executing a command "WFI".

Note 4: Returning from the STOP1/STOP2 mode, related bits <WUPSEL2>, <WUPSEL1>,<OSCSEL>, <XEN3>, <XEN2>, <XEN1> of the register CGOSCCR and CGPLLSEL<PLLSEL>,<PLLON> are initialized because of internal high-speed oscillator starts up.

6.3.5 Clock Multiplication Circuit (PLL)

This circuit outputs the fpll clock (120MHz max.) that is multiplied by 2, 3, 4, 5, 6, 8,10 or 12 of the high-speed oscillator output clock (fosc: 8 to 40MHz). As a result, the input frequency to oscillator can be low frequency, 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<PLLSET> register's PLL multiplying value after CGPLLSEL<PLLON> bit is set to "0", and then set CGPLLSEL<PLLON> bit set to "1". As CGPLLSEL<PLLSEL> bit is set to "1", the fpll clock which is multiplied from fosc is output.

6.3.5.2 Stability time

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

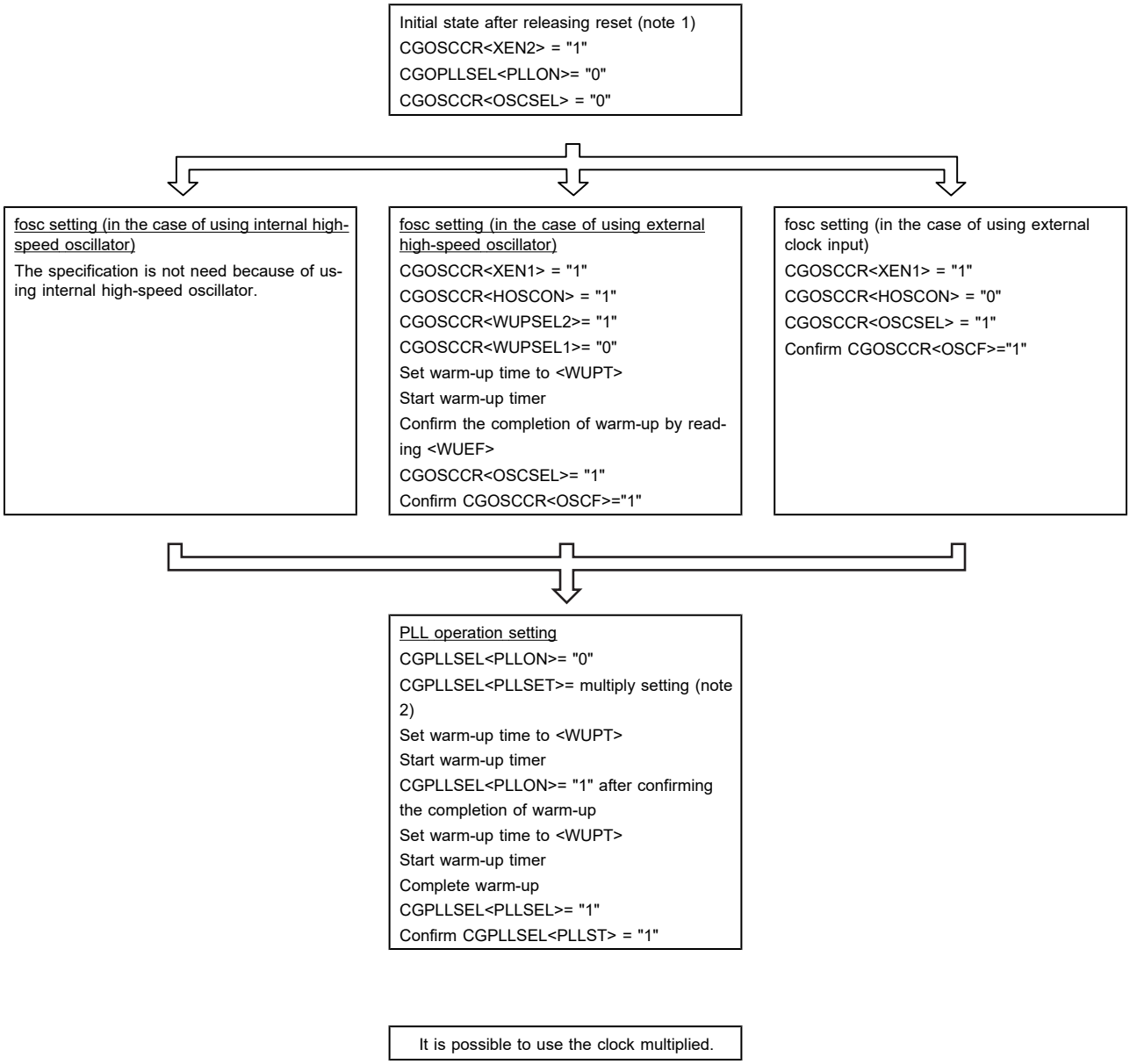
When the CGPLLSEL <PLLON> is set to "1" and operation starts, it is necessary to take approximately 100μs as the Lock-up time.

The CGPLLSEL<PLLSEL> is first made "0" when the multiplying value is changed and PLL is stopped. When the multiplying <PLLSEL> value is changed, the <PLLON> is set to "1" after approximately 100μs elapses as initialization time of PLL, and the state of PLL starts. Afterwards, please secure the Lock-up time.

6.3.5.3 The sequence of PLL setting

The sequence of PLL setting is shown below.

The sequence of PLL setting



Note 1: Internal high-speed oscillator and voltage supply need to be stable.

Note 2: When the multiplied value is changed, it is necessary to keep CGPLLSEL<PLLON> "0" for equal or more than 100μs as initialization time of PLL.

6.3.6 System clock

The internal high-speed oscillation clock and the external high-speed oscillation clocks which are an oscillator connecting or an inputting clock can be used as a source clock of the system clock.

Source clock		Frequency	Using PLL
Internal high-speed oscillation (f_{IHOSC})		10MHz	Not use, 2, 3, 4, 5, 6, 8,10 or 12 multiplying (note)
External high-speed oscillation	Oscillator (f_{EHOSC})	8 to 16MHz	
	Input clock ($f_{EHCLKIN}$)	8 to 40MHz	

Note: When PLL is used, fc frequency must be kept equal or less then 120MHz even if source clock contains errors.

The system clock can be divided by CGSYSCR<GEAR>. Although the setting can be changed while operating, the actual switching takes place after a slight delay.

Table 6-2 shows the example of the operation frequency by the setting of PLL and the clock gear.

Table 6-2 System clock (Unit: MHz, "-": Reserved)

External oscillator (MHz)	External clock input (MHz)	PLL multi-plying val-ue	Max oper-ation freq. (fc) (MHz)	ADC Max operation freq. (fc,fc/2, fc/4) (MHz)	Clock gear (CG) PLL = ON					Clock gear (CG) PLL = OFF				
					1/1	1/2	1/4	1/8	1/16	1/1	1/2	1/4	1/8	1/16
8	8	12	96	24	96	48	24	12	6	8	4	2	1	-
8	8	10	80	40	80	40	20	10	5					
8	8	6	48	24	48	24	12	6	3					
8	8	5	40	40	80	40	20	10	5					
8	8	4	32	32	32	16	8	4	2					
10	10	12	120	30	120	60	30	15	7.5	10	5	2.5	1.25	-
10	10	10	100	25	100	50	25	12.5	6.25					
10	10	8	80	40	80	40	20	10	5					
10	10	6	60	30	60	30	15	7.5	3.75					
10	10	5	50	25	50	25	12.5	6.25	3.13					
10	10	4	40	40	40	20	10	5	2.5					
12	12	10	120	30	120	60	30	15	7.5	12	6	3	1.5	-
12	12	8	96	24	96	48	24	12	6					
12	12	5	60	30	60	30	15	7.5	3.75					
12	12	4	48	24	48	24	12	6	3					
16	16	4	64	32	64	32	16	8	4	16	8	4	2	1
-	30	3	90	22.5	90	45	22.5	11.25	5.63	30	15	7.5	3.75	1.875
-	30	2	60	30	60	30	15	7.5	3.75					
-	39	3	117	39	117	58.5	29.25	14.63	7.31	39	19.5	9.75	4.88	2.44
-	39	2	78	39	78	39	19.5	9.75	4.88					
-	40	3	120	30	120	60	30	15	7.5	40	20	10	5	2.5
-	40	2	80	40	80	40	20	10	5					

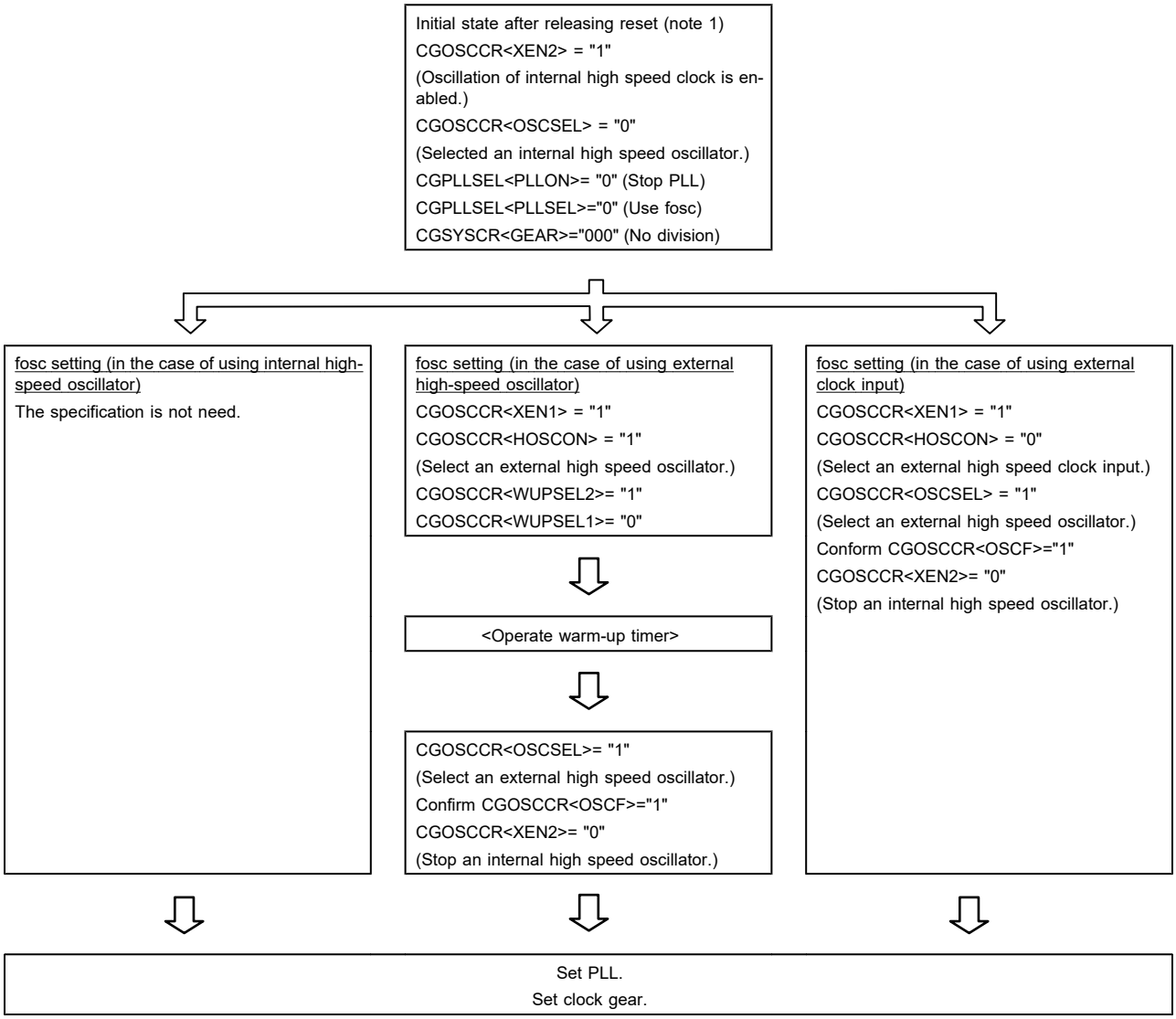
↑ initial value after reset

Note: Do not use 1/16 when SysTick is used.

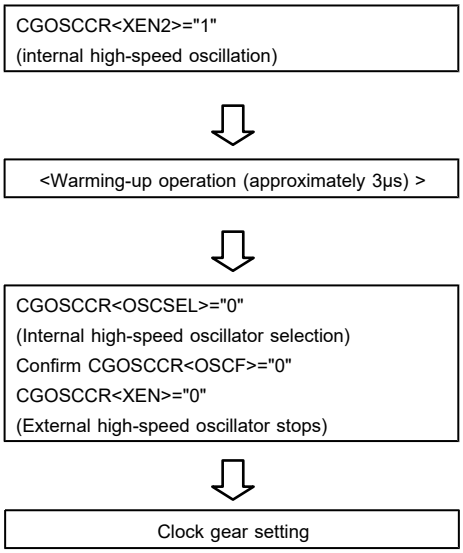
6.3.6.1 The sequence of System clock setting

The system clock is selected by CGOSCCR. After setting CGOSCCR, the PLL is set by CGPLLSEL and CGOSCCR and the clock gear is set by CGSYSCR.

The sequence of System clock setting (Initial setting)



The sequence of System clock setting (Switch from external to internal high-speed clock)



6.3.7 Prescaler Clock Control

Peripheral function has a prescaler for dividing a clock. As the clock φT0 to be input to each prescaler, the "fperiph" clock specified in the CGSYSCR<FPSEL> 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

TMPM46BF10FG enables to output the system clock from a pin. The SCOUT pin can output low speed clock fs and the system clock fsys/4 and fsys/8, and the fosc.

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 should be disabled when changing the clock gear.

When port is used as SCOUT, refer to chapter "Input/Output ports".

The output clock is selected by CGSYSCR<SCOSEL>.

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

Mode / SCOUT selection CGSYSCR	NORMAL	Low power consumption mode	
		IDLE	STOP1/STOP2
<SCOSEL> = "00"	Output the fs clock		
<SCOSEL> = "01"	Output the fsys/8 clock		
<SCOSEL> = "10"	Output the fsys/4 clock		
<SCOSEL> = "11"	Output the fosc clock		

6.4 Modes and Mode Transitions

6.4.1 Operation Mode Transitions

TMPM46BF10FG has the NORMAL mode which is used high-speed for system clock.

The IDLE and STOP1 modes can be used as the low power consumption mode that enables to reduce power consumption by halting processor core and some peripheral circuits operation.

And TMPM46BF10FG has STOP2 mode that enables to reduce power consumption significantly by halting main voltage supply, retaining some peripheral circuits operations.

Figure 6-2 shows a mode transition diagram.

For a detail of sleep-on-exit, refer to "Arm documentation set for the Arm cortex-M4."

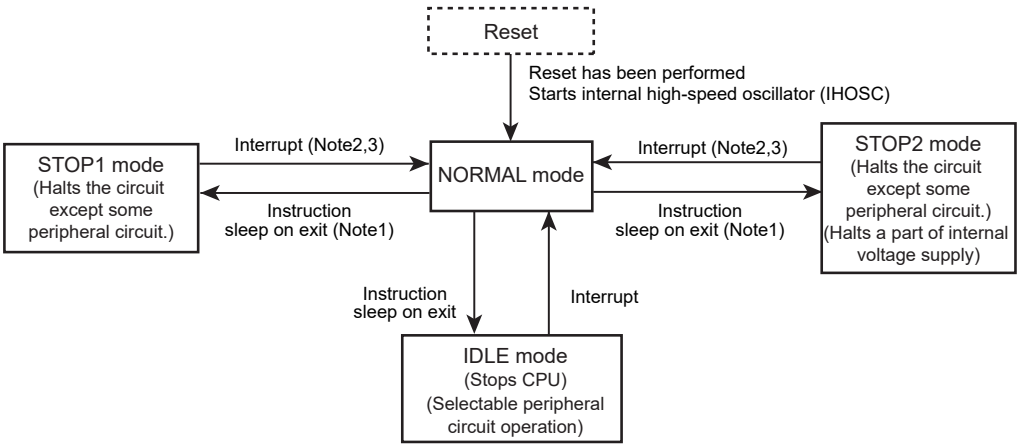


Figure 6-2 Mode Transition Diagram

Note 1: The warm-up is needed when returning from the STOP1/STOP2 mode.The warm-up time is needed to set in NORMAL mode, Regarding to warm-up time, refer to "6.6.8 Clock Operations in Mode Transition"

Note 2: Returning from the STOP1/STOP2 mode, related bits <WUPSEL2>, <WUPSEL1>,<OSCSEL>, <XEN3>, <XEN2>, <XEN1> of the register CGOSCCR and CGPLLSEL<PLLSEL>,<PLLON> are initialized because of internal high-speed oscillator starts up.

Note 3: It branches to interrupt service routine of reset when returning from the STOP2 mode and it branches to interrupt service routine of interrupt factor when returning from the STOP1 mode.

6.5 Operation mode

6.5.1 NORMAL mode

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

It is shifted to the NORMAL mode after reset.

6.6 Low Power Consumption Modes

The TMPM46BF10FG has three low power consumption modes: IDLE, STOP1 and STOP2. 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 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 TMPM46BF10FG 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 TMPM46BF10FG does not support the low power consumption mode configured with the SLEEPDEEP bit in the Cortex-M4 core. Setting the <SLEEPDEEP> bit of the system control register is prohibited.

Note 3: Transition to the power consumption mode while the warm-up timer is operating is prohibited.

The features of IDLE, STOP1, STOP2 mode are described as follows.

6.6.1 IDLE mode

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

Refer to "Table 6-6 Operational Status in Each Mode" for the peripheral circuits can be enabled or disabled in the IDLE mode. For setting details, see the chapter on each peripheral circuits.

6.6.2 STOP1 mode

Except some peripheral circuits, all the internal circuits including the internal oscillator are brought to a stop in STOP1 mode. When releasing STOP1 mode, an internal oscillator begins to operate and the operation mode changes to NORMAL mode.

The STOP1 mode enables to select the pin status by setting the port register. Table 6-4 shows the pin status in the STOP1 mode.

6.6.3 STOP2 mode

This mode halts voltage supply, retaining some peripheral circuits operation. This enables to reduce power consumption significantly compares to STOP1 mode.

Before entering STOP2 mode, set CGSTBYCR<PTKEEP>="0"→"1" and keeps each port conditions. If internal voltage is halted, it can be held interface to the external IC, and STOP2 release source interrupt is available. Because the signals are fixed, be careful not to affect the external IC when setting <PTKEEP>.

Before the MCU entering STOP2 mode, execute the WFI instruction on condition that interrupts are enabled.

When the WFI instruction is executed to enter STOP2 mode, if an interrupt request for release from the low-power consumption mode occurs, the MCU does not enter STOP2 mode because the interrupt for release has a higher priority than the WFI instruction. Therefore, the following process must be added:

1. The MCU branches to the interrupt service routine. Write the interrupt service routine.

2. After the interrupt service routine is complete, the instructions following the WFI instruction are executed; write the process following the WFI instruction in case that the MCU does not enter STOP2 mode.

After releasing the STOP2 mode, voltage is supplied to the halted voltage supply then an internal high-speed oscillator starts, returns to NORMAL mode, and branches interrupt service routine of reset. Use the Reset Flag (CGRSTFLG) Register of the Clock Generator to identify the source of a reset.

Note: The STOP2 mode should secure the period for 45 μs or more from mode transition to release in order to perform internal electrical power source interception. If it cancels within a period, the internal electrical power source management cannot operate normally.

Table 6-4 shows the pin status in the STOP2 mode.

Table 6-4 Pin States in the STOP1/STOP2 mode

Function	Function name	I/O	STOP1 (note1)	STOP2(note2) <PTKEEP> = 1
PORT	PAx to PLx	Input	Depend on PxIE[m]	keep state
		Output	Depend on PxCR[m]	keep state
Debug	$\overline{\text{TRST}}$, TCK, TMS, TDI, SWCLK, SWDIO	Input	Depend on PxIE[m]	keep state
	TDO, SWDIO, SWV, TRACECLK, TRACEDATA0/1/2/3	Output	Depend on PxCR[m] and enable when data is valid	keep state
Interrupt	INT0 to F	Input	Depend on PxIE[m]	keep state
SSP	SPxCLK, SPxFSS, SPxDO	Output	Depend on PxCR[m] and enable when data is valid	keep state
MPT	MTxOUT0,MTxOUT1	Output	Depend on PxCR[m] and enable when data is valid	keep state
except above	Except above	Input	Depend on PxIE[m]	keep state
	Except above	Output	Depend on PxCR[m]	keep state

(note1) x: port number / m: corresponding bit / n: function register number
(note2) Set <PTKEEP> to "1" before moving to STOP2 mode.

The flow chart of entering and releasing to or from STOP2 mode is shown bellows.
[ ] indicates hardware handling, [ ] indicates software handling.

Ready to entering to STOP2 mode
- CGSYSCR<GEAR> = "000" (Setting Gear = 1/1)
- WDT disable
- CGOSCCR<OSCSEL> = "0" (Setting IHOSC)
- Setting Warm-up timer with IHOSC
- CGSTBYCR<STBY> = "101" (Setting STOP2)
- CGPLLSEL<PLLSEL>= "0" (Used for fosc)
- Confirm CGPLLSEL<PLLST>="0"
- CGPLLSEL<PLLON> = "0" (Stop PLL for fsys)
- Interrupt enable

↓

Setting to keep port state
- CGSTBYCR<PTKEEP> = "1" (Keep the state when <PTKEEP> changes "0" to "1".)

↓

Enter to STOP2 mode.
- Execute WFI instruction

↓

During STOP2 mode
- The power supply of RVDD3 is shut-downed by stopping the main regulator.
- Port state is kept.

↓

The release factor is occurred.
- The interrupt is occurred and STOP2 mode is released.

↓

Start the sequence of releasing STOP2 mode
- The main regulator is started up
- Internal high-speed oscillator is started.
- Return to NORMAL mode.
- Reset operation is done.
- Branch to the reset interrupt routine.

↓

Check CGRSTFLG
- Read CGRSTFLG (Confirm the reset factor)

↓

Interrupt procedure
- Set interrupt enable register and interrupt priority register.
- Jump to an interrupt handler and confirm the release factor from STOP2 mode in the interrupt handler.

↓

Release the kept port state.
- Initial setting at the time of STOP2 return (or state restoration), and a re-setup of a port register
- CGSTBYCR<PTKEEP> = "0"

6.6.4 Low power Consumption Mode Setting

The low power consumption mode is specified by the setting of the standby control register 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]>
STOP1	001
IDLE	011
STOP2	101

Note: Do not set any value other than those shown above in <STBY[2:0]>.

6.6.5 Operational Status in Each Mode

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

Table 6-6 Operational Status in Each Mode

Block	NORMAL Internal high-speed oscillator use (IHOSC)	IDLE Internal high-speed oscillator use (IHOSC)	STOP1	STOP2
Processor core	o	–	–	×
DMAC	o	o	–	×
IO port	o	o	–(note 1)	–(note 2)
EBIF	o	o	–	–
ADC	o	o	–(note 3)	×(note 3)
SSP	o	o	–	×
SIO/UART	o	o	–	×
UART	o	o	–	×
I2C	o	o	–	×
WDT	o	o(note 4)	Δ–(note7)	×
TMRB	o	o	–	×
MPT	o	o	–	×
RTC	o	o	o	o(note 5 & 6)
LVD	o	o	o	Δ–
PLL	o	o	Δ–	Δ×
External high-speed oscillator (EHOSC)	o	o	–	Δ–
External low-speed oscillator (ELOSC)	o	o	o	o
Internal high-speed oscillator 1 (IHOSC)	o	o	–	–
Backup RAM	Accessible	Accessible	Keeping Data	Keeping Data
Main RAM				×
Flash ROM				×

o : Operation is available when in the target mode.
– : The clock to module stops automatically when transiting to the target mode.
Δ : Enables to select enabling or disabling module operation by software when in the target mode.
× :Voltage supply to module turns off automatically when transiting to the target mode.

Note 1: The status depends on the port registers.

Note 2: The status depends on the CGSTBYCR<PTKEEP> bit. The state of port keeps the state when <PTKEEP> is set to "1".

Note 3: It is available to reduce leakage current by stopping reference voltage for AD converter.

Note 4: Pay attention that the counter of watch dog timer function can not be cleared by CPU while in IDLE mode.

Note 5: Available for using the Low-speed oscillator only

Note 6: RTCOUT outputs are fixed.

Note 7: before entering STOP1/STOP2 mode, it should be stopped.

6.6.6 Releasing the Low Power Consumption Mode

The low power consumption 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			IDLE	STOP1	STOP2
Release source	Interrupt	INT1, INT2, INT7, INT8, INTD, INTE, INTF	o	o	o (note 4)
		INTSSP0 to 2	o	×	×
		INTI2C0 to 2	o	×	×
		INTRX0 to 3, INTTX0 to 3	o	×	×
		INTUART0 to 1	o	×	×
		INTRTC	o	o	o
		INTTB0 to 7, INTCAP00 to 71	o	×	×
		INTMTTB00 to 31 INTMTCAP00 to 31, INTMTEMG0 to 3	o	×	×
		INTAD, INTADHP, INTADM0 to 1	o	×	×
		INTDMAAERR, INTDMABERR, INTDMACERR, INTDMAA,INTDMAB,INTDMAC	o	×	×
		INTFLRDY	o	×	×
	SysTick interrupt		o	×	×
	Non-Maskable Interrupt (INTWDT)		o	×	×
	Non-Maskable Interrupt (INTLVD)		o	×	×
	RESET (WDT)		o	×	×
	RESET (LVD)		o	o	o
	RESET ($\overline{\text{RESET}}$ pin)		o	o	o

o : Starts the interrupt handling after the mode is released. (The $\overline{\text{reset}}$ initializes the LSI)

× : Unavailable

Note 1: Regarding to the warm-up time which is need to return from each mode, refer to "6.6.7 Warm-up".

Note 2: After STOP2 mode is released, reset operation initializes the internal supply voltage cut off peripheral circuit (Refer to Table 6-6). But back-up module is not initialized.

Note 3: 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 interrupt.

Note 4: When releasing from IDLE,STOP1/2 mode by interrupting level mode, hold the level until the interrupt handling starts. If the level is changed before that, the correct interrupt handling cannot be started.

- Release by interrupt request

  To release the low power consumption mode by an interrupt, the interrupt is set to detect interrupt request before entering the low power consumption mode.

  regarding to setting the interrupt to be used to release the STOP1 and STOP2 modes, refer to "Exceptions".

- Release by Non-Maskable Interrupt (NMI)

  There are three kinds of NMI sources: WDT interrupt (INTWDT), LVD interrupt (INTLVD) . WDT interrupt (INTWDT) can only be used in the IDLE mode.

- Release by reset

  Any low power consumption mode can be released by reset from the $\overline{\text{RESET}}$ pin, LVD.

  IDLE mode can be released by reset from WDT.

  After that, the mode switches to the NORMAL mode and all the registers are initialized as is the case with normal reset.

- Release by SysTick interrupt

  SysTick interrupt can only be used in the IDLE mode.

Refer to the "Interrupt" of chapter "Exceptions" for details.

6.6.7 Warm-up

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

In the mode transition from STOP1/STOP2 to the NORMAL, an internal high-speed oscillator is activated automatically. And an internal high-speed oscillator is selected as the source clock of warm-up timer, warm-up timer is activated automatically.

Then the system clock output is started after the elapse of warm-up time. It is necessary to set a warm-up time in the CGOSCCR<WUPT[11:0]> before executing the instruction to enter the STOP1/STOP2 mode. Regarding to warm-up time, refer to "6.6.8 Clock Operations in Mode Transition".

Note: Returning from the STOP1/STOP2 mode, related bits <WUPSEL2>, <WUPSEL1>, <OSCSEL>, <XEN2>, <XEN1> of the register CGOSCCR and CGPLLSEL<PLLSEL>,<PLLON> are initialized because of internal high-speed oscillator starts 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 → IDLE	Not required
NORMAL → STOP1	Not required
NORMAL → STOP2	Not required
IDLE → NORMAL	Not required
STOP1 → NORMAL	Auto-warm-up
STOP2 → NORMAL	Auto-warm-up

Note: When releasing by reset is executed, automatic warm-up is not performed. Input a reset until the oscillator becomes stable.

6.6.8 Clock Operations in Mode Transition

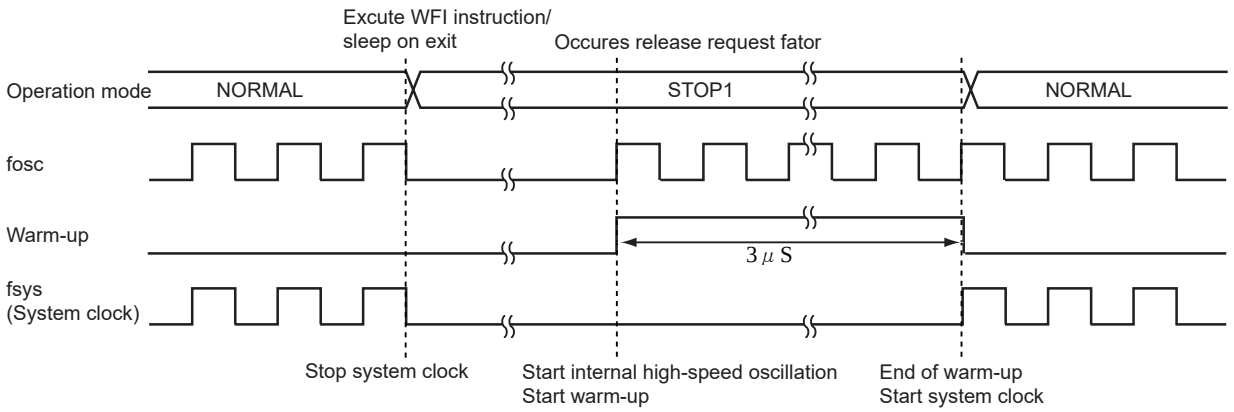
The clock operations in mode transition are described as follows.

6.6.8.1 Transition of operation modes: NORMAL → STOP1 → NORMAL

When returning to the NORMAL mode from the STOP1 mode, the warm-up is activated automatically.

Returning to the NORMAL mode by reset does not induce the automatic warm-up. The reset signal as same as cold reset should be inputted.

Note: It is necessary for the warm-up to be set "3μs".



6.6.8.2 Transition of operation modes: NORMAL → STOP2 → NORMAL

When returning to the NORMAL mode from the STOP2 mode, the warm-up is activated automatically.

After STOP2 mode is released, reset operation initializes the internal supply voltage cut off peripheral circuits. But the peripheral circuits which are cut the connection with the internal supply voltage are not initialized.

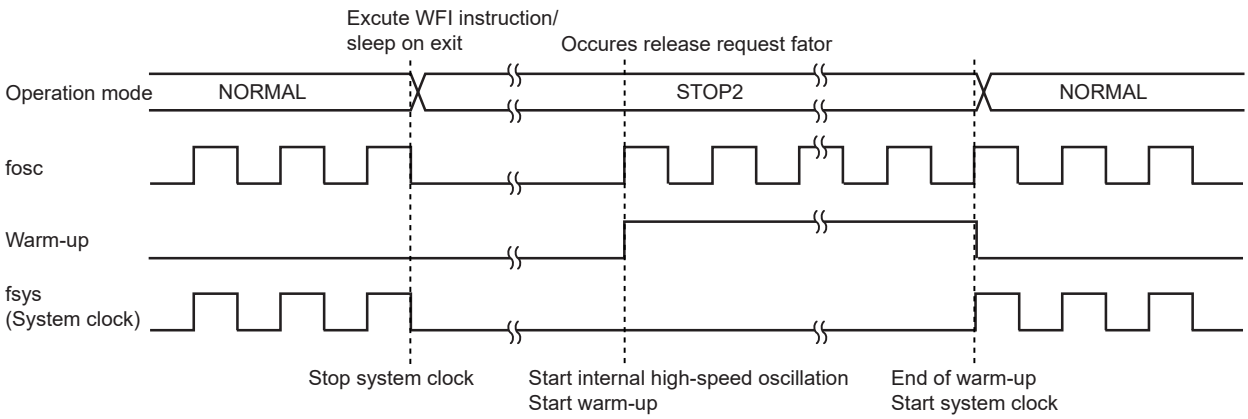
Returning to the NORMAL mode by reset does not induce the automatic warm-up. The reset signal as same as cold reset should be inputted.

Even returning to the NORMAL mode by without reset, it would be branched to interrupt service routine of reset.

Note 1: When releasing STOP2 by a external interrupt which is set as level-sensitive, the signal must be kept at the releasing level during 500μs or more.

Note 2: When releasing STOP2 by external interrupt pin, set <PTKEEP> to "1" before entering STOP2 mode.

Note 3: Before STOP2 moving, execute a WFI command after setting the CPU interrupt enable.



6.6.9 Precaution on Transition to the Low-power Consumption Mode

6.6.9.1 Case when the MCU Enters IDLE or STOP1 Mode

1. When the WFI instruction is executed to enter IDLE mode or STOP1 mode, if an interrupt request for release from the low-power consumption mode occurs, the MCU does not enter IDLE or STOP1 mode. This is because the interrupt request has a higher priority than the WFI instruction. Therefore, the following process must be added depending on enabling or disabling the interrupt:
   a. Case when the interrupts are disabled (masked only by PRIMASK)

      Write eight or more NOP instructions immediately after the WFI instruction, and then write the instruction to be executed.
   b. Case when the interrupts are enabled

      Write the interrupt process routine because the MCU branches to the interrupt service routine.

2. Before the MCU entering STOP1 mode, select the clock with CGOSCCR<WUPSEL>, which is the same as the clock selected with CGOSCCR<OSCSEL>, to use the same source clock for both the warm-up timer and fosc.

3. A non-maskable interrupt can be used to release only in IDLE mode.

4. Do not use non-maskable interrupts as a release factor of STOP1 mode. Before the MCU entering STOP1 mode, inhibit non-maskable interrupts, specify as follows:

   (Stop the watchdog timer, Stop the LVD)

6.6.9.2 Case when the MCU enters to STOP2 mode

1. Before the MCU entering STOP2 mode, execute the WFI instruction on condition that interrupts are enabled.

2. When the WFI instruction is executed to enter STOP2 mode, if an interrupt request for release from the low-power consumption mode occurs, the MCU does not enter STOP2 mode because the interrupt for release has a higher priority than the WFI instruction. Then the MCU branches to the interrupt service routine. Therefore, the following process must be added:
   a. Write the interrupt service routine.
   b. After the interrupt service routine is complete, the instructions following the WFI instruction are executed; write the process following the WFI instruction in case that the MCU does not enter STOP2 mode.

3. If the MCU did not enter STOP2 mode, CGOSCCR<WUPSEL2>,<WUPSEL1>, <OSCSEL>, <XEN2>, <XEN1>,and CGPLLSEL <PLLSEL>, <PLLON> are not initialized. These registers maintain the former condition before entering the STOP2 mode.

4. Before the MCU entering STOP2 mode, to use the internal high-speed oscillator (IHOSC) as the source clock for the system clock, specify as follows:

   CGOSCCR<OSCSEL>=0, CGPLLSEL<PLLSEL>=0, and CGSYSCR<GEAR[2:0]>=000

5. Before the MCU entering STOP2 mode, select the clock with CGOSCCR<WUPSEL2>,<WUPSEL1> which is the same as the clock selected with CGOSCCR<OSCSEL>, to use the same source clock for both the warm-up timer and fosc.

6. Do not use a non-maskable interrupt as a release factor of STOP2 mode. Before the MCU entering STOP2 mode, to inhibit non-maskable interrupts, specify as follows:

   (Stop the watchdog timer, Stop the LVD)

7. Exceptions

This chapter describes features, types and handling of exceptions.

Exceptions have close relation to the CPU core. Refer to "Arm documentation set for the Arm Cortex-M4" if needed.

7.1 Overview

An exception causes the CPU to stop the currently executing process and handle another process.

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-M4.

For detailed descriptions on each exception, refer to "Arm documentation set for the Arm Cortex-M4".

- 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 7.1.2.3
Return from exception	The CPU branches to another ISR or returns to the previous program.	Section 7.1.2.4

Note: ISR : Interrupt Service Routine

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)	$\overline{\text{Reset}}$ pin, WDT, STOP2 , LVD or SYSRETREQ
2	Non-Maskable Interrupt	−2	LVD 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 to 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	Pending system service request
15	SysTick	Configurable	Notification from system timer
up to 16	External Interrupt	Configurable	External interrupt pin or peripheral function (Note 2)

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

Note 2: **External interrupts have different sources and numbers in each product. For details, see "7.5.2 List of Interrupt Sources".**

(3) Priority setting

- Priority levels

  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.

  TMPM46BF10FG has a 3-bit configuration.

  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.

- 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 sub-priorities
	Pre-emption field	Sub-priority 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 (ISP). 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 Status Register (xPSR)
- Program Counter (PC)
- Link Register (LR)
- r12
- r0 - r3

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 (xPSR, PC, KR, r0 to r3 and r12) 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 sources.

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

- External $\overline{\text{RESET}}$ pin

  A reset exception occurs when an external $\overline{\text{RESET}}$ 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 STOP2 mode release

  A reset is generated when STOP2 mode is released. For details. see the chapter on the CG.

- Reset exception by SYSRESETREQ

  A reset can be generated by setting the <SYSRESETREQ> bit in the NVIC's Application Interrupt and Re-set Control Register.

- Reset exception by LVD

  The low voltage detection circuit (LVD) has a reset generating feature. For details, see the chapter on the LVD.

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.

- Non-maskable interrupt by WDT

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

- Non-maskable interrupt by LVD

  The low voltage detection circuit (LVD) has a reset generating feature. For details, see the chapter on the LVD.

Note: When moving into STOP1/STOP2 mode, non-maskable interrupt (NMI) should not be generated.

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".

Note: **In this product, fosc which is selected by CGOSCCR <OSCSEL> <HOSCON> 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 low power consumption mode (route 2) are input to the logic for releasing low power consumption mode in CG. After the active level for each interrupt request is detected, the logic for releasing low power consumption mode makes the active level to the new interrupt request signal. This signal is input to CPU. (route 6, 7 and 8).

The interrupt request of an external interrupt pins (route 3) is used as the either releasing or no releasing source of low power consumption mode by <INTxEN>.

The interrupt request of an external interrupt pin used as releasing the low power consumption mode is input to the logic for releasing low power consumption mode. If the specified active level is detected, the logic for releasing low power consumption mode makes it to the interrupt request, the interrupt request is input to CPU (route 2, 4, 5).

The interrupt request of an external interrupt pin not used as releasing the low power consumption mode is input CPU directly (route 2, 3, 5).

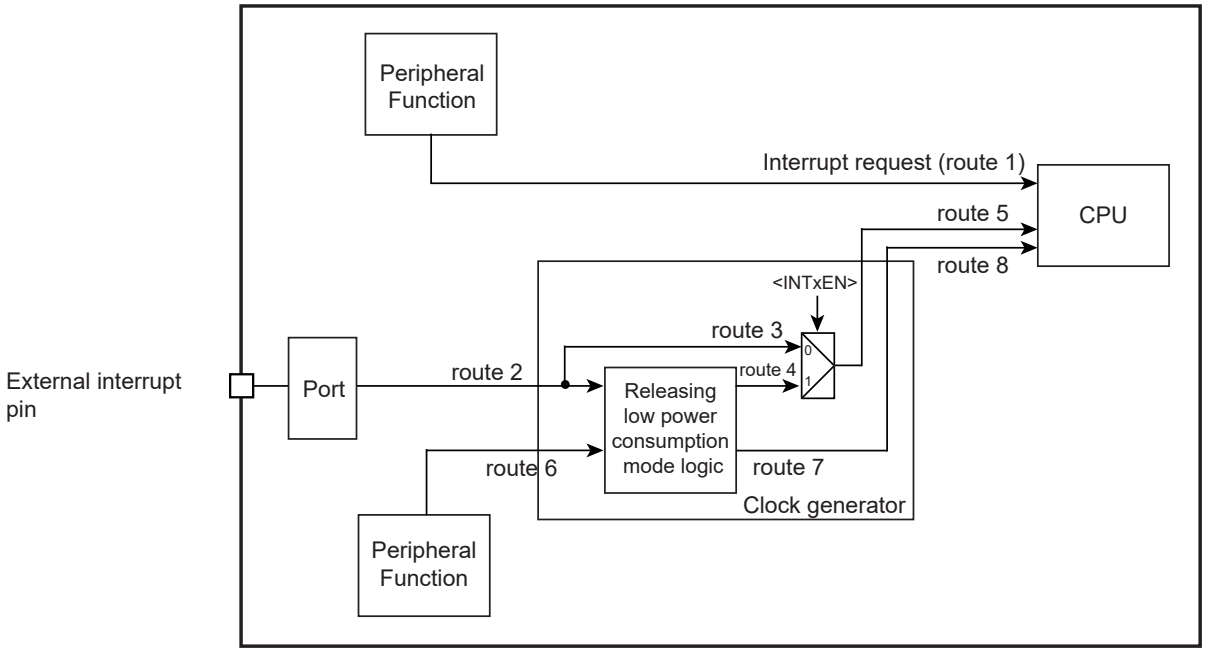


Figure 7-1 Interrupt Route

7.5.1.2 Generation of the interrupt request

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 interrupt pin

  Set the port control register so that the external interrupt 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.

CPU recognizes the "H" level of the interrupt request signal as interrupt request.

7.5.1.3 Setting the registers for releasing the low power consumption mode

Some interrupt request among all interrupt request can be used to release the low power consumption mode.

To use the interrupt request as the releasing the low power consumption mode, <INTxEN> is set to "1" and the active level for the releasing the low power consumption mode is specified by <EMCGx[2:0]>.

When the level specified by <EMCGx[2:0]> for the releasing the low power consumption mode is input into an external interrupt pin, the low power consumption mode is released and the interrupt of "H" level is occurred.

When <EMCGx[2:0]> is "100", the active level detected until releasing the low power consumption mode is read from <EMSTx[1:0]>.

The clearing the interrupt request is specified by CGICRCG<ICRCG>. <EMSTx[1:0]> is initialized by clearing the interrupt request.

7.5.2 List of Interrupt Sources

Table 7-3 shows the list of interrupt sources.

Table 7-3 List of Interrupt Sources

No.	Interrupt Source		The active level to release the low power consumption mode					CG interrupt mode control register
			"Low" level	"High" level	Rising edge	Falling edge	Both edge	
0	INT0	External interrupt pin 0						
1	INT1	External interrupt pin 1	o	o	o	o	o	CGIMCGA
2	INT2	External interrupt pin 2	o	o	o	o	o	
3	INT3	External interrupt pin 3						
4	INT4	External interrupt pin 4						
5	INT5	External interrupt pin 5						
6	INT6	External interrupt pin 6						
7	INT7	External interrupt pin 7	o	o	o	o	o	CGIMCGA
8	INT8	External interrupt pin 8	o	o	o	o	o	
9	INT9	External interrupt pin 9						
10	INTA	External interrupt pin A						
11	INTB	External interrupt pin B						
12	INTC	External interrupt pin C						
13	INTD	External interrupt pin D	o	o	o	o	o	CGIMCGB
14	INTE	External interrupt pin E	o	o	o	o	o	
15	INTF	External interrupt pin F	o	o	o	o	o	
16	INTRX0	SC receive interrupt (Channel 0)						
17	INTTX0	SC transmit interrupt (Channel 0)						
18	INTRX1	SC receive interrupt (Channel 1)						
19	INTTX1	SC transmit interrupt (Channel 1)						
20	INTRX2	SC receive interrupt (Channel 2)						
21	INTTX2	SC transmit interrupt (Channel 2)						
22	INTRX3	SC receive interrupt (Channel 3)						
23	INTTX3	SC transmit interrupt (Channel 3)						
24	INTUART0	UART interrupt (channel 0)						
25	INTUART1	UART interrupt (channel 1)						
26	INTI2C0	I2C0 interrupt						
27	INTI2C1	I2C1 interrupt						
28	INTI2C2	I2C2 interrupt						
29	INTSSP0	SPI serial interface (channel 0)						
30	INTSSP1	SPI serial interface (channel 1)						
31	INTSSP2	SPI serial interface (channel 2)						
32	INTADHP	Highest priority AD conversion complete interrupt						
33	INTADM0	AD conversion monitoring function interrupt 0						
34	INTADM1	AD conversion monitoring function interrupt 1						
35	INTAD	AD conversion completion interrupt						
36	INTAES	AES completion interrupt						
37	INTSHA	SHA completion interrupt						
38	INTMLA	MLA completion interrupt						
39	INTESG	ESG completion interrupt						

Table 7-3 List of Interrupt Sources

No.	Interrupt Source		The active level to release the low power consumption mode					CG interrupt mode control register
			"Low" level	"High" level	Rising edge	Falling edge	Both edge	
40	INTSNFCSEQ	SNFC interrupt (command sequence end)						
41	INTSNFCPRTAE	SNFC interrupt (page RAM transmission A completion)						
42	INTSNFCPRTCE	SNFC interrupt (page RAM transmission C completion)						
43	INTSNFCFAIL	SNFC interrupt (decoding fail)						
44	Reserved	-						
45	Reserved	-						
46	INTSNFCDEC	SNFC interrupt (decoding sequence end)						
47	INTMTEMG0	MPT EMG interrupt (channel 0)						
48	INTMTPTB00	MPT compare match 0 / overflow, IGBT cycle interrupt (channel 0)						
49	INTMTTTB01	MPT compare match 1, IGBT trigger interrupt (channel 0)						
50	INTMTCAP00	MPT input capture 0 (channel 0)						
51	INTMTCAP01	MPT input capture 1 (channel 0)						
52	INTMTEMG1	MPT EMG interrupt (channel 1)						
53	INTMTPTB10	MPT compare match 0 / overflow, IGBT cycle interrupt (channel 1)						
54	INTMTTTB11	MPT compare match 1, IGBT trigger interrupt (channel 1)						
55	INTMTCAP10	MPT input capture 0 (channel 1)						
56	INTMTCAP11	MPT input capture 1 (channel 1)						
57	INTMTEMG2	MPT EMG interrupt (channel 2)						
58	INTMTPTB20	MPT compare match 0 / overflow, IGBT cycle interrupt (channel 2)						
59	INTMTTTB21	MPT compare match 1, IGBT trigger interrupt (channel 2)						
60	INTMTCAP20	MPT input capture 0 (channel 2)						
61	INTMTCAP21	MPT input capture 1 (channel 2)						
62	INTMTEMG3	MPT EMG interrupt (channel 3)						
63	INTMTPTB30	MPT compare match 0 / overflow, IGBT cycle interrupt (channel 3)						
64	INTMTTTB31	MPT compare match 1, IGBT trigger interrupt (channel 3)						
65	INTMTCAP30	MPT input capture 0 (channel 3)						
66	INTMTCAP31	MPT input capture 1 (channel 3)						
67	INTTB0	16-bit TMRB compare match 0 / 1 / overflow (channel 0)						
68	INTCAP00	TMRB0 input capture 0 interrupt						
69	INTCAP01	TMRB0 input capture 1 interrupt						
67	INTTB1	16-bit TMRB compare match 0 / 1 / overflow (channel 1)						
68	INTCAP10	TMRB1 input capture 0 interrupt						
69	INTCAP11	TMRB1 input capture 1 interrupt						
73	INTTB2	16-bit TMRB compare match 0 / 1 / overflow (channel 2)						
74	INTCAP20	TMRB2 input capture 0 interrupt						
75	INTCAP21	TMRB2 input capture 1 interrupt						

Table 7-3 List of Interrupt Sources

No.	Interrupt Source		The active level to release the low power consumption mode					CG interrupt mode control register
			"Low" level	"High" level	Rising edge	Falling edge	Both edge	
76	INTTB3	16-bit TMRB compare match 0 / 1 / overflow (channel 3)						
77	INTCAP30	TMRB3 input capture 0 interrupt						
78	INTCAP31	TMRB3 input capture 1 interrupt						
79	INTTB4	16-bit TMRB compare match 0 / 1 / overflow (channel 4)						
80	INTCAP40	TMRB4 input capture 0 interrupt						
81	INTCAP41	TMRB4 input capture 1 interrupt						
82	INTTB5	16-bit TMRB compare match 0 / 1 / overflow (channel 5)						
83	INTCAP50	TMRB5 input capture 0 interrupt						
84	INTCAP51	TMRB5 input capture 1 interrupt						
85	INTTB6	16-bit TMRB compare match 0 / 1 / overflow (channel 6)						
86	INTCAP60	TMRB6 input capture 0 interrupt						
87	INTCAP61	TMRB6 input capture 1 interrupt						
88	INTTB7	16-bit TMRB compare match 0 / 1 / overflow (channel 7)						
89	INTCAP70	TMRB7 input capture 0 interrupt						
90	INTCAP71	TMRB7 input capture 1 interrupt						
91	INTRTC	RTC interrupt	×	×	×	o	×	CGIMCGB
92	INTDMAA	DMAC transmission completion interrupt (UnitA, channel4 to channel31)						
93	INTDMAB	DMAC transmission completion interrupt (UnitB, channel24 to channel31)						
94	INTDMAC	DMAC transmission completion interrupt (UnitC, channel12 to channel31)						
95	INTDMACTC8	DMAC transmission completion interrupt (UnitC,channel8)						
96	INTDMACTC9	DMAC transmission completion interrupt (UnitC,channel9)						
97	INTDMACTC10	DMAC transmission completion interrupt (UnitC,channel10)						
98	INTDMACTC11	DMAC transmission completion interrupt (UnitC,channel11)						
99	INTDMAAERR	DMAC transmission error interrupt (unit A)						
100	INTDMABERR	DMAC transmission error interrupt (unit B)						
101	INTDMACERR	DMAC transmission error interrupt (unit C)						
102	INTFLRDY	FLASH Ready interrupt						

Note: The active level that is marked with "o" to release the low power consumption mode cab be used.
The active level that is marked with "×" can not be used.

7.5.3 Interrupt Handling

7.5.3.1 Flowchart

The following shows how an interrupt is handled.

In the following descriptions, ▭ indicates hardware handling. ▭ indicates software handling.

Processing	Details	See
Settings for detection	Set the relevant NVIC registers for detecting interrupts. Set the clock generator as well if each interrupt source is used to clear a stand-by mode. oCommon setting NVIC registers oSetting to clear standby mode Clock generator	"7.5.3.2 Preparation"
Settings for sending interrupt signal	Execute an appropriate setting to send the interrupt signal depending on the interrupt type. oSetting for interrupt from external pin Port oSetting for interrupt from peripheral function Peripheral function (See the chapter of each peripheral function for details.)	
⇩		
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.	"7.5.3.3 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.	"7.5.3.4 Detection by CPU"
⇩		
CPU handles interrupt	The CPU handles the interrupt. The CPU pushes register contents to the stack before entering the ISR.	"7.5.3.5 CPU processing"
⇩		

Processing	Details	See
ISR execution	Program for the ISR. Clear the interrupt source if needed.	"7.5.3.6 Interrupt Service Routine (ISR)"
Return to preceding program	Configure to return to the preceding program of the ISR.	

7.5.3.2 Preparation

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

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

To configure the clock generator, you must follow the order indicated here not to cause any unexpected 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. Pre configuration (1) (Interrupt from external interrupt pin)
4. Pre configuration (2) (Interrupt from peripheral function)
5. Pre configuration (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 PRIMASK 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 level.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 field in the Application Interrupt and Reset Control Register.

NVIC register		
<PRI_n>	←	"priority"
<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) Pre configuration (1) (Interrupt from external interrupt pin)

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

Port register		
PxIE<PxmIE>	←	"1"

Note: **x: port number / m: corresponding bit**
Setting PxIE to enable input enables the corresponding interrupt input. Be careful not to enable interrupts that are not used.

(4) Pre configuration (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) Pre configuration (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. According to the active level, refer to "Table 7-3 List of Interrupt Sources".

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 "7.6.3.2 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.

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-Enable [m]	←	"1"
Interrupt mask register		
PRIMASK	←	"0"

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

7.5.3.3 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.

7.5.3.4 Detection by CPU

The CPU detects an interrupt request with the highest priority.

7.5.3.5 CPU processing

On detecting an interrupt, the CPU pushes the contents of xPSR, PC, LR, R12 and r3 to r0 to the stack then enter the ISR.

7.5.3.6 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) Procedure during ISR

An ISR normally pushes register contents to the stack and handles an interrupt as required. The Cortex-M4 core automatically pushes the contents of xPSR, PC, LR, R12 and r3 to r0 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

7.6.1 Register List

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

For detail of the base address, refer to "Address lists of peripheral functions" of "Memory Map" chapter.

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 0x047F
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

peripheral function name : CG

Register name		Address
CG Interrupt Mode Control Register A	CGIMCGA	0x0040
CG Interrupt Mode Control Register B	CGIMCGB	0x0044
CG Interrupt Request Clear Register	CGICRCG	0x0060
Reset Flag Register	CGRSTFLG	0x0064
NMI Flag Register	CGNMIFLG	0x0068

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 which is selected by CGOSCCR <OSCSEL> <HOSCON> 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	1	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	1	0	0
	7	6	5	4	3	2	1	0
bit symbol	TENMS							
After reset	0	0	0	0	0	0	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 10 ms.
29-24	–	R	Read as 0.
23-0	TENMS	R	Calibration value (Note)

Note: This product does not prepare the calibration value.

7.6.2.5 Interrupt control registers

Each interrupt source has interrupt set-enable register, interrupt clear-enable register, interrupt set-pending register and interrupt clear-pending register.

(1) Interrupt Set-Enable Register

This register specifies enabling interrupt and confirms the enable/disable state of interrupt.

When set this register to "1", the corresponding interrupt is enabled.

Writing to "0" is no meaning.

To read this register, be able to confirm the enable/disable state of corresponding interrupt.

To clear the bit of this register, the corresponding bit of interrupt clear-enable register is cleared to "0".

Bit symbol	Type	Function
SETENA	R/W	Interrupt No. [102:0] [Write] 1 : Enable interrupt [Read] 0 : The interrupt state is disabled 1 : The interrupt state is enabled

(a) Interrupt Set-Enable Register 1

	31	30	29	28	27	26	25	24
bit symbol	SETENA (Interrupt 31)	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	SETENA (Interrupt 15)	SETENA (Interrupt 14)	SETENA (Interrupt 13)	SETENA (Interrupt 12)	SETENA (Interrupt 11)	SETENA (Interrupt 10)	SETENA (Interrupt 9)	SETENA (Interrupt 8)
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

(b) Interrupt Set-Enable Register 2

	31	30	29	28	27	26	25	24
bit symbol	SETENA (Interrupt 63)	SETENA (Interrupt 62)	SETENA (Interrupt 61)	SETENA (Interrupt 60)	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 36)	SETENA (Interrupt 35)	SETENA (Interrupt 34)	SETENA (Interrupt 33)	SETENA (Interrupt 32)
After reset	0	0	0	0	0	0	0	0

(c) Interrupt Set-Enable Register 3

	31	30	29	28	27	26	25	24
bit symbol	SETENA (Interrupt 95)	SETENA (Interrupt 94)	SETENA (Interrupt 93)	SETENA (Interrupt 92)	SETENA (Interrupt 91)	SETENA (Interrupt 90)	SETENA (Interrupt 89)	SETENA (Interrupt 88)
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)	SETENA (Interrupt 73)	SETENA (Interrupt 72)
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SETENA (Interrupt 71)	SETENA (Interrupt 70)	SETENA (Interrupt 69)	SETENA (Interrupt 68)	SETENA (Interrupt 67)	SETENA (Interrupt 66)	SETENA (Interrupt 65)	SETENA (Interrupt 64)
After reset	0	0	0	0	0	0	0	0

(d) 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 102)	SETENA (Interrupt 101)	SETENA (Interrupt 100)	SETENA (Interrupt 99)	SETENA (Interrupt 98)	SETENA (Interrupt 97)	SETENA (Interrupt 96)
After reset	0	0	0	0	0	0	0	0

(2) Interrupt Clear-Enable Register

This register specifies disabling interrupt and confirms the enable/disable state of interrupt.

When set this register to "1", the corresponding interrupt is disabled.

Writing to "0" is no meaning.

To read this register, be able to confirm the enable/disable state of corresponding interrupt.

To clear the bit of this register, the corresponding bit of interrupt clear-enable register is cleared to "0".

Bit symbol	Type	Function
CLRENA	R/W	Interrupt No. [102:0] [Write] 1 : Disable interrupt [Read] 0 : The interrupt state is disabled 1 : The interrupt state is enabled

(a) Interrupt Clear-Enable Register 1

	31	30	29	28	27	26	25	24
bit symbol	CLRENA (Interrupt 31)	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	CLRENA (Interrupt 15)	CLRENA (Interrupt 14)	CLRENA (Interrupt 13)	CLRENA (Interrupt 12)	CLRENA (Interrupt 11)	CLRENA (Interrupt 10)	CLRENA (Interrupt 9)	CLRENA (Interrupt 8)
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

(b) Interrupt Clear-Enable Register 2

	31	30	29	28	27	26	25	24
bit symbol	CLRENA (Interrupt 63)	CLRENA (Interrupt 62)	CLRENA (Interrupt 61)	CLRENA (Interrupt 60)	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 36)	CLRENA (Interrupt 35)	CLRENA (Interrupt 34)	CLRENA (Interrupt 33)	CLRENA (Interrupt 32)
After reset	0	0	0	0	0	0	0	0

(c) Interrupt Clear-Enable Register 3

	31	30	29	28	27	26	25	24
bit symbol	CLRENA (Interrupt 95)	CLRENA (Interrupt 94)	CLRENA (Interrupt 93)	CLRENA (Interrupt 92)	CLRENA (Interrupt 91)	CLRENA (Interrupt 90)	CLRENA (Interrupt 89)	CLRENA (Interrupt 88)
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)	CLRENA (Interrupt 73)	CLRENA (Interrupt 72)
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CLRENA (Interrupt 71)	CLRENA (Interrupt 70)	CLRENA (Interrupt 69)	CLRENA (Interrupt 68)	CLRENA (Interrupt 67)	CLRENA (Interrupt 66)	CLRENA (Interrupt 65)	CLRENA (Interrupt 64)
After reset	0	0	0	0	0	0	0	0

(d) 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 102)	CLRENA (Interrupt 101)	CLRENA (Interrupt 100)	CLRENA (Interrupt 99)	CLRENA (Interrupt 98)	CLRENA (Interrupt 97)	CLRENA (Interrupt 96)
After reset	0	0	0	0	0	0	0	0

(3) Interrupt Set-Pending Register

This register specifies pending interrupt and confirms the pending state of interrupt.

When set this register to "1", the corresponding interrupt is pended. But this register is invalid for the interrupt which is already pended or disabled.

Writing to "0" is no meaning.

To read this register, be able to confirm the pending state of corresponding interrupt.

To clear the bit of this register, the corresponding bit of interrupt clear-pending register is cleared to "0".

Bit symbol	Type	Function
SETPEND	R/W	Interrupt No. [102:0] [Write] 1 : Pending interrupt [Read] 0 : No pending 1 : Pending

(a) Interrupt Set-Pending Register 1

	31	30	29	28	27	26	25	24
bit symbol	SETPEND (Interrupt 31)	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	SETPEND (Interrupt 15)	SETPEND (Interrupt 14)	SETPEND (Interrupt 13)	SETPEND (Interrupt 12)	SETPEND (Interrupt 11)	SETPEND (Interrupt 10)	SETPEND (Interrupt 9)	SETPEND (Interrupt 8)
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

(b) Interrupt Set-Pending Register 2

	31	30	29	28	27	26	25	24
bit symbol	SETPEND (Interrupt 63)	SETPEND (Interrupt 62)	SETPEND (Interrupt 61)	SETPEND (Interrupt 60)	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 36)	SETPEND (Interrupt 35)	SETPEND (Interrupt 34)	SETPEND (Interrupt 33)	SETPEND (Interrupt 32)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

(c) Interrupt Set-Pending Register 3

	31	30	29	28	27	26	25	24
bit symbol	SETPEND (Interrupt 95)	SETPEND (Interrupt 94)	SETPEND (Interrupt 93)	SETPEND (Interrupt 92)	SETPEND (Interrupt 91)	SETPEND (Interrupt 90)	SETPEND (Interrupt 89)	SETPEND (Interrupt 88)
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)	SETPEND (Interrupt 73)	SETPEND (Interrupt 72)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	SETPEND (Interrupt 71)	SETPEND (Interrupt 70)	SETPEND (Interrupt 69)	SETPEND (Interrupt 68)	SETPEND (Interrupt 67)	SETPEND (Interrupt 66)	SETPEND (Interrupt 65)	SETPEND (Interrupt 64)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

(d) Interrupt Set-Pending Register 4

	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	-	SETPEND (Interrupt 102)	SETPEND (Interrupt 101)	SETPEND (Interrupt 100)	SETPEND (Interrupt 99)	SETPEND (Interrupt 98)	SETPEND (Interrupt 97)	SETPEND (Interrupt 96)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

(4) Interrupt Clear-Pending Register

This register specifies clearing the pended interrupt and confirms the pending state of interrupt.

When set this register to "1", the corresponding pended interrupt is cleared. But this register is invalid for the interrupt which is already started.

Writing to "0" is no meaning.

To read this register, be able to confirm the pending state of corresponding interrupt.

Bit symbol	Type	Function
SETPEND	R/W	Interrupt No. [102:0] [Write] 1 : Clear Pended interrupt [Read] 0 : No pending 1 : Pending

(a) Interrupt Clear-Pending Register 1

	31	30	29	28	27	26	25	24
bit symbol	CLRPEND (Interrupt 31)	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	CLRPEND (Interrupt 15)	CLRPEND (Interrupt 14)	CLRPEND (Interrupt 13)	CLRPEND (Interrupt 12)	CLRPEND (Interrupt 11)	CLRPEND (Interrupt 10)	CLRPEND (Interrupt 9)	CLRPEND (Interrupt 8)
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

(b) Interrupt Clear-Pending Register 2

	31	30	29	28	27	26	25	24
bit symbol	CLRPEND (Interrupt 63)	CLRPEND (Interrupt 62)	CLRPEND (Interrupt 61)	CLRPEND (Interrupt 60)	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 36)	CLRPEND (Interrupt 35)	CLRPEND (Interrupt 34)	CLRPEND (Interrupt 33)	CLRPEND (Interrupt 32)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

(c) Interrupt Clear-Pending Register 3

	31	30	29	28	27	26	25	24
bit symbol	CLRPEND (Interrupt 95)	CLRPEND (Interrupt 94)	CLRPEND (Interrupt 93)	CLRPEND (Interrupt 92)	CLRPEND (Interrupt 91)	CLRPEND (Interrupt 90)	CLRPEND (Interrupt 89)	CLRPEND (Interrupt 88)
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)	CLRPEND (Interrupt 73)	CLRPEND (Interrupt 72)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	CLRPEND (Interrupt 71)	CLRPEND (Interrupt 70)	CLRPEND (Interrupt 69)	CLRPEND (Interrupt 68)	CLRPEND (Interrupt 67)	CLRPEND (Interrupt 66)	CLRPEND (Interrupt 65)	CLRPEND (Interrupt 64)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

(d) Interrupt Clear-Pending Register 4

	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	-	CLRPEND (Interrupt 102)	CLRPEND (Interrupt 101)	CLRPEND (Interrupt 100)	CLRPEND (Interrupt 99)	CLRPEND (Interrupt 98)	CLRPEND (Interrupt 97)	CLRPEND (Interrupt 96)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

7.6.2.6 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	PRI_11	PRI_10	PRI_9	PRI_8
0xE000_E40C	PRI_15	PRI_14	PRI_13	PRI_12
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_31	PRI_30	PRI_29	PRI_28
0xE000_E420	PRI_35	PRI_34	PRI_33	PRI_32
0xE000_E424	PRI_39	PRI_38	PRI_37	PRI_36
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	PRI_63	PRI_62	PRI_61	PRI_60
0xE000_E440	PRI_67	PRI_66	PRI_65	PRI_64
0xE000_E444	PRI_71	PRI_70	PRI_69	PRI_68
0xE000_E448	PRI_75	PRI_74	PRI_73	PRI_72
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	PRI_91	PRI_90	PRI_89	PRI_88
0xE000_E45C	PRI_95	PRI_94	PRI_93	PRI_92
0xE000_E460	PRI_99	PRI_98	PRI_97	PRI_96
0xE000_E464	-	PRI_102	PRI_101	PRI_100

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.

7.6.2.7 Vector Table Offset Register

	31	30	29	28	27	26	25	24
bit symbol	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
31-7	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.

7.6.2.8 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 (Write)/ VECTKEY-STAT(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 endian
14-11	–	R	Read as 0.
10-8	PRIGROUP	R/W	Interrupt priority grouping 000: seven bits of pre-emption priority, one bit of sub-priority 001: six bits of pre-emption priority, two bits of sub-priority 010: five bits of pre-emption priority, three bits of sub-priority 011: four bits of pre-emption priority, four bits of sub-priority 100: three bits of pre-emption priority, five bits of sub-priority 101: two bits of pre-emption priority, six bits of sub-priority 110: one bit of pre-emption priority, seven bits of sub-priority 111: no pre-emption priority, eight bits of sub-priority 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 is the responsibility of the application to re initialize 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.**

7.6.2.9 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.

7.6.2.10 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: Enable
17	BUSFAUL TENA	R/W	Bus Fault 0: Disabled 1: Enable
16	MEMFAULT ENA	R/W	Memory Management 0: Disabled 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 repair stack contents.**

7.6.3 Clock generator registers

7.6.3.1 CG Interrupt Mode Control Register

This register specifies the active level to release the low power consumption mode and enable/disable the releasing the low power consumption mode. And detecting active level can be read from this register.

Bit symbol	Type	Function
EMCGx[2:0]	R/W	Select the active level to release the low power consumption mode The active level can be selected from Table 7-4. 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges 101 to 111 : setting prohibited
EMSTx[1:0]	R	Detecting active level (This bit is valid in EMCGx[2:0]="100".) 00: – 01: Rising edge 10: Falling edge 11: Both edges
INTxEN	R/W	Release the low power consumption mode 0 : Disable 1 : Enable

Table 7-4 The Active Level to release the Low Power Consumption Mode

Interrupt Source		Active level control register	The active level to release the low power consumption mode				
			"Low" level	"High" level	Rising edge	Falling edge	Both edge
INT1	External interrupt pin 1	CGIMCGA <EMCG00[2:0]>	o	o	o	o	o
INT2	External interrupt pin 2	CGIMCGA <EMCG01[2:0]>	o	o	o	o	o
INT7	External interrupt pin 7	CGIMCGA <EMCG02[2:0]>	o	o	o	o	o
INT8	External interrupt pin 8	CGIMCGA <EMCG03[2:0]>	o	o	o	o	o
INTD	External interrupt pin D	CGIMCGB <EMCG04[2:0]>	o	o	o	o	o
INTE	External interrupt pin E	CGIMCGB <EMCG05[2:0]>	o	o	o	o	o
INTF	External interrupt pin F	CGIMCGB <EMCG06[2:0]>	o	o	o	o	o
INTRTC	RTC interrupt	CGIMCGB <EMCG07[2:0]>	×	×	×	o	×

Note: The active level that is marked with "o" to release the low power consumption mode cab be used.
The active level that is marked with "×" can not be used.

(1) CGIMCGA(CG Interrupt Mode Control Register A)

	31	30	29	28	27	26	25	24
bit symbol	-	EMCG03			EMST03		-	INT03EN
After reset	0	0	1	0	0	0	undefined	0
	23	22	21	20	19	18	17	16
bit symbol	-	EMCG02			EMST02		-	INT02EN
After reset	0	0	1	0	0	0	undefined	0
	15	14	13	12	11	10	9	8
bit symbol	-	EMCG01			EMST01		-	INT01EN
After reset	0	0	1	0	0	0	undefined	0
	7	6	5	4	3	2	1	0
bit symbol	-	EMCG00			EMST00		-	INT00EN
After reset	0	0	1	0	0	0	undefined	0

Note 1: The active level specified by <EMCGx[2:0]> is depend on the interrupt request. To set correctly, refer to Table 7-4.

Note 2: **<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 3: Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

Note 4: "0" is read from bit 31, 23, 15 and 7.

Note 5: Undefined value is read from bit 25, 17, 9 and 1.

(2) CGIMCGB(CG Interrupt Mode Control Register B)

	31	30	29	28	27	26	25	24
bit symbol	-	EMCG07			EMST07		-	INT07EN
After reset	0	0	1	0	0	0	undefined	0
	23	22	21	20	19	18	17	16
bit symbol	-	EMCG06			EMST06		-	INT06EN
After reset	0	0	1	0	0	0	undefined	0
	15	14	13	12	11	10	9	8
bit symbol	-	EMCG05			EMST05		-	INT05EN
After reset	0	0	1	0	0	0	undefined	0
	7	6	5	4	3	2	1	0
bit symbol	-	EMCG04			EMST04		-	INT04EN
After reset	0	0	1	0	0	0	undefined	0

Note 1: The active level specified by <EMCGx[2:0]> is depend on the interrupt request. To set correctly, refer to Table 7-4.

Note 2: **<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 3: Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

Note 4: "0" is read from bit 31, 23, 15 and 7.

Note 5: Undefined value is read from bit 25, 17, 9 and 1.

7.6.3.2 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: INT1 0_0001: INT2 0_0010: INT7 0_0011: INT8 0_0100: INTD 0_0101: INTE 0_0110: INTF 0_0111: INTRTC 0_1000 to 1_1111: Prohibited Read as 0.

7.6.3.3 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	-	-	-	-	-	NMIFLG2	-	NMIFLG0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	–	R	Read as 0.
2	NMIFLG2	R	NMI source generation flag 0: not applicable 1: Only lower than the setting voltage when voltage decreasing.
1	–	R	Read as 0.
0	NMIFLG0	R	NMI source generation flag 0: not applicable 1: generated from WDT

7.6.3.4 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	-	LVDRSTF	-	SYSRSTF	BUPRSTF	WDTRSTF	OSCFLF	PINRSTF
After pin reset	0	0	0	0	0	0	undefined	1

Bit	Bit Symbol	Type	Function
31-7	–	R	Read as 0.
6	LVDRSTF	R/W	LVD reset flag [Read] 0: - 1: Reset from LVD [Write] 0 : Clear reset flag (Note2) 1 : don't care
5	–	R	Read as 0.
4	SYSRSTF	R/W	<SYSRESETREQ> reset flag (Note1) [Read] 0: - 1: Reset from SYSRESETREQ [Write] 0 : Clear reset flag (Note2) 1 : don't care
3	BUPRSTF	R/W	STOP2 reset flag [Read] 0: - 1: Reset flag by STOP2 mode release [Write] 0 : Clear reset flag (Note2) 1 : don't care
2	WDTRSTF	R/W	WDT reset flag [Read] 0: - 1: Reset from WDT [Write] 0 : Clear reset flag (Note2) 1 : don't care
1	OSCFLF	R	Flag for stopping of the internal high-speed oscillator or writing to the flash memory 0: Can't stop the internal high-speed oscillator and write to the flash memory 1: Can stop the internal high-speed oscillator and write to the flash memory
0	PINRSTF	R/W	$\overline{\text{RESET}}$ pin flag [Read] 0: - 1: Reset from $\overline{\text{RESET}}$ pin. [Write] 0 : Clear reset flag (Note2) 1 : don't care

Note 1: The reset which is generated by application interrupt in NVIC of CPU and setting reset control register <SYSRESETREQ> is displayed.

Note 2: This bit is not cleared automatically. Therefore, clear the corresponded bit to "0" to clear it.

8. µDMA Controller (µDMAC)

8.1 Overview

8.1.1 Function List

The main functions per unit are shown as below:

For the information on the start trigger of peripheral functions, refer to the chapter on "Product Information."

Table 8-1 µDMA outline (Per unit)

Functions	Features		Descriptions
Channels	32 channels		-
Start trigger	Start by Hardware		DMA requests from peripheral functions
	Start by Software		Specified by the DMAxChnlSwRequest register
Priority	Between channels	ch0 (high priority) > … > ch31 (high priority) > ch0 (Normal priority) > … > ch31 (Normal priority)	High-priority can be configured by the DMAxChnl-PrioritySet register
Transfer data size	8/16/32 bits		-
The number of transfer	1 to 1024 times		-
Address	Transfer source address	Increment / fixed	Transfer source address and destination address can be selected from the increment setting or fixed setting.
	Transfer destination address	Increment / fixed	
Endian	Little-endian		-
Interrupt function	Transfer completion interrupt		Output for each unit
	Error interrupt		
Operation mode	Basic mode Automatic request mode Ping-pong mode Memory scatter/gather mode Peripheral scatter/gather mode		-

8.2 Block Diagram

The µDMA controller contains the following blocks:

- APB block

  This block controls the access to the control register.
- AHB block

  This block controls the bus cycle of the DMA transfer.
- DMA control block

  This block controls the whole operation of the DMA.
- Interrupt control block

  This block integrates interrupts and sets the flag register.

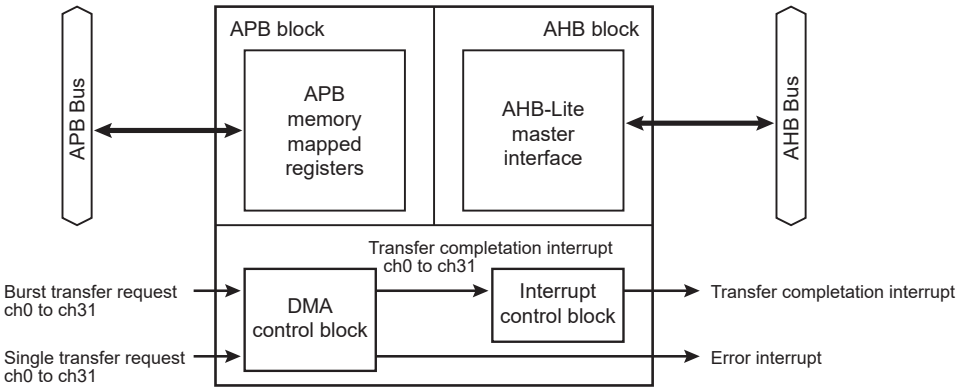


Figure 8-1 µDMA block diagram

8.3 Registers

8.3.1 Register List

The following table shows control registers and addresses:

For the base address, refer to "A list of peripheral function base addresses" in the chapter on "Memory Map."

Then name of peripheral: DMA

Register names		Address (Base+)
DMA status register	DMAxStatus	0x0000
DMA configuration register	DMAxCfg	0x0004
Channel control data base pointer register	DMAxCtrlBasePtr	0x0008
Channel alternate control data base pointer register	DMAxAltCtlBasePtr	0x000C
Channel software request status register	DMAxChnlSwRequest	0x0014
Channel useburst set register	DMAxChnlUseburstSet	0x0018
Channel useburst clear register	DMAxChnlUseburstClr	0x001C
Channel request mask set register	DMAxChnlReqMaskSet	0x0020
Channel request mask clear register	DMAxChnlReqMaskClr	0x0024
Channel enable set register	DMAxChnlEnableSet	0x0028
Channel enable clear register	DMAxChnlEnableClr	0x002C
Channel primary-alternate set register	DMAxChnlPriAltSet	0x0030
Channel primary-alternate clear register	DMAxChnlPriAltClr	0x0034
Channel priority set register	DMAxChnlPrioritySet	0x0038
Channel priority clear register	DMAxChnlPriorityClr	0x003C
Bus error clear register	DMAxErrClr	0x004C

Then name of peripheral: DMAIF

Register name		Address (Base+)
Flag register A	DMAIFFLGA	0x0000
Flag register B	DMAIFFLGB	0x0004
Flag register C	DMAIFFLGC	0x0008

Note:Access the registers in units of words (32 bits).

8.3.2 DMAxStatus (DMAC Status Register)

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	1	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	1	1	1	1	1
	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	-	-	-	-	-	-	-	master_ enable
After reset	Undefined	Undefined	Undefined	Undefined	0	0	0	0

Bit	Bit symbol	Type	Functions
31-29	-	R	Read as "0".
28	-	R	Read as "1".
27-21	-	R	Read as "0".
20-16	-	R	Read as "1".
15-8	-	R	Read as "0".
7-4	-	R	Read as an undefined value.
3-1	-	R	Read as "0".
0	master_enable	R	DMA operation 0: Disabled 1: Enabled

8.3.3 DMAxCfg (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	-	-	-	-	-	-	-	master_ enable
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit symbol	Type	Function
31-1	-	W	Write as "0".
0	master_ enable	W	DMA operation 0: Disabled 1: Enabled

Note: After DMAxCfg = 0x00000001 DMAxChnlReqMaskSet = 0xFFFFFFFF and DMAxChnlEnableSet = 0xFFFFFFFF are set to all units, release the masked channel of the unit to be used (the corresponding bit of DMAxChanlReqMaskClr is set to "1"). However, do not release the same factor in each DMAC unit at the same time.

8.3.4 DMAxCtrlBasePtr (Channel Control Data Base-pointer Register)

	31	30	29	28	27	26	25	24
Bit symbol	ctrl_base_ptr							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	ctrl_base_ptr							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	ctrl_base_ptr						-	-
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
31-10	ctrl_base_ptr	R/W	Primary data base-pointer Specifies the base address of the primary data.
9-0	-	R	Read as "0".

8.3.5 DMAxAltCtrlBasePtr (Channel Alternate Control Data Base-pointer Register)

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

Bit	Bit symbol	Type	Function
31-0	alt_ctrl_base_pt	R	Alternative data base-pointer. Reads the base address of the alternative data.

8.3.6 DMAxChnlSwRequest (Channel Software Request Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_sw_request (ch31)	chnl_sw_request (ch30)	chnl_sw_request (ch29)	chnl_sw_request (ch28)	chnl_sw_request (ch27)	chnl_sw_request (ch26)	chnl_sw_request (ch25)	chnl_sw_request (ch24)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
Bit symbol	chnl_sw_request (ch23)	chnl_sw_request (ch22)	chnl_sw_request (ch21)	chnl_sw_request (ch20)	chnl_sw_request (ch19)	chnl_sw_request (ch18)	chnl_sw_request (ch17)	chnl_sw_request (ch16)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
Bit symbol	chnl_sw_request (ch15)	chnl_sw_request (ch14)	chnl_sw_request (ch13)	chnl_sw_request (ch12q)	chnl_sw_request (ch11)	chnl_sw_request (ch10)	chnl_sw_request (ch9)	chnl_sw_request (ch8)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
Bit symbol	chnl_sw_request (ch7)	chnl_sw_request (ch6)	chnl_sw_request (ch5)	chnl_sw_request (ch4)	chnl_sw_request (ch3)	chnl_sw_request (ch2)	chnl_sw_request (ch1)	chnl_sw_request (ch0)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit symbol	Type	Function
31-0	chnl_sw_request	W	DMA request 0: A transfer request does not occur. 1: A transfer request occurs. Specifies transfer requests to the each channel.

8.3.7 DMAxChnlUseburstSet (Channel useburst Set Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_useburst _set (ch31)	chnl_useburst _set (ch30)	chnl_useburst _set (ch29)	chnl_useburst _set (ch28)	chnl_useburst _set (ch27)	chnl_useburst _set (ch26)	chnl_useburst _set (ch25)	chnl_useburst _set (ch24)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	chnl_useburst _set (ch23)	chnl_useburst _set (ch22)	chnl_useburst _set (ch21)	chnl_useburst _set (ch20)	chnl_useburst _set (ch19)	chnl_useburst _set (ch18)	chnl_useburst _set (ch17)	chnl_useburst _set (ch16)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	chnl_useburst _set (ch15)	chnl_useburst _set (ch14)	chnl_useburst _set (ch13)	chnl_useburst _set (ch12)	chnl_useburst _set (ch11)	chnl_useburst _set (ch10)	chnl_useburst _set (ch9)	chnl_useburst _set (ch8)
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	chnl_useburst _set (ch7)	chnl_useburst _set (ch6)	chnl_useburst _set (ch5)	chnl_useburst _set (ch4)	chnl_useburst _set (ch3)	chnl_useburst _set (ch2)	chnl_useburst _set (ch1)	chnl_useburst _set (ch0)
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-0	chnl_useburst_set	R/W	Single-transfer is disabled [Write] 1: Single-transfer is disabled. [Read] 0: Single-transfer is enabled. 1: Single-transfer is disabled. Each bit corresponds to the channels in the specified number. Writing "1" disables the single-transfer to the corresponding channel, and only burst transfer request becomes valid. Writing "0" has no meaning. Set the DMAxChnlUseburstClr register in order to cancel the disabled the single-transfer. By reading the bit, the channel state of the corresponding bit can be checked whether it is enabled or disabled. Bits are automatically set in the following conditions: ・ This bit is cleared to "0", if the number of remaining transfer is less than 2^R times at the end of second 2^R time transfer from the end ("R" is specified by the channel_cfg<R_power> of the control data). ・ If the channel_cfg<next_useburst> of the control data is set to "1" in the peripheral scatter/gather mode, this bit is set to "1" when the DMA transfer of the alternative data ends.

Note: Do not set this bit to "1" if you do not use the burst transfer request on the condition where the number of transfers is less than 2^R times.

8.3.8 DMAxChnlUseburstClr (Channel useburst Clear Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_useburst _clr (ch31)	chnl_useburst _clr (ch30)	chnl_useburst _clr (ch29)	chnl_useburst _clr (ch28)	chnl_useburst _clr (ch27)	chnl_useburst _clr (ch26)	chnl_useburst _clr (ch25)	chnl_useburst _clr (ch24)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
Bit symbol	chnl_useburst _clr (ch23)	chnl_useburst _clr (ch22)	chnl_useburst _clr (ch21)	chnl_useburst _clr (ch20)	chnl_useburst _clr (ch19)	chnl_useburst _clr (ch18)	chnl_useburst _clr (ch17)	chnl_useburst _clr (ch16)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
Bit symbol	chnl_useburst _clr (ch15)	chnl_useburst _clr (ch14)	chnl_useburst _clr (ch13)	chnl_useburst _clr (ch12)	chnl_useburst _clr (ch11)	chnl_useburst _clr (ch10)	chnl_useburst _clr (ch9)	chnl_useburst _clr (ch8)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
Bit symbol	chnl_useburst _clr (ch7)	chnl_useburst _clr (ch6)	chnl_useburst _clr (ch5)	chnl_useburst _clr (ch4)	chnl_useburst _clr (ch3)	chnl_useburst _clr (ch2)	chnl_useburst _clr (ch1)	chnl_useburst _clr (ch0)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit symbol	Type	Function
31-0	chnl_useburst_clr	W	Single-transfer is enabled. 1: Enables the single-transfer. Each bit corresponds to the channels in the specified number. Writing "1" enables the single-transfer to the corresponding channel. Writing "0" has no meaning. To disable or confirm the signal-transfer, configure the DMAxChnlUseburstSet register.

8.3.9 DMAxChnlReqMaskSet (Channel Request Mask Set Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_req_mask_set (ch31)	chnl_req_mask_set (ch30)	chnl_req_mask_set (ch29)	chnl_req_mask_set (ch28)	chnl_req_mask_set (ch27)	chnl_req_mask_set (ch26)	chnl_req_mask_set (ch25)	chnl_req_mask_set (ch24)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	chnl_req_mask_set (ch23)	chnl_req_mask_set (ch22)	chnl_req_mask_set (ch21)	chnl_req_mask_set (ch20)	chnl_req_mask_set (ch19)	chnl_req_mask_set (ch18)	chnl_req_mask_set (ch17)	chnl_req_mask_set (ch16)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	chnl_req_mask_set (ch15)	chnl_req_mask_set (ch14)	chnl_req_mask_set (ch13)	chnl_req_mask_set (ch12)	chnl_req_mask_set (ch11)	chnl_req_mask_set (ch10)	chnl_req_mask_set (ch9)	chnl_req_mask_set (ch8)
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	chnl_req_mask_set (ch7)	chnl_req_mask_set (ch6)	chnl_req_mask_set (ch5)	chnl_req_mask_set (ch4)	chnl_req_mask_set (ch3)	chnl_req_mask_set (ch2)	chnl_req_mask_set (ch1)	chnl_req_mask_set (ch0)
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-0	chnl_req_mask_set	R/W	DMA request masking [Write] 1: Mask a DMA request [Read] 0: A DMA request is valid. 1: A DMA request is invalid. Each bit corresponds to the channels in the specified number. Writing "1" disables the single-transfer for the corresponding channel. Writing "0" has no meaning. To disable masking, configure the DMAxChnlReqMaskClr register. By reading the bit, the status of the DMA request setting can be checked whether it is enabled or disabled.

8.3.10 DMAxChnlReqMaskClr (Channel Request Mask Clear Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_req_mask_clr (ch31)	chnl_req_mask_clr (ch30)	chnl_req_mask_clr (ch29)	chnl_req_mask_clr (ch28)	chnl_req_mask_clr (ch27)	chnl_req_mask_clr (ch26)	chnl_req_mask_clr (ch25)	chnl_req_mask_clr (ch24)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
Bit symbol	chnl_req_mask_clr (ch23)	chnl_req_mask_clr (ch22)	chnl_req_mask_clr (ch21)	chnl_req_mask_clr (ch20)	chnl_req_mask_clr (ch19)	chnl_req_mask_clr (ch18)	chnl_req_mask_clr (ch17)	chnl_req_mask_clr (ch16)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
Bit symbol	chnl_req_mask_clr (ch15)	chnl_req_mask_clr (ch14)	chnl_req_mask_clr (ch13)	chnl_req_mask_clr (ch12)	chnl_req_mask_clr (ch11)	chnl_req_mask_clr (ch10)	chnl_req_mask_clr (ch9)	chnl_req_mask_clr (ch8)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
Bit symbol	chnl_req_mask_clr (ch7)	chnl_req_mask_clr (ch6)	chnl_req_mask_clr (ch5)	chnl_req_mask_clr (ch4)	chnl_req_mask_clr (ch3)	chnl_req_mask_clr (ch2)	chnl_req_mask_clr (ch1)	chnl_req_mask_clr (ch0)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit symbol	Type	Function
31-0	chnl_req_mask_clr	W	DMA request mask clear 1: Clears the corresponding channel of the DMA request mask. Each bit corresponds to the channels in the specified number. Writing "1" disables the DMA request mask setting of the corresponding channel. Writing "0" has no meaning. Configure the DMAxChnlReqMaskSet register to enable and confirm the setting.

8.3.11 DMAxChnlEnableSet (Channel Enable Set Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_enable_set (ch31)	chnl_enable_set (ch30)	chnl_enable_set (ch29)	chnl_enable_set (ch28)	chnl_enable_set (ch27)	chnl_enable_set (ch26)	chnl_enable_set (ch25)	chnl_enable_set (ch24)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	chnl_enable_set (ch23)	chnl_enable_set (ch22)	chnl_enable_set (ch21)	chnl_enable_set (ch20)	chnl_enable_set (ch19)	chnl_enable_set (ch18)	chnl_enable_set (ch17)	chnl_enable_set (ch16)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	chnl_enable_set (ch15)	chnl_enable_set (ch14)	chnl_enable_set (ch13)	chnl_enable_set (ch12)	chnl_enable_set (ch11)	chnl_enable_set (ch10)	chnl_enable_set (ch9)	chnl_enable_set (ch8)
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	chnl_enable_set (ch7)	chnl_enable_set (ch6)	chnl_enable_set (ch5)	chnl_enable_set (ch4)	chnl_enable_set (ch3)	chnl_enable_set (ch2)	chnl_enable_set (ch1)	chnl_enable_set (ch0)
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-0	chnl_enable_set	R/W	DMA operation [Write] 1: Enable the corresponding channel. [Read] 0: The corresponding bit is invalid. 1: The corresponding bit is valid. Each bit corresponds to the channels in the specified number. Writing "1" enables the corresponding channels. Writing "0" has no meaning. To disable the setting, configure the DMAxChnlEnableClr register. By reading the bit, the corresponding channel can be checked whether it is enabled or disabled. In the following conditions, the function automatically becomes invalid. ・DMA cycle ends. ・If the channel_cfg<cycle_ctrl> reads the control data of "000". ・A bus error occurs.

8.3.12 DMAxChnlEnableClr (Channel Enable Clear Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_enable_clr (ch31)	chnl_enable_clr (ch30)	chnl_enable_clr (ch29)	chnl_enable_clr (ch28)	chnl_enable_clr (ch27)	chnl_enable_clr (ch26)	chnl_enable_clr (ch25)	chnl_enable_clr (ch24)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
Bit symbol	chnl_enable_clr (ch23)	chnl_enable_clr (ch22)	chnl_enable_clr (ch21)	chnl_enable_clr (ch20)	chnl_enable_clr (ch19)	chnl_enable_clr (ch18)	chnl_enable_clr (ch17)	chnl_enable_clr (ch16)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
Bit symbol	chnl_enable_clr (ch15)	chnl_enable_clr (ch14)	chnl_enable_clr (ch13)	chnl_enable_clr (ch12)	chnl_enable_clr (ch11)	chnl_enable_clr (ch10)	chnl_enable_clr (ch9)	chnl_enable_clr (ch8)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
Bit symbol	chnl_enable_clr (ch7)	chnl_enable_clr (ch6)	chnl_enable_clr (ch5)	chnl_enable_clr (ch4)	chnl_enable_clr (ch3)	chnl_enable_clr (ch2)	chnl_enable_clr (ch1)	chnl_enable_clr (ch0)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit symbol	Type	Function
31-0	chnl_enable_clr	W	DMA disabled 1: Disables the corresponding channel. Each bit corresponds to the channels in the specified number. Writing "1" disables the corresponding channel. Writing "0" has no meaning. Configure the DMAxChnlEnableSet register in order to enable and confirm the setting. In the following conditions, the function automatically becomes invalid. ・ DMA cycle ends. ・ The channel_cfg<cycle_ctrl> reads the control data of "000". ・ A bus error occurs.

8.3.13 DMAxChnlPriAltSet (Channel Primary-alternate Set Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_pri_alt_set (ch31)	chnl_pri_alt_set (ch30)	chnl_pri_alt_set (ch29)	chnl_pri_alt_set (ch28)	chnl_pri_alt_set (ch27)	chnl_pri_alt_set (ch26)	chnl_pri_alt_set (ch25)	chnl_pri_alt_set (ch24)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	chnl_pri_alt_set (ch23)	chnl_pri_alt_set (ch22)	chnl_pri_alt_set (ch21)	chnl_pri_alt_set (ch20)	chnl_pri_alt_set (ch19)	chnl_pri_alt_set (ch18)	chnl_pri_alt_set (ch17)	chnl_pri_alt_set (ch16)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	chnl_pri_alt_set (ch15)	chnl_pri_alt_set (ch14)	chnl_pri_alt_set (ch13)	chnl_pri_alt_set (ch12)	chnl_pri_alt_set (ch11)	chnl_pri_alt_set (ch10)	chnl_pri_alt_set (ch9)	chnl_pri_alt_set (ch8)
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	chnl_pri_alt_set (ch7)	chnl_pri_alt_set (ch6)	chnl_pri_alt_set (ch5)	chnl_pri_alt_set (ch4)	chnl_pri_alt_set (ch3)	chnl_pri_alt_set (ch2)	chnl_pri_alt_set (ch1)	chnl_pri_alt_set (ch0)
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-0	chnl_pri_alt_set	R/W	Selects primary data or alternative data [Write] 1: Uses alternative data [Read] 0: Primary data 1: Alternative data Each bit corresponds to the channels in the specified number. Writing "1" specifies the data that is firstly used in the corresponding channel as "alternative data". Writing "0" has no meaning. To disable this bit, use the DMAxChnlEnableClr register. Only in basic mode, automatic request mode, and ping-pong mode the first data can be specified as alternative data. When this bit is read, data of the corresponding channel can be checked whether data is primary data or alternative data. In the following conditions, the settings are automatically changed. ・ The primary data transfer is completed in ping-pong mode, memory scatter / gather mode or peripheral scatter / gather mode. ・ Data transfer of the alternative data is completed in the ping-pong mode, memory scatter / gather mode or peripheral scatter / gather mode.

8.3.14 DMAxChnlPriAltClr (Channel Primary-alternate Clear Register)

	31	30	29	28	27	26	25	24
Bit symbol	chn_pri_alt_clr (ch31)	chn_pri_alt_clr (ch30)	chn_pri_alt_clr (ch29)	chn_pri_alt_clr (ch28)	chn_pri_alt_clr (ch27)	chn_pri_alt_clr (ch26)	chn_pri_alt_clr (ch25)	chn_pri_alt_clr (ch24)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
Bit symbol	chn_pri_alt_clr (ch23)	chn_pri_alt_clr (ch22)	chn_pri_alt_clr (ch21)	chn_pri_alt_clr (ch20)	chn_pri_alt_clr (ch19)	chn_pri_alt_clr (ch18)	chn_pri_alt_clr (ch17)	chn_pri_alt_clr (ch16)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
Bit symbol	chn_pri_alt_clr (ch15)	chn_pri_alt_clr (ch14)	chn_pri_alt_clr (ch13)	chn_pri_alt_clr (ch12)	chn_pri_alt_clr (ch11)	chn_pri_alt_clr (ch10)	chn_pri_alt_clr (ch9)	chn_pri_alt_clr (ch8)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
Bit symbol	chn_pri_alt_clr (ch7)	chn_pri_alt_clr (ch6)	chn_pri_alt_clr (ch5)	chn_pri_alt_clr (ch4)	chn_pri_alt_clr (ch3)	chn_pri_alt_clr (ch2)	chn_pri_alt_clr (ch1)	chn_pri_alt_clr (ch0)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit symbol	Type	Function
31-0	chnl_pri_alt_clr	W	Clears the alternative data setting. 1: Uses the primary data Each bit corresponds to the channels in the specified number. Writing "1" sets the data of the corresponding channel to the primary data. Setting "0" is invalid. Configure the DMAxChnlPriAltSet register to set the primary data or to confirm the setting. In the following conditions, the setting is automatically changed. ・ The primary data transfer in memory scatter/gather mode or peripheral scatter/gather mode is complete. ・ The primary data transfer in ping-pong mode is complete. ・ The alternative transfer in ping-pong mode, memory scatter/gather mode, or peripheral scatter/gather mode is complete.

8.3.15 DMAxChnlPrioritySet (Channel Priority Set Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_priority_ set (ch31)	chnl_priority_ set (ch30)	chnl_priority_ set (ch29)	chnl_priority_ set (ch28)	chnl_priority_ set (ch27)	chnl_priority_ set (ch26)	chnl_priority_ set (ch25)	chnl_priority_ set (ch24)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	chnl_priority_ set (ch23)	chnl_priority_ set (ch22)	chnl_priority_ set (ch21)	chnl_priority_ set (ch20)	chnl_priority_ set (ch19)	chnl_priority_ set (ch18)	chnl_priority_ set (ch17)	chnl_priority_ set (ch16)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	chnl_priority_ set (ch15)	chnl_priority_ set (ch14)	chnl_priority_ set (ch13)	chnl_priority_ set (ch12)	chnl_priority_ set (ch11)	chnl_priority_ set (ch10)	chnl_priority_ set (ch9)	chnl_priority_ set (ch8)
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	chnl_priority_ set (ch7)	chnl_priority_ set (ch6)	chnl_priority_ set (ch5)	chnl_priority_ set (ch4)	chnl_priority_ set (ch3)	chnl_priority_ set (ch2)	chnl_priority_ set (ch1)	chnl_priority_ set (ch0)
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-0	chnl_priority_set	R/W	Priority settings [Write] 1: Sets the high-priority [Read] 0: Normal priority 1: High priority Each bit corresponds to the channels in the specified number. Writing "1" sets the priority of the corresponding channel high. Writing "0" has no meaning.To change the priority again to the normal, configure the DMAxChnlPriorityClr register. the priority of the corresponding channel, high-priority or normal priority, can be confirmed by reading the bit.

8.3.16 DMAxChnlPriorityClr (Channel Priority Clear Register)

	31	30	29	28	27	26	25	24
Bit symbol	chnl_priority_clr (ch31)	chnl_priority_clr (ch30)	chnl_priority_clr (ch29)	chnl_priority_clr (ch28)	chnl_priority_clr (ch27)	chnl_priority_clr (ch26)	chnl_priority_clr (ch25)	chnl_priority_clr (ch24)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
Bit symbol	chnl_priority_clr (ch23)	chnl_priority_clr (ch22)	chnl_priority_clr (ch21)	chnl_priority_clr (ch20)	chnl_priority_clr (ch19)	chnl_priority_clr (ch18)	chnl_priority_clr (ch17)	chnl_priority_clr (ch16)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
Bit symbol	chnl_priority_clr (ch15)	chnl_priority_clr (ch14)	chnl_priority_clr (ch13)	chnl_priority_clr (ch12)	chnl_priority_clr (ch11)	chnl_priority_clr (ch10)	chnl_priority_clr (ch9)	chnl_priority_clr (ch8)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
Bit symbol	chnl_priority_clr (ch7)	chnl_priority_clr (ch6)	chnl_priority_clr (ch5)	chnl_priority_clr (ch4)	chnl_priority_clr (ch3)	chnl_priority_clr (ch2)	chnl_priority_clr (ch1)	chnl_priority_clr (ch0)
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit symbol	Type	Function
31-0	chnl_priority_clr	W	Clears the high-priority setting. [Write] 1:Sets normal priority setting Each bit corresponds to the channels in the specified number. Writing "1" changes the priority of the corresponding channel to normal priority. Writing "0" has no meaning. Configure the DMAxChnlPrioritySet register to set the high-priority and to confirm the setting.

8.3.17 DMAxErrClr (Bus Error 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	-	-	-	-	-	-	-	err_clr
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-1	-	R	Read as "0".
0	err_clr	R/W	Bus error [Write] 1: Clears a bus error. [Read] 0: No bus error 1: The state of a bus error A bus error occurrence can be confirmed by reading the bit. Writing "1" clears a bus error. Writing "0" has no meaning.

8.3.18 DMAIFFLGx (DMA Flag Register)

	31	30	29	28	27	26	25	24
Bit symbol	FLG31	FLG30	FLG29	FLG28	FLG27	FLG26	FLG25	FLG24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	FLG23	FLG22	FLG21	FLG20	FLG19	FLG18	FLG17	FLG16
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	FLG15	FLG14	FLG13	FLG12	FLG11	FLG10	FLG9	FLG8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	FLG7	FLG6	FLG5	FLG4	FLG3	FLG2	FLG1	FLG0
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Functions
31-1	FLG31 - FLG0	R	DMA factor flag 0:- 1: A completion interrupt occurs. A bit number corresponds with a channel number. If a transfer completion interrupt occurs, the corresponding bit is set to "1". This register is cleared by reading automatically.

Note: Before a transfer completion interrupt is enabled, this register must be read and cleared.

8.4 Operation

This DMA is controlled by the channel control data, which locates on the memory. A channel of the each data is four words and allocated in the contiguous areas same as the number of channels.

There are two types of channel control data: primary data and alternative data. According to the operation mode, one of them is selected by setting the register or both data is used.

8.4.1 Channel Control Data Memory Map

Figure 8-2 shows the example of memory map of the channel control data.

Set the start address of the primary data to the DMAxCtrlBasePtr and the start address of the alternative data to the DMAxAltCtrlBasePtr.

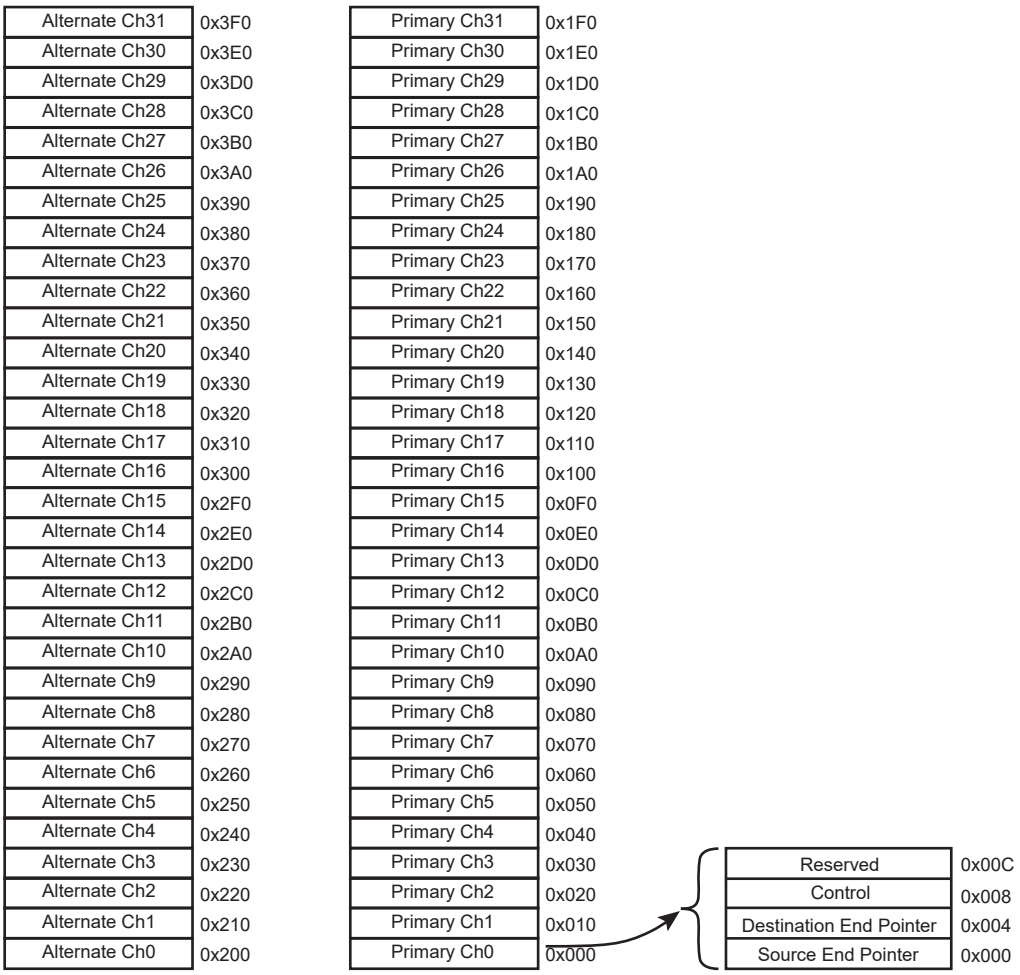


Figure 8-2 Memory map of the control data

Figure 8-2 shows the memory map of which all 32 channels can be used. Necessary areas are determined by the number of usable channels. Table 8-2 shows the relationship between the number of channels and addresses.

Table 8-2 Address bit setting of channel control

Channel	Address							Settable base address
	[9]	[8]	[7]	[6]	[5]	[4]	[3:0]	
0	-	-	-	-	-	A	Channel control data setting	0xXXXX_XX00, 0xXXXX_XX20, 0xXXXX_XX40, 0xXXXX_XX60, 0xXXXX_XX80, 0xXXXX_XXA0, 0xXXXX_XXC0, 0xXXXX_XXE0
0 to 1	-	-	-	-	A	C[0]		0xXXXX_XX00, 0xXXXX_XX40, 0xXXXX_XX80, 0xXXXX_XXC0
0 to 3	-	-	-	A	C[1:0]			0xXXXX_XX00, 0xXXXX_XX80
0 to 7	-	-	A	C[2:0]				0xXXXX_X000, 0xXXXX_X100, 0xXXXX_X200, 0xXXXX_X300, 0xXXXX_X400, 0xXXXX_X500, 0xXXXX_X600, 0xXXXX_X700, 0xXXXX_X800, 0xXXXX_X900, 0xXXXX_XA00, 0xXXXX_XB00, 0xXXXX_XC00, 0xXXXX_XD00, 0xXXXX_XE00, 0xXXXX_XF00
0 to 15	-	A	C[3:0]					0xXXXX_X000, 0xXXXX_X200, 0xXXXX_X400, 0xXXXX_X600, 0xXXXX_X800, 0xXXXX_XA00, 0xXXXX_XC00, 0xXXXX_XE00
0 to 31	A	C[4:0]						0xXXXX_X000, 0xXXXX_X400, 0xXXXX_X800, 0xXXXX_XC00

A: Primary/alternative setting (0:primary, 1:alternative)
C[x:0]: Channel number setting

8.4.2 Channel Control Data Structure

The channel control data contains the three kinds of data shown below:

- The final address of the transfer source address
- The final address of the transfer destination address
- Control data

Each data is described in the following sections:

8.4.2.1 Final Address of the Transfer Source Data

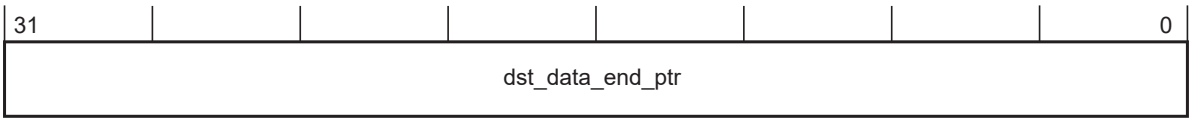
Specify the final address of the data to be transferred. The alignment of an address should be adjusted to a transfer data size. The DMA calculates the start address of the source address using this data.

31							0
src_data_end_ptr							

bit	Bit symbol	Function
[31:0]	src_data_end_ptr	The final address of source transfer data

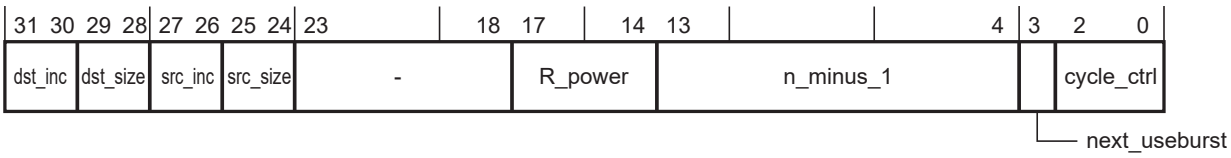
8.4.2.2 Final Address of the Transfer Destination Address

Specify the final address of the destination address. The alignment of an address should be adjusted to a transfer data size. The DMA calculates the start address of the destination address of the transfer destination address.



bit	Bit symbol	Function
[31:0]	dst_data_end_ptr	The final address of the transfer destination address.

8.4.2.3 Control Data Setting



bit	Bit symbol	Function
[31:30]	dst_inc	Increments the transfer destination address (note 2) 00: 1 byte 01: 2 bytes 10: 4 bytes 11: No increment
[29:28]	dst_size	Data size of transfer destination (note1) 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved
[27:26]	src_inc	Increments the transfer source address (note 2) 00: 1 byte 01: 2 bytes 10: 4 bytes 11: No increment
[25:24]	src_size	Data size of transfer source (note 1) 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved
[23:18]	-	Set "000000".

bit	Bit symbol	Function
[17:14]	R_power	Arbitration 0000: After 1 transfer 0001: After 2 transfers 0010: After 4 transfers 0011: After 8 transfers 0100: After 16 transfers 0101: After 32 transfers 0110: After 64 transfers 0111: After 128 transfers 1000: After 256 transfers 1001: After 512 transfers 1010 - 1111: No arbitration A transfer request is checked after the specified number of transfers. If a higher-priority request exists, the controller arbitrates the DMA transfer.
[13:4]	n_minus_1	The number of transfers 0x000: Once 0x001: Twice 0x002: Three times : 0x3FF: 1024 times
[3]	next_useburst	Changes the setting of single-transfer 0: Do not change the value of <chnl_useburst_set>. 1: Sets <chnl_useburst_set> to "1". Specifies whether to set "1" to the <chnl_useburst_set> bit at the end of the DMA transfer using alternative data in the peripheral scatter/ gather mode. Note) This bit <chnl_useburst_set> is zero cleared, if the number of remaining transfer is less than 2^R times at the end of second 2^Rtime transfer from the end ("R" is specified by the <R_power>). Setting this bit to "1" sets "1" to the <chnl_userburst_set>.
[2:0]	cycle_ctrl	Operation mode 000: Invalid. The DMA stops the operation. 001: Basic mode 010: Automatic request mode 011: Ping-pong mode 100: Memory scatter / gather mode (primary data) 101: Memory scatter / gather mode (alternative data) 110: Peripheral memory scatter / gather mode (primary data) 111: Peripheral memory scatter / gather mode (alternative data)

Note 1: The setting value of <dst_size> must be the same as <src_size>.

Note 2: According to the settings of <dst_size> and <src_size>, the settings of <dst_inc> and <src_inc> are limited as shown below:

<src_inc>/<dst_inc>	<src_size>/<dst_size>		
	00 (1 byte)	01 (2 bytes)	10 (4 bytes)
00 (1byte)	o	-	-
01 (2bytes)	o	o	-
10 (4bytes)	o	o	o
No increment	o	o	o

8.4.3 Operation Modes

This section describes the operation modes configured by channel_cfg<cycle_ctrl> of the channel control data.

8.4.3.1 Invalid Setting

The DMA sets the operation mode invalid after the end of transfer. This operation prevents a transfer from being performed again. Also, the operation completes if invalid data is read either in ping-pong mode, memory scatter / gather mode or peripheral scatter / gather mode.

8.4.3.2 Basic Mode

In basic mode, data structure can be selected from primary data or alternative data.

A transfer is started by receiving a transfer request.

An arbitration is performed for every transfer configured by <R_power>. If a higher-priority request exits, the DMA switches a channel. If a transfer request for the operating channel is received, the transfer is continued.

After performing transfers for the number of times specified by <n_minus_1>, a transfer completion interrupt occurs.

8.4.3.3 Automatic Request Mode

In this mode, a single-transfer request stops the DMA transfer. The data structure can be selected from primary data or alternative data.

The DMA transfer is started by a transfer request.

In each transfer configured by <R_power>, a channel is switched if a higher-priority request is received. If not, the transfer is continued.

After performing transfers for the number of times specified by <n_minus_1>, a transfer completion interrupt occurs.

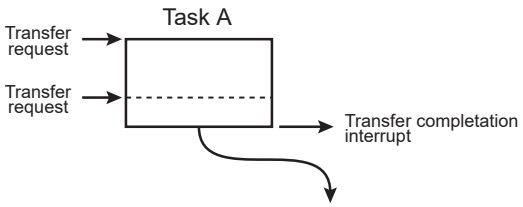
8.4.3.4 Ping-pong Mode

In ping-pong mode, a continuous DMA transfer that uses primary data and alternative data alternately is performed. If <cycle_ctrl> reads data specified to be invalid ("000"), or the channel is specified to be invalid, the transfer is stopped. Every time a DMA transfer (task) that uses primary data or alternative data is complete, a transfer completion interrupt occurs.

Preparation:

Prepare primary data and alternative data, and set "1" to the bits of the channels corresponding to both DMAxCfg<master_enable> and DMAxChnlEnableSet.

Task A: Primary data
<cycle_ctrl[2:0]> = "011"
(ping-pong mode)
<R_power[3:0]> = "0010"
(4 times)
<n_minus_1[9:0]> = "0x005" (6 times)



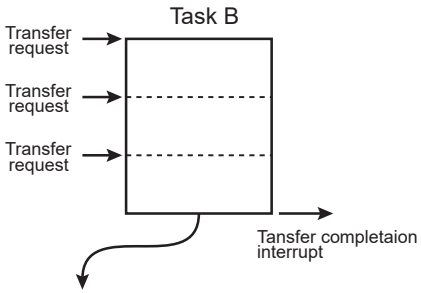
Receiving a transfer request, the DMA performs a transfer four times and performs arbitration.

If there is no other high-priority requests, the DMA performs remaining transfers twice toward a request for a transfer to the corresponding channels.

The DMA generates a transfer completion interrupt request and performs an arbitration.

After completing Task A, primary data for Task C can be set.

Task B: Alternative data
<cycle_ctrl[2:0]> = "011"
<R_power[3:0]> = "0010"
(4 times)
<n_minus_1[9:0]> = "0x00B" (12 times)



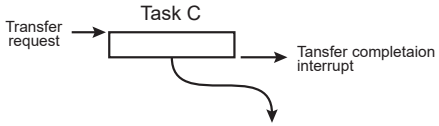
Receiving a transfer request, The DMA performs a transfer four times and performs arbitration.

If there is no other high-priority requests, The DMA performs transfers twice toward a request for a transfer to the corresponding channels.

The DMA generates a transfer completion interrupt request and performs an arbitration.

After completing Task B, alternative data for Task D can be set.

Task C: Primary data
<cycle_ctrl[2:0]> = "011"
<R_power[3:0]> = "0001"
(2 times)
<n_minus_1[9:0]> = "0x001" (2 times)

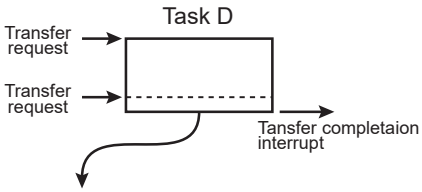


Receiving a transfer request, the DMA performs a transfer twice and performs arbitration.

The DMA generates a transfer completion interrupt request and performs an arbitration.

After completing Task C, alternative data for Task E can be set.

Task D: Alternative data
<cycle_ctrl[2:0]> = "011"
<R_power[4:0]> = "0010"
(4 times)
<n_minus_1[9:0]> = "0x004" (5 times)

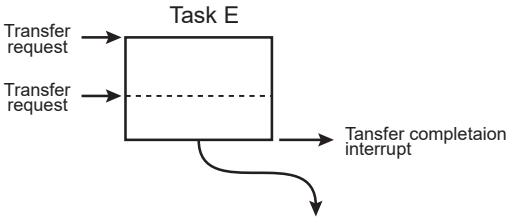


Receiving a transfer request, the DMA performs a transfer four times and performs arbitration.

If there is no other high-priority requests, the DMA performs a transfer once toward a request for a transfer to the corresponding channels.

The DMA generates a transfer completion interrupt request and performs an arbitration.

Task E: Primary data
<cycle_ctrl[2:0]> = "011"
<R_power[3:0]> = "0010"
(4 times)
<n_minus_1[9:0]> = "0x006" (7 times)

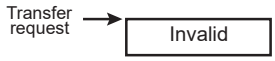


Receiving a transfer request, the DMA performs a transfer four times and performs arbitration.

If there is no other high-priority requests, the DMA performs transfers three times toward a request for a transfer to the corresponding channels.

The DMA generates a transfer completion interrupt request and performs an arbitration.

Final: Alternative data
<cycle_ctrl[2:0]> = "000"
(invalid)



Even receiving a transfer request, the operation stops because <cycle_ctrl[2:0]> is set to invalid.

(The operation can be also stopped by setting the <cycle_ctrl[2:0]> of Task E to normal mode "001".)

8.4.3.5 Memory Scatter/Gather Mode

In memory scatter/gather mode, primary data is used in order to transfer data for alternative data.

Receiving a transfer request, the DMA transfers four alternative data using primary data. If there is no new requests, it starts data transferring using alternative data. Then, it keeps transferring alternative data using primary data and transfer using alternative data, until either invalid setting ("000") of the <cycle_ctrl [2:0]> or setting data of the basic mode ("001") is read. A new transfer request is not required during this period. After the transfer operation, an interrupt is generated.

The settings of the channel_cfg of primary data must be configured as shown below:

Table 8-3 Setting values of Memory scatter/gather mode (Primary data)

Bit	Bit symbol	Setting values	Description
[31:30]	dst_inc	10	4-byte increment is specified for transfer destination address.
[29:28]	dst_size	10	4 bytes are specified as transfer destination address.
[27:26]	src_inc	10	4-byte increment is specified for transfer source address.
[25:24]	src_size	10	4 bytes are specified as transfer source address.
[17:14]	R_power	0010	4 is specified as arbitration cycle.
[13:4]	n_minus_1	N	The number of alternative task to be prepared ×4 is specified.
[3]	next_useburst	0	"0" is specified in memory scatter/gather mode.
[2:0]	cycle_ctrl	100	Memory scatter/gather mode (primary data) is specified. note)

Note: If the transfers specified in the <n_minus_1> are complete, invalid data "000" is automatically set.

Preparation:

Prepare primary data. Set "100" to <cycle_ctrl> and set four task data 4 × 4 = 16 as the number of transfers <n_minus_1>.

Set alternative data for Task A,B,C and D to the memory location which is set to the <src_data_end_ptr>.

Set "1" to bits of channels corresponding to DMAxCfg <master_enable> and DMAxChnlSet.

Copy A: Primary data

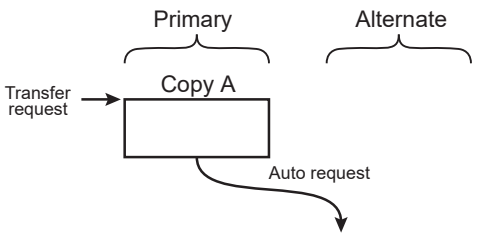
<cycle_ctrl[2:0]> = "100"

(Memory scatter / gather mode)

<R_power[3:0]> = "0010"

(4 times)

<n_minus_1[9:0]> = "0x00F" (16 times)



Receiving a transfer request, the DMA performs a transfer for alternative data of Task A for four times.

After completing the transfer, a transfer request is automatically generated and arbitration starts.

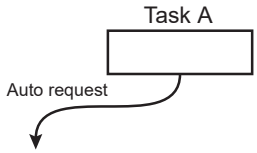
Task A: Alternative data

<cycle_ctrl[2:0]> = "100"

<R_power[3:0]> = "0010"

(4 times)

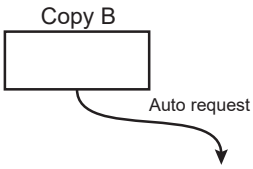
<n_minus_1[9:0]> = "0x002" (3 times)



The DMA performs Task A.

After completing the transfer, a transfer request is automatically generated and arbitration starts.

Copy B: Primary data



The DMA performs transfers for alternative data of Task B for four times.

After completing the transfer, a transfer request is automatically generated and arbitration starts.

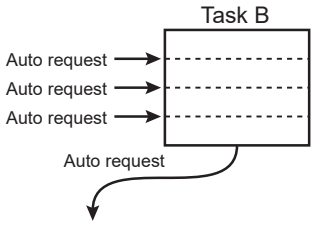
Task B: Alternative data

<cycle_ctrl[2:0]> = "100"

<R_power[3:0]> = "0001"

(2 times)

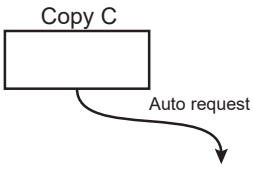
<n_minus_1[9:0]> = "0x007" (8 times)



The DMA performs Task B.

After completing the transfer, a transfer request is automatically generated and arbitration starts.

Copy C: Primary data



The DMA performs transfers for alternative data of Task C for four times.

After completing the transfer, a transfer request is automatically generated and arbitration starts.

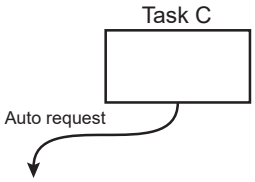
Task C: Alternative data

<cycle_ctrl[2:0> = "100"

<R_power[3:0]> = "0011"

(8 times)

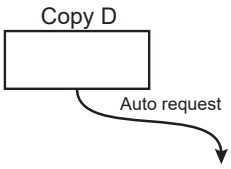
<n_minus_1[9:0]> = "0x004" (5 times)



The DMA performs Task C.

After completing the transfer, a transfer request is automatically generated and arbitration starts.

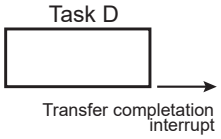
Copy D: Primary data



The DMA performs transfers for alternative data of Task D for four times. The DMA also sets "000" to <cycle_ctrl> of the primary data in order to set the next primary data invalid.

A transfer request is automatically generated and arbitration starts.

Task D: Alternative data
<cycle_ctr[2:0]l> = "001"
<R_power[3:0]> = "0010"
(4 times)
<n_minus_1[9:0]> = "0x003" (4 times)



The DMA performs Task D.
Since <cycle_ctrl[2:0]> is set to the basic mode "001", the DMA generates a transfer completion interrupt request after the end of the transfer, and completes the operation.

8.4.3.6 Peripheral Scatter/Gather Mode

Primary data is used in order to transfer data for alternative data in peripheral scatter/gather mode.

Receiving a transfer request, the DMA transfers four alternative data using primary data, and then starts transfer using alternative data.

After that, if a transfer request is generated, it starts alternative data transferring using primary data. Then, it keeps transferring alternative data using primary data and transfer using alternative data, until either invalid setting ("000") of the <cycle_ctrl> or setting data of the basic mode ("001") is read. A new transfer request is not required during this period. After the transfer operation, an interrupt is generated.

The settings of the channel_cfg of primary data must be configured as shown below:

Table 8-4 Fixed values in peripheral scatter / gather mode (Primary data)

Bit	Bit symbol	Setting value	Description
[31:30]	dst_inc	10	A 4-byte increment is specified for transfer destination address.
[29:28]	dst_size	10	4 bytes are specified as transfer destination address.
[27:26]	src_inc	10	A 4-byte increment is specified for transfer source address.
[25:24]	src_size	10	4 bytes are specified as transfer source address.
[17:14]	R_power	0010	4 is specified as arbitration cycle.
[13:4]	n_minus_1	N	The number of alternative task to be prepared ×4 is specified.
[2:0]	cycle_ctrl	110	Specify peripheral scatter/gather mode (Primary data).

Note: If the transfers specified in the <n_minus_1> are complete, invalid data "000" is automatically set.

Preparation:

Prepare primary data. Set "110" to <cycle_ctrl> and 4 × 4 = 16 for four tasks to the number of transfers <n_minus_1>.

Set alternative data for Task A,B,C and D to the memory location which is set to the <src_data_end_ptr>.

Set "1" to bits of channels corresponding to DMAxCfg <master_enable> and DMAxChnlEnableSet.

Copy A: Primary data

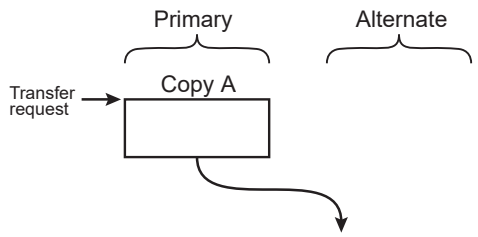
<cycle_ctrl[2:0]> = "110"

(Peripheral scatter / gather)

<R_power[3:0]> = "0010"

(4 times)

<n_minus_1[9:0]> = "0x00F" (16 times)



Receiving a transfer request, the DMA performs transfers for alternative data of Task A for four times.

After completing the transfer, operation automatically moves onto Task A.

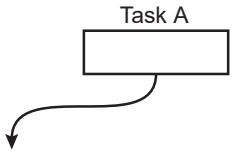
Task A: Alternative data

<cycle_ctrl[2:0]> = "111"

<R_power[3:0]> = "0010"

(4 times)

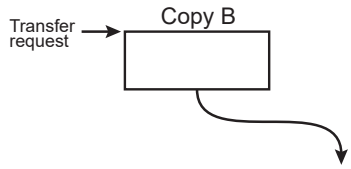
<n_minus_1[9:0]> = "0x002" (3 times)



The DMA performs Task A.

After completing the transfer, if a transfer request is sent from peripheral function and if it is high-priority request, the next operation starts.

Copy B: Primary data



The DMA performs transfers for alternative data of Task B for four times.

After completing the transfer, processing of Task B automatically starts.

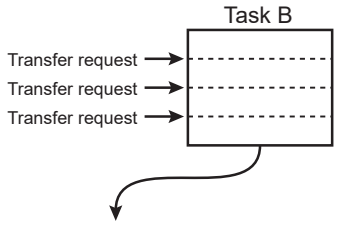
Task B: Alternative data

<cycle_ctrl[2:0]> = "111"

<R_power[3:0]> = "0001"

(2 times)

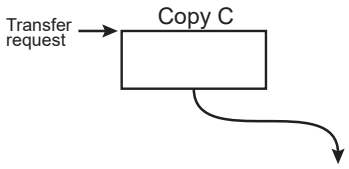
<n_minus_1[9:0]> = "0x007" (8 times)



The DMA performs Task B. Since an arbitration occurs every 2^R times of transfers, three times of transfer is required at least to complete Task B.

After completing the transfer, if a transfer request is sent from peripheral function and if it is high-priority request, the next operation starts

Copy C: Primary data



The DMA performs transfers for alternative data of Task C for four times.

After completing the transfer, operation automatically moves onto Task C.

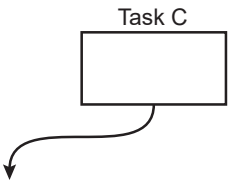
Task C: Alternative data

<cycle_ctrl[2:0]> = "111"

<R_power[3:0]> = "0011"

(8 times)

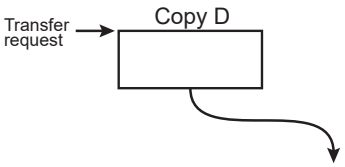
<n_minus_1[9:0]> = "0x004" (5 times)



The DMA performs Task C.

After completing the transfer, if a transfer request is sent from peripheral function and if it is high-priority request, the next operation starts.

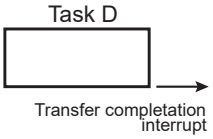
Copy D: Primary data



The DMA performs transfers for alternative data of Task D for four times. Also, The DMA performs transfers for alternative data for four times. Sets "000" to <cycle_ctrl> of the primary data and makes the next primary data invalid.

The operation automatically moves onto Task D.

Task D: Alternative data
<cycle_ctrl[2:0]> = "001"
<R_power[3:0]> = "0010"
(4 times)
<n_minus_1[9:0]> = "0x003" (4 times)



The DMA performs Task D.
Since <cycle_ctrl> is set to the basic mode "001", The DMA generates a transfer completion interrupt request after the end of the transfer, and completes the operation.

8.5 Precautions

Extra caution should be exercised when a DMA transfer request is used in the following peripheral functions:

- Synchronous serial interface (SSP)
- Asynchronous serial communication circuit (UART)
- Serial channel with 4-byte FIFO (SIO/UART)
- 16-bit timer/event counter (TMRB)
- Analog-to-digital converter (ADC)

8.5.1 When the SSP and UART are Used

When the SSP and UART are used, a water mark level and the number of transfers should be taken into consideration. Perform a transfer in transmission or reception as follows respectively:

- Transmission

  It is recommended to use a basic mode as a transfer mode.

  Disable the single-transfer.

  The following two methods are used according to the number of transfers:

  a. When a DMA transfer rate is set to "after 1 transfer" as an arbitration rate.

     This method can be used in any cases.

     Specify "0000" as the arbitration rate setting <R_power> for the control data.

  b. When the number of transfers is a multiple of the watermark level of the FIFO.

     This method can be used when the number of transfers is a multiple of the watermark level of the FIFO and the watermark level and the number of arbitration rate are the same number.

     Set the number of arbitration rates <R_power> for the control data to a multiple of the watermark level of the FIFO.

- Reception

  Use the SSP and UART according to the number of transfers as follows:

  a. Less than the watermark level

     Only a single-transfer request occurs.

     It is recommended to use a basic mode as a transfer mode.

     Specify "0000" as the arbitration rate <R_power> for the control data.

  b. A multiple of the watermark level

     Disable the single-transfer.

     It is recommended to use a basic mode as a transfer mode.

Set the number of arbitration rates <R_power> for the control data to a multiple of the watermark level of the FIFO.

c. Other than the above

Use the peripheral scatter/gather mode as a transfer mode.

Prepare two tasks.

Set the first task to the same setting as <b>. Disable the single-transfer. Set the number of arbitration rates <R_power> to a multiple of the watermark level of the FIFO. Perform DMA transfers up to the number of multiple of the watermark level.

Set the second task to the same setting as <a>. Enable the single-transfer. Specify "0000" as the arbitration setting <R_power>. Transfer the remaining data.

8.5.2 SIO/UART, TMRB, ADC are Used

The following points should be considered:

- It is recommended to use the basic mode as a transfer mode.
- Set "after 1 transfer" as a DMA transfer rate.

  Specify "0000" as the arbitration rate <R_power> for the control data.
- Do not use the FIFO of the SIO/UART.

  Use the SIO/UART with the single-buffer or double-buffer setting.

A new request occurs after the DMA transfer is started, only one transfer is performed. Design the program to perform a DMA transfer surely.

In case that transfer will not be started, the following circumstances can be expected:

- A higher priority transfer request occurs in the same unit
- A transfer destination conflict occurs between other higher bus master and a sender.

As a guide, this µDMA controller takes 11 clocks on pre-/post-processing. It takes approximately 5 clocks for a data transfer between the peripheral functions and internal RAM.

9. Input / Output port

This chapter describes port-related registers, their setting and circuits.

9.1 Registers

When the port registers are used, the following registers must be set.

All registers are 32 bits. The configurations are different depend on the number of port bits and assignation of the function.

"x" means the name of ports and "n" means the function number in the following description.

Register Name		Setting Value	
PxDATA	Data register	0 or 1	This register reads / writes port data.
PxCR	Output control register	0 : Output Disable 1 : Output Enable	This register controls output.
PxFRn	Function register n	0 : PORT 1 : Function	This register sets the function. The assigned function can be enabled by setting "1". This register exists for the each function assigned to the port. In case of having some function, only one function can be enabled.
PxOD	Open-drain control register	0 : CMOS 1 : Open-drain	This register controls programmable open-drain outputs. Programmable open-drain outputs are set with PxOD. When output data is "1", output buffer is disabled and becomes a pseudo-open-drain output.
PxPUP	Pull-up control register	0 : Pull-up Disable 1 : Pull-up Enable	This register controls programmable pull-ups.
PxPDN	Pull-down control register	0 : Pull-down Disable 1 : Pull-down Enable	This register controls programmable pull-downs.
PxIE	Input control register	0 : Input Disable 1 : Input Enable	This register controls inputs. Some time is required after enabling PxIE until external data is reflected in PxDATA.

9.1.1 Register list

For detail of the base address, refer to "Address lists of peripheral functions" of "Memory Map" chapter.

Register name	Address (Base+)	PORT A	PORT B	PORT C	PORT D	PORT E
Data register	0x0000	PADATA	PBDATA	PCDATA	PDDATA	PEDATA
Output control register	0x0004	PACR	PBCR	PCCR	PDCR	PECR
Function register 1	0x0008	PAFR1	PBFR1	PCFR1	PDFR1	PEFR1
Function register 2	0x000C	PAFR2	PBFR2	PCFR2	–	–
Function register 3	0x0010	PAFR3	PBFR3	–	–	PEFR3
Function register 4	0x0014	PAFR4	PBFR4	–	–	PEFR4
Function register 5	0x0018	PAFR5	PBFR5	–	–	PEFR5
Function register 6	0x001C	–	–	–	–	–
Open-drain control register	0x0028	PAOD	PBOD	PCOD	PDOD	PEOD
Pull-up control register	0x002C	PAPUP	PBPUP	PCPUP	PDPUP	PEPUP
Pull-down control register	0x0030	PAPDN	–	–	–	–
Input control register	0x0038	PAIE	PBIE	PCIE	PDIE	PEIE

Register name	Address (Base+)	PORT F	PORT G	PORT H	PORT J	PORT K
Data register	0x0000	PFDATA	PGDATA	PHDATA	PJDATA	PKDATA
Output control register	0x0004	PFCR	PGCR	PHCR	PJCR	PKCR
Function register 1	0x0008	PFFR1	PGFR1	PHFR1	PJFR1	–
Function register 2	0x000C	–	–	PHFR2	PJFR2	PKFR2
Function register 3	0x0010	PFFR3	PGFR3	PHFR3	PJFR3	PKFR3
Function register 4	0x0014	PFFR4	PGFR4	PHFR4	–	PKFR4
Function register 5	0x0018	PFFR5	PGFR5	–	–	–
Function register 6	0x001C	–	–	–	–	–
Open-drain control register	0x0028	PFOD	PGOD	PHOD	PJOD	PKOD
Pull-up control register	0x002C	PFPUP	PGPUP	PHPUP	PJPUP	PKPUP
Pull-down control register	0x0030	–	–	–	–	–
Input control register	0x0038	PFIE	PGIE	PHIE	PJIE	PKIE

Register name	Address (Base+)	PORT L
Data register	0x0000	PLDATA
Output control register	0x0004	PLCR
Function register 1	0x0008	–
Function register 2	0x000C	–
Function register 3	0x0010	PLFR3
Function register 4	0x0014	PLFR4
Function register 5	0x0018	PLFR5
Function register 6	0x001C	PLFR6
Open-drain control register	0x0028	PLOD
Pull-up control register	0x002C	PLPUP
Pull-down control register	0x0030	–
Input control register	0x0038	PLIE

Note: The address shown as "-" is not accessed.

9.1.2 Port function and setting list

The list of the function and setting register for each port is shown bellows.

- "Table 9-1 PORT A Setting List"
- "Table 9-2 PORT B Setting List"
- "Table 9-3 PORT C Setting List"
- "Table 9-4 PORT D Setting List"
- "Table 9-5 PORT E Setting List"
- "Table 9-6 PORT F Setting List"
- "Table 9-7 PORT G Setting List"
- "Table 9-8 PORT H Setting List"
- "Table 9-9 PORT J Setting List"
- "Table 9-10 PORT K Setting List"
- "Table 9-11 PORT L Setting List"

The cell of PxFRn shows the function register which must be set to select a function. If this register is set to "1", the corresponding function is enabled.

A bit in the cell filled with a hatch is read as "0" and the writing a data to this bit is invalid.

"0" or "1" in the table is shown the value which is set to the register. "0/1" is shown that the optional value can be set to the register.

9.1.2.1 PORT A

Table 9-1 PORT A Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PADATA	PACR	PAFRn	PAOD	PAPUP	PAPDN	PAIE
PA0	After reset (TDO/SWV)	Output	FT3	0	1	PAFR1	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	UT1DTR	Output	FT1	0/1	1	PAFR2	0/1	0/1		0
PA1	After reset (TMS/SWDIO)	I/O	FT3	0	1	PAFR1	0	1		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	UT1DSR	Input	FT1	0/1	0	PAFR2	0/1	0/1		1
PA2	After reset (TCK/SWCLK)	Input	FT3	0	0	PAFR1	0		1	1
	Input Port	Input		0/1	0	0	0/1		0/1	1
	Output Port	Output		0/1	1	0	0/1		0/1	0
	UT1RIN	Input	FT1	0/1	0	PAFR2	0/1		0/1	1
PA3	After reset (TDI)	Input	FT3	0	0	PAFR1	0	1		1
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	INT3	Input	FT4	0/1	0		0/1	0/1		1
	UT1DCD	Input	FT1	0/1	0	PAFR2	0/1	0/1		1
PA4	After reset ($\overline{TRST}$)	Input	FT9	0	0	PAFR1	0	1		1
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	$\overline{UT1RTS}$	Output	FT1	0/1	1	PAFR2	0/1	0/1		0
PA5	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	TRACECLK	Output	FT1	0/1	1	PAFR1	0/1	0/1		0
	UT1RXD	Input	FT1	0/1	0	PAFR2	0/1	0/1		1
	UT1IRIN	Input	FT1	0/1	0	PAFR3	0/1	0/1		1
PA6	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	TRACEDATA0	Output	FT1	0/1	1	PAFR1	0/1	0/1		0
	UT1TXD	Output	FT1	0/1	1	PAFR2	0/1	0/1		0
	UT1IROUT	Output	FT1	0/1	1	PAFR3	0/1	0/1		0
PA7	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	TRACEDATA1	Output	FT1	0/1	1	PAFR1	0/1	0/1		0
	$\overline{UT1CTS}$	Output	FT1	0/1	1	PAFR2	0/1	0/1		0
	SC3SCK	Input	FT1	0/1	0	PAFR3	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0
	$\overline{SC3CTS}$	Input	FT1	0/1	0	PAFR4	0/1	0/1		1

Table 9-1 PORT A Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PADATA	PACR	PAFRn	PAOD	PAPUP	PAPDN	PAIE
	TB7OUT	Output	FT1	0/1	1	PAFR5	0/1	0/1		0

9.1.2.2 PORT B

Table 9-2 PORT B Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PBDATA	PBCR	PBFRn	PBOD	PBPUP	PBPDN	PBIE
PB0	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	TRACEDATA2	Output	FT1	0/1	1	PBFR1	0/1	0/1		0
	SC3TXD	Output	FT1	0/1	1	PBFR3	0/1	0/1		0
PB1	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	TRACEDATA3	Output	FT1	0/1	1	PBFR1	0/1	0/1		0
	SC3RXD	Input	FT1	0/1	0	PBFR3	0/1	0/1		1
PB2	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	$\overline{WR}$	Output	FT1	0/1	1	PBFR1	0/1	0/1		0
	MT3OUT0	Output	FT2	0/1	1	PBFR3	0/1	0/1		0
	MT3TBOUT	Output	FT1	0/1	1	PBFR4	0/1	0/1		0
	$\overline{SNFCWE}$	Output	FT1	0/1	1	PBFR5	0/1	0/1		0
PB3	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	$\overline{RD}$	Output	FT1	0/1	1	PBFR1	0/1	0/1		0
	MT3OUT1	Output	FT2	0/1	1	PBFR3	0/1	0/1		0
	MT3TBIN	Input	FT1	0/1	0	PBFR4	0/1	0/1		1
	$\overline{SNFCRE}$	Output	FT1	0/1	1	PBFR5	0/1	0/1		0
PB4	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	$\overline{CS0}$	Output	FT1	0/1	1	PBFR1	0/1	0/1		0
	$\overline{GEMG3}$	Input	FT1	0/1	0	PBFR3	0/1	0/1		1
	SNFCCLE	Output	FT1	0/1	1	PBFR5	0/1	0/1		0
PB5	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	ALE	Output	FT1	0/1	1	PBFR1	0/1	0/1		0
	MT3IN	Input	FT1	0/1	0	PBFR3	0/1	0/1		1
	SNFCALE	Output	FT1	0/1	1	PBFR5	0/1	0/1		0
PB6	After ($\overline{BOOT}$)	Input	FT6	0	0	0	0	0		
	After reset			0	0	0	0	0		
	Output Port	Output		0/1	1	0	0/1	0/1		
	$\overline{BELL}$	Output	FT1	0/1	1	PBFR1	0/1	0/1		
	SCOUT	Output	FT1	0/1	1	PBFR2	0/1	0/1		
	TB3OUT	Output	FT1	0/1	1	PBFR4	0/1	0/1		
	$\overline{SNFCCE}$	Output	FT1	0/1	1	PBFR5	0/1	0/1		
PB7	After reset			0	0	0	0	0		0

Table 9-2 PORT B Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PBDATA	PBCR	PBFRn	PBOD	PBPUP	PBPDN	PBIE
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SNFCRB	Input	FT1	0/1	0	PBFR5	0/1	0/1		1

Note: PB6 works as a BOOT function. It is enabled to be input and pulled-up while RESET pin is "Low". At the rising edge of the reset signal, if PB6 is "High", the device enters single chip mode and boots from the on-chip flash memory. If PB6 is "Low", the device enters single BOOT mode and boots from the internal BOOT program.

9.1.2.3 PORT C

Table 9-3 PORT C Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PCDATA	PCCR	PCFRn	PCOD	PCPUP	PCPDN	PCIE
PC0	After reset			0	0		0	0		0
	Input Port	Input		0/1	0		0/1	0/1		1
	Output Port	Output		0/1	1		0/1	0/1		0
	INTE	Input	FT4	0/1	0		0/1	0/1		1
PC1	After reset			0	0		0	0		0
	Input Port	Input		0/1	0		0/1	0/1		1
	Output Port	Output		0/1	1		0/1	0/1		0
	INTF	Input	FT4	0/1	0		0/1	0/1		1
PC2	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	TB3IN0	Input	FT1	0/1	0	PCFR1	0/1	0/1		1
PC3	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	TB4IN0	Input	FT1	0/1	0	PCFR1	0/1	0/1		1
PC4	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	INT1	Input	FT4	0/1	0		0/1	0/1		1
	TB6IN0	Input	FT1	0/1	0	PCFR1	0/1	0/1		1
PC5	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	TB7IN0	Input	FT1	0/1	0	PCFR1	0/1	0/1		1
	RTCOUT	Output	FT1	0/1	1	PCFR2	0/1	0/1		0

9.1.2.4 PORT D

Table 9-4 PORT D Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PDDATA	PDCR	PDFRn	PDOD	PDPUP	PDPDN	PDIE
PD0	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SP2FSS	Input	FT2	0/1	0	PDFR1	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0
PD1	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SP2DI	Input	FT2	0/1	0	PDFR1	0/1	0/1		1
PD2	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SP2DO	Output	FT2	0/1	1	PDFR1	0/1	0/1		0
PD3	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SP2CLK	Input	FT2	0/1	0	PDFR1	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0
PD4	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	INT7	Input	FT4	0/1	0		0/1	0/1		1
	TB5IN0	Input	FT1	0/1	0	PDFR1	0/1	0/1		1

9.1.2.5 PORT E

Table 9-5 PORT E Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PEDATA	PECR	PEFRn	PEOD	PEPUP	PEPDN	PEIE
PE0	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	INT4	Input	FT4	0/1	0		0/1	0/1		1
	A16	Output	FT1	0/1	1	PEFR3	0/1	0/1		0
	TB0IN0	Input	FT1	0/1	0	PEFR5	0/1	0/1		1
PE1	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	INT5	Input	FT4	0/1	0		0/1	0/1		1
	SC0RXD	Input	FT1	0/1	0	PEFR1	0/1	0/1		1
	A17	Output	FT1	0/1	1	PEFR3	0/1	0/1		0
	TB1IN0	Input	FT1	0/1	0	PEFR5	0/1	0/1		1
PE2	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SC0TXD	Output	FT1	0/1	1	PEFR1	0/1	0/1		0
	A18	Output	FT1	0/1	1	PEFR3	0/1	0/1		0
	TB1OUT	Output	FT1	0/1	1	PEFR5	0/1	0/1		0
PE3	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SC0SCK	Input	FT1	0/1	0	PEFR1	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0
	A19	Output	FT1	0/1	1	PEFR3	0/1	0/1		0
	$\overline{\text{SC0CTS}}$	Input	FT1	0/1	0	PEFR4	0/1	0/1		1
	TB0OUT	Output	FT1	0/1	1	PEFR5	0/1	0/1		0
PE4	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SC1SCK	Input	FT1	0/1	0	PEFR1	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0
	A20	Output	FT1	0/1	1	PEFR3	0/1	0/1		0
	$\overline{\text{SC1CTS}}$	Input	FT1	0/1	0	PEFR4	0/1	0/1		1
	TB2OUT	Output	FT1	0/1	1	PEFR5	0/1	0/1		0
PE5	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SC1TXD	Output	FT1	0/1	1	PEFR1	0/1	0/1		0
	A21	Output	FT1	0/1	1	PEFR3	0/1	0/1		0
PE6	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SC1RXD	Input	FT1	0/1	0	PEFR1	0/1	0/1		1
	A22	Output	FT1	0/1	1	PEFR3	0/1	0/1		0

Table 9-5 PORT E Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PEDATA	PECR	PEFRn	PEOD	PEPUP	PEPDN	PEIE
PE7	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	INT6	Input	FT4	0/1	0		0/1	0/1		1
	A23	Output	FT1	0/1	1	PEFR3	0/1	0/1		0
	TB2IN0	Input	FT1	0/1	0	PEFR5	0/1	0/1		1

9.1.2.6 PORT F

Table 9-6 PORT F Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PFDATA	PFCR	PFFRn	PFOD	PFPUP	PFPDN	PFIE
PF0	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD0	I/O	FT7	0/1	1	PFFR1	0/1	0/1		1
	$\overline{\text{UT0CTS}}$	Input	FT1	0/1	0	PFFR3	0/1	0/1		1
PF1	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD1	I/O	FT7	0/1	1	PFFR1	0/1	0/1		1
	UT0TXD	Output	FT1	0/1	1	PFFR3	0/1	0/1		0
	UT0IROUT	Output	FT1	0/1	1	PFFR4	0/1	0/1		0
PF2	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD2	I/O	FT7	0/1	1	PFFR1	0/1	0/1		1
	UT0RXD	Input	FT1	0/1	0	PFFR3	0/1	0/1		1
	UT0IRIN	Input	FT1	0/1	0	PFFR4	0/1	0/1		1
PF3	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD3	I/O	FT7	0/1	1	PFFR1	0/1	0/1		1
	$\overline{\text{UT0RTS}}$	Output	FT1	0/1	1	PFFR3	0/1	0/1		0
	SP1CLK	Input	FT2	0/1	0	PFFR5	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0
PF4	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	INT0	Input	FT4	0/1	0		0/1	0/1		1
	AD4	I/O	FT7	0/1	1	PFFR1	0/1	0/1		1
	UT0DCD	Input	FT1	0/1	0	PFFR3	0/1	0/1		1
	SP1DO	Output	FT2	0/1	1	PFFR5	0/1	0/1		0
PF5	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD5	I/O	FT7	0/1	1	PFFR1	0/1	0/1		1
	UT0RIN	Input	FT1	0/1	0	PFFR3	0/1	0/1		1
	SP1DI	Input	FT2	0/1	0	PFFR5	0/1	0/1		1
PF6	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD6	I/O	FT7	0/1	1	PFFR1	0/1	0/1		1
	UT0DSR	Input	FT1	0/1	0	PFFR3	0/1	0/1		1
	I2C1SCL	I/O	FT1	0/1	1	PFFR4	1	0/1		1
	SP1FSS	Input	FT2	0/1	0	PFFR5	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0

Table 9-6 PORT F Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PFDATA	PFCR	PFFRn	PFOD	PFPUP	PFPDN	PFIE
PF7	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD7	I/O	FT7	0/1	1	PFFR1	0/1	0/1		1
	UT0DTR	Output	FT1	0/1	1	PFFR3	0/1	0/1		0
	I2C1SDA	I/O	FT1	0/1	1	PFFR4	1	0/1		1

9.1.2.7 PORT G

Table 9-7 PORT G Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PGDATA	PGCR	PGFRn	PGOD	PGPUP	PGPDN	PGIE
PG0	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD8	I/O	FT7	0/1	1	PGFR1	0/1	0/1		1
	MT0IN	Input	FT1	0/1	0	PGFR3	0/1	0/1		1
	SNFCD0	I/O	FT7	0/1	1	PGFR5	0/1	0/1		1
PG1	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD9	I/O	FT7	0/1	1	PGFR1	0/1	0/1		1
	$\overline{GEMG0}$	Input	FT1	0/1	0	PGFR3	0/1	0/1		1
	SNFCD1	I/O	FT7	0/1	1	PGFR5	0/1	0/1		1
PG2	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD10	I/O	FT7	0/1	1	PGFR1	0/1	0/1		1
	MT0OUT1	Output	FT2	0/1	1	PGFR3	0/1	0/1		0
	MT0TBIN	Input	FT1	0/1	0	PGFR4	0/1	0/1		1
	SNFCD2	I/O	FT7	0/1	1	PGFR5	0/1	0/1		1
PG3	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD11	I/O	FT7	0/1	1	PGFR1	0/1	0/1		1
	MT0OUT0	Output	FT2	0/1	1	PGFR3	0/1	0/1		0
	MT0TBOUT	Output	FT1	0/1	1	PGFR4	0/1	0/1		0
	SNFCD3	I/O	FT7	0/1	1	PGFR5	0/1	0/1		1
PG4	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD12	I/O	FT7	0/1	1	PGFR1	0/1	0/1		1
	SNFCD4	I/O	FT7	0/1	1	PGFR5	0/1	0/1		1
PG5	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD13	I/O	FT7	0/1	1	PGFR1	0/1	0/1		1
	SNFCD5	I/O	FT7	0/1	1	PGFR5	0/1	0/1		1
PG6	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD14	I/O	FT7	0/1	1	PGFR1	0/1	0/1		1
	SNFCD6	I/O	FT7	0/1	1	PGFR5	0/1	0/1		1
PG7	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	AD15	I/O	FT7	0/1	1	PGFR1	0/1	0/1		1

Table 9-7 PORT G Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PGDATA	PGCR	PGFRn	PGOD	PGPUP	PGPDN	PGIE
	SNFCD7	I/O	FT7	0/1	1	PGFR5	0/1	0/1		1

9.1.2.8 PORT H

Table 9-8 PORT H Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PHDATA	PHCR	PHFRn	PHOD	PHPUP	PHPDN	PHIE
PH0	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	$\overline{BELH}$	Output	FT1	0/1	1	PHFR1	0/1	0/1		0
	TB5OUT	Output	FT1	0/1	1	PHFR2	0/1	0/1		0
	MT2IN	Input	FT1	0/1	0	PHFR3	0/1	0/1		1
	I2C2SDA	I/O	FT1	0/1	1	PHFR4	1	0/1		1
PH1	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	$\overline{CS1}$	Output	FT1	0/1	1	PHFR1	0/1	0/1		0
	TB4OUT	Output	FT1	0/1	1	PHFR2	0/1	0/1		0
	$\overline{GEMG2}$	Input	FT1	0/1	0	PHFR3	0/1	0/1		1
	I2C2SCL	I/O	FT1	0/1	1	PHFR4	1	0/1		1
PH2	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	$\overline{CS2}$	Output	FT1	0/1	1	PHFR1	0/1	0/1		0
	MT2OUT1	Output	FT2	0/1	1	PHFR3	0/1	0/1		0
	MT2TBIN	Input	FT1	0/1	0	PHFR4	0/1	0/1		1
PH3	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	$\overline{CS3}$	Output	FT1	0/1	1	PHFR1	0/1	0/1		0
	MT2OUT0	Output	FT2	0/1	1	PHFR3	0/1	0/1		0
	MT2TBOUT	Output	FT1	0/1	1	PHFR4	0/1	0/1		0

9.1.2.9 PORT J

Table 9-9 PORT J Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PJDATA	PJCR	PJFRn	PJOD	PJPUP	PJPDN	PJIE
PJ0	After reset (AIN0)	Input	FT5	0	0		0	0		0
	Input Port	Input		0	0		0/1	0/1		1
	Output Port	Output		0	1		0/1	0/1		0
PJ1	After reset (AIN1)	Input	FT5	0	0		0	0		0
	Input Port	Input		0	0		0/1	0/1		1
	Output Port	Output		0	1		0/1	0/1		0
PJ2	After reset (AIN2)	Input	FT5	0	0		0	0		0
	Input Port	Input		0	0		0/1	0/1		1
	Output Port	Output		0	1		0/1	0/1		0
PJ3	After reset (AIN3)	Input	FT5	0	0		0	0		0
	Input Port	Input		0	0		0/1	0/1		1
	Output Port	Output		0	1		0/1	0/1		0
	INT9	Input	FT4	0/1	0		0/1	0/1		1
PJ4	After reset (AIN4)	Input	FT5	0	0		0	0		0
	Input Port	Input		0	0		0/1	0/1		1
	Output Port	Output		0	1		0/1	0/1		0
	INTA	Input	FT4	0/1	0		0/1	0/1		1
PJ5	After reset (AIN5)	Input	FT5	0	0		0	0		0
	Input Port	Input		0	0		0/1	0/1		1
	Output Port	Output		0	1		0/1	0/1		0
	INTB	Input	FT4	0/1	0		0/1	0/1		1
PJ6	After reset (AIN6)	Input	FT5	0	0		0	0		0
	Input Port	Input		0	0		0/1	0/1		1
	Output Port	Output		0	1		0/1	0/1		0
	INTC	Input	FT4	0/1	0		0/1	0/1		1
PJ7	After reset (AIN7)	Input	FT5	0	0	0	0	0		0
	Input Port	Input		0	0	0	0/1	0/1		1
	Output Port	Output		0	1	0	0/1	0/1		0
	$\overline{\text{DMAREQ0}}$	Input	FT1	0/1	0	PJFR1	0/1	0/1		1
	$\overline{\text{DMAREQ1}}$	Input	FT1	0/1	0	PJFR2	0/1	0/1		1
	$\overline{\text{DMAREQ2}}$	Input	FT1	0/1	0	PJFR3	0/1	0/1		1

Note: To use the Port J as an analog input of the AD converter, disable input on PJIE and disable pull-up on PJPUP.

9.1.2.10 PORT K

Table 9-10 PORT K Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PKDATA	PKCR	PKFRn	PKOD	PKPUP	PKPDN	PKIE
PK0	After reset			0	0		0	0		0
	Input Port	Input		0/1	0		0/1	0/1		1
	Output Port	Output		0/1	1		0/1	0/1		0
	INTD	Input	FT4	0/1	0		0/1	0/1		1
PK1	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	INT8	Input	FT4	0/1	0		0/1	0/1		1
	SP0FSS0	Input	FT2	0/1	0	PKFR2	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0
	TB6OUT	Output	FT1	0/1	1	PKFR4	0/1	0/1		0
PK2	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SP0DI	Input	FT2	0/1	0	PKFR2	0/1	0/1		1
	I2C0SDA	I/O	FT1	0/1	1	PKFR3	1	0/1		1
PK3	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SP0DO	Output	FT2	0/1	1	PKFR2	0/1	0/1		0
	I2C0SCL	I/O	FT1	0/1	1	PKFR3	1	0/1		1
PK4	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	SP0CLK	Input	FT2	0/1	0	PKFR2	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0

9.1.2.11 PORT L

Table 9-11 PORT L Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PLDATA	PLCR	PLFRn	PLOD	PLPUP	PLPDN	PLIE
PL0	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	INT2	Input	FT4	0/1	0		0/1	0/1		1
	MT1IN	Input	FT1	0/1	0	PLFR3	0/1	0/1		1
	$\overline{\text{ADTRG}}$	Input	FT1	0/1	0	PLFR4	0/1	0/1		1
PL1	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	$\overline{\text{GEMG1}}$	Input	FT1	0/1	0	PLFR3	0/1	0/1		1
	SC2RXD	Iutput	FT1	0/1	0	PLFR5	0/1	0/1		1
PL2	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	MT1OUT1	Output	FT2	0/1	1	PLFR3	0/1	0/1		0
	MT1TBIN	Input	FT1	0/1	0	PLFR4	0/1	0/1		1
	SC2TXD	Output	FT1	0/1	1	PLFR5	0/1	0/1		0
PL3	After reset			0	0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1		1
	Output Port	Output		0/1	1	0	0/1	0/1		0
	MT1OUT0	Output	FT2	0/1	1	PLFR3	0/1	0/1		0
	MT1TBOUT	Output	FT1	0/1	1	PLFR4	0/1	0/1		0
	SC2SCK	Input	FT1	0/1	0	PLFR5	0/1	0/1		1
		Output		0/1	1		0/1	0/1		0
	$\overline{\text{SC2CTS}}$	Input	FT1	0/1	0	PLFR6	0/1	0/1		1

9.2 Block Diagrams of Ports

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".

The operation of "Direct reset" shown in the circuit diagram is enabled when the cold-reset occurrs or when the STOP2 mode is released by the reset pin.

9.2.1 Type FT1

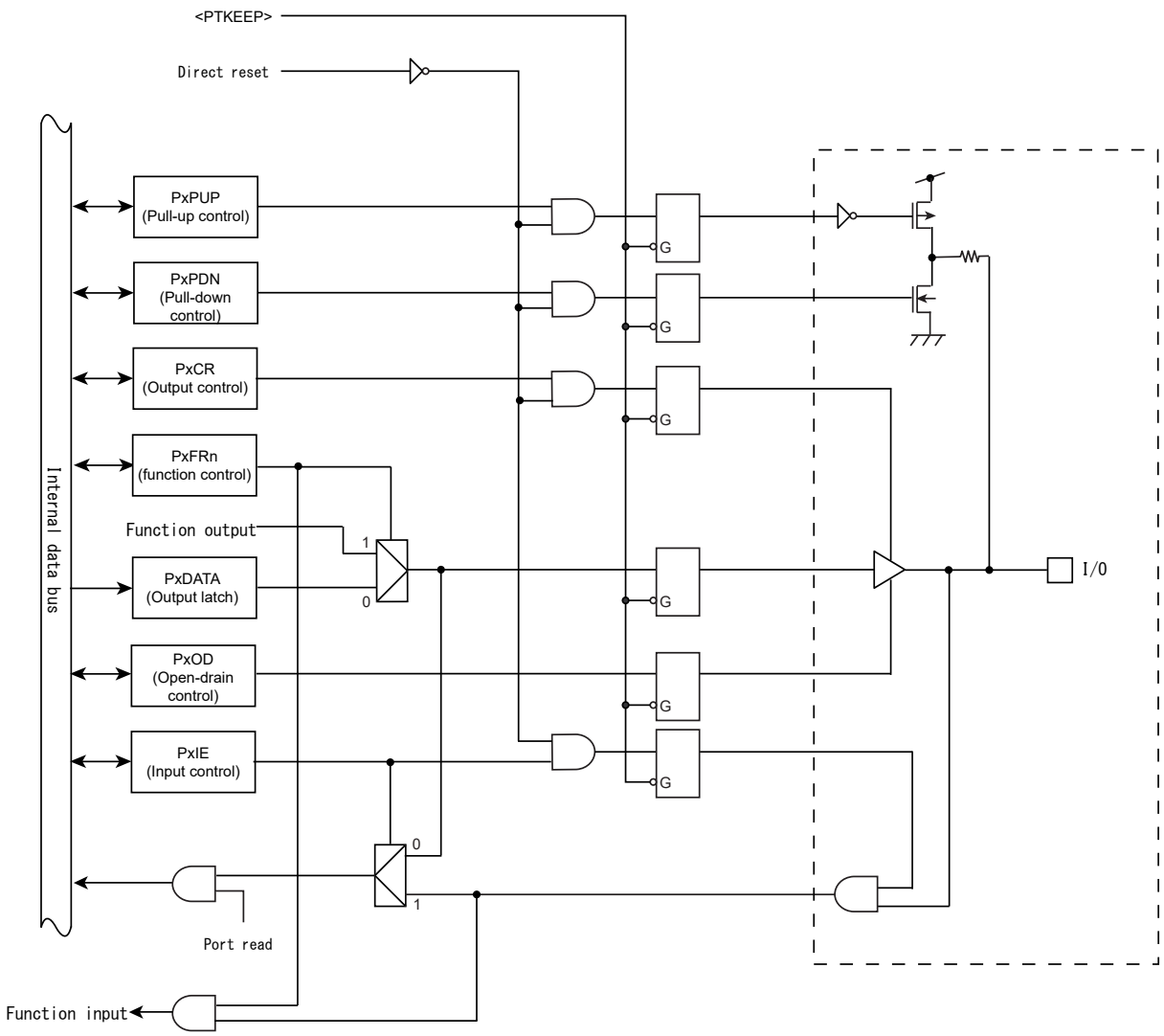


Figure 9-1 Port Type FT1

9.2.2 Type FT2

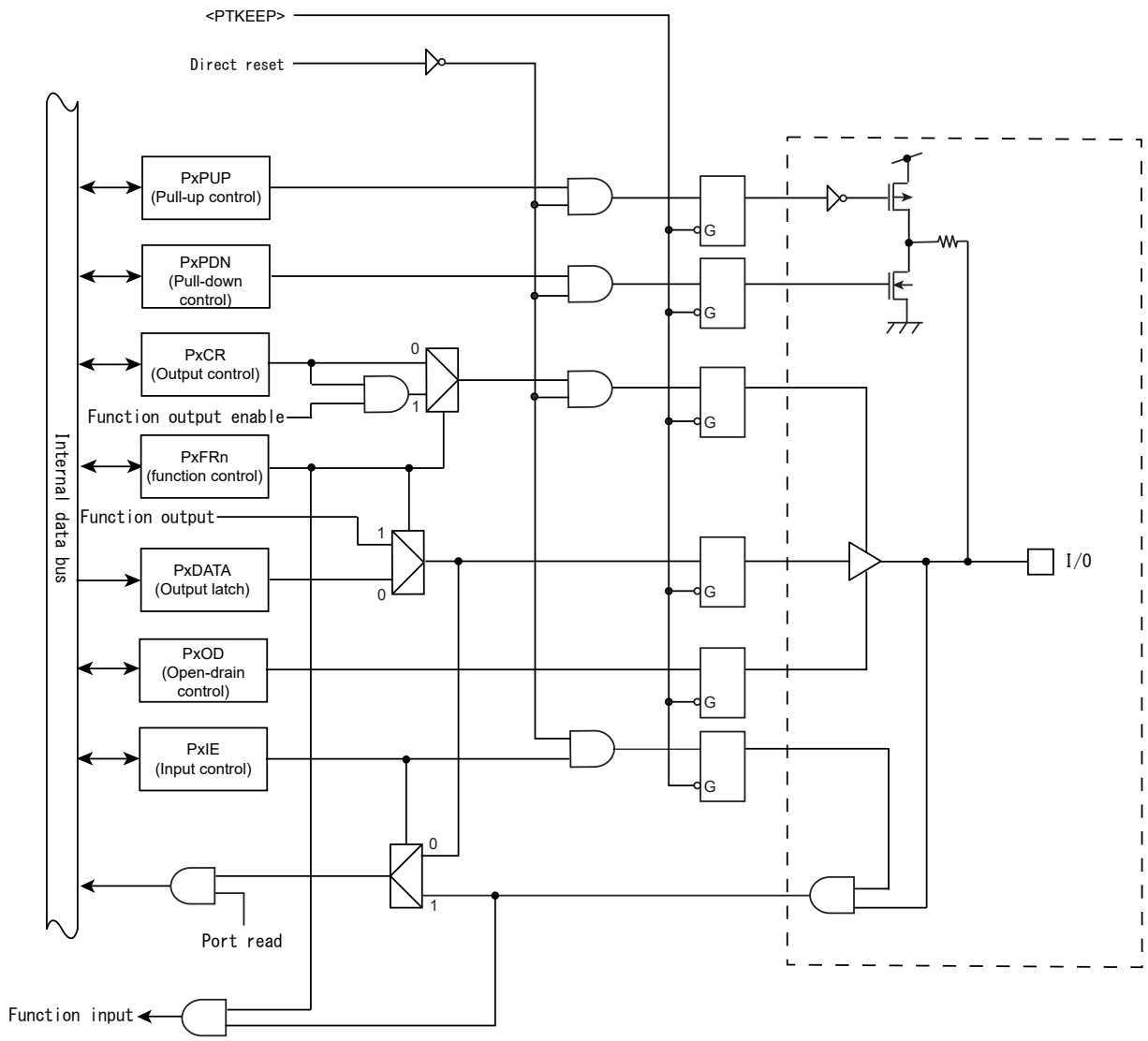


Figure 9-2 Port Type FT2

9.2.3 Type FT3

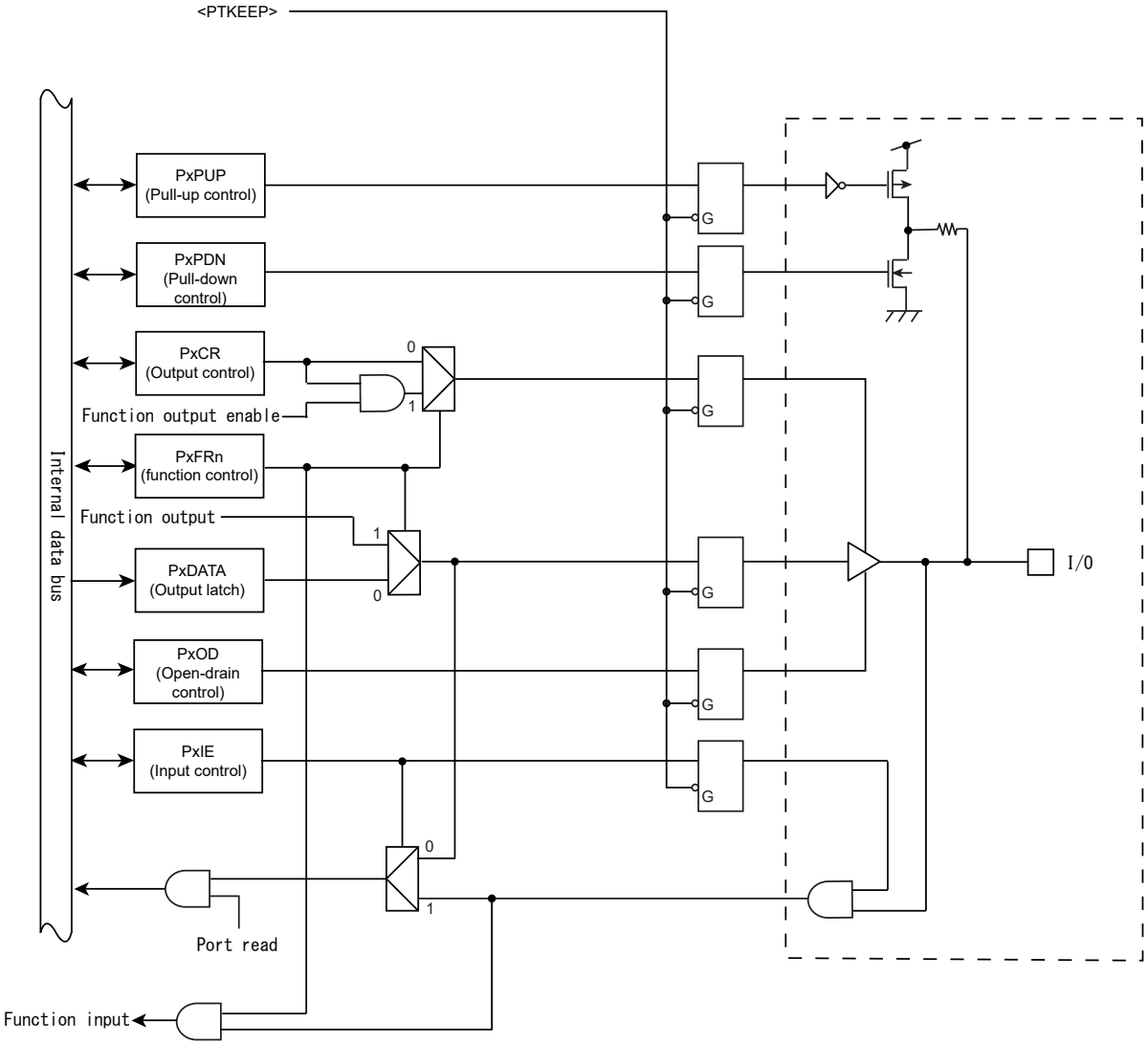


Figure 9-3 Port Type FT3

9.2.4 Type FT4

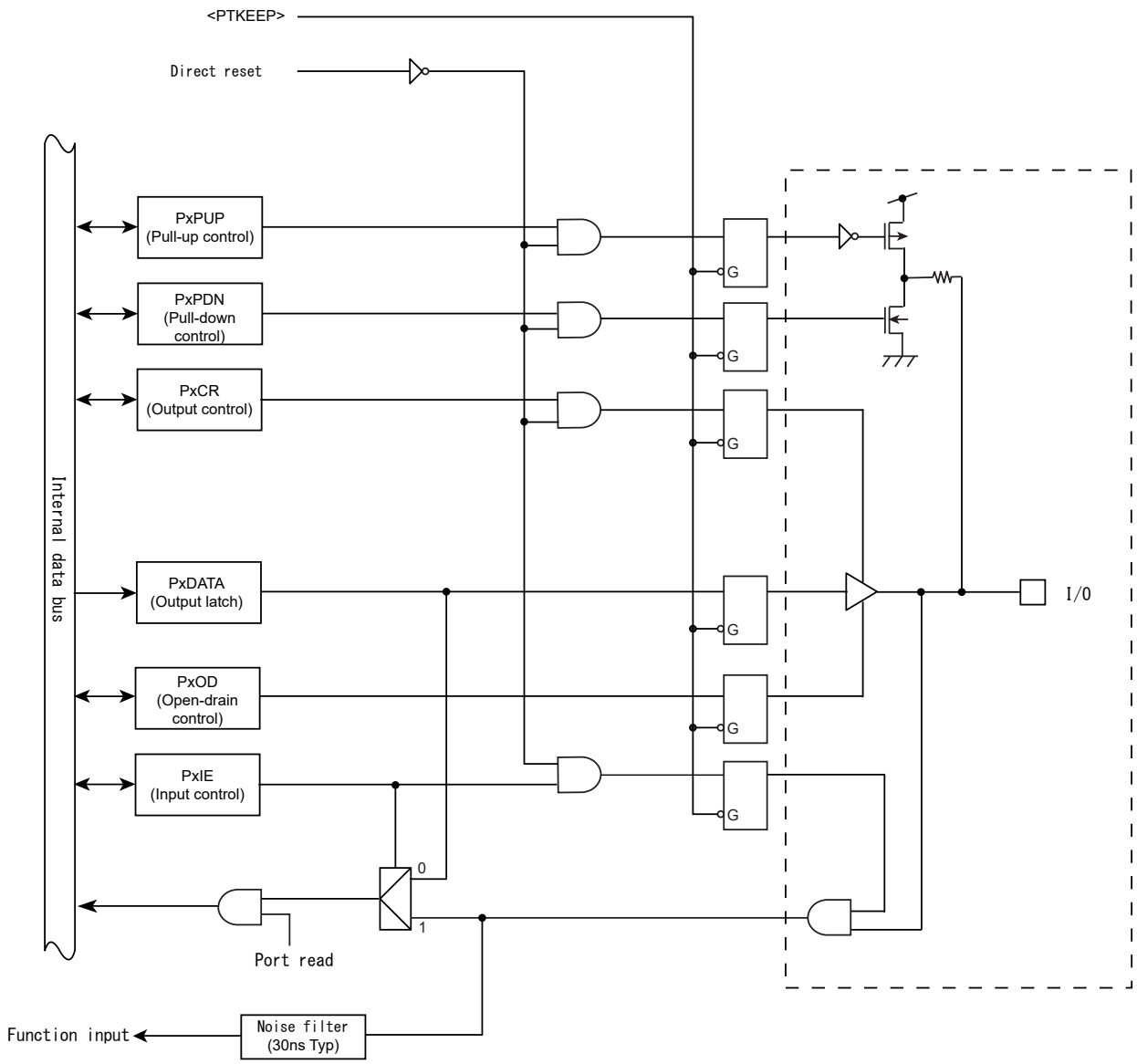


Figure 9-4 Port Type FT4

9.2.5 Type FT5

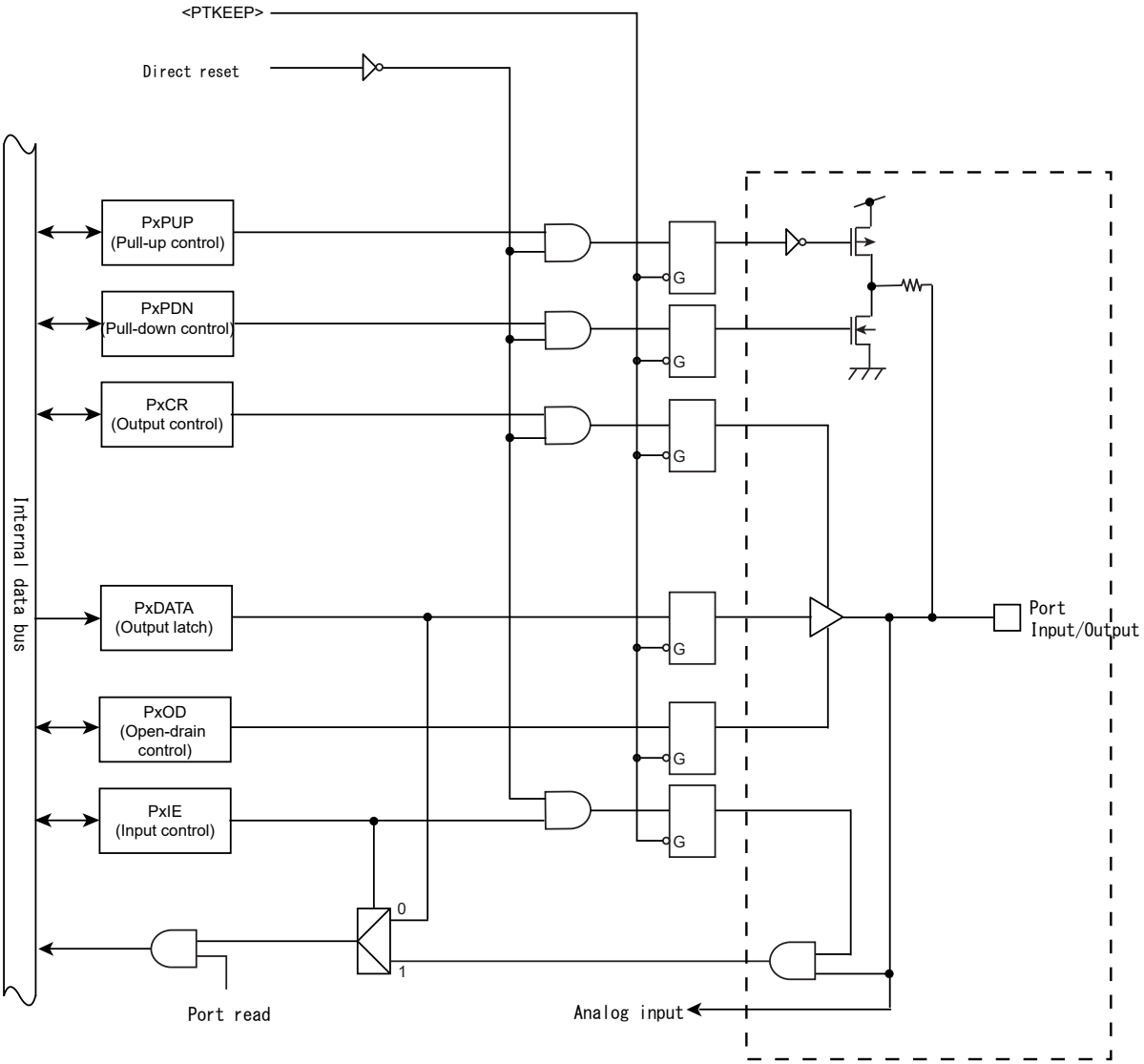


Figure 9-5 Port Type FT5

9.2.6 Type FT6

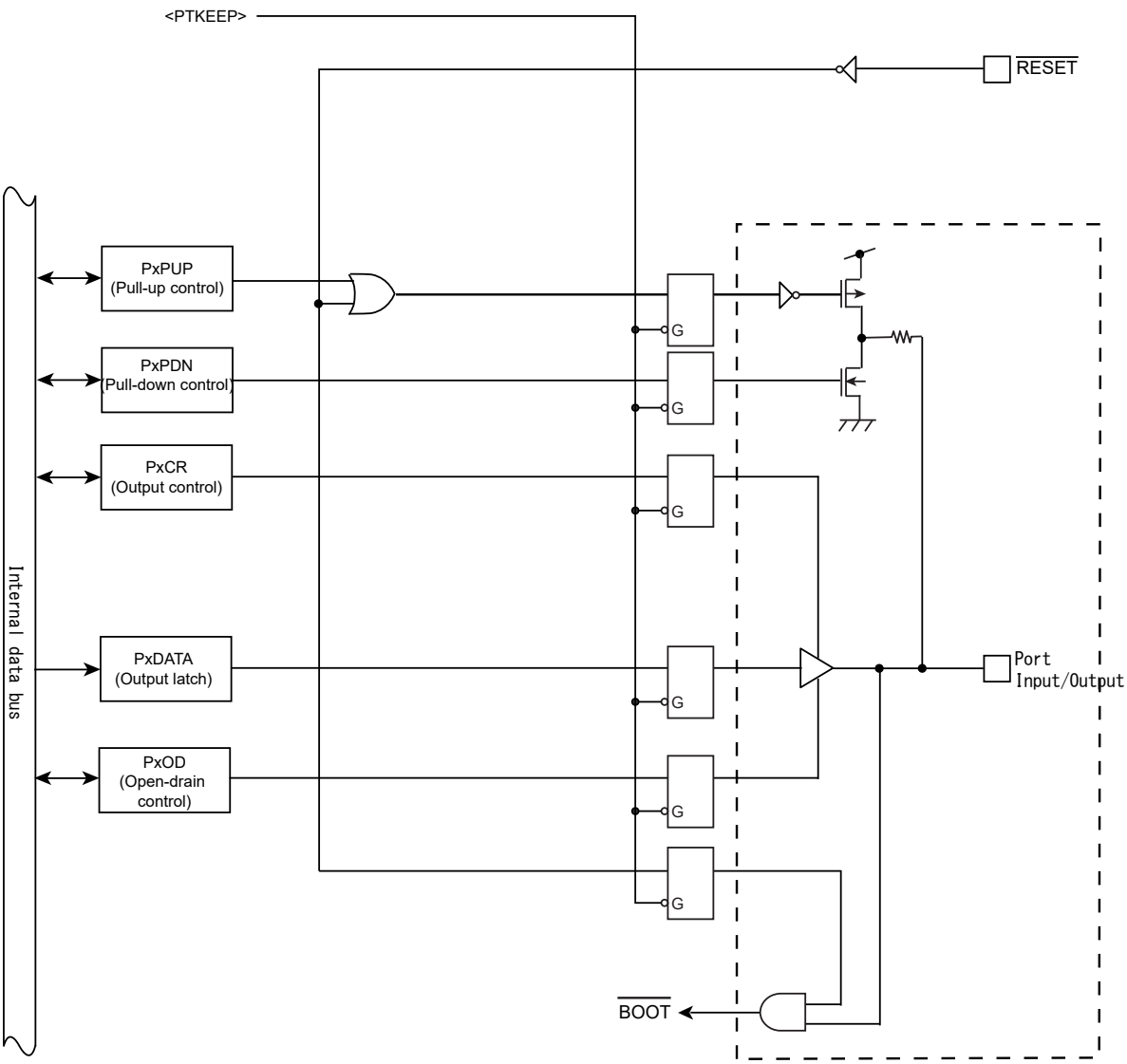


Figure 9-6 Port Type FT6

9.2.7 Type FT7

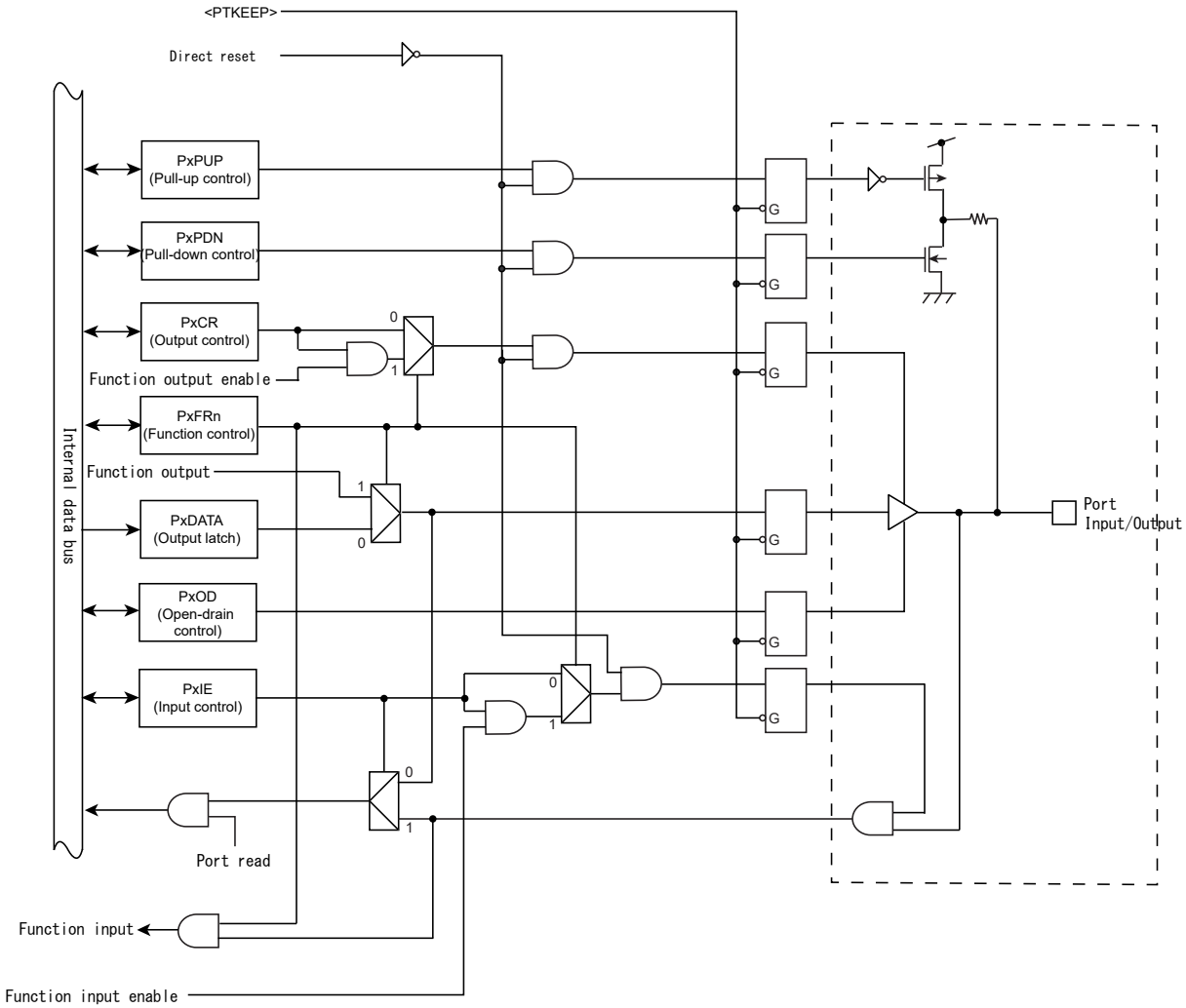


Figure 9-7 Port Type FT7

9.2.8 Type FT8

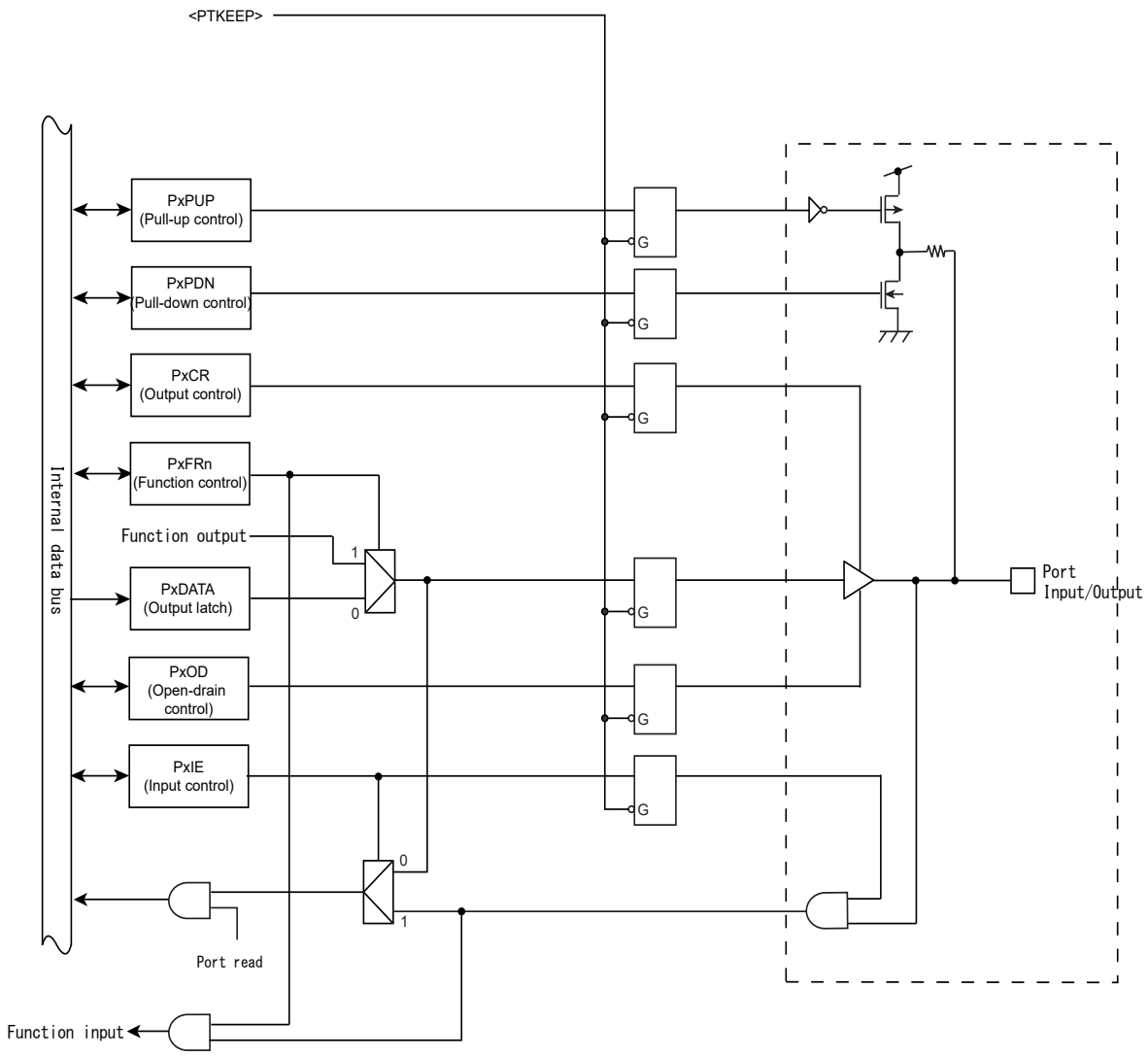


Figure 9-8 Port Type FT8

9.2.9 Type FT9

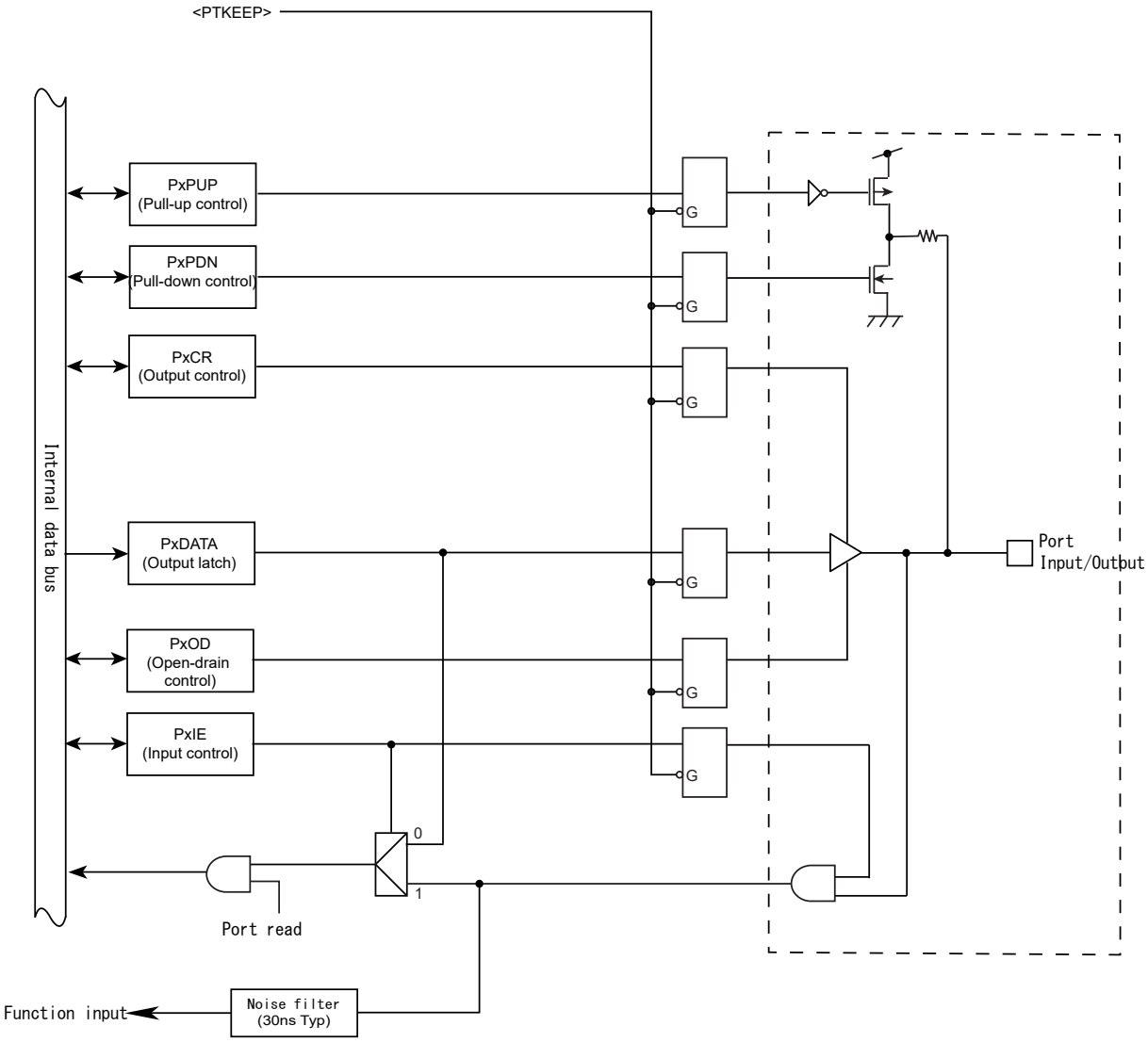


Figure 9-9 Port Type FT9

10. External Bus Interface (EBIF)

10.1 Overview

The TMPM46BF10FG has a built-in external bus interface function to connect to external memory and I/Os. These features are shown in the following table.

Table 10-1 Features of External bus interface

features	
Memory supports	NOR Flash memory, SRAM, Peripheral I/O Multiplex bus mode
Data bus width	Either an 8- or 16-bit width can be set for each channel.
Chip select	4 channels ($\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$, $\overline{CS3}$)
Address access spaces	Supports up to 64MB memory spaces 0x6000_0000 to 0x63FF_FFFF (Up to 16MB for each CS)
Internal wait function	This function can be enabled for each channel. A wait can be automatically inserted up to 15 cycles.
ALE assert time	An assert time can be selected from 1, 2, 3, or 5 cycles for each channel.
Setup cycle insertion function	This function can be enabled for each channel. A $\overline{RD}$ or $\overline{WR}$ setup cycle can be automatically inserted. (tAC cycle expanded)
Recovery (Hold) cycle insertion function	In consecutive external bus cycles, a dummy cycle up to 8 clocks can be inserted and this dummy cycle can be specified for each channel. (tCAR, tRAE cycles expanded)
Bus expansion function	Internal wait, ALE assert time, Setup cycle and Recovery cycle can be expanded double or quadruple. (Used in common in all channels.)
Control pins	Multiplex bus mode: AD[15:0], A[23:16], $\overline{RD}$, $\overline{WR}$, $\overline{BELL}$, $\overline{BELH}$, $\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$, $\overline{CS3}$, ALE

10.2 Address and Data Pin Setting

The external bus interface has separate bus and multiplexed modes. The TMPM46BF10FG can use multiplexed bus mode.

EXBMOD is used for this setting. when EXBMOD<EXBSEL> is set to "0", the multiplexed bus mode is selected.

Table 10-2 shows addresses and data pins used in each mode. For the information of ports used for connecting to external devices, refer to Chapter "Product Information".

Table 10-2 Bus Mode, Address and Data Pins

Separate Bus EXBMOD<EXBSEL> = "1"	Multiplex Bus EXBMOD<EXBSEL> = "0"
-	A16 to A23
-	AD0 to AD15

When access is changed from the external area to internal area, the address buses maintain the address output of the previous external area and the data buses become high impedance.

10.3 Registers

10.3.1 Registers List

The following table shows control registers and addresses. For the base address, refer to "Peripheral Function Base Address List" in Chapter "Memory Map".

Register name		Address (Base+)
External Bus Mode Control Register	EXBMOD	0x0000
External Bus Area and Start Address Configuration Register 0	EXBAS0	0x0010
External Bus Area and Start Address Configuration Register 1	EXBAS1	0x0014
External Bus Area and Start Address Configuration Register2	EXBAS2	0x0018
External Bus Area and Start Address Configuration Register 3	EXBAS3	0x001C
External Bus Chip Select Control Register 0	EXBCS0	0x0040
External Bus Chip Select Control Register 1	EXBCS1	0x0044
External Bus Chip Select Control Register 2	EXBCS2	0x0048
External Bus Chip Select Control Register 3	EXBCS3	0x004C

Note:Reading/writing registers are allowed only in the unit of word (32-bit).

10.3.2 EXBMOD (External Bus Mode 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	-	-	-	-	-	EXBWAIT		EXBSEL
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	–	R	Read as 0.
2-1	EXBWAIT[1:0]	R/W	Bus cycle extension 00: None 01: Double 10: Quadruple 11: Prohibited These bits are used to set the setup, wait and recovery of the bus cycle to be double or quadruple. For example, if a Read setup cycle is set as two cycles by setting <EXBWAIT>="00" (no extension), the two cycles can be quadruplicated by changing the bit setting to <EXBWAIT>="01" (double). It also can be octuplicated by setting the bits to <EXBWAIT>="10" (quadruple). The extended cycle is configured by setting Read/Write setup, chip select/Read/Write recovery, ALE/internal wait cycle and <EXBWAIT> (double or quadruple).
0	EXBSEL	R/W	Select external bus mode (Note) Write as "0".

10.3.3 EXBAS0 to 3 (External Bus Area and Start Address Configuration Register)

	31	30	29	28	27	26	25	24
bit symbol	SA31	SA30	SA29	SA28	SA27	SA26	SA25	SA24
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16
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	EXAR							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	SA31-SA16	R/W	Chip select Start address (Note) The A[23:16] is specified as start address.
15-8	–	R	Read as 0.
7-0	EXAR[7:0]	R/W	Chip select Address space size The size of address space can be specified nine kind of setting from 64Kbyte up to 16Mbyte. 0000_0000: 16 Mbyte 0000_0011: 2 Mbyte 0000_0110: 256 Kbyte 0000_0001: 8 Mbyte 0000_0100: 1 Mbyte 0000_0111: 128 Kbyte 0000_0010: 4 Mbyte 0000_0101: 512 Kbyte 0000_1000: 64 Kbyte Others: Prohibited

Note: If same address space is specified between $\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$ and $\overline{CS3}$, the chip selector will be given priority shown below

High priority $\overline{CS0}$ > $\overline{CS1}$ > $\overline{CS2}$ > $\overline{CS3}$ Low priority

Table 10-3 Address area size and start address setting

The address area size of a chip select	SA																-	EXAR							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15 to 8	7	6	5	4	3	2	1	0
16Mbyte	0	1	1	0	0	0	x	x	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0
8Mbyte	0	1	1	0	0	0	x	x	x	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	1
4Mbyte	0	1	1	0	0	0	x	x	x	x	0	0	0	0	0	0	-	0	0	0	0	0	0	1	0
2Mbyte	0	1	1	0	0	0	x	x	x	x	x	0	0	0	0	0	-	0	0	0	0	0	0	1	1
1Mbyte	0	1	1	0	0	0	x	x	x	x	x	x	0	0	0	0	-	0	0	0	0	0	1	0	0
512Kbyte	0	1	1	0	0	0	x	x	x	x	x	x	x	0	0	0	-	0	0	0	0	0	1	0	1
256Kbyte	0	1	1	0	0	0	x	x	x	x	x	x	x	x	0	0	-	0	0	0	0	0	1	1	0
128Kbyte	0	1	1	0	0	0	x	x	x	x	x	x	x	x	x	0	-	0	0	0	0	0	1	1	1
64Kbyte	0	1	1	0	0	0	x	x	x	x	x	x	x	x	x	x	-	0	0	0	0	1	0	0	0

x: Optional

10.3.4 EXBCS0 to 3 (External Bus Chip Select Control Register)

	31	30	29	28	27	26	25	24
bit symbol	CSR		WRR			RDR		
After reset	0	1	0	0	1	0	0	1
	23	22	21	20	19	18	17	16
bit symbol	-	-	ALEW		WRS		RDS	
After reset	0	0	0	1	0	1	0	1
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	CSIW				
After reset	0	0	0	0	0	0	1	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	CSW		CSW0
After reset	0	0	0	0	0	0	1	0

Bit	Bit Symbol	Type	Function
31-30	CSR[1:0]	R/W	Chip select ($\overline{CSx}$) Recovery cycle 00: None 01: 1 cycle 10: 2 cycles 11: 4 cycles
29-27	WRR[2:0]	R/W	Write ($\overline{WR}$) Recovery cycle 000: None 010: 2 cycles 100: 4 cycles 110: 6 cycles 001: 1 cycle 011: 3 cycles, 101: 5 cycles 111: 8 cycles
26-24	RDR[2:0]	R/W	Read ($\overline{RD}$) Recovery cycle 000: None 010: 2 cycles 100: 4 cycles 110: 6 cycles 001: 1 cycle 011: 3 cycles, 101: 5 cycles 111: 8 cycles
23-22	–	R	Read as 0.
21-20	ALEW[1:0]	R/W	ALE wait cycle for multiplex bus Read ($\overline{RD}$) Recovery cycle 00: None 01: 1 cycles 10: 2 cycles 11: 4 cycles
19-18	WRS[1:0]	R/W	Write ($\overline{WR}$) Setup cycle 00: None 01: 1 cycles 10: 2 cycles 11: 4 cycles
17-16	RDS[1:0	R/W	Read ($\overline{RD}$) Setup cycle 00: None 01: 1 cycles 10: 2 cycles 11: 4 cycles
15-13	–	R	Read as 0.
12-8	CSIW[4:0]	R/W	Selection of number of waits 0_0000 : 0 waits 0_0001 : 1 wait 0_0010 : 2 waits 0_0011 : 3 waits 0_0100 : 4 waits 0_0101 : 5 waits 0_0110 : 6 waits 0_0111 : 7 waits 0_1000 : 8 waits 0_1001 : 9 waits 0_1010 : 10 waits 0_1011 : 11waits 0_1100 : 12 waits 0_1101 : 13 waits 0_1110 : 14 waits 0_1111 : 15 waits
7-3	–	R	Read as 0.
2-1	CSW[2:1]	R/W	Data bus width 00: 8-bit 01: 16-bit Other than the above settings: Prohibited
0	CSW0	R/W	CS Enable 0: Disable 1: Enable

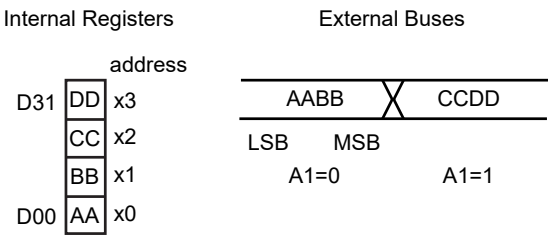
10.4 Data Format

Internal registers and external bus interfaces of the TMPM46BF10FG are configured as described below.

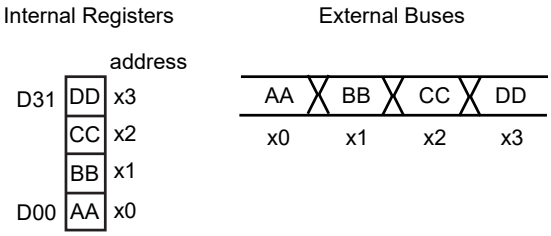
10.4.1 Little-Endian Mode

10.4.1.1 Word Access

- 16-bit bus width

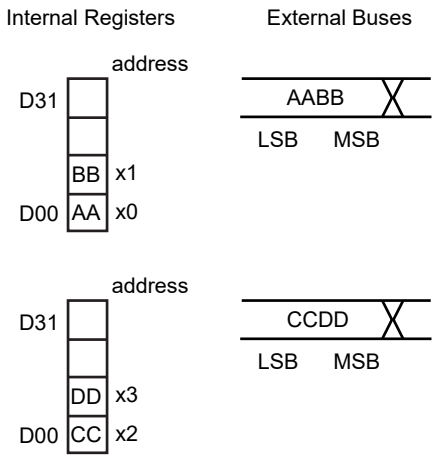


- 8-bit bus width

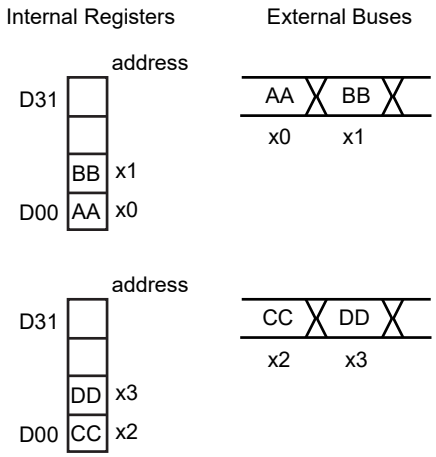


10.4.1.2 Half word access

- 16-bit bus width

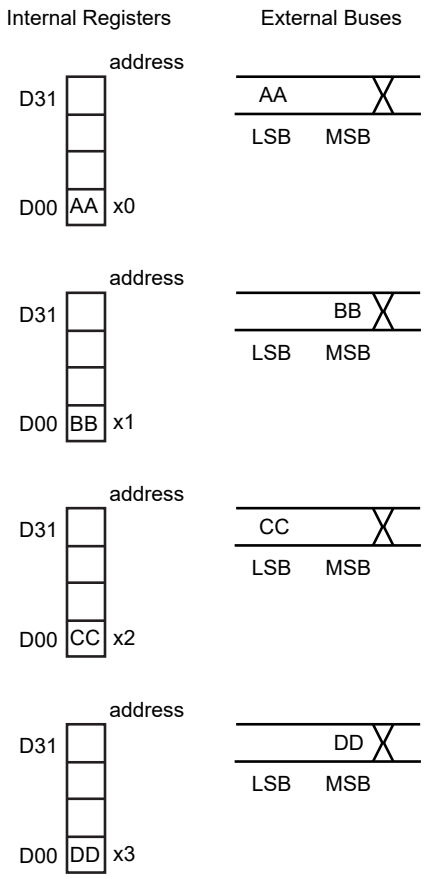


- 8-bit bus width

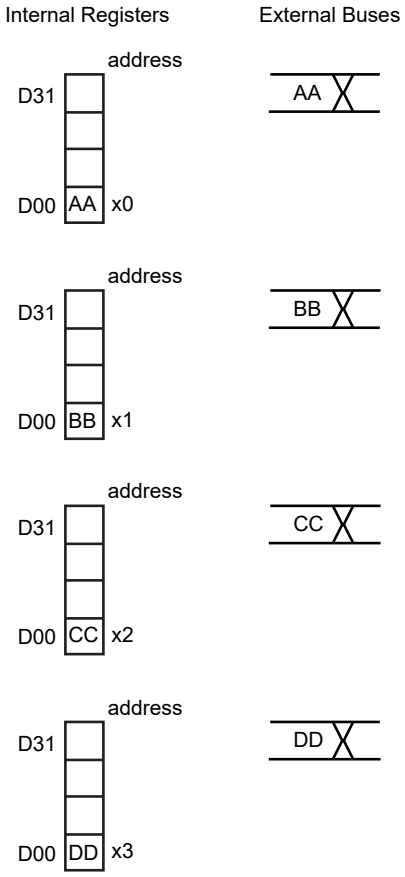


10.4.1.3 Byte access

- 16-bit bus width



- 8-bit bus width



10.5 External Bus Operations (Multiplexed Bus Mode)

This section describes various bus timings. The timing diagram shown below assumes that the address buses are A23 through A16 and data buses are AD15 through AD0.

Bus cycle (tsys) becomes 1 cycle of fsys when the clock output function is not used.

10.5.1 Basic Bus Operation

TMPM46BF10FG's external bus cycle is basically 4 clocks.

Figure 10-1 shows a read bus timing and Figure 10-2 shows a write bus timing. If internal areas are accessed, address buses remain unchanged and the ALE does not output latch pulses as shown in these figures.

Additionally, address/data buses are in a state of high impedance and control signals such as $\overline{RD}$ and $\overline{WR}$ do not become active.

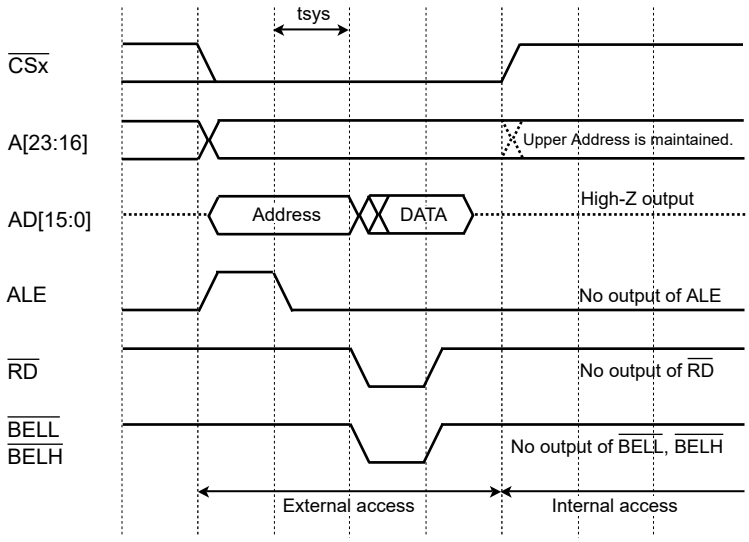


Figure 10-1 Read operation timing

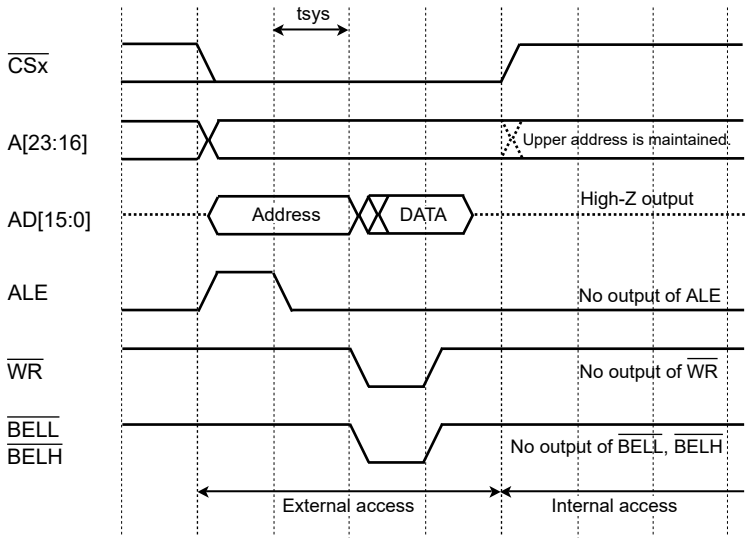


Figure 10-2 Write operation timing

10.5.2 Wait Insertion

A wait cycle can be inserted for each channel by using the internal wait controller.

The following waits can be inserted.

- An internal wait up to 15 clocks can be automatically inserted.

10.5.2.1 Internal Wait

To use the internal wait function, set "0" to EXBCSx<EXWAIT>. The setting of the number of waits can be set using EXBCSx<CSIW[4:0]>.

Figure 10-3 and Figure 10-4 show the read and write timing diagrams in which internal 2 waits have been inserted.

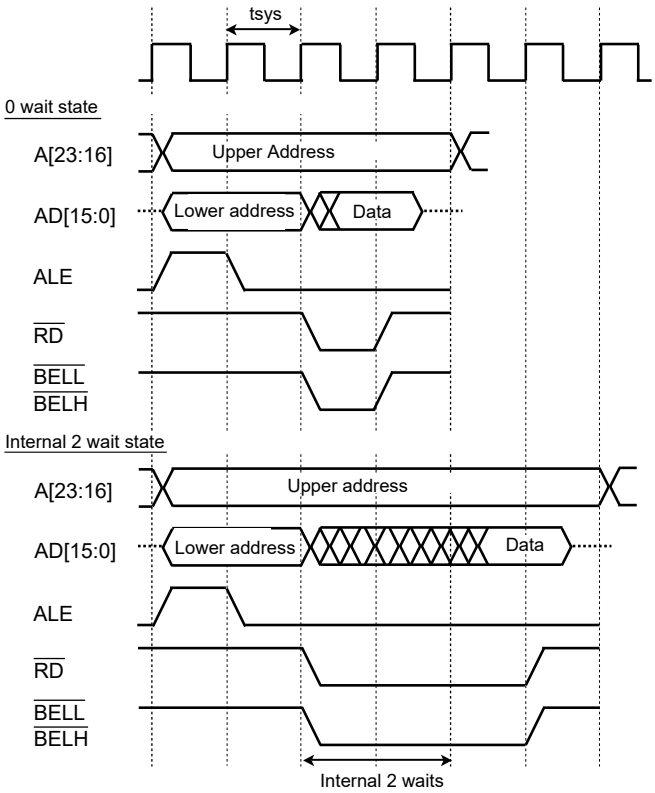


Figure 10-3 Read operation timing

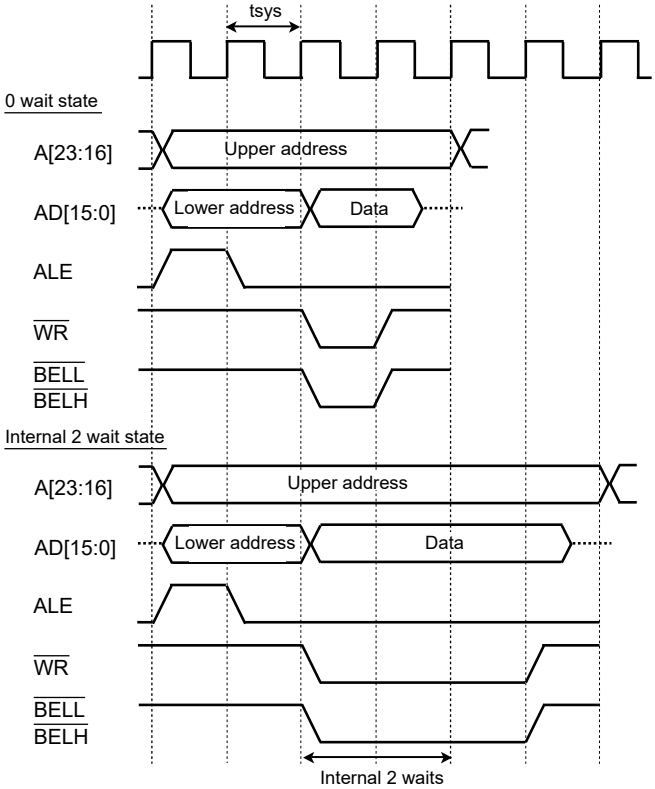


Figure 10-4 Write operation timing

10.5.3 ALE Assert Time

An ALE assert time can be selected from 1, 2, 3 or 5 of system clock. The setting bit is EXBCSx<ALEW>. In the default setting, $\overline{RD}$ or $\overline{WR}$ signal is asserted after 2 system clocks (internal) after the address is generated.

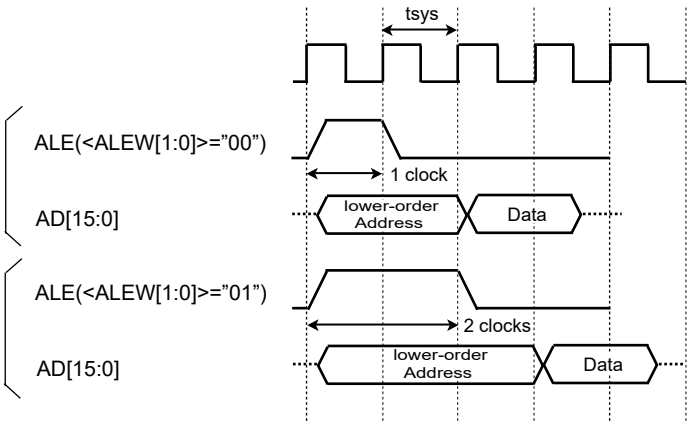


Figure 10-5 ALE asserted time

Figure 10-6 shows the timing when the ALE is 1 clock and 2 clocks.

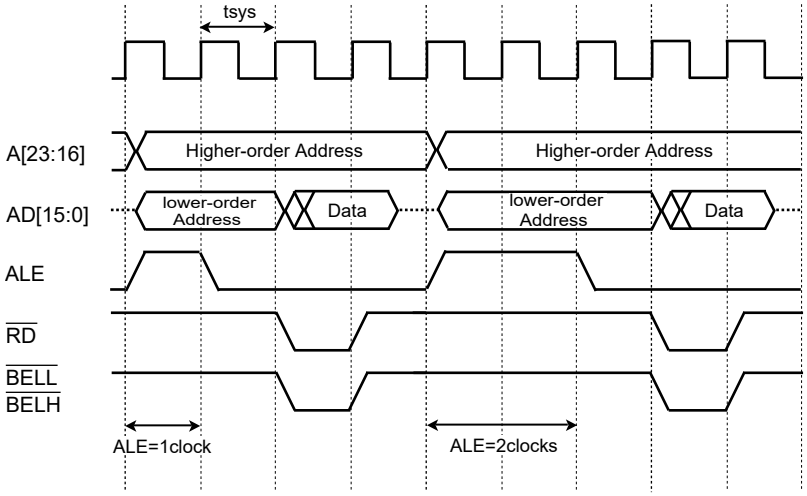


Figure 10-6 Read Operation Timing (When the ALE is 1 Clock or 2 Clocks)

10.5.4 Read and Write Recovery Time

If access to external areas occurs consecutively, a dummy cycle can be inserted as recovery time.

A dummy cycle can be inserted in both a read and a write cycle. The dummy cycle insertion can be set by EXBCSx<WRR[2:0]> (write recovery cycle) and <RDR[2:0]> (read recovery cycle). As for the number of dummy cycles, 0 through six and eight system clocks (internal) can be specified for each channel. Figure 10-7 shows the timing of recovery time insertion.

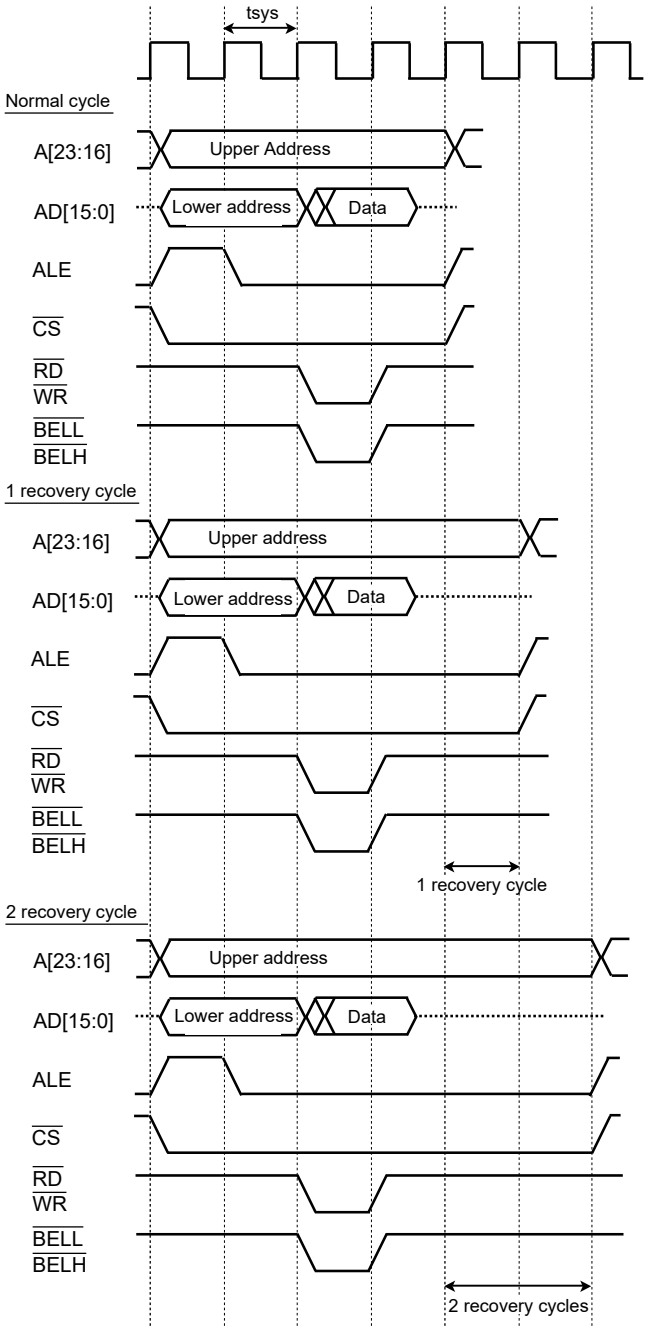


Figure 10-7 Timing of Recovery Time Insertion

10.5.5 Chip Select Recovery Time

If access to external areas occurs consecutively, a dummy cycle can be inserted as recovery time.

The dummy cycle insertion can be set by EXBCSx<CSR[1:0]>. As for the number of dummy cycles, 0 through two and four system clocks (internal) can be specified for each channel. Figure 10-8 shows the timing of recovery time insertion.

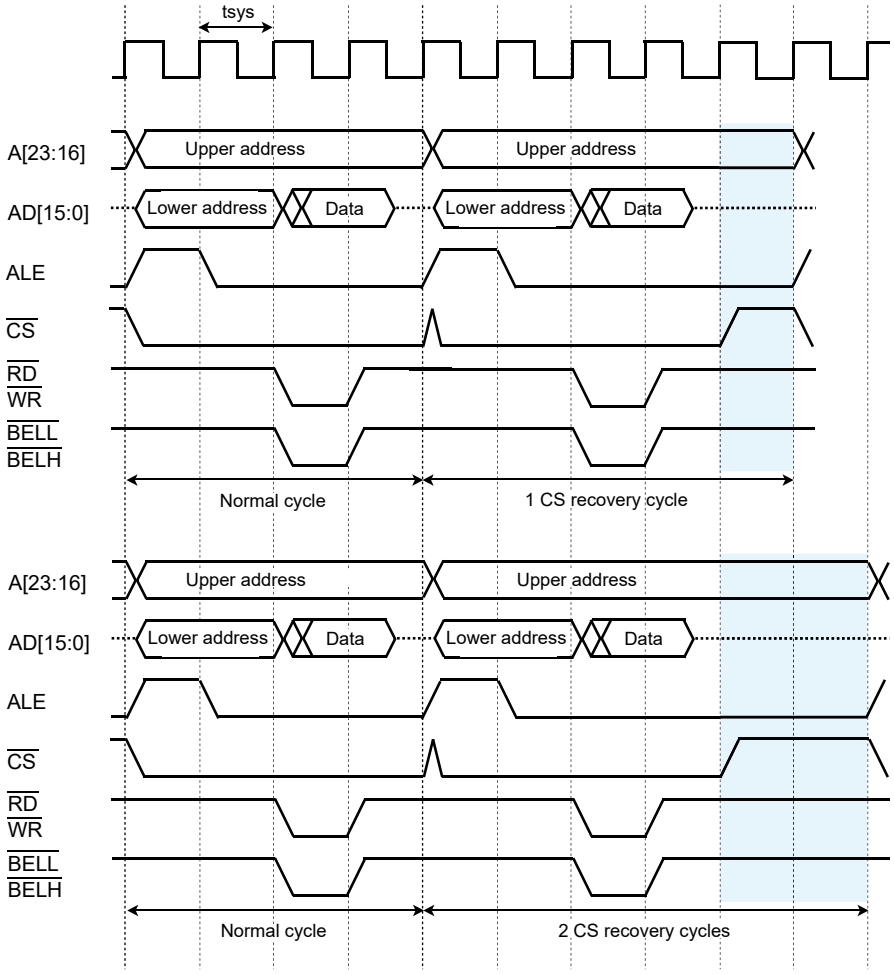


Figure 10-8 Timing of recovery time insertion (ALE width: 1 clock)

10.5.6 Read and Write Setup Cycle

A read and a write setup cycle can be inserted for each channel by using the internal setup controller. The following cycle can be inserted.

- An internal read and write setup cycle up to 4 clocks can be automatically inserted.

The setting of the number of setup cycles can be set using EXBCSx<WRS[1:0]> and <RDS[1:0]>.

Figure 10-9 shows the timing diagrams in which the read or write setup cycle has been inserted.

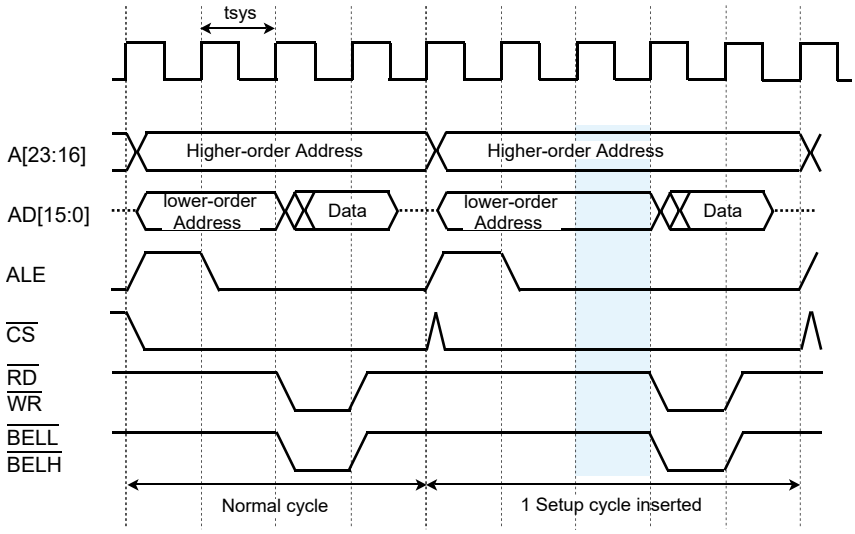


Figure 10-9 Timing of read or write setup time insertion

10.6 Connection Example of External memory

10.6.1 Connection Example of External 16-bit SRAM and NOR-Flash(Non-synchronous Multiplex mode)

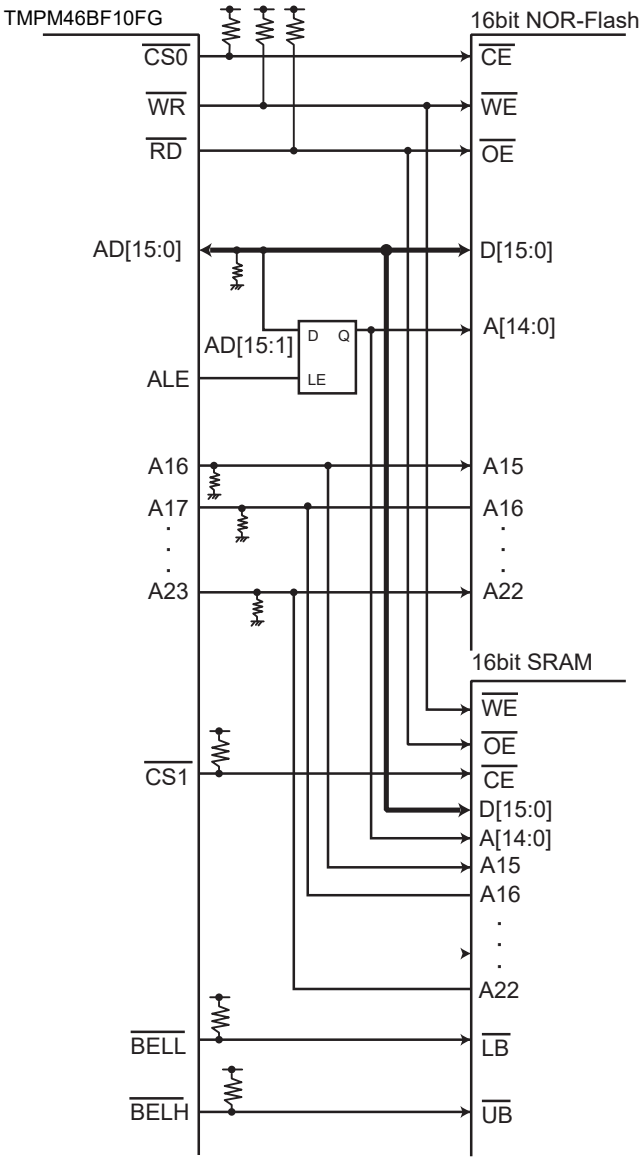


Figure 10-10 Connection Example of external 16-bit SRAM and NOR-Flash (Non-synchronous Multiplex mode)

11. SLC NAND Flash Controller (SNFC)

11.1 Outline

The SLC NAND flash controller supports an SLC (Single Level Cell) type NAND flash memory.

It is equipped with dedicated pins to connect a NAND flash memory.

Its error correction function employs BCH code ECC that can perform 4 bits per 512bytes error correction and 8 bits per 512bytes error correction. It also can output error correction patterns.

The features are as follows:

Table 11-1 Features of the SLC NAND flash controller

Feature	Description
Support memory	NAND flash memory: 1 G bit to 4 G bit, 8-bit data bus width, 3V VDD
Data bus width	8-bit width only (SNFCD[7:0])
Chip select signal	1 channel ($\overline{\text{SNFCCE}}$)
Internal wait function	Up to 8 cycles can be inserted.
Setup/recovery insert function	A setup cycle/recovery (hold) cycle can be inserted.
Control pin	SNFCD[7:0], SNFCCLE, SNFCALE, $\overline{\text{SNFCRE}}$, $\overline{\text{SNFCWE}}$, SNFCRB,$\overline{\text{SNFCCE}}$ (The WP pin is not supported.)
ECC	BCH encoding/decoding, built-in error correction circuit (4 bits per 512 Bytes and 8 bits pre 512 Bytes)
Automatic load function	High-speed data transfer is achieved by combining with the DMA controller (DMAA, DMAB, and DMAC).

1. Supported memory

   The SNFC supports an SLC type NAND flash memory.

   The table below lists the supported memories:

Table 11-2 Specifications of supported memories

Memory specification									BCH mode (ECC)
Memory size (bit)	Block			Page				Bus width (bit)	
	The number of blocks	Size (byte)	The number of pages	The number of pages	Size (Byte)				
						Data	Redundancy		
1G (128M x 8 bit)	1024	128K + 4K	64	65536	2112	2048	64	8	4bit / 512Byte
		128K + 8K			2176		128		8bit / 512Byte
2G (256M x 8 bit)	2048	128K + 4K		131072	2112		64		4bit / 512Byte
		128K + 8K			2176		128		8bit / 512Byte
4G (512M x 8 bit)		256K + 14K			4320	4096	224		4bit / 512Byte
		256K + 16K			4352		256		8bit / 512Byte

2. Waveform adjustment function

   The SNFC can insert an internal wait and setup cycle/recovery (hold) cycle depending on the connected memory type.

   - Setup time/hold time of SNFCCLE
   - Setup time/hold time of SNFCALE
   - Setup time/wait time/hold time of $\overline{\text{SNFCRE}}$
   - Setup time/wait time/hold time of $\overline{\text{SNFCWE}}$

3. Control pin

   The SNFC is equipped with the dedicated pins to connect a NAND flash memory. It enables direct connection.

4. ECC

   The SNFC incorporates an error detection/correction circuit that can perform BCH encoding/decoding on 4 bits per 512 Bytes or 8 bits per 512 Bytes.

   The ECC process is executed in the units of sector. Automatic correction and software correction can be selected.

Figure 11-1 shows the block diagram.

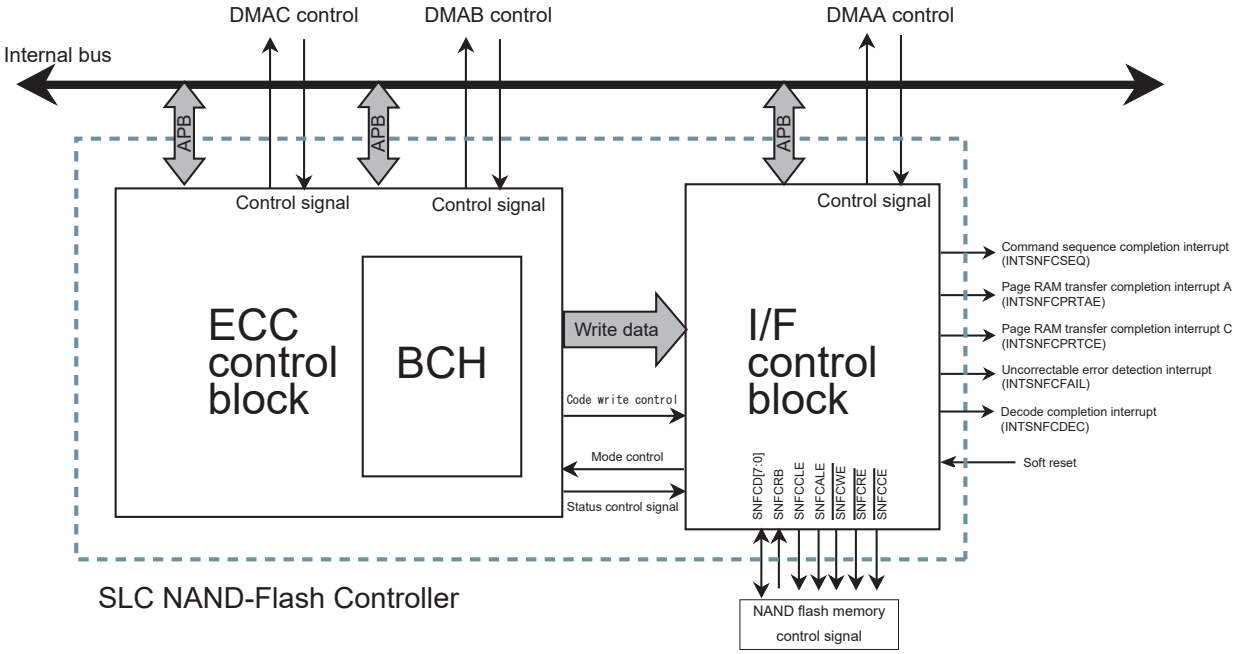


Figure 11-1 Block diagram

11.2 Pin Description

The table below lists the pins of the SNFC and their functions:

Table 11-3 SLC NAND flash controller pins

Pin name	Input/output	Function
$\overline{\text{SNFCCE}}$	Output	A chip enable signal for the flash memory (This pin enables the flash memory that is connected to this LSI.)
SNFCCLE	Output	A command latch enable signal for the flash. (The signal is asserted when this LSI outputs a command.)
SNFCALE	Output	An address latch enable signal for the flash memory. (The signal is asserted when this LSI outputs an address. The signal is negated when this LSI outputs/inputs data.)
$\overline{\text{SNFCRE}}$	Output	A read enable signal for the flash memory. (The SNFC reads data on the rising edge of $\overline{\text{SNFCRE}}$.)
$\overline{\text{SNFCWE}}$	Output	A write enable signal for the flash memory. (A flash memory latches a command, address, and data on the rising edge of $\overline{\text{SNFCWE}}$.)
SNFCD [7 to 0]	Input/output	An I/O signal between the flash memory. (The output signal is for commands/address and the input/output signal for data.)
SNFCRB	Input	A ready/busy signal from the flash memory. (A high-level signal is recognized as ready state; a low-level signal is recognized as busy state.)

Note 1: The WP pin is not available.

Note 2: When the SNFC is used, specify the port setting and then specify the SNFC setting.
Note that when the port setting is specified, specify the function registers first and then specify the control registers. This order reduces unnecessary signals to flow to the control pins and data bus.

Note 3: Connection pins with **the NAND flash memory are shared with the general-purpose ports. The connection pins become high-impedance after reset. These connection pins are high-impedance state until the port setting is specified. Therefore, place the pull-up registers to the LSI to avoid influence to the connected memory.**

11.3 Registers

11.3.1 Register List

The following table lists the control registers and their addresses:

For the base address, refer to "A list of peripheral function base addresses" in the chapter on "Memory Map."

SNFC_IF Block

Register name		Address (Base+)
Enable control register	SNFCENC	0x0000
ECC mode register	SNFCECCMOD	0x0004
Interrupt enable register	SNFCIE	0x0008
Page size register	SNFCPS	0x0010
Page read column status register	SNFCPRCS	0x0014
Sector register	SNFCS	0x0018
Sector status register	SNFCSS	0x001C
Decode input count register	SNFCDIC	0x0020
Decode output count register	SNFCDOC	0x0024
Encode input count register	SNFCEIC	0x0028
Address register 1	SNFCA1	0x0030
Address register 2	SNFCA2	0x0034
Write register	SNFCW	0x0038
Bus interface control register	SNFCBIC	0x003C
Command sequence register 1	SNFCCS1	0x0040
Command sequence register 2	SNFCCS2	0x0044
Command sequence register 3	SNFCCS3	0x0048
Command sequence register 4	SNFCCS4	0x004C
Command sequence enable register	SNFCCSE	0x0050
Page read buffer register	SNFCPRDB	0x0100
ID read register 1	SNFCIR1	0x0104
ID read register 2	SNFCIR2	0x0108
ECC parity register 1	SNFCEP1	0x0110
ECC parity register 2	SNFCEP2	0x0114
ECC parity register 3	SNFCEP3	0x0118
ECC parity register 4	SNFCEP4	0x011C
ECC CRC register	SNFCEC	0x0120

SNFC_ECC Block

Register name		Address (Base+)
ECC write buffer register	SNFCEWRB	0x0000

SNFC_GO block

Register name		Address (Base+)
Correction data read buffer register	SNFCCDRB	0x0000
ECC busy status register	SNFCEBS	0x0008
ECC error status register	SNFCEES	0x0020
ECC decode state register 1	SNFCEDS1	0x0040

SNFC_GO block

Register name		Address (Base+)
ECC decode state register 2	SNFCEDS2	0x0044
ECC decode state register 3	SNFCEDS3	0x0048
ECC decode state register 4	SNFCEDS4	0x004C
ECC decode state register 5	SNFCEDS5	0x0050
ECC decode state register 6	SNFCEDS6	0x0054
ECC decode state register 7	SNFCEDS7	0x0058
ECC decode state register 8	SNFCEDS8	0x005C
Sector 1 ECC error 1 position information register	SNFCS1EE1PI	0x0080
Sector 1 ECC error 2 position information register	SNFCS1EE2PI	0x0084
Sector 1 ECC error 3 position information register	SNFCS1EE3PI	0x0088
Sector 1 ECC error 4 position information register	SNFCS1EE4PI	0x008C
Sector 2 ECC error 1 position information register	SNFCS2EE1PI	0x0090
Sector 2 ECC error 2 position information register	SNFCS2EE2PI	0x0094
Sector 2 ECC error 3 position information register	SNFCS2EE3PI	0x0098
Sector 2 ECC error 4 position information register	SNFCS2EE4PI	0x009C
Sector 3 ECC error 1 position information register	SNFCS3EE1PI	0x00A0
Sector 3 ECC error 2 position information register	SNFCS3EE2PI	0x00A4
Sector 3 ECC error 3 position information register	SNFCS3EE3PI	0x00A8
Sector 3 ECC error 4 position information register	SNFCS3EE4PI	0x00AC
Sector 4 ECC error 1 position information register	SNFCS4EE1PI	0x00B0
Sector 4 ECC error 2 position information register	SNFCS4EE2PI	0x00B4
Sector 4 ECC error 3 position information register	SNFCS4EE3PI	0x00B8
Sector 4 ECC error 4 position information register	SNFCS4EE4PI	0x00BC
Sector 5 ECC error 1 position information register	SNFCS5EE1PI	0x00C0
Sector 5 ECC error 2 position information register	SNFCS5EE2PI	0x00C4
Sector 5 ECC error 3 position information register	SNFCS5EE3PI	0x00C8
Sector 5 ECC error 4 position information register	SNFCS5EE4PI	0x00CC
Sector 6 ECC error 1 position information register	SNFCS6EE1PI	0x00D0
Sector 6 ECC error 2 position information register	SNFCS6EE2PI	0x00D4
Sector 6 ECC error 3 position information register	SNFCS6EE3PI	0x00D8
Sector 6 ECC error 4 position information register	SNFCS6EE4PI	0x00DC
Sector 7 ECC error 1 position information register	SNFCS7EE1PI	0x00E0
Sector 7 ECC error 2 position information register	SNFCS7EE2PI	0x00E4
Sector 7 ECC error 3 position information register	SNFCS7EE3PI	0x00E8
Sector 7 ECC error 4 position information register	SNFCS7EE4PI	0x00EC
Sector 8 ECC error 1 position information register	SNFCS8EE1PI	0x00F0
Sector 8 ECC error 2 position information register	SNFCS8EE2PI	0x00F4
Sector 8 ECC error 3 position information register	SNFCS8EE3PI	0x00F8
Sector 8 ECC error 4 position information register	SNFCS8EE4PI	0x00FC

Peripheral function:SRST

Register name		Address (Base+)
Software reset protection register	SRSTPROTECT	0x0000
Peripheral function software reset register	SRSTIPRST	0x0004

11.3.2 Registers Description

11.3.2.1 Registers in the SNFC_IF Block

(1) SNFCENC (Enable 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	-	-	-	-	-	-	-	EN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	-	R	Read as "0".
0	EN	R/W	Enables/disables the SNFC circuit operation. 0: Disables the operation. 1: Enables the operation. Set <EN>=1 before setting the register of the SNFC. The registers of the SNFC cannot be set unless this bit to "1".

(2) SNFCECCMOD (ECC 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	-	-	-	-	-	-	GOUTMODE	SELBCH
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	-	R	Read as "0".
1	GOUTMODE	R/W	Selects a correction mode of BCH. 0: Corrected by software 1: Automatic correction
0	SELBCH	R/W	Selects mode of BCH. 0: BCH4+CRC 1: BCH8

(3) SNFCIE (Interrupt Enable Register)

	31	30	29	28	27	26	25	24
Bit symbol	-	DECCLR	DECFLG	DECEN	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	-	-	FAILCLR	FAILFLG	FAILEN	PRTCECLR	PRTCEFLG	PRTCEFEN8
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	PRTCEFEN8							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	PRTAECLR	PRTAEFLG	PRTAEEN2			SEQCLR	SEQFLG	SEQEN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	-	R	Read as "0".
30	DECCLR	R/W	Clears a decode completion interrupt. (INTSNFCDEC). [Write] 0: Invalid 1: Clears [Read] Always read as "0".
29	DECFLG	R	Indicates a flag of decode completion interrupt.INTSNFCDEC). 0: No interrupts 1: Interrupts occur.
28	DECEN	R/W	Enables a decode completion interrupt (INTSNFCDEC). 0: Disabled 1: Enabled
27	-	R/W	Write as "0". Read as "0".
26	-	R	Read as an undefined value.
25-24	-	R/W	Write as "0". Read as "0".
23	-	R	Read as an undefined value.
22	-	R/W	Write as "0". Read as "0".
21	FAILCLR	R/W	Clears an uncorrectable error detection interrupt (INTSNFCFAIL). [Write] 0: Invalid 1: Clears [Read] Always read as "0".
20	FAILFLG	R	Indicates the flag of which an uncorrectable error is detected (INTSNFCFAIL). 0: No interrupts 1: Interrupts occur.
19	FAILEN	R/W	Enables uncorrectable error detection interrupt (INTSNFCFAIL). 0: Disabled 1: Enabled
18	PRTCECLR	R/W	Clears a page RAM transfer C completion interrupt (INTSNFCPRTCE). [Write] 0: Invalid 1: Clears [Read] Always read as "0".

Bit	Bit Symbol	Type	Function
17	PRTCEFLG	R	Indicates the flag of page RAM transfer C completion interrupt (INTSNFCPRTCE). 0: No interrupts 1: Interrupts occur.
16-8	PRTCEFEN8 [8:0]	R/W	Enables a page RAM transfer C completion interrupt (INTSNFCPRTCE). 0_0000_0000: Interrupts are disabled. 1_1000_0000: Enabled when the sector number is 8. 1_0100_0000: Enabled when the sector number is 7. 1_0010_0000: Enabled when the sector number is 6. 1_0001_0000: Enabled when the sector number is 5. 1_0000_1000: Enabled when the sector number is 4. 1_0000_0100: Enabled when the sector number is 3. 1_0000_0010: Enabled when the sector number is 2. 1_0000_0001: Enabled when the sector number is 1. Other than the above-mentioned is prohibited. +Automatic correction (<GOUTMODE>=0): At the DMAC transfer data completion.
7	PRTAECLR	R/W	Clears a page RAM transfer A completion interrupt (INTSNFCPRTAE). [Write] 0: Invalid 1: Clears [Read] Always read as "0".
6	PRTAEFLG	R	Indicates the flag of page RAM transfer A completion interrupt (INTSNFCPRTAE). 0: No interrupts 1: Interrupts occur.
5 - 3	PRTAEEN2 [2:0]	R/W	Enables a page RAM transfer A completion interrupt (INTSNFCPRTAE). (DMAA transfers data completely.) 000: Interrupts are disabled. 110: The number of transfers is 1025 or more. An interrupt is enabled at 4 KB per page. 101: The number of transfers is 1024 or less. An interrupt is enabled at 2 KB per Page. Other than the above-mentioned is prohibited.
2	SEQCLR	R/W	Clears a command sequence completion interrupt (INTSNFCSEQ). [Write] 0: Invalid 1: Clears [Read] Always read as "0".
1	SEQFLG	R	Indicates a flag of command sequence completion interrupt (INTSNFCSEQ). 0: No interrupts 1: Interrupts occur.
0	SEQEN	R/W	Enables a command sequence completion interrupt (INTSNFCSEQ). 0: Disabled 1: Enabled

(4) SNFCPS (Page Size 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	-	-	-	CA				
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	CA							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-13	–	R	Read as "0".
12-0	CA[12:0]	R/W	Sets the page size of the NAND flash memory (1 to 4352). Zero is prohibited to be set.

(5) SNFCPRCS (Page Read Column 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	-	-	-	CAST				
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	CAST							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-13	-	R	Read as "0".
12-0	CAST[12:0]	R	Indicates the column address status in read operation (at one-page transfer). (Note) When the memory read operation is started at column address 2 through 4352, the information of nth column address is read.

(6) SNFCS (Sector 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	-	-	-	-	SEC			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	-	R	Read as "0".
3-0	SEC[3:0]	R/W	Sets the number of sectors when page read/write command is issued (1 to 8). Zero is prohibited to be set.

(7) SNFCSS (Sector 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	-	-	-	-	SECST			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	-	R	Read as "0".
3-0	SECST [3:0]	R	Indicates the status of the sector number that is being accessed or is in operation (at 1-page transfer). (Note) When the memory read operation is started at sector 2 through 8, the information of nth sector is read.

(8) SNFCDIC (Decode Input Count 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	-	-	-	-	-	-	DECIN	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	DECIN							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-10	-	R	Read as "0".
9-0	DECIN[9:0]	R/W	Sets the number of input bytes at decoding (the number of bytes per sector). Valid setting range in BCH8 mode: 14 to 544 Valid setting range in BCH4+CRC mode: 10 to 544

(9) SNFCDOC (Decode Output Count 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	-	-	-	-	-	-	DECOUT	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	DECOUT							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-10	-	R	Read as "0".
9-0	DECOUT[9:0]	R/W	Sets the number of output bytes at decoding (the number of bytes in 1 sector: data+management information). Valid setting range in BCH8 mode: 14 to 544 Valid setting range in BCH4+CRC mode: 10 to 544

(10) SNFCEIC (Encode input Count 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	-	-	-	-	-	-	ENCIN	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	ENCIN							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-10	-	R	Read as "0".
9-0	ENCIN[9:0]	R/W	Sets the input bytes at encoding. (1 sector: 512 bytes+management information) Valid setting range in BCH8 mode: 1 to 531 Valid setting range in BCH4+CRC mode: 1 to 535

(11) SNFCA1, 2 (Address Register 1, 2)

SNFCA1

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

Bit	Bit Symbol	Type	Function
31-16	PA[15:0]	R/W	Sets a page address (PA0 to 15)
15-13	-	R	Read as "0".
12-0	CA[12:0]	R/W	Sets a column address (CA0 to 12)

SNFCA2

	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	-	-	-	-	-	-	PA	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	-	R	Read as "0".
1-0	PA[1:0]	R/W	Sets a page address (PA16 to 17)

(12) SNFCW (Write 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	WRDATA							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	ALECODE							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-8	WRDATA[7:0]	R/W	Data setting for 1-byte write.
7-0	ALECODE [7:0]	R/W	1-byte address setting (When SNFCCS<ALE>=1 is set, a 1-byte address is used. Use a 1-byte address when an ID is read or other cases where a 1-byte address is required.)

(13) SNFCBIC (Bus Interface Control Register)

	31	30	29	28	27	26	25	24
Bit symbol	CLES		CLEH		ALES		ALEH	
After reset	0	1	0	0	1	0	0	1
	23	22	21	20	19	18	17	16
Bit symbol	WES		WEW			WEH		
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	RES		REW			REH		
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	DMYC2			DMYC1		RECYC		
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-30	CLES[1:0]	R/W	Sets the setup time of SNFCCLE. 00: 0 clocks 01: 1 clock 10: 2 clocks 11: 3 clocks
29-28	CLEH[1:0]	R/W	Sets the hold time of SNFCCLE. 00: 0 clocks 01: 1 clock 10: 2 clocks 11: 3 clocks
27-26	ALES[1:0]	R/W	Sets the setup time of SNFCALE. 00: 0 clocks 01: 1 clock 10: 2 clocks 11: 3 clocks
25-24	ALEH[1:0]	R/W	Sets the hold time of SNFCALE. 00: 0 clocks 01: 1 clock 10: 2 clocks 11: 3 clocks
23-22	WES[1:0]	R/W	Sets the setup time of $\overline{\text{SNFCWE}}$. 00: 1 clock 01: 2 clocks 10: 3 clocks 11: 4 clocks
21-19	WEW[2:0]	R/W	Sets the wait time of $\overline{\text{SNFCWE}}$. 000: 1 clock 001: 2 clocks 010: 3 clocks 011: 4 clocks 100: 5 clocks 101: 6 clocks 110: 7 clocks 111: 8 clocks
18-16	WEH[2:0]	R/W	Sets the hold time of $\overline{\text{SNFCWE}}$. 000: 0 clocks 001: 1 clock 010: 2 clocks 011: 3 clocks 100: 4 clocks 101: 5 clocks 110: 6 clocks 111: 7 clocks
15-14	RES[1:0]	R/W	Sets the setup time of $\overline{\text{SNFCRE}}$. 00: 1 clock 01: 2 clocks 10: 3 clocks 11: 4 clocks
13-11	REW[2:0]	R/W	Sets the wait time of $\overline{\text{SNFCRE}}$. 000: 1 clock 001: 2 clocks 010: 3 clocks 011: 4 clocks 100: 5 clocks 101: 6 clocks 110: 7 clocks 111: 8 clocks
10-8	REH[2:0]	R/W	Sets the hold time of $\overline{\text{SNFCRE}}$. 000: 0 clocks 001: 1 clock 010: 2 clocks 011: 3 clocks 100: 4 clocks 101: 5 clocks 110: 6 clocks 111: 7 clocks
7-5	DMYC2[2:0]	R/W	Sets the time of dummy period 2. 000: 1 clock 001: 2 clocks 010: 3 clocks 011: 4 clocks 100: 5 clocks 101: 6 clocks 110: 7 clocks 111: 8 clocks
4-3	DMYC1[1:0]	R/W	Sets the time of dummy period 1. (Sets the time until the next cycle occurs.) 00: 1 clock 01: 2 clocks 10: 3 clocks 11: 4 clocks
2-0	RECYC[2:0]	R/W	Sets the number of cycles when <RE[1:0]>of the SNFCCSx register is 01. 000: 1 cycle 001: 2 cycles 010: 3 cycles 011: 4 cycles 100: 5 cycles 101: 6 cycles 110: 7 cycles 111: 8 cycles

(14) SNFCCS 1 to 4 (Command Sequence Register 1 to 4)

	31	30	29	28	27	26	25	24
Bit symbol	CLE2	ALE	DMYB		BSY	DMYA	RE	
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	CMD2							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	CLE1	CA	PA	PA3	WE		-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	CMD1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	CLE2	R/W	Indicates the occurrence of SNFCCLE (second command or ID read) 0: No command latch signal occur. 1: A command latch signal occurs.
30	ALE	R/W	Indicates the occurrence of SNFCALE(1 byte address or ID read) 0: No address latch signal occur. 1: An address latch signal occurs.
29-28	DMYB[1:0]	R/W	Sets a dummy period and the number of clocks in a dummy period. 00 : No dummy period is set. 01 : A dummy period is set (the number of clocks of SNFCBIC<DMYC2> x 1). 10 : A dummy period is set (the number of clocks of SNFCBIC<DMYC2> x 2). 11 : A dummy period is set (the number of clocks SNFCBIC<DMYC2> x 3). When a busy cycle (<BSY>=1) is set, select the no dummy period setting (<DMYB[1:0]>=00) (Busy cycle). When no busy cycle (<BSY>=0) is set, select the time (<DMYB[1:0]>=00 to 11) up to the next cycle (Wait cycle).
27	BSY	R/W	Sets a busy cycle. 0: No busy cycle is set. 1: A busy cycle is set.
26	DMYA	R/W	Sets a dummy period when a busy cycle (<BSY>=1) is set.(the time between on the rising edge of SNFCRB and the next cycle execution.) 0: No dummy period is set. 1: A dummy period is set (the number of clocks of SNFCBIC<DMYC1>).
25-24	RE[1:0]	R/W	Sets read operation. 00 : No read operation is executed. 01 : Read operation + No µDMA transfers (Note) Read operation is executed for the number of cycles (1 to 8) that is specified with SNFCBIC <RECYC>. Read SNFCIR1 to 2. 10 : Read operation + No µDMA transfers (Note) Read operation is executed for 1 cycle regardless of the setting value of SNFCBIC <RECYC>. Read SNFCIR1. 11 : Read operation + µDMA transfers (Note) The number of read bytes is specified with SNFCPS<CA>. SNFCCSE <DECMODE>=1: Page read including BCH process. SNFCCSE <DECMODE>=0: Page read including BCH process or the case where an unique ID is 9 bytes or more.
23-16	CMD2[7:0]	R/W	Sets the command 2.
15	CLE1	R/W	Indicates the occurrence of SNFCCLE.

Bit	Bit Symbol	Type	Function
			0: No command latch signal occurs. 1: A command latch signal occurs.
14	CA	R/W	Sets a column address. 0: No column address is set. 1: A column address is set.
13	PA	R/W	Sets a page address. 0: No page address is set. 1: A page address is set.
12	PA3	R/W	Sets a (expansion) page address (when 2 GB/4 GB memory is used). 0: No (expansion) page address is set. 1: An (expansion) page address is set.
11-10	WE[1:0]	R/W	Sets write operation 00 : No write operation is executed. 01 : Reserved 10 : Write operation + No µDMA transfers (Note) Write the value of SNFCW<WRDATA> for 1 byte. 11 : Write operation + µDMA transfers The number of transfer bytes in each sector is set with SNFCEIC<ENCIN>.
9-8	-	R	Read as "0".
7-0	CMD1[7:0]	R/W	Sets the command 1.

(15) SNFCCSE (Command Sequence 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	DECMODE	RAMSEL	-	-	CMDSQ4	CMDSQ3	CMDSQ2	CMDSQ1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7	DECMODE	R/W	Selects decoding/encoding mode. 0: Encoding 1: Decoding
6	RAMSEL	R/W	Selects the destination RAM when sequential page read is performed. 0: RAM1 (Odd page) 1: RAM2 (Even page) (Note) This bit cannot be set with the CMDSQ1 bit to CMDSQ4 bit simultaneously. When a command is executed, specify the RAMSEL bit before the CMDSQx bit is set.
5-4	-	R	Read as "0".
3	CMDSQ4	R/W	[Write] 0: Invalid 1: Executes the command. (Run the sequence that is set with SNFCCS4.) [Read] 0: The command is being prepared. 1: The command is being executed.
2	CMDSQ3	R/W	[Write] 0: Invalid 1: Executes the command. (Run the sequence that is set with SNFCCS3.) [Read] 0: The command is being prepared. 1: The command is being executed.
1	CMDSQ2	R/W	[Write] 0: Invalid 1: Executes the command. (Run the sequence that is set with SNFCCS2.) [Read] 0: The command is being prepared. 1: The command is being executed.
0	CMDSQ1	R/W	[Write] 0: Invalid 1: Executes the command. (Run the sequence that is set with SNFCCS1.) [Read] 0: The command is being prepared. 1: The command is being executed.

(16) SNFCPRDB (Page Read Buffer Register)

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

Bit	Bit Symbol	Type	Function
31-0	PRDBUF [31:0]	R	Data read buffer accessed by the NAND flash memory.

(17) SNFCIR1, 2 (ID Read Register 1, 2)

SNFCIR1

	31	30	29	28	27	26	25	24
Bit symbol	IDRD4							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	IDRD3							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	IDRD2							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	IDRD1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-24	IDRD4[7:0]	R	Read data 4 created by a command (ID Read command, status read command, or other ID commands) other than the data transfer command.
23-16	IDRD3[7:0]	R	Read data 3 created by a command (ID Read command, status read command, or other ID commands) other than the data transfer command.
15-8	IDRD2[7:0]	R	Read data 2 created by a command (ID Read command, status read command, or other ID commands) other than the data transfer command.
7-0	IDRD1[7:0]	R	Read data 1 created by a command (ID Read command, status read command, or other ID commands) other than the data transfer command.

SNFCIR2

	31	30	29	28	27	26	25	24
Bit symbol	IDRD8							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	IDRD7							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	IDRD6							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	IDRD5							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-24	IDRD8[7:0]	R	Read data 8 created by a command (ID Read command, status read command, or other ID commands) other than the data transfer command.
23-16	IDRD7[7:0]	R	Read data 7 created by a command (ID Read command, status read command, or other ID commands) other than the data transfer command.
15-8	IDRD6[7:0]	R	Read data 6 created by a command (ID Read command, status read command, or other ID commands) other than the data transfer command.
7-0	IDRD5[7:0]	R	Read data 5 created by a command (ID Read command, status read command, or other ID commands) other than the data transfer command.

(18) SNFCEP1 to 4 (ECC Parity Register 1 to 4)

SNFCEP1

	31	30	29	28	27	26	25	24
Bit symbol	PRTY4							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	PRTY3							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	PRTY2							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	PRTY1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-24	PRTY4[7:0]	R	Parity data 4 at encoding in BCH8 mode or BCH4 mode.
23-16	PRTY3[7:0]	R	Parity data 3 at encoding in BCH8 mode or BCH4 mode.
15-8	PRTY2[7:0]	R	Parity data 2 at encoding in BCH8 mode or BCH4 mode.
7-0	PRTY1[7:0]	R	Parity data 1 at encoding in BCH8 mode or BCH4 mode.

SNFCEP2

	31	30	29	28	27	26	25	24
Bit symbol	PRTY8							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	PRTY7							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	PRTY6							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	PRTY5							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-24	PRTY8[7:0]	R	Parity data 8 at encoding in BCH8 mode.
23-16	PRTY7[7:0]	R	Parity data 7 at encoding in BCH8 mode or BCH4 mode.
15-8	PRTY6[7:0]	R	Parity data 6 at encoding in BCH8 mode or BCH4 mode.
7-0	PRTY5[7:0]	R	Parity data 5 at encoding in BCH8 mode or BCH4 mode.

SNFCEP3

	31	30	29	28	27	26	25	24
Bit symbol	PRTY12							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	PRTY11							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	PRTY10							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	PRTY9							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-24	PRTY12[7:0]	R	Parity data 12 at encoding in BCH8 mode.
23-16	PRTY11[7:0]	R	Parity data 11 at encoding in BCH8 mode.
15-8	PRTY10[7:0]	R	Parity data 10 at encoding in BCH8 mode.
7-0	PRTY9[7:0]	R	Parity data 9 at encoding in BCH8 mode.

SNFCEP4

	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	PRTY13							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7-0	PRTY13[7:0]	R	Parity data 13 at encoding in BCH8 mode.

(19) SNFCEC (ECC CRC 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	CRC2							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	CRC1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-8	CRC2[7:0]	R	CRC data 2 at encoding in BCH4 mode.
7-0	CRC1[7:0]	R	CRC data 1 at encoding in BCH4 mode.

11.3.2.2 Registers in the SNFC_ECC Block

(1) SNFCEWRB (ECC Write Buffer Register)

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

Bit	Bit Symbol	Type	Function
31-0	GIBUF[31:0]	R/W	Transfer data buffer at encoding or decoding.

11.3.2.3 Registers in the SNFC_GO Block

(1) SNFCCDRB (Correction Data Read Buffer Register)

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

Bit	Bit Symbol	Type	Function
31-0	RDBUF [31:0]	R	Transfer data buffer to the RAM3 at decoding. (Transfer data of <RDBUF[31:0]> to RAM3)

(2) SNFCEBS (ECC Busy 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	-	-	-	-	-	BSYST2	BSYST1	BSYST0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	-	R	Read as "0".
2	BSYST2	R	Sets the operation status 2 at decoding. 0: Chain search is stopping. 1: Chain search is in process.
1	BSYST1	R	Sets the operation status 1 at decoding. 0: σ calculation is stopping. 1: σ calculation is in process.
0	BSYST0	R	Sets the operation status 0 at decoding. 0: Syndrome is stopping. 1: Syndrome is in process.

(3) SNFCEES (ECC Error 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	ERR8	ERR7	ERR6	ERR5	ERR4	ERR3	ERR2	ERR1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7	ERR8	R	The result of error detection in Sector 8. 0: An error exists. 1: No errors exist.
6	ERR7	R	The result of error detection in Sector 7. 0: An error exists. 1: No errors exist.
5	ERR6	R	The result of error detection in Sector 6. 0: An error exists. 1: No errors exist.
4	ERR5	R	The result of error detection in Sector 5. 0: An error exists. 1: No errors exist.
3	ERR4	R	The result of error detection in Sector 4. 0: An error exists. 1: No errors exist.
2	ERR3	R	The result of error detection in Sector 3. 0: An error exists. 1: No errors exist.
1	ERR2	R	The result of error detection in Sector 2. 0: An error exists. 1: No errors exist.
0	ERR1	R	The result of error detection in Sector 1. 0: An error exists. 1: No errors exist.

(4) SNFCEDS1 to 8 (ECC Decode State Register 1 to 8)

This register is not updated if errors do not exist in automatic correction mode.

	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	-	-	-	-	-	-	DECST	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	ERRST3	ERRST2	ERRST1	ERRST0	SERR			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-10	-	R	Read as "0".
9-8	DECST[1:0]	R	Status 1 at decoding 00: No errors and no correction 01: Correctable error exists. 1x: Uncorrectable error exists.
7	ERRST3	R	Error status 3 at decoding 0: No CRC error after correction. 1: A CRC error exists after correction. (Note) Valid only in BCH4 mode + <GOUTMODE>=1
6	ERRST2	R	Error status 2 at decoding 0: No uncorrectable error is detected using chain search. 1: An uncorrectable error is detected using chain search.
5	ERRST1	R	Error status 1 at decoding 0 No uncorrectable error is detected at σ calculation. 1: An uncorrectable error is detected at σ calculation.
4	ERRST0	R	Error status 0 at decoding 0: No CRC error at Syndrome ALL 0. 1: An error exist at Syndrome ALL 0.
3-0	SERR[3:0]	R	The number of errors at decoding. 0000: No errors exist. 0001 to 1000: The number of errors. (For BCH4, 0001 to 0100.) 1111: Uncorrectable errors.

(5) SNFCSxEE1PI (Sector x ECC error 1 Position Information Register (x=1 to 8))

For uncorrectable errors, this register is invalid.

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	AADD2				
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	AADD2							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	-	-	-	AADD1				
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	AADD1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-29	-	R	Read as "0".
28-16	AADD2[12:0]	R	Error position information 2
15-13	-	R	Read as "0".
12-0	AADD1[12:0]	R	Error position information 1

(6) SNFCSxEE2PI (Sector x ECC error 2 Position Information Register (x=1 to 8))

For uncorrectable errors, this register is invalid.

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	AADD4				
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	AADD4							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	-	-	-	AADD3				
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	AADD3							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-29	-	R	Read as "0".
28-16	AADD4[12:0]	R	Error position information 4
15-13	-	R	Read as "0".
12-0	AADD3[12:0]	R	Error position information 3

(7) SNFCSxEE3PI (Sector x ECC error 3 Position Information Register (x=1 to 8))

For uncorrectable errors, this register is invalid.

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	AADD6				
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	AADD6							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	-	-	-	AADD5				
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	AADD5							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-29	-	R	Read as "0".
28-16	AADD6[12:0]	R	Error position information 6 (This bit is disabled when SNFCECCMOD<SELBCH>=0)
15-13	-	R	Read as "0".
12-0	AADD5[12:0]	R	Error position information 5 (This bit is disabled when SNFCECCMOD<SELBCH>=0)

(8) SNFCSxEE4PI (Sector x ECC error 4 position information register (x=1 to 8))

For uncorrectable errors, this register is invalid.

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	AADD8				
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	AADD8							
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	-	-	-	AADD7				
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	AADD7							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-29	-	R	Read as "0".
28-16	AADD8[12:0]	R	Error position information 8 (This bit is disabled when SNFCECCMOD<SELBCH>=0)
15-13	-	R	Read as "0".
12-0	AADD7[12:0]	R	Error position information 7 (This bit is disabled when SNFCECCMOD<SELBCH>=0)

11.3.2.4 SRSTPROTECT (Software Reset Protect 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	PROTECT							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7-0	PROTECT [7:0]	R/W	Enables the access to the SRSTIPRST register. 0x6B: Enabled Other than 0x6B: Disabled (The SRSTIPRST register cannot be accessed.) This bit becomes "0x00" and disabled after reset. To write data to the SRSTIPRST register, set <PROTECT[7:0]>=0x6B.

11.3.2.5 SRSTIPRST (Peripheral Function 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	-	-	-	IPRST4	IPRST3	IPRST2	IPRST1	IPRST0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	-	R	Read as "0".
4	IPRST4	R/W	SNFC reset [Write] 0: Releases the reset. 1: Performs the reset.
3	IPRST3	R/W	ESG reset [Write] 0: Releases the reset. 1: Performs the reset.
2	IPRST2	R/W	MLA reset [Write] 0: Releases the reset. 1: Performs the reset.
1	IPRST1	R/W	SHA reset [Write] 0: Releases the reset. 1: Performs the reset.
0	IPRST0	R/W	AES reset [Write] 0: Releases the reset. 1: Performs the reset.

11.4 ECC Function

The ECC incorporates an error detection/correction circuit to encode/decode data.

11.4.1 Encoding/Decoding

To select encoding or decoding, set the command sequence register (SNFCCSE<DECMODE>).

When data is written (encoding) to the SNFC, set "0" to <DECMODE>; when data is read (decoding) from the SNFC, set "1" to <DECMODE>.

11.4.2 The Number of Correction Bits

An error correction bit can be supported in BCH4+CRC mode that detects 4-bit/512 bytes error and BCH8 mode that detects 8-bit/512 bytes.

The correction mode can be selected with the ECC mode register (SNFCECCMOD<SELBCH>).

To use BCH4+CRC mode, set "0" to <SELBCH>. To use BCH8 mode, set "1" to <SELBCH>.

Table 11-4 BCH mode and the number of correction bits

BCH mode	The number of correction bits
BCH4+CRC	4 bits per 512 Bytes
BCH8	8 bits per 512 Bytes

11.4.3 Error Correction Mode

As an error correction function, software correction and automatic correction can be selected with the ECC mode register (SNFCECCMOD <GOUTMODE>).

To use software correction, set "0" to <GOUTMODE>. To use automatic correction, set "1" to <GOUTMODE>.

When the software correction is used, correct the software with the Sector x ECC error "n" position information register (SNFCSxEEnPI). (x: Sector number, n: Error number)

Table 11-5 shows the ECC mode at write/read operation.

Table 11-5 ECC mode at read/write operation.

<table>
<tr><th colspan="2" rowspan="2">Operation</th><th colspan="3">ECC mode</th></tr>
<tr><th>Decoding mode</th><th>BCH mode</th><th>Correction mode</th></tr>
<tr><th rowspan="2">The number of correction bits.</th><th>SNFCCSE<DECMODE>
0: Encoding
1: Decoding</th><th>SNFCECCMOD <SELBCH>
0: BCH4+CRC
1: BCH8</th><th>SNFCECCMOD <GOUTMODE>
0: Software correction
（No correction data is output.）
1: Automatic correction
（Correction data is output.）</th></tr>
<tr><td rowspan="2">Write</td><td>4 bits per 512 Bytes</td><td rowspan="2">0</td><td>0</td><td rowspan="2">-</td></tr>
<tr><td>8 bits per 512 Bytes</td><td>1</td></tr>
<tr><td rowspan="4">Read</td><td>4 bits per 512 Bytes, software correction</td><td rowspan="4">1</td><td rowspan="2">0</td><td>0</td></tr>
<tr><td>4 bits per 512 Bytes, automatic correction</td><td>1</td></tr>
<tr><td>8 bits per 512 Bytes, software correction</td><td rowspan="2">1</td><td>0</td></tr>
<tr><td>8 bits per 512 Bytes, automatic correction</td><td>1</td></tr>
</table>

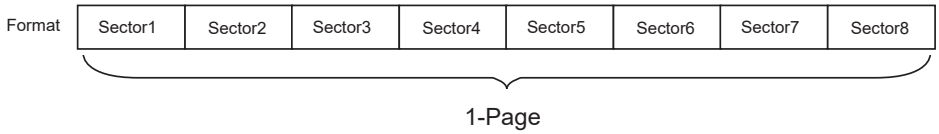
11.4.4 Data Format

The built-in ECC supports the format described in the later section.

11.4.4.1 Format of One Page

The ECC deals data in the units of sector; therefore, data format of one page consists of the several sectors.

The example of one page format : In case of eight sectors

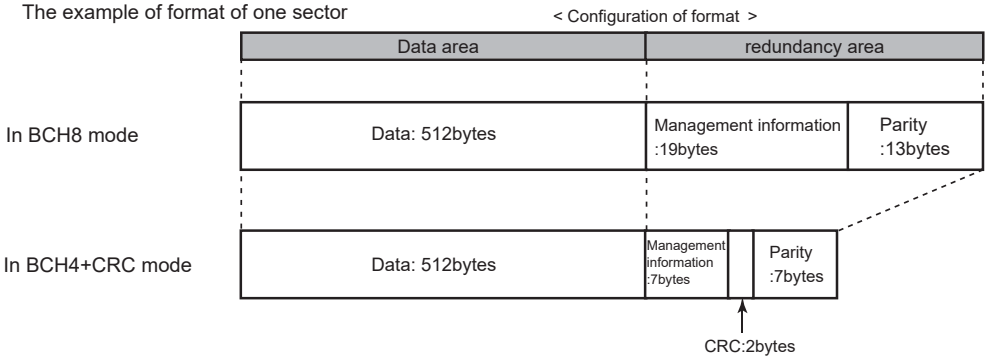


11.4.4.2 Format of One Sector

One sector consists of the data area and redundancy area.

A data area is fixed to 512 bytes. A redundancy area varies depending on the size of redundancy area of the memory to be used.

The example of format of one sector



Note that the configuration of the redundancy area varies depending on the page size of the memory and depending on BCH mode.

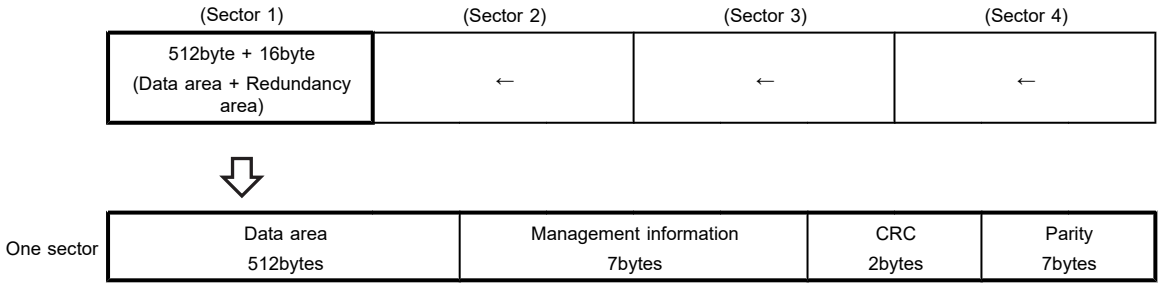
Table 11-6 shows the relationship between a page size and a sector format.

Table 11-6 The relationship between a page size and a sector format

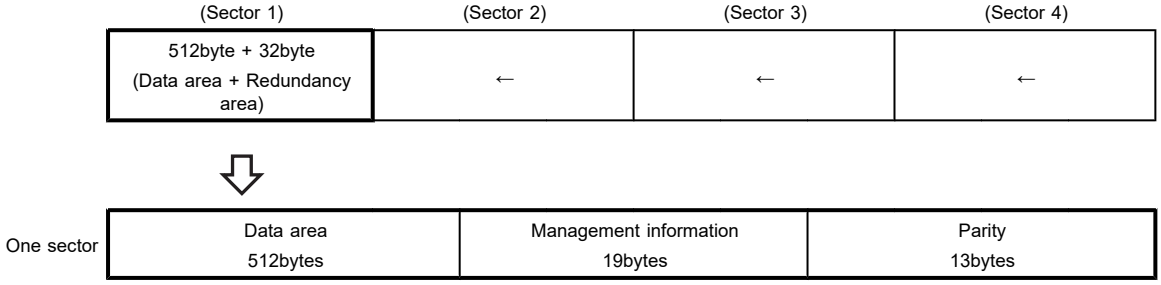
Page size [Data area+Redundancy area] (bytes)	1 sector [Data area+Redundancy area] (bytes)	Configuration of a redundancy area per sector			The number of sectors	Notes (BCH mode)
		Management information (bytes)	CRC (bytes)	Parity (bytes)		
2048 + 64	512 + 16	7	2 (Fixed)	7 (Fixed)	4	BCH4+CRC
2048 + 128	512 + 32	19	-	13 (Fixed)	4	BCH8
4096 + 224	512 + 28	19	2 (Fixed)	7 (Fixed)	8	BCH4+CRC
4096 + 256	512 + 32	19	-	13 (Fixed)	8	BCH8

<Example of a format>

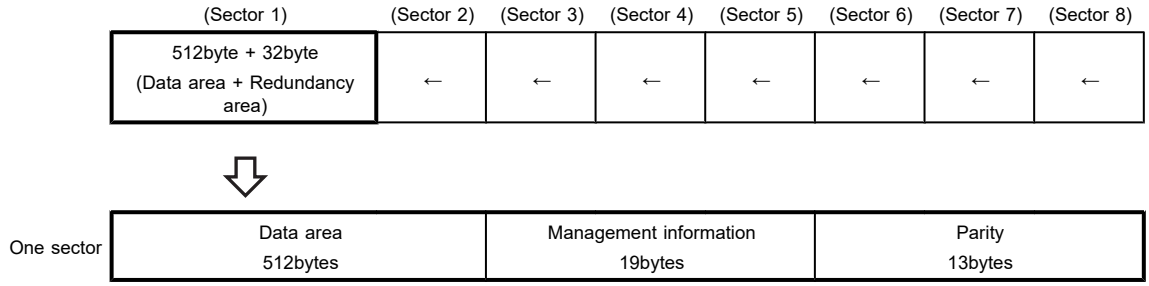
One page: 2048 bytes + 64 bytes (BCH4+CRC, 4 sectors)



One page: 2048 bytes + 128 bytes (BCH8, 4 sectors)



One page: 4096 bytes + 256 bytes (BCH8, 8 sectors)



11.4.5 Code Language and Format

11.4.5.1 BCH8 Mode

The figure below shows the code language on coding theory and an actual data (page) format used to be read/written from/to NAND flash in BCH8 mode when one sector is 544 bytes (data block: 512 bytes, management information block: 19 bytes, and parity block: 13 bytes).

Note: When the sizes of data block and management block are changed, the value of N in the symbol of i<N> is changed.

Read/write order from/to the NAND flash is the order of bit address expression (from i0, i1, to i4350, and i4351). This is indicated as the direction of broken line arrows. In the case of bytes unit, the order is from DD1st, DD2nd to DDlast-1, and DDlast).

Table 11-7 Data Line in BCH8 mode

	[7]	[6]		[1]	[0]		
Byte 0 (ED1st,DD1st)	i<7>	i<6>	←----------	i<1>	i<0>	Data block 512 Bytes	From the value of data and management information block
Byte 1 (ED2nd,DD2nd)	i<15>	i<14>	←----------	i<9>	i<8>		
. . .	←----------						
Byte 510	i<4087>	i<4086>	←----------	i<4081>	i<4080>		
Byte 511	i<4095>	i<4094>	←----------	i<4089>	i<4088>		
Byte 512	i<4103>	i<4102>	←----------	i<4097>	i<4096>	management information block 19 Bytes	
. .	←----------						
Byte 530 (EDlast)	i<4247>	i<4246>	←----------	i<4241>	i<4240>		
Byte 531	i<4255> PARITY[7]	i<4254> PARITY[6]	←----------	i<4249> PARITY[1]	i<4248> PARITY[0]	PARITY block 13 Bytes	Generates PARITY block
. . .	←----------						
Byte 542 (DDlast-1)	i<4333> PARITY[95]	i<4332> PARITY[94]	←----------	i<4327> PARITY[89]	i<4326> PARITY[88]		
Byte 543 (DDlast)	i<4351> PARITY[103]	i<4350> PARITY[102]	←----------	i<4345> PARITY[97]	i<4344> PARITY[96]		

Note 1: ED*: Input data in units of byte at encoding.
Note 2: DD*: Input data in units of byte at decoding.

11.4.5.2 BCH4+CRC Mode

The figure below shows the code language on coding theory and an actual data (page) format used to be read/written from/to NAND flash in BCH4+CRC mode when one sector is 528 bytes (data block: 512 bytes, management information block: 7 bytes, CRC: 2 bytes, and parity block: 7 bytes).

Note: When the sizes of data block and management block are changed, the value of N in the symbol of i<N> is changed.

Read/write order from/to the NAND flash is the order of bit address expression (from i0, i1, to i4350, and i4351). This is indicated as the direction of broken line arrows. In the case of bytes unit, the order is from DD1st, DD2nd to DDlast-1, and DDlast).

Table 11-8 Data Line in BCH4+CRC mode

*GIN[7:0] / SRCGIN[7:0]

	[7]	[6]				[1]	[0]			
Byte 0 (ED1st,DD1st)	i7	i6	←			i1	i0	Data block 512 Bytes	From the value of data and management information block	From the value of data, management information, CRC PARITY and Padding block
Byte 1 (ED2nd,DD2nd)	i15	i14	←			i9	i8			
. . .	←									
Byte 510	i4087	i4086	←			i4081	i4080			
Byte 511	i4095	i4094	←			i4089	i4088			
Byte 512	i4103	i4102	←			i4097	i4096	Management information 7 Bytes		
. .	←									
Byte 518	i4151	i4150	←			i4145	i4144			
Byte 519 (EDlast-1)	i4159 CRC16 [7]	i4158 CRC16 [6]	←			i4153 CRC16 [1]	i4152 CRC16 [0]	CRC PARITY block 2 Bytes	Generates CRC PARITY block	
Byte 520 (EDlast)	i4167 CRC16 [15]	i4166 CRC16 [14]	←			i4161 CRC16 [9]	i4160 CRC16 [8]			
Byte 521	i4175 PARITY [7]	i4174 PARITY [6]	i4173 PARITY [5]	i4172 PARITY [4]	i4171 PARITY [3] ←		i4168 PARITY [0]	PARITY block 7 Bytes		Generates the valid part of PARITY
. .	←									
Byte 526 (DDlast-1)	i4215 PARITY [47]	i4214 PARITY [46]	←			i4209 PARITY [41]	i4208 PARITY [40]			
Byte 527 (DDlast)	i4223 PARITY [55]	i4222 PARITY [54]	←			i4217 PARITY [49]	i4216 PARITY [48]			

Note 1: ED*: Input data in units of byte at encoding.
Note 2: DD*: Input data in units of byte at decoding.

11.5 Interrupts

The SLC NAND flash controller has the following five types of interrupts:

- Command sequence completion interrupt
- Page RAM transfer completion interrupt A
- Page RAM transfer completion interrupt C
- Uncorrectable error detection interrupt
- Decode completion interrupt

Each interrupt can be controlled with the interrupt enable register (SNFCIE).

Table 11-9 The relationship between interrupt factors, flags, and bits

Interrupt factor	Interrupt flag	Clearing bit	Enable bit
Command sequence is complete.	<SEQFLG>	<SEQCLR>	<SEQEN>
Page RAM transfer A ends.	<PRTAEFLG>	<PRTAECLR>	<PRTAEEN2>
Page RAM transfer C ends.	<PRTCEFLG>	<PRTCECLR>	<PRTCEFEN8>
Uncorrectable error is detected.	<FAILFLG>	<FAILCLR>	<FAILEN>
Decode is complete.	<DECFLG>	<DECCLR>	<DECEN>

11.5.1 Command Sequence Completion Interrupt

A command sequence completion interrupt (INTSNFCSEQ) occurs when the sequence specified with command sequence register (SNFCCS 1 to 4) is completed.

At reading, a command sequence completion interrupt occurs when data is read from the NAND flash completely. At writing, it occurs when data is written to the NAND flash completely (after on the rising edge of SNFCRB).

11.5.2 Page RAM Transfer Completion Interrupt A

A page RAM transfer completion interrupt A (INTSNFCPRTAE) occurs when read data from the NAND flash is transferred to the RAM completely using Unit A (DMAA) of the DMA.

11.5.3 Page RAM Transfer Completion Interrupt C

A page RAM transfer completion interrupt C (INTSNFCPRTCE) occurs when one-page error corrected data is transferred to the RAM3 completely using Unit C (DMAC) of the DMA in automatic correction mode (<GOUTMODE>=1).

11.5.4 Uncorrectable Error Detection Interrupt

An uncorrectable error detection interrupt (INTSNFCFAIL) occurs when data at decoding contains errors that are uncorrectable.

The details of uncorrectable data can be checked with the ECC decode state register (SNFCEDS 1 to 8).

11.5.5 Decode Completion Interrupt

A decode completion interrupt (INTSNFCDEC) occurs on the completion of decoding. At this time, decode status (SNFCEDSx) and error position information (SNFCSxEExPI) are determined. When software correction is used, check whether an error exists using this interrupt, and then correct it.

11.6 Precautions

11.6.1 Pin Connection

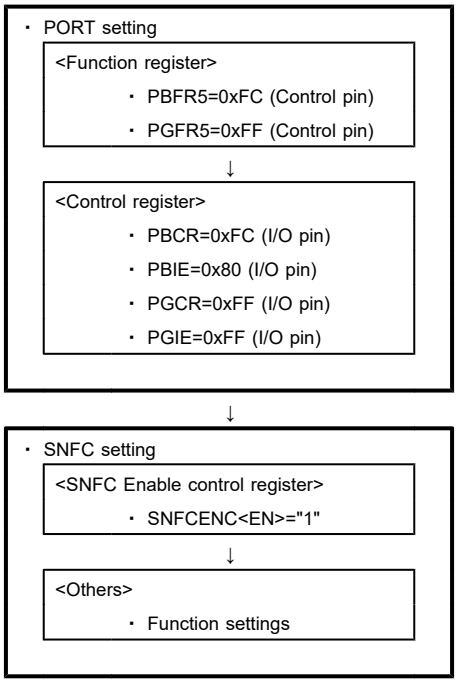
Connection pins of the SNFC between the NAND flash memory are shared with the general-purpose ports. These ports become high-impedance after reset.

Since these ports are high-impedance until the ports are set, take measures, such as placing pull-up resistor, to prevent an adverse effect on the connected memory.

11.6.2 Port Setting

When using the SLC NAND FLASH controller, set the SNFC after setting the ports.

Not to output unexpected signal to a control pin and a data bus, set control registers after setting function registers.



11.6.3 Software Reset

The SLC NAND flash controller can perform software reset to cancel a command during the command sequence and to initialize the controller.

To perform software reset, set "1" to SRSTIPRST<IPRST4> to reset the all internal circuits of the SLC NAND flash controller.

To release software reset, set "0" to SRSTIPRST<IPRST4>.

The SRSTIPRST register is write-protected after a cold reset or warm reset has been performed by the SRSTPROTECT register.

When a software reset is performed, write "0x6B" to the SRSTPROTECT register to be write enabled (unprotected).

11.6.4 Register Setting

Registers of the SLC NAND flash controller are specified by setting "1" to the enable bit of the enable control register SNFCENC<EN>.

Since the enable bit is "0" after reset, set "1" to the enable bit when the registers are set.

11.6.5 Command Execution

Command execution should be performed according to the specifications of the memory to be used.

The reset-command is only the command that can be executed while another command is being executed.

At decoding (SNFCCSE<DECMODE>=1) the next command execution is prohibited even if a command is complete (SNFCCSE<CMDSQx>=0) until one-page corrected data is transferred completely (INTSNFPRTCE).

However, sequential page reading is performed, the page read command can be executed by specifying the bit of the command sequence enable register (SNFCCSE<RAMSEL>).

For details, refer to the chapter on "11.7.4 Sequential Page Reading".

11.6.6 Check of Error Correction

When detailed information of an error is checked, confirm the error sector using the error status register (SNFCEES). Then check the corresponding sector of the decode status register (SNFCEDSx).

Note that the decode state register remains the state which holds the sector information of the previous page if there is no error in automatic correction mode.

In the case of software correction mode, the decode state register (SNFCEDSx) will be updated.

11.7 Operation Description

The SLC NAND flash controller (SNFC) executes error detection/correction at high-speed by combining the DMA and RAM when ECC codes are created or read.

11.7.1 Configuration

The SLC NAND flash controller consists of the interface control block and the ECC control block. The interface block controls the NAND flash memory; and the ECC control block executes error correction.

The DMA has 3 channels: DMAA, DMAB, and DMAC. Each DMA channel is used for RAM transfer from the NAND flash data, data transfer to the ECC control block, RAM transfer for automatic correction data respectively. The RAM1, RAM2, and RAM3 are used for temporary data storage.

- RAM1/RAM2: Storage for data from the NAND flash memory (odd page/even page)
- RAM3: Storage for automatic correction primary data (in automatic correction mode)
- DMAA: DMA used for page RAM transfer A to RAM1/RAM2 from the interface block.
- DMAB: DMA used for encoding/decoding data transfer B to the ECC control block from RAM1/RAM2
- EDMAC: DMA used for page RAM transfer C to RAM3 from the ECC control block.

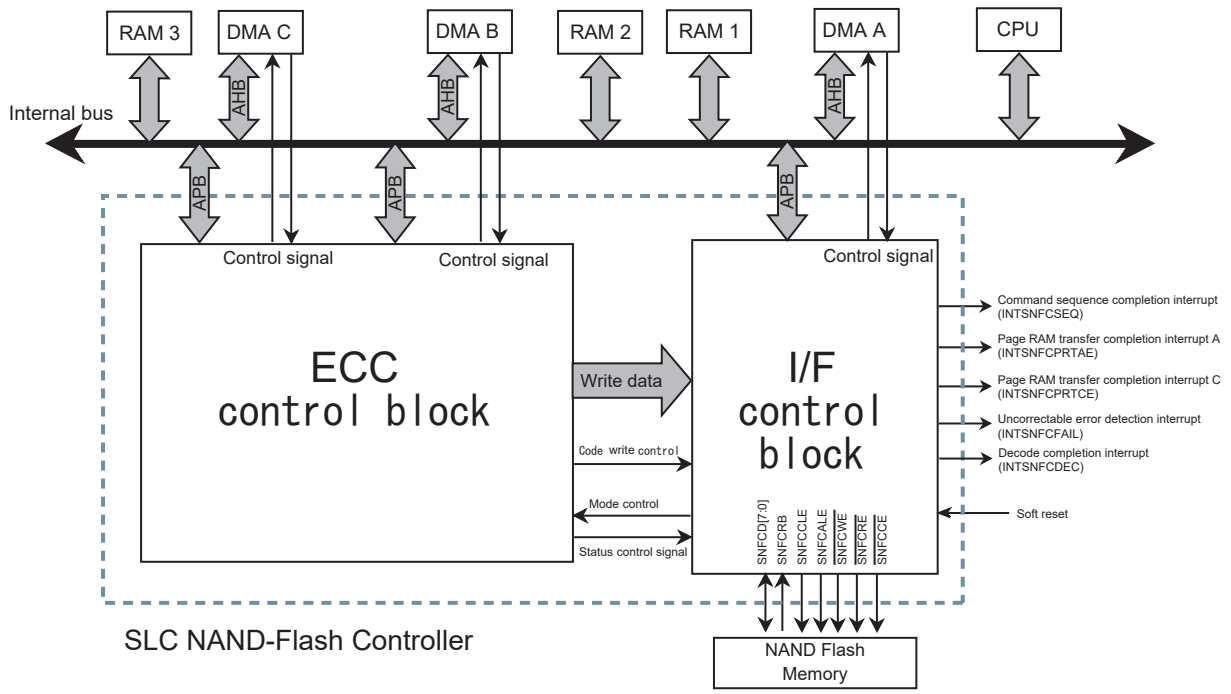


Figure 11-2 SLC NAND flash controller and Peripheral Functions

11.7.2 Page Write

11.7.2.1 Write Operation

The flowchart shows the basic operation at writing.

Table 11-10 Flowchart (Write operation)

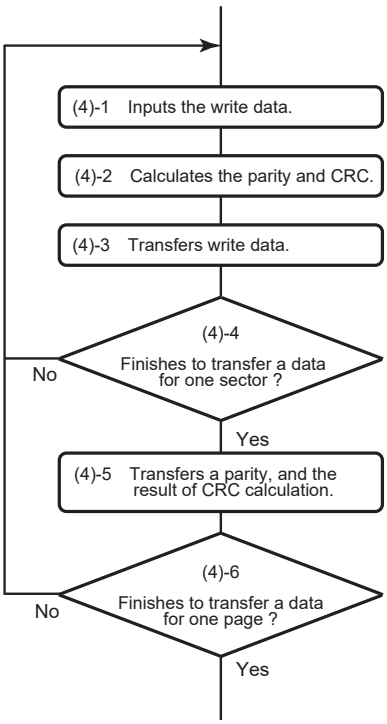
(1)	Starts a command sequence	…	Executes the page write command.
(2)	Issues the page write command	…	Issues the page write command "0x80" to the NAND flash memory.
(3)	Outputs the write address	…	Outputs the write address to the NAND flash memory.
(4)	Executes encoding and transfers write data.	…	Transfers data in the units of sector to the ECC control block from RAM1 using DMAB. Calculates the parity and CRC. Add the write transfer data, a parity, and the result of CRC calculation (at encoding one-page data).
(5)	Issues the write execution command	…	Issues the write execution command "0x10".
(6)	Wait until data is written to the NAND flash completely.	…	Wait until the SNFCRB pin becomes ready state.
(7)	Writing is complete and the interrupt occurs.	…	A command sequence completion interrupt (INTSNFCSEQ) occurs.
(8)	Command execution ends		

(Note) ▭ : Practice by the software

▭ : Practice with the hardware

The below subsection explains the process (4) "data encoding and data transfer".

Table 11-11 (4) data encoding and data transfer



(4)-1: Inputs the write data
The data which is written into ECC controller is read from RAM1.

(4)-2: Calculates the parity and CRC
Calculates the parity and CRC every byte.

(4)-3: Transfers write data.
Transfers a data to NAND FLASH memory.

(4)-4: Finishes to transfer a data for one sector?
Continues to executes (4)-1 to (4)-3 for one sector.

(4)-5: Transfers a result of parity and CRC calculations.
Transfers a result of parity and CRC calculations after transferring a data.

(4)-6: Finishes to transfer a data for one page?
Continues to execute (4)-1 to (4)-5 for one page.
Executes (5) A command for writing after completing the process for one page.

11.7.2.2 Data Flow at Write Operation

This section explains the process of (4) data encoding and data transfer of the writing data.

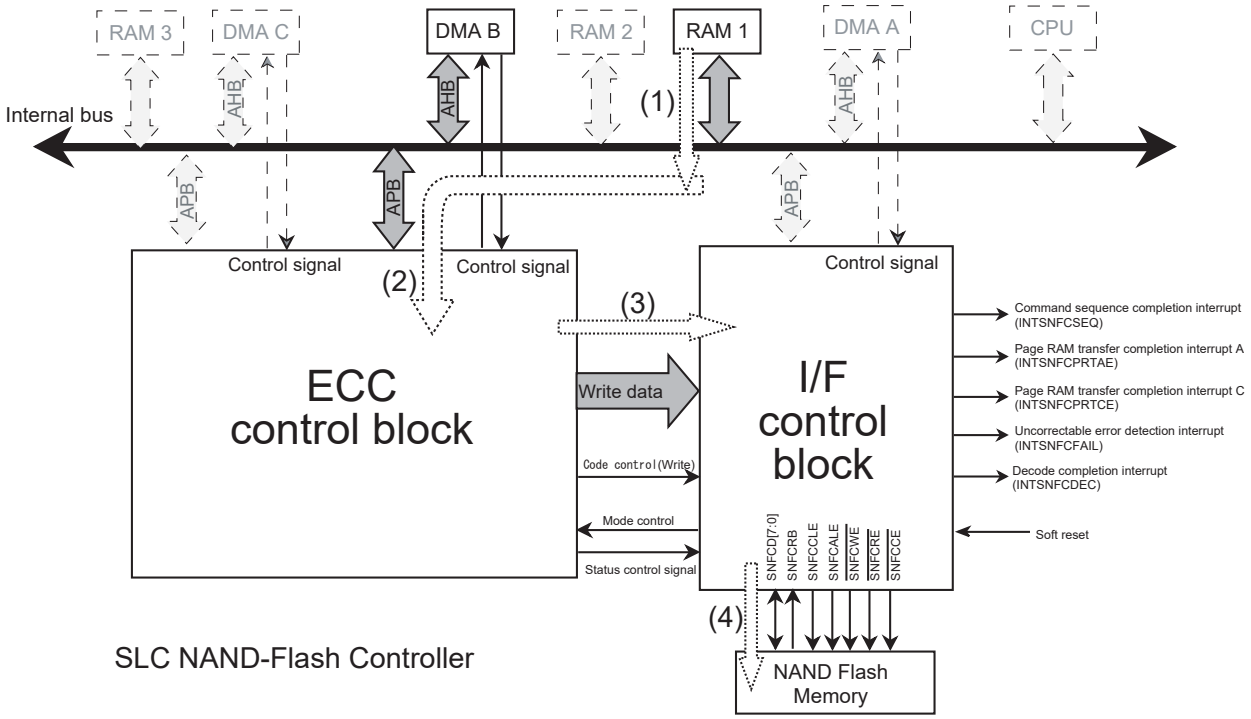


Figure 11-3 The data flow when writing

(1) Page Write

When a command is executed, one sector data (512 bytes + management information) is transferred from the RAM1 to ECC Write Buffer Register (SNFCEWRB <GIBUF>) of the ECC control block in the units of one word using the DMAB (described in the Table 11-11 (4)-1).

(2) The ECC control block converts transferred one word data to the data in the units of bytes sequentially.

Transferred data is used for parity calculation and CRC calculation (*1). This is called encoding (described in the Table 11-11 (4)-2).

(3) The ECC control block transfers the converted data that is converted to the units of byte to the interface control block sequentially. After one sector data (512 bytes + management information) is transferred, the calculated parity and CRC data (*1) is transferred to the interface block (described in the process Table 11-11 (4)-3).

(4) The interface control block transfers (writes) the data (512 bytes + management information) to the NAND flash memory in the units of byte sequentially.

After one sector data (512 bytes + management information) is transferred, the transferred parity and CRC data (*1) is transferred (written) to the NAND flash memory in the units of byte (described in the Table 11-11 (4)-5).

Note 1: CRC data is used only in BCH4 + CRC mode.

The processes (1) to (4) above mentioned are the data flow for one sector. If multiple sectors are transferred, the processes (1) to (4) should be executed several times.

After data of all sectors is transferred completely, the controller issues the write execution command (5). This indicates writing is complete (the SNFC becomes ready state from busy state). At this time, a command sequence completion interrupt (INTSNFCSEQ) occurs. After that writing process is complete and the page write command is issued.

11.7.3 Page Read

11.7.3.1 Read Operation

This section explains the basic process flow of read operation.

Table 11-12 Flowchart (Read)

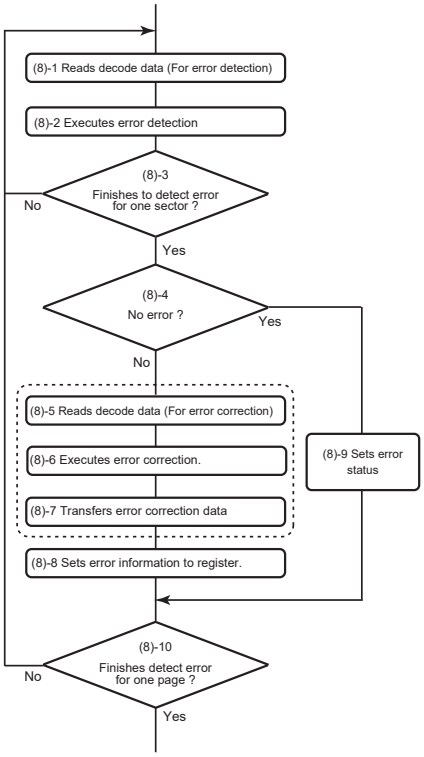
(1)	Starts a command sequence.	…	Executes the page read command.
(2)	Issues the page read command.	…	Issues the page read command "0x00" to the NAND flash memory.
(3)	Outputs the read address.	…	Outputs the read address to the NAND flash memory.
(4)	Executes the read command.	…	Issues the read execution command "0x30" to the NAND flash memory.
(5)	Wait until data is read from the NAND flash completely.	…	Wait until the SNFCRB pin becomes ready state.
(6)	Data reading (Transferred to the RAM1)	…	Data is read from the NAND flash memory. (Transferred to the RAM1 simultaneously) RAM1: for odd page
(7)	Data reading is complete. An interrupt occurs.	…	A command sequence completion interrupt (INTSNFCSEQ) and page RAM transfer completion A interrupt (INTSNFCPRTAE) occur.
(8)	Decoding is executed.	…	Using the DMAB data is transferred to the ECC control block from the RAM1 to perform decoding. When automatic correction is used, corrected data is transferred to the RAM3 using DMAC.
(9)	Error correction data is transferred completely. An interrupt occurs.	…	A page RAM transfer completion C interrupt (INTSNFCPRTCE) occurs.
(10)	Decoding ends.		

(Note) : Practice by the software

: Practice with the hardware

The following figure shows the process (8) decoding (automatic correction).

Table 11-13 (8) decoding (automatic correction)



(8)-1: Reads decode data (For error detection)
Read a data for error detection from RAM1.

(8)-2 : Executes error detection
Executes error detection every one byte.

(8)-3 : Finishes to detect error for one sector?
Executes error detection for one sector.

(8)-4 : No error?
If no error for one sector, error correction is not executed.
Only if errors exist, error correction is executed. (When automatic error correction)

(8)-5 : Reads decode data (For error correction)
Reads a data as same as error detection for error correction.

(8)-6 : Executes error correction.
Executes error correction every one byte.

(8)-7 : Transfers error correction data
Transfers the generated error correction data to RAM3.

(8)-8 : Set error information to registers.
Set a detected error information to registers.
- ECC decode state register (SNFCEDSx)
- ECC error n position information register (SNFCSxEEnPI)

If an error correction can not be executed by error information, an uncorrectable error detection interrupt.

(8)-9 : Sets error status.
Sets status flag in (8)-4.
- ECC Error status register (SNFCEES)

(8)-10 : Completes for one page?
Executes sectors for one page and completes to decode.

Note that when software correction is used, the processes ((8)-5 to 7) indicated by dashed lines are not executed. Based on error information, correct errors by software.

11.7.3.2 Data Flow at Read Operation

This section explains data flow of the processes from (6) to (9).

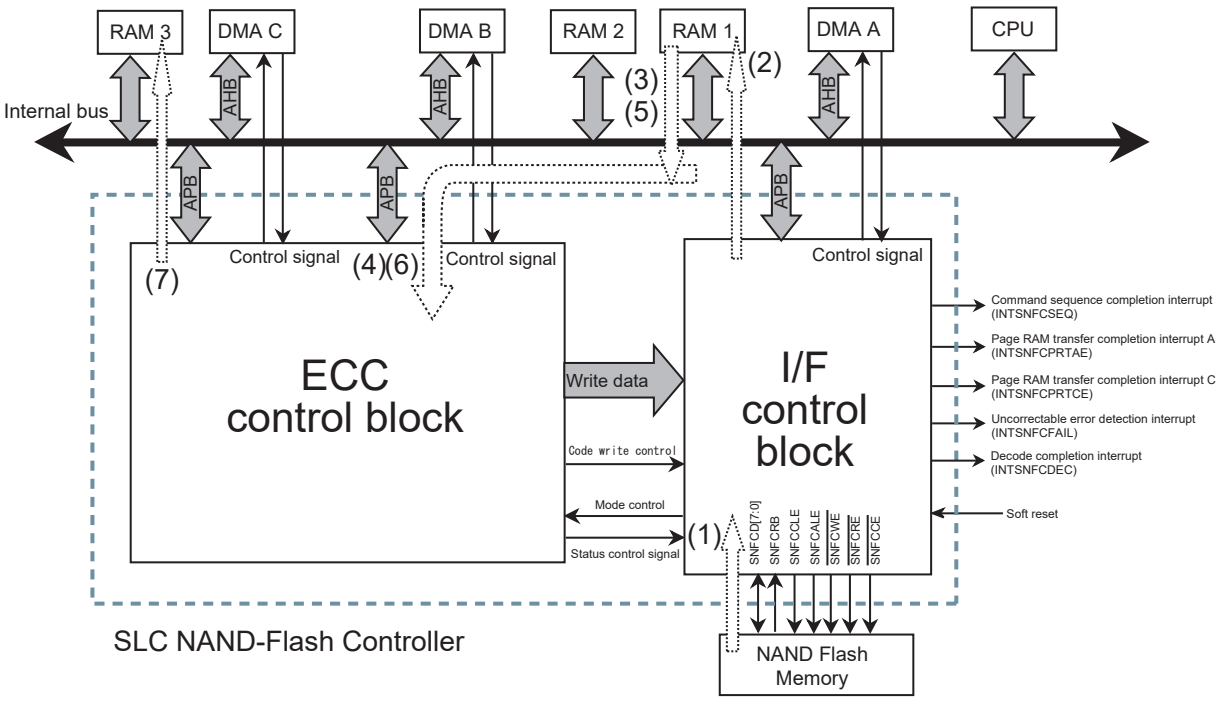


Figure 11-4 The data flow when reading

<Page read when automatic correction is used.>

(1) When the page read command is executed, the interface control block reads one-page data from the NAND flash memory in the units of byte (described in Table 11-12 (6)).

(2) Read data that is read in the units of byte is converted to the units of one-word (4 bytes) in the interface control block and is transferred to the RAM1 in the units of one-page sequentially using the DMAA.

When one-page data is transferred, a page RAM transfer completion A interrupt (INTSNFCPRTAE) and command sequence completion interrupt (INTSNFCSEQ) occur (described in Table 11-12 (7)).

(3) After one-page data is transferred to the RAM1, one-sector data is transferred to the ECC write buffer register (SNFCEWRB <GIBUF>) of the ECC control block in the units of word (4 bytes) sequentially described in Table 11-13 (8)-1).

(4) The ECC control block converts transferred one-word data to the data in the units of byte sequentially and it executes error detection (described in Table 11-13 (8)-2).

(5) After one-sector data is transferred and error detection is complete, data (same as (3)) is sequentially transferred to the ECC write buffer register (SNFCEWRB <GIBUF>) of the ECC control block to perform error detection again using the DMAB (described in Table 11-13 (8)-5).

(6) The ECC control block sequentially converts the one-word transferred data to the data in units of byte to create error correction data (described in Table 11-13 (8)-6).

(7) Creation of error correction data is sequentially performed in the units of byte after data transfer.

The created data is converted to the data in the units of word (4 bytes). One-sector corrected data is transferred to the RAM3 using the DMAC (described in Table 11-13 (8)-7).

Note 1: The processes from (3) to (7) are executed for the number of sectors.

Note 2: When software correction is used, the processes (5) to (7) are not executed. It is necessary to read correction information from the ECC control block and to deal the data.

Note 3: Even if an error is not detected in the process (4), the processes from (5) to (7) are not executed.

The processes from (3) to (7) above mentioned explains the data flow for one sector. If multiple sectors are transferred, the processes from (3) to (7) should be executed several times.

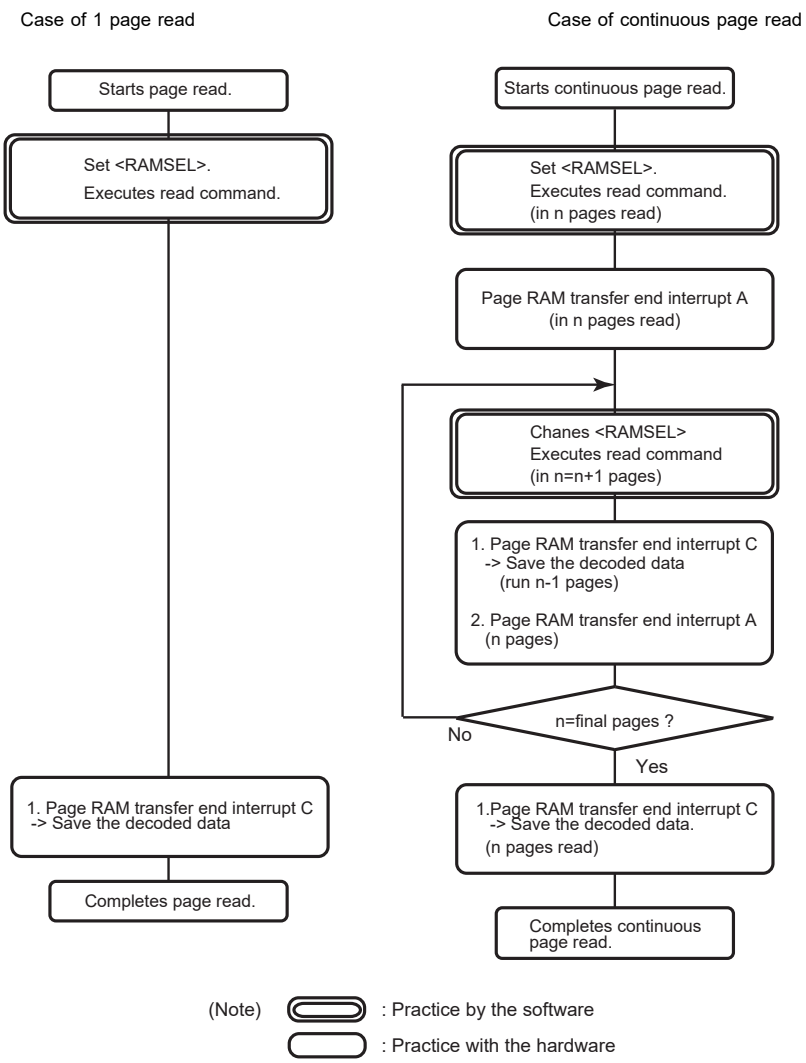
When decoding is complete, when the last sectors of one-page data is complete, a page RAM transfer completion C interrupt (INTSNFCPRTCE) occurs.

11.7.4 Sequential Page Reading

To perform sequential page reading, the command sequence enable register (SNFCCSE<RAMSEL>) can divide the RAM area, which stores data, into odd-numbered pages and even-numbered pages.

Note: Only the reset command can be executed under the following commands execution; however, the page read command can also be executed by specifying <RAMSEL>.

The figure below shows the flowchart of page reading (including data correction).



11.7.5 Data after Decoding Process

Decoded data by the ECC is stored in the different location depending on the correction mode, odd/even page, and corrected/non-corrected data.

For the data after correction, see the table below:

Table 11-14 The relationship between correction mode and data after correction

Correction mode	<GOUTMODE>= 0 (Software correction)						<GOUTMODE>= 1 (Automatic correction)					
Page	Odd page			Even page			Odd page			Even page		
Sector correction *1	Sector contains error correction.		No sector correction	Sector contains error correction.		No sector correction	Sector contains error correction.		No sector correction	Sector contains error correction.		No sector correction
	Uncorrectable	Correctable		Uncorrectable	Correctable		Uncorrectable	Correctable		Uncorrectable	Correctable	
Data after correction	Option *2	RAM1 *3	RAM1	Option *2	RAM2 *3	RAM2	Disabled (RAM3)	RAM3	RAM1	Disabled (RAM3)	RAM3	RAM2

Note 1: Use the ECC error status register (SNFCEES) to check whether the data is the corrected sector or non-corrected sector,

Note 2: Uncorrectable sectors are optional.

Note 3: When sectors are corrected on the RAM1 or RAM2 by software, the other settings are optional.

11.8 Operation Setting

The SLC NAND flash controller is specified with the SNFC Command sequence register (SNFCCSx). The command sequence and operation sequence are set to the SNFC Command sequence register (SNFCCSx).

To execute the command, set "1" to the <CMDSQx> bit of the SNFC Command Sequence Enable Register (SNFCCSE).

11.8.1 Command

The table below shows an example of major operation commands:

When an operation command is set, specify the command issued to the NAND flash memory and the operation timing.

Table 11-15 Setting value of the major six commands example of the command sequence register

Command	Setting code (SNFCCSx)	Setting bits of the SNFC command sequence register																	
		31	30	29	28	27	26	25	24	23-16	15	14	13	12	11	10	9	8	7-0
		CLE2	ALE	DMYB		BSY	DMYA	RE		CMD2	CLE1	CA	PA	PA3	WE		-	-	CMD1
Page Read	0x8F30_F000	1	0	0	0	1	1	1	1	0x30	1	1	1	1 (Note 1)	0	0	-	-	0x00
Page Write	0x8810_FC80	1	0	0	0	1	0	0	0	0x10	1	1	1	1 (Note 1)	1	1	-	-	0x80
Block Erase	0x88D0_B060	1	0	0	0	1	0	0	0	0xD0	1	0	1	1 (Note 1)	0	0	-	-	0x60
Status Read	0xA270_00XX	1	0	1	0	0	0	1	0	0x70	0	0	0	0	0	0	-	-	XX (Note 2)
ID Read	0xC190_00XX	1	1 (Note 3)	0	0	0	0	0	1	0x90	0	0	0	0	0	0	-	-	XX (Note 2)
Reset	0x88FF_00XX	1	0	0	0	1	0	0	0	0xFF	0	0	0	0	0	0	-	-	XX (Note 2)

Note 1: Set "0" (4-cycle address) if the expansion address (5-cycle address) is not used.
Note 2: Set <CLE1>=0, this value is optional because the first command latch is not necessary.
Note 3: When <ALE>=1 is set, specify an address value to SNFCW<ALECODE[7:0].

11.8.2 Automatic Load Function

The automatic load function transfers data based on the combination of the DMA controller and the NAND flash controller.

The table below shows the allocation of the DMA channels corresponding to the usage of the NAND flash controller. The NAND flash controller starts up the DMA to execute the transfer data in each operation.

Table 11-16 DMA and the channel to be used (Unit A)

DMA	ch (demand)	Target sector	Description	Transfer	DMA to be used	
					When writing (4 sectors / 8 sectors)	When reading (4 sectors / 8 sectors)
Unit A (DMAA)	ch0 (PRD11)	Sectors 1 to 4	For the transfer of one-page read data (Odd page)	I/F control block ↓ RAM (At decoding)	non use / non use	use / use
	ch1 (PRD12)	Sectors 5 to 8				non use / use
	ch2 (PRD21)	Sectors 1 to 4	For the transfer of one-page read data (Even page)			use / use
	ch3 (PRD22)	Sectors 5 to 8				non use / use

Table 11-17 DMA and the channel to be used (UnitB)

DMA	ch (demand)	Target sector	Description	Transfer	DMA to be used	
					When writing (4 sectors / 8 sectors)	When reading (4 sectors / 8 sectors)
Unit B (DMAB)	ch0 (GIE1)	Sector 1	For the transfer of one-page write data	RAM ↓ ECC control block (At encoding)	use / use	non use / non use
	ch1 (GIE2)	Sector 2				
	ch2 (GIE3)	Sector 3				
	ch3 (GIE4)	Sector 4				
	ch4 (GIE5)	Sector 5			non use / use	
	ch5 (GIE6)	Sector 6				
	ch6 (GIE7)	Sector 7				
	ch7 (GIE8)	Sector 8				
	ch8 (GID11)	Sector 1	For the transfer of error correction data (Odd page)	RAM ↓ ECC control block (At decoding)	non use / non use	use / use
	ch9 (GID12)	Sector 2				
	ch10 (GID13)	Sector 3				
	ch11 (GID14)	Sector 4				
	ch12 (GID15)	Sector 5				non use / use
	ch13 (GID16)	Sector 6				
	ch14 (GID17)	Sector 7				
	ch15 (GID18)	Sector 8				
	ch16 (GID21)	Sector 1	For the transfer of error correction data (Even page)		non use / non use	use / use
	ch17 (GID22)	Sector 2				
	ch18 (GID23)	Sector 3				
	ch19 (GID24)	Sector 4				
	ch20 (GID25)	Sector 5				non use / use
	ch21 (GID26)	Sector 6				
	ch22 (GID27)	Sector 7				
	ch23 (GID28)	Sector 8				

Table 11-18 DMA and the channel to be used (Unit C)

<table>
<tr><th rowspan="2">DMA</th><th rowspan="2">ch
(demand)</th><th rowspan="2">Target sector</th><th rowspan="2">Description</th><th rowspan="2">Transfer</th><th colspan="2">DMA to be used</th></tr>
<tr><th>When writing
(4 sectors / 8 sectors)</th><th>When reading
(4 sectors / 8 sectors)</th></tr>
<tr><td rowspan="8">Unit C
(DMAC)</td><td>ch0
(RD1)</td><td>Sector 1</td><td rowspan="8">For the transfer of corrected data
↓
(Only when the automatic correction is used.)</td><td rowspan="8">ECC control block
↓
RAM
(At decoding)</td><td rowspan="8">non use / non use</td><td rowspan="4">use / use</td></tr>
<tr><td>ch1
(RD2)</td><td>Sector 2</td></tr>
<tr><td>ch2
(RD3)</td><td>Sector 3</td></tr>
<tr><td>ch3
(RD4)</td><td>Sector 4</td></tr>
<tr><td>ch4
(RD5)</td><td>Sector 5</td><td rowspan="4">non use / use</td></tr>
<tr><td>ch5
(RD6)</td><td>Sector 6</td></tr>
<tr><td>ch6
(RD7)</td><td>Sector 7</td></tr>
<tr><td>ch7
(RD8)</td><td>Sector 8</td></tr>
</table>

11.8.3 Example of Setting at Page Write (Encoding)

This section shows the example of the case where data is transferred to the NAND flash controller from RAM1 at page write (encoding).

Data is written to the NAND flash memory by transferring data to the NAND flash controller.

11.8.3.1 Setting Conditions

The table below list the setting conditions:

Table 11-19 Setting conditions

Target memory		Capacity	: 4 GB
		Page size	: 4096+256 bytes
		1 sector	: 512+32 bytes
		The number of sectors	: 8 sectors
		ECC	: BCH8 mode
		Write address page	: 0x1_0000
DMA to be used	:Unit B : DMAB RAM ↓ ECC control block	Channels to be used	: 0 to 7ch (8 sectors)
		Channel control data start address	: 0x2003_F800 (DMABCtrlBasePtr=0x2003_F800)
		Operation mode	: Basic mode (Control<cycle_ctrl>="0x001")
		Source	: RAM1 * Start address: 0x2004_0000 (Source End Pointer=0x2004_0210) * Last address setting. The setting value can be changed in each channel.
		Destination (Fixed)	: SNFCEWRB register of the SNFC (Address: 0x4005_C800) (Destination End Pointer=0x4005_C800)
		The number of transfers	: 133 times per channel (Control<n_minus_1>="00_1000_0100")
		Transfer size	: Source: 4 bytes/Destination: 4 bytes (Control<src_size>="10", Control<dst_size>="10")
SNFC		Execution mode	: Encoding (SNFCCSE<DECMODE>="0")
		BCH mode	: BCH8 (SNFCECCMOD<SELBCH>="1")
		Page/column address	: 0x1_0000_0000 (SNFCA1=0x0000_0000, SNFCA2=0x0000_0001)
		Encode input	: 531 bytes *Data (512 bytes) + management information (19 bytes) (SNFCEIC=0x0000_0213)
		The number of sectors	: 8 sectors (SNFCS=0x0000_0008)
		Interrupt	: Command sequence completion interrupt (SNFCIE=0x0000_0001)

11.8.3.2 Example of DMA (DMAB) Setting

The DMAB setting is required to be set to the control registers and to be set a control data to RAM.

Table 11-20 (1) DMAB register

Register name	Address	Setting code				Note
DMABStatus	0x4004_D000	-				Read register
DMABCfg	0x4004_D004	00	00	00	01	Operation setting
DMABCtrlBasePtr	0x4004_D008	20	03	F8	00	Primary data base pointer setting
DMABAltCtlBasePtr	0x4004_D00C	-				Undefined
DMABChnlSwRequest	0x4004_D014	00	00	00	00	Software request setting
DMABChnlUseburstSet	0x4004_D018	-				Undefined
DMABChnlUseburstClr	0x4004_D01C	-				Undefined
DMABChnlReqMaskSet	0x4004_D020	FF	FF	FF	00	Request mask setting
DMABChnlReqMaskClr	0x4004_D024	-				Undefined
DMABChnlEnableSet	0x4004_D028	00	00	00	FF	Operation enable setting
DMABChnlEnableClr	0x4004_D02C	-				Undefined
DMABChnlPriAltSet	0x4004_D030	-				Undefined
DMABChnlPriAltClr	0x4004_D034	-				Undefined
DMABChnlPrioritySet	0x4004_D038	-				Undefined
DMABChnlPriorityClr	0x4004_D03C	-				Undefined
DMABErrClr	0x4004_D04C	-				Undefined
DMAIFFLGB	0x4005_F004	-				Read register

Table 11-21 (2) DMAB channel control data

Channel control data name	RAM	Setting code				Note
Reserved	0x2003_F80C	-				ch0 of DMAB (For primary data)
Control	0x2003_F808	EA	00	08	41	
Destination End Pointer	0x2003_F804	40	05	C8	00	
Source End Pointer	0x2003_F800	20	04	02	10	
Reserved	0x2003_F81C	-				ch1 of DMAB (For primary data)
Control	0x2003_F818	EA	00	08	41	
Destination End Pointer	0x2003_F814	40	05	C8	00	
Source End Pointer	0x2003_F810	20	04	04	24	
Reserved	0x2003_F82C	-				ch2 of DMAB (For primary data)
Control	0x2003_F828	EA	00	08	41	
Destination End Pointer	0x2003_F824	40	05	C8	00	
Source End Pointer	0x2003_F820	20	04	06	38	
Reserved	0x2003_F83C	-				ch3 of DMAB (For primary data)
Control	0x2003_F838	EA	00	08	41	
Destination End Pointer	0x2003_F834	40	05	C8	00	
Source End Pointer	0x2003_F830	20	04	08	4C	
Reserved	0x2003_F84C	-				ch4 of DMAB (For primary data)
Control	0x2003_F848	EA	00	08	41	
Destination End Pointer	0x2003_F844	40	05	C8	00	
Source End Pointer	0x2003_F840	20	04	0A	60	

Table 11-21 (2) DMAB channel control data

Channel control data name	RAM	Setting code				Note
Reserved	0x2003_F85C	-				ch5 of DMAB (For primary data)
Control	0x2003_F858	EA	00	08	41	
Destination End Pointer	0x2003_F854	40	05	C8	00	
Source End Pointer	0x2003_F850	20	04	0C	74	
Reserved	0x2003_F86C	-				ch6 of DMAB (For primary data)
Control	0x2003_F868	EA	00	08	41	
Destination End Pointer	0x2003_F864	40	05	C8	00	
Source End Pointer	0x2003_F860	20	04	0E	88	
Reserved	0x2003_F87C	-				ch7 of DMAB (For primary data)
Control	0x2003_F878	EA	00	08	41	
Destination End Pointer	0x2003_F874	40	05	C8	00	
Source End Pointer	0x2003_F870	20	04	10	9C	

11.8.3.3 Example of the SNFC Setting

Register name	Address	Setting code				Note
SNFCENC	0x4005_C400	00	00	00	01	-
SNFCECCMOD	0x4005_C404	00	00	00	01	-
SNFCIE	0x4005_C408	00	00	00	01	-
SNFCPS	0x4005_C410	00	00	11	00	-
SNFCPRCS	0x4005_C414	-				Read register
SNFCS	0x4005_C418	00	00	00	08	-
SNFCSS	0x4005_C41C	-				Read register
SNFCDIC	0x4005_C420	00	00	00	00	Undefined
SNFCDOC	0x4005_C424	00	00	00	00	Undefined
SNFCEIC	0x4005_C428	00	00	02	13	-
SNFCA1	0x4005_C430	00	00	00	00	-
SNFCA2	0x4005_C434	00	00	00	01	-
SNFCW	0x4005_C438	00	00	00	00	-
SNFCBIC	0x4005_C43C	**	**	**	**	Set according to the spec of memory
SNFCCS1	0x4005_C440	B8	10	FC	80	-
SNFCCS2	0x4005_C444	-				Undefined
SNFCCS3	0x4005_C448	-				Undefined
SNFCCS4	0x4005_C44C	-				Undefined
SNFCCSE	0x4005_C450	00	00	00	01	-
SNFCPRDB	0x4005_C500	-				Read register
SNFCIR1	0x4005_C504	-				Read register
SNFCIR2	0x4005_C508	-				Read register
SNFCEP1	0x4005_C510	-				Read register
SNFCEP2	0x4005_C514	-				Read register
SNFCEP3	0x4005_C518	-				Read register
SNFCEP4	0x4005_C51C	-				Read register
SNFCEC	0x4005_C520	-				Read register
SNFCEWRB	0x4005_C800	-				Write data input (from RAM1)
SNFCCDRB	0x4005_CC00	-				Read register
SNFCEBS	0x4005_CC08	-				Read register
SNFCEES	0x4005_CC20	-				Read register

Register name	Address	Setting code	Note
SNFCEDS1 to 8	0x4005_CC40 to 5C	-	Read register
SNFCS1EE1 to 4PI	0x4005_CC80 to 8C	-	Read register
SNFCS2EE1 to 4PI	0x4005_CC90 to 9C	-	Read register
SNFCS3EE1 to 4PI	0x4005_CCA0 to AC	-	Read register
SNFCS4EE1 to 4PI	0x4005_CCB0 to BC	-	Read register
SNFCS5EE1 to 4PI	0x4005_CCC0 to CC	-	Read register
SNFCS6EE1 to 4PI	0x4005_CCD0 to DC	-	Read register
SNFCS7EE1 to 4PI	0x4005_CCE0 to EC	-	Read register
SNFCS8EE1 to 4PI	0x4005_CCF0 to FC	-	Read register

11.8.4 Setting Conditions at Page Reading (Decoding)

This section shows the example in the case where data is transferred to RAM3 from the NAND flash memory at page read (decoding).

Data is written to RAM3 by reading data from the NAND flash memory.

11.8.4.1 Setting Conditions

Target memory		Capacity	: 4 GB
		Page size	: 4096+256 bytes
		1 sector	: 512+32 bytes
		The number of sectors	: 8
		ECC	: BCH8 mode
		Read address page	: 0x1_0000
DMA to be used	Unit A : DMAA I/F control block ↓ RAM 1	Channels to be used	: 0 to 1 channels (1 page)
		Channel control data start address	: 0x2003_F400 (DMAACtrlBasePtr=0x2003_F400)
		Operation mode	: Basic mode (Control<cycle_ctrl>="0x001")
		Source (Fixed)	: SNFCPRDB register of the SNFC (Address: 0x4005_C500) (Source End Pointer=0x4005_C500)
		Destination	: RAM1 * Start address: 0x2004_0000 (Destination End Pointer=0x2004_087C) * Set the last address for each channel.
		The number of transfers	: 133 times per channel (Control<n_minus_1>="10_0001_1111")
		Transfer size	: Source:4 bytes/Destination: 4 bytes (Control<src_size>="10", control data <dst_size>="10")
DMA to be used	Unit B : DMAB RAM 1 ↓ ECC control block	Channels to be used	: 8 to 15 channels (8 sectors) *Odds page setting
		Channel control data start address (for primary data)	: 0x2003_F880 (DMABCtrlBasePtr=0x2003_F880)
		Channel control data start address (for alternative data)	: 0x2003_FA80 (DMABAltCtlBasePtr=0x2003_FA80)
		Operation mode	: Ping-pong mode (Control<cycle_ctrl>="0x011")
		Source	: RAM1 * Start address: 0x2004_0000 (Source End Pointer=0x2004_0210) * Set the last address for each channel.
		Destination (Fixed)	: SNFCEWRB register of the SNFC (Address: 0x4005_C800) (Destination End Pointer=0x4005_C800)
		The number of transfers	: 136 times per channel (Control<n_minus_1>="00_1000_0111")
		Transfer size	: Source:4byte /Destination: 4 bytes (Control<src_size>="10", Control<dst_size>="10")

DMA to be used	Unit C : DMAC ECC control block ↓ RAM 3	Channels to be used	: 0 to 7 channels (8 sectors)
		Channel control data start address	: 0x2003_FC00 (DMACCtrlBasePtr=0x2003_FC00)
		Operation mode	: Basic mode (Control<cycle_ctrl>="0x001")
		Source (Fixed)	: SNFCCDRB register of the SNFC (Address: 0x4005_CC00) (Source End Pointer=0x2005_CC00)
		Destination	: RAM3 * Start address: 2004_8000 (Destination End Pointer=0x2004_821C) * Set the last address for each channel.
		The number of transfers	: 136 times per channel (Control<n_minus_1>="00_1000_0111")
		Transfer size	: Source:4byte /Destination: 4 bytes (Control<src_size>="10", Control<dst_size>="10")
SNFC		Execution mode	: Encoding (SNFCCSE<DECMODE>="0")
		BCH mode	: BCH8 (SNFCECCMOD<SELBCH>="1")
		Page/column address	: 0x1_0000_0000 (SNFCA1=0x0000_0000, SNFCA2=0x0000_0001)
		Decode input	: 544 bytes * Data (512 bytes)+management information (19 bytes) +Parity (13 bytes) (SNFCDIC=0x0000_0220)
		Decode output	: 544 bytes * Data (512 bytes)+management information (19 bytes) +Parity (13 bytes) (SNFCDOC=0x0000_0220)
		The number of sectors	: 8 sectors (SNFCS=0x0000_0008)
		Interrupt	: Uncorrectable error detection interrupt, page RAM transfer completion interrupt A, page RAM transfer completion interrupt C (SNFCIE=0x0009_8030)
		Automatic correction	: (SNFCECCMOD<GOUTMODE>="1")

11.8.4.2 Example of DMA (DMAA/DMAB/DMAC) Setting

(1) DMAA

Table 11-22 (1)-1 DMAA register

Register name	Address	Setting code				Note
DMAAStatus	0x4004_C000	-				Read register
DMAACfg	0x4004_C004	00	00	00	01	Operation setting
DMAACtrlBasePtr	0x4004_C008	20	03	F4	00	Primary data base pointer setting
DMAAAltCtlBasePtr	0x4004_C00C	-				Undefined
DMAAChnlSwRequest	0x4004_C014	00	00	00	00	Software request setting
DMAAChnlUseburstSet	0x4004_C018	-				Undefined
DMAAChnlUseburstClr	0x4004_C01C	-				Undefined
DMAAChnlReqMaskSet	0x4004_C020	FF	FF	FF	FC	Request mask setting
DMAAChnlReqMaskClr	0x4004_C024	-				Undefined
DMAAChnlEnableSet	0x4004_C028	00	00	00	03	Operation enable setting
DMAAChnlEnableClr	0x4004_C02C	-				Undefined
DMAAChnlPriAltSet	0x4004_C030	-				Undefined
DMAAChnlPriAltClr	0x4004_C034	-				Undefined
DMAAChnlPrioritySet	0x4004_C038	-				Undefined
DMAAChnlPriorityClr	0x4004_C03C	-				Undefined
DMAAErrClr	0x4004_C04C	-				Undefined
DMAIFFLGA	0x4005_F000	-				Read register

Table 11-23 (1)-2 Channel control data

Channel control data name	RAM	Setting code				Note
Reserved	0x2003_F40C	-				For ch0 of DMAA
Control	0x2003_F408	AE	00	21	F1	
Destination End Pointer	0x2003_F404	20	04	08	7C	
Source End Pointer	0x2003_F400	40	05	C5	00	
Reserved	0x2003_F41C	-				For ch1 of DMAA
Control	0x2003_F418	AE	00	21	F1	
Destination End Pointer	0x2003_F414	20	04	10	FC	
Source End Pointer	0x2003_F410	40	05	C5	00	

(2) DMAB

Table 11-24 (2)-1 DMAB register

Register name	Address	Setting code				Note
DMABStatus	0x4004_D000	-				Read register
DMABCfg	0x4004_D004	00	00	00	01	Operation setting
DMABCtrlBasePtr	0x4004_D008	20	03	F8	80	Primary data base pointer setting
DMABAltCtlBasePtr	0x4004_D00C	20	03	FA	80	Alternative database pointer setting
DMABChnlSwRequest	0x4004_D014	00	00	00	00	Software request setting
DMABChnlUseburstSet	0x4004_D018	-				Undefined
DMABChnlUseburstClr	0x4004_D01C	-				Undefined
DMABChnlReqMaskSet	0x4004_D020	FF	FF	00	FF	Request mask setting
DMABChnlReqMaskClr	0x4004_D024	-				Undefined
DMABChnlEnableSet	0x4004_D028	00	00	00	0F	Operation enable setting
DMABChnlEnableClr	0x4004_D02C	-				Undefined
DMABChnlPriAltSet	0x4004_D030	-				Undefined
DMABChnlPriAltClr	0x4004_D034	-				Undefined
DMABChnlPrioritySet	0x4004_D038	-				Undefined
DMABChnlPriorityClr	0x4004_D03C	-				Undefined
DMABErrClr	0x4004_D04C	-				Undefined
DMAIFFLGB	0x4005_F004	-				Read register

(2)-2 Channel control data

Table 11-25 ・Primary data

Channel control data name	RAM	Setting code				Note
Reserved	0x2003_F88C	-				ch8 of DMAB (For primary data)
Control	0x2003_F888	EA	00	08	73	
Destination End Pointer	0x2003_F884	40	05	C8	00	
Source End Pointer	0x2003_F880	20	04	02	1C	
Reserved	0x2003_F89C	-				ch9 of DMAB (For primary data)
Control	0x2003_F898	EA	00	08	73	
Destination End Pointer	0x2003_F894	40	05	C8	00	
Source End Pointer	0x2003_F890	20	04	04	3C	
Reserved	0x2003_F8AC	-				ch10 of DMAB (For primary data)
Control	0x2003_F8A8	EA	00	08	73	
Destination End Pointer	0x2003_F8A4	40	05	C8	00	
Source End Pointer	0x2003_F8A0	20	04	06	5C	
Reserved	0x2003_F8BC	-				ch11 of DMAB (For primary data)
Control	0x2003_F8B8	EA	00	08	73	
Destination End Pointer	0x2003_F8B4	40	05	C8	00	
Source End Pointer	0x2003_F8B0	20	04	08	7C	
Reserved	0x2003_F8CC	-				ch12 of DMAB (For primary data)
Control	0x2003_F8C8	EA	00	08	73	
Destination End Pointer	0x2003_F8C4	40	05	C8	00	
Source End Pointer	0x2003_F8C0	20	04	0A	9C	
Reserved	0x2003_F8DC	-				ch13 of DMAB (For primary data)
Control	0x2003_F8D8	EA	00	08	73	
Destination End Pointer	0x2003_F8D4	40	05	C8	00	
Source End Pointer	0x2003_F8D0	20	04	0C	BC	
Reserved	0x2003_F8EC	-				ch14 of DMAB (For primary data)
Control	0x2003_F8E8	EA	00	08	73	
Destination End Pointer	0x2003_F8E4	40	05	C8	00	
Source End Pointer	0x2003_F8E0	20	04	0E	DC	
Reserved	0x2003_F8FC	-				ch15 of DMAB (For primary data)
Control	0x2003_F8F8	EA	00	08	73	
Destination End Pointer	0x2003_F8F4	40	05	C8	00	
Source End Pointer	0x2003_F8F0	20	04	10	FC	

Table 11-26 ・Alternative data

Channel control data name	RAM	Setting code				Note
Reserved	0x2003_FA8C	-				ch8 of DMAB (for alternative data)
Control	0x2003_FA88	EA	00	08	73	
Destination End Pointer	0x2003_FA84	40	05	C8	00	
Source End Pointer	0x2003_FA80	20	04	02	1C	
Reserved	0x2003_FA9C	-				ch9 of DMAB (for alternative data)
Control	0x2003_FA98	EA	00	08	73	
Destination End Pointer	0x2003_FA94	40	05	C8	00	
Source End Pointer	0x2003_FA90	20	04	04	3C	

Table 11-26 ・Alternative data

Channel control data name	RAM	Setting code				Note
Reserved	0x2003_FAAC	-				ch10 of DMAB (for alternative data)
Control	0x2003_FAA8	EA	00	08	73	
Destination End Pointer	0x2003_FAA4	40	05	C8	00	
Source End Pointer	0x2003_FAA0	20	04	06	5C	
Reserved	0x2003_FABC	-				ch11 of DMAB (for alternative data)
Control	0x2003_FAB8	EA	00	08	73	
Destination End Pointer	0x2003_FAB4	40	05	C8	00	
Source End Pointer	0x2003_FAB0	20	04	08	7C	
Reserved	0x2003_FACC	-				ch12 of DMAB (for alternative data)
Control	0x2003_FAC8	EA	00	08	73	
Destination End Pointer	0x2003_FAC4	40	05	C8	00	
Source End Pointer	0x2003_FAC0	20	04	0A	9C	
Reserved	0x2003_FADC	-				ch13 of DMAB (for alternative data)
Control	0x2003_FAD8	EA	00	08	73	
Destination End Pointer	0x2003_FAD4	40	05	C8	00	
Source End Pointer	0x2003_FAD0	20	04	0C	BC	
Reserved	0x2003_FAEC	-				ch14 of DMAB (for alternative data)
Control	0x2003_FAE8	EA	00	08	73	
Destination End Pointer	0x2003_FAE4	40	05	C8	00	
Source End Pointer	0x2003_FAE0	20	04	0E	DC	
Reserved	0x2003_FAFC	-				ch15 of DMAB (for alternative data)
Control	0x2003_FAF8	EA	00	08	73	
Destination End Pointer	0x2003_FAF4	40	05	C8	00	
Source End Pointer	0x2003_FAF0	20	04	10	FC	

(3) DMAC

Table 11-27 (3)-1 DMAC register

Register name	Address	Setting code				Note
DMACStatus	0x4004_E000	-				Read register
DMACCfg	0x4004_E004	00	00	00	01	Operation setting
DMACCtrlBasePtr	0x4004_E008	20	03	FC	00	Primary data base pointer setting
DMACAltCtlBasePtr	0x4004_E00C	-				Undefined
DMACChnlSwRequest	0x4004_E014	00	00	00	00	Software request setting
DMACChnlUseburstSet	0x4004_E018	-				Undefined
DMACChnlUseburstClr	0x4004_E01C	-				Undefined
DMACChnlReqMaskSet	0x4004_E020	FF	FF	FF	00	Request mask setting
DMACChnlReqMaskClr	0x4004_E024	-				Undefined
DMACChnlEnableSet	0x4004_E028	00	00	00	FF	Operation enable setting
DMACChnlEnableClr	0x4004_E02C	-				Undefined
DMACChnlPriAltSet	0x4004_E030	-				Undefined
DMACChnlPriAltClr	0x4004_E034	-				Undefined
DMACChnlPrioritySet	0x4004_E038	-				Undefined
DMACChnlPriorityClr	0x4004_E03C	-				Undefined
DMACErrClr	0x4004_E04C	-				Undefined
DMAIFFLGC	0x4005_F008	-				Read register

Table 11-28 (3)-2 Channel control data

Channel control data name	RAM	Setting code				Note
Reserved	0x2003_FC0C	-				ch0 of DMAC (For primary data)
Control	0x2003_FC08	AE	00	08	71	
Destination End Pointer	0x2003_FC04	20	04	82	1C	
Source End Pointer	0x2003_FC00	40	05	CC	00	
Reserved	0x2003_FC1C	-				ch1 of DMAC (For primary data)
Control	0x2003_FC18	AE	00	08	71	
Destination End Pointer	0x2003_FC14	20	04	84	3C	
Source End Pointer	0x2003_FC10	40	05	CC	00	
Reserved	0x2003_FC2C	-				ch2 of DMAC (For primary data)
Control	0x2003_FC28	AE	00	08	71	
Destination End Pointer	0x2003_FC24	20	04	86	5C	
Source End Pointer	0x2003_FC20	40	05	CC	00	
Reserved	0x2003_FC3C	-				ch3 of DMAC (For primary data)
Control	0x2003_FC38	AE	00	08	71	
Destination End Pointer	0x2003_FC34	20	04	88	7C	
Source End Pointer	0x2003_FC30	40	05	CC	00	
Reserved	0x2003_FC4C	-				ch4 of DMAC (For primary data)
Control	0x2003_FC48	AE	00	08	71	
Destination End Pointer	0x2003_FC44	20	04	8A	9C	
Source End Pointer	0x2003_FC40	40	05	CC	00	
Reserved	0x2003_FC5C	-				ch5 of DMAC (For primary data)
Control	0x2003_FC58	AE	00	08	71	
Destination End Pointer	0x2003_FC54	20	04	8C	BC	
Source End Pointer	0x2003_FC50	40	05	CC	00	

Table 11-28 (3)-2 Channel control data

Channel control data name	RAM	Setting code				Note
Reserved	0x2003_FC6C	-				ch6 of DMAC (For primary data)
Control	0x2003_FC68	AE	00	08	71	
Destination End Pointer	0x2003_FC64	20	04	8E	DC	
Source End Pointer	0x2003_FC60	40	05	CC	00	
Reserved	0x2003_FC7C	-				ch7 of DMAC (For primary data)
Control	0x2003_FC78	AE	00	08	71	
Destination End Pointer	0x2003_FC74	20	04	90	FC	
Source End Pointer	0x2003_FC70	40	05	CC	00	

11.8.4.3 Example of the SNFC Setting

Register name	Address	Setting code				Note
SNFCENC	0x4005_C400	00	00	00	01	-
SNFCECCMOD	0x4005_C404	00	00	00	01	-
SNFCIE	0x4005_C408	00	09	80	30	-
SNFCPS	0x4005_C410	00	00	11	00	-
SNFCPRCS	0x4005_C414	-				Read register
SNFCS	0x4005_C418	00	00	00	08	-
SNFCSS	0x4005_C41C	-				Read register
SNFCDIC	0x4005_C420	00	00	02	20	-
SNFCDOC	0x4005_C424	00	00	02	20	-
SNFCEIC	0x4005_C428	-				Undefined
SNFCA1	0x4005_C430	00	00	00	00	-
SNFCA2	0x4005_C434	00	00	00	01	-
SNFCW	0x4005_C438	00	00	00	00	-
SNFCBIC	0x4005_C43C	**	**	**	**	Set according to the spec of memory
SNFCCS1	0x4005_C440	B8	30	F0	00	-
SNFCCS2	0x4005_C444	-				Undefined
SNFCCS3	0x4005_C448	-				Undefined
SNFCCS4	0x4005_C44C	-				Undefined
SNFCCSE	0x4005_C450	00	00	00	81	-
SNFCPRDB	0x4005_C500	-				Undefined
SNFCIR1	0x4005_C504	-				Read register
SNFCIR2	0x4005_C508	-				Read register
SNFCEP1	0x4005_C510	-				Read register
SNFCEP2	0x4005_C514	-				Read register
SNFCEP3	0x4005_C518	-				Read register
SNFCEP4	0x4005_C51C	-				Read register
SNFCEC	0x4005_C520	-				Read register
SNFCEWRB	0x4005_C800	-				Undefined
SNFCCDRB	0x4005_CC00	-				Undefined
SNFCEBS	0x4005_CC08	-				Read register
SNFCEES	0x4005_CC20	-				Read register
SNFCEDS1 ot 8	0x4005_CC40 to 5C	-				Read register
SNFCS1EE1 to 4PI	0x4005_CC80 to 8C	-				Read register
SNFCS2EE1 to 4PI	0x4005_CC90 to 9C	-				Read register
SNFCS3EE1 to 4PI	0x4005_CCA0 to AC	-				Read register

Register name	Address	Setting code	Note
SNFCS4EE1 to 4PI	0x4005_CCB0 to BC	-	Read register
SNFCS5EE1 to 4PI	0x4005_CCC0 to CC	-	Read register
SNFCS6EE1 to 4PI	0x4005_CCD0 to DC	-	Read register
SNFCS7EE1 to 4PI	0x4005_CCE0 to EC	-	Read register
SNFCS8EE1 to 4PI	0x4005_CCF0 to FC	-	Read register

11.9 Operation Timing

11.9.1 Basic Cycle

The cycle of the SNFC consists of the combination of the following cycles:

- Command cycle
- Address cycle
- Read cycle
- Write cycle
- Busy cycle
- Wait cycle

In addition, each cycle has the waveform adjustment function.

11.9.1.1 Command Cycle

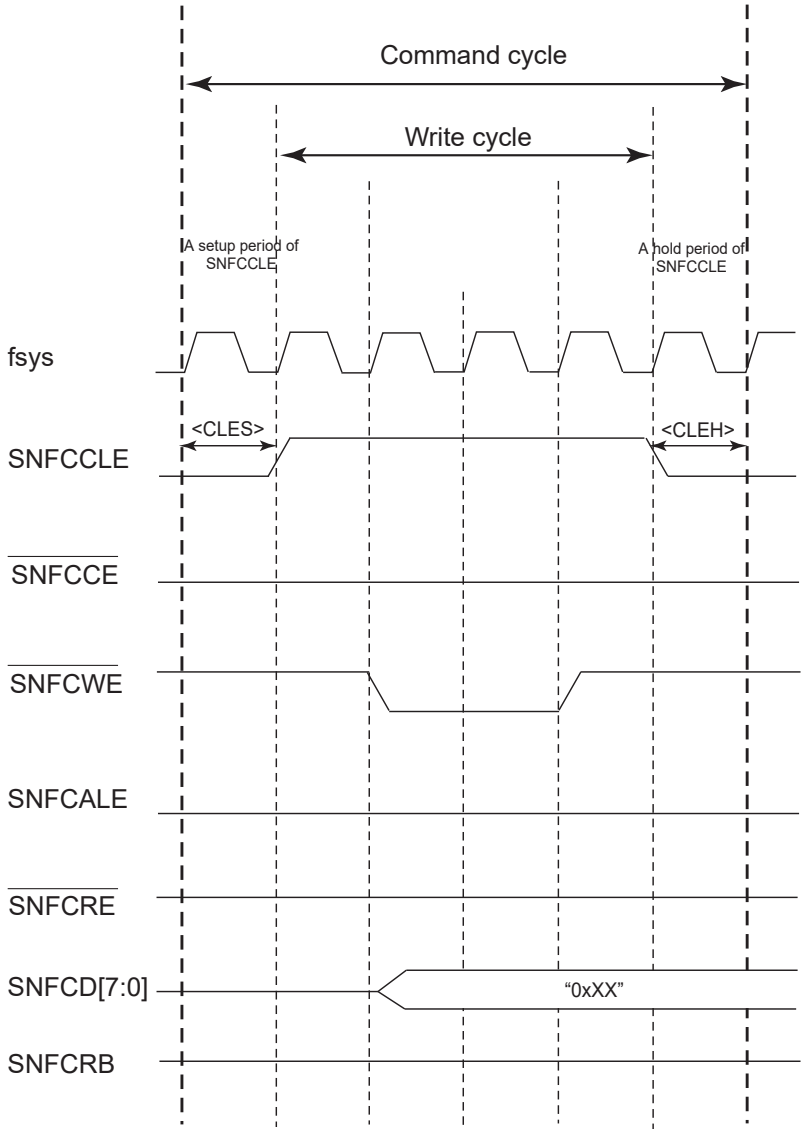
A command cycle is the cycle in which the SNFC issues commands to NAND flash memories.

Waveforms in a setup time or hold time of SNFCCLE can be adjusted with SNFCBIC<CLES>/<CLEH>.

The command cycle includes write cycles.

- Adjustment range
  - Setup time (<CLES[1:0]>): 0 to 3 clocks
  - Hold time (<CLEH[1:0]>): 0 to 3 clocks

The figure below shows waveforms of the case of <CLES>=1 and <CLEH>=1.



11.9.1.2 Address Cycle

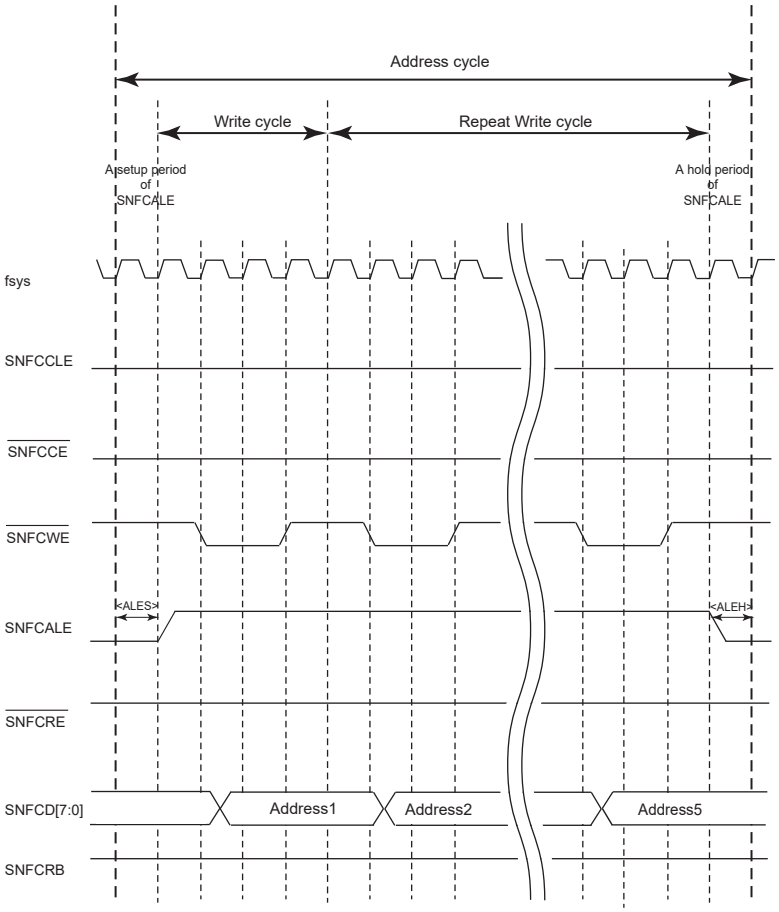
An address cycle is the cycle in which the SNFC outputs an address to the NAND flash memories.

Waveforms in a setup time or hold time of SNFCALE can be adjusted with SNFCBIC<ALES>/<ALEH>.

The address cycle includes write cycles.

- Adjustment range
  - Setup time (<ALES[1:0]>): 0 to 3 clocks
  - Hold time (<ALEH[1:0]>): 0 to 3 clocks

The figure below shows waveforms of the case of <ALES>=1 and <ALEH>=1.



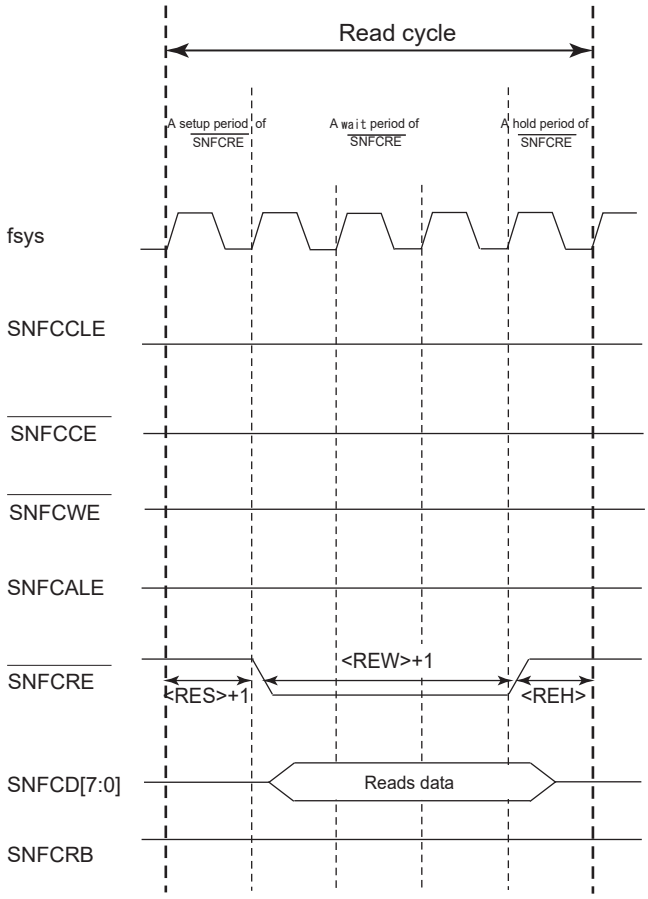
11.9.1.3 Read Cycle

A data cycle is the cycle in which the SNFC reads data from the NAND flash memories.

Waveforms in a setup time or hold time of $\overline{\text{SNFCRE}}$ can be adjusted with SNFCBIC<RES>/<REW>/<REH>.

- Adjustment range
  - Setup time (<RES[1:0]>+1): 1 to 4 clocks
  - Wait time (<REW[2:0]>+1): 1 to 8 clocks
  - Hold time (<REH[2:0]>): 0 to 7 clocks

The figure below shows waveforms of the case of <RES>=0, <REW>=2 and <REH>=1.



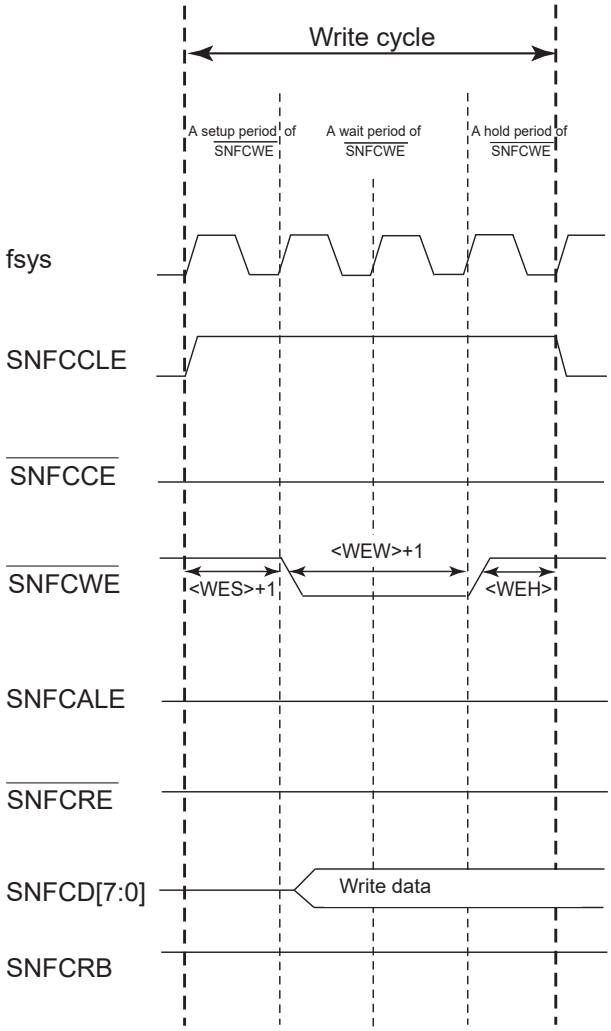
11.9.1.4 Write Cycle

A write cycle is the cycle in which the SNFC writes data to the NAND flash memories.

Waveforms in a setup time or hold time of $\overline{\text{SNFCWE}}$ can be adjusted with SNFCBIC<WES>/<WEW>/<WEH>.

- Adjustment range
  - Setup time (<WES[1:0]>+1): 1 to 4 clocks
  - Wait time (<WEW[2:0]>+1): 1 to 8 clocks
  - Hold time (<WEH[2:0]>): 0 to 7 clocks

The figure below shows waveforms of the case of <WES>=0, <WEW>=1 and <WEH>=1.



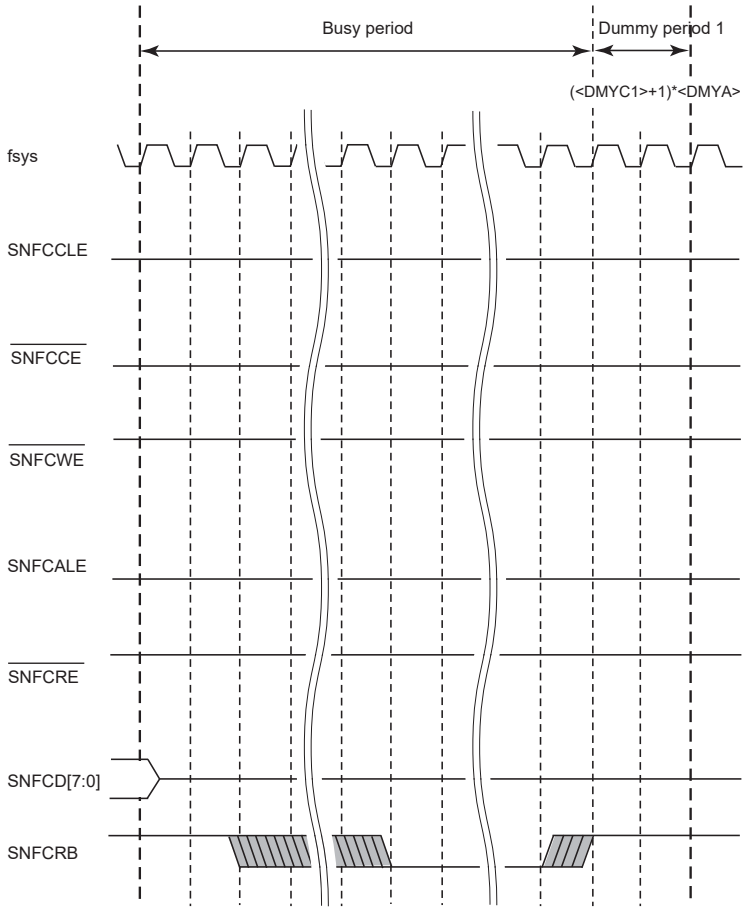
11.9.1.5 Busy Cycle

A busy cycle is the cycle in which the NAND flash memories reads/writes/erases/resets data.

A dummy time (time until the next cycle starts after ready state) can be adjusted with SNFCBIC<DMYC1> and SNFCCSx<DMYA>/<BSY>.

- Adjustment range (dummy period 1)
  - Dummy time (<DMYC1[1:0]>+1)*<DMYA>: 0 to 4 clocks

The figure below shows waveforms of the case of <DMYC1>=1 and <DMYA>=1.



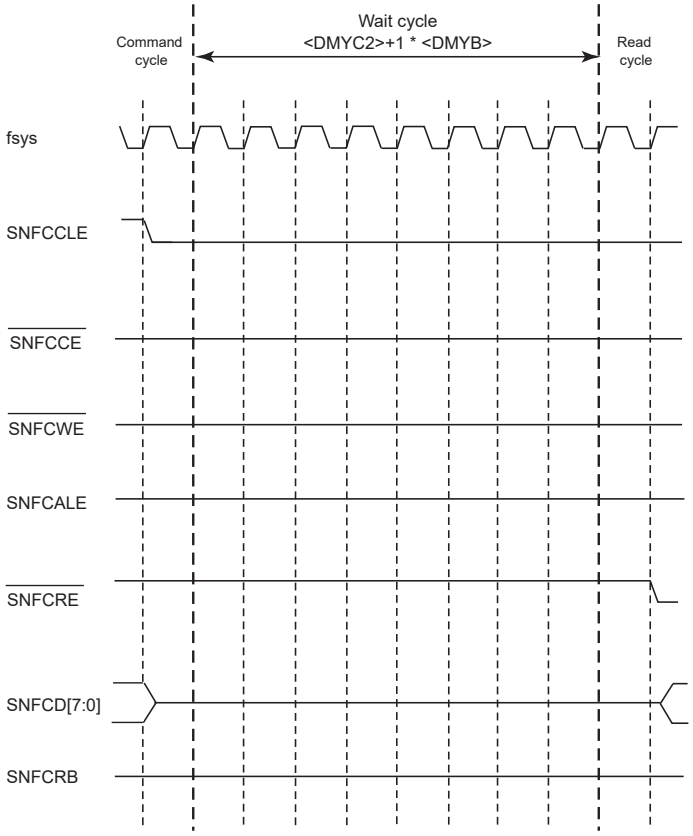
11.9.1.6 Wait Cycle

A wait cycle can be inserted between a command cycle (issuing commands) and a read cycle (reading the status).

A wait time can be adjusted with SNFCBIC<DMYC2> and SNFCCSx<DMYB>.

- Adjustment range (dummy period 2)
  - Wait time: (<DMYC2[2:0]>+1)*<DMYB>: 0 to 24 clocks

The figure below shows waveforms of the case of <DMYC2>=3 and <DMYB>=2.

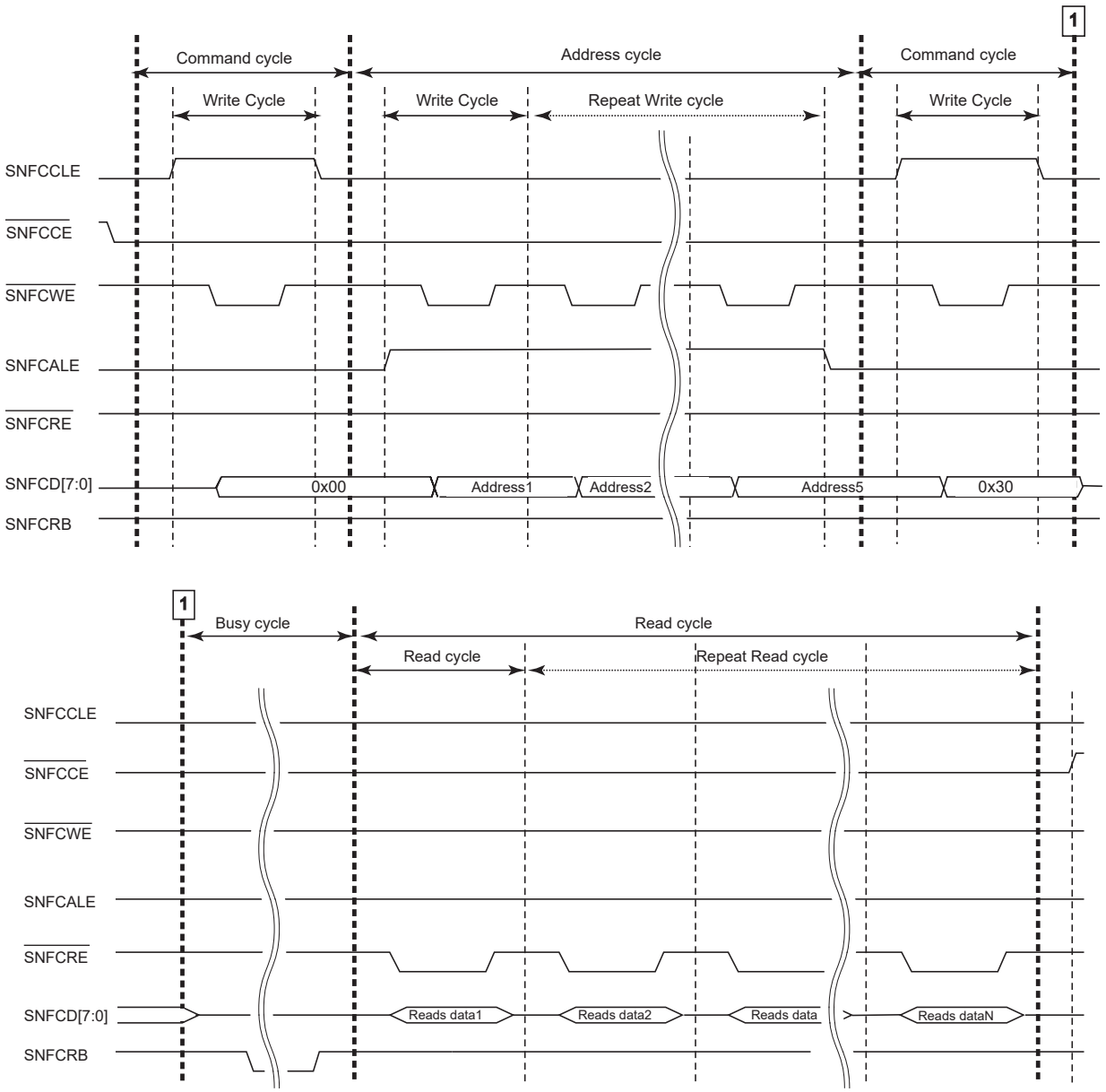


11.9.2 Bus Operation

This section shows the bus operation that is the combination of the basic cycles on the major command execution.

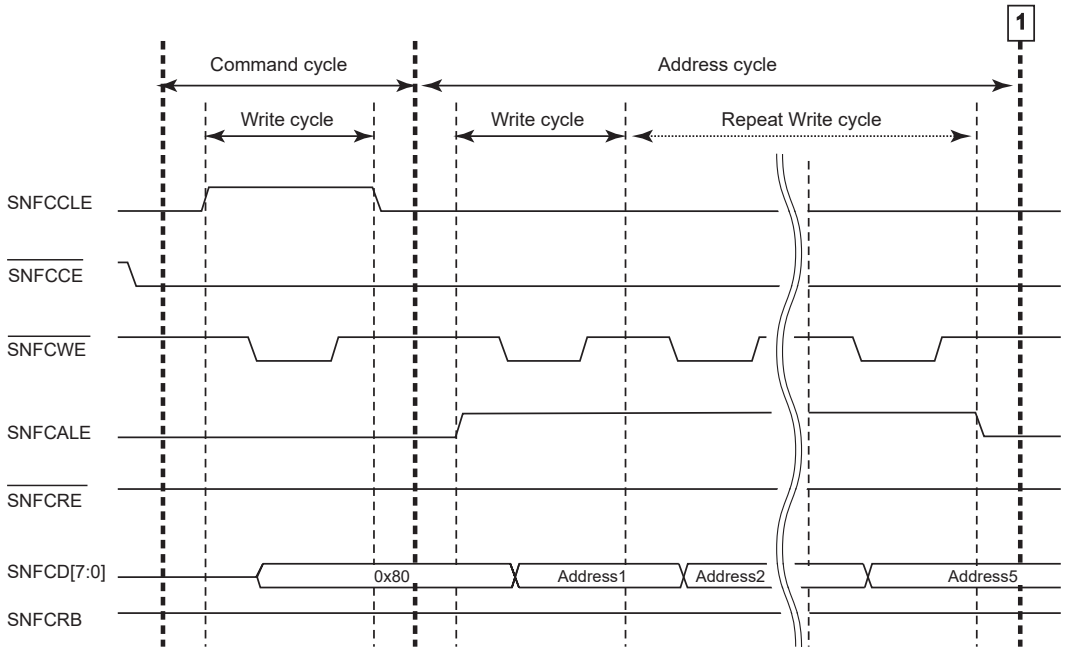
11.9.2.1 One-page Read

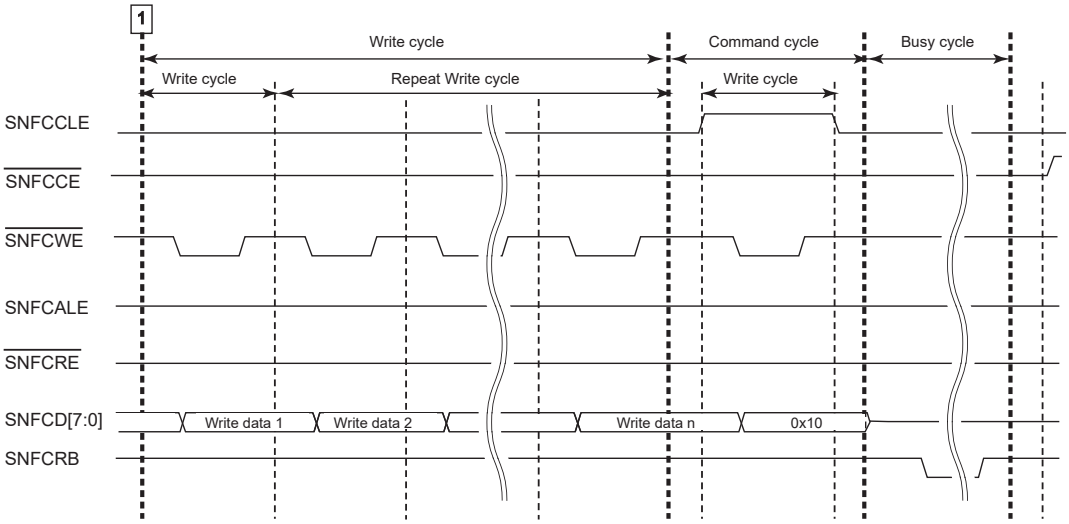
- Command: First Cycle=0x00 /Second Cycle=0x30
- Address: 5 cycles
- The number of data: n bytes (read)



11.9.2.2 One-page Write

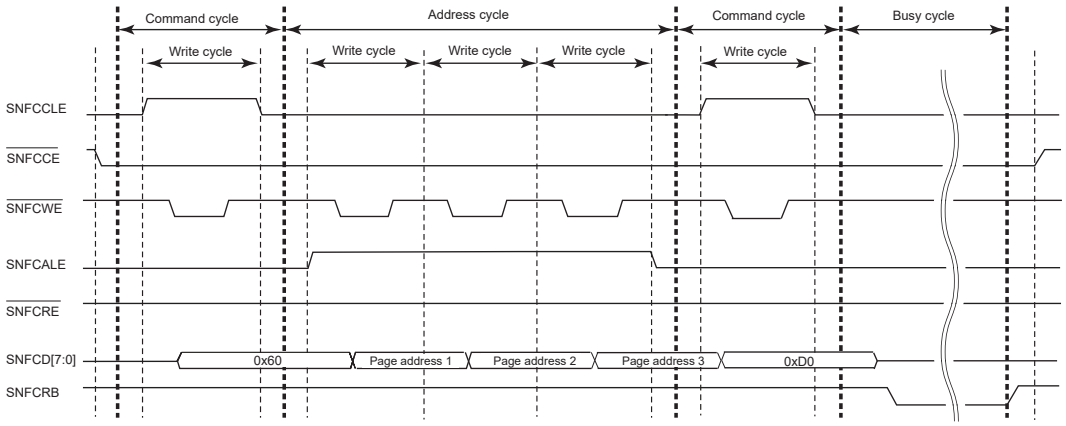
- Command: First Cycle= 0x80 / Second Cycle=0x10
- Address: 5 cycles
- The number of data: n bytes (write)





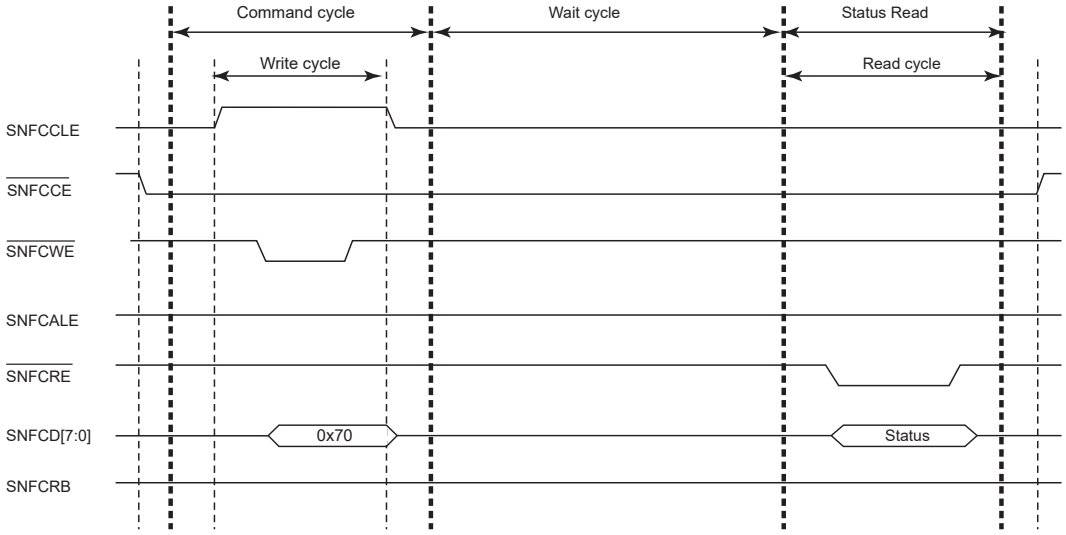
11.9.2.3 Block Erase

- Command: First Cycle=0x60 / Second Cycle= 0xD0



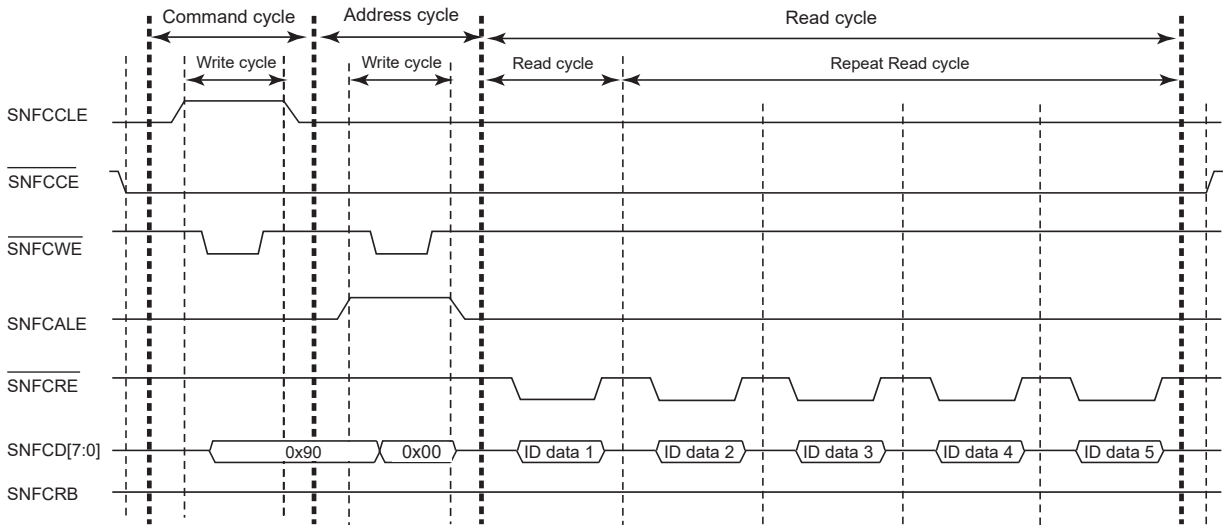
11.9.2.4 Status Read

- Command: First Cycle=0x70



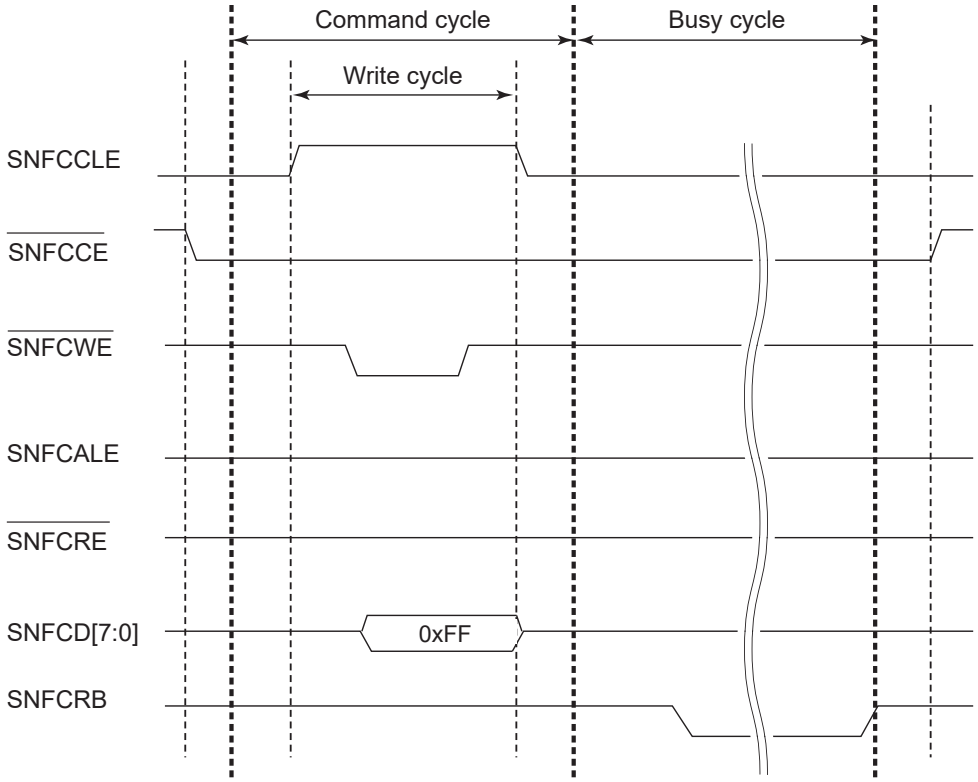
11.9.2.5 ID Read

- Command: First Cycle= 0x90
- Address: 0x00



11.9.2.6 Reset

- Command: First Cycle= 0xFF



11.10 Example of Memory Connection Diagram

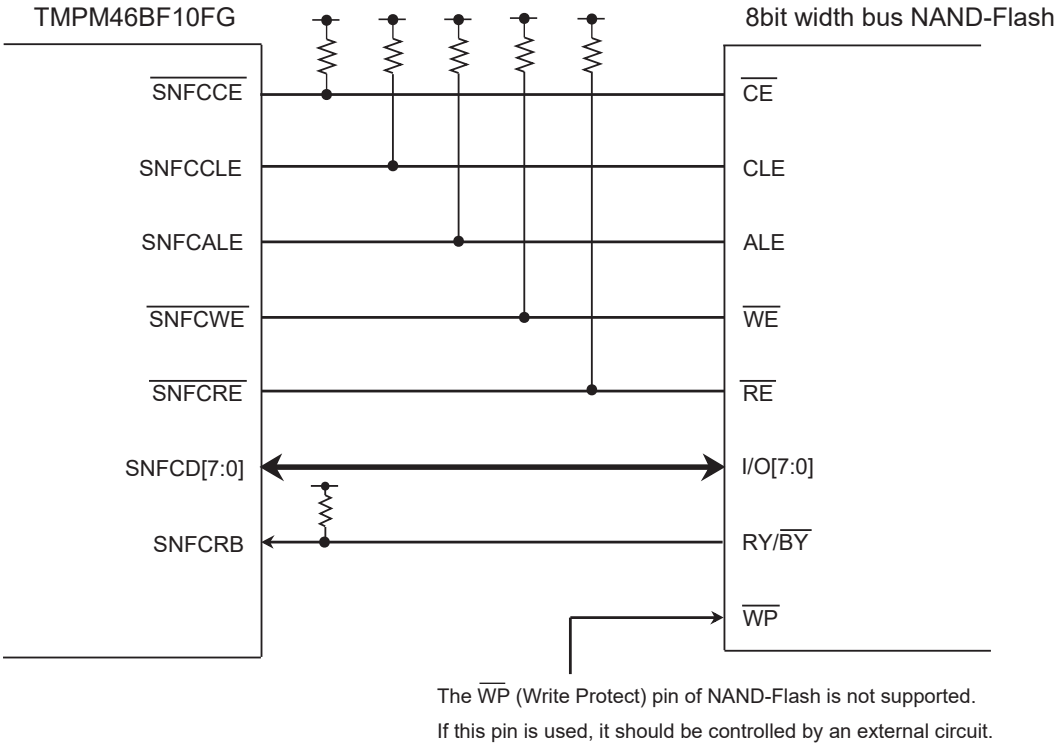


Figure 11-5 Connection Example

12. AES Processor (AES)

The AES processor (AES: Advanced Encryption Standard) encrypts/decrypts data in units of 128-bit block.

12.1 Outline

The AES processor has the following features:

- Supports 3 algorithms.

  ECB mode, CBC mode, and CTR mode
- Supports 3 key lengths

  128-bit length, 192-bit length, and 256-bit length
- Supports 2 transfer modes

  CPU transfer and DMA transfer
- Provides 4-word FIFOs

  Provides two 4-word FIFOs for input data and output data.

12.2 Configuration

The figure below is a block diagram of the AES.

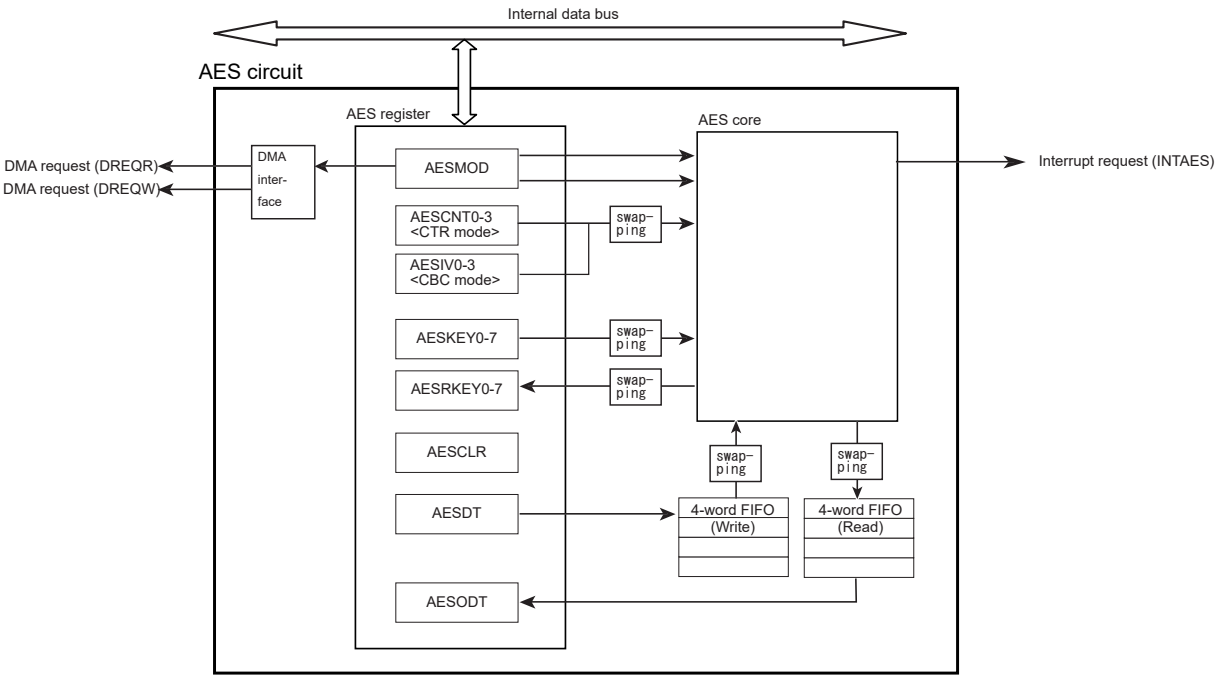


Figure 12-1 Block diagram of the AES

12.3 Registers

The following table lists the control registers and their addresses:

For the base address, refer to "A list of peripheral function base addresses" in the chapter on "Memory Map."

12.3.1 Register List

Peripheral function: AES

Register name		Address (Base+)
Plaintext/encrypted text data register	AESDT	0x0000
Input key data register (bit 31 - 0)	AESKEY7	0x0004
Input key data register (bit 63 - 32)	AESKEY6	0x0008
Input key data register (bit 95 - 64)	AESKEY5	0x000C
Input key data register (bit 127 - 96)	AESKEY4	0x0010
Input key data register (bit 159 - 128)	AESKEY3	0x0014
Input key data register (bit 191 - 160)	AESKEY2	0x0018
Input key data register (bit 223 - 192)	AESKEY1	0x001C
Input key data register (bit 255 - 224)	AESKEY0	0x0020
Counter initial value register (bit 31 - 0)	AESCNT3	0x0024
Counter initial value register (bit 63 - 32)	AESCNT2	0x0028
Counter initial value register (bit 95 - 64)	AESCNT1	0x002C
Counter initial value register (bit 127 - 96)	AESCNT0	0x0030
Initial vector register (bit 31 - 0)	AESIV3	0x0034
Initial vector register (bit 63 - 32)	AESIV2	0x0038
Initial vector register (bit 95 - 64)	AESIV1	0x003C
Initial vector register (bit 127 - 96)	AESIV0	0x0040
Arithmetic result store register	AESODT	0x0044
Output key store register (bit 31 - 0)	AESRKEY7	0x0048
Output key store register (bit 63 - 32)	AESRKEY6	0x004C
Output key store register (bit 95 - 64)	AESRKEY5	0x0050
Output key store register (bit 127 - 96)	AESRKEY4	0x0054
Output key store register (bit 159 - 128)	AESRKEY3	0x0058
Output key store register (bit 191 - 160)	AESRKEY2	0x005C
Output key store register (bit 223 - 192)	AESRKEY1	0x0060
Output key store register (bit 255 - 224)	AESRKEY0	0x0064
FIFO clear register	AESCLR	0x0068
Mode setting register	AESMOD	0x006C
Status register	AESSTATUS	0x0070

Note: Do not write any value to the above registers when AESSTATUS<BUSY>=1.

Peripheral function: SRST

Register name		Address (Base+)
Soft reset protect register	SRSTPROTECT	0x0000
Peripheral function soft reset register	SRSTIPRST	0x0004

12.3.2 AESDT (Plaintext/Encrypted Data Register)

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

Bit	Bit symbol	Type	Function
31 - 0	DT[31:0]	W	Plaintext/encrypted data Sets data for encryption or decryption. This register has a 4-word FIFO. The FIFO is required to write data four times per calculation. The FIFO is cleared when AESCLR<FIFOCLR>=1.

Note: Do not write any value to the register when AESSTATUS<BUSY>=1.

Write data is allocated from the lower side shown below and is input to the AES core after the data is swapped. For details, refer to the chapter on "12.5 Data Allocation".

127 96	95 64	63 32	31 0
AESDT (4th)	AESDT(3rd)	AESDT(2nd)	AESDT(1st.)

12.3.3 AESKEY0-7 (Input Key Data Register)

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

Bit	Bit symbol	Type	Function
31 - 0	KEY[31:0]	R/W	Key data Sets the key data for calculation.

Note: Do not write any value to the register when AESSTATUS<BUSY>=1.

The registers to be used vary depending on the key length specified with AESMOD<KEYLEN[1:0]>. Write data is input to the AES core after the data is swapped. For details, refer to the chapter on "12.5 Data Allocation".

bit	255 224	223 192	191 160	159 128	127 96	95 64	63 32	31 0
128-bit key length					AESKEY4	AESKEY5	AESKEY6	AESKEY7
192-bit key length			AESKEY2	AESKEY3	AESKEY4	AESKEY5	AESKEY6	AESKEY7
256-bit key length	AESKEY0	AESKEY1	AESKEY2	AESKEY3	AESKEY4	AESKEY5	AESKEY6	AESKEY7

12.3.4 AESCNT0 to 3 (Counter Initial Value Register)

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

Bit	Bit symbol	Type	Function
31 - 0	CNT[31:0]	R/W	The counter initial value Sets the counter initial value in CTR mode. When the counter overflows, it returns to the initial value and continues to count up. To avoid an overflow, for example, set "0x00" to one most significant byte.

Note: Do not write any value to the register when AESSTATUS<BUSY>=1.

Write data is allocated as shown below and is input to the AES core after the data is swapped. For details, refer to the chapter on "12.5 Data Allocation".

Bit	127 96	95 64	63 32	31 0
Register	AESCNT0	AESCNT1	AESCNT2	AESCNT3

12.3.5 AESIV0 to 3 (Initial Vector Register)

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

Bit	Bit symbol	Type	Function
31 - 0	IV[31:0]	R/W	Initial vector Sets the initial vector in CBC mode.

Note: Do not write any value to the register when AESSTATUS<BUSY>=1.

Write data is allocated as shown below and is input to the AES core after the data is swapped. For details, refer to the chapter on "12.5 Data Allocation".

Bit	127 96	95 64	63 32	31 0
Register	AESIV0	AESIV1	AESIV2	AESIV3

12.3.6 AESODT (Calculation Result Store Register)

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

Bit	Bit symbol	Type	Function
31 - 0	ODT[31:0]	R	Calculation result Stores the calculation result. This register has 4-word FIFO. The FIFO is required to write data four times per calculation. The FIFO is cleared when AESCLR<FIFOCLR>=1.

The arithmetic result is output from the AES core. The result is swapped and stored in the register as shown below. The data can be read from the lower side. For details, refer to the chapter on "12.5 Data Allocation".

127 96	95 64	63 32	31 0
AESODT (4th)	AESODT(3rd)	AESODT (2nd)	AESODT (1st)

12.3.7 AESRKEY7 to 0 (Output Key Store Register)

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

Bit	Bit symbol	Type	Function
31 - 0	RKEY[31:0]	R	Encryption key/decryption key data After calculation, the output key is stored.

The registers to be used vary depending on the key length specified with AESMOD<KEYLEN[1:0]>. Data is output from the AES core. The data is swapped and stored as shown below. For details, refer to the chapter on "12.5 Data Allocation".

Bit	255 224	223 192	191 160	159 128	127 96	95 64	63 32	31 0
128-bit key length					AESRKEY4	AESRKEY5	AESRKEY6	AESRKEY7
192-bit key length			AESRKEY2	AESRKEY3	AESRKEY4	AESRKEY5	AESRKEY6	AESRKEY7
256-bit key length	AESRKEY0	AESRKEY1	AESRKEY2	AESRKEY3	AESRKEY4	AESRKEY5	AESRKEY6	AESRKEY7

12.3.8 AESCLR (FIFO 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	–	–	–	–	–	–	–	FIFOCLR
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 1	–	R	Read as "0".
0	FIFOCLR	W	Clears the FIFO 1: Clears the FIFO When "1" is set to this bit, both the write FIFO and the read FIFO are cleared. Writing “0” has no meaning. Read as "0".

Note: Do not write any value to the register when AESSTATUS<BUSY>=1.

12.3.9 AESMOD (Mode 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	–	–	ALGO		KEYLEN		DMAEN	OP
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 6	–	R	Read as "0".
5 - 4	ALGO[1:0]	R/W	Algorithm setting 00: ECB mode 01: CBC mode 10: CTR mode 11: Reserved
3 - 2	KEYLEN[1:0]	R/W	Key length setting 00: 128-bit key length 01: 192-bit key length 10: 256-bit key length 11: Reserved
1	DMAEN	R/W	DMA transfer 0: Disabled 1: Enabled
0	OP	R/W	Operation setting 0: Encryption 1: Decryption In CTR mode, set "0" for both encryption and decryption.

Note: Do not write any value to the register when AESSTATUS<BUSY>=1.

12.3.10 AESSTATUS (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	–	–	–	–	–	RFIFOST	WFIFOST	BUSY
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 3	–	R	Read as "0".
2	RFIFOST	R	Reads FIFO status 0: No data 1: Data exists
1	WFIFOST	R	Writes FIFO status 0: No data 1: Data exists
0	BUSY	R	Arithmetic status 0: Calculation is complete. 1: Calculation is in process.

12.3.11 SRSTPROTECT (Soft Reset Protect 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	PROTECT							
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 8	–	R	Read as "0".
7 - 0	PROTECT	R/W	0x6B: Enabled Other than 0x6B: Disabled (The SRSTIPRST register cannot be accessed.) To write data to the SRSTIPRST register, set "0x6B" to this bit.

12.3.12 SRSTIPRST (Peripheral Function Soft 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	–	–	–	IPRST4	IPRST3	IPRST2	IPRST1	IPRST0
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 5	–	R	Read as "0".
4	IPRST4	R/W	SNFC reset 0: Clears the reset 1: Resets
3	IPRST3	R/W	ESG reset 0: Clears the reset 1: Resets
2	IPRST2	R/W	MLA reset 0: Clears the reset 1: Resets
1	IPRST1	R/W	SHA reset 0: Clears the reset 1: Resets
0	IPRST0	R/W	AES reset 0: Clears the reset 1: Resets

12.4 Algorithm

This MCU supports 3 encryption algorithms.

The AES processor encrypts/decrypts plaintext or encrypted text in units of 128-bit blocks. If data is long, encryption/decryption must be repeated in units of 128-bit blocks.

12.4.1 ECB (Electric Code Book) Mode

ECB mode is one of the encryption modes in which plaintext is encrypted as-is. The relationship between plaintext blocks and encrypted blocks is a one-to-one relationship.

In addition, if the combination of an input plaintext block and encryption key is not changed, the same encryption text is generated.

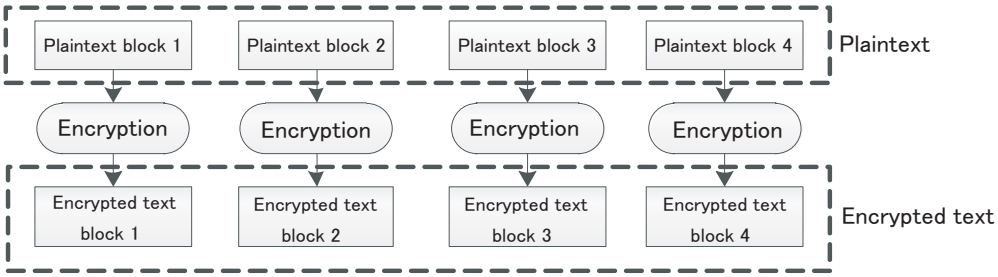


Figure 12-2 ECB mode (Encryption)

12.4.2 CBC (Cipher Block Chaining) Mode

In CBC mode, a plaintext block is encrypted after the plaintext block takes the exclusive logical OR with the previous encrypted text block. Therefore; even if plaintext block 2 is the same as block 1, encryption block 2 differs from encryption block 1.

When the first plaintext block is encrypted, there is no previous encrypted block. Thus, "the initial vector" is required on behalf of the previous encrypted block.

Generally, a different bit column is used as the initial vector every time encryption is performed.

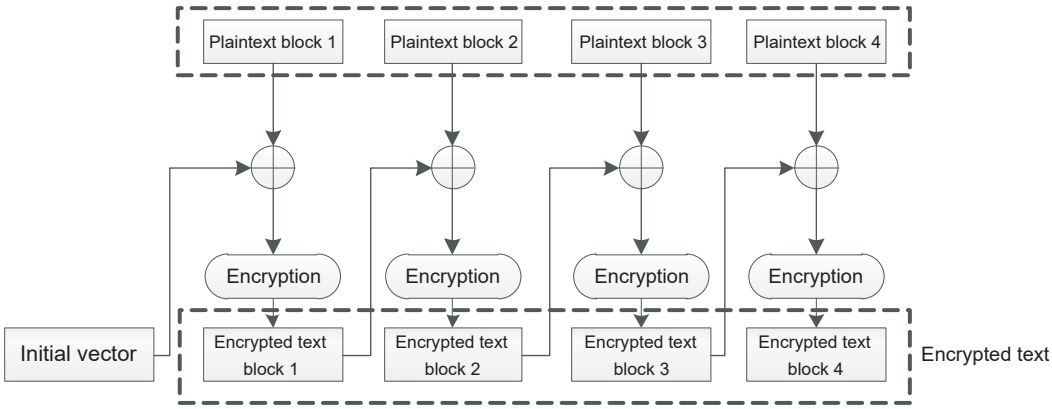


Figure 12-3 CBC mode (Encryption)

12.4.3 CTR (Counter) Mode

CTR mode is one of the encryption modes in which plaintext blocks are encrypted using the counter's values incrementing by 1. The bit column encrypted using the value of the counter and plaintext block takes the exclusive OR, and its result is the encrypted block.

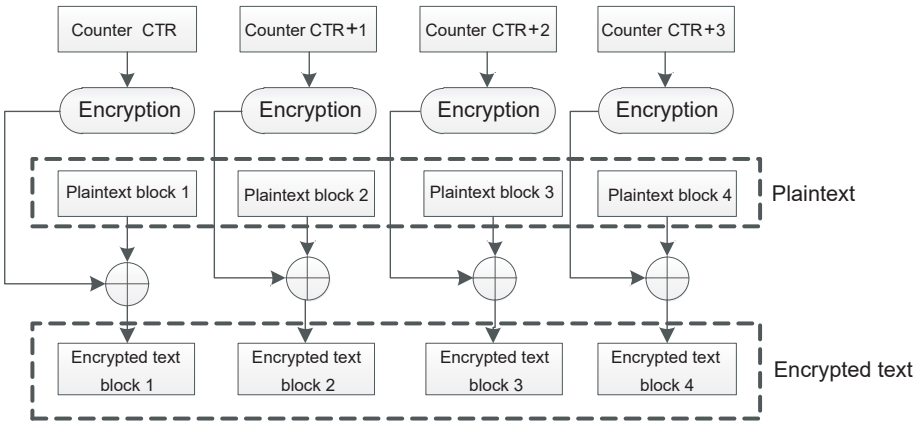


Figure 12-4 CTR mode (Encryption)

12.4.4 Input/Output Data in Each Algorithm

The table below lists the input data/output data in each algorithm.

Table 12-1 Encryption targets and their keys in each mode

Algorithm	Encryption/decryption	Input data			Output data	Decryption
ECB	Encryption	Plaintext	Encryption key	-	Encrypted text	
	Decryption	Encrypted text	Decryption key	-	Plaintext	
CBC	Encryption	Plaintext	Encryption key	Initial vector	Encrypted text	In CBC mode, input the same initial vector both on encryption and decrypting.
	Decryption	Encrypted text	Decryption key	Initial vector	Plaintext	
CTR	Encryption	Plaintext	Encryption key	Initial value of the counter	Encrypted text	In CTR mode, input the same counter value and the encryption key both on encryption and decrypting.
	Decryption	Encrypted text	Encryption key	Initial value of the counter	Plaintext	

12.5 Data Allocation

The data to be input/output to/from the AES core is swapped by the swap circuit.

12.5.1 Data Allocation of AESIV, AESCNT, AESKEY, and AESRKEY

The data stored in AESIV0 to 3 is swapped in units of byte as shown below when the data is input to the AES core. The data stored in AESKEY0 to 7 and AESCNT0 to 3 are swapped in the same manner.

The data in AESRKEY0 to 7 is output from the core. This data is also swapped and stored as shown below:

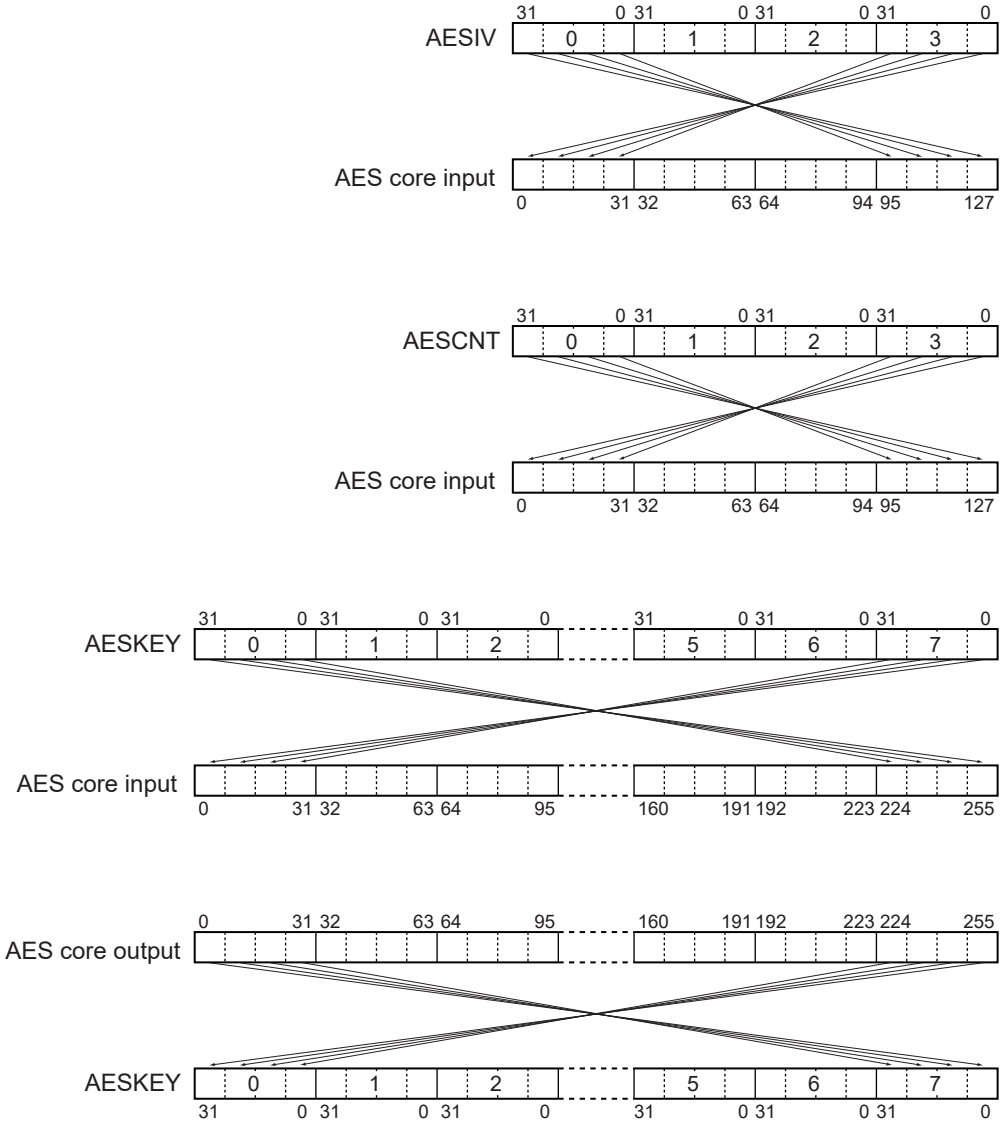


Figure 12-5 Data Allocation (AESIV, AESCNT, AESKEY)

12.5.2 FIFO Structure of AESODT and AESDT

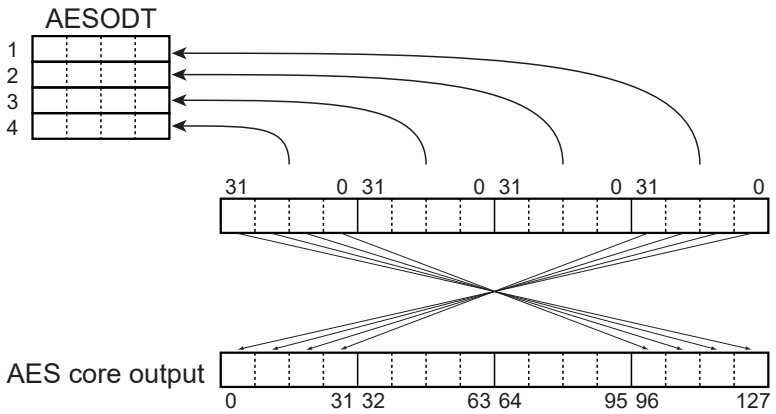
The AESODT register and AESDT register have a 4-word FIFO storing 1-block data.

12.5.2.1 Data Allocation

- AESODT (Read FIFO)

  The arithmetic result is stored and swapped as follows:

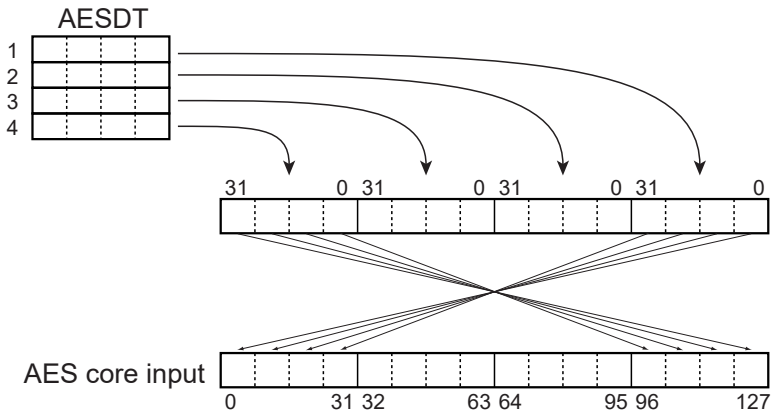
  This arithmetic result can be read from the lower side.



- AESDT (Write FIFO)

  Written data is allocated from the lower side and is swapped as follows:

  This data is input to the AES core.



12.5.2.2 Clearing the FIFOs

When "1" is set to AESCLR<FIFOCLR>, both the write FIFO and the read FIFO are cleared.

Do not clear the FIFOs when AESSTATUS<BUSY> is "1".

12.6 Data Transfer

Reading/writing data from/to the FIFO is performed by the DMA controller or CPU.

12.6.1 DMAC Transfer

Set the DMAC to transfer data to the write FIFO (AESDT) and set the DMAC to transfer data from the read FIFO (AESODT).

When "1" is set to AESMOD<DMAEN>, the AES outputs a transfer request (DREQW) to the write FIFO. The DMAC transfers 1-block data to the AESDT register. After data transfer, the AES core automatically starts calculation.

After calculation, the AES outputs a transfer request (DREQR) to the read FIFO. The DMA transfers data to the AESODT register.

12.6.2 CPU Transfer

When the CPU inputs 1-block data to the write FIFO, the AES core automatically starts calculation.

After calculation, the arithmetic result is stored in the read FIFO and AESRKEY register. To check whether calculation is complete, occurrence of interrupts (INTAES) or polling of AESSATUS<BUSY> are used.

After completion of calculation is confirmed, read the calculation result using AESODT<ODT[31:0]>.

12.7 Operation Procedure

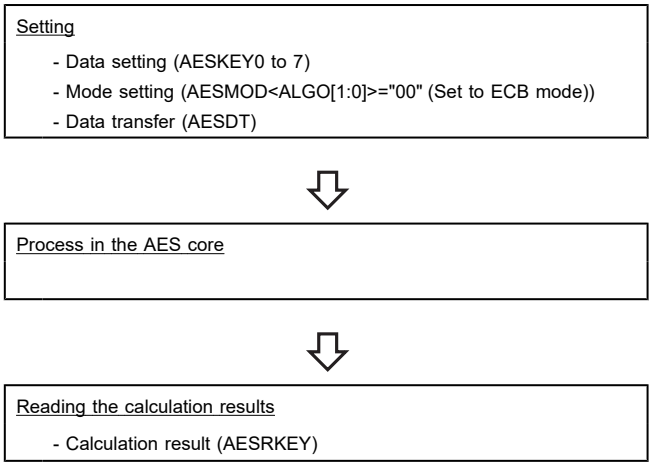
In encryption/decryption operation, when registers are set according to algorithms and key lengths, if 128-bit data is stored in the AESDT register, calculation automatically starts. After calculation, the calculation result is stored in the AESODT register.

To generate a composite key, set an encryption key and dummy data. The composite key is stored in AESRKEY.

The AESDT register and AESODT register have a 4-word FIFO each. Data transfer to/from the FIFO can be selected either from DMA transfer or CPU transfer.

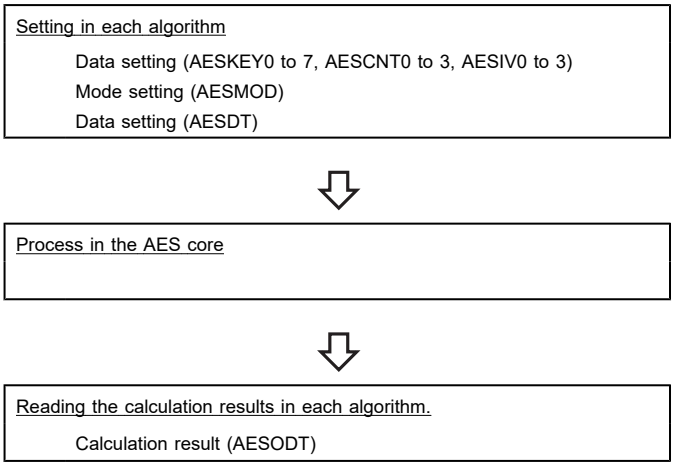
12.7.1 Composite Key Generation Procedure

Composite key generation basic procedure



12.7.2 Basic Procedure of Encryption/Decryption

Basic procedure of encryption/decryption



Note 1: In both DMA transfer and CPU transfer, settings of registers (AESDT, AESKEY0 to 7, AESCNT0 to 3, AESIV0 to 3, AESMOD, and AESCLR) cannot be changed after AESDT is set.

Note 2: FIFOCRL<CLR> clears both the write FIFO and the read FIFO. When AESSTATUS<BUSY>=1, do not set "1" to FIFOCLR<CLR>.

The number of cycles from the start of calculation by the AES to the output of the result varies depending on the key length specified with AESMOD<KEYLEN[1:0]>.

Key length	Setting value of <KEYLEN[1:0]>	Number of calculation cycles
128 bits	00	12 cycles
192 bits	01	14 cycles
256 bits	10	16 cycles

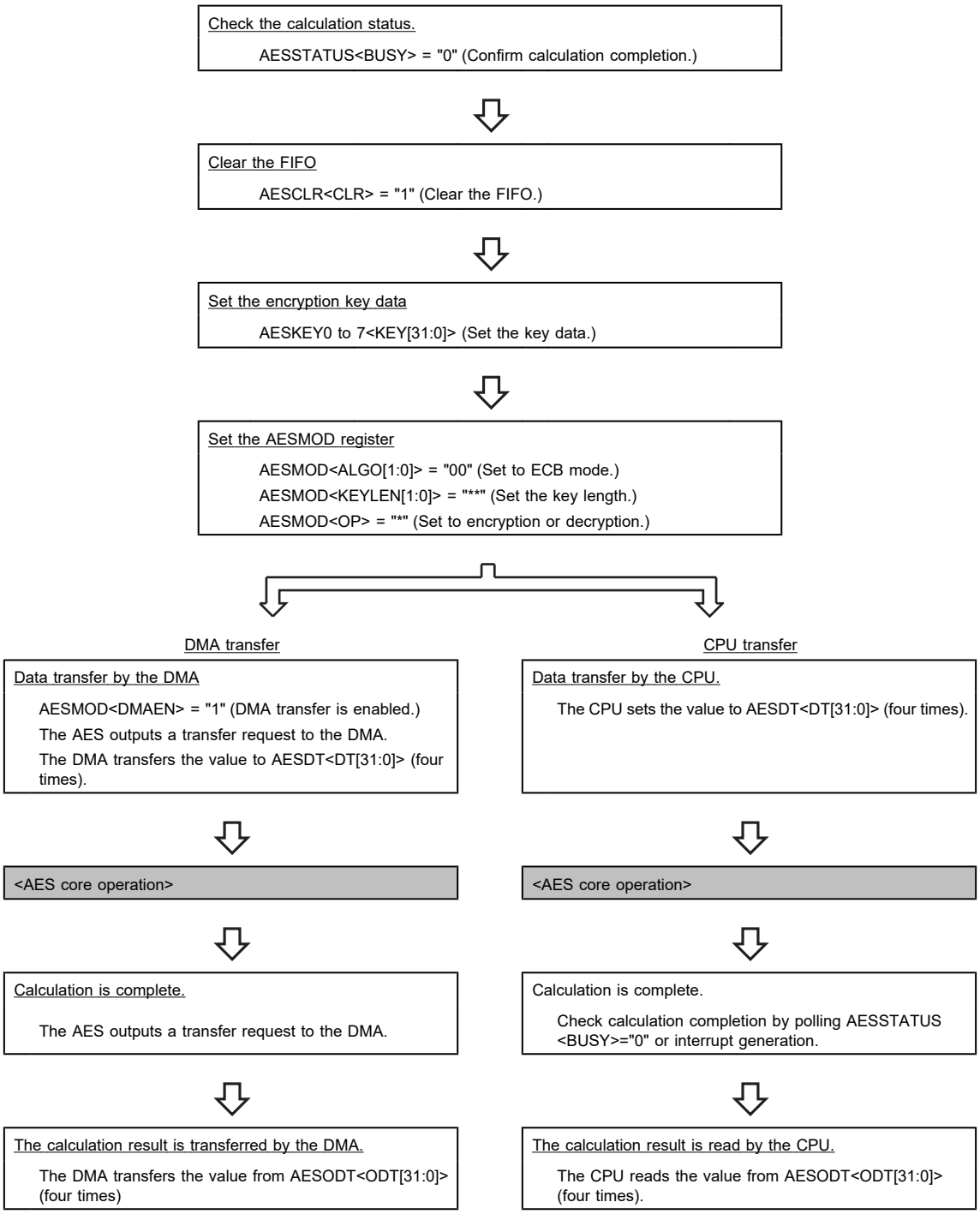
The following shows the details of the operation procedure in each algorithm.

12.7.3 Operation Procedure in Each Algorithm

When the AES is used, the operation procedure must be followed.

12.7.3.1 ECB Mode

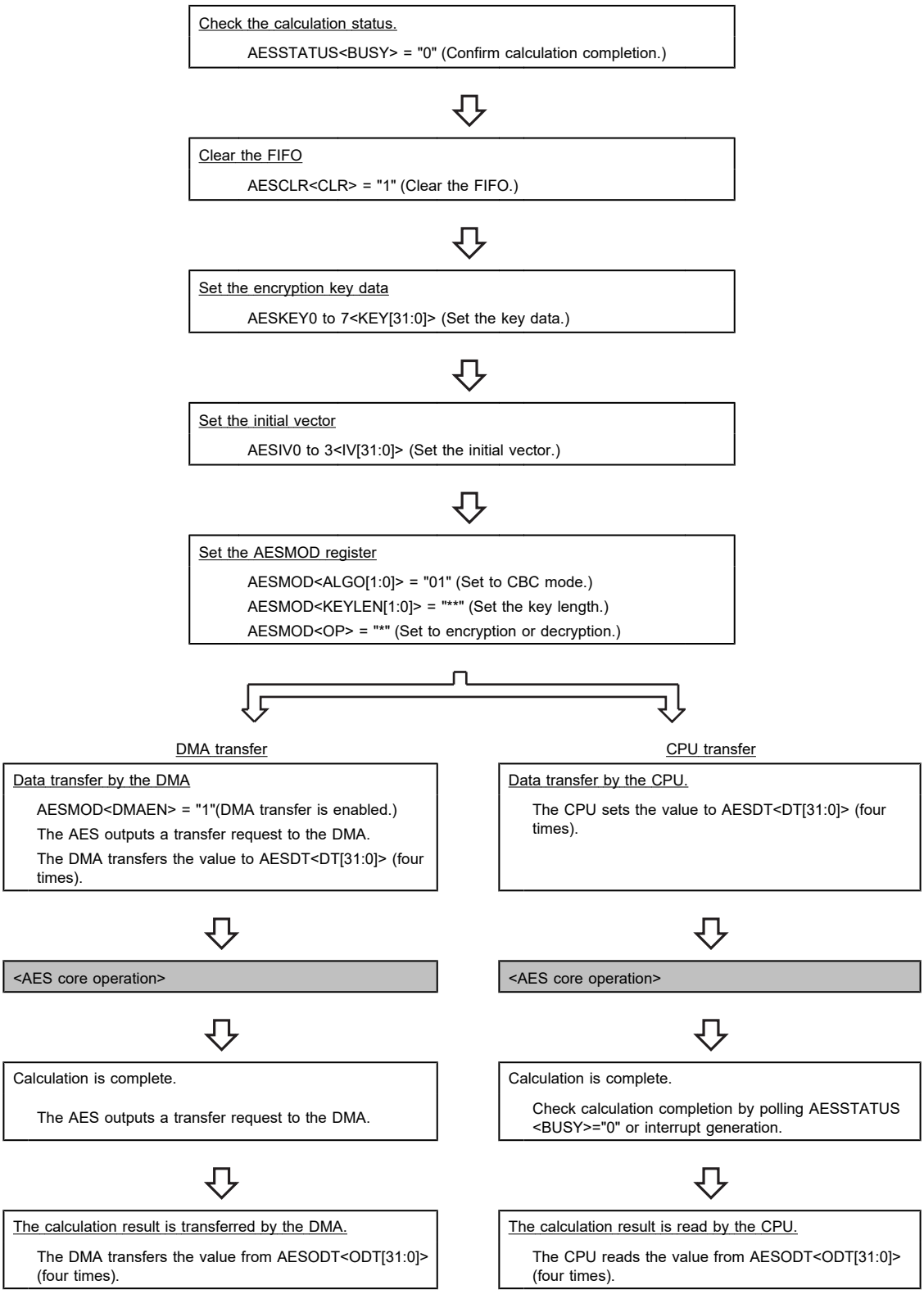
ECB mode setting procedure



12.7.3.2 CBC Mode

The initial vector data input in decryption is the same as the initial vector data used in encryption.

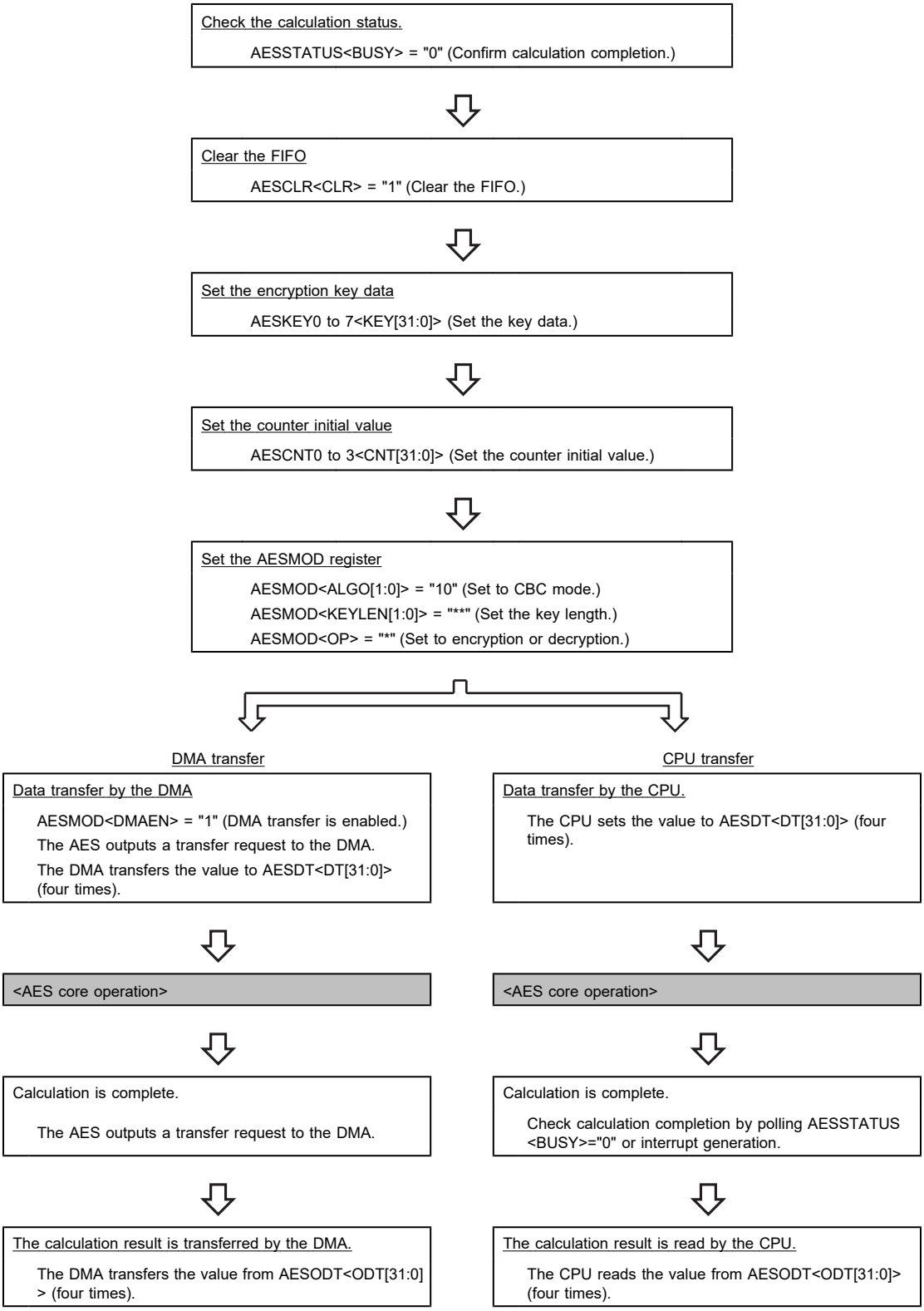
CBC mode setting procedure



12.7.3.3 CTR Mode

In CTR mode, the key data used in encryption is also used in decryption. Therefore, it is not necessary to read encrypted key data (AESRKEY0 to 7) in decryption. The initial counter value used in decryption is the same as the value used in encryption.

CTR mode setting procedure



12.8 Reset Operation

When "1" is set to SRSTIPRST<IPRST0>, all internal circuits and the registers in the AES are reset. When "0" is set to SRSTIPRST<IPRST0>, the reset state is released.

The SRSTIPRST register becomes write protection state with the SRSTPROTECT register after reset.

When a reset is performed, write "0x6B" to the SRSTPROTECT register to enable the register to be written (protection state is released).

13. Secure Hash Algorithm Processor (SHA)

The Secure Hash Algorithm processor (SHA) generates fixed length (256-bit) Hash values from message data.

13.1 Outline

The SHA has the following features:

- Conforms to FIPS PUB 180-3 Secure Hash standard Algorithm (SHA2)

  Supports SHA-224/SHA-256
- Message length

  Up to (2^{61} - 1) bytes. Calculations are performed in unit of 512 bits.
- Automatic padding
- Halting or restarting of calculation

  Thanks to stacking results of calculation in progress, calculation can be restarted.

13.2 Configuration

The figure below shows the block diagram of the SHA.

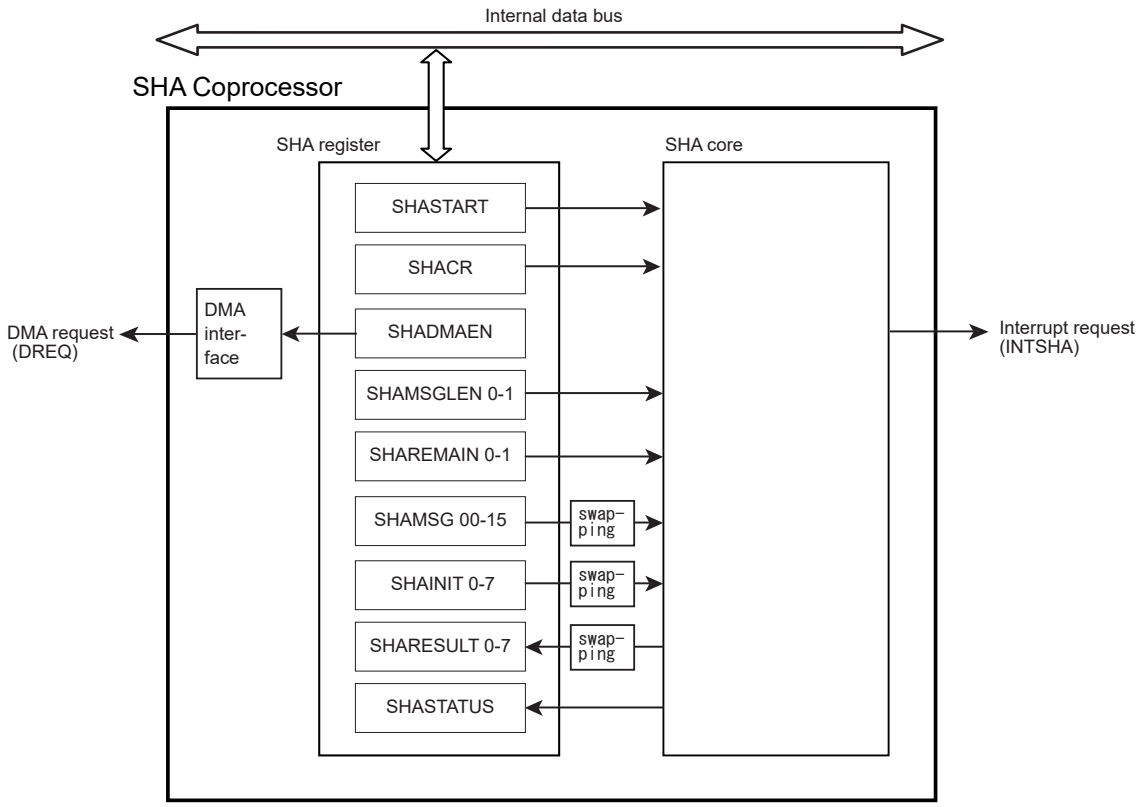


Figure 13-1 Block diagram of the SHA

13.3 Registers

The following table lists the control registers and their addresses:

For the base address, refer to "A list of peripheral function base addresses" in the chapter on "Memory Map."

13.3.1 Register List

Peripheral function: SHA

Register name		Address (Base+)
Process start register	SHASTART	0x0000
Control register	SHACR	0x0004
DMA enable register	SHADMAEN	0x0008
Whole message length register (bit 31 - 0)	SHAMSGLEN0	0x000C
Whole message length register (bit 60 - 32)	SHAMSGLEN1	0x0010
Unhandled message length register (bit 31 - 0)	SHAREMAIN0	0x0014
Unhandled message length register (bit 60 - 32)	SHAREMAIN1	0x0018
Message register (bit 31 - 0)	SHAMSG00	0x001C
Message register (bit 63 - 32)	SHAMSG01	0x0020
Message register (bit 95 - 64)	SHAMSG02	0x0024
Message register (bit 127 - 96)	SHAMSG03	0x0028
Message register (bit 159 - 128)	SHAMSG04	0x002C
Message register (bit 191 - 160)	SHAMSG05	0x0030
Message register (bit 223 - 192)	SHAMSG06	0x0034
Message register (bit 255 - 224)	SHAMSG07	0x0038
Message register (bit 287 - 256)	SHAMSG08	0x003C
Message register (bit 319 - 288)	SHAMSG09	0x0040
Message register (bit 351 - 320)	SHAMSG10	0x0044
Message register (bit 383 - 352)	SHAMSG11	0x0048
Message register (bit 415 - 384)	SHAMSG12	0x004C
Message register (bit 447 - 416)	SHAMSG13	0x0050
Message register (bit 479 - 448)	SHAMSG14	0x0054
Message register (bit 511 - 480)	SHAMSG15	0x0058
Hash initial value register (bit 31 - 0)	SHAINIT0	0x005C
Hash initial value register (bit 63 - 32)	SHAINIT1	0x0060
Hash initial value register (bit 95 - 64)	SHAINIT2	0x0064
Hash initial value register (bit 127 - 96)	SHAINIT3	0x0068
Hash initial value register (bit 159 - 128)	SHAINIT4	0x006C
Hash initial value register (bit 191 - 160)	SHAINIT5	0x0070
Hash initial value register (bit 223 - 192)	SHAINIT6	0x0074
Hash initial value register (bit 255 - 224)	SHAINIT7	0x0078
Calculation result register (bit 31 - 0)	SHARESULT0	0x007C
Calculation result register (bit 63 - 32)	SHARESULT1	0x0080
Calculation result register (bit 95 - 64)	SHARESULT2	0x0084
Calculation result register (bit 127 - 96)	SHARESULT3	0x0088
Calculation result register (bit 159 - 128)	SHARESULT4	0x008C
Calculation result register (bit 191 - 160)	SHARESULT5	0x0090
Calculation result register (bit 223 - 192)	SHARESULT6	0x0094
Calculation result register (bit 255 - 224)	SHARESULT7	0x0098
Status register	SHASTATUS	0x009C

Note: Do not write any value to the above registers when SHASTATUS<BUSY>=1.

Peripheral function: SRST

Register name		Address (Base+)
Soft reset protect register	SRSTPROTECT	0x0000
Peripheral function soft reset register	SRSTIPRST	0x0004

13.3.2 SHASTART (Process Start 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	–	–	–	–	–	–	–	START
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31 - 1	–	R	Read as "0".
0	START	W	Operation setting 0: Stop 1: Start

Note 1: Do not write any value to this register when SHASTATUS<BUSY>=1.
Note 2: Writing values are ignored when SHADMAEN<DMAEN>=1.

13.3.3 SHACR (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	–	–	–	–	INTEN	HASHINIT		
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31 - 4	–	R	Read as "0".
3	INTEN	R/W	Interrupt control 0: An interrupt is output only at the last calculation. 1: Interrupts are output every time calculation is complete when continuous data is handled.
2 - 0	HASHINIT[2:0]	R/W	Sets the Hash initial value. 000: The Hash value in the previous block is used. 011: The Hash value specified with the SHAINITx register 100: A 256-bit Hash value specified with FIPS PUB 180-3 stored in the core internally. 111: A 224-bit Hash value specified with FIPS PUB 180-3 stored in the core internally. Other settings: Reserved Set the value before the first message block is processed. After calculation is complete, this bit is "000." If calculation is performed continuously, the Hash value of the previous block is used automatically.

Note: Do not write any value to this register when SHASTATUS<BUSY>=1.

13.3.4 SHADMAEN (DMA 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	–	–	–	–	–	–	–	DMAEN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31 - 1	–	R	Read as "0".
0	DMAEN	R/W	DMA operation 0: Disabled 1: Enabled

Note: Do not write any value to this register when SHASTATUS<BUSY>=1.

13.3.5 SHAMSGLEN0 (Whole Message Length Register)

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

Bit	Bit Symbol	Type	Function
31 - 0	LEN[31:0]	R/W	Whole message length (31 bits - 0 bits) Set a message length in unit of byte. If whole message length is undefined, set the maximum value (0xffff_ffff). The whole message length should be set until start of calculation of the last data.

Note: Do not write any value to this register when SHASTATUS<BUSY>=1.

13.3.6 SHAMSGLEN1 (Whole Message Length Register)

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

Bit	Bit Symbol	Type	Function
31 - 29	–	R	Read as "0".
28 - 0	LEN[28:0]	R/W	The whole message length (60 bits to 32 bits) Set a message length in units of byte. If whole message length is undefined, set the maximum value (0xffff_ffff). The whole message length should be set until start of calculation of the last data.

Note: Do not write any value to this register when SHASTATUS<BUSY>=1.

13.3.7 SHAREMAIN0 (Unhandled Message Length Register)

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

Bit	Bit Symbol	Type	Function
31 - 0	REMAIN[31:0]	R/W	Unhandled message length (31 bits - 0 bit) Set a message length in units of byte first. The value of <REMAIN> is updated at each calculation; therefore a remained unhandled message length can be checked. If the whole message length is undefined, set the maximum value (0xffff_ffff). The whole message length should be set until start of calculation of the last data.

Note: Do not write any value to this register when SHASTATUS<BUSY>=1.

13.3.8 SHAREMAIN1 (Unhandled Message Length Register)

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

Bit	Bit Symbol	Type	Function
31 - 29	–	R	Read as "0".
28 - 0	REMAIN[28:0]	R/W	Unhandled message length (60 bits - 32 bits) Set a message length in units of byte first. The value of <REMAIN> is updated at each calculation; therefore a remained unhandled message length can be checked. If the whole message length is undefined, set the maximum value (0xffff_ffff). The whole message length should be set until start of calculation of the last data.

Note: Do not write any value to this register when SHASTATUS<BUSY>=1.

13.3.9 SHAMSG00 to 15 (Message Register)

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

Bit	Bit Symbol	Type	Function
31 - 0	MSG[31:0]	R/W	Message Sets a 512-bit message.

Note: Do not write any value to this register when SHASTATUS<BUSY>=1.

Data is stored as shown below and is input to the SHA core after the data is swapped. For details, refer to the chapter on "13.5 Data Allocation".

Bit	31 24	23 16	15 8	7 0
Bit 511 - 480	SHAMSG15			
Bit 479 - 448	SHAMSG14			
Bit 447 - 416	SHAMSG13			
Bit 415 - 384	SHAMSG12			
Bit 383 - 352	SHAMSG11			
Bit 351 - 320	SHAMSG10			
Bit 319 - 288	SHAMSG09			
Bit 287 - 256	SHAMSG08			
Bit 255 - 224	SHAMSG07			
Bit 223 - 192	SHAMSG06			
Bit 191 - 160	SHAMSG05			
Bit 159 - 128	SHAMSG04			
Bit 127 - 96	SHAMSG03			
Bit 95 - 64	SHAMSG02			
Bit 63 - 32	SHAMSG01			
Bit 31 - 0	SHAMSG00			

13.3.10 SHAINIT0 to 7 (Hash Initial Value Register)

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

Bit	Bit Symbol	Type	Function
31 - 0	INIT[31:0]	R/W	Hash initial value Sets the Hash initial value. The set values are input after these values are swapped. For details, refer to the chapter on "13.5 Data Allocation". When calculation is halted, save the value of the SHARESULT0 to 7 register on the stack in either case of SHA-224 or SHA-256. When calculation is restarted, set the value stored on the stack to the SHAINIT0 to 7 register.

Note: Do not write any value to this register when SHASTATUS<BUSY>=1.

13.3.11 SHARESULT0 to 7 (Calculation Result Register)

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

Bit	Bit Symbol	Type	Function
31 - 0	RESULT[31:0]	R	Calculation result Stores calculation results. When calculation is halted, save the value of the SHARESULT0 to 7 register on the stack in either case of SHA-224 or SHA-256. When calculation is restarted, set the value stored on the stack to the SHAINIT0 to 7 register.

The arithmetic result is output from the SHA core. The result is swapped and stored in the register as shown below. The data can be read from the lower side. For details, refer to the chapter on "13.5 Data Allocation".

bit	255 224	223 192	191 160	159 128	127 96	95 64	63 32	31 0
SHA-224		SHARESULT6	SHARESULT5	SHARESULT4	SHARESULT3	SHARESULT2	SHARESULT1	SHARESULT0
SHA-256	SHARESULT7	SHARESULT6	SHARESULT5	SHARESULT4	SHARESULT3	SHARESULT2	SHARESULT1	SHARESULT0

13.3.12 SHASTATUS (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	–	–	–	–	–	–	–	BUSY
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31 - 1	–	R	Read as "0".
0	BUSY	R	Status of the calculation 0: Calculation is complete. 1: Calculation in process.

13.3.13 SRSTPROTECT (Soft Reset Protect 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	PROTECT							
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 8	–	R	Read as "0".
7 - 0	PROTECT	R/W	0x6B: Enabled Other than 0x6B: Disabled (The SRSTIPRST register cannot be accessed.) This bit becomes "0x00" and disabled after reset. To write data to the SRSTIPRST register, set "0x6B" to this bit.

13.3.14 SRSTIPRST (Peripheral Function Soft 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	–	–	–	IPRST4	IPRST3	IPRST2	IPRST1	IPRST0
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 5	–	R	Read as "0".
4	IPRST4	R/W	SNFC reset 0: Clears the reset 1: Resets
3	IPRST3	R/W	ESG reset 0: Clears the reset 1: Resets
2	IPRST2	R/W	MLA reset 0: Clears the reset 1: Resets
1	IPRST1	R/W	SHA reset 0: Clears the reset 1: Resets
0	IPRST0	R/W	AES reset 0: Clears the reset 1: Resets

13.4 Calculation Process

The SHA calculates 512-bit message blocks. To obtain Hash values of consecutive data, calculate each 512-bit message block, and then the result is used as the next initial Hash value. The result of the last message block is the Hash value to be used.

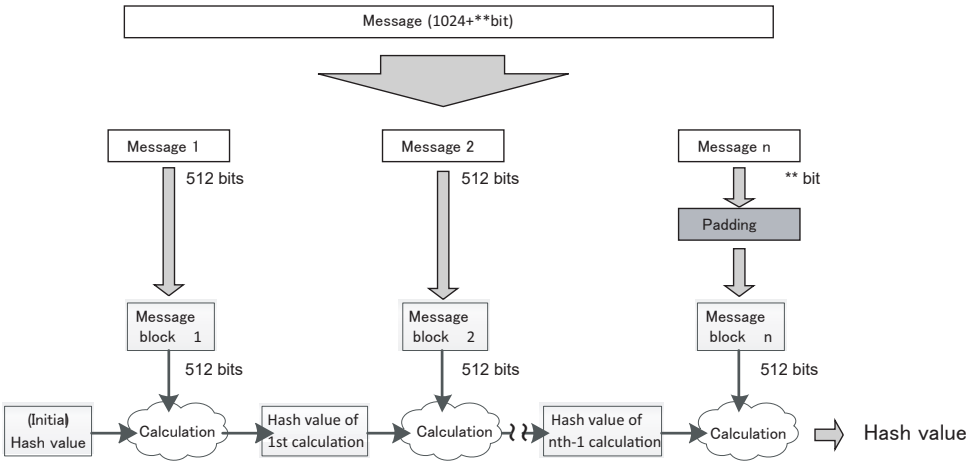


Figure 13-2 Calculation process flow

Specify the message blocks to 512-bit blocks except the last block.

The SHA core performs padding to increase data in the last block, if required.

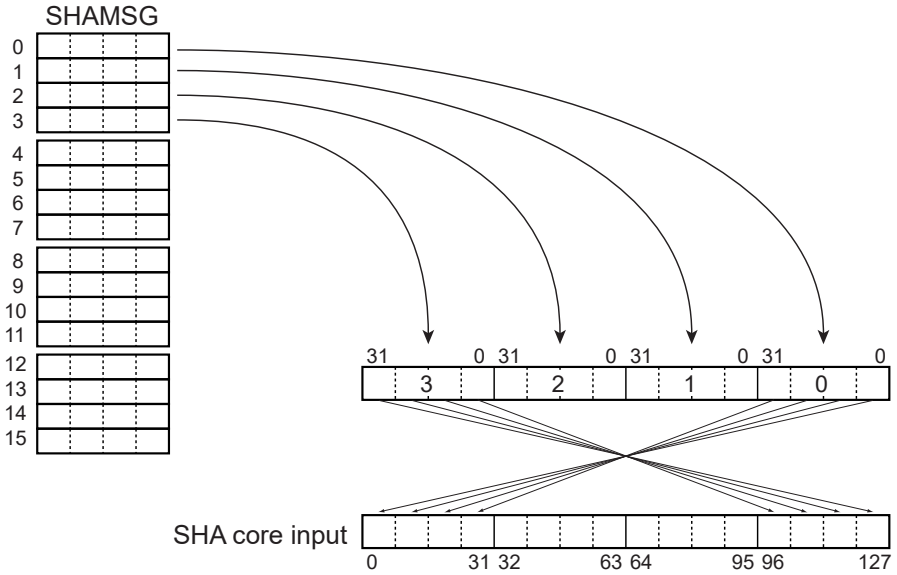
A 64-bit value, which indicates the whole message length, is contained in the last message block. When the message length is 448 bits to 512 bits, the last message block becomes 2 blocks adding 64-bit whole message length. These process is performed automatically.

The last message length cannot be set to "0". In this case, the Hash value can be calculated as follows:

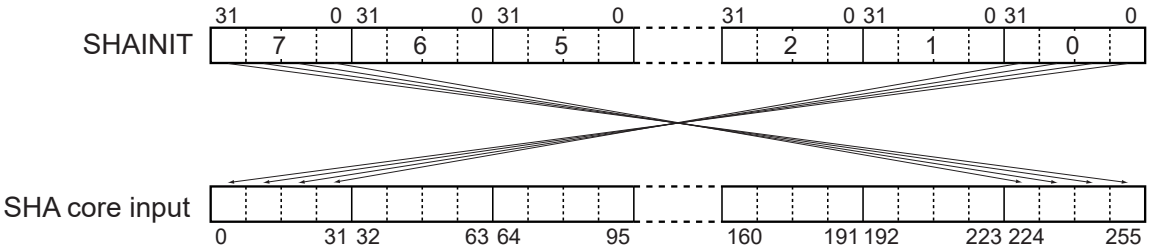
1. Set the Hash values up to ahead of the last message to SHAINIT0 to 7.
2. Set any larger value than 512 bit (64 bytes) to SHAREMAIN0,1.
3. Set 0x80 to SHAMSG00 and set 0x00 to SHAMSG01 to 13.
4. Set the processed message length to SHAMSG14 and 15.

13.5 Data Allocation

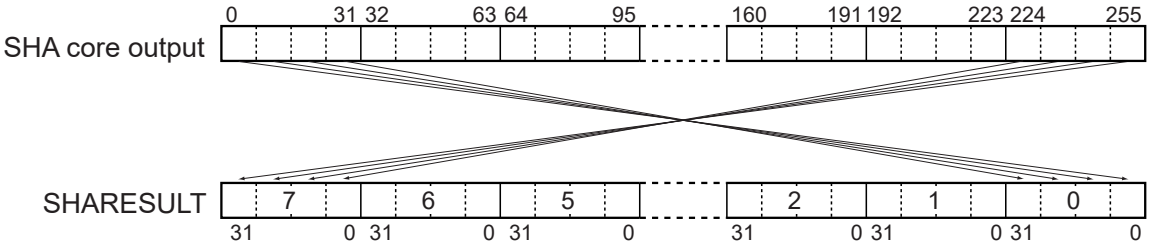
The data stored in SHAMSG0 to 15 is input to the SHA core in units of 4 words (128 bits). The SHA core swaps data as follows:



The data stored in SHAINT0 to 7 is allocated as below and is swapped in units of byte when the data is input to the SHA core.



The arithmetic result is output from the SHA core. The result is swapped in units of byte. For SHA-226, the result is stored in SHARESULT0 to 7. For SHA-224, the result is stored in SHARESULT0 to 6.



13.6 Data Transfer

Message blocks are transferred by the DMA controller or CPU.

13.6.1 DMAC Transfer

Data transfer to the message registers is set with DMAC.

When "1" is set to SHADMAEN<DMAEN>, the SHA outputs a data transfer request (DREQ) to transfer message blocks using DMAC.

After the message block is transferred, the SHA core starts calculation automatically.

When the calculation is complete, if the SHAREMAIN0,1 register is not "0" and has unhandled messages, the SHA outputs a DREQ. The next message is transferred by the DMA. At this time, "000" (the previous Hash value is used) is automatically set to SHACR<HASHINIT>.

When the calculation of the last message is complete, SHADMAEN<DMAEN> is "0", and an interrupt signal (INTSHA) is output. Then, read the calculation result from the SHARESULT0 to 7 register.

13.6.2 CPU Transfer

A message block is set to the SHAMSGI00 to 15 register by the CPU. When "1" is set to SHASTA<START>, the SHA core starts calculation.

When the calculation is complete, if the SHAMSGI00 to 15 register has unhandled messages, the CPU sets the next message block to the SHAMSGI00 to 15 register and continues calculation. When "1" is set to SHACR<INTEN>, an interrupt is output at completion of each calculation.

When the calculation of the last message is complete, an interrupt signal (INTSHA) of calculation completion for the last message block is output. Then, read the calculation result from the SHARESULT0 to 7 register.

13.7 Operation Procedure

The initial Hash value and interrupt setting are specified to SHACR. Set the value of SHAINIT0 to 7 to SHACR if this value is used as the initial Hash value.

Set a whole message length and unhandled message length to SHAMSGLEN0 to 1 and SHAREMAIN0 to 1 respectively. If the whole message length is undefined, set the maximum value. In this case, the whole message length and unhandled message length must be specified until the last message block is calculated.

Set messages to SHAMSG00 to 15 by the CPU or DMAC. When the DMAC transfer is used, calculation starts automatically. When the CPU transfer is used, set "1" to SHASTART<START> to start calculation. For details, refer to "13.6 Data Transfer".

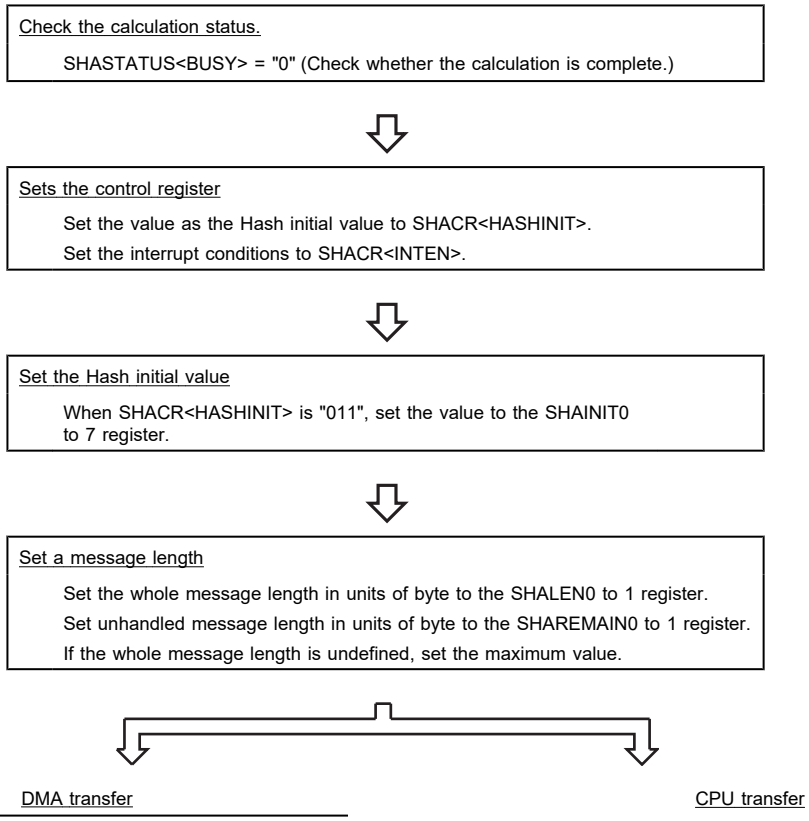
When all messages are calculated, an interrupt is output and the calculation result is stored in the register.

The table below lists the required number of cycles in which Hash calculation for the message blocks (0 to 64 bytes) is performed.

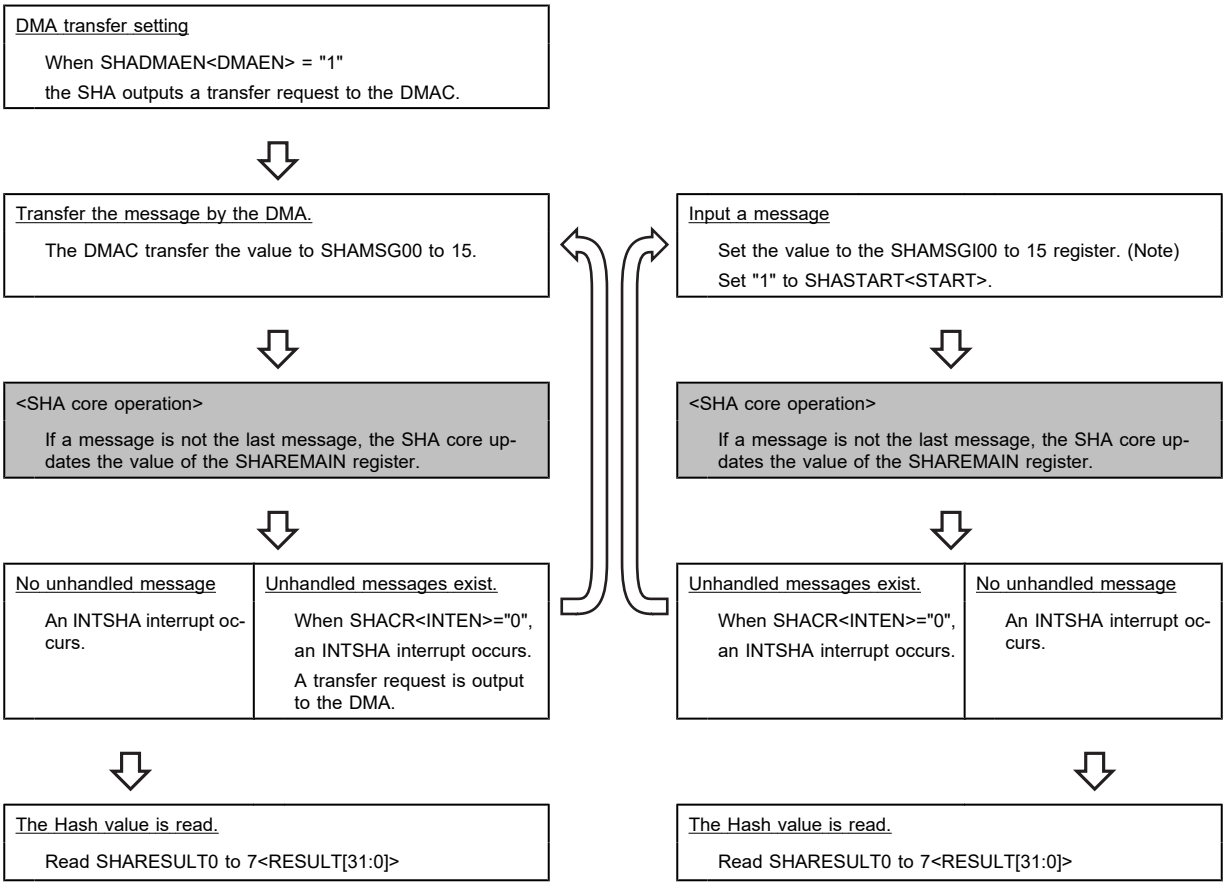
Message block length	The number of calculation cycles
0 to 16 bytes	75 cycles
17 to 32 bytes	76 cycles
33 to 48 bytes	77 cycles
49 to 55 bytes	78 cycles
56 to 64 bytes	148 cycles

Details of setting procedure is as shown below:

Setting procedure



Setting procedure



Note: If calculation is consecutively performed, confirm whether SHASTATUS<BUSY> is "0", and then set a new message to the SHAMSGI00 to 15 register.

13.8 Reset Operation

When "1" is set to SRSTIPRST<IPRST1>, all internal circuits and the registers in the SHA are reset. When "0" is set to SRSTIPRST<IPRST1>, the reset state is released.

The SRSTIPRST register becomes write protection state with the SRSTPROTECT register after reset.

When the reset is performed, write "0x6B" to the SRSTPROTECT register to enable to be written (protection state is released).

14. Entropy Seed Generator (ESG)

14.1 Outline

The entropy seed generator (ESG) is a circuit that generates a 512-bit entropy seed using a ring oscillator.

14.2 Structure

The figure below shows the block diagram of the ESG.

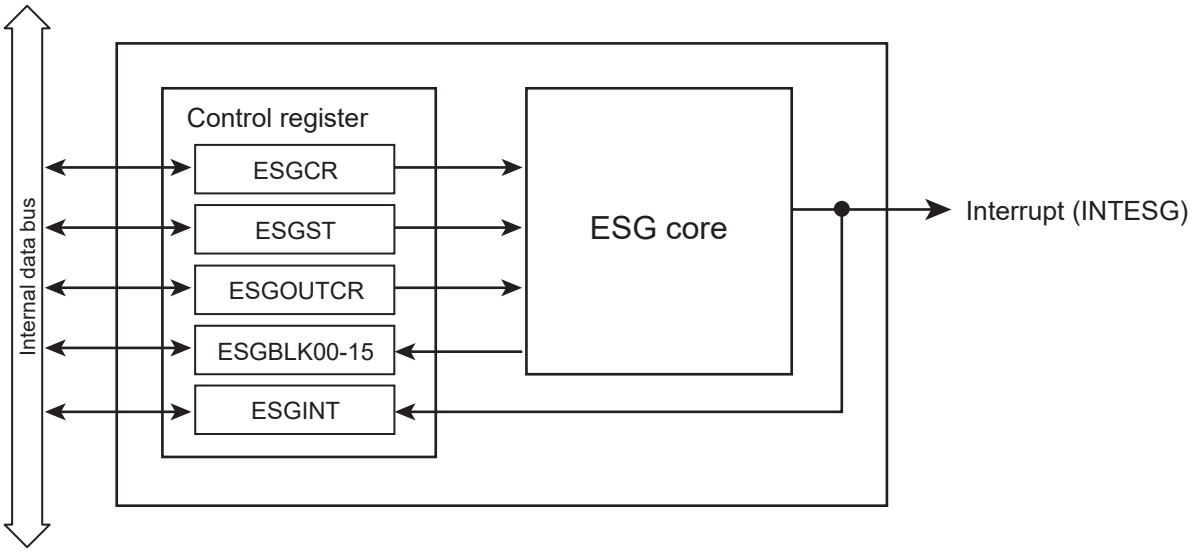


Figure 14-1 Block diagram of the ESG

14.3 Registers

The following table lists the control registers and their addresses:

For the base address, refer to "A list of peripheral function base addresses" in the chapter on "Memory Map."

14.3.1 Register List

Peripheral function: ESG

Register name		Address (Base+)
Control register	ESGCR	0x0000
Status register	ESGST	0x0004
Output control Register	ESGOUTCR	0x0008
Interrupt status register	ESGINT	0x000C
Entropy seed store block 00	ESGBLK00	0x0010
Entropy seed store block 01	ESGBLK01	0x0014
Entropy seed store block 02	ESGBLK02	0x0018
Entropy seed store block 03	ESGBLK03	0x001C
Entropy seed store block 04	ESGBLK04	0x0020
Entropy seed store block 05	ESGBLK05	0x0024
Entropy seed store block 06	ESGBLK06	0x0028
Entropy seed store block 07	ESGBLK07	0x002C
Entropy seed store block 08	ESGBLK08	0x0030
Entropy seed store block 09	ESGBLK09	0x0034
Entropy seed store block 10	ESGBLK10	0x0038
Entropy seed store block 11	ESGBLK11	0x003C
Entropy seed store block 12	ESGBLK12	0x0040
Entropy seed store block 13	ESGBLK13	0x0044
Entropy seed store block 14	ESGBLK14	0x0048
Entropy seed store block 15	ESGBLK15	0x004C

Note: Do not write any value to the above registers when ESGST<BUSY>=1.

Peripheral function: SRST

Register name		Address (Base+)
Soft reset protect register	SRSTPROTECT	0x0000
Peripheral function soft reset register	SRSTIPRST	0x0004

14.3.2 ESGCR (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	–	–	–	–	–	–	–	START
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 9	–	R	Read as "0."
8	–	R/W	Write as "0."
7 - 5	–	R	Read as "0."
4	–	R/W	Write as "0."
3 - 1	–	R	Read as "0."
0	START	W	Startup setting 1: Startup Writing "0" has no meaning. Read as "0."

Note: Do not write any value to the register when ESGST<BUSY>=1.

14.3.3 ESGST (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	–	–	–	–	–	–	–	BUSY
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 1	–	R	Read as "0."
0	Busy	R	Operation status 0: Stop 1: In operation

14.3.4 ESGOUTCR (Entropy Seed Output Timing 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	–	–	–	–	LATCHTIMING			
After reset	0	0	0	0	1	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	FINTIMING							
After reset	0	0	0	1	0	0	1	0
	7	6	5	4	3	2	1	0
Bit symbol	FINTIMING							
After reset	0	0	0	0	0	0	0	1

Bit	Bit symbol	Type	Function
31 - 20	–	R	Read as "0."
19 - 16	LATCHTIMING [3:0]	R/W	Entropy seed latch timing 0000 : 1 cycle 0001: 2 cycles 1111: 16 cycles Sets the cycle in which a 512-bit entropy seed is latched one-by-one.
15 - 0	FINTIMING [15:0]	R/W	Entropy seed output timing Settable minimum value must be satisfied the conditions below: FINTIMING > (512 × (LATCHTIMING + 1)) + 2

Note: Do not write any value to the register when ESGST<BUSY>=1.

14.3.5 ESGINT (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	–	–	–	–	–	–	–	INT
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 1	–	R	Read as "0."
0	INT	R/W	[Read]: Interrupt status 0: No interrupt 1: Interrupt occurs. [Write]: Interrupt clear setting 1: Clear the interrupt Writing "0" has no meaning.

Note: Do not write any value to the register when ESGST<BUSY>=1.

14.3.6 ESGBLK00-15 (Entropy Seed Store Block 00-15)

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

Bit	Bit symbol	Type	Function
31 - 2	BLK[31:0]	R	Stores an entropy seed.

14.3.7 SRSTPROTECT (Soft Reset Protect 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	PROTECT							
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 8	–	R	Read as "0".
7 - 0	PROTECT	R/W	0x6B: Enabled Other than 0x6B: Disabled (The SRSTIPRST register cannot be accessed.) This bit becomes "0x00" and disabled after reset. To write data to the SRSTIPRST register, set "0x6B" to this bit.

14.3.8 SRSTIPRST (Peripheral Function Soft 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	–	–	–	IPRST4	IPRST3	IPRST2	IPRST1	IPRST0
After reset	0	0	0	0	0	0	0	0

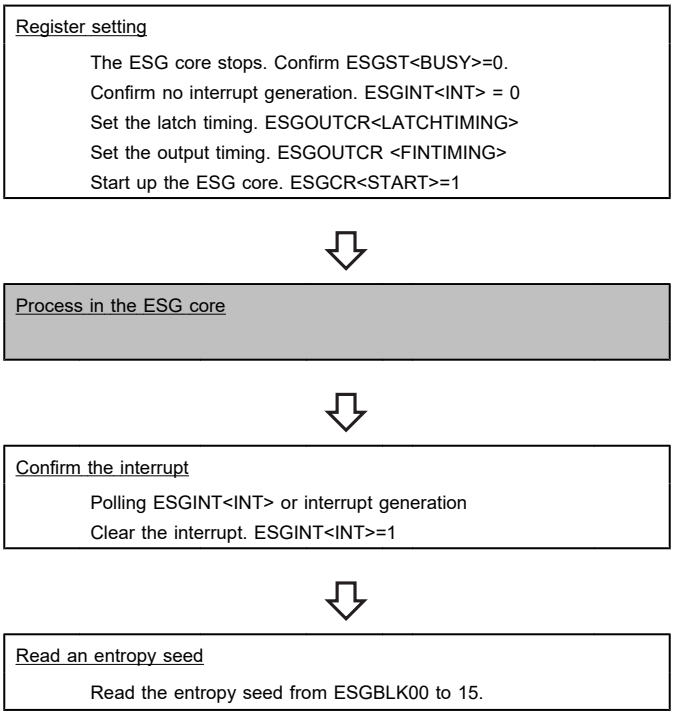
Bit	Bit symbol	Type	Function
31 - 5	–	R	Read as "0".
4	IPRST4	R/W	SNFC reset 0: Clears the reset 1: Resets
3	IPRST3	R/W	ESG reset 0: Clears the reset 1: Resets
2	IPRST2	R/W	MLA reset 0: Clears the reset 1: Resets
1	IPRST1	R/W	SHA reset 0: Clears the reset 1: Resets
0	IPRST0	R/W	AES reset 0: Clears the reset 1: Resets

14.4 Operation Description

To start up the ESG core, set the latch timing and output timing. Before starting up the ESG core, must confirm that the core stops and the interrupts are cleared.

When the ESG core ends the operation, it outputs an interrupt. Then an entropy seed can be read. Interrupt generation can be checked by polling of the interrupt status register.

Process flow



14.5 Reset Operation

When "1" is set to SRSTIPRST<IPRST3>, all internal circuits and the registers in the ESG are reset. When "0" is set to SRSTIPRST<IPRST3>, the reset state is released

The SRSTIPRST register becomes write protection state with the SRSTPROTECT register after reset.

When the reset is performed, write "0x6B" to the SRSTPROTECT register to enable to be written (protection state is released).

14.6 Precautions

14.6.1 Use time of ESG

Use the ESG at a duty ratio of 1% or less.

15. Multiple Length Arithmetic Coprocessor (MLA)

The Multiple Length Arithmetic coprocessor (MLA) performs calculation for Elliptic Curve Cryptography (ECC) with 256-bit key length.

15.1 Outline

The MLA supports 3 algorithms below:

- Montgomery multiplication (256bit)
- Multiple length addition
- Multiple length subtraction

15.2 Configuration

The figure below shows the block diagram of the MLA.

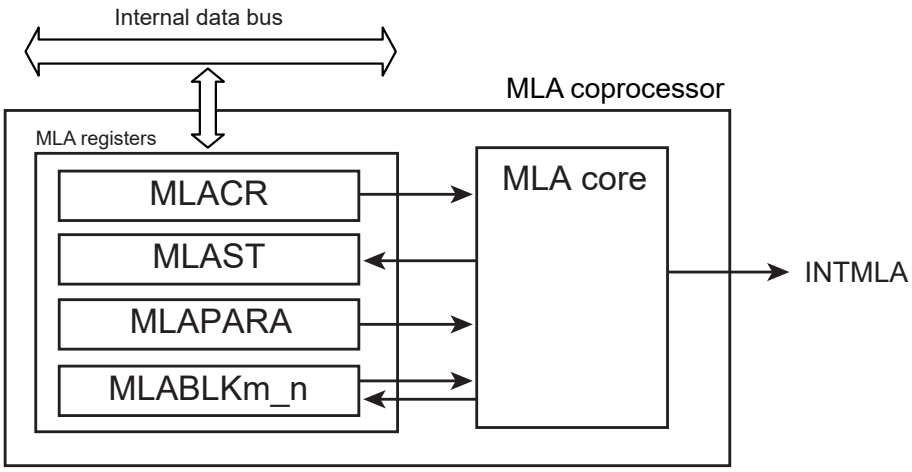


Figure 15-1 Block diagram of the MLA

15.3 Registers

The following table lists the control registers and their addresses:

For the base address, refer to "A list of peripheral function base addresses" in the chapter on "Memory Map."

15.3.1 Register List

Peripheral function: MLA

Register name		Address (Base+)
Control register	MLACR	0x0000
Status register	MLAST	0x0004
Montgomery parameter register	MLAPARA	0x000C
General-purpose register block 1	MLABLK1_n(note2)	0x0010 - 0x002C
General-purpose register block 2	MLABLK2_n(note2)	0x0030 - 0x004C
General-purpose register block 3	MLABLK3_n(note2)	0x0050 - 0x006C
General-purpose register block 4	MLABLK4_n(note2)	0x0070 - 0x008C
General-purpose register block 5	MLABLK5_n(note2)	0x0090 - 0x00AC
General-purpose register block 6	MLABLK6_n(note2)	0x00B0 - 0x00CC
General-purpose register block 7	MLABLK7_n(note2)	0x00D0 - 0x00EC
General-purpose register block 8	MLABLK8_n(note2)	0x00F0 - 0x010C
General-purpose register block 9	MLABLK9_n(note2)	0x0110 - 0x012C
General-purpose register block 10	MLABLK10_n(note2)	0x0130 - 0x014C
General-purpose register block 11	MLABLK11_n(note2)	0x0150 - 0x016C
General-purpose register block 12	MLABLK12_n(note2)	0x0170 - 0x018C
General-purpose register block 13	MLABLK13_n(note2)	0x0190 - 0x01AC
General-purpose register block 14	MLABLK14_n(note2)	0x01B0 - 0x01CC
General-purpose register block 15	MLABLK15_n(note2)	0x01D0 - 0x01EC
General-purpose register block 16	MLABLK16_n(note2)	0x01F0 - 0x020C
General-purpose register block 17	MLABLK17_n(note2)	0x0210 - 0x022C
General-purpose register block 18	MLABLK18_n(note2)	0x0230 - 0x024C
General-purpose register block 19	MLABLK19_n(note2)	0x0250 - 0x026C
General-purpose register block 20	MLABLK20_n(note2)	0x0270 - 0x028C
General-purpose register block 21	MLABLK21_n(note2)	0x0290 - 0x02AC
General-purpose register block 22	MLABLK22_n(note2)	0x02B0 - 0x02CC
General-purpose register block 23	MLABLK23_n(note2)	0x02D0 - 0x02EC
General-purpose register block 24	MLABLK24_n(note2)	0x02F0 - 0x030C
General-purpose register block 25	MLABLK25_n(note2)	0x0310 - 0x032C
General-purpose register block 26	MLABLK26_n(note2)	0x0330 - 0x034C
General-purpose register block 27	MLABLK27_n(note2)	0x0350 - 0x036C
General-purpose register block 28	MLABLK28_n(note2)	0x0370 - 0x038C
General-purpose register block 29	MLABLK29_n(note2)	0x0390 - 0x03AC
General-purpose register block 30	MLABLK30_n(note2)	0x03B0 - 0x03CC
General-purpose register block 31	MLABLK31_n(note2)	0x03D0 - 0x03EC
General-purpose register block 0	MLABLK0_n(note2)	0x0800 - 0x081C

Note 1: Do not access to the registers above except to read the MLAST register when MLAST<BUSY>=1.
Note 2: n=0 to 7

Peripheral function: SRST

Register name		Address (Base+)
Soft reset protect register	SRSTPROTECT	0x0000
Peripheral function soft reset register	SRSTIPRST	0x0004

15.3.2 MLACR (Control Register)

	31	30	29	28	27	26	25	24
Bit symbol	–	–	–	–	–	COM		
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
Bit symbol	–	–	–	SRC1				
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
Bit symbol	–	–	–	SRC2				
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	RDB				
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31 - 27	–	R/W	Write as "0."
26 - 24	COM[2:0]	R/W	Calculation mode 001: Montgomery multiplication (256bit) 010: Multiple length addition 100: Multiple length subtraction Do not other values other than the above.
23 - 21	–	R/W	Write as "0."
20 - 16	SRC1[4:0]	R/W	Data block number Set a data block number that is substituted into "a" in "Table 15-1 equations". Do not set "0" or "1" when Montgomery calculation is performed.
15 - 13	–	R/W	Write as "0."
12 - 8	SRC2[4:0]	R/W	Data block number Set a data block number that is substituted into "b" in "Table 15-1 equations". Do not set "0" or "1" when Montgomery calculation is performed.
7 - 5	–	R/W	Write as "0."
4 - 0	RDB[4:0]	R/W	Block number for the calculation result output register Set a number that is substituted into "w" in "Table 15-1 equations".

Note: Do not access to this register when MLAST<BUSY>=1.

Set data block numbers, which are substituted into to "a", "b", and "w" in the following equations, to <SRC1>, <SRC2>, and <RDB> respectively.

Table 15-1 equations

<COM>	Equation
001	$w = a*b*R^{-1} \bmod P$
010	w=a+b
100	w=a-b

15.3.3 MLAST (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	–	–	–	–	–	–	BUSY	CABO
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31 - 2	–	R/W	Write as "0."
1	BUSY	R/W	Status of the calculation 0: Stop 1: Calculation in progress This flag indicates the status of the calculation.
0	CABO	R/W	Carry and borrow generation flag 0: No carry or borrow flag 1: A carry or borrow flag occurs. This flag indicates the occurrence of a carry or borrow flag during the calculation.

Note: This register is automatically cleared when the next calculation starts.

15.3.4 MLAPARA (Montgomery Parameter Register)

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

Bit	Bit Symbol	Type	Function
31 - 0	PARA[31:0]	R/W	Stores montgomery parameter.

Note: Do not access to this register when MLAST<BUSY>=1.

15.3.5 MLABLKm_n (m=0 to 31, n=0 to 7) (General-purpose Register Block)

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

Bit	Bit Symbol	Type	Function
31 - 0	BLK[31:0]	R/W	Stores calculation input data, calculation value in progress, and calculation results. When Montgomery calculation is performed, MLABLK1_n is used to store the divisor.

Note: Do not access to this register when MLAST<BUSY>=1.

15.3.6 SRSTPROTECT (Soft Reset Protect 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	PROTECT							
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 8	–	R	Read as "0".
7 - 0	PROTECT	R/W	0x6B: Enabled Other than 0x6B: Disabled (The SRSTIPRST register cannot be accessed.) This bit becomes "0x00" and disabled after reset. To write data to the SRSTIPRST register, set "0x6B" to this bit.

15.3.7 SRSTIPRST (Peripheral Function Soft 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	–	–	–	IPRST4	IPRST3	IPRST2	IPRST1	IPRST0
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31 - 5	–	R	Read as "0".
4	IPRST4	R/W	SNFC reset 0: Clears the reset 1: Resets
3	IPRST3	R/W	ESG reset 0: Clears the reset 1: Resets
2	IPRST2	R/W	MLA reset 0: Clears the reset 1: Resets
1	IPRST1	R/W	SHA reset 0: Clears the reset 1: Resets
0	IPRST0	R/W	AES reset 0: Clears the reset 1: Resets

15.4 Operation Description

15.4.1 Calculation

The MLA circuit can select one of three calculations: Montgomery multiplication, multiple length addition, or Multiple length subtraction.

The number of calculation execution clocks is determined by the calculation mode to be used and the block number specified with <RDB>, <SRC1>, and <SRC2>.

- Montgomery multiplication

  Equation: $w = a*b*R^{-1}modP$

Condition	The number of execution clocks
<RDB> is 0.	$2n^2+4n+5$
<RDB> is not 0.	$2n^2+5n+6$

Note1) This function does not implement the process in which the calculation result called Final Subtraction is suppressed below the range of Modulus p. It is assumed that users additionally execute the multiple length subtraction provided in this circuit.

Note2) The data length which this product can treat is 256 bit.

- Multiple length addition and multiple length subtraction

  Equation: w=a+b, w=a-b

Condition		Number of execution clocks
<RDB>	<SRC1>/<SRC2>	
0	<SRC1>=<SRC2> or at least either <SRC1> or <SRC2> is 0.	n+3
	<SRC1> and <SRC2> is not equal and <SRC1> or <SRC2> is 0x1 to 0x1F.	2n+4
Other than 0	<SRC1>=<SRC2> or at least either <SRC1> or <SRC2> is 0.	2n+4
	<SRC1> and <SRC2> are not equal and <SRC1> or <SRC2> is 0x1 to 0x1F.	3n+5

15.4.2 Setting Procedure

<u>Check the status of the calculation.</u>

MLAST<BUSY>="0"(Check the calculation result.)

⇩

<u>Sets calculation input data.</u>

Set values into "a" and "b" of the equations. (MLABLKxx, MLABLKyy)

⇩

<u>Sets the MLACR register</u>

Calculation mode setting: MLACR<COM>

Set the calculation input data block: MLACR<SRC1><SRC2>

Set the block number for the calculation result output register MACR<RDB>

⇩

<u>Automatic process</u>

MLAST<BUSY>=1 and the calculation starts.

⇩

Check whether the calculation is complete.
An INTMLA interrupt occurs.

15.5 Reset Operation

When "1" is set to SRSTIPRST<IPRST2>, all internal circuits and the registers in the MLA are reset. When "0" is set to SRSTIPRST<IPRST2>, the reset state is released.

The SRSTIPRST register becomes write protection state with the SRSTPROTECT register after reset.

When the reset is performed, write "0x6B" to the SRSTPROTECT register to enable to be written (protection state is released).

16. 16-bit Timer / Event Counters (TMRB)

16.1 Outline

TMRBx has the operation modes shown below.

- Interval timer mode
- Event counter mode
- Programmable pulse generation (PPG) mode
- Programmable pulse generation (PPG) external trigger mode

The use of the capture function allows TMRBx to perform the following measurements.

- Frequency measurement
- Pulse width measurement

16.2 Block Diagram

TMRBx consists of a 16-bit up-counter, two 16-bit timer register (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 registers.

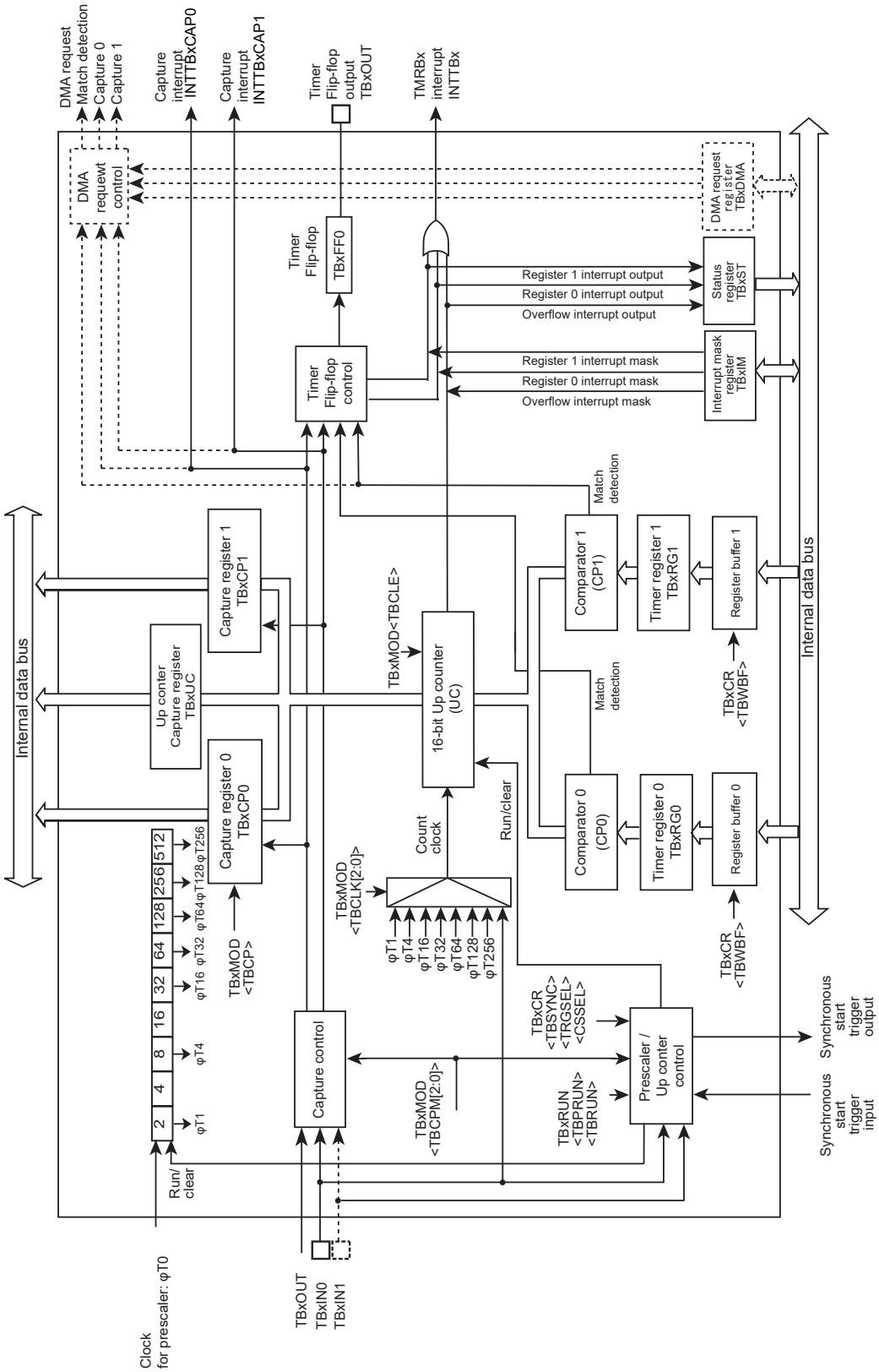


Figure 16-1 TMRBx Block Diagram

16.3 Registers

16.3.1 Register List

The table below shows control registers and their addresses.

For details of the base address, refer to "Address lists of peripheral functions" of "Memory Map" chapter.

Register name		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

Note: Do not modify the timer control register, timer mode register and timer flip-flop control register during timer operation. Users can modify them after stopping timer operation.

16.3.2 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	TBHALT	-	-	-	-	-	-
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: Disabled 1: Enabled Specifies the TMRBx operation. When the operation is disabled, no clock is supplied to the other registers in the TMRBx. This can reduce power consumption. (This disables reading from and writing to the other registers except TBxEN register.) To use the TMRB, enable the TMRBx operation (set to "1") before programming each register in the TMRBx. If the TMRBx operation is executed and then disabled, the settings will be maintained in each register.
6	TBHALT	R/W	Clock operation during debug HALT 0: run 1: stop Specifies the TMRBx clock setting to run or stop when the debug tool transits to HALT mode while in use.
5-0	–	R	Read as "0".

16.3.3 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

16.3.4 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	-	TRGSEL	CSSEL
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: Disabled 1: Enabled
6	–	R/W	Write "0".
5	TBSYNC	R/W	Synchronous mode switching 0: individual (Each channel) 1: synchronous
4	–	R	Read as "0".
3	I2TB	R/W	Operation at IDLE mode 0: Stop 1:Operation
2	–	R/W	Write "0".
1	TRGSEL	R/W	Selects the edge when the external trigger is used. 0: rising 1: falling Selects the edge when counting is started by external triggers (TBxIN).
0	CSSEL	R/W	Selects the count start 0: starts by software 1: starts by external trigger

16.3.5 TBxMOD (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	-	-	-	-	-	TBCPM		
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	TBCP	-	-	TBCLE	TBCLK		
After reset	0	1	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-11	−	R	Read as "0".
10-8	TBCPM[2:0]	R/W	Sets capture timing by TBxIN0/1 and up-counter clearing timing. 000: Disabled 001: TBxIN0↑ TBxIN1↑ Captures a counter value on rising edge of TBxIN0 input into Capture register 0 (TBxCP0). Captures a counter value on rising edge of TBxIN1 input into Capture register 1 (TBxCP1). 010: TBxIN0↑ TBxIN0↓ Captures a counter value on rising edge of TBxIN0 input into Capture register 0 (TBxCP0). Captures a counter value on falling edge of TBxIN0 input into Capture register 1 (TBxCP1). 011: TBxFF0↑ TBxFF0↓ Captures a counter value on rising edge of TBxFF0 input into Capture register 0 (TBxCP0). Captures a counter value on falling edge of TBxFF0 input into Capture register 1 (TBxCP1). 100: Clears up-counter on TBxIN1↑ 101: Captures a counter value on TBxIN0↑ into Capture register 0(TBxCP0); clears up-counter on TBxIN1↑ If capture timing and up-counter clearing timing are same, capturing is performed first, and then up-counter is cleared. 110 to 111:Reserved
7	−	R/W	Write "0".
6	TBCP	W	Capture control by software 0: Capture by software 1: Don’t care When "0" is written, the capture register 0 (TBxCP0) captures a count value. Read as "1".
5-4	−	R	Read as "0".
3	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 up-counter matches with timer regsiter1 (TBxRG1).
2-0	TBCLK[2:0]	R/W	Selects the TMRBx source clock. 000: TBxIN pin input 001: φT1 010: φT4 011: φT16 100: φT32 101: φT64 110: φT128 111: φT256

Note: Do not make any changes of TBxMOD register while the TMRBx is running.

16.3.6 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 (reversed by software). 01: Set Sets TBxFF0 to "1". 10: Clear Clears TBxFF0 to "0". 11: Don't care This is always read as "11".

16.3.7 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 interrupt request flag 0:No overflow occurs 1:Overflow occurs When an up-counter is overflow, "1" is set.
1	INTTB1	R	Match (TBxRG1) interrupt request flag 0:No match is detected. 1:Detects a match with TBxRG1 When a match with the timer register 1 (TBxRG1) is detected, "1" is set.
0	INTTB0	R	Match(TBxRG0) interrupt request flag 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: Even if mask configuration by TBxIM register is valid, the status is set to TBxST register.

Note 2: When the interrupt mask configuration is disabled by the corresponding bit of TBxIM register, the interrupt is issued to the CPU.

Note 3: To clear the flag, read TBxST register.

16.3.8 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 request mask 0:Disable 1:Enable Sets the up-counter overflow interrupt to disable or enable.
1	TBIM1	R/W	Match (TBxRG1) interrupt request mask 0:Disable 1:Enable Sets the match interrupt request mask with the timer register 1 (TBxRG1) to enable or disable.
0	TBIM0	R/W	Match (TBxRG0) interrupt request mask 0:Disable 1:Enable Sets the match interrupt request mask with the Timer register 0 (TBxRG0) to enable or disable.

Note: Even if mask configuration by TBxIM register is valid, the status is set to TBxST register.

16.3.9 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 during the counter operation, the current value of up-counter will be captured.

16.3.10 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.

16.3.11 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.

16.3.12 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.

16.3.13 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.

16.4 Description of Operation

16.4.1 Prescaler

TMRBx has 4-bit prescaler to generate the source clock for up-counter.

A input clock φT0 to the prescaler is either among fperiph/1, fperiph/2, fperiph/4, fperiph/8, fperiph/16 or fperiph/32 selected by CGSYSCR<PRCK[2:0] in the CG circuit. The peripheral clock is either fgear (a clock selected by CGSYSCR<FPSEL> in the CG circuit) or fc (a clock before divided by the clock gear).

The operation or the stoppage of prescaler is set by TBxRUN<TBPRUN>. Writing "1" to the bit is to start counting; writing "0" to the bit is to stop counting.

16.4.2 Up-counter (UC)

UC is a 16-bit binary counter.

16.4.2.1 Source clock

The up-counter’s source clock is specified by TBxMOD<TBCLK[2:0]>.

It can be selected from the prescaler output clock - φT1, φT4, φT16, φT32, φT64, φT128 and φT256 - or the external clock of the TBxIN pin.

16.4.2.2 Counter start / stop

There are software start, external trigger start and synchronous start to start the counter.

1. Software start

   If <TBRUN> is set to "1", the counter will start. If "0" is set to the <TBRUN>, the counter will stop and the up-counter will be cleared at the same time.

2. External trigger start

   In the external trigger mode, the counter will be started by external signals.

   If TBxCR<CSSEL> is set to "1", the external trigger start mode is set. At this time, if <TBRUN> is set to "1", the condition of the counter will be trigger wait. The counter will start on the rising/falling edge of TBxIN.

   TBxCR<TRGSEL> bit specifies the switching external trigger edges.

   <TRGSEL>="0": Rising edge of TBxIN is selected.

   <TRGSEL>="1": Falling edge of TBxIN is selected.

   If <TBRUN> is set to "0", the counter will stop and the up-counter will be cleared at the same time.

3. Synchronous start

   In the timer synchronous mode, synchronous start timers can be possible. If timer synchronous mode is used in the PPG output mode, motor drive application can be achieved.

   Depending on products, the combination of master channels and slave channels have already been determined. For the combination of master channels and slave channels of this product, refer to Chapter Product Information.

TBxCR<TBSYNC> bit specifies the switching of synchronous mode. If <TBSYNC> bit of a slave channel is set to "1", the counter will start/stop synchronously with the software or external trigger start of a master channel. TBxRUN<TBPRUN, TBRUN> bit of a slave channel is not required to set. <TBSYNC> bit of the master channel must be set to "0".

Note that if the external trigger counter mode and timer synchronous mode are both set, the timer synchronous mode gains a higher priority.

16.4.2.3 Counter Clear

The up-counter is cleared at the timings below:

1. When a match with TBxRG1 is detected

   By setting TBxMOD<TBCLE> = "1", up-counter is cleared if the comparator detects a match between UC and TBxRG1.

2. When up-counter stops

   Up-counter stops and is cleared when TBxRUN<TBRUN> = "0".

3. When TBxIN1 rises

   Up-counter is cleared on rising edge of TBxIN1 when TBxMOD<TBCPM[2:0]> is "100" or "101".

16.4.2.4 Up-Counter Overflow

If up-counter overflows, the INTTBx overflow interrupt is generated.

16.4.3 Timer Registers (TBxRG0, TBxRG1)

TBxRG0 and TBxRG1 are registers to compare with a value of up-counter. The two registers are built into each channel. If the comparator detects a match between a value set in timer register and a value in the up-counter, the comparator outputs the match detection signal.

TBxRG0 and TBxRG1 are double buffered configuration that are paired with register buffers. The double buffering is disabled in the initial state.

Disabling or Enabling of double buffering is specified by TBxCR<TBWBF>. If <TBWBF> = 0, double buffering becomes disable; if <TBWBF> ="1", it becomes enable.

When double buffering is enabled, data is transferred from the register buffer to the timer register (TBxRG0/1) when up-counter is matched with TBxRG1.

When up-counter is stopped even if double buffering is enabled, the double buffering operates as a single buffer, and data can be written to the TBxRG0 and TBxRG1 directly.

16.4.4 Capture Control

This is a circuit that controls the timing of latching values from UC into the TBxCP0 and TBxCP1. The capture timing of up-counter is specified by TBxMOD<TBCPM[1:0]>.

Software can also capture the value of up-counter into capture registers. The value of up-counter are captured into the TBxCP0 in each time "0" is written to TBxMOD<TBCP>.

16.4.5 Capture Registers (TBxCP0, TBxCP1)

These registers capture the value of up-counter.

16.4.6 Up-Counter Capture Register (TBxUC)

If TBxUC register is read during the counter operation, the current value of up-counter will be captured and the value will be read. The value captured at the end is held while the counter is stopping.

16.4.7 Comparators (CP0, CP1)

These circuits compare up-counter and values set to TBxRG0/1 to detect a match. If a match is detected, INTTBx occurs.

16.4.8 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. Reversing is enabled or disabled by setting the TBxFFCR<TBC1T1, TBC0T1, TBC1T1, TBC1T0>.

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, program the corresponding port settings beforehand.

16.4.9 Capture Interrupt (INTTBxCAP0, INTTBxCAP1)

INTTBxCAP0 and INTTBxCAP1 can be generated at the timing of latching value from the up-counter into TBxCP0 and TBxCP1.

16.5 Description of Operation for each mode

16.5.1 Interval Timer Mode

When interrupts is generated in constant intervals, set the interval time to the timer register (TBxRG1) to generate the INTTBx interrupt.

TBxEN<TBEN> = "1"	Enables TMRBx operation.
TBxRUN<TBPRUN><TBRUN> = "00"	Stops prescaler and counter.
Enable INTTBx interrupt	Set "1" to the bit corresponding to INTTBx interrupt to enable the interrupt.
TBxFFCR<TBC1T1><TBC0T1><TBE1T1><TBE0T1> = "0000"	Sets TBxFF0 not to invert the signal by detection of a match between TBxRG0 and TBxRG1, capturing TBxCP0 and TBxCP1.
TBxMOD<TBCPM[2:0]> = "000" TBxMOD<TBCP> = "1" TBxMOD<TBCLE> = "0" TBxMOD<TBCLK[2:0]> = "***" (*** = "001" to "111")	Changes to prescaler output clock as input clock. Specifies capture function to disable.
TBxRG1 = 0x****	Specifies a timer interval.
TBxRUN<TBPRUN><TBRUN> = "11"	Starts prescaler and counter.

Note:*; Optional value

16.5.2 Event Counter Mode

It is possible to make TMRBx the event counter by using a source clock as an external clock (TBxIN pin input).

The up-counter counts up on the rising edge of TBxIN pin input. The value of up-counter can be captured by software. It is possible to read the count value by reading capture values.

TBxEN<TBEN> = "1"	Enables TMRBx operation.
TBxRUN<TBPRUN><TBRUN> = "00"	Stops prescaler and counter.
Allocate the corresponding port to .	
TBxFFCR<TBC1T1><TBC0T1><TBE1T1><TBE0T1> = "0000"	Sets TBxFF0 not to reverse the signal by detection of a match between TBxRG0 and TBxRG1, capturing TBxCP0 and TBxCP1.
TBxMOD<TBCPM[2:0]> = "000" TBxMOD<TBCP> = "1" TBxMOD<TBCLE> = "0" TBxMOD<TBCLK[2:0]> = "000"	Changes input clock to .
TBxRUN<TBPRUN><TBRUN> = "11"	Starts prescaler and counter.
TBxMOD<TBCP> = "0"	Captures a counter value by software.

Note:*; Optional value

16.5.3 Programmable Pulse Generation (PPG) Output Mode

Square wave can be output in any frequency and duty. The output pulse can be either low-active or high-active.

TBxFF0 is reversed when the up-counter matches the set value of TBxRG0 and TBxRG1. TBxFF0 can be output from TBxOUT pin.

Note that the set value of TBxRG0 and TBxRG1 must be satisfy the following requirement.

Set value of TBxRG0 < Set value of TBxRG1

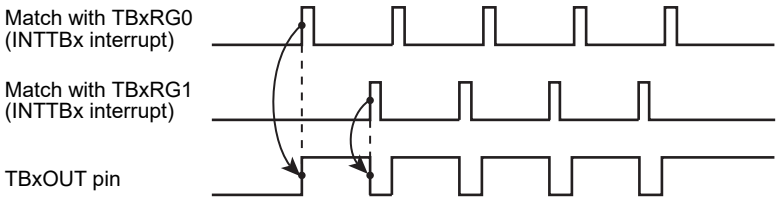


Figure 16-2 Example of programmable pulse generation output

In this mode, by enabling the double buffering, the value of register buffer 0 and 1 are shifted into TBxRG0 and TBxRG1 when UC matches the value of TBxRG1.

This makes possible to modify frequency and duty without a concern of a change timing of TBxRG0 and TBxRG1.

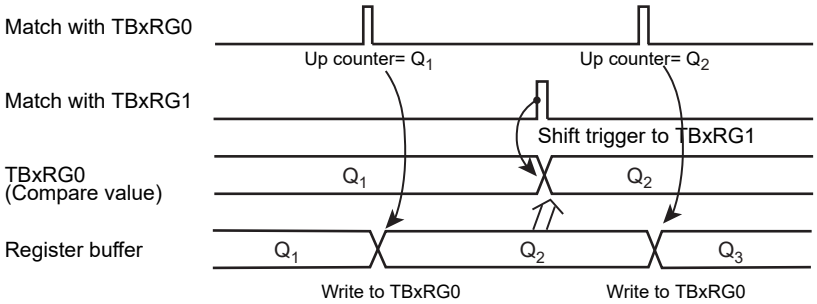


Figure 16-3 Register buffer operation

The block diagram of this mode is shown below.

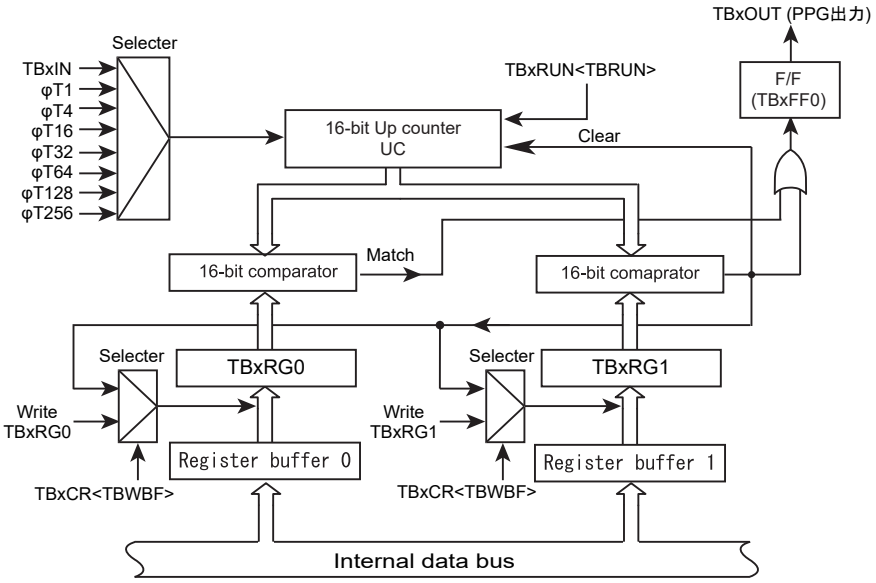


Figure 16-4 Block diagram of 16-bit PPG mode

Each register in the 16-bit PPG output mode should be programmed as listed below.

TBxEN<TBEN> = "1"	Enables TMRBx operation.
TBxRUN<TBPRUN><TBRUN> = "00"	Stops prescaler and counter.
TBxCR<TBWBF> = "1"	Enables double buffer.
TBxRG0 = 0x****	Sets a duty ratio.
TBxRG1 = 0x****	Sets a cycle.
TBxFFCR<TBC1T1><TBC0T1><TBE1T1><TBE0T1> = "0011" TBxFFCR<TBFF0C[1:0]> = "10"	Sets TBxFF0 to invert the signal by detection of a match between TBxRG0 or TBxRG1 and the up-counter; sets TBxFF0 not to reverse the signal by capturing TBxCP0 or TBxCP1.Sets an initial value of TBxFF0 to "0".
TBxMOD<TBCPM[2:0]> = "000" TBxMOD<TBCP> = "1" TBxMOD<TBCLE> = "1" TBxMOD<TBCLK[2:0]> = "***" (*** = "001" to "111")	Changes to prescaler output clock as input clock. Specifies capture function to disable.
Allocate the corresponding port to TBxOUT.	
TBxRUN<TBPRUN><TBRUN> = "11"	Starts prescaler and counter.

Note: *; Optional value

16.5.4 Programmable Pulse Generation (PPG) External Trigger Output Mode

A PPG wave with a short delay time can be output when TMRBx is started by the external trigger count start mode in the PPG (Programmable Pulse Generation) output mode.

The example of one-shot pulse output (with delay) using external trigger count start is shown below.

To start TMRB in the external trigger count start mode, set "1" to TBxCR<CSSEL> and set "0" to TBxCR<TRGSEL> to count up TBxIN on the rising edge while 16-bit counter has been stopped.

TBxRG0 is set the delay time (d) from an external trigger signal. TBxRG1 is set the value (d)+(p) of which the delay time (d) is added to the width (p) of one-shot pulse.

To reverse TBxFF0 when the up-counter matches TBxRG0/1, set "1" to TBxFFCR<TBE1T1>, <TBE0T1>.

To start the up-counter, set "1" to TBxRUN<TBPRUN>, <TBRUN>.

At this time, if the external trigger pulse is input to TBxIN, the up-counter is started on the rising edge of external trigger pulse.

TBxFF0 is reversed when the up-counter counts up to (d). It matches TBxRG0 and then TBxFF0 becomes "High" level.

TBxFF0 is reversed when the up-counter counts up to (d)+(p). It matches TBxRG1 and then TBxFF0 becomes "Low" level.

To avoid changing the level of TBxFF0 by INTTBx that occurs when the up-counter matches TBxRG1, clear TBxFFCR<TBE1T1>, <TBE0T1> to "0" or stop the up-counter by TBxRUN<TBPRUN>, <TBRUN>.

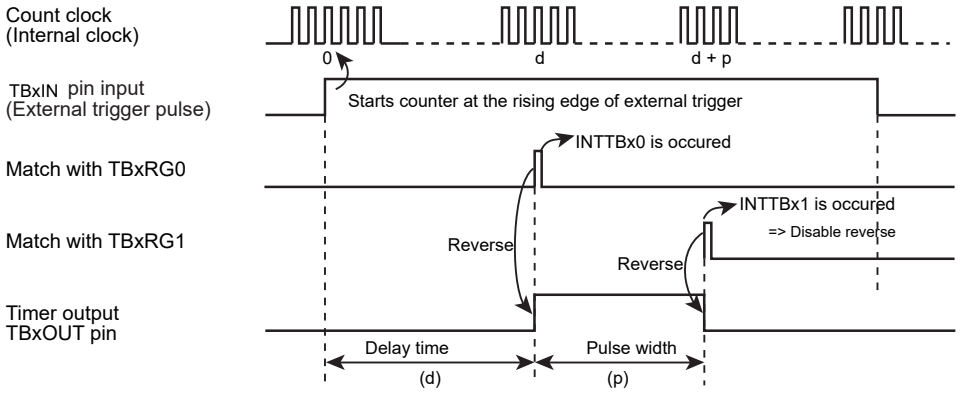


Figure 16-5 One-shot pulse output with delay using external trigger count start

The following shows the case that 2 ms width one-shot pulse is output by triggering TBxIN input on the rising edge after 3 ms has been elapsed. In this example, the source clock is φT1.

[Main process]	
Allocates a corresponding port to	
TBxEN<TBEN> = "1"	Enables TMRBx operation.
TBxRUN<TBPRUN><TBRUN> = "00"	Stops prescaler and counter.
TBxRG0 = 0x****	Sets a counter value. (3 ms/φT1
TBxRG1 = 0x****	Sets a cycle. ((3+2)ms/φT1)
TBxFFCR<TBC1T1><TBC0T1><TBE1T1><TBE0T1> = "0011" TBxFFCR<TBFF0C[1:0]> = "10"	Sets TBxFF0 to invert the signal by detection of a match between TBxRG0 or TBxRG1 and the up-counter; sets TBxFF0 not to reverse by capturing TBxCP0 or TBxCP1. Sets an initial value of TBxFF0 to "0".
TBxMOD<TBCPM[2:0]> = "000" TBxMOD<TBCP> = "1" TBxMOD<TBCLE> = "1" TBxMOD<TBCLK[2:0]> = "001"	Sets a source clock to φT1. Disables the capture function.
Allocates a corresponding port to TBxOUT.	
TBxIM<TBIMOF><TBIM1><TBIM0> = "101"	Masks interrupts except one caused by a match with TBxRG1.
Capture enable set register = 0x********	Permits INTTBx interrupt by setting the corresponding bit to "1".
TBxRUN<TBPRUN><TBRUN> = "11"	Starts prescaler and counter.
[Process in INTTBx interrupt service routine]	
TBxFFCR<TBE1T1><TBE0T1> = "00"	Disables TBxFF0 reverse trigger setting.
TBxRUN<TBPRUN><TBRUN> = "00"	Stops prescaler and counter.

Note: *; Optional value

16.6 Applications Using Capture Function

The capture function can be used in many applications.

The applications are shown below.

1. Frequency measurement
2. Pulse width measurement

16.6.1 Frequency Measurement

The following are examples of measuring a frequency of external clock.

In this section, TMRBm is used as 16-bit interval timer mode and TMRBn is used as 16-bit event counter mode.

To count the up-counter of TMRBn freely by an external clock, set TMnMOD<TBCLK> to "000" and set TBnRUN<TBE1T1><TBE0T1> to "11".

To reverse TBmFF0 when the up-counter of TMRBm matches TBmRG0 and TBmRG1, set TBmFFCR<TBE1T1> <TBE0T1> to "11".

To capture a value of the up-counter into TBnCP0 on the rising edge of TBmFF0 and to capture a value of the up-counter into TBnCP1 on the falling edge of TBmFF0, set TBnMOD<TBCPM[2:0]> to "011".

Set the number of counting external clocks to TBmRG0/1 and start TMRBm.

When a value the up-counter of TMRBm matches TBmRG0, TBmFF0 rises and a value of the up-counter of TMRBn is captured into TBnCP0. When the up-counter of TMRBm matches TBmRG1, TBmFF0 falls and a value of the up-counter of TMRBn is captured into TBnCP1.

A frequency can be measured from (TBnCP1 - TBnCP0) ÷ (TBmRG1 - TBmRG0) using INTTBm.

For example, the difference between TBmRG1 and TBmRG0 is 0.5 s and the difference between TBnCP1 and TBnCP0 is 100, the frequency is 200 Hz (100 ÷ 0.5 s = 200Hz)

Depending on the changing timing of TBmFF0, the result of TBnCP1 - TBnCP0 will be minus. Correct the value if TBnCP1 - TBnCP0 is minus.

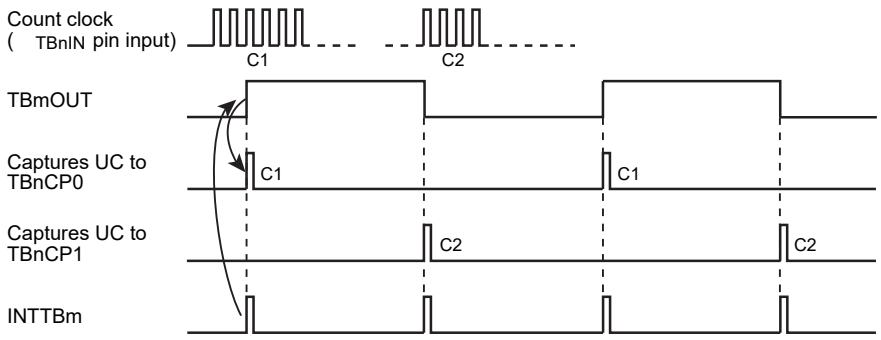


Figure 16-6 Frequency measurement

The following shows in the case that the measured pulse is input to TBnIN. In this example, the source clock is φT1.

[Main process] TBmFF0 capture setting	
Allocates a corresponding port to	
TBmEN<TBEN> = "1"	Enables TMRBm operation.
TBmRUN<TBPRUN><TBRUN> = "00"	Stops prescaler and counter.
TBnEN<TBEN> = "1"	Enables TMRBn operation.
TBnRUN<TBPRUN><TBRUN> = "00"	Stops prescaler and counter.
TBmCR<TBWBF> = "1"	Enables double buffer.
TBmRG0 = 0x****	Sets an external clock measurement time 1.
TBmRG1 = 0x****	Sets an external clock measurement time 2.
TBmFFCR<TBC1T1><TBC0T1><TBE1T1><TBE0T1> = "0011" TBmFFCR<TBFF0C[1:0]> = "10"	Sets TBmFF0 to invert the signal by detection of a match between TBmRG0 or TBmRG1 and the up-counter. Sets TBmFF0 not to reverse the signal by capturing TBmCP0 or TBmCP1. Sets an initial value of TBmFF0 to "0".
TBnMOD<TBCPM[2:0]> = "011" TBnMOD<TBCP> = "1" TBnMOD<TBCLE> = "0" TBnMOD<TBCLK[2:0]> = "000"	Captures a value of up-counter on rising/falling edge of TBmFF0. Disables up-counter clear. Sets an input clock to .
TBmIM<TBIMOF><TBIM1><TBIM0> = "101"	Masks interrupts except a match with TBmRG1.
Capture enable set register = 0x********	Sets "1" to the bit corresponding to INTTBm to enable the interrupt.
TBmRUN<TBPRUN><TBRUN> = "11"	Starts prescaler and counter.
TBnRUN<TBPRUN><TBRUN> = "11"	Starts prescaler and counter.
[Process in INTTBm interrupt service routine]	
TBmFFCR<TBE1T1><TBE0T1> = "00"	Disables TBmFF0 reverse trigger setting
Interrupt enable clear register = 0x********	Sets "1" to the bit corresponding to INTTBm to disable the interrupt.
Reads TBnCP0/1 and calculate a frequency.	

Note:m, n ; Optional channel number, *; Optional value

16.6.2 Pulse Width Measurement

"High" level width of the external pulse can be measured.

To capture a value of the up-counter into TBxCP0 on the rising edge of TBxIN and to capture a value of the up-counter into TBxCP1 on falling edge of TBxIN, set TBxMOD<TBCPM> to "010".

Enables INTTBxCAP1 interrupt.

Enables TMRBx operation.

If rising edge of external pulse is input to TBxIN, a value of the up-counter is captured into TBxCP0. If falling edge of external pulse is input to TBxIN, a value of the up-counter is captured into TBxCP1 and INTTBxCAP1 interrupt occurs.

The "High" level width of the external pulse can be calculated by multiplying the difference between TBxCP0 and TBxCP1 by the clock cycle of a prescaler output 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.

If a pulse width is beyond the maximum count time of the up-counter, make a correction.

In addition, "Low" level width of an external pulse can be measured.

To calculate the "Low" level width, enable INTTBxCAP0 interrupt and multiply the time difference between first C2 and second C1 in "Figure 16-7 Pulse width measurement" at the second time of INTTBxCAP0 by the cycle of the prescaler output clock.

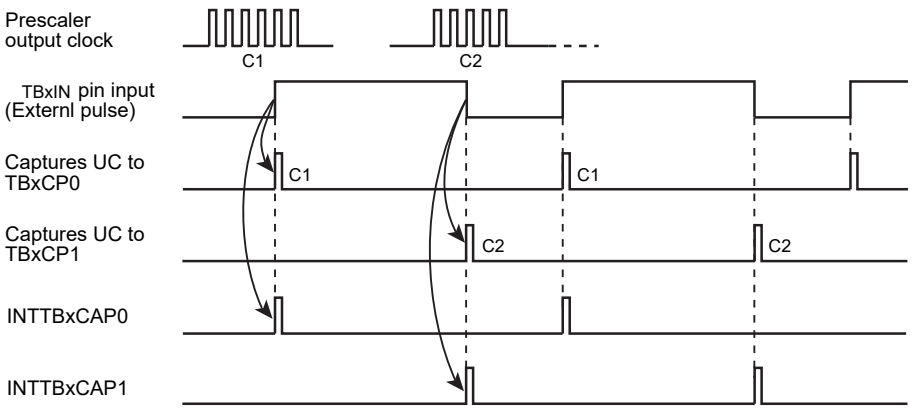


Figure 16-7 Pulse width measurement

The following describes the example of the measurement of "High" level width of the external pulse into TBxIN. In this example, the source clock is φT1.

[Main process] capture setting	
Allocates a corresponding port to .	
TBxEN<TBEN> = "1"	Enables TMRBx operation.
TBxRUN<TBPRUN><TBRUN> = "00"	Stops prescaler and counter.
TBxFFCR<TBC1T1><TBC0T1><TBE1T1><TBE0T1> = "0000" TBxFFCR<TBFF0C[1:0]> = "10"	Sets TBxFF0 not to invert the signal if TMRBx detects a match between TBxRG0 or TBxRG1 and the up-counter, or when capturing TBxCP0 or TBxCP1. Sets an initial value of TBxFF0 to "0".
TBxMOD<TBCPM[2:0]> = "010" TBxMOD<TBCP> = "1" TBxMOD<TBCLE> = "0" TBxMOD<TBCLK[2:0]> = "001"	Sets the source clock to φT1. Capture a value of up-counter on rising edge of pin into TBxCP0; captures a value of up-counter on falling edge of pin into TBxCP1.
Interrupt enable set register = 0x********	Set "1" to the bit corresponding to INTTBxCAP1 interrupt to enable the interrupt.
TBxRUN<TBPRUN><TBRUN> = "11"	Starts prescaler and counter.
[Process in INTTBxCAP1 interrupt service routine] Calculates "High" level width	
TBxFFCR<TBE1T1><TBE0T1> = "00"	Disables TBxFF0 reverse trigger setting
Interrupt enable clear register = 0x********	Set "1" to the bit corresponding to INTTBxCAP1 interrupt to disable the interrupt.
Reads a value of TBxRG0/1 and calculates "High" level width.	

Note:*; Optional value

17. 16-bit Multi-Purpose Timer (MPT)

17.1 Outline

The MPT provides the operational modes as follows.

1. Timer mode
   - 16-bit interval timer mode
   - 16-bit event counter mode
   - 16-bit programmable rectangular waveform output (PPG, one output) mode
   - Pulse width measurement (capture)

2. IGBT mode
   - 16-bit programmable rectangular waveform output (PPG, two outputs) mode
   - External trigger start
   - Cycle match detection
   - Emergency stop function
   - Synchronous start mode

17.2 Block Diagram

The MPT consists of three modules including a timer and IGBT.

Each module is switched by registers.

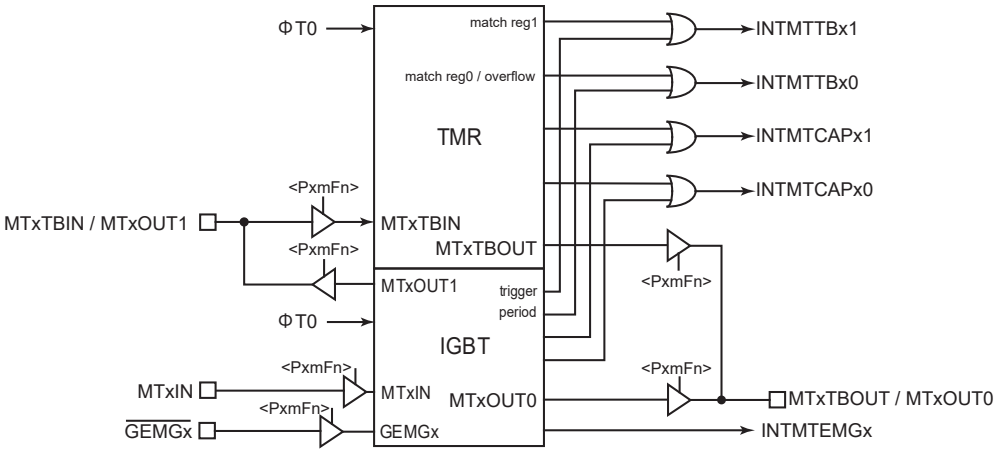


Figure 17-1 Block Diagram of MPTx

17.3 Operation Description of Timer Mode

17.3.1 Block Diagram

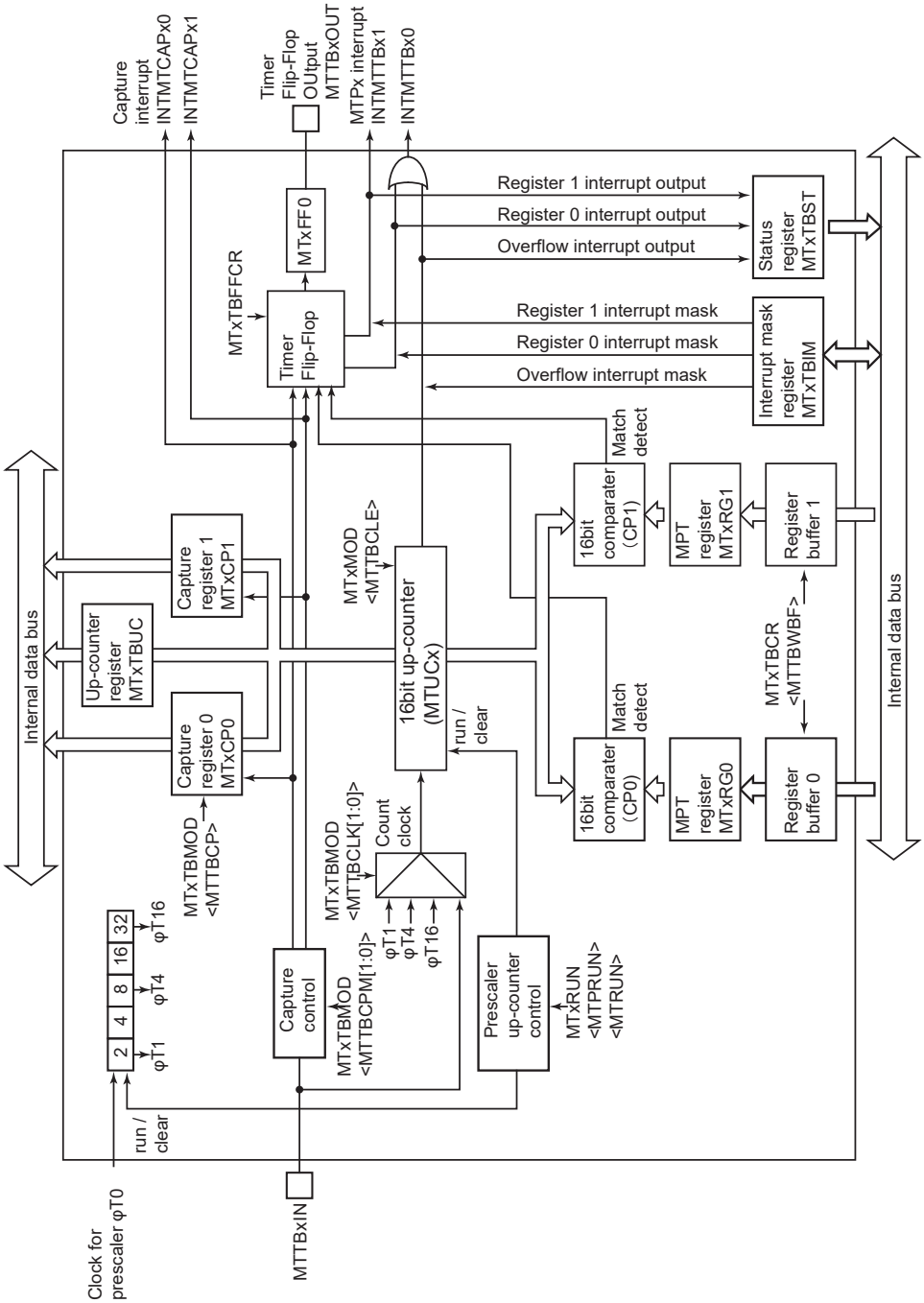


Figure 17-2 Block Diagram of Timer Mode

17.3.2 Registers categorized by timer mode channel

The following table shows control registers and addresses.

For the base address, refer to "Address lists of peripheral functions" of "Memory Map" chapter.

Register name		Address (Base+)
MPT enable register	MTxEN	0x0000
MPT RUN register	MTxRUN	0x0004
MPT control register	MTxTBCR	0x0008
MPT mode register	MTxTBMOD	0x000C
MPT flip-flop control register	MTxTBFFCR	0x0010
MPT status register	MTxTBST	0x0014
MPT interrupt mask register	MTxTBIM	0x0018
MPT up-counter register	MTxTBUC	0x001C
MPT register	MTxRG0	0x0020
MPT register	MTxRG1	0x0024
MPT capture register	MTxCP0	0x0028
MPT capture register	MTxCP1	0x002C

17.3.3 MTxEN (MPT 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	MTEN	MTHALT	-	-	-	-	-	MTMODE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7	MTEN	R/W	Specifies MPT operation. 0: Disable 1: Enable When MTEN is disabled, feeding clock to other registers of MPT module is stopped, so that power consumption can be reduced. (Read or write to other registers cannot be done.)
6	MTHALT	R/W	Specifies MPT operation when core halts (debug break). [TMR function] 0: Clock stopping operation is disabled while core halts. 1: Clock stopping operation is enabled while core halts. [IGBT function] 0: Not control clock stopping operation and MTxOUT0/MTxOUT1 output. 1: Clock stopping operation is enabled while core halt. It controls MTxOUT0/MTxOUT1 output according to the MTxIGEMGCR<IGEMGOC> setting.
5-1	-	R	Read as "0".
0	MTMODE	R/W	Specifies operation modes 0: Timer mode 1: IGBT mode

Note: **When MPT is used, MPT operation is enabled (<MTEN>="1")before each register of MPT module is set. Even if MPT operation is disabled after MPT is stopped, each register setting is maintained.**

17.3.4 MTxRUN (MPT 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	-	-	-	-	-	MTPRUN	-	MTRUN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	-	R	Read as "0".
2	MTPRUN	R/W	Controls MPT prescaler operation 0: Stops prescaler operation. Prescaler is cleared to "0". 1: Starts prescaler operation.
1	-	R	Read as "0".
0	MTRUN	R/W	Controls MPT counting operation 0: Stops counting operation. Counter is cleared to "0". 1: Starts counting operation.

17.3.5 MTxTBCR (MPT 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	MTTBWBF	-	-	-	MTI2TB	-	MTTB TRGSEL	MTTBCSSEL
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7	MTTBWBF	R/W	Specifies double buffer to enable/disable 0:Disabled 1:Enabled
6-5	-	R/W	Write "0".
4	-	R	Read as "0".
3	MTI2TB	R/W	Controls clock operation to star/stop in IDLE mode 0:Stop 1:Start
2	-	R	Read as "0".
1	MTTBTRGSEL	R/W	Selects rising or falling edge of external trigger. 0:Rising edge 1:Falling edge
0	MTTBCSSEL	R/W	Selects how to start counting 0:Soft start 1:External trigger

Note: **Do not modify MTxTBCR during timer in operation (MTxRUN<MTRUN>="1").**

17.3.6 MTxTBMOD (MPT 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	-	MTTBRSWR	MTTBCP	MTTBCPM		MTTBCLE	MTTBCLK	
After reset	0	0	1	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	-	R	Read as "0".
6	MTTBRSWR	R/W	Controls the write timing to timer register 0 and 1 when double-buffer is used. 0: If either timer register 0 or timer register 1 is prepared to be written, one register can be written at a time. 1: If both timer register 0 and timer register 1 are not prepared, timer register cannot be written.
5	MTTBCP	W	Controls software capture 0: Capture count values to the capture register 0 (MTxCP0) 1: Don't care
4-3	MTTBCPM[1:0]	R/W	Sets capture timing 00: Capture is disabled. 01: At the rising edge of MTxTBIN input, counter values are captured to the capture register 0 (MTxCP0) 10: At the rising edge of MTxTBIN input, counter values are captured to the capture register 0 (MTxCP0). At the falling edge of MTxTBIN input, counter values are captured to the capture register 1 (MTxCP1). 11: Capture is disabled.
2	MTTBCLE	R/W	Clear MPT up-counter 0: Clear is disabled. 1: Clear MPT up-counter by matching with timer register 1 (MTxRG1)
1-0	MTTBCLK[1:0]	R/W	Selects timer count clock of MPT 00: MTxTBIN input 01: φT1 10: φT4 11: φT16

Note 1: **MTxTBMOD<MTTBCP> reads as "1".**

Note 2: **Do not modify MTxTBMOD during timer in operation (MTxRUN<MTRUN>="1").**

17.3.7 MTxTBFFCR (MPT 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	-	-	MTTBC1T1	MTTBC0T1	MTTBE1T1	MTTBE0T1	MTTBFF0C	
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 "11".
5	MTTBC1T1	R/W	Controls timer flip-flop reverse when up-counter values are captured to the capture register 1 (MTxCP1). 0: MTxFF0 does not reverse. 1: MTxFF0 reverses.
4	MTTBC0T1	R/W	Controls timer flip-flop reverse when up-counter values are captured to the capture register 1 (MTxCP0) 0: MTxFF0 does not reverse. 1: MTxFF0 reverses.
3	MTTBE1T1	R/W	Controls timer flip-flop reverse when up-counter values and the timer register 1 (MTxRG1) are matched. 0: MTxFF0 does not reverse. 1: MTxFF0 reverses.
2	MTTBE0T1	R/W	Controls timer flip-flop reverse when up-counter values and the timer register 1 (MTxRG0) are matched. 0: MTxFF0 does not reverse. 1: MTxFF0 reverses.
1-0	MTTBFF0C	R/W	Controls timer flip-flop 00: Reverses a value of MTxFF0. 01: Sets "1" to MTxFF0. 10: Sets "0" MTxFF0 to clear. 11: Don't care. Read as "11".

Note: Do not modify **MTxTBFFCR during timer in operation (MTxRUN<MTRUN>="1").**

17.3.8 MTxTBST (MPT 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	-	-	-	-	-	MTTBINT TBOF	MTTBINTTB1	MTTBINTTB0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	-	R	Read as "0".
2	MTTBINTTBOF	R	Indicates the status of up-counter overflow interrupt generation. 0: No interrupt generation 1: Interrupt generation
1	MTTBINTTB1	R	Indicates the interrupt generation by matching with timer register 1 (MTxRG1) 0: No interrupt generation 1: Interrupt generation
0	MTTBINTTB0	R	Indicates the interrupt generation by matching with timer register 0 (MTxRG0) 0: No interrupt generation 1: Interrupt generation

Note: **Once any interrupt generates, corresponding flag in MTxTBST register is set to notify CPU of an interrupt generation. If MTxTBST register is read, the flag is cleared to "0".**

17.3.9 MTxTBIM (MPT 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	-	-	-	-	-	MTTBIMOF	MTTBIM1	MTTBIM0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	-	R	Read as "0".
2	MTTBIMOF	R/W	Controls up-counter overflow interrupt to mask 0: Not mask interrupt 1: Masks interrupt
1	MTTBIM1	R/W	Controls to mask the interrupt when the match between timer register 1 (MTxRG1) and up-counter. 0: Not mask interrupt 1: Masks interrupt
0	MTTBIM0	R/W	Controls to mask the interrupt when the match between timer register 0 (MTxRG0) and up-counter. 0: Not mask interrupt 1: Masks interrupt

Note: MTxTBST reflects interrupt requests even if **MTxTBIM masks interrupts.**

17.3.10 MTxTBUC (MPT read 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	MTUC							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MTUC							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	MTUC[15:0]	R	Captures a value by reading up-counter out. If MTxTBUC is read during the counter operation, the current value of up-counter will be captured.

17.3.11 MTxRG0/MTxRG1 (MPT timer register)

MTxRG0

	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	MTRG0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MTRG0							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	MTRG0[15:0]	R/W	Timer count value When up-counter values match with MTRG0[15:0], match detection interrupt (INTMTTBx0) occurs. Also, MTxTBOUT can be reversed when matching,

Note 1: **Use half word access or word access.**

Note 2: **Set to the condition of 0x0000<MTxRG0 < MTxRG1≤0xFFFF.**

MTxRG1

	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	MTRG1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MTRG1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	MTRG1[15:0]	R/W	Timer count value When up-counter values match with MTRG1[15:0], match detection interrupt (INTMTTBx1) occurs. Also, MTxTBOUT can be reversed when matching.

Note 1: **Use half word access or word access.**

Note 2: **Set to the condition of 0x0000<MTxRG0 < MTxRG1≤0xFFFF.**

17.3.12 MTxCP0 /MTxCP1 (MPT capture register)

MTxCP0

	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	MTCP0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MTCP0							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	MTCP0[15:0]	R	Read captured up-counter values.

Note: **During the timer stopping, a value of timer counter (MTUCx) cannot be read. When the timer stops, a value previously captured is held and the value can be read.**

MTxCP1

	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	MTCP1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MTCP1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	MTCP1[15:0]	R	Read captured up-counter values.

Note: **During the timer stopping, a value of timer counter (MTUCx) cannot be read. When the timer stops, a value previously captured is held and the value can be read.**

17.3.13 Operational Description categorized by circuit

17.3.13.1 Prescaler

This 4-bit prescaler generates the source clock for up-counter MTUCx.

Input clock φT0 to the prescaler is chosen among fperiph/1, fperiph/2, fperiph/4, fperiph/8, fperiph/16 and fperiph/32 by specifying with CGSYSCR<PRCK[2:0]>. This peripheral clock (fperiph) is either fgear specified with CG SYSCR<FPSEL> or fc that is pre-divided clock gear.

Prescaler is set to enable/disable with MTxRUN<MTPRUN>. When MTxRUN<MTPRUN> is set to "1", counting starts. When MTxRUN<MTPRUN> is set to "0", the counter is stopped and cleared.

17.3.13.2 Up-Counter (MTUC0)

This counter is a 16-bit binary counter.

- Source clock

  The source clock can be set with MTxTBMOD<MTTBCLK[1:0]>.

  Prescaler output clock can choose among φT1, φT4, φT16 or external clock of MTxTBIN pin.

- Start/Stop counter operation

  Counter operation is set with MTxRUN<MTRUN>. When <MTRUN>="1" is set, counter operation starts. When <MTRUN>="1" is set, the counter is stopped and cleared at the same time.

  When a value of up-counter MTUCx detects the match with a setting value of timer register MTxRG0/MTxRG1, INTMTTB0x or INTMTTB1x occurs.

- Counter clear timing

  1. Comparing a match

     If MTxTBMOD<MTTBCLE>="1" is set, the counter is cleared when comparing matches with MTxRG1.

     If MTxTBMOD<MTTBCLE>="0" is set, the counter becomes a free-running counter.

  2. Counter stopping

     If MTxRUN<MTRUN>="0" is set, the counter is stopped and cleared.

- Overflow of the counter

  If MTUCx is overflowed, an overflow interrupt INTMTTB0x occurs.

17.3.13.3 Timer Register (MTxRG0, MTxRG1)

Timer register sets a values to compare with up-counter MTUCx. Comparator compares a value of timer register with a value of up-counter. If these two are matched, the match detection signal is output.

- Structure

  In the timer register, MTxRG0/1 is double-buffering structure paired with register buffer.

  Double-buffer is set to enable/disable with MTxTBCR<MTTBWBF>. If <MTTBWBF>="0" is set, double-buffer is disabled. If <MTTBWBF>="1" is set, double-buffer is enabled.

  While double-buffer is enabled, data transfer is taken place from register buffer 0 to timer register MTxRG0/1 when MTUCx matches with MTxRG1.

- Initial state

  After reset, MTxRG0 and MTxRG1 are undefined and double-buffer is disabled.

- How to set

  1. If double-buffer is not used.

     Use half-word access or word access

  2. If double-buffer is used.

     MTxRG0 and 1, and register buffer 0 and 1 are assigned to the same address respectively.

     When <MTTBWBF> is "0", MTxRG0 and 1 and each register buffer are written the same value. When <MTTBWBF> is "1", only corresponding register buffer is written data. Thus when writing the initial value to timer register, set as follows; firstly register buffer is disabled, secondly timer register is written data, thirdly <MTTBWBF> is set to "1". Finally next data is written to register buffer.

17.3.13.4 Capture Control

This circuit controls the timing when a value of up-counter MTUCx is latched by capture register MTxCP0/MTxCP1. This latch timing is set with MTxTBMOD<MTTBCPM[1:0]>.

Also the timing is controlled by software. Every time MTxTBMOD<MTTBCP> is set to "0", a value of MTUCx is captured to the capture register MTxCP0 at the time. Note that prescaler must be set to RUN status (MTxRUN<MTPRUN> "1").

17.3.13.5 Capture Register (MTxCAP0、MTxCAP1)

This register captures a value of up-counter MTUCx.

17.3.13.6 Up-counter Capture Register (MTxTBUC)

If MTxTBUC register is read during the counter operation, the current value of up-counter will be captured and the value will be read. The value captured at the end is held while the counter is stopping.

17.3.13.7 Comparators (CP0, CP1)

This comparator detects the match comparing a value of up-counter (MTUCx) with a setting value of timer register MTxRG0/MTxRG1. If these values are matched, INTMTTBx0 or INTMTTBx1 occurs.

17.3.13.8 Timer Flip-flop (MTxFF0)

Timer flip-flop circuit (MTxFF0) reverses by a match signal from comparators or a latch signal to the capture register. This reverse is enabled/disabled with MTxTBFFCR<MTTBC1T1, MTTBC0T1, MTTBE1T1, MTTBE0T1>.

After reset, a value of MTxFF0 is undefined. If MTxTBFFCR<MTTBFF0C[1:0]> is set to "00", the reverse is enabled. If MTxTBFFCR<MTTBFF0C[1:0]> is set to "01", MTxFF0 is set to "1". MTxTBFFCR<MTTBFF0C[1:0]> is set to "10", MTxFF0 is set to "0" to clear.

A value of MTxFF0 can be output to timer output pin MTxTBOUT. If timer output is used, port related registers (PxCR and PxFR) must be set beforehand.

17.3.13.9 Capture Interrupts (INTMTCAPx0, INTMTCAPx1)

Capture interrupts (INTMTCAPx0 and INTMTCAPx1) occur respectively at the timing when data is latched to each capture register (MTxCP0 and MTxCP1). Interrupt setting is set by CPU.

17.4 Operational Description in IGBT mode

17.4.1 Block Diagram

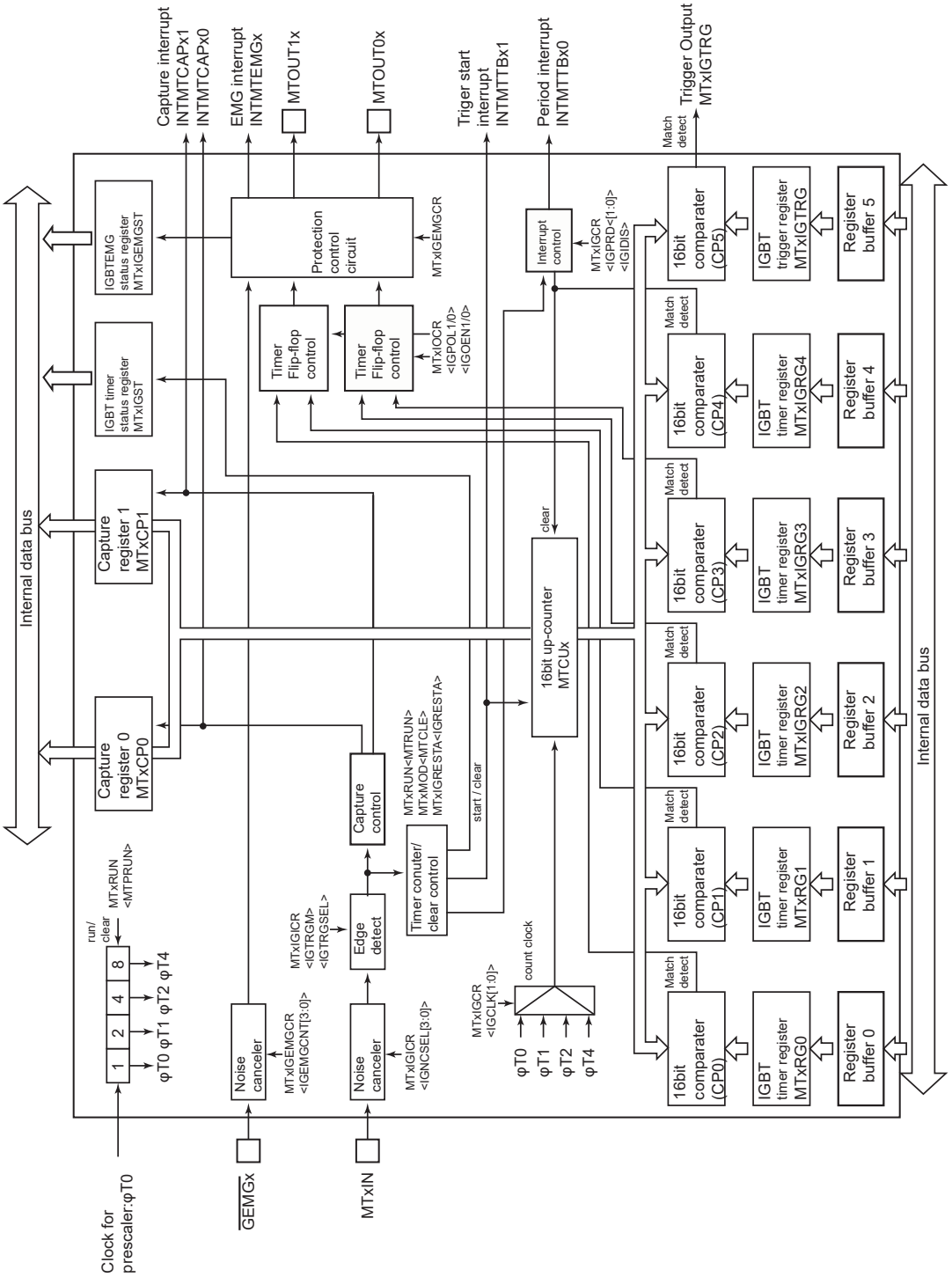


Figure 17-3 Block Diagram in IGBT mode

17.4.2 Registers in IGBT mode categorized by channel

The following table shows control registers and addresses.

For the base address, refer to "Address lists of peripheral functions" of "Memory Map" chapter .

Register name(x=0 to 3)		Address (Base+)
MPT enable register	MTxEN	0x0000
MPT RUN register	MTxRUN	0x0004
MPT register 0	MTxRG0	0x0020
MPT register 1	MTxRG1	0x0024
MPT capture register 0	MTxCP0	0x0028
MPT capture register 1	MTxCP1	0x002C
IGBT control register	MTxIGCR	0x0030
IGBT timer restart register	MTxIGRESTA	0x0034
IGBT timer status register	MTxIGST	0x0038
IGBT input control register	MTxIGICR	0x003C
IGBT output control register	MTxIGOCR	0x0040
IGBT timer register 2	MTxIGRG2	0x0044
IGBT timer register 3	MTxIGRG3	0x0048
IGBT timer register 4	MTxIGRG4	0x004C
IGBT EMG control register	MTxIGEMGCR	0x0050
IGBT EMG status register	MTxIGEMGST	0x0054
IGBT trigger register	MTxIGTRG	0x0058

17.4.3 MTxEN (MPT 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	MTEN	MTHALT	-	-	-	-	-	MTMODE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7	MTEN	R/W	Specifies MPT operation. 0: Disabled 1: Enabled When MTEN is disabled, feeding clock to other registers of MPT module is stopped, so that power consumption can be reduced. (Read or write to other registers cannot be done.)
6	MTHALT	R/W	Specifies MPT operation when core halts (debug break). [TMR function] 0: Clock stopping operation is disabled while core halts. 1: Clock stopping operation is enabled while core halts. [IGBT function] 0: Not control clock stopping operation and MTxOUT0/MTxOUT1 output. 1: Clock stopping operation is enabled while core halt. It controls MTxOUT0/MTxOUT1 output according to the MTxIGEMGCR<IGEMGOC> setting.
5-1	-	R	Read as "0".
0	MTMODE	R/W	Specifies operation mode. 0: Timer mode 1: IGBT mode

Note: When MPT is used, MPT operation is enabled (<MTEN>="1")before each register of MPT module is set. If MPT operation is disabled after MPT is stopped, each register setting is maintained.

17.4.4 MTxRUN (MPT 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	-	-	-	-	-	MTPRUN	-	MTRUN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	-	R	Read as "0".
2	MTPRUN	R/W	Controls MPT prescaler operation 0: Stops prescaler operation. Prescaler is cleared to "0". 1: Starts prescaler operation.
1	-	R	Read as "0".
0	MTRUN	R/W	Controls MPT counting operation 0: Stops counting operation. Counter is cleared to "0". 1: Starts counting operation.

17.4.5 MTxRG0/MTxRG1 (MPT timer register)

MTxRG0

	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	MTRG0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MTRG0							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	MTRG0[15:0]	R/W	Timer count value When up-counter values match with MTRG0[15:0], MTxOUT0 becomes active level.

Note 1: **Use half word access or word access.**

Note 2: **Set to the condition of 0x0000<MTxRG0 < MTxRG1≤MTxIRG4≤0xFFFF.**

MTxRG1

	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	MTRG1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MTRG1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	MTRG1[15:0]	R/W	Timer count value When up-counter values match with MTRG1[15:0], MTxOUT0 becomes inactive level.

Note 1: **Use half word access or word access.**

Note 2: **Set to the condition of 0x0000<MTxRG0 < MTxRG1<≤MTxIRG4≤0xFFFF.**

17.4.6 MTxCP0 /MTxCP1 (MPT capture register)

MTxCP0

	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	MTCP0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MTCP0							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	MTCP0[15:0]	R	Read captured up-counter values.

MTxCP1

	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	MTCP1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MTCP1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	MTCP1[15:0]	R	Read captured up-counter values.

Note: **During the timer stopping, a value of timer counter (MTUCx) cannot be read. When the timer stops, a value previously captured is held and the value can be read.**

17.4.7 MTxIGCR (IGBT 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	-	-	-	-	-	IGIDIS	IGPRD	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	IGCLSYNC	IGSNGL	IGSTP		IGSTA		IGCLK	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-11	-	R	Read as "0".
10	IGIDIS	R/W	Controls an interrupt when commands start 0: Enabled 1: Disabled
9-8	IGPRD[1:0]	R/W	Chooses an interrupt cycle 00: Every one cycle 01: Every two cycles 10: Every four cycles 11: Reserved
7	IGCLSYNC	R/W	Clears the up-counter 0: individual (unit of channel) 1: synchronous
6	IGSNGL	R/W	Chooses IGBT operation 0: Continuous operation 1: Single operation
5-4	IGSTP[1:0]	R/W	Chooses stopping status [Master channel] 00: Initial output status and counter immediately stops to clear 01: Sustains output status and counter immediately stops to clear 10: After cycle time has elapsed then counter stops to clear 11: Reserved [Slave channel] 00: Stops in the output initial state 01: Stops maintaining output condition 10: Reserved 11: Reserved
3-2	IGSTA[1:0]	R/W	Chooses start mode 00: Command start and trigger capture 01: Command start and trigger start 10: Trigger start 11: Synchronous start (sets only slave channels)
1-0	IGCLK[1:0]	R/W	Chooses a source clock of IGBT 00: φT0 01: φT1 10: φT2 11: φT4

Note 1: **Do not modify MTxIGCR during timer in operation (MTxRUN<MTRUN>="1").**

Note 2: **When the counter stops after specified cycle time has elapsed, or counter is stopped with (MTxIGCR<IGSTP>="10") and cleared with MTxRUN<MTRUN>, check if the timer is stopped by cycle interrupt generation. Then change the setting and restart.**

17.4.8 MTxIGRESTA (IGBT timer restart 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	-	-	-	-	-	-	-	IGRESTA
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	-	R	Read as "0".
0	IGRESTA	W	Controls counting restart 0: Don't care 1: Restart Read as "0".

Note: **If MTxIGRESTA<IGRESTA> is set to "1" during timer in operation, timer counter can be cleared and restart. Please check the status of output waveform before setting is changed.**

17.4.9 MTxIGST (IGBT timer 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	-	-	-	-	-	-	-	IGST
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	-	R	Read as "0".
0	IGST	R	Counter operation status 0: Stop 1: Operating

17.4.10 MTxIGICR (IGBT 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	IGTRGM	IGTRGSEL	-	-	IGNCSEL			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7	IGTRGM	R/W	Controls trigger edge accept mode 0: Always accept 1: Acceptance is disabled during active level
6	IGTRGSEL	R/W	Chooses start trigger edges and its active levels. 0: Rising edge start and active level is "High". 1: Falling edge start and active level is "Low".
5-4	-	R	Read as "0".
3-0	IGNCSEL[3:0]	R/W	Trigger input noise elimination time selection Noise elimination time is calculated with the following formula: IGNCSEL[3:0] × 16 / fsys 0000: Noise filter is not used. 0001: Noise elimination time 16 / fsys [s] 0010: Noise elimination time 32 / fsys [s] 0011: Noise elimination time 48 / fsys [s] 0100: Noise elimination time 64 / fsys [s] 0101: Noise elimination time 80 / fsys [s] 0110: Noise elimination time 96 / fsys [s] 0111: Noise elimination time 112 / fsys [s] 1000: Noise elimination time 128 / fsys [s] 1001: Noise elimination time 144 / fsys [s] 1010: Noise elimination time 160 / fsys [s] 1011: Noise elimination time 176 / fsys [s] 1100: Noise elimination time 192 / fsys [s] 1101: Noise elimination time 208 / fsys [s] 1110: Noise elimination time 224 / fsys [s] 1111: Noise elimination time 240 / fsys [s]

Note 1: **Do not modify MTxIGCR during timer in operation (MTxRUN<MTRUN>="1").**

Note 2: **When MTxGCR<IGNCSEL[3:0]> is used, EMG protection circuit must be disabled (MTxIGEMGCR<IGEM-GEN>="0").**

Note 3: **When MTxIGCR<IGNCSEL[3:0]> is changed, specified noise elimination time or more is required to start the timer with (MTxRUN<MTRUN>="1").**

Note 4: **When the synchronous start is set (MTxIGCR<IGSTA[1:0]>="11"), MTxIGICR<IGTRGM><IGTRGSEL> bit on the slave channels are disabled.**

17.4.11 MTxIGOCR (IGBT 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	-	-	IGPOL1	IGPOL0	-	-	IGOEN1	IGOEN0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	-	R	Read as "0".
5	IGPOL1	R/W	Indicates the initial state of MTxOUT1 0: Low 1: High
4	IGPOL0	R/W	Indicates the initial state of MTxOUT0 0: Low 1: High
3-2	-	R	Read as "0".
1	IGOEN1	R/W	Controls MTxOUT1 output 0: Disable 1: Enable
0	IGOEN0	R/W	Control MTxOUT0 output 0: Disabled 1: Enabled

Note: **MTxOUT0/MTxOUT1 output is changing according to a content of IGBT output control register (MTxIGOCR) regardless of timer in operation/stopping. Check the operation status before MTxGOCR is set.**

17.4.12 MTxIGRG2 (IGBT timer 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	IGRG2							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	IGRG2							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	IGRG2[15:0]	R/W	Timer count value When up-counter matches with IGRG2[15:0], MTxOUT1 becomes active level.

Note 1: **Use half-word access or word access.**
Note 2: **Set the value to the condition of 0x0000<MTxIGRG2<MTxIGRG3≤MTxIGRG4≤0xFFFF.**

17.4.13 MTxIGRG3 (IGBT timer 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	IGRG3							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	IGRG3							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	IGRG3[15:0]	R/W	Timer count value When up-counter matches with IGRG3[15:0], MTxOUT1 becomes inactive level.

Note 1: **Use half-word access or word access.**
Note 2: **Set the value to the condition of 0x0000<MTxIGRG2<MTxIGRG3≤MTxIGRG4≤0xFFFF.**

17.4.14 MTxIGRG4 (IGBT timer 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	IGRG4							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	IGRG4							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	IGRG4[15:0]	R/W	Timer count value Specifies IGBT mode cycle

Note 1: **Use half-word access or word access.**
Note 2: **Set the value to the condition of 0x0000<MTxRG0<MTxRG1≤MTxIGRG4≤0xFFFF.**
Note 3: **Set the value to the condition of 0x0000<MTxIGRG2<MTxIGRG3≤MTxIGRG4≤0xFFFF.**

17.4.15 MTxIGEMGCR (IGBT EMG 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	IGEMGCNT				-	IGEMGRS	IGEMGOC	IGEMGEN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7-4	IGEMGCNT[3:0]	R/W	GEMG input noise elimination time selection Noise elimination time is calculated with the following formula: IGEMGCNT[3:0]×16 / fsys 0000: Noise filter is not used. 0001: Input noise elimination time 16 / fsys [s] 0010: Input noise elimination time 32 / fsys [s] 0011: Input noise elimination time 48 / fsys [s] 0100: Input noise elimination time 64 / fsys [s] 0101: Input noise elimination time 80 / fsys [s] 0110: Input noise elimination time 96 / fsys [s] 0111: Input noise elimination time 112 / fsys [s] 1000: Input noise elimination time 128 / fsys [s] 1001: Input noise elimination time 144 / fsys [s] 1010: Input noise elimination time 160 / fsys [s] 1011: Input noise elimination time 176 / fsys [s] 1100: Input noise elimination time 192 / fsys [s] 1101: Input noise elimination time 208 / fsys [s] 1110: Input noise elimination time 224 / fsys [s] 1111: Input noise elimination time 240 / fsys [s]
3	-	R	Read as "0".
2	IGEMGRS	W	Return from EMG protection status 0: Don't care 1: Returned (automatically cleared to "0".) (Read as "0".)
1	IGEMGOC	R/W	Set the polarity of MTxOUT0/MTxOUT1 at EMG protection 0: Inactive level 1: High-impedance
0	IGEMGEN	R/W	Controls EMG protection circuit operation 0: Disable 1: Enable

Note: **Do not modify MTxIGEMGCR during timer in operation (MTxRUN<MTRUN>="1").**

17.4.16 MTxIGEMGST (IGBT EMG 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	-	-	-	-	-	-	IGEMGIN	IGEMGST
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	-	R	Read as "0".
1	IGEMGIN	R	EMG input status after noise elimination 0: Low 1: High
0	IGEMGST	R	EMG protection status 0: Normal operation 1: During in protection Read value indicates EMG protection status

17.4.17 MTxIGTRG (IGBT trigger 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	IGTRG							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	IGTRG							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	R	Read as "0".
15-0	IGTRG[15:0]	R/W	Timer count value A trigger (MTxIGTRG) is output when the up-counter matches IGTRG[15:0].

Note 1: **Use half-word access or word access.**

Note 2: **Set the value to the condition of 0x0000<MTxIGTRG≤MTxIGRG4≤0xFFFF.**

17.4.18 Operation Description categorized by circuit

17.4.18.1 Prescaler

This 4-bit prescaler generates the source clock for up-counter MTUCx.

Input clock φT0 to the prescaler is chosen among fperiph/1, fperiph/2, fperiph/4, fperiph/8, fperiph/16 and fperiph/32 by specifying with CGSYSCR<PRCK[2:0]>. This peripheral clock (fperiph) is either fgear specified with CGSYSCR<FPSEL> or fc that is a pre-dividing clock gear.

Prescaler is set to enable/disable with MTxRUN<MTPRUN>. When MTxRUN<MTPRUN> is set to "1", counting starts. When MTxRUN<MTPRUN> is set to "0", the counter is stopped and cleared.

17.4.18.2 Up-Counter (MTUCx)

This counter is a 16-bit binary counter.

- Source clock

  The source clock can be set with MTxIGCR<IGCLK[1:0]>.

  Prescaler output clock can be chosen among φT0, φT1, φT2, φT4

- Start/Stop counter operation

  Counter operation is set with MTxRUN<MTRUN>. When <MTRUN>="1" is set, counter operation starts. When <MTRUN>="1" is set, the counter is stopped and cleared at the same time.

  And when MTxIGRESTA<IGRESTA>="1" is set, counter is cleared and started count-up from zero.

- Counter clear timing

  1. Comparing a match

     The counter is cleared when a value of up-counter (MTUCx) is match with MTxIGRG4.

  2. Counter stopping

     If Mx0RUN<MTRUN>="0" is set, the counter is stopped and cleared.

  3. Counter restarts

     If MTxIGRESTA<IGRESTA>="1" is set, the counter is cleared and counted-up from 0.

  4. In trigger start mode

     In trigger start mode, the counter is stopped and cleared when MTxIN pin becomes the stopping to clear level.

- Count-up & clear operation

  Count-up & clear operation and setting cycle are described in the two cases respectively; one is the case that φT0 is chosen as a source clock, the other case is that φT1, φT2 or φT4 is chosen as a source clock.

  1. φT0 is selected as a source clock

     When φT0 is selected as a source clock, two source clocks are required for match counting and clear counting, so that setting cycle is M+1.

  2. φT1, φT2 or φT4 is selected as a source clock

     When φT1, φT2 or φT4 is selected as a source clock, one source clock is required for match counting and clear counting, so that setting cycle is M.

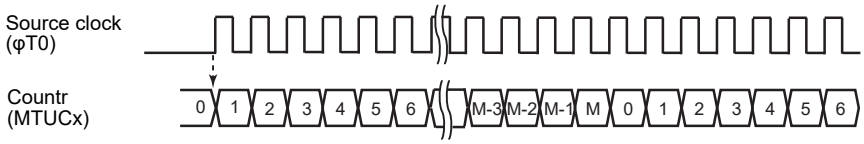


Figure 17-4 Count-up/clear operation when φT0 is selected as a source clock

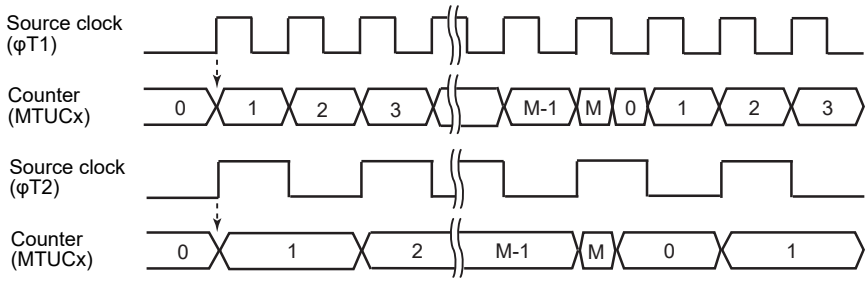


Figure 17-5 Count-up/clear operation when φT1, φT2 or φT4 is selected as a source clock

17.4.18.3 Cycle Setting Register (MTxIGRG4)

This register sets the cycle of PPG output consisting of double-buffering structure. Data update timing is one cycle after when MTxIGRG4 matches with up-counter MTUCx clearing the counter. At this time, data transfer is taken place from register buffer 4 to timer register MTxIGRG4.

17.4.18.4 Timer register (MTxRG0, MTxRG1, MTxIGRG2, MTxIGRG3, MTxIGRG4), Trigger register (MTxIGTRG)

This register sets a value to compare with up-counter MTUCx. When these are matched, the match detect signal is output. Timer register, MTxRG0/1, MTxIGRG2/3 and trigger register MTxIGTRG are double-buffering structure paired with each register buffer. When MTxIGRG4 matches with up-counter MTUCx, the counter is cleared and data is updated at the same time. Also at this time, data transfer is taken place from register buffer 2/3/5 to timer register MTxIGRG2/3 and trigger register MTxIGTRG.

In IGBT mode, MTxRG0/1 is always double-buffering structure.

- Write/read operation of timer registers (MTxRG0, MTxRG1, MTxIGRG2, MTxIGRG3 and trigger register MTx IGTRG) and cycle register (MTxIGRG4)

  1. Write

     When timer is stopping, above registers can be written directly. In timer in operation, data is latched in each register. When MTxIGRG4 matches with up-counter MTUCx, the counter is cleared and data is updated at the same time.

  2. Read

     Read the current value of target register comparing with 16-bit comparator. A value of register buffer cannot be read.

Note: **Use half-word access or word access.**

17.4.18.5 Capture Control

If command start or capture mode is set, this circuit captures up-counter values (MTUCx) at the.rising-es and falling edges of Maxine to MTxCP0 and MTxCP1 respectively.

17.4.18.6 Capture Register (MTxCAP0, MTxCAP1)

This register captures a value of up-counter MTUCx.

17.4.18.7 Comparators (CP0, CP1, CP2, CP3, CP4, CP5)

This comparator detects the match comparing a value of up-counter (MTUCx) with a setting value of timer register MTxRG0, MTxRG1, MTxIGRG2, MTxIGRG3, MTxIGRG4 and trigger register MTxIGTRG.

17.4.18.8 MTxOUT0, MTxOUT1 Output Control

When up-counter matches with timer register, MTxOUT0 or MTxOUT1 is output.

Initial setting of output pin is set with MTxIGOCR<IGPOL0,1>. After reset, initial state is low. When MTxIGOCR<IGPOL0,1>=0 is set, initial state is low. When MTxIGOCR<IGPOL0,1>=1 is set, initial state is high. Output control is set with MTxIGOCR<IGOEN0,1>. After reset, MTxIGOCR<IGOEN0,1> is disabled. If MTxIGOCR<IGOEN0,1> is enabled, set to 1.

17.4.18.9 Trigger Output

A trigger (MTxIGTRG) is output when the up-counter matches the trigger register.

17.4.18.10 Capture Interrupts (INTMTCAPx0,INTMTCAPx1)

Capture interrupts (INTMTCAPx0 and INTMTCAPx1) occur respectively when each capture register (MTxCP0 and MTxCP1) latches data. Interrupt setting is set by CPU.

17.4.18.11 Trigger Start Interrupt (INTMTTBx1)

When command start & trigger start mode or trigger start mode is chosen, trigger interrupt occurs when the edge specified with MTxIGCR<IGTRGSEL> is input and the counter starts. In the trigger capture mode, INTMTTBx1 interrupt does not generate at the trigger edge. When emergency output is stopping, a start trigger interrupt occurs.

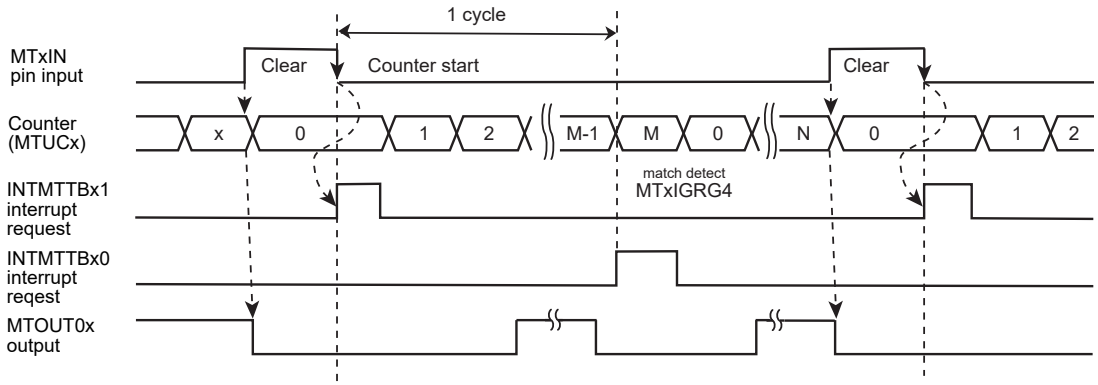


Figure 17-6 Trigger start interrupt operation

17.4.18.12 Cycle Interrupt (INTMTTBx0)

When command start & trigger capture mode or command start & trigger start mode is chosen, a cycle interrupt occurs when count starts in command start or counter reaches to a value of counter cycle setting

(MTxIGREG4) (cycle finishes by matching with a value of cycle setting). Also, a cycle interrupt occurs by matching with a value of counter cycle when emergency output is stopping. Interrupt cycle can be set to among every one cycle, every two cycles or every four cycles with MTxIGCR<IGPRD[1:0]>.

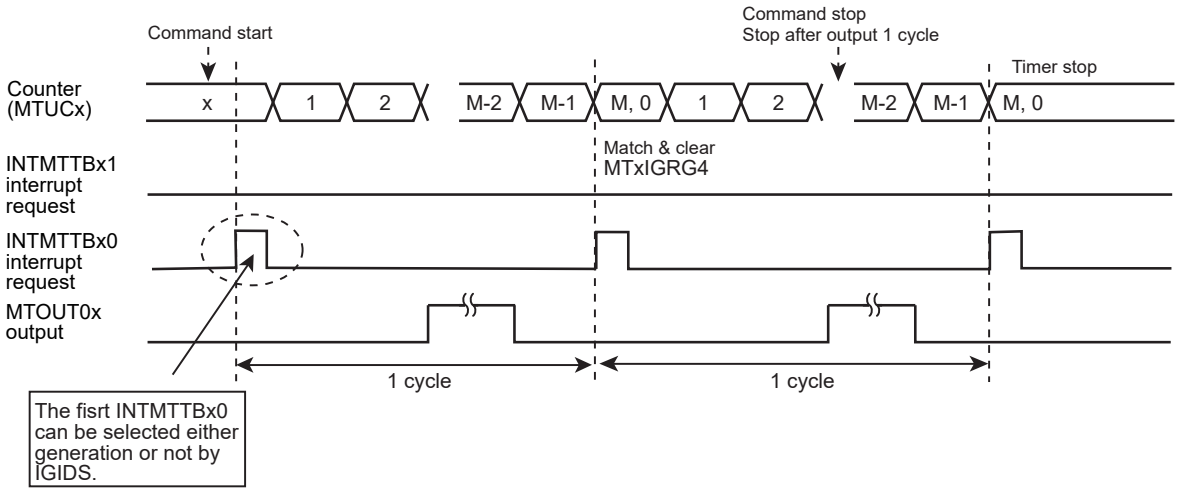


Figure 17-7 Cycle interrupt operation

In command start, a cycle interrupt at the starting count is set to enable/disable with interrupt control register MTxIGCR<IGIDIS>. At starting command (MTxRUN<MTRUN> is set to "1"), if MTxIN pin is stopping level, counting does not start (INTMTTBx0 does not occur). Counting starts by trigger start edge and NTMTTBx1 occurs.

17.4.18.13 Basic Operation

Each MTxOUT0 pin and MTxOUT1 pin output PPG.

This circuit controls waveform by comparing data set in the timer register (MTxRG0/1, MTxIGRG2/3/4) with a value of 16-bit up-counter.

A trigger is output when the data set in the trigger register (MTxIGTRG) matches the 16-bit up-counter.

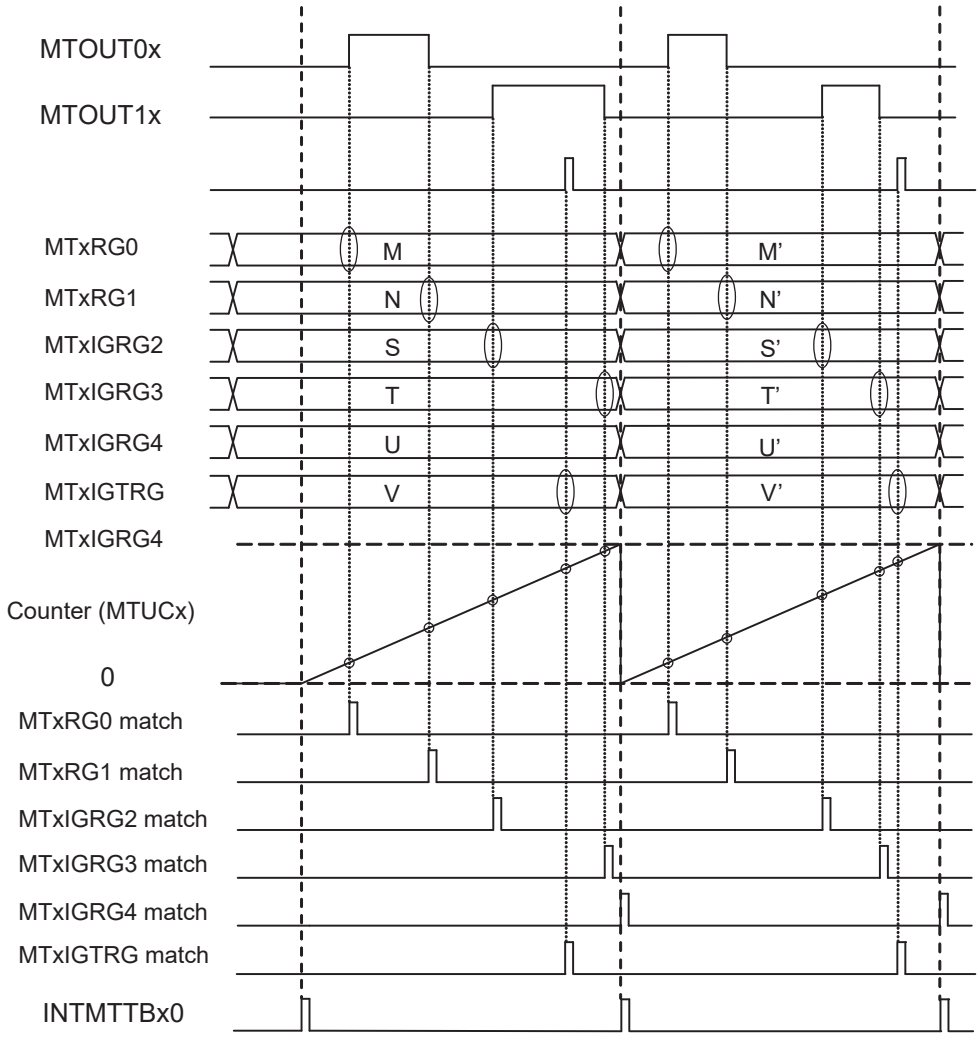


Figure 17-8 IGBT mode basic timing

17.4.18.14 Start modes

In IGBT mode, four start modes are available.

(1) Command Start & Trigger Capture Mode

When MTxRUN<MTRUN> is set to "1", counting-up starts. If the counter reaches to the setting cycle, the counter is cleared. At this time, continuous mode is set with MTxIGCR<IGSNGL>, count-up starts again. If single mode is set, counting stops.

If MTxIGRESTA<IGRESTA> is set to "1" before reaching to the setting cycle, counter is cleared at this point and count-up continues.

Counter value at the rising edge/falling edge of MTxIN input can be stored to capture registers.

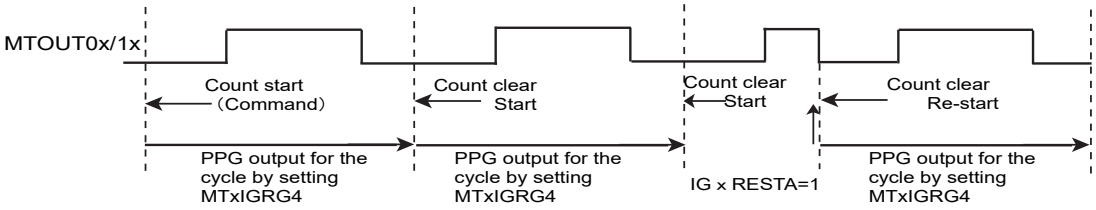


Figure 17-9 Continuous mode in command start

In the command start & trigger capture mode, when the counter starts, a counter value is captured at the each rising/falling edge of MTxIN input to each capture register (MTxCAP0 and MTxCAP1).When capture operation is done, INTMTCAPx0 and INTMTCAPx1 occur at each edge.

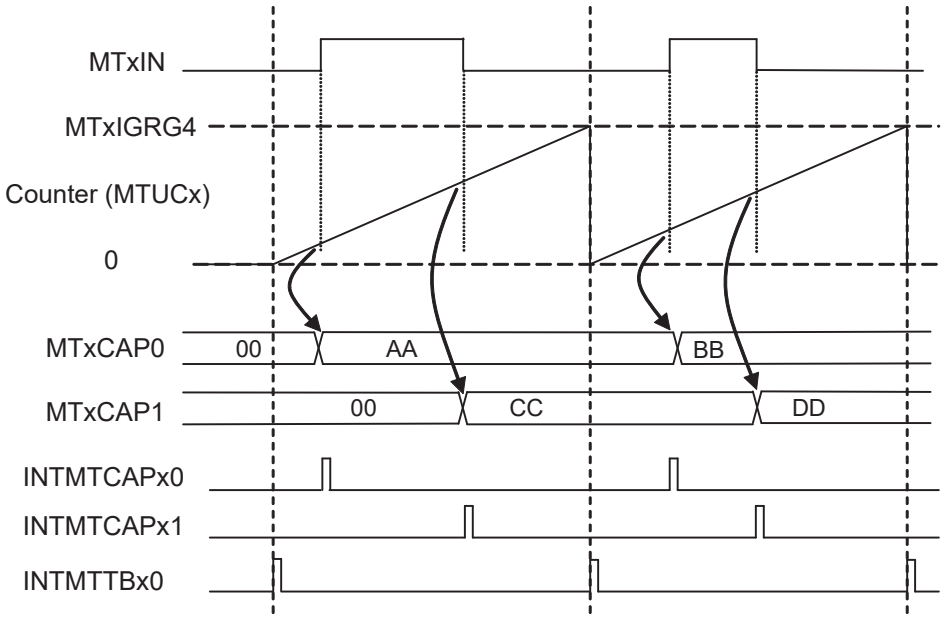


Figure 17-10 Capture operation

(2) Command Start & Trigger Start Mode

When MTxRUN<MTRUN> is set to "1", count-up starts. If there is no trigger inputs to MTxIN input, same operation previously described in command start & capture mode starts. If an edge input specified with MTxIGICR<IGTRGSEL> to MTxIN pin exists, timer counting starts. While specified clear stopping level is input, the counter is not cleared. At the starting command (when MTxRUN<MTRUN> is set to "1"), if MTxIN pin is in the stopping level, the counter does not start (INTMTTBx1 does not generate). Counting starts by trigger start edge and NTMTTBx1 occurs. (Trigger input is prior to command start.)

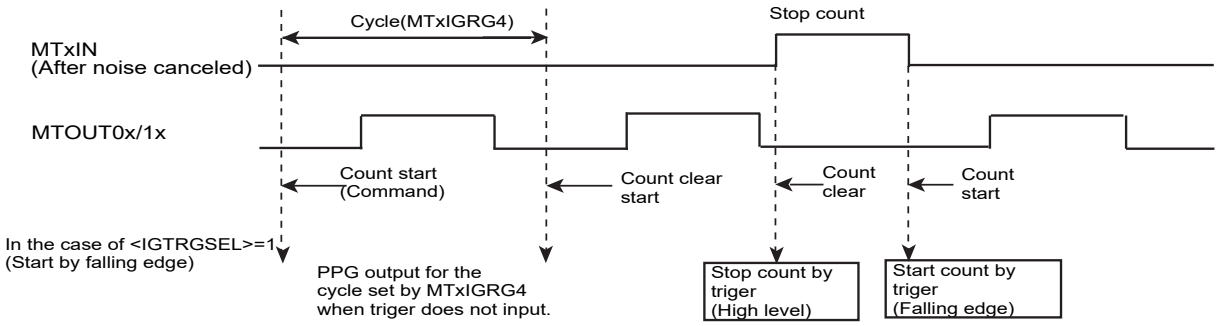


Figure 17-11 Command start & trigger start

(3) Trigger Start Mode

If an edge input specified with MTxIGICR<IGTRGSEL> exists, timer counting starts.While specified clear stopping level is input, the counter is not cleared.

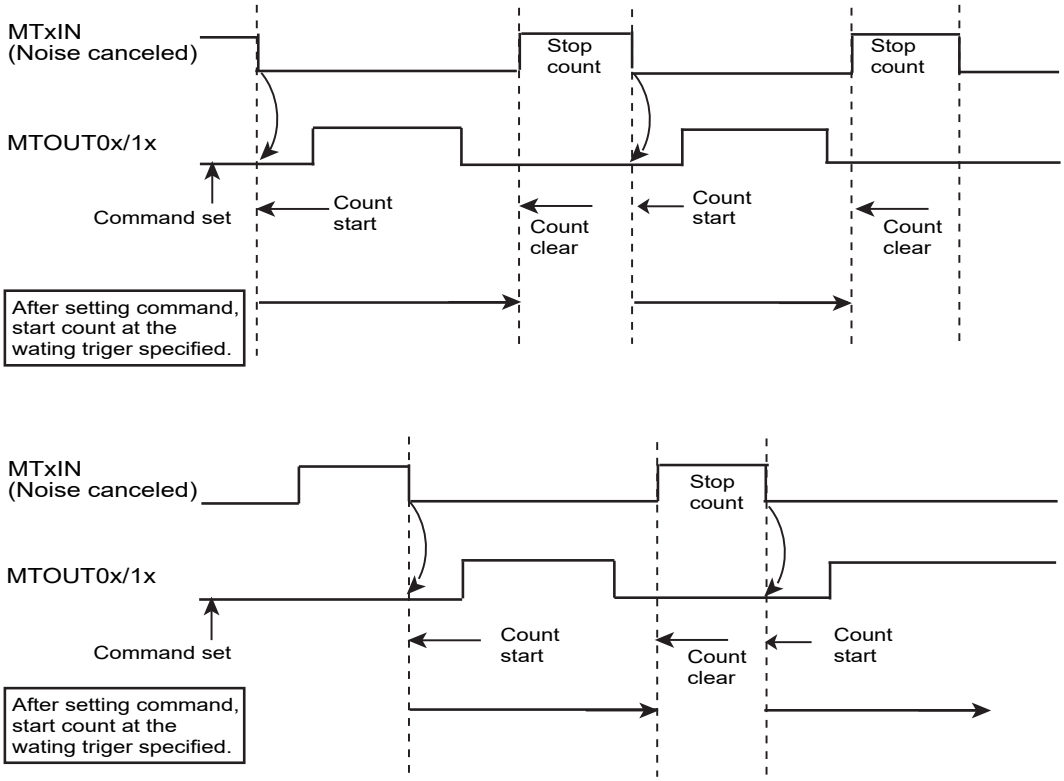


Figure 17-12 Trigger Start

(4) Synchronous Start Mode

If the synchronous start mode is used, the counter operation in each timer can operate synchronously. In addition, the up-counters in each timer are cleared synchronously by using the synchronous counter clear operation.

The MPT consists of four channels. Other three channels are synchronized with one of four channels. In TMPM46BF10FG, the following combination can be used.

Channel for start trigger (Master channel)	Synchronous operating channel (Slave channel)
MPT0	MPT1, MPT2, MPT3

(a) Synchronous start

To use the synchronous start mode, set "11" to MTxIGCR<IGSTA[1:0]> on the slave channels and set other than "11" to the master channel. The register setting for starting or stopping the counter is valid only on the master channel. The following settings on the slave channels are disabled.

- MTxRUN<MTRUN><MTPRUN>
- MTxIGCR<IGSNGL>
- MTxIGICR<IGTRGSEL><IGTRGM>

In the command start and capture mode both on master and slave channels, capturing is enabled by MTxIN.

The following register settings related to MTxOUT0, MTxOUT1 output and MTxIGTRG output can be set in each channel regardless of master/slave setting. Thus, a desired rectangle wave and trigger output can be used in each channel.

- MTxIGCR<IGSTP[1:0]>
- MTxIGOCR<IGOEN[1:0]><IGPOL[1:0]>
- MTxRG0,MTxRG1,MTxIGRG2,MTxIGRG3,MTxIGRG4
- MTxIGTRG

MTxIGCR<IGSTP[1:0]> specifying the stop condition of MTxOUT0 and MTxOUT1 is set to "00" (stopping in the output initial state) on the slave channel or "01" (stopping maintaining outputs).

A cycle interrupt can be set with MTxIGCR<IGPRD[1:0]> in each channel. A command start interrupt occurs only on the master channel. Therefore, the setting of MTxIGCR<IGIDIS> is enabled on the master channel.

Figure 17-13 shows counter operations of master channels and slave channels in the synchronous start mode. A counter operation of master channel synchronously starts with those of slave channels. Counters operate on each cycle. If a counter operation of master channel stops, at the same time those of slave channels stop.

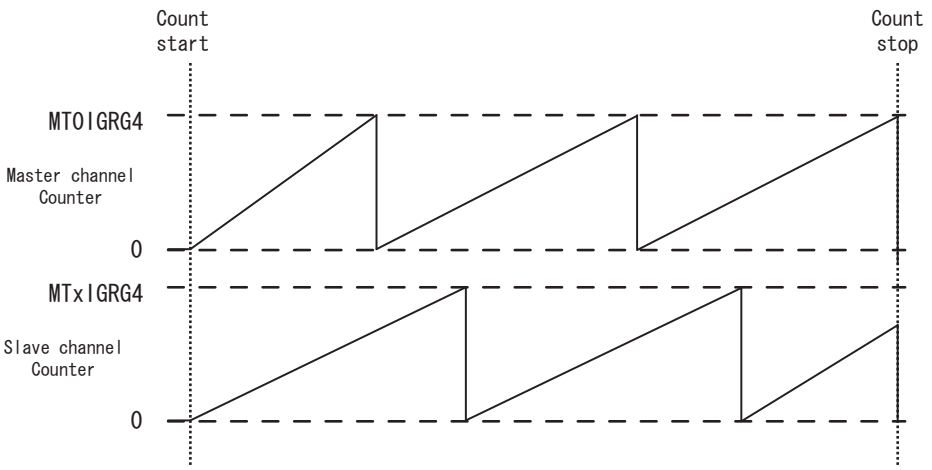


Figure 17-13 Synchronous start operation

(b) Synchronous clearing

A counter clearing timing of slave channels can be synchronized with those of the master channel.

To use synchronous counter clearing, set "1" to MTxIGCR<IGCLSYNC> of slave channels. Synchronous clearing setting is enabled regardless of MTxIGCR<IGSTA[1:0]> setting. Restarting of slave channels is enabled in each channel.

Figure 17-14 shows counter operations of master channel and slave channel at synchronous clearing setting in the synchronous start mode. The counter on the slave channel synchronously starts with those of the master channel. The counter on the slave channel is cleared at the timing when cycle match detection is found in the master channel.

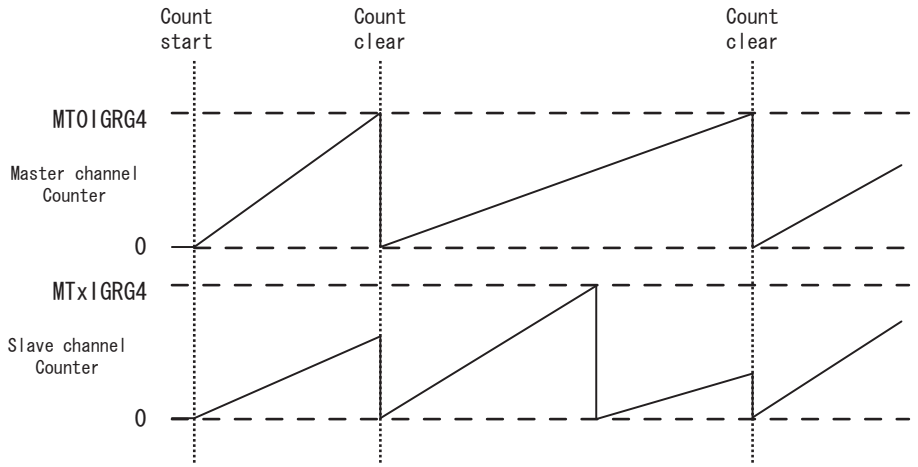


Figure 17-14 Synchronous clearing operation

17.4.18.15 Single/Continuous Output Mode

Single/continuous output mode can be set with IGBT output.

(1) Continuous Output Mode

At the starting timer (MTxRUN<MTRUN>="1"), if MTxIGCR<IGSNGL>="0" is set, continuous output mode is chosen. In the continuous output mode, specified continuous waveform can be output.

(2) Single Output Mode

At the starting timer (MTxRUN<MTRUN>="1"), if MTxIGCR<IGSNGL>="1" is set, single output mode is chosen. In the single output mode, the counter stops after output every single cycle.

At the trigger starting, the counter stops until triggers are input. Counting starts by the specified trigger input, and after one cycle has elapsed, counting stops. If trigger starts again, set MTxRUN<MTRUN>="1". After 1 cycle is output or a trigger input receives a stop level signal, IGBT output waits for a trigger again.

17.4.18.16 Stopping Type

By setting "0" to MTxRUN<MTRUN>, outputs and timers stop according to MTxIGCR<IGSTP[1:0]> setting.

(1) Counter Stops with Initial State Output

When MTxIGCR<IGSTP[1:0]> is set to "00", the counter immediately stops and MTxOUT0/1x output becomes an initial value set with MTxIGOCR<IGPOL[1:0]>.

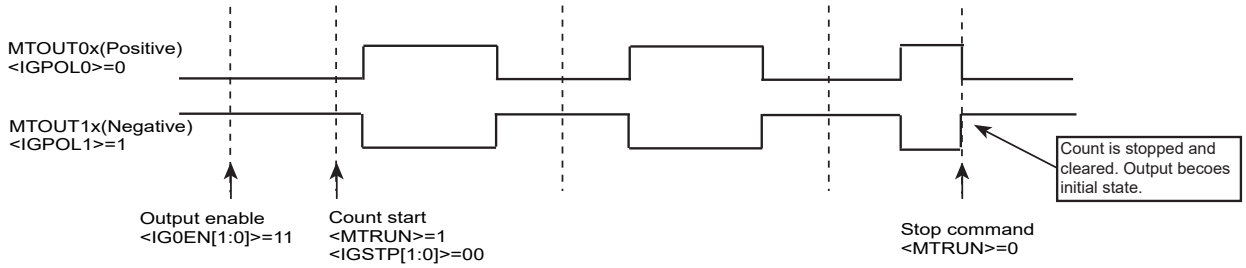


Figure 17-15 Counter stops with initial state output

(2) Counter Stops with maintaining the output status

When <IGSTP[1:0]> is set to "01", the counter immediately stops and MTxOUT0/1x output is maintained.

If the counter starts again, set MTxRUN<MTRUN>="1". At this time, outputs become an initial value (setting value of <IGPOL0> or <IGPOL1>) and then restarts.

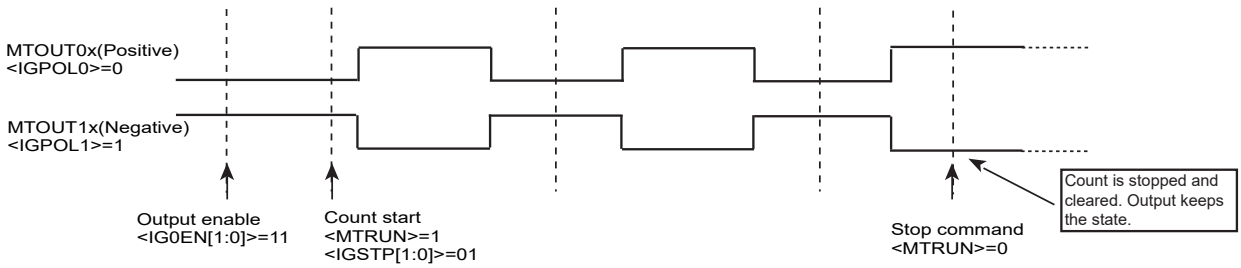


Figure 17-16 Counter stops with maintaining the output status

(3) Counter Stops with Initial State after Cycle finished

When <IGSTP[1:0]> is set to "10", the counter operates until the cycle has finished. After cycle has finished, the counter stops. However, if trigger input becomes stop level until the cycle finishes, the counter stops at this point.

If the timer is set again, check if the counter stops after cycle has finished.

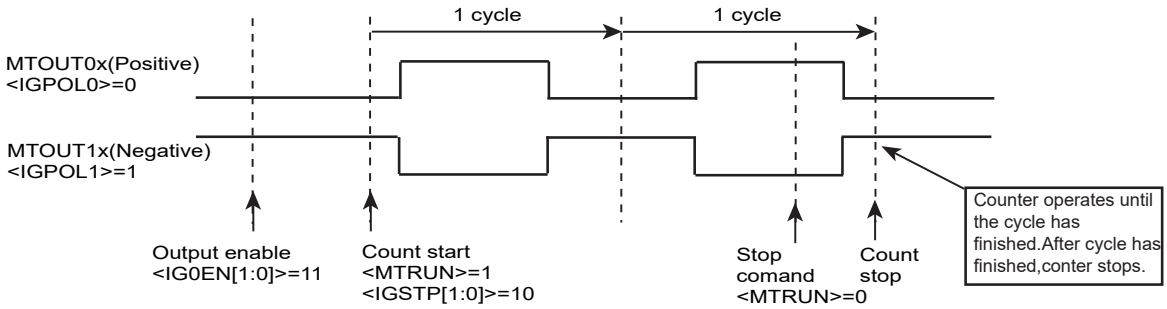


Figure 17-17 Counter stops with initial state after cycle has finished

17.4.18.17 Trigger Input

(1) Logic of Trigger Input

The valid condition of MTxIN input is set with MTxIGICR<IGTRGSEL>.

- <IGTRGSEL>=0 : Rising edge detection to start counting

  During "High" level, count-up is performed. During "Low" level, counter stops.
- <IGTRGSEL>=1 : Falling edge detection to start counting

  During "Low" level, count-up is performed. During "High" level, counter stops.

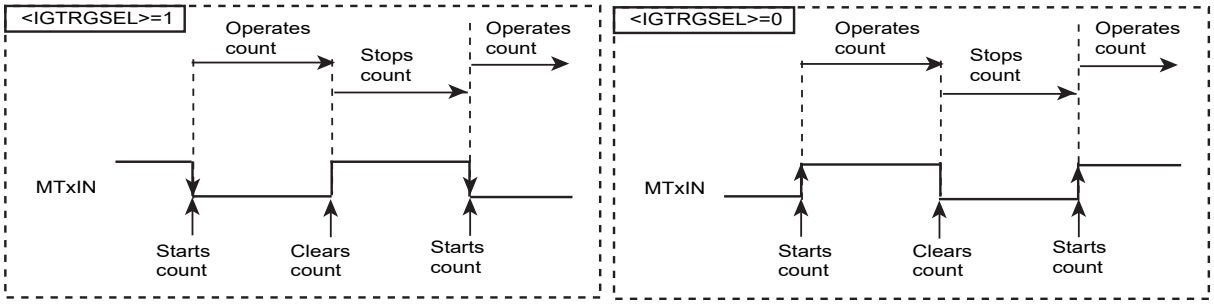


Figure 17-18 Logic of trigger input

While cycles are stopping, a stop trigger signal is accepted but a start signal is not. (Once a stop trigger signal is accepted while cycles are stopping, outputs become an initial value then the counter stops.)

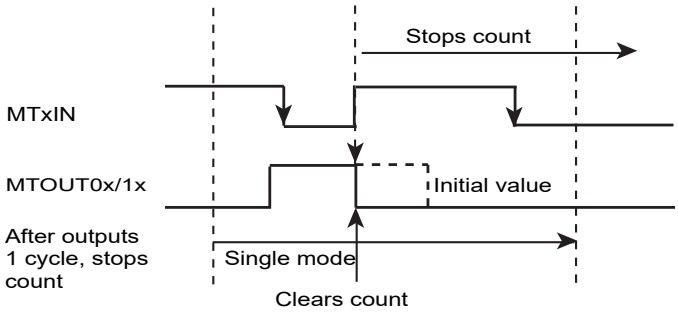


Figure 17-19 Trigger acceptance while cycles are stopping

(2) Trigger Constant Acceptance/prohibit accessing during active level

MTxIGICR<IGTRGM> can choose either condition; one is a trigger from MTxIN is always accepted during PPG output, or another is a trigger is prohibited accessing during PPG output in active. This setting is only valid for enabled pin with MTxIGOCR <IGOEN[1:0]>.

When <IGTRGM>="0" is set, a trigger input from MTxIN is always accepted regardless of MTxOUT0/1x in active/non-active. During this period, timer is started/stopped to clear and MTxOUT0/1x output becomes non-active.

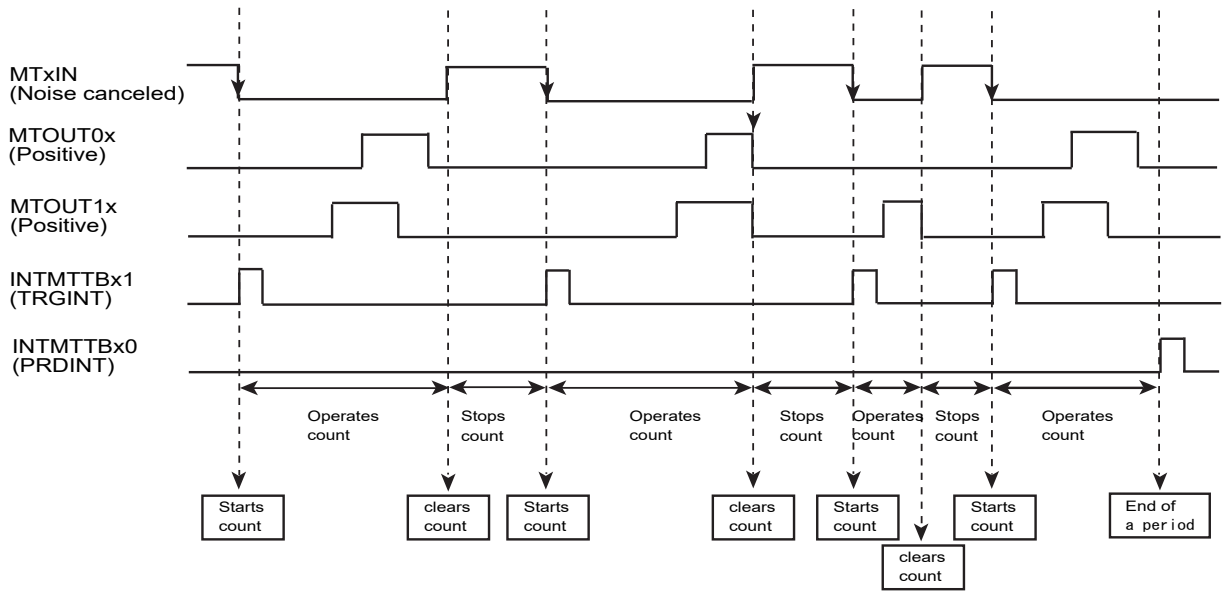


Figure 17-20 Trigger constant acceptance

When <IGTRGM>="1" is set, input edge at MTxOUT0/1x output in non-active is accepted and cleared to stop.

If input edge at MTxOUT0/1x output in active, the counter does not immediately stops. It continues to count until MTxOUT0/1x output becomes non-active. When MTxOUT0/1x output is non-active, if trigger signal is not in active level, the counter is cleared to stop and waits next start trigger signal.

If the counter operates when both MTxOUT0 and MTxOUT1 are enabled, both outputs must be in non-active. Otherwise triggers are not accepted.

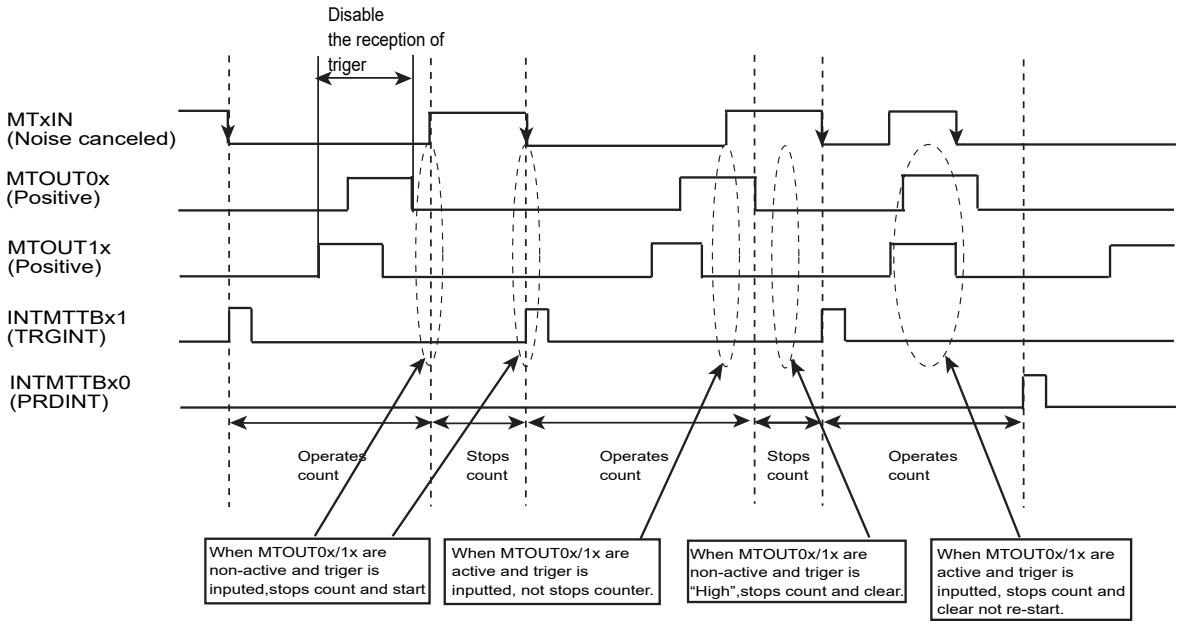


Figure 17-21 Prohibit accessing during active level

17.4.18.18 Emergency Stop Function

(1) Operation Description

When MTxIGEMGCR<IGEMGEN>="1" is set, the emergency stop function is enabled ($\overline{\text{GEMGx}}$ pin is enabled to input).

If $\overline{\text{GEMGx}}$ pin detects a low level input, MTxOUT0/ MTxOUT1 waveform becomes initial state (set with IGPOL0/IGPOL1) according to MTxIGEMGCR<IGEMGOC> setting or becomes high-impedance and generates a GEMGx interrupt.

This function prohibits only MTxOUT0/ MTxOUT1 output. The counter does not stop so that timer must be stopped in the GEMG interrupt service routine.

(2) Emergency stop monitor

On the emergency stop condition, MTxIGEMGST<IGEMGST> is set to "1". When IGEMGST is read, "1" indicates of emergency stopping.

(3) GEMG interrupts

When an emergency stop input is received, a GEMG interrupt (INTMTEMGx) occurs. If this process uses interrupt service routine, the INTMTEMGx interrupt must be enabled in advance.

If $\overline{\text{GEMGx}}$ pin is "Low" and exits emergency stop status, GEMG interrupt occurs again, and MCU is in emergency stop condition again.

(4) Exiting Emergency Stop Condition

When MCU exits emergency stop condition, check if $\overline{\text{GEMGx}}$ input is high and MTxRUN<MTRUN> is set to "0". Then confirm the timer stops (MTxIGST<IGST>=0), later MTxIGEMGCR<IGEMGRS>="1" is set for exiting emergency stop condition.

When MTxIGCR<IGSTP[1:0]>="01" or "10" is set in the stopping type selection register, set the initial setting with MTxIGOCR<IGPOL[1:0]> before writing MTxIGEMGCR<IGEMGRS>="1".

17.4.18.19 Noise Canceller

The digital noise canceller eliminate noise inputting to external input pins (MTxIN and $\overline{\text{GEMGx}}$).

It can be chosen the noise elimination time with MTxIGICR<IGNCSEL[3:0]> or MTxIGEMGCR <IGEMGCNT[3:0]> setting.

18. Serial Channel with 4bytes FIFO (SIO/UART)

18.1 Overview

Serial channel (SIO/UART) has the modes shown below.

- Synchronous communication mode (I/O interface mode)
- Asynchronous communication mode (UART mode)

Their features are given in the following.

- Transfer Clock
  - Dividing by the prescaler, from the peripheral clock (φT0) frequency into 1/1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, 1/128.
  - Make it possible to divide from the prescaler output clock frequency into 1 to 16.
  - Make it possible to divide from the prescaler output clock frequency into N+m/16 (N=2 to 15, m=1 to 15). (only UART mode)
  - The usable system clock (fsys) (only UART mode).
- Buffer
  - The usable double buffer function.
  - Make it possible to clear the transmit buffer.
- FIFO

  The usable 4 byte FIFO including transmit and receive.
- I/O Interface Mode
  - Transfer Mode: the half duplex (transmit/receive), the full duplex
  - Clock: Output (fixed rising edge) /Input (selectable either rising or falling edge)
  - Make it possible to specify the interval time of continuous transmission.
  - The state of SCxTXD pin after output of the last bit can be selected as follow:

    Keep a "High" level, "Low" level or the state of the last bit
  - The state of SCxTXD pin when an under run error is occurred in clock input mode can be selected as follow:

    Keep a "High" level or "Low" level
  - The last bit hold time of SCxTXD pin can be specified in clock input mode.
- UART Mode
  - Data length: 7 bits, 8bits, 9bits
  - Add parity bit (to be against 9bits data length)
  - Serial links to use wake-up function
  - Handshaking function with $\overline{\text{SCxCTS}}$ pin
  - Noise cancel for SCxRXD pin

In the following explanation, "x" represents channel number.

18.2 Configuration

Serial channel block diagram and serial clock generator circuit diagram are shown in bellows.

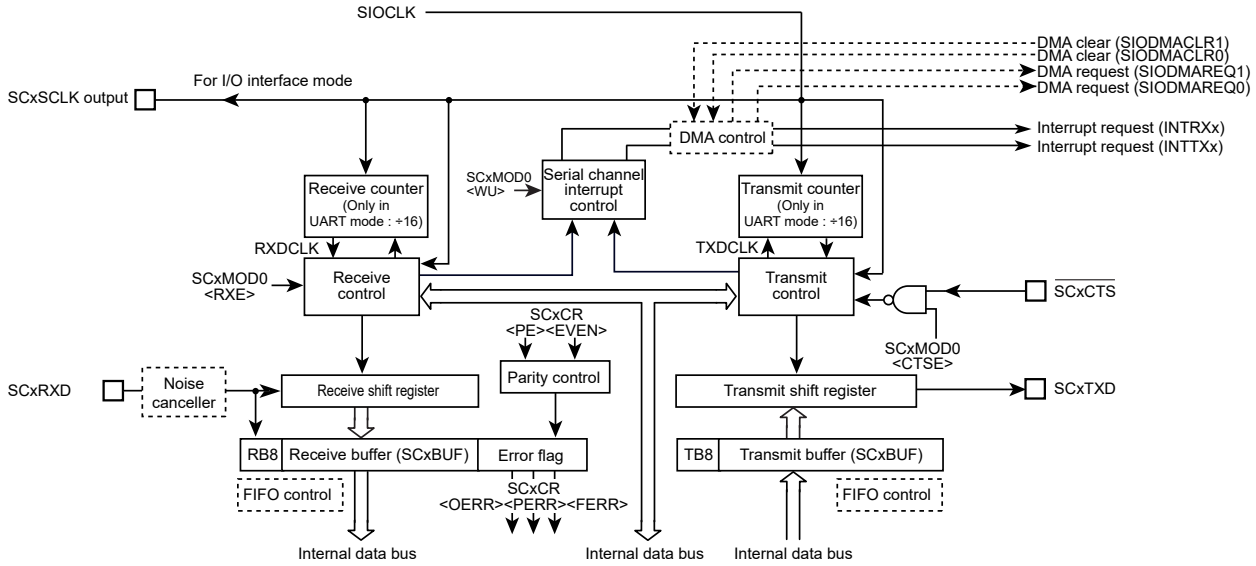


Figure 18-1 Serial Channel Block Diagram

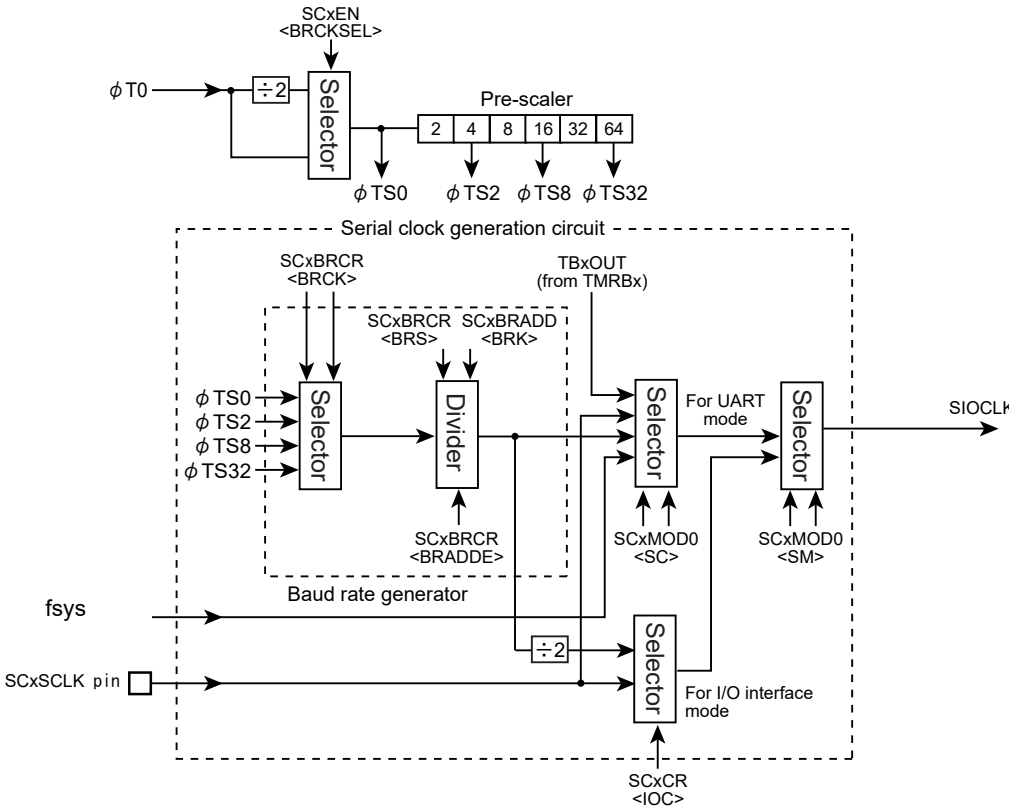


Figure 18-2 Serial clock generation circuit block diagram

18.3 Registers Description

18.3.1 Registers List

The table below shows control registers and their addresses.

For detail of the base address, refer to "Address lists of peripheral functions" of "Memory Map" chapter.

Register name		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
Receive FIFO configuration register	SCxRFC	0x0020
Transmit FIFO configuration register	SCxTFC	0x0024
Receive FIFO status register	SCxRST	0x0028
Transmit FIFO status register	SCxTST	0x002C
FIFO configuration register	SCxFCNF	0x0030

Note: Do not modify any control register when data is being transmitted or received.

18.3.2 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	-	-	-	-	-	-	BRCKSEL	SIOE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	–	R	Read as "0".
1	BRCKSEL	R/W	Selects input clock for prescaler. 0: φT0/2 1: φT0
0	SIOE	R/W	Serial channel operation 0: Disabled 1: Enabled Specified the Serial channel operation. To use the Serial channel, set <SIOE> = "1". When the operation is disabled, no clock is supplied to the other registers in the Serial channel module. This can reduce the power consumption. If the Serial channel operation is executed and then disabled, the settings will be maintained in each register.

18.3.3 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 or FIFO [read] RB: Receive buffer or FIFO

18.3.4 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	-	EHOLD			-	TXDEMP	TIDLE	
After reset	0	0	0	0	0	1	1	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
31-15	–	R	Read as "0".
14-12	EHOLD[2:0]	R/W	The last bit hold time of a SCxTXD pin in clock input mode (For only I/O interface mode) Set the last bit hold time and SCLK cycle to keep the last bit hold time equal or less than SCLK cycle/2. 000: 2/fsys 100: 32/fsys 001: 4/fsys 101: 64/fsys 010: 8/fsys 110: 128/fsys 011: 16/fsys 111: Reserved
11	-	R	Read as "0".
10	TXDEMP	R/W	The state of SCxTXD pin when an under run error is occurred in clock input mode. (For only I/O interface mode) 0: "Low" level output 1: "High" level output
9-8	TIDLE[1:0]	R/W	The state of SCxTXD pin after output of the last bit (For only I/O interface mode) When <TIDLE[1:0]> is set to "10", set "000" to <EHOLD[2:0]>. 00: Keep a "Low" level output 01 :Keep a "High" level output 10: Keep a last bit 11: Reserved
7	RB8	R	Receive data bit 8 (For only UART mode) 9th bit of the received data in the 9-bit UART mode.
6	EVEN	R/W	Parity (For only UART mode) Selects even or odd parity. The parity bit may be used only in the 7- or 8-bit UART mode. 0: Odd 1: Even Selects even or odd parity.
5	PE	R/W	Add parity (For only UART mode) Controls disabled or enabled parity. The parity bit may be used only in the 7- or 8-bit UART mode. 0: Disabled 1: Enabled
4	OERR	R	Over-run error flag (Note) 0: Normal operation 1: Error
3	PERR	R	Parity / Under-run error flag (Note) 0: Normal operation 1: Error
2	FERR	R	Framing error flag (Note) 0: Normal operation 1: Error

Bit	Bit Symbol	Type	Function
1	SCLKS	R/W	Selecting input clock edge (For I/O Interface mode) Set to "0" in the clock output mode. 0: Data in the transmit buffer is sent to SCxTXD pin every one bit on the falling edge of SCxRXD pin. Data from SCxRXD pin is received in the receive buffer every one bit on the rising edge of SCxRXD pin. In this case, the state of a SCxRXD pin starts from "High" level. (Rising edge mode) 1: Data in the transmit buffer is sent to SCxTXD pin every one bit on the rising edge of SCxSCLK pin. Data from SCxRXD pin is received in the receive buffer every one bit on the falling edge of SCxSCLK pin. In this case, the state of a SCxSCLK starts from "Low" level.
0	IOC	R/W	Selecting clock (For I/O Interface mode) 0: Clock output mode (A transfer clock is output from SCxSCLK pin.) 1: Clock input mode (A transfer clock is input to SCxSCLK pin.)

Note:<OERR>, <PERR> and <FERR> are cleared to "0" when read.

18.3.5 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 only UART mode) Writes the 9th bit of transmit data in the 9-bit UART mode.
6	CTSE	R/W	Handshake function control (For only UART mode) 0: CTS disabled 1: CTS enabled Controls handshake function. Setting "1" enables handshake function using $\overline{\text{SCxCTS}}$ pin.
5	RXE	R/W	Receive control (Note1)(Note2) 0: Disabled 1: Enabled
4	WU	R/W	Wake-up function (For only UART mode) 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 enabled, interrupt is occurred only when RB9 = "1" in a 9-bit UART mode.
3-2	SM[1:0]	R/W	Specifies transfer mode. 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode
1-0	SC[1:0]	R/W	Serial transfer clock (For only UART mode) 00: TMRB output 01: Baud rate generator 10: System clock (fsys) 11: External clock (SCxSCLK pin input) (For the I/O interface mode, the transfer clock in I/O interface mode is selected by SCxCR<IOC>.)

Note 1: Specify the all mode control registers first and then the <RXE>.

Note 2: Do not stop the receive operation (by setting SCxMOD0<RXE> to "0") when data is being received.

18.3.6 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 operation in the IDLE mode.
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. And when FIFO is enabled, specify the configuration of FIFO. In UART mode, specify the only configuration of FIFO.
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 mode) 000: None 001: 1 x SCLK cycle 010: 2 x SCLK cycle 011: 4 x SCLK cycle 100: 8 x SCLK cycle 101: 16 x SCLK cycle 110: 32 x SCLK cycle 111: 64 x SCLK cycle This parameter is valid only for the I/O interface mode when SCLK output mode is selected. In other modes, this parameter 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 a "0".

Note 1: Specify the all mode control registers first and then enable the <TXE>.

Note 2: Do not stop the transmit operation (by setting <TXE> to "0") when data is being transmitted.

18.3.7 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" .When reading the receive buffer ,this bit is cleared 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 is completed. 0 | 0 | Wait state with data in transmit buffer
4	SBLEN	R/W	STOP bit length (for UART mode) 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>.
3	DRCHG	R/W	Setting transfer direction 0: LSB first 1: MSB first Specifies the direction of data transfer. In the UART mode, set this bit to LSB first.
2	WBUF	R/W	Enable 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 i n the I/O interface mode and to transmit in the UART mode. When receiving data in the I/O interface mode (in clock input mode) and UART mode, double buffering is enabled regardless of the <WBUF>.

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 a software reset is executed, the following bits are initialized and the transmit/receive circuit and FIFO become initial state (Note1)(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.

18.3.8 SCxBRCR (Baud Rate Generator 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	-	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 "0".
6	BRADDE	R/W	N + (16 − K)/16 divider function (Only for UART mode) 0: disabled 1: enabled
5-4	BRCK[1:0]	R/W	Select input clock to the baud rate generator. 00:φTS0 01:φTS2 10:φTS8 11:φTS32
3-0	BRS[3:0]	R/W	Division ratio "N" 0000: N = 16 0001: N = 1 0010: N = 2 ... 1111: N = 15

Note 1: 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 2: 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.

18.3.9 SCxBRADD (Baud Rate Generator 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	-	-	-	-	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 the "N + (16 − K)/16" division (For UART mode) 0000: Prohibited 0001: K = 1 0010: K = 2 ... 1111: K = 15

Table 18-1 lists the settings of baud rate generator division ratio.

Table 18-1 Setting division ratio

	<BRADDE> = "0"	<BRADDE> = "1" (Note1) (Only in the UART mode)
<BRS>	Specify "N"	
<BRK>	No setting required	Specify "K" (Note2)
Division ratio	Divide by N	$N + \frac{(16 - K)}{16}$ 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: Specifying "K = 0" is prohibited.

18.3.10 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 receive FIFO. 0: Maximum 1: Same as FILL level of receive FIFO The number of receive 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 SC0RFC <RIL[1:0]>.
3	TFIE	R/W	Specify transmit interrupt for transmit FIFO. 0: Disabled 1: Enabled When transmit FIFO is enabled, transmit interrupts are enabled or disabled by this parameter.
2	RFIE	R/W	Specify receive interrupt for receive FIFO. 0: Disabled 1: Enabled When receive 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 disable Controls automatic disabling of transmission and reception. Setting "1" enables to operate as follows. Half duplex Receive: When the receive shift register, receive fuffers and receive FIFO are filled up to the specified number of valid bytes, SCxMOD0<RXE> is automatically set to "0" to inhibit further reception. Half duplex Transmit: When the transimit shift register, transmit buffers and the transmit FIFO are empty, SCxMOD1<TXE> is automatically set to "0" to inhibit further transmission. Full duplex: When either of the above two conditions is satisfied, <TXE> and <RXE> are automatically set to "0" to inhibit further transmission and reception.
0	CNFG	R/W	FIFO enable. 0: Disabled 1: Enabled Enables FIFO.(Note2) When <CNFG> is set to "1", FIFO is enabled. If FIFO is enabled, the SCOMOD1 <FDPX[1:0]> setting automatically configures FIFO as follows: Half duplex Receive: Receive FIFO 4bytes Half duplex Transmit: Transmit FIFO 4bytes Full duplex: Receive FIFO 2bytes and Transmit FIFO 2bytes

Note 1: Regarding Transmit FIFO, the maximum number of bytes being configured is always available. (See also <CNFG>.)

Note 2: The FIFO can not be used in 9 bit UART mode.

18.3.11 SCxRFC (Receive 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	Receive FIFO clear (Note1) 1: Clear When SCxRFC<RFCS> is set to "1", the receive FIFO is cleared and SCxRST<RLVL[2:0]> is "000". And also the read pointer is initialized. Read as "0".
6	RFIS	R/W	Select interrupt generation condition. 0: When FIFO fill level (SCxRST<RLVL[2:0]>) = Receive FIFO fill level to generate receive interrupt (<RIL[1:0]>) 1: When FIFO fill level (SCxRST<RLVL[2:0]>) ≥ Receive FIFO fill level to generate receive interrupt (<RIL[1:0]>) For the detail of interrupt condition, refer to "18.13.1.2 FIFO"
5-2	-	R	Read as "0".
1-0	RIL[1:0]	R/W	FIFO fill level to generate receive interrupts. .

	Half duplex	Full duplex
00	4 bytes	2 bytes
01	1 byte	1 byte
10	2 bytes	2 bytes
11	3 bytes	1 byte

Note: To use Transmit/Receive FIFO buffer, Transmit/Receive FIFO must be cleared after setting the SIO transfer mode (half duplex/full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1")

18.3.12 SCxTFC (Transmit 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	-	-	-	-	-	-	-	TBCLR
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
31-9	–	R	Read as "0".
8	TBCLR	W	Transmit buffer clear 0: Don't care 1: Clear When SCxTFC<TBCLR> is set to "1", the transmit buffer is cleared. Read as "0".
7	TFCS	W	Transmit FIFO clear (Note1) 0: Don't care 1: Clear When SCxTFC<TFCS> is set to "1", the transmit FIFO is cleared and SCxTST<TLVL[2:0]> is "000". And also the write pointer is initialized. Read as "0".
6	TFIS	R/W	Selects interrupt generation condition. 0: When FIFO fill level (SCxTST<TLVL[2:0]>) = Transmit FIFO fill level to generate transmit interrupt (<TIL[1:0]>) 1: When FIFO fill level (SCxTST<TLVL[2:0]>) ≤ Transmit FIFO fill level to generate transmit interrupt (<TIL[1:0]>) For the detail of interrupt condition, refer to "18.13.2.2 FIFO"
5-2	–	R	Read as "0".
1-0	TIL[1:0]	R/W	Fill level which transmit interrupt is occurred. (Half duplex / Full duplex) 00: Empty / Empty 01: 1 byte / 1 byte 10: 2 bytes / Empty 11: 3 bytes / 1 byte

Note 1: To use Transmit/Receive FIFO buffer, Transmit/Receive FIFO must be cleared after setting the SIO transfer mode (half duplex/full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").

Note 2: In case that SCxEN<SIOE>="0" (Stop SIO/UART operation) or the operation mode is changed to IDLE mode with SCxMOD<I2SC>="0" (Stop SIO/UART operation in IDLE mode), SCxTFC is initialized again.After you perform the following operations, configure the SCxTFC register again.

18.3.13 SCxRST (Receive 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	Receive FIFO Overrun. (Note) 0: Not generated 1: Generated
6-3	–	R	Read as "0".
2-0	RLVL[2:0]	R	Status of Receive FIFO fill level. 000: Empty 001: 1 byte 010: 2 bytes 011: 3 bytes 100: 4 bytes

Note: **<ROR> is cleared to "0" when receive data is read from the SCxBUF.**

18.3.14 SCxTST (Transmit 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	Transmit FIFO Under run. (Note) 0: Not generated 1: Generated
6-3	–	R	Read as "0".
2-0	TLVL[2:0]	R	Status of Transmit FIFO level 000: Empty 001: 1 byte 010: 2 byte 011: 3 byte 100: 4 byte

Note: **<TUR> is cleared to "0" when transmit data is written to the SCxBUF.**

18.4 Operation in Each Mode

Table 18-2 shows the modes.

Table 18-2 Modes

Mode	type	Data length	Transfer direction	Specifies whether to use parity bits.	STOP bit length (transmit)
Mode 0	Synchronous communication mode (I/O interface mode)	8 bits	LSB first/MSB first	-	-
Mode 1	Asynchronous communication mode (UART mode)	7 bits	LSB first	o	1 bit or 2 bits
Mode 2		8 bits		o	
Mode 3		9 bits		×	

The Mode 0 is a synchronous communication and can be used to extend I/O. This mode transmits and receives data in synchronization with SCLK clock. SCLK clock can be used for both input and output modes. The direction of data transfer can be selected from LSB first or MSB first. This mode is not allowed either to use parity bits or STOP bits.

The mode 1, mode 2 and mode 3 are asynchronous modes and the transfer directions can be selected as only the LSB first.

Parity bits can be added in the mode 1 and mode 2. The mode 3 has a wakeup 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.

18.5 Data Format

18.5.1 Data Format List

Figure 18-3 shows data format.

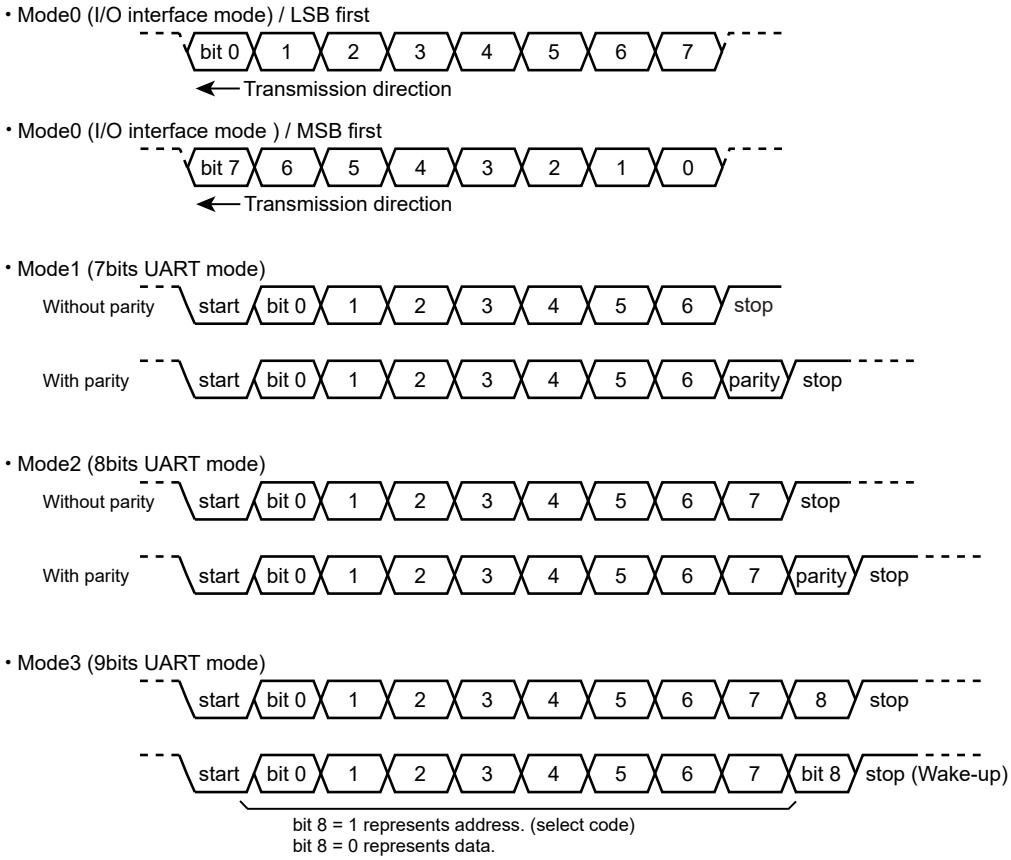


Figure 18-3 Data Format

18.5.2 Parity Control

The parity bit can be added with a transmitted data only in the 7- or 8-bit UART mode. And the received parity bit can be compared with a generated one.

Setting "1" to SCxCR<PE> enables the parity. SCxCR<EVEN> selects either even or odd parity.

18.5.2.1 Transmission

Upon data transmission, the parity control circuit automatically generates the parity with the data in the transmit buffer. The parity bit will be stored in SCxBUF<TB7> in the 7-bit UART mode and SCxMOD<TB8> in the 8-bit UART mode.

The <PE> and <EVEN> settings must be completed before data is written to the transmit buffer.

18.5.2.2 Receiption

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>, 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 SCxCR<PERR> is set to "1".

In use of the FIFO, <PERR> indicates that a parity error was generated in one of the received data.

18.5.3 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.

18.6 Clock Control

18.6.1 Prescaler

There is a 7-bit prescaler to divide a prescaler input clock φT0 by 1, 2, 4, 8, 16, 32, 64 and 128.

Use the CGSYSCR and SCxEN<BRCKSEL> in the clock/mode control block to select the input clock 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".

18.6.2 Serial Clock Generation Circuit

The serial clock generation 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.

18.6.2.1 Baud Rate Generator

The baud rate generator generates transmit and receive clocks to determine the serial channel transfer rate.

(1) Baud Rate Generator input clock

The input clock of the baud rate generator is selected from the prescaler outputs divided by 1, 4, 16 and 64.

This input clock is selected by setting the SCxEN<BRCKSEL> and SCxBRCR<BRCK>.

SCxEN<BRCKSEL>	SCxBRCR<BRCK>	Baud rate generator input clock φTx
0	00	φT0/2
0	01	φT0/8
0	10	φT0/32
0	11	φT0/128
1	00	φT0
1	01	φT0/4
1	10	φT0/16
1	11	φT0/64

(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 1/(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[3:0]>	Divide by K SCxBRADD<BRK[3:0]>
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.

The input clock to the divider of baud rate generator is φTx, the baud rate generator output clock in the case of 1/N and N + (16-K)/16 is shown below.

- Divide by N

$$\text{Baud rate generator output clock} = \frac{\phi Tx}{N}$$

- N + (16-K)/16 division

$$\text{Baud rate generator output clock} = \frac{\phi Tx}{N + \frac{(16 - K)}{16}}$$

18.6.2.2 Clock Selection Circuit

A clock can be selected by setting the modes and the register.

Modes can be specified by setting the SCxMOD0<SM[1:0]>

The clock in I/O interface mode is selected by setting SCxCR<IOC><SCLKS>.

The clock in UART mode is selected by setting SCxMOD0<SC[1:0]>.

(1) Transfer Clock in I/O interface mode

Table 18-3 shows clock selection in I/O interface mode.

Table 18-3 Clock Selection in I/O Interface Mode

Mode SCxMOD0<SM[1:0]>	Input/Output selection SCxCR<IOC>	Clock edge selection SCxCR<SCLKS>	Clock of use
"00" (I/O interface mode)	"0" (Clock output mode)	"0" (Transmit : falling edge, Receive : rising edge)	Divided by 2 of the baud rate generator output.
	"1" (Clock input mode)	"0" (Transmit : falling edge, Receive : rising edge)	SCxSCLK pin input
		"1" (Transmit : rising edge, Receive : falling edge)	SCxSCLK pin input

To use SCxSCLK input, the following conditions must be satisfied.

- <u>If double buffer is used</u>
  - SCLK cycle > 6/fsys
- <u>If double buffer is not used</u>
  - SCLK cycle > 8/fsys

(2) Transfer clock in the UART mode

Table 18-4 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 18-4 Clock Selection in UART Mode

Mode SCxMOD0<SM[1:0]>	Clock selection SCxMOD0<SC[1:0]>
UART Mode ("01", "10", "11")	"00" : TMRB output
	"01" : Baud rate generator
	"10" : fsys
	"11" : SCxSCLK pin input

To use SCxSCLK pin input, the following conditions must be satisfied.

- SCLK cycle > 2/fsys

To enable the timer output, 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 rates can be obtained by using the following formula.

Baud rate calculation

$$\text{Transfer rate} = \frac{\text{Clock frequency selected by CGSYSCR<PRCK[2:0]>}}{(\text{TBxRG1} \times 2) \times 2 \times 16}$$

In the case the timer prescaler clock ΦT1 (2division ratio) is selected.

One clock cycle is a period that the timer flip-flop is inverted twice.

18.7 Transmit/Receive Buffer and FIFO

18.7.1 Configuration

Figure 18-4 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.

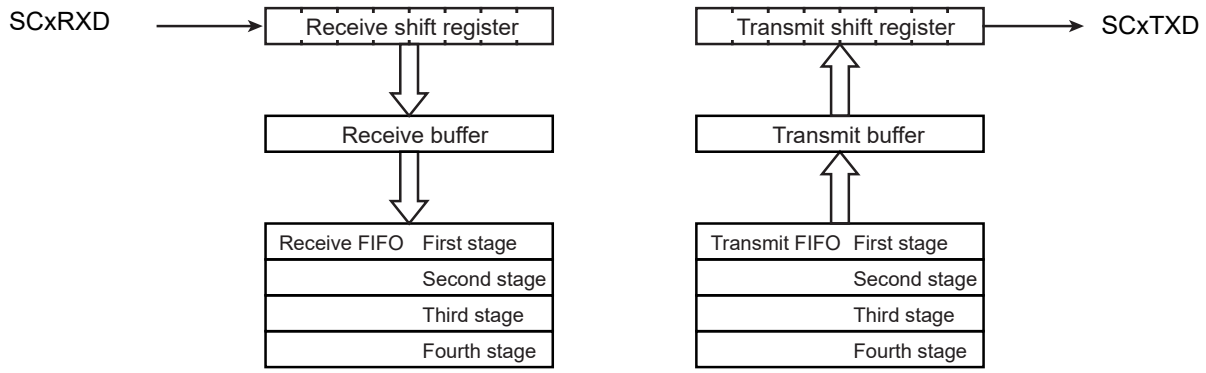


Figure 18-4 The Configuration of Buffer and FIFO

18.7.2 Transmit/Receive Buffer

Transmit buffer and receive buffer are double-buffered. The buffer configuration is specified by SCxMOD2<WBUF>.

When serial channel is operated as receive, if it is operated as clock input mode in the I/O interface mode or it is operated as the UART mode, it's double buffered regardless of <WBUF> settings.

In other modes, it's according to the <WBUF> settings.

Table 18-5 shows correlation between modes and buffers.

Table 18-5 Mode and buffer Composition

Mode		SCxMOD2<WBUF>	
		"0"	"1"
UART mode	Transmit	Single	Double
	Receive	Double	Double
I/O interface mode (Clock input mode)	Transmit	Single	Double
	Receive	Double	Double
I/O interface mode (Clock output mode)	Transmit	Single	Double
	Receive	Single	Double

18.7.3 Initialize Transmit Buffer

When transmission is stopped with a data in the transmit buffer, it is necessary to initialize the transmit buffer before new transmit data is written to transmit buffer.

The transmit buffer must be initialized when the transmit operation is stopped. To stop the transmit operation can be confirmed by reading SCxMOD2<TXRUN>. After confirming to stop the transmit operation, SCxTFC<TBCLR> is set to "1" and initialize the transmit buffer.

When a transmit FIFO is enabled, the initialize operation is depend on the data in a transmit FIFO. If transmit FIFO has data, a data is transferred from a transmit FIFO to a transmit buffer. If is does not have data, SCxMOD2<RBEMP> is set to "1".

Note: In the I/O interface mode with clock input mode is input asynchronously. When transmit operation is stopped, do not input the clock.

18.7.4 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 Transmit/Receive FIFO buffer, Transmit/Receive FIFO must be cleared after setting the SIO transfer mode (half duplex/ full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").**

Table 18-6 shows correction between modes and FIFO.

Table 18-6 Mode and FIFO Composition

	SCxMOD1<FDPX[1:0]>	Receive FIFO	Transmit FIFO
Half duplex Receive	"01"	4byte	-
Half duplex Transmit	"10"	-	4byte
Full duplex	"11"	2byte	2byte

18.8 Status Flag

The SCxMOD2 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". When reading the receive buffer is read, this bit is cleared to "0".

<TBEMP> shows that the transmit buffer is 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".

18.9 Error Flag

Three error flags are provided in the SCxCR. The meaning of the flags is changed depending on the modes. The table below shows the meanings in each mode.

These flags are cleared to "0" after reading the SCxCR.

Mode	Flag		
	<OERR>	<PERR>	<FERR>
UART mode	Over-run error	Parity error	Framing error
I/O Interface mode (Clock input mode)	Over-run error	Under-run error (When a double buffer and FIFO are used)	Fixed to 0
		Fixed to 0 (When a double buffer and FIFO are not used)	
I/O Interface mode (Clock output mode)	Undefined	Undefined	Fixed to 0

18.9.1 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 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 mode with clock output mode, the SCxSCLK pin output stops upon setting the flag.

Note: **To switch from the I/O interface mode with clock output mode to other modes, read the SCxCR and clear the overrun flag.**

18.9.2 PERR Flag

This flag indicates a parity error in the UART mode and an under-run error or completion of transmit 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 received parity bit.

In the I/O interface mode, <PERR> is set to "1" under the following conditions when a double buffer is enabled.

In the clock input mode, <PERR> is set to "1" when the clock is input after completing data output of the transmit shift register with no data in the transmit buffer.

In the clock output mode, <PERR> is set to "1" after completing output of all data and the clock output stops.

Note: To switch from the I/O interface mode with clock output mode to other modes, read the SCxCR and clear the under-run flag.

18.9.3 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>, the stop bit status is determined by only 1’st STOP bit.

This bit is fixed to "0" in the I/O interface mode.

18.10 Receive

18.10.1 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 eighth pulse.

18.10.2 Receive Control Unit

18.10.2.1 I/O interface mode

In the clock output mode with SCxCR <IOC> set to "0", the SCxRXD pin is sampled on the rising edge of SCxSCLK pin.

In the clock input mode with SCxCR <IOC> set to "1", the SCxRXD pin is sampled on the rising or falling edge of SCxSCLK pin depending on the SCxCR <SCLKS>.

18.10.2.2 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.

18.10.3 Receive Operation

18.10.3.1 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 INTRXx 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 cleared to "0" by reading the receive buffer. When the double buffer is disabled, the receive buffer full flag has no meaning.

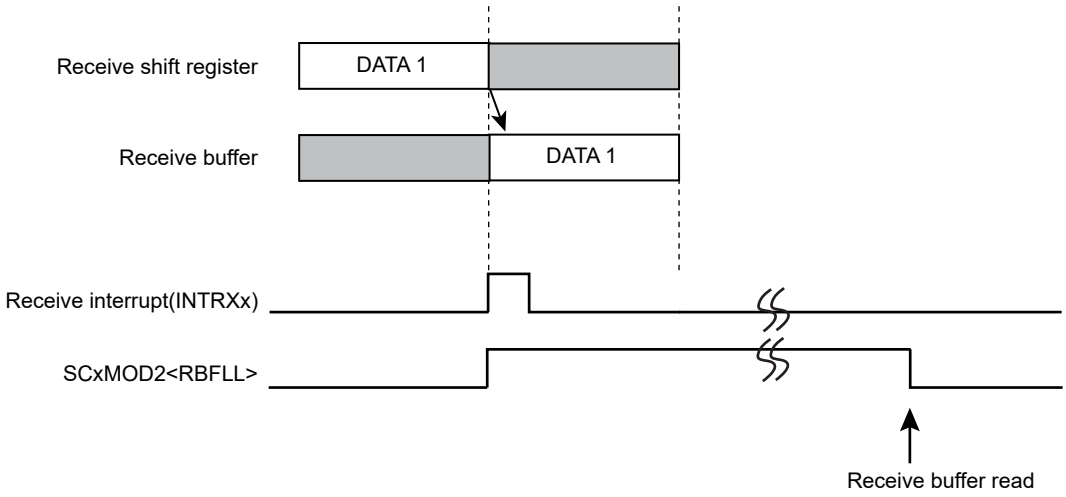


Figure 18-5 Receive Buffer Operation

18.10.3.2 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[1:0]>.

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 Receive mode are described as follows.

SCxMOD1<FDPX[1:0]> = "01"	:Transfer mode is set to half duplex mode
SCxFCNF<RFST><TFIE><RFIE>	:Automatically inhibits continuous reception after reaching the fill level.
<RXTCNT><CNFG> = "10111"	:The number of bytes to be used in the receive FIFO is the same as the interrupt generation fill level.
SCxRFC<RIL[1:0]> = "00"	:The fill level of FIFO in which generated receive interrupt is set to 4 bytes
SCxRFC<RFCS><RFIS> = "01"	: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 completed.

In the above condition, if the continuous reception after reaching the fill level is enabled, it becomes possible to receive a data continuously by reading the data in the FIFO.

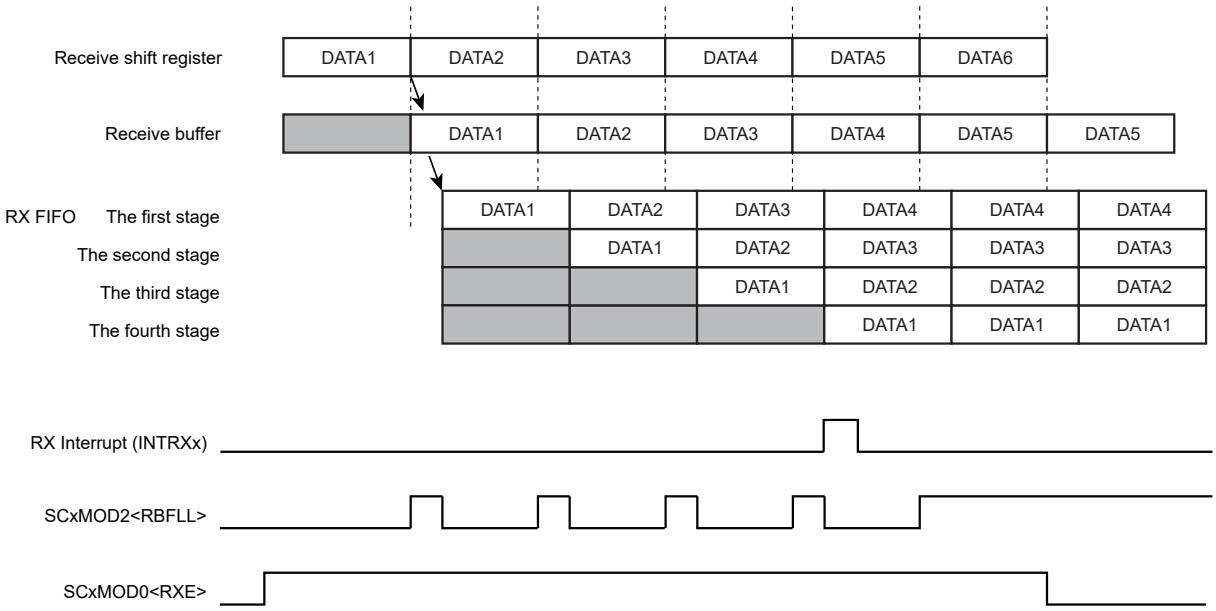


Figure 18-6 Receive FIFO Operation

18.10.3.3 I/O interface mode with clock output mode

In the I/O interface mode with clock output mode setting, clock stops when all received data is stored in the receive buffer and FIFO. So, in this mode, the over-run 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 clock 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, clock output is restarted.

(2) Case of double buffer

Stop clock output after receiving the data into a receive shift register and a receive buffer.

When a data is read, clock output is restarted.

(3) Case of FIFO

Stop clock 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 clock output restarts.

And if SCxFCNF<RXTXCNT>is set to "1", clock stops and receive operation stops with clearing SCxMOD0<RXE>.

18.10.3.4 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. 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 enabled, the 9-bit UART mode is prohibited because up to 8-bit data can be stored in receive 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>.

18.10.3.5 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".

18.10.3.6 Overrun Error

When receive FIFO is disabled, the overrun error occurs 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 receive FIFO is enabled, overrun error is occurred and set overrun flag by no reading receive FIFO before moving the next data into received buffer when receive FIFO is full. In this case, the contents of receive FIFO are not lost.

In the I/O interface mode with clock output mode, the clock output automatically stops, so this flag has no meaning.

Note: When the mode is changed from I/O interface mode with clock output mode to the other modes, read SCxCR and clear overrun flag.

18.11 Transmit

18.11.1 Transmit 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.

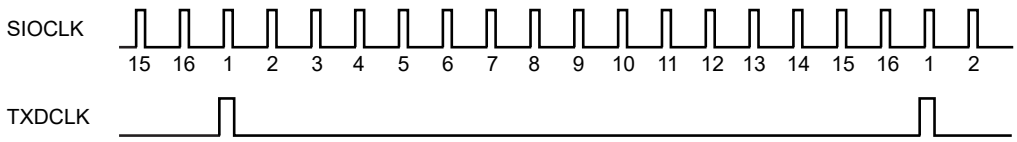


Figure 18-7 Generation of Transmission Clock in UART mode

18.11.2 Transmit Control

18.11.2.1 In I/O Interface Mode

In the clock output mode with SCxCR<IOC> set to "0", each bit of data in the transmit buffer is outputted to the SCxTXD pin on the falling edge of SCxSCLK pin.

In the clock input mode with SCxCR<IOC> set to "1", each bit of data in the transmit buffer is outputted to the SCxTXD pin on the rising or falling edge of the SCxSCLK pin according to the SCxCR<SCLKS>.

18.11.2.2 In 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.

18.11.3 Transmit Operation

18.11.3.1 Operation of Transmit Buffer

If double buffering is disabled, the CPU writes data only to transmit shift register and the transmit interrupt INTTXx is generated upon completion of data transmission.

When double buffering is enabled (including the case where the transmit FIFO is enabled), if "1" is set to SCxMOD1<TXE>, data in the transmit buffer is transferred 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".

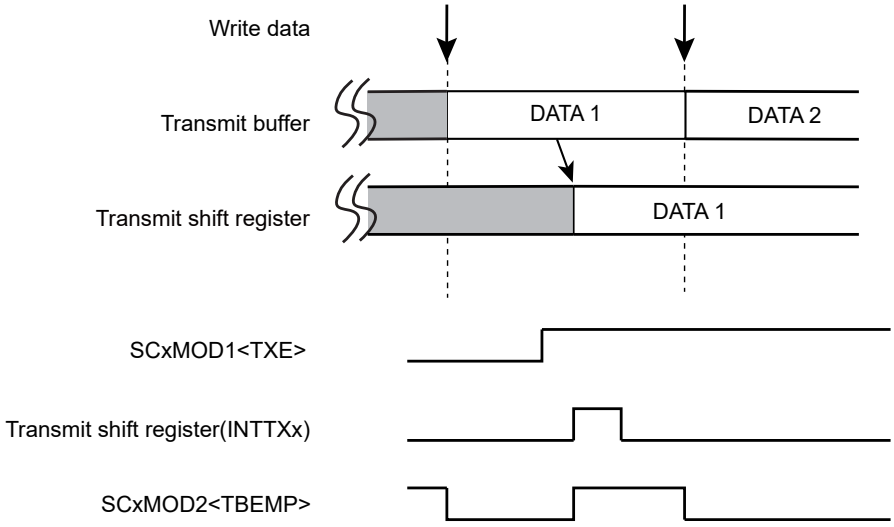


Figure 18-8 Operation of Transmit Buffer (Double-buffer is enabled)

18.11.3.2 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".

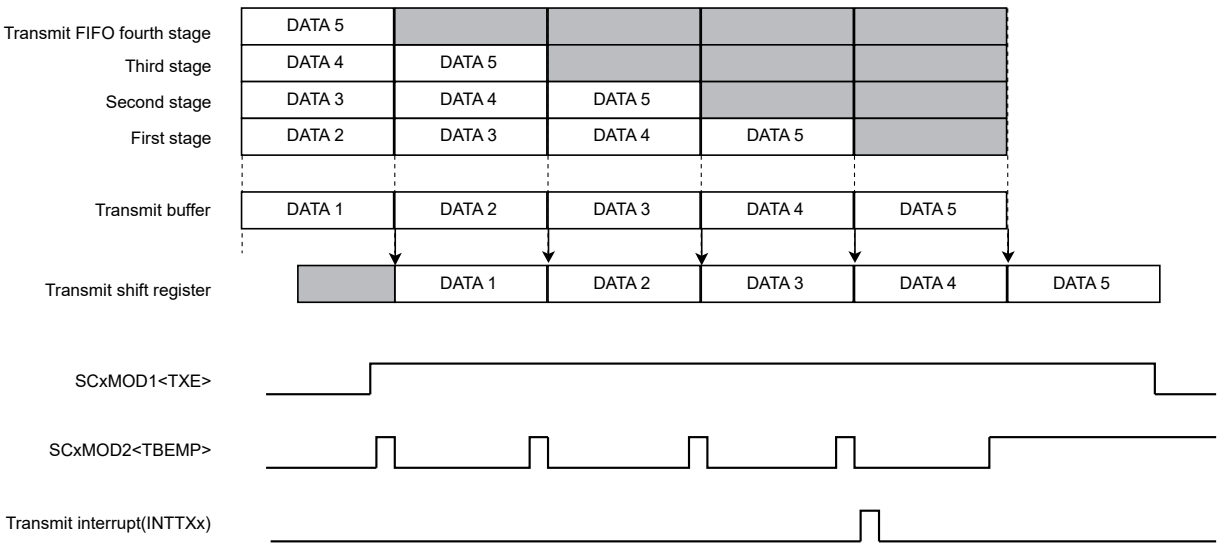
Note: To use Transmit FIFO buffer, Transmit FIFO must be cleared after setting the SIO transfer mode (half duplex/ full duplex) and enabling FIFO (SCxFCNF<CNFG>="1").

Settings and operations to transmit 5 bytes data stream by setting the transfer mode to half duplex are shown as below.

SCxMOD1<FDPX[1:0]> = "10"	:Transfer mode is set to half duplex.
SCxFCNF<RFST><TFIE><RFIE>	:Transmission is automatically disabled if FIFO becomes empty.
<RXTXCNT><CNFG> = "11011"	:The number of bytes to be used in the receive FIFO is the same as the interrupt :generation fill level.
SCxTFC<TIL[1:0]> = "00"	:Sets the interrupt generation fill level to "0".
SCxTFC<TFCS><TFIS> = "11"	:Clears receive FIFO and sets the condition of interrupt generation.
SCxFCNF<CNFG> = "1"	:Enable FIFO

After above settings are configured, data transmission can be initiated by writing 5 bytes of data to the transmit buffer and FIFO, and setting the SCxMOD1<TXE> bit to "1". When the last transmit data is moved to the transmit buffer, the transmit 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.



18.11.3.3 Transmit in I/O interface Mode with Clock Output Mode

In the I/O interface mode with clock output mode, the clock output automatically stops when all data transmission is completed and underrun error will not occur.

The timing of suspension and resume of clock output is different depending on the buffer and FIFO usage.

(1) Single Buffer

The clock output stops each time one frame of data is transferred. Handshaking for each data with the other side of communication can be enabled. The clock output resumes when the next data is written in the buffer.

(2) Double Buffer

The clock output stops upon completion of data transmission in the transmit shift register and the transmit buffer. The clock 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, clock output resumes.

If SCxFCNF<RXTXCNT> is configured, SCxMOD0<TXE> bit is cleared at the same time as clock stops and the transmission stops.

18.11.3.4 Level of SCxTXD pin after the last bit is output in I/O interface mode

The level of SCxTXD pin after the data hold time is passed after the last bit is output is specified by SCxCR<TIDLE>.

When SCxCR<TIDLE> is "00", the level of SCxTXD pin is output "Low" level. When SCxCR<TIDLE> is "01", the level of SCxTXD pin is output "High" level. When SCxCR<TIDLE> is "10", the level of SCxTXD pin is output the level of the last bit.

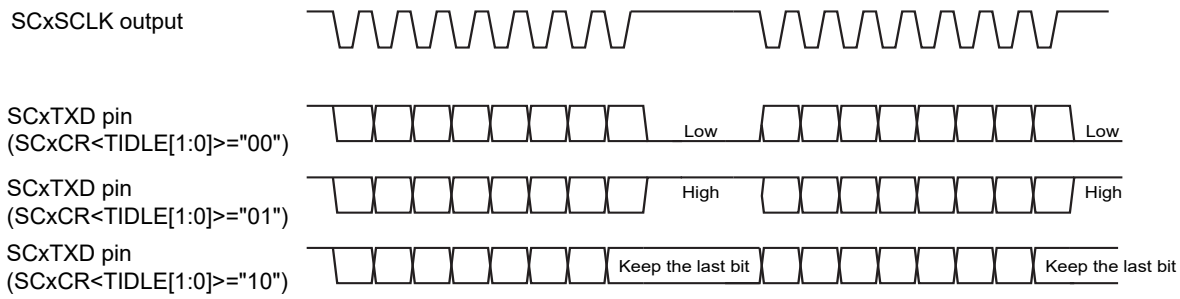


Figure 18-9 Level of SCxTXD pin After the last bit is output

18.11.3.5 Under-run error

In the I/O interface mode with clock input mode and if FIFO is empty 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".

The level of a SCxTXD pin can be specified by SCxCR<TXDEMP>. When SCxCR<TXDEMP> is "0", a SCxTXD pin outputs "Low" level during data output period. When SCxCR<TXDEMP> is "1", a SCxTXD pin outputs "High" level.

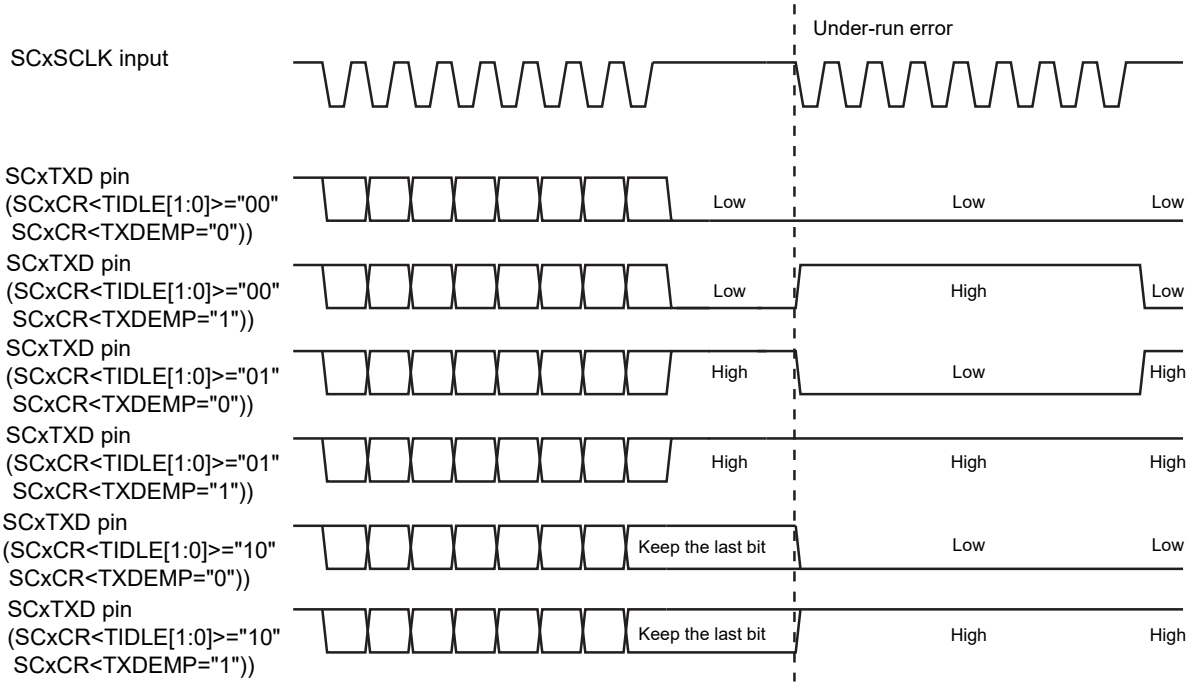


Figure 18-10 Level of SCxTXD pin when Under-run Error is Occurred

In the I/O interface mode with SCLK output setting, the clock output automatically stops, so SCxCR<PERR> has no meaning.

Note: Before switching the I/O interface mode with clock output mode to other modes, read the SCxCR and clear the under-run flag.

18.11.3.6 Data Hold Time In the I/O interface mode with clock input mode

In the I/O interface mode with clock input mode, a data hold time of the last bit can be adjusted by SCxCR<EHOLD[2:0]>. Specify a data hold time and the period of the SCLK to satisfy the following formula.

The data hold time of the last bit ≤ The period of SCLK / 2

18.12 Handshake function

The function of the handshake is to enable frame-by-frame data transmission by using the $\overline{\text{SCxCTS}}$ (Clear to send) pin and to prevent over-run errors. This function can be enabled or disabled by SCxMOD0<CTSE>.

When the $\overline{\text{SCxCTS}}$ pin is set to "High" level, the current data transmission can be completed but the next data transmission is suspended until the $\overline{\text{SCxCTS}}$ pin returns to the "Low" level. 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" level 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" level.

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.

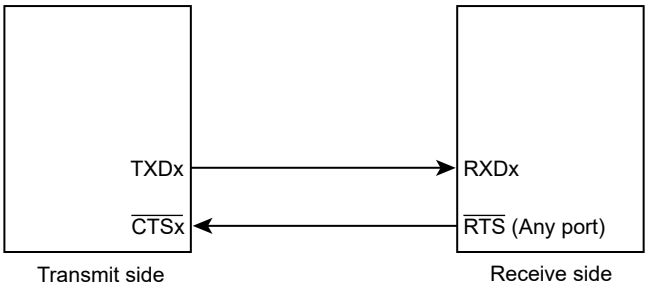


Figure 18-11 Handshake Function

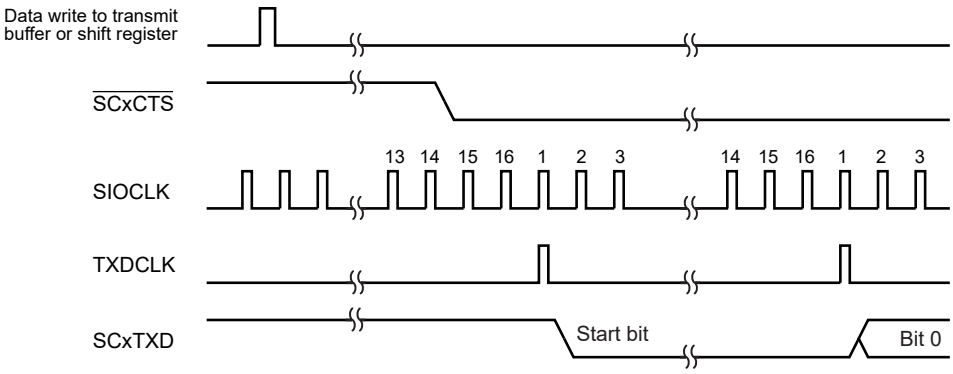


Figure 18-12 $\overline{\text{SCxCTS}}$ Signal timing

18.13 Interrupt/Error Generation Timing

18.13.1 Receive Interrupts

Figure 18-13 shows the data flow of receive operation and the route of read.

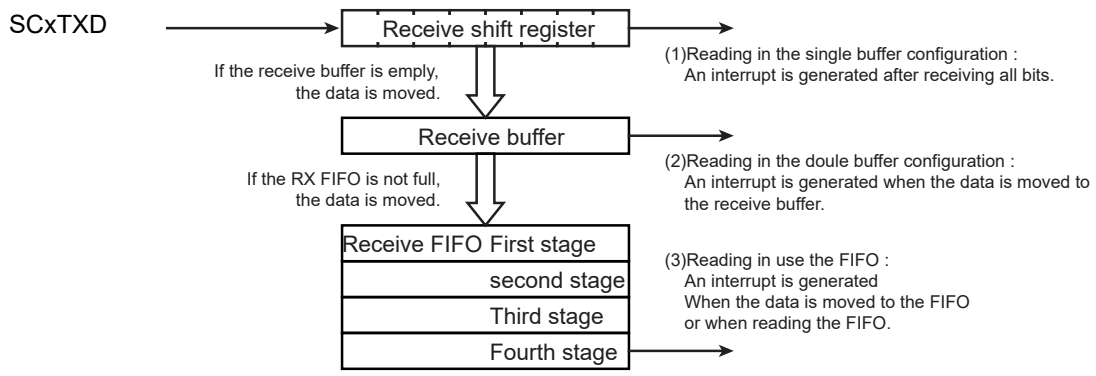


Figure 18-13 Receive Buffer/FIFO Configuration Diagram

18.13.1.1 Single Buffer / Double Buffer

Receive interrupts are generated at the time depends on the transfer mode and the buffer configurations, which are given as follows.

Table 18-7 Receive Interrupt Conditions in use of Single Buffer / Double Buffer

Buffer Configurations	UART modes	IO interface modes
Single Buffer	–	Immediately after the raising / falling edge of the last SCxSCLK pin (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Double Buffer	A receive interrupt occurs when data is transferred from the receive shift register to the receive buffter.Specific timings are : ・If data does not exist in the receive buffer, a receive interrupt occurs in the vicinity of the center of the 1st stop bit. ・If data exists in both the receive shift register and the receive buffer, a receive interrupt occurs when the buffer is read.	A receive interrupt occurs when data is transferred from the receive shift register to the receive buffer.Specific timings are: ・If data does not exit int the receive buffer, a receive interrupt occurs immediately after on rising/falling edge of SCxSCLK pin of the last bit. (The setting of rising edge or falling edge is specified with SCxCR<SCLKS>.) ・If data exists in both the receive shift register and the receive buffer, a receive interrupt occurs when the buffer is read.

Note: Interrupts are not generated when an over-run error is occurred.

18.13.1.2 FIFO

When the FIFO is used, a receive interrupt occurs on depending on the timing described in Table 18-8 and the condition specified with SCxRFC<RFIS>.

Table 18-8 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 transfer a received data 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 read a receive data from receive FIFO ・When transfer a received data from receive buffer to receive FIFO

18.13.2 Transmit interrupts

Figure 18-14 shows the data flow of transmit operation and the route of read.

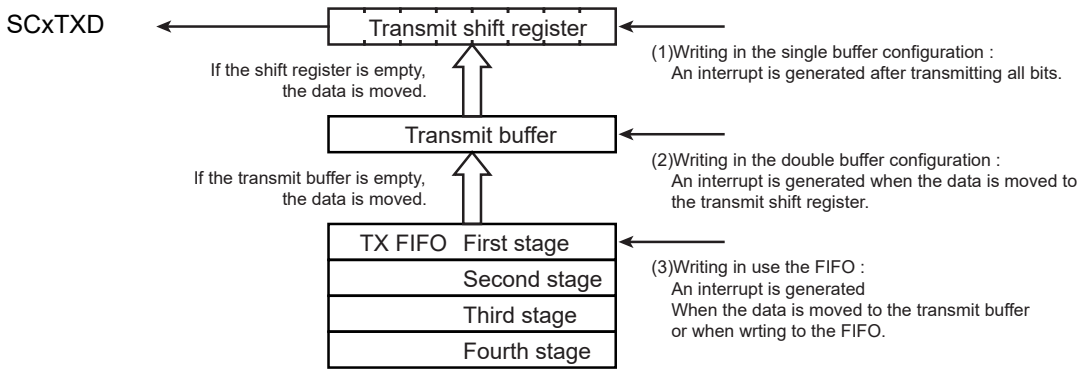


Figure 18-14 Transmit Buffer / FIFO Configuration Diagram

18.13.2.1 Singe Buffer / Double Buffer

Transmit interrupts are generated at the time depends on the transfer mode and the buffer configurations, which are given as follows.

Table 18-9 Transmit Interrupt conditions in use of Single Buffer/Double Buffer

Buffer Configurations	UART modes	IO interface modes
Single Buffer	Just before the stop bit is sent	Immediately after the raising / falling edge of the last SCxSCLK pin (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. If "1" is set to SCxMOD1<TXE> and the transmit shift register is empty, a transmit interrupt occurs because data is immediately transferred to the transmit shift register from the transmit buffer.	

18.13.2.2 FIFO

When the FIFO is used, a transmit interrupt occurs depending on the timing described in Table 18-10 and the condition specified with SCxTFC<TFIS>.

Table 18-10 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 transmit data is write into transmit FIFO ・When transmitted data is transferred from transmit FIFO to transmit buffer

18.13.3 Error Generation

18.13.3.1 UART Mode

Error	9 bits	7 bits 8 bits 7 bits + Parity 8 bits + Parity
Framing Error over-run Error	Around the center of stop bit	
Parity Error	–	Detection: Around the center of parity bit Flag change: Around the center of stop bit

18.13.3.2 I/O Interface Mode

over-run Error	Immediately after the raising / falling edge of the last SCxSCLK pin (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Under-run Error	Immediately after the rising or falling edge of the next SCxSCLK pin. (Rising or falling is determined according to SCxCR<SCLKS> setting.)

Note: Over-run error and Under-run error have no meaning in clock output mode.

18.14 DMA Request

DMA transfer can be started at the timing of interrupt request.

Please refer to the chapter of "product information" for the channel which can be used for a DMA request with this product.

Note 1: In case using DMA transfer by transmit or receive interrupt request, enabled DMA and set transmit and receive registers after generating software reset by SCxMOD<SWRST>.

Note 2: When the DMA transfer is used, the FIFO cannot be used.

18.15 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 and the transmit circuit become initial state.

Other states are maintained.

18.16 Operation in Each Mode

18.16.1 Mode 0 (I/O interface mode)

The I/O interface mode is selected by setting SCxMOD<SM[1:0]> to "00".

Mode 0 consists of two modes, the clock output mode to output synchronous clock (SCLK) and the clock input mode to accept synchronous clock (SCLK) from an external source.

The operation with disabling a FIFO in each mode is described below. Regarding a FIFO, refer to a receive FIFO and a transmit FIFO which are described before.

18.16.1.1 Transmit

(1) Clock Output Mode

- If the transmit double buffer is disabled (SCxMOD2<WBUF> = "0")

  Data is output from the SCxTXD pin and the clock is output from the SCxSCLK 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")

  When data is written to the transmit buffer and the shift register is empty, or when data transmission from the shift register is completed, data is transferred to the shift register from the transmit buffer. Simultaneously, SCxMOD2<TBEMP> is set to "1", and the INTTXx interrupt is generated.

  If the transmit buffer has no data to be moved to the transmit shift register, INTTXx interrupt is not generated and the clock output stops.

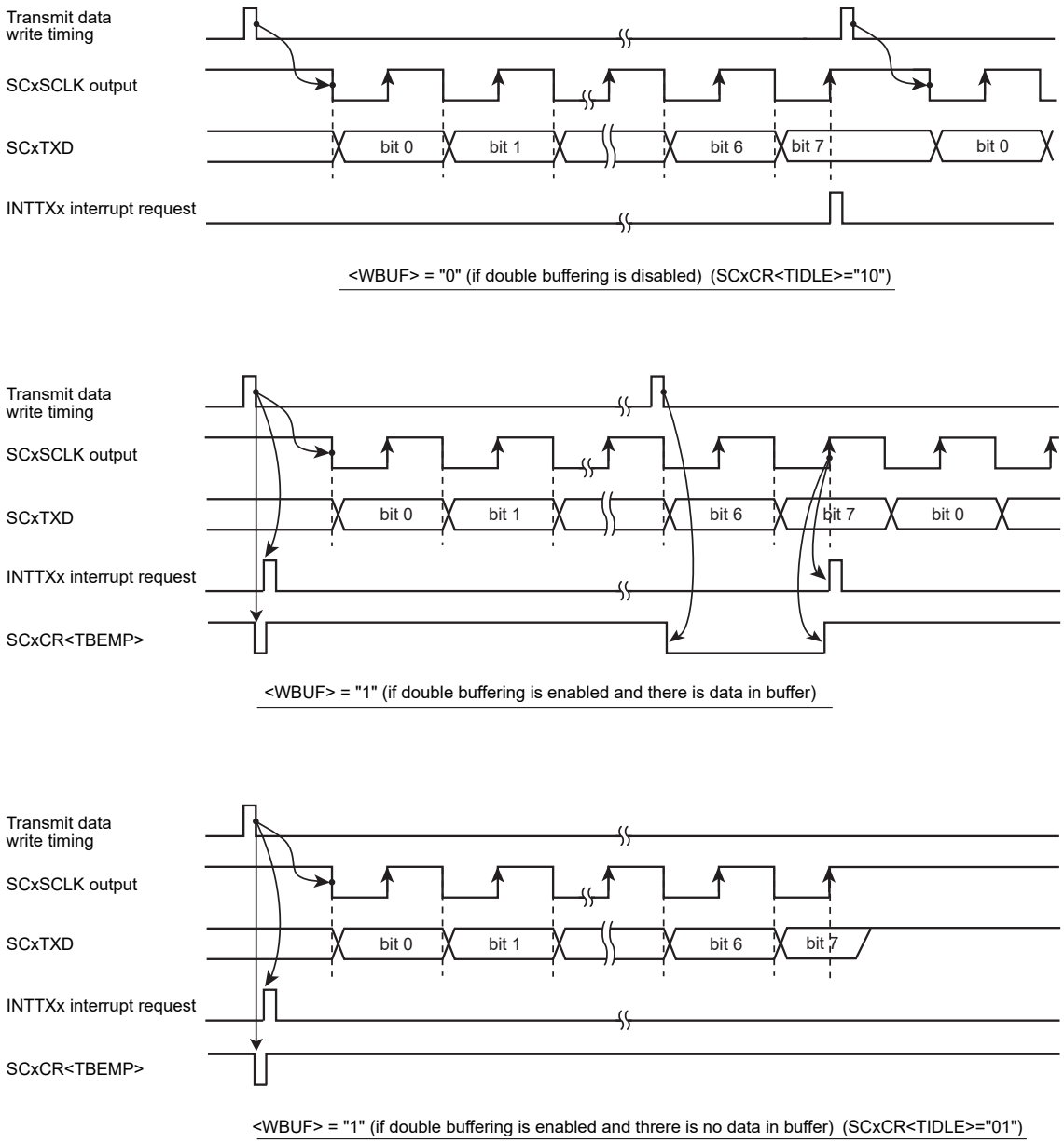


Figure 18-15 Transmit Operation in the I/O Interface Mode (Clock Output Mode)

(2) Clock Input Mode

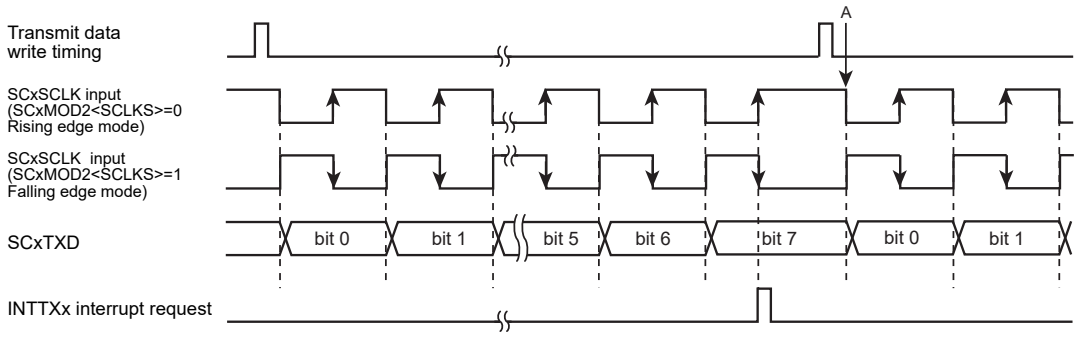
- If double buffering is disabled (SCxMOD2<WBUF> = "0")

  If the clock is input in the condition where data is written in the transmit buffer, 8-bit data is output from the SCxTXD pin. When all data is output, an interrupt INTTXx is generated. The next transmit data must be written before the timing of point "A" as shown in Figure 18-16.

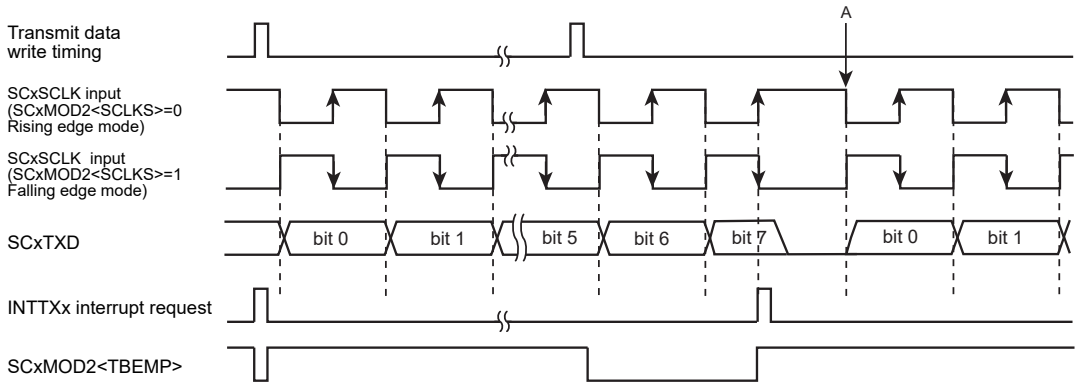
- If double buffer is enabled (SCxMOD2<WBUF> = "1")

  Data is moved from the transmit buffer to the transmit shift register when the CPU writes data to the transmit buffer before the clock input becomes active or when data transmission from the transmit shift register is completed. Simultaneously, SCxMOD2<TBEMP> is set to "1", and the INTTXx interrupt is generated.

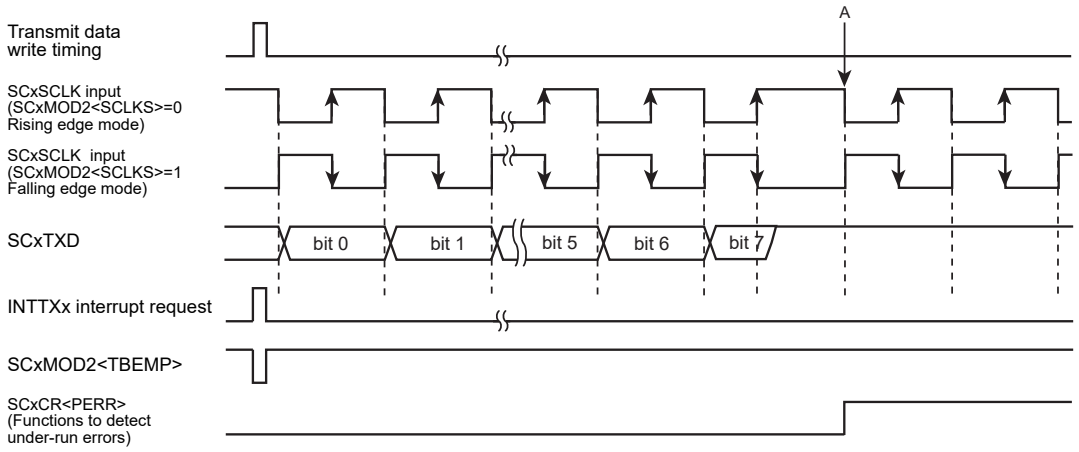
  If the clock input becomes active while no data is in the transmit buffer, although the internal bit counter is started, an under-run error occurs and the level which is specified by SCxCR<TXDEMP> is output to SCxTXD pin.



<WBUF> = "0" (if double buffering is disabled) (SCxCR<TILDE>="10")



<WBUF> = "1" (if double buffering is enabled and there is data in buffer2) (SCxCR<TILDE>="00")



<WBUF> = "1" (if double buffering is enabled and there is no data in buffer2) (SCxCR<TXDEMP><TILDE>="100")

Figure 18-16 Transmit Operation in the I/O Interface Mode (Clock Input Mode)

18.16.1.2 Receive

(1) Clock Output Mode

The clock output starts by setting the receive enable bit SCxMOD0<RXE> to "1".

- If double buffer is disabled (SCxMOD2<WBUF> = "0")

  A clock is output from the SCxSCLK 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, SCxMOD2<RBFLL> is set to "1" and the INTRXx is generated.

  When a data is in the receive buffer, if the data is not read from the receive buffer before completing reception of the next 8 bits, the INTRXx interrupt is not generated and the clock 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.

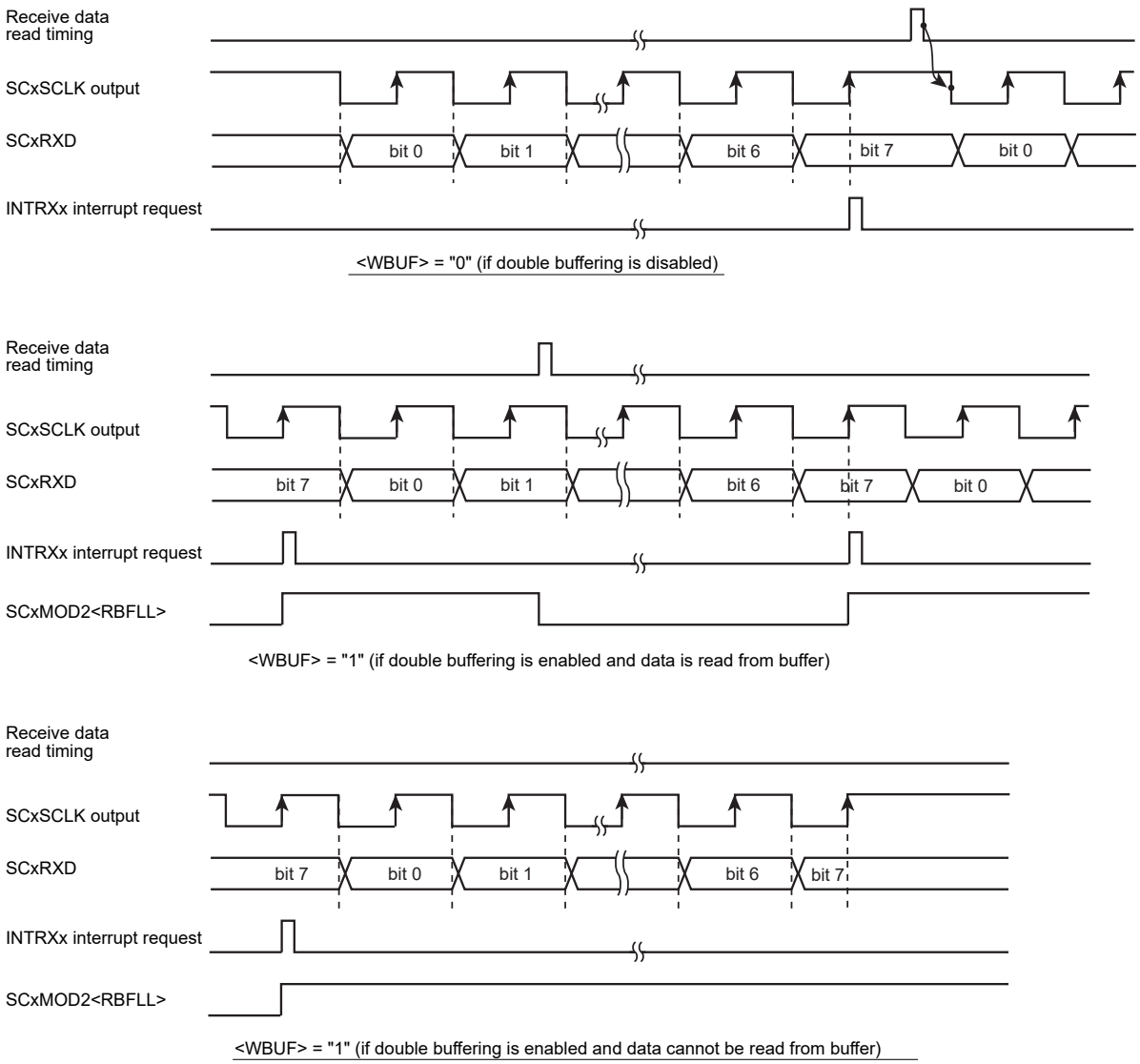


Figure 18-17 Receive Operation in the I/O Interface Mode (Clock Output Mode)

(2) clock input mode

In the clock input mode, receiving double buffering is always enabled, the received data can be moved to the receive buffer from the shift register, and the receive buffer can receive the next data successively.

The INTRXx receive interrupt is generated each time received data is moved to the receive buffer.

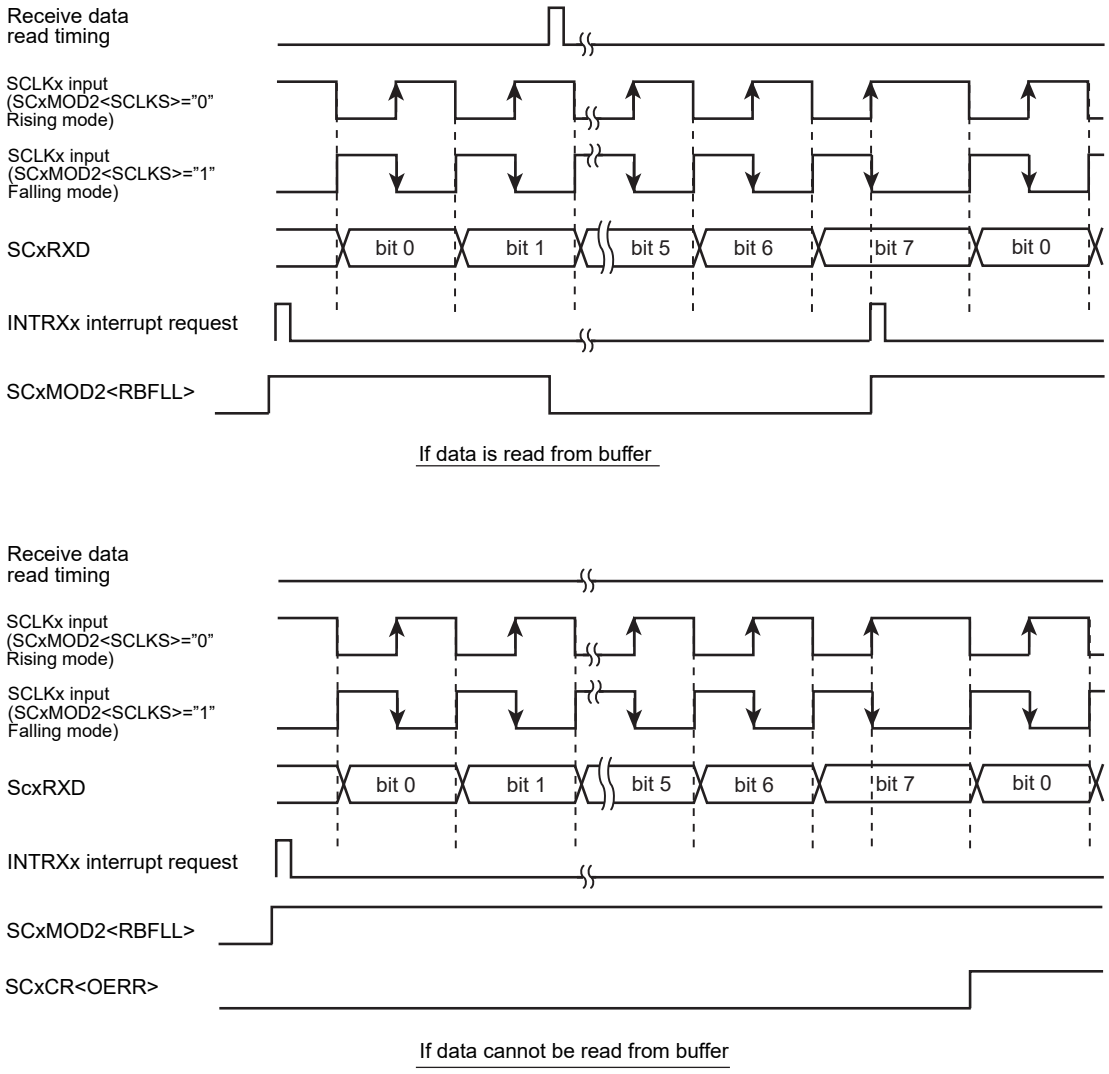


Figure 18-18 Receive Operation in the I/O Interface Mode (Clock Input Mode)

18.16.1.3 Transmit and Receive (Full-duplex)

(1) Clock Output Mode

- If double buffers are disabled (SCxMOD2<WBUF> = "0")

  Clock is output when the CPU writes data to the transmit buffer.

  Subsequently, a data is shifted into receive buffer and the INTRXx is generated. Concurrently, a data written to the transmit buffer is output from the SCxTXD pin, the INTTXx is generated when transmission of all data has been completed. Then, the clock 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 double buffers are enabled (SCxMOD2<WBUF> = "1")

  Clock is outputted when the CPU writes data to the transmit buffer.

  A data is shifted into the receive shift register, moved to the receive buffer, and the INTRXx is generated. While a data is received, a transmit data is output from the SCxTXD pin. When all data are sent out, the INTTXx 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<RBFLL> = "1"), the clock output stops. When both conditions, receive data is read and transmit data is written, are satisfied, the clock output is resumed and the next round of data transmission and reception is started.

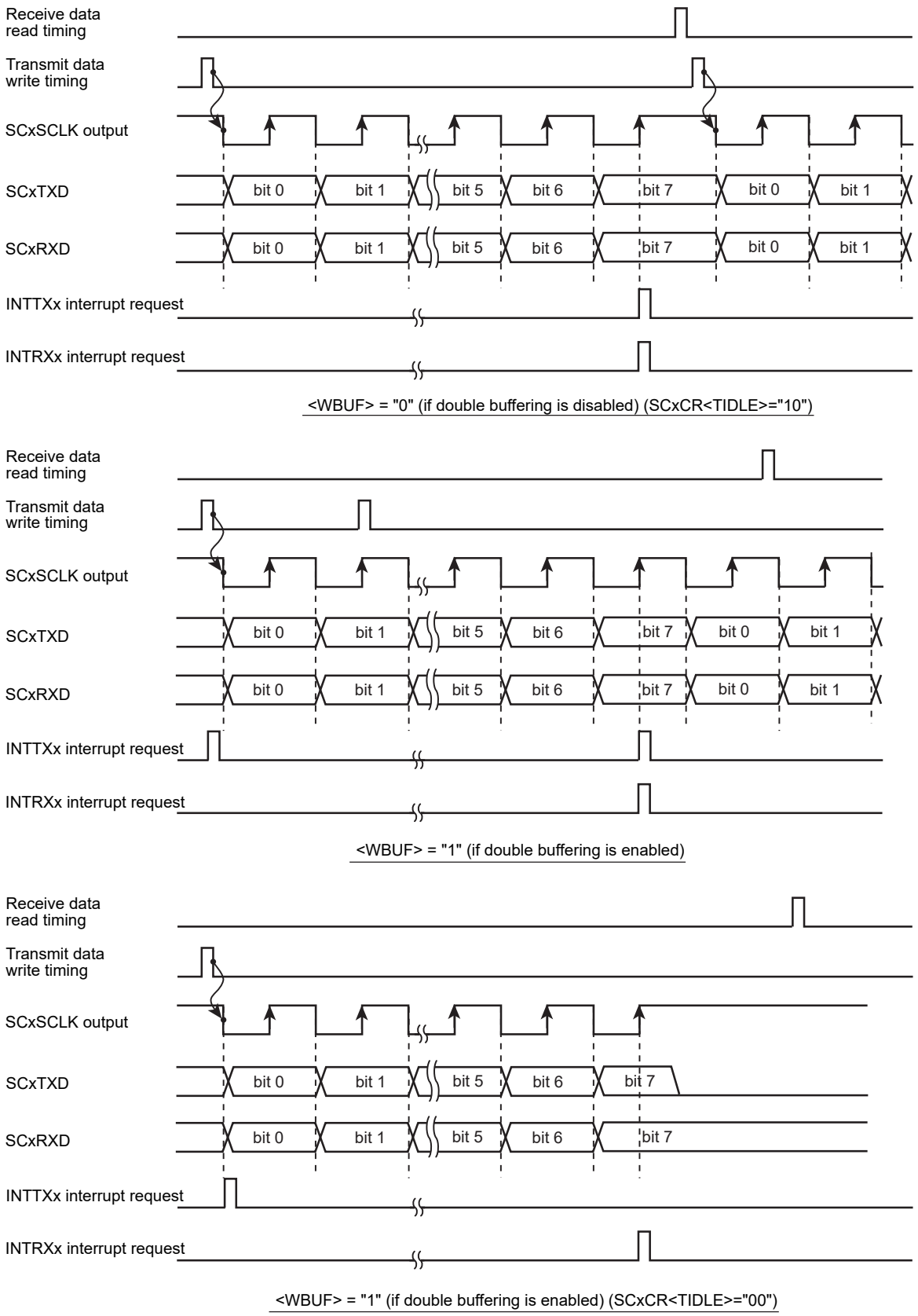


Figure 18-19 Transmit/Receive Operation in the I/O Interface Mode (SCLK Output Mode)

(2) Clock Input Mode

- If double buffers are disabled. (SCxMOD2<WBUF> = "0")

  When receiving data, double buffer is always enabled regardless of the SCxMOD2 <WBUF> settings.

  A data written in the transmit buffer is outputted from the SCxTXD pin and a data is shifted into the receive buffer when the clock input becomes active. The INTTXx is generated upon completion of data transmission. The INTRXx 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 clock input for the next data (data must be written before the point A in Figure 18-20). Data must be read before completing reception of the next data.

- If double buffers are enabled. (SCxMOD2<WBUF> = "1")

  The INTTXx 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 is generated.

  Note that transmit data must be written into the transmit buffer before the clock input for the next data (data must be written before the point A in Figure 18-20). Data must be read before completing reception of the next data.

  Upon the clock input for the next data, 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 data is received, an overrun error occurs.

  If there is no data written to transmit buffer when clock for the next data is input, an underrun error occurs. The level which is specified by SCxCR<TXDEMP> is output to SCxTXD pin.

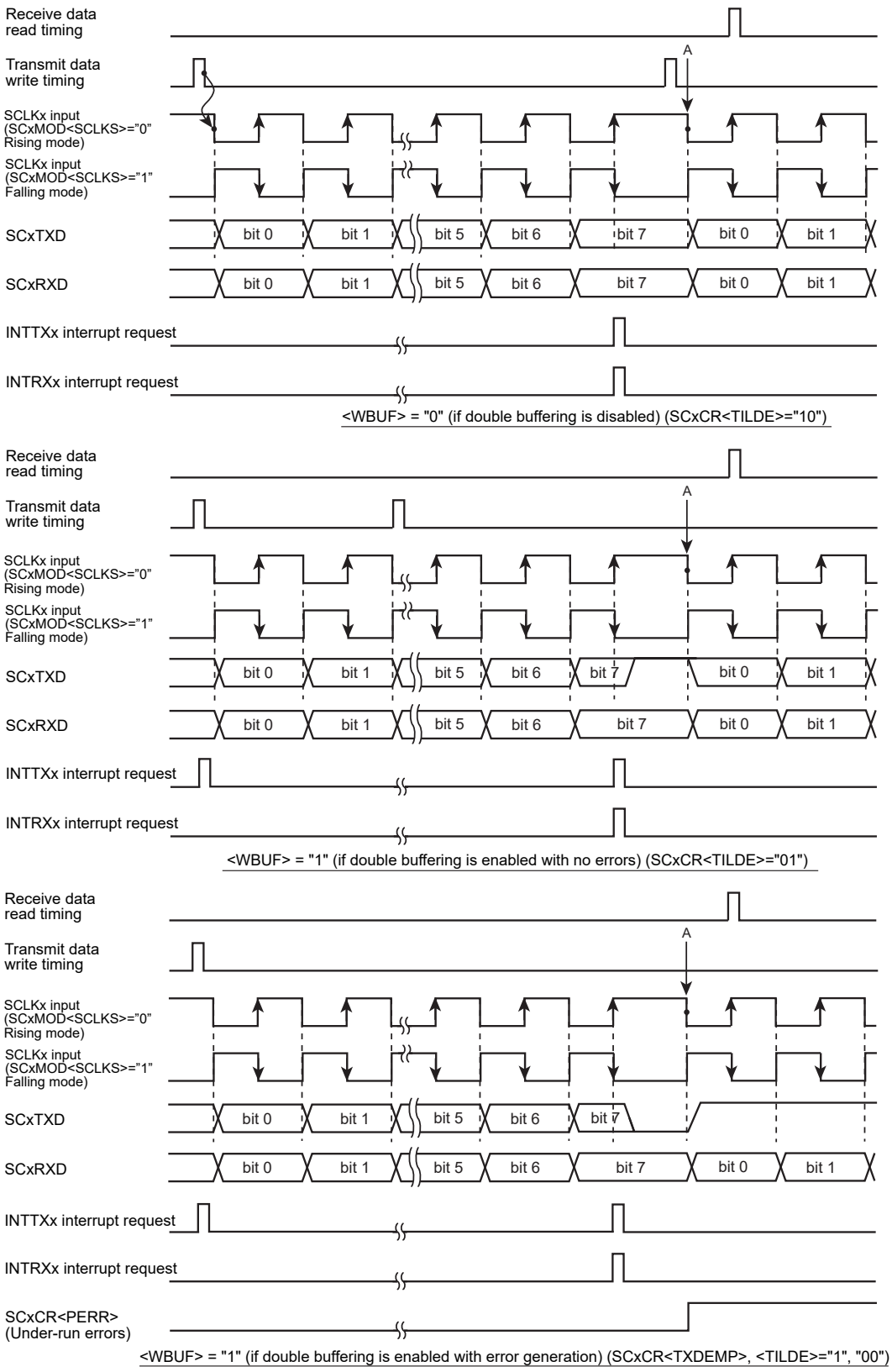


Figure 18-20 Transmit/Receive Operation in the I/O Interface Mode (Clock Input Mode)

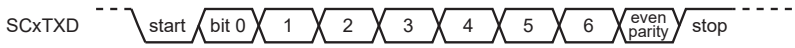
18.16.2 Mode 1 (7-bit UART mode)

The 7-bit UART mode is selected by setting SCxMOD<SM[1:0]> to "01".

In this mode, parity bits can be added to the transmit data stream; 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>. 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.



(Transmittion rate of 2400 bps @fsys = 9.8304 MHz)

Clocking conditions		
	system clock:	High-speed (fc)
	High-speed clock gear:	x 1 (fc)
	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

18.16.3 Mode 2 (8-bit UART mode)

The 8-bit UART mode is 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:

(Reception rate of 9600 bps @fc = 9.8304 MHz)

Clocking conditions		
	System clock:	High-speed (fc)
	High-speed clock gear:	x 1 (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

18.16.4 Mode 3 (9-bit UART mode)

The 9-bit UART mode is 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 SCxMOD0<TB8> for transmitting data. The data is stored in SCxCR<RB8> for receiving data.

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>.

18.16.4.1 Wakeup function

In the 9-bit UART mode, slave controllers can be operated in the wake-up mode by setting SCxMOD0<WU> to "1".

In this case, the interrupt INTRXx will be generated only when SCxCR<RB8> is set to "1".

Note: The SCxTXD pin of the slave controller must be set to the open drain output mode using the PxOD.

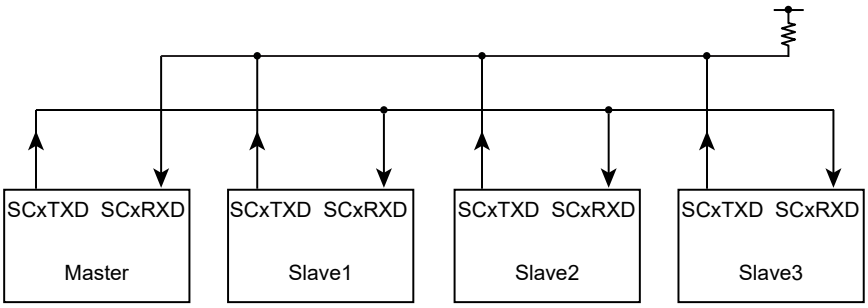


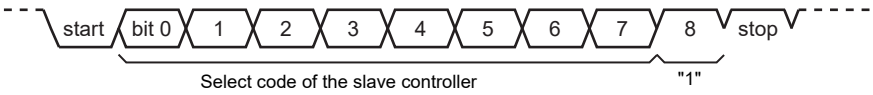
Figure 18-21 Serial Links to Use Wake-up Function

18.16.4.2 Protocol

1. Select the 9-bit UART mode for the master and slave controllers.

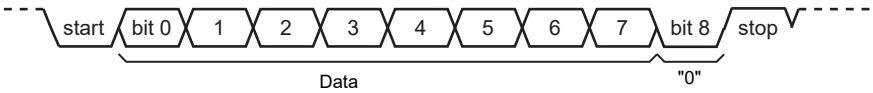
2. Set SCxMOD<WU> to "1" for the slave controllers to make them ready to receive data.

3. The master controller is to transmit a single frame of data that includes the slave controller select 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 controller's own select code, it clears the <WU> to "0".

5. The master controller transmits data to the designated slave controller (the controller of which SCxMOD<WU> 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> set to "1" ignore the receive data because the most significant bit (bit 8) <RB8> is set to "0" and thus no interrupt (INTRXx) is generated. Also, the slave controller with the <WU> set to "0" can transmit data to the master controller to inform that the data has been successfully received.

19. Universal Asynchronous Receiver-Transmitter Circuit (UART)

19.1 Outline

The universal asynchronous receiver-transmitter circuit provides the following the following features:

- Transmit/receive data format
  - Data length: selectable from 5, 6, 7 or 8 bits
  - With/without parity
  - STOP bit length : 1bit or 2 bits selectable
- FIFO
  - Transmission : 8-bit width/ 32-deep, Reception : 12-bit width/ 32-deep
  - Enable/disable decision possible
- Interrupt function
  - Multiple interrupt event outputs
  - Each interrupt permission can be specified.
- Baud-rate generator

  Transmit and receive common clock can be generated using fsys.
- Support DMA
- IrDA 1.0 Function
  - Max data rate : 115.2 kbps (half-duplex).
  - support low power mode
  - IrDA control pin

    Corresponds following control pins.

    UTxTXD(UTxIROUT)

    UTxRXD(UTxIRIN)
- Modem control pin

  Corresponds following control pins.
  - $\overline{\text{UTxCTS}}$
  - UTxRIN
  - $\overline{\text{UTxRTS}}$
  - UTxDCD
  - UTxDSR
  - UTxDTR
- Hardware flow control can be set by RTS or CTS.

19.2 Structure

Figure 19-1 shows a block diagram of UART.

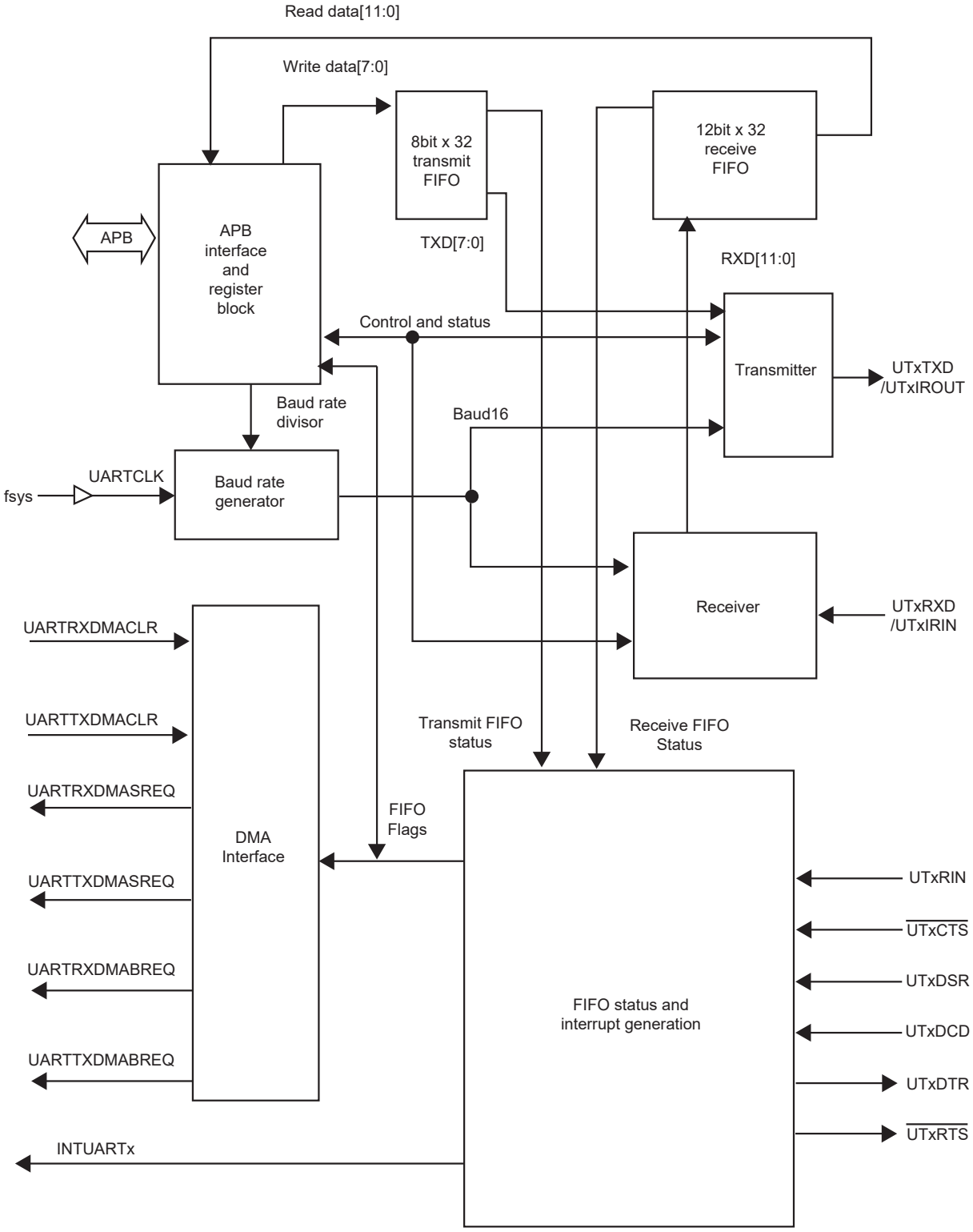


Figure 19-1 Block diagram of UART

19.3 Registers

19.3.1 List of Registers

The following table lists the control registers and addresses.

For base addresses, refer to "Address lists of peripheral functions" in Chapter "Memory Map".

Register name		Address (Base+)
Data register	UARTxDR	0x0000
Receive status register	UARTxRSR	0x0004
Error clear register	UARTxECR	0x0004
Flag register	UARTxFR	0x0018
IrDA low-power counter	UARTxILPR	0x0020
Integer baud rate register	UARTxIBRD	0x0024
Fractional baud rate register	UARTxFBRD	0x0028
Line control register	UARTxLCR_H	0x002C
Control register	UARTxCR	0x0030
interrupt FIFO level select register	UARTxIFLS	0x0034
Interrupt mask set/clear register	UARTxIMSC	0x0038
Raw interrupt status register	UARTxRIS	0x003C
Masked interrupt status register	UARTxMIS	0x0040
Interrupt clear register	UARTxICR	0x0044
DMA control register	UARTxDMACR	0x0048

Note 1: Access the registers by using word (32 bit) reads and word writes.

Note 2: When control registers are re-set, disable UART to operate. If the operation is disabled during receiving or transmitting, UART will stop after on-going transmission is complete.

19.3.2 UARTxDR (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	-	-	-	-	OE	BE	PE	FE
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
31-12	–	R	Read as "0".
11	OE	R	Overrun error 0 : No error 1 : Error If FIFO has been full when receiving data, this bit is set to "1". When FIFO has an empty space in the FIFO and a new data can be written to FIFO, this bit is cleared to "0".
10	BE	R	Break error 0 : No error 1 : Error Break condition (RXDx input was held "low" for longer than a full-word transmission time defined as start, data, parity and stop bits) is detected, this bit is set to "1". If FIFO is enabled, this error is stored at the top of FIFO. If a break error occurs, "0" is stored in FIFO as data. Next data reception is enabled after RXDx input is "1" (marking state) and the start bit is received.
9	PE	R	Parity error 0 : No error 1 : Error When this bit is set to "1", received data parity does not match with the parity programmed with UARTxLCR_H<EPS> and <SPS>. If FIFO is enabled, this error is stored at the top of FIFO.
8	FE	R	Framing error 0 : No error 1 : Error When this bit is set to "1", this indicates that received data does not include a valid stop bit. (A valid stop bit length is "1".) If FIFO is enabled, this error is stored at the top of FIFO.
7-0	DATA[7:0]	R/W	[Read] Receive data [Write] Transmit data

Note: Error status can be identified by reading UARTxRSR as well.

19.3.3 UARTxRSR (Receive Status Register)

Both UARTxRSR and UARTxECR register are mapped on the same addresses.

	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	-	-	-	-	OE	BE	PE	FE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	R	Read as "0".
3	OE	R	Overrun error 0 : No error 1 : Error When data is received, if FIFO has already been full, this bit is set to "1". This bit is cleared to "0" by writing data to UARTxECR. If FIFO is full, further data cannot be written. Thus, the content of FIFO is valid and only the content of shift register is overwritten. CPU must be read data in order to empty FIFO.
2	BE	R	Break error 0 : No error 1 : Error If break condition (RXDx input is held "low" for longer than a full-word transmission time defined as start, data, parity, and stop bits, is detected, this bit is set to "1". This bit is cleared to "0" by writing data to UARTxECR. If FIFO is enabled, this error is input to the top of FIFO. If a break error occurs, "0" is input to FIFO as data. In the next data reception, RXDx input is set to "1" (marking status), this bit is enabled after a start bit is received.
1	PE	R	Parity error 0 : No error 1 : Error When this bit is "1", this indicates that the parity of received data does not match with the parity set in UARTxLCR_H<EPS> and <SPS>. This bit is cleared to "0" by writing data to UARTxECR. If FIFO is enabled, this error is input to the first deep of FIFO.
0	FE	R	Framing error 0 : No error 1 : Error If this bit is set to "1", this indicates that a valid stop bit is not included in the received data. (A valid stop bit length is "1".) This bit is cleared to "0" by writing data to UARTxECR. If FIFO is enabled, this error is input to the top of FIFO.

Note 1: Overrun error is immediately set when an error occurs.

Note 2: UARTxRSR is updated when data is read from UARTxDR. So received data must be read from UARTxDR before an error status is read from UARTxRSR. This read sequence cannot be reversed. In addition, an error status can be read by reading UARTxDR.

19.3.4 UARTxECR (Error 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	-	-	-	-	OE	BE	PE	FE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	R	Read as "0".
3	OE	W	When data is written to UARTxECR, each framing, parity, break and overrun error are cleared. This clearing is executed regard less of a value of data. The address of this register is the same as those of UARTxRSR register.
2	BE	W	
1	PE	W	
0	FE	W	

19.3.5 UARTxFR (UART 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	-	-	-	-	-	-	-	RI
After reset	0	0	0	0	0	0	0	Undefined
	7	6	5	4	3	2	1	0
bit symbol	TXFE	RXFF	TXFF	RXFE	BUSY	DCD	DSR	CTS
After reset	0	0	0	0	0	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-9	–	R	Read as an undefined value.
8	RI	R	RIng indicator flag. 0 : Modem status input = 1 1 : Modem status input = 0
7	TXFE	R	When UARTxLCR_H<FEN>="1" 0 : Transmit FIFO is not empty. 1 : Transmit FIFO is empty.
			When UARTxLCR_H<FEN>="0" 0 : Transmit hold register is not empty. 1 : Transmit hold register is empty.
6	RXFF	R	When UARTxLCR_H<FEN>="1" 0 : Receive FIFO is not full. 1 : Receive FIFO is full.
			When UARTxLCR_H<FEN>="0" 0 : Receive hold register is not full. 1 : Receive hold register is full.
5	TXFF	R	When UARTxLCR_H<FEN>="1" 0 : Transmit FIFO is not full. 1 : Transmit FIFO is full.
			When UARTxLCR_H<FEN>="0" 0 : Transmit hold register is not full. 1 : Transmit hold register is full.
4	RXFE	R	When UARTxLCR_H<FEN>="1" 0 : Receive FIFO is not empty. 1 : Receive FIFO is empty.
			When UARTxLCR_H<FEN>="0" 0 : Receive hold register is not empty. 1 : Receive hold register is empty.
3	BUSY	R	UART busy 0 : UART transmission is stopping. 1 : UART transmission is performing. This bit is set to "1" when transmit FIFO becomes not empty regardless of whether UART operation is enabled or not.
2	DCD	R	Data carrier detect (DCD) flag 0 : UTxDCD is "High" 1 : UTxDCD is "Low"

Bit	Bit Symbol	Type	Function
1	DSR	R	Data set ready (DSR) flag 0 : UTxDSR is "High" 1 : UTxDSR is "Low"
0	CTS	R	Clear to send 0 : $\overline{\text{UTxCTS}}$ is "High" 1 : $\overline{\text{UTxCTS}}$ is "Low" A reverse state of $\overline{\text{UTxCTS}}$ pin can be read.

Note: <TXFE> does not indicate the status of shift register.

19.3.6 UARTxILPR (UART IrDA low-power 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	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ILPDVSR							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as 0.
7-0	ILPDVSR [7:0]	R/W	Low-power divisor <ILPDVSR> = (fUARTCLK / fIrLPBaud16). The UARTxILPR register is the IrDA low-power counter register. This is an 8-bit read/write register that stores the low-power counter divisor value used to generate the IrLPBaud16 signal by dividing down of UARTCLK. All the bits are cleared to 0 when reset

Note 1: <Set this register before the UARTxCR<SIRLP> is set to 1.

Note 2: 0x00 setting is prohibited.

19.3.7 UARTxIBRD (UART Integer Baud-rate 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	BAUDDIVINT							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	BAUDDIVINT							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as an undefined value.
15-0	BAUDDIVINT [15:0]	R/W	Integer baud-rate divisor (0x0002 to 0xFFFF) An integer part of baud-rate divisor value

Note 1: A value written to UARTxIBRD will not be valid until current on-going transmission or reception is complete.
Note 2: A value written to UARTxIBRD will be valid when data is written to UARTxLCR_H.
Note 3: Set <BAUDDIVINT[15:0]> before UARTxCR>UARTEN< is set to "1".
Note 4: 0x0000, 0x0001 cannot be set.
Note 5: The value of the worst case baud rate divisor of the set value due to the baud rate shift (total error) between the transmitter side and the receiver side is as shown in the table below.(8 bit data + Parity / 9 bit data)

Total error	BAUDDIVINT(Lower limit)
2.0% or less	0x0002
2.8% or less	0x0003
3.3% or less	0x0004

19.3.8 UARTxFBRD (UART Fractional Baud-rate 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	-	-	BAUDDIVFRAC					
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	–	R	Read as "0".
5 -0	BAUD DIVFRAC [5:0]	R/W	Fractional baud-rate divisor (0x01 to 0x3F) A fractional part of baud-rate divisor value.

Note 1: A value written to UARTxFBRD will not be valid until current on-going transmission or reception is complete.

Note 2: A value written to UARTxFBRD will be valid when data is written to UARTxLCR_H.

Note 3: Set <BAUDDIVFRAC[5:0]> before "1" is set to UARTxCR>UARTEN>.

Note 4: The minimum value of baud-rate divisor is 1 and maximum one is 65535. Therefore, the integer part of baud-rate divisor can not be set to 0. And the fraction part of baud-rate divisor must be set 0, when the integer part of baud-rate divisor is 65535.

19.3.9 UARTxLCR_H (UART Line 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	SPS	WLEN		FEN	STP2	EPS	PEN	BRK
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7	SPS	RW	Stick parity selection 0 : Stick parity disabling 1 : When <EPS> = "0" is set, a parity is sent and checked as "1". When <EPS> = "1" is set, a parity is sent and checked as "0". <SPS> has no meaning when <PEN> is set to "0" and the parity check and generation are disabled. For details of the table of truth value of <SPS>, <EPS> and <PEN>, refer to Table 19-1.
6-5	WLEN[1:0]	RW	Word length 00 : 5bit 01 : 6bit 10 : 7bit 11 : 8bit These bits indicate the number of data bits transmitted/received in the frame.
4	FEN	RW	Selection of FIFO permission 0 : FIFO is disabled (FIFO becomes a 1-deep hold register.) 1 : FIFO is enabled
3	STP2	RW	Selection of transmission stop bit length 0: 1bit 1: 2 bit When receiving, stop bit of 2-bit length is not checked.
2	EPS	RW	Even parity selection 0: Odd parity 1: Even parity Control to selection of a parity bit when transmitting/receiving. If <PEN> is set to "0", when parity check and generation are disabled, this bit has no meaning.
1	PEN	RW	Parity enable 0: Disabled (Parity is disabled. Parity bits is not added.) 1 : Enabled (Parity check and generation are enabled.)
0	BRK	RW	Break transmission selection 0: No break transmission 1: Performs break transmission While <BRK> is set to "1", TXDx will be continued to output "low" after transmission of current on-going frame is complete. To generate break state, <BRK> must be keep "1" for two frame period. If break state is generated, the contents of transmit FIFO is not influenced. When the break is not transmitted, <BRK> must be cleared to "0".

Note: When the contents of UARTxIBRD or UARTxFBRD is updated, always must write UARTxLCR_H in the end.

Table 19-1 Table of truth value UARTxLCR_H <SPS>, <EPS>, <PEN>

Parity enable <PEN>	Even parity selection <EPS>	Stick parity selection <SPS>	Parity selection (Transmission or checking)
0	×	×	No transmission and check
1	1	0	Even parity
1	0	0	Odd parity
1	0	1	Transmit /receive "1"
1	1	1	Transmit/receive "0"

19.3.10 UARTxCR (UART 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	CTSEN	RTSEN	-	-	RTS	DTR	RXE	TXE
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	SIRLP	SIREN	UARTEN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as an undefined value.
15	CTSEN	RW	CTS hardware flow control enable 0 : Disabled 1 : Enabled When <CTSEN> is set to "1", CTS hardware flow control is enabled. Data is sent only when $\overline{\text{UTxCTS}}$ pin is "L".
14	RTSEN	RW	RTS hardware flow control enable 0 : Disabled 1 : Enabled When <RTSEN> is set to "1", RTS hardware flow control is enabled. Data is required when receive FIFO is empty.
13-12	-	R	Read as an undefined value.
11	RTS	RW	Complement of the UART Request To Send (RTS) modem status output : 0 : Modem status output is 1 1 : Modem status output is 0 This bit is the inverting UART Request To Send (RST) modem status output signal. When this bit is set to 1, the output is 0.
10	DTR	RW	Complement of the UART Data Set Ready (DTS) modem status output : 0 : Modem status output is 1 1 : Modem status output is 0 This bit is the inverting UART Data Transmit Ready (DTR) modem status output signal. When this bit is set to 1, the output is 0.
9	RXE	RW	Reception permission setting 0 : Disabled 1 : Enabled When <RXE> is set to "1", reception is enabled. According to a value of <SIREN>, data is received using UART function or SIR function. If reception is disabled during receiving, reception stops after current on-going data reception is complete.
8	TXE	RW	Transmission permission setting 0 : Disabled 1 : Enabled When <TXE> is set to "1", transmission is enabled. According to a value of <SIREN>, data is received using UART function or SIR function. If transmission is disabled during transmitting, transmission stops after current on-going transmission is complete.
7	-	RW	Write as "0".
6-3	-	R	Read as an undefined value.

Bit	Bit Symbol	Type	Function
2	SIRLP	RW	IrDA encoding mode select for transmitting 0 bit : 0 : 0 bit are transmitted as an active high pulse of 3/16th of the bit period. 1 : 0 bit are transmitted with a pulse width that is 3 times the period of the IrLPBaud16 input signal. <SIRLP> selects IrDA encoding mode. When this bit is cleared to 0, 0 bits of the IrDA transmission data are transmitted as an active high pulse (UTxIROUT) with a width of 3/16th of the bit period. When this bit is set to 1, 0 bits of the IrDA transmission data are transmitted with a pulse width that is 3 times the period of the IrLPBaud16 input signal. Setting this bit can reduce power consumption but might decrease transmission distances.
1	SIREN	RW	SIR enable 0 : Disabled 1 : Enabled When this bit is set to 1, the IrDA circuit is enabled. To use the UART, the <UARTEN> must be set to 1. When the IrDA circuit is enabled, the UTxIROUT and UTxIRIN pins are enabled. The UTxTXD pin remains in the marking state (set to 1). Signal transitions on the UTxRXD pin or modem status input have no effect. When IrDA circuit is disabled, UTxIROUT remains cleared to 0 (no light pulse is generated) and the UTxIRIN pin has no effect.
0	UARTEN	R/W	UART permission setting 0 : Disabled 1 : Enabled When <UARTEN> is set to "0", UART is disabled. If UART is disabled during transmission or reception, UART stops after current on-going data transmission or reception is complete. When <UARTEN> is set to "1", data is sent or received using UART function or SIR function according to a value of <SIREN>.

19.3.11 UARTxIFLS (UART Interrupt FIFO Level 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	-	-	RXIFLSEL			TXIFLSEL		
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	–	R	Read as an undefined value.
5-3	RXIFLSEL[2:0]	RW	Reception interrupt FIFO level selection 000: Receive FIFO ≥ 1/8 full 001: Receive FIFO ≥ 1/4 full 010: Receive FIFO ≥ 1/2 full 011: Receive FIFO ≥ 3/4 full 100: Receive FIFO ≥ 7/8 full Other than the above settings: Reserved This bit selects the receive FIFO interrupt level. The FIFO level is not a trigger of interrupts. Interrupts is generated by the transition through the specified FIFO level. For example, if FIFO level is set to 1/8 full (4 bytes), an interrupt occurs when the 5th byte data is stored in the receive FIFO.
2-0	TXIFSEL[2:0]	RW	Transmission interrupt FIFO level selection 000: Transmit FIFO ≤ 1/8 full 001: Transmit FIFO ≤ 1/4 full 010: Transmit FIFO ≤ 1/2 full 011: Transmit FIFO ≤ 3/4 full 100: Transmit FIFO ≤ 7/8 full Other than the above settings: Reserved This bit selects the transmit FIFO interrupt level. Interrupts are not generated at reaching to the specified interrupt level. They are generated after the transition at specified interrupt level. For example, if FIFO level is set to 1/8 full (4 bytes), an interrupt occurs when the 4th byte data is read out from the receive FIFO.

19.3.12 UARTxIMSC (UART Interrupt Disable/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	-	-	-	-	-	OEIM	BEIM	PEIM
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	FEIM	RTIM	TXIM	RXIM	DSRMIM	DCDMIM	CTSMIM	RIMIM
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-11	–	R	Read as an undefined value.
10	OEIM	RW	Overrun error interrupt mask 0: Disabled 1: Enabled
9	BEIM	RW	Break error interrupt mask 0: Disabled 1: Enabled
8	PEIM	RW	Parity error interrupt mask 0: Disabled 1: Enabled
7	FEIM	RW	Framing error interrupt mask 0: Disabled 1: Enabled
6	RTIM	RW	Receive timeout interrupt mask 0: Disabled 1: Enabled
5	TXIM	RW	Receive interrupt mask 0: Disabled 1: Enabled
4	RXIM	RW	Receive interrupt mask 0: Disabled 1: Enabled
3	DSRMIM	RW	DSR modem interrupt mask 0: Disabled 1: Enabled
2	DCDMIM	RW	DCD modem interrupt mask 0: Disabled 1: Enabled
1	CTSMIM	RW	CTS modem interrupt mask 0: Disabled 1: Enabled
0	RIMIM	RW	RIN modem interrupt mask 0: Disabled 1: Enabled

19.3.13 UARTxRIS (UART Raw 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	-	-	-	-	-	OERIS	BERIS	PERIS
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	FERIS	RTRIS	TXRIS	RXRIS	DSRRMIS	DCDRMIS	CTSRMIS	RIRMIS
After reset	0	0	0	0	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-11	–	R	Read as an undefined value.
10	OERIS	R	Overrun error interrupt status : 0: No interrupt request 1: Interrupt request.
9	BERIS	R	Break error interrupt status : 0: No interrupt request 1: Interrupt request
8	PERIS	R	Parity error interrupt status : 0: No interrupt request 1: Interrupt request
7	FERIS	R	Framing error interrupt status : 0: No interrupt request 1: Interrupt request
6	RTRIS	R	Receive timeout interrupt status : 0: No interrupt request 1: Interrupt request
5	TXRIS	R	Transmit interrupt status : 0: No interrupt request 1: Interrupt request
4	RXRIS	R	Receive interrupt status : 0: No interrupt request 1: Interrupt request
3	DSRRMIS	R	Read as an undefined value. 0: No interrupt request 1: Interrupt request
2	DCDRMIS	R	DCD modem raw interrupt status 0: No interrupt request 1: Interrupt request
1	CTSRMIS	R	CTS modem interrupt status : 0: No interrupt request 1: Interrupt request
0	RIRMIS	R	RIN modem raw interrupt status 0: No interrupt request 1: Interrupt request

Note: All the bits, except the modem raw status interrupt bits (bits 3 to 0), are cleared to 0 when reset. Themodem status bit are undefined after reset.

19.3.14 UARTxMIS (UART Masked 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	-	-	-	-	-	OEMIS	BEMIS	PEMIS
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	FEMIS	RTMIS	TXMIS	RXMIS	DSRMMIS	DCDMMIS	CTSMMIS	RIMMIS
After reset	0	0	0	0	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-11	–	R	Read as an undefined value.
10	OEMIS	R	Overrun error mask interrupt status: 0: No interrupt request 1: Interrupt request
9	BEMIS	R	Break error mask interrupt status: 0: No interrupt request 1: Interrupt request
8	PEMIS	R	Parity error mask interrupt status: 0: No interrupt request 1: Interrupt request
7	FEMIS	R	Framing error mask interrupt status: 0: No interrupt request 1: Interrupt request
6	RTMIS	R	Receive timeout mask interrupt status: 0: No interrupt request 1: Interrupt request
5	TXMIS	R	Transmit mask interrupt status: 0: No interrupt request 1: Interrupt request
4	RXMIS	R	Receive mask interrupt status: 0: No interrupt request 1: Interrupt request
3	DSRMMIS	R	DSR modem masked interrupt status: 0: No interrupt request 1: Interrupt request
2	DCDMMIS	R	DCD modem masked interrupt status: 0: No interrupt request 1: Interrupt request
1	CTSMMIS	R	CTS modem mask interrupt status: 0: No interrupt request 1: Interrupt request
0	RIMMIS	R	RIN modem masked interrupt status: 0: No interrupt request 1: Interrupt request

Note: All the bits, except the modem masked status interrupt bits (bits 3 to 0), are cleared to 0 when reset. The modem status bits are undefined after reset.

19.3.15 UARTxICR (UART Interrupt 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	-	-	-	-	-	OEIC	BEIC	PEIC
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	FEIC	RTIC	TXIC	RXIC	DSRMIC	DCDMIC	CTSMIC	RIMIC
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-11	–	W	Write as "0".
10	OEIC	W	Overrun error interrupt clear: 0: Invalid 1: Clear
9	BEIC	W	Break error interrupt clear: 0: Invalid 1: Clear
8	PEIC	W	Parity error interrupt clear: 0: Invalid 1: Clear
7	FEIC	W	Framing error interrupt clear: 0: Invalid 1: Clear
6	RTIC	W	Receive timeout interrupt clear: 0: Invalid 1: Clear
5	TXIC	W	Transmit interrupt clear: 0: Invalid 1: Clear
4	RXIC	W	Receive interrupt clear: 0: Invalid 1: Clear
3	DSRMIC	W	DSR modem interrupt clear : 0: Invalid 1: Clear
2	DCDMIC	W	DCD modem interrupt clear : 0: Invalid 1: Clear
1	CTSMIC	W	CTS modem interrupt clear: 0: Invalid 1: Clear
0	RIMIC	W	RIN modem interrupt clear : 0: Invalid 1: Clear

Note: UARTxICR register is interrupt clear register for write-only. If this bit is set to "1", corresponding interrupt is cleared. Writing "0" is invalid.

19.3.16 UARTxDMACR (UART DMA 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	-	-	-	-	-	DMAONERR	TXDMAE	RXDMAE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	–	R	Read as an undefined value.
2	DMAONERR	RW	DMA ON error 0: With 1: Without While this bit is set to "1", if an error occurs during receiving data, a DMA receive request, UARTxRXDMASREQ (UART receive DMA single request), or UARTxRXDMABREQ (UART receive DMA burst request) is disabled.
1	TXDMAE	RW	Transmit DMA enable selection 0: Disabled 1: Enabled
0	RXDMAE	RW	Receive DMA enable selection 0: Disabled 1: Enabled

Note: If transmit/receive FIFO data is transferred using DMAC, set the bus width to 8-bit.

19.4 Operation Description

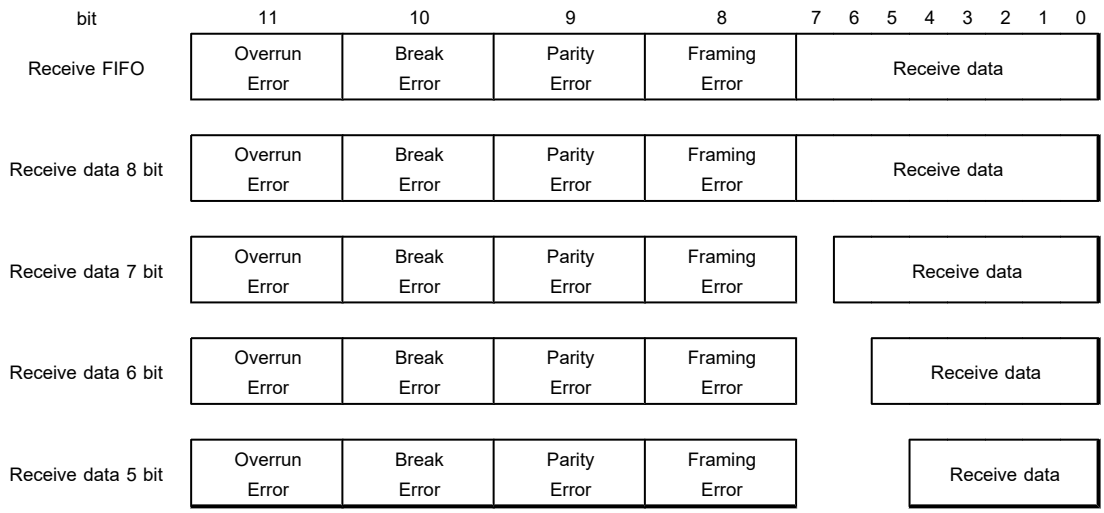
19.4.1 Transmit FIFO and Receive FIFO

19.4.1.1 Transmit FIFO

A transmit FIFO is a memory buffer that contains 8-bit width and 32-deep. CPU data written via APB interface is stacked to this FIFO until a transmit logic reads them. By disabling a transmit FIFO, it can be served as a 1-byte hold register.

19.4.1.2 Receive FIFO

A receive FIFO is a memory buffer that contains 12-bit width and 32-deep. Error bits corresponding to receive data are stacked into receive FIFO using receive logic until CPU reads the data via APB interface. If receive FIFO is disabled, it serve as a 1-byte hold register.



Note: Empty bits of receive data are undefined.

19.4.2 Transmit Data and Receive data

Data written in UARTDR is stacked into transmit FIFO when FIFO is enabled.

If FIFO is disabled to write, the data is transferred to transmit hold register.

By writing data, a transfer starts. Data includes a start bit. If a parity bit is enabled, the data is transmitted with a parity and stop bit.

Received data is 12-bit width and stacked into receive FIFO with the status (break error, framing error, parity error, overrun error) using 4 bits. If FIFO is disabled, received data and the status are transferred to receive hold register.

19.4.3 Baud-rate Generator

The baud-rate generator outputs internal clocks Baud16 and IrLPBaud16. Baud16 generates the timing for UART transmission/reception control. IrLPBaud16 generates a pulse width of IrDA encode transmit bit stream when in the low power mode.

A baud-rate is calculated by the following formula using $f_{UARTCLK}$ input from UART and a baud-rate divisor.

Baud-rate = ($f_{UARTCLK}$) / (16 x baud-rate divisor)

19.4.3.1 Calculating Baud-rate Divisor

The baud-rate divisor value is calculated as follows:

Baud-rate divisor BAUDDIV = ($f_{UARTCLK}$) / (16 x baud rate)

where $f_{UARTCLK}$ is a clock frequency of UART.

BAUDDIV consists of integer part (BAUDDIVINT) and fractional part (BAUDDIVFRAC).

Example: Calculation of the divisor value

Required baud-rate is 230400 and $f_{UARTCLK}$ = 4 MHz

Baud rate divisor = $(4 \times 10^6)/ (16 \times 230400) = 1.085$

Accordingly, the integer part of baud-rate (BAUDDIVINT) = 1 and the part of fractional part of baud-rate = 0.085 are calculated.

Therefore, BAUDDIVFRAC is calculated as follows:

BAUDDIVFRAC = $((0.085 \times 64) + 0.5) = 5.94 = 5$ (Figures below the decimal point are omitted.)

The baud-rate divisor is calculated using above figure of integer part and fractional part as below.

BAUDDIV = 1 + 5/64 = 1.078

At this time, the baud-rate is calculated as follows:

Generated baud-rate = $(4 \times 10^6)/ (16 \times 1.078) = 231911$

A margin of error = $(231911 - 230400)/ 230400 \times 100 = 0.656\%$

In addition, the maximum margin of error = $1/64 \times 100 = 1.56\%$ when UARTxFBRD register is used. This margin of error is generated when UARTxFBRD = 1.

19.4.4 Transmit Logic

The transmit logic performs parallel-to-serial conversion on data read out from transmit FIFO. Control logic outputs the signal beginning with a start bit, data bits with LSB, followed by the parity bit, and the stop bits according to the programmed configuration in control registers.

19.4.5 Receive Logic

The receive logic performs serial-to-parallel conversion on the received bit stream after a star bit has been detected. Overrun, parity, frame error checking and line break detection are also performed. Data related to a error bit of overrun, parity, framing and break is written to receive FIFO.

19.4.6 Interrupt Generation Logic

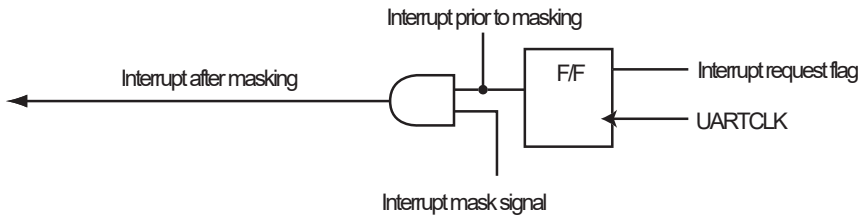
The UART outputs a maskable interrupt according to interrupt events.

19.4.6.1 UART Interrupt Generation Circuit

(1) Interrupt request flag generation circuit

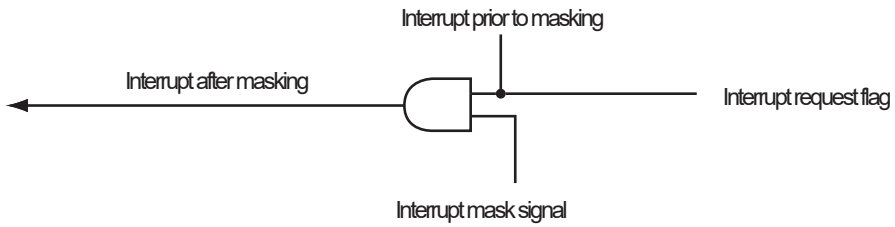
1. Generation circuit of break, parity and framing error flags

   An interrupt request flag is changing in real-time associated with F/F. Each flag is cleared when corresponding interrupt clear register is written.



2. Overrun error flag generation circuit

   An interrupt request flag is changing with overrun errors in real-time. The status is not maintained. An overrun flag is cleared by reading receive FIFO.



(2) UART interrupt

Each masked interrupt status is ORed and output from UART as INTUARTx.

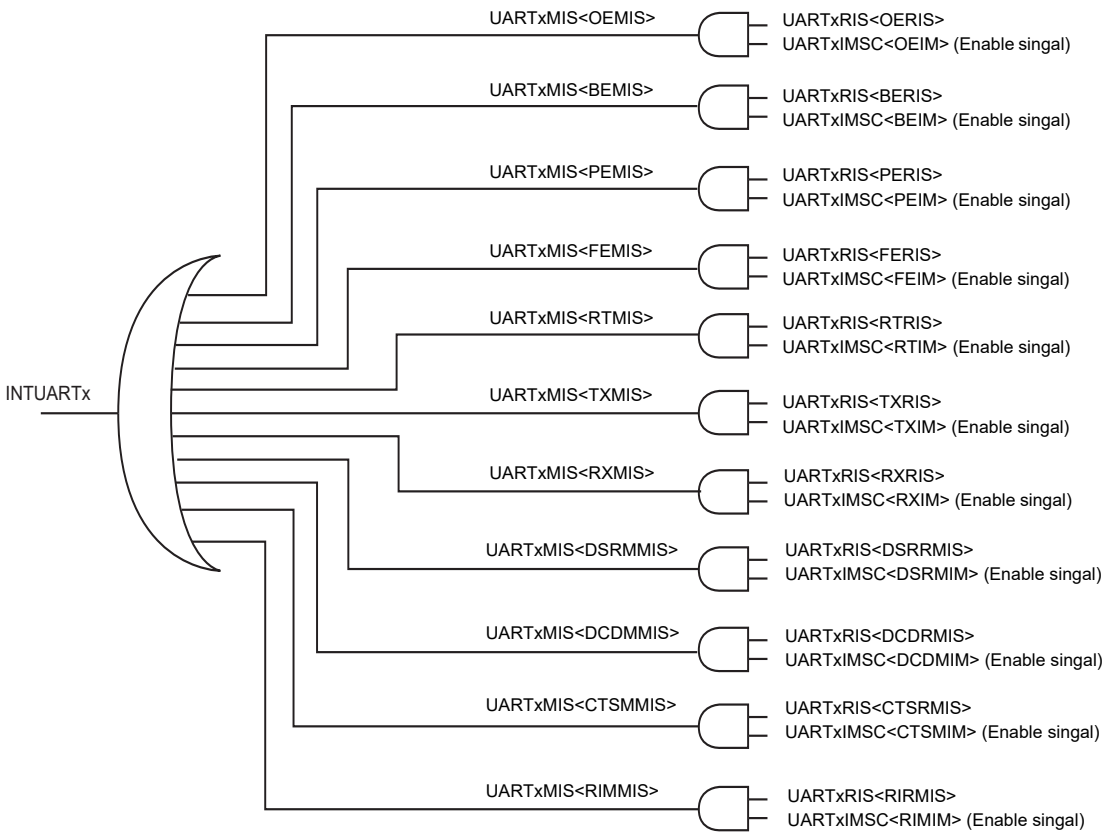


Figure 19-2 UART interrupt block

19.4.6.2 Interrupt Generation Timing

Interrupt source	Interrupt generation timing
Overrun error generation	After a stop bit is received when FIFO is full.
Break error interrupt	After a stop bit is received.
Parity error generation	After a parity data is received.
Framing error generation	After bit data that generates frame over is received.
Reception timeout interrupt	After data is received in receive FIFO, then 511 clocks of Baud16 has elapsed.
Transmission interrupt	After MSB of last data is transmitted.
Reception interrupt	After a stop bit is received.

Note: In this table, a stop bit means a last stop bit. (A stop bit length is selectable with UARTxLCR_H<STP2>)

19.4.7 DMA Interface

The UART supports DMA.

19.4.7.1 DMA Interface Signal

The DMA interface has the following signals:

a. UARTRXDMASREQ

Single DMA transfer request signal that becomes valid by the UART. When receive FIFO contains at least one word data, this signal becomes valid.

b. UARTRXDMABREQ

Burst DMA transfer request signal that becomes valid by the UART. When receive FIFO contains more data than the watermark level programmed with UARTxIFLS<RXIFSEL[2:0]>, this signal becomes valid.

c. UARTRXDMACLR

DMA request signal is validated by a DMA controller to clears the receive request signals. If DMA burst transfer is requested, the signal becomes valid during the transfer of the last data in the burst.

d. UARTTXDMASREQ

Single DMA transfer request signal, validated by the UART. If transmit FIFO contains at least 1 word empty location, this signal becomes valid.

e. UARTRXDMABREQ

Burst DMA transfer request signal, validated by the UART. When transmit FIFO has UARTxIFLS<TXIFSEL[2:0]> and data in FIFO does not reach the programmed watermark level, this signal is valid.

f. UARTRXDMACLR

DMA request clear signal, validated by a DMA controller to clear the transmit request signals. If DMA burst transfer is requested, the signal becomes valid during the transfer of the last data in the burst.

The burst DMA transfer and single DMA request signals can be both valid at the same time. For example, when there is more data than the watermark level in the receive FIFO. When the amount of data in receive FIFO is less than the watermark level, only single DMA request is valid.

For example, if 19 character have to be received and the watermark level is set to 4 level, the DMA controller transfers 4 bursts of 4 characters and 3 single transfers to complete the transfer.

19.4.8 IrDA circuit description

The IrDA is comprised of : the serial data flow control by using $\overline{\text{UTxRTS}}$ and $\overline{\text{UTxCTS}}$ pins.

- IrDA SIR transmit encoder
- IrDA SIR receive decoder

Note: The transmit encoder output (IROUT) has the opposite polarity to the receive decoder input (IRIN). Please refer to Figure 19-4.

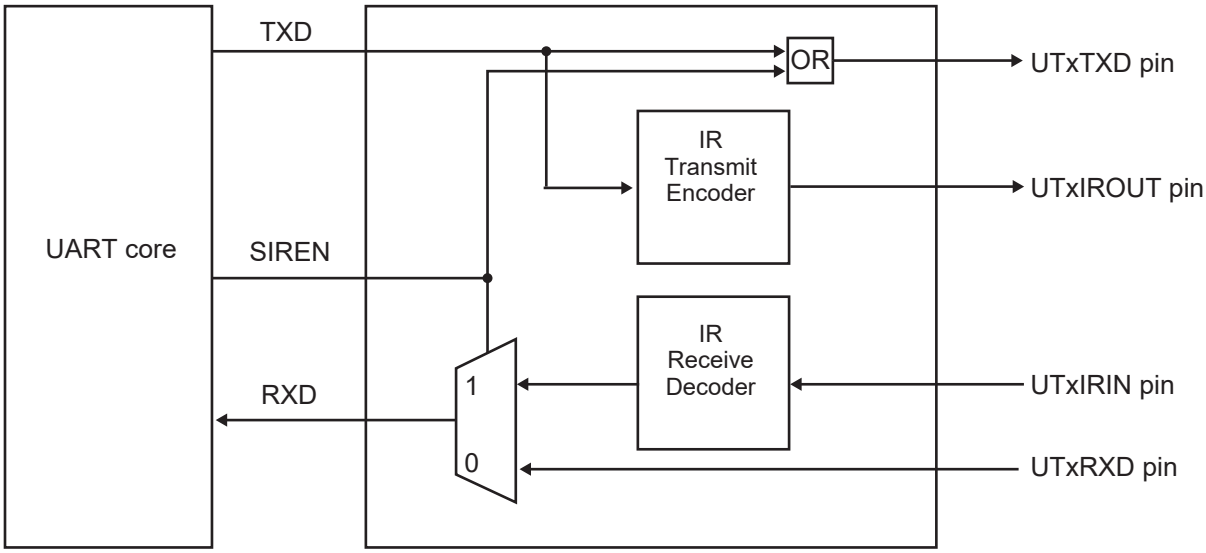


Figure 19-3 IrDA Circuit Block Diagram

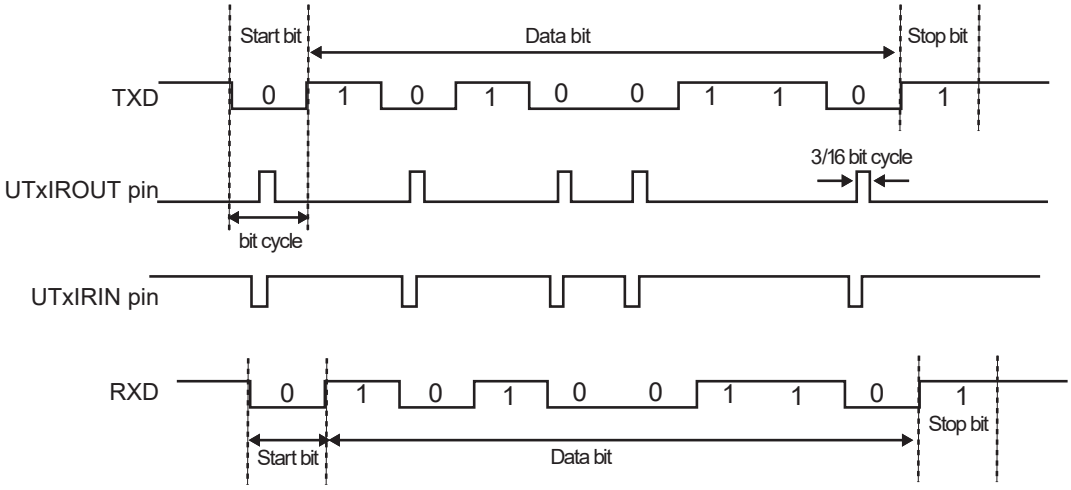


Figure 19-4 IrDA Data Modulation

19.4.9 Hardware Flow Control

The hardware flow control provides the serial data flow control by using $\overline{\text{UTxRTS}}$ and $\overline{\text{UTxCTS}}$ pins.

Figure 19-5 shows the connection between 2 devices.

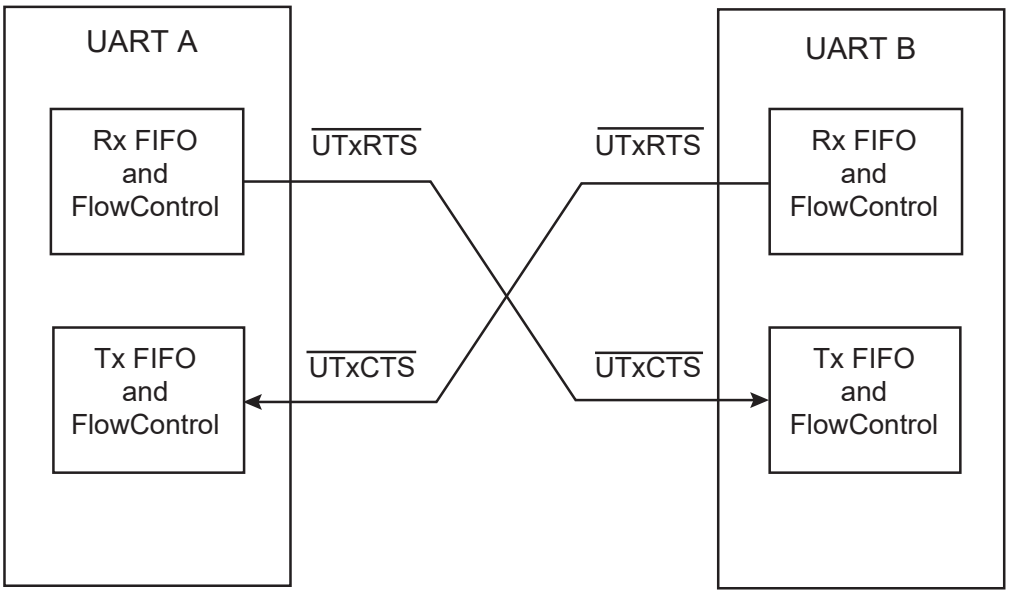


Figure 19-5 Hardware flow control

1. RTS hardware flow control

   The operation of RTS hardware flow logic control is associated with the watermark level of receive FIFO programmed with UARTxIFLS<RXIFSEL[2:0]>.

   When RTS hardware flow control is enabled, $\overline{\text{UTxRTS}}$ pin is valid if receive FIFO does not reach the watermark level. If receive FIFO is over the watermark level, $\overline{\text{UTxRTS}}$ pin becomes invalid to indicate no more room to receive any more data in receive FIFO.

   Once data is read out from receive FIFO and becomes less than the watermark level, $\overline{\text{UTxRTS}}$ pin becomes valid again.

   Communication is possible when RTS hardware flow control is disabled.

2. CTS hardware flow control

   If CTS hardware flow control is enabled, $\overline{\text{UTxCTS}}$ pin is checked before transmitting. If $\overline{\text{UTxCTS}}$ pin is valid, data is transmitted otherwise transmission does not occur.

   The data continues to be transmitted while $\overline{\text{UTxCTS}}$ pin is valid and transmit FIFO is not empty. If transmit FIFO is empty, data is not transmitted even $\overline{\text{UTxCTS}}$ pin is valid.

   If $\overline{\text{UTxCTS}}$ pin becomes invalid when CTS hardware flow control is enabled, the data continues to be transmitted until current data transmission is complete then stops.

   Communication is possible when CTS hardware flow control is disabled.

Table 19-2 Enabling/disabling hardware flow control

UARTxCR		Description
<CTSEN>	<RTSEN>	
1	1	Hardware flow control of RTS and CTS are enabled.
1	0	Only CTS hardware flow control is enabled.
0	1	Only RTS hardware flow control is enabled.
0	0	Both hardware flow control of RTS and CTS are disabled.

20. I2C Bus Interface

The TMPM46BF10FG contains I2C Bus Interface with typical I2C bus standard (Philips specifications).

The main features are as follows.

- Allows selection between master and slave.
- Allows selection between transmission and reception.
- Support multiple masters (arbitration, clock synchronization recognition).
- Baud rate (Support standard mode and fast mode)
- Supports the addressing format of 7 bit only.
- Supports transfer data sizes of 1 to 8 bits.
- Provides one transfer (transmission or reception) complete interrupt (level sensitive).
- Enable or disable the interrupts.

This module also supports Toshiba's proprietary data format called " Free data format".

Table 20-1 I2C Bus Standard specifications

I2C bus feature	I2C Standard	TMPM46BF10FG
STANDARD mode (up to 100KHz)	Required	Supported
FAST mode (up to 400KHz)	Required	Supported
Hs (High speed) mode (up to 3.4Mbps)	Required	Not Supported
7-bit addressing	Required	Supported
10-bit addressing	Required	Not Supported
START byte	Required	Supported
Noise canceller	Required	Supported (digital)
Slope control	Required	Not Supported
I/O at power off	Required	Not Supported
Schmidt (VIL/VHL)	VDD × 0.3 / VDD × 0.7	Supported
Output current at ¨VOL = 0.4V, VDD > 2V	3mA	Not Supported

20.1 Configuration

The configuration is shown in Figure 20-1.

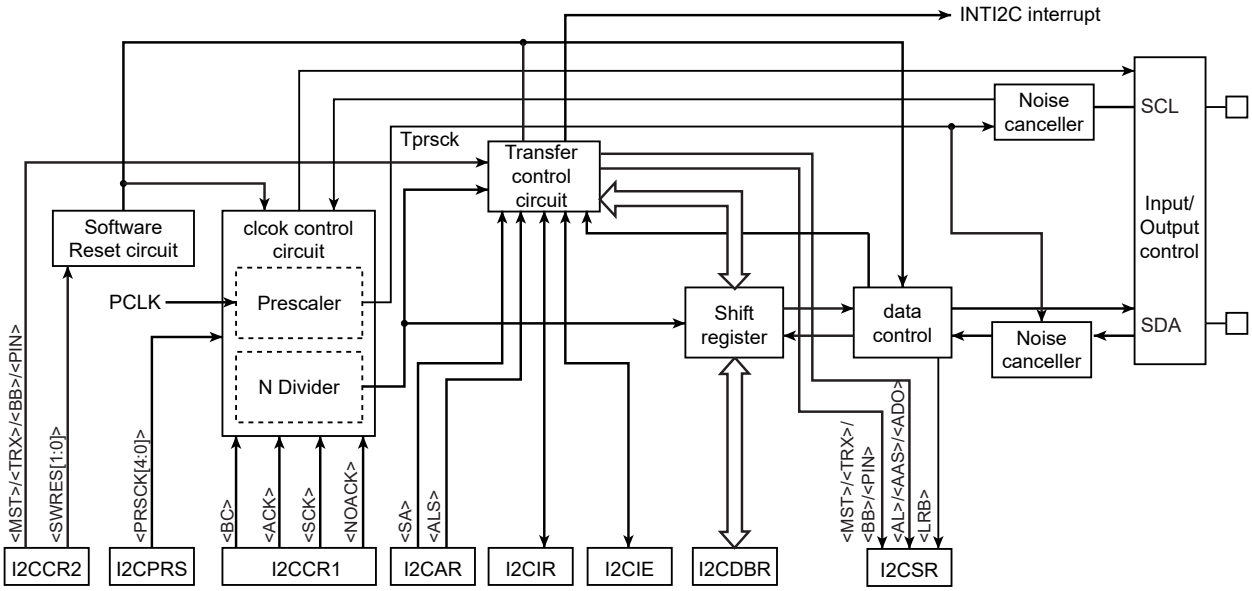


Figure 20-1 I2C Bus Block

20.2 I2C Bus mode

The I2C bus is connected to devices via the SDA and SCL pins and can communicate with multiple devices.

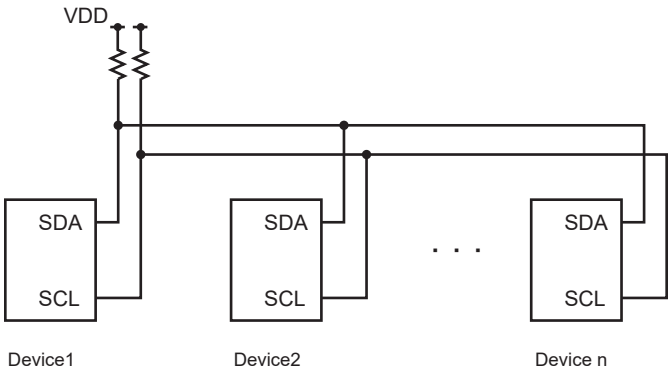


Figure 20-2

This module operates as a mater or slave device on the I2C bus.The master device drives the serial clock line (SCL) of the bus, send 8-bit address, and sends or receives data of 1 to 8bits.

The slave device send a 8-bit addresses and sends or receives serial data of 1 to 8 bits in synchronization with serial clock on the bus.

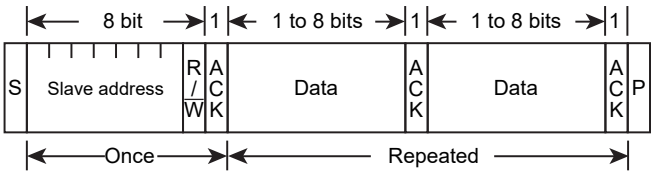
The device that operates as a receiver can output an acknowledge signal after reception of serial data and the device the operates as a transmitter can receive that acknowledge signal, regardless of whether the device is a master or slave. The master device can output a clock for the acknowledge signal.

In multimaster mode in which multiple masters exist on the same bus, serial clock synchronization and arbitration lost to maintain consistency of serial data are supported.

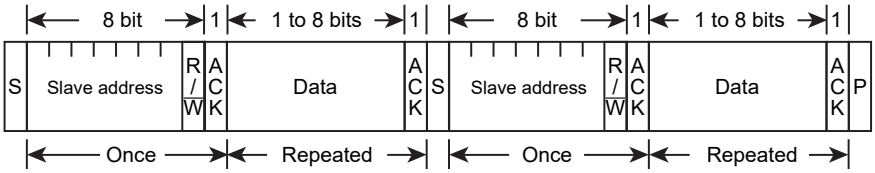
20.2.1 I2C Bus Mode Data Format

Figure 20-3 shows the data formats used in the I2C bus mode.

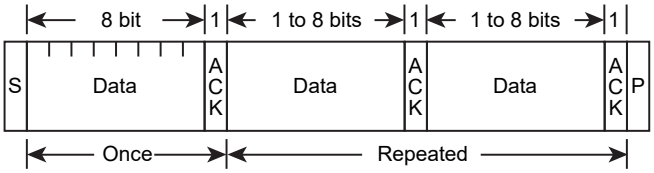
(a) Addressing format



(b) Addressing format (with repeated start condition)



(c) Free data format (master-transmitter to slave-receiver)



Note) $\underline{S}$: Start condition
R/$\overline{W}$: Direction bit
ACK : Acknowledge bit
P : Stop condition

Figure 20-3 I2C Bus Mode Data Formats

20.3 Register

20.3.1 Registers for each channel

The following registers table and address of the I2C bus interface.

For the base address, refer to the "Address lists of peripheral functions" of Chapter "Memory Map".

Register name		Address(Base+)
Control register 1	I2CxCR1	0x0000
Data buffer register	I2CxDBR	0x0004
I2C bus address register	I2CxAR	0x0008
Control register 2	I2CxCR2 (writing)	0x000C
Status register	I2CxSR (reading)	
Prescaler Clock setting register	I2CxPRS	0x0010
Interrupt Enable Register	I2CxIE	0x0014
interrupt Register	I2CxIR	0x0018

Note: These registers can be read or written by an only word access.

20.3.2 I2CxCR1(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	NOACK	SCK		
After reset	0	0	0	0	0	0	0	0

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)

<BC>	When <ACK> = 0		When <ACK> = 1	
	Number of clock cycles	Data length	Number of clock cycles	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

Bit	Bit Symbol	Type	Function
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	NOACK	R/W	Slave address match detection and general call detection. 0:the slave address match or general call detection are enabled. 1:the slave address match or general call detection are disabled. When I2CxAR<ALS>="1", this bit has no meaning.
2-0	SCK[2:0]	R/W	Select internal SCL output clock frequency (Note 2). 000 n = 0 000 n = 4 001 n = 1 001 n = 5 010 n = 2 010 n = 6 011 n = 3 011 n = 7

Note 1: Writing to this register must be done before a start condition is generated or after a stop condition is generated or between the instant when an address or data transfer interrupt occurs and the instant when the internal iterrupt is released. Do not write to this register during address or data transfer.

Note 2: For details on the SCL line clock frequency, refer to "20.4.1 Serial Clock".

Note 3: The initial value for selecting a frequency is <SCK[2:0]>=000 and is independent of the read initial value.

20.3.3 I2CxDBR (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)	Receive data
		W (Transmit)	Transmit data The transmission data must be written in to the register from the MSB (bit 7). The received data is stored in the LSB.

When the master needs to transmit a slave address, the transfer target address is written to I2CxDBR<DB[7:1]> and the transfer direction is specified in I2CxDBR<DB[0]> as follows:

<DB[0]>=0:Master(transmission) →Slave/reception

<DB[0]>=1:Master(reception) ←Slave/transmission

When all the bits in the I2CxDBR register are written as "0",a general call can be sent out on the bus.

In both transmission and reception modes, a write to the I2CxDBR register or read from the I2CxDBR register release the internal interrupt after the current transfer and initiates the next transfer.

I2CxDBR is provided as a transmit/receive buffer, it should be used as a dedicated transmit buffer in transmit mode and as a dedicated receive buffer in receive mode. This register in transmit mode and as a dedicated receive buffer in receive mode. this register should be accessed on a transfer -by - transfer basis.

Note: This register are initialized by Hardware RESET only. Software RESET can not initialize and are keeping the last data.

20.3.4 I2CxAR (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 I2C 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 I2CxAR 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 I2CxAR to "0x00" in slave mode. (If I2CxAR is set to "0x00", it's recognized that the slave address matches the START byte ("0x01") of the I2C standard received in slave mode.)

20.3.5 I2CxCR2(Control register 2)

This register serves as I2CxSR 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	I2CM	-	SWRES	
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 INTI2CIx interrupt request 0: – 1: Clear interrupt request
3	I2CM	W	I2C operation control 0: Disable 1: Enable
2	-	R	Read as 0.
1-0	SWRES[1:0]	W	Software reset generation Write "10" followed by "01" to generate a reset. For detail, refer to "20.4.11 Software Reset".

Note 1: 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 mode.

Note 2: The I2CxCR2<I2CM> bit cannot be cleared to 0 to disable I2C operation while transfer operation is being performed. before clearing this bit, mask sure that transfer operation is completely stopped by reading the status register.

20.3.6 I2CxSR (Status Register)

This register serves as I2CxCR2 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	AD0	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	INTI2Cx 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	AD0	R	General call detection 0: − 1:Detected
0	LRB	R	Last received bit monitor 0:Last received bit "0" 1:Last received bit "1"

20.3.7 I2CxPRS(Prescaler 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	-	-	-	PRSCK				
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	-	R	Read as 0.
4-0	PRSCK	R/W	Prescaler clock frequency for generating the Serial clock 00000: P = divided by 32 00001: P = divided by 1 - - - - - - - - - - 11111: P = divided by 31

Note:Refer to Section "20.3.2 I2CxCR1(Control register 1)", "20.4.1 Serial Clock".

20.3.8 I2CxIE(Interrupt 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	-	0	0	0	0	0	0	IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	-	R	Read as 0.
0	IE	R/W	I2C interrupt enable or disable setting. 0: Disable 1: Enable

20.3.9 I2CxIR(Interruppt 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	-	-	-	-	-	-	-	ISIC
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	-	R	Read as 0.
0	ISIC	R	I2C interrupt status. 0: not interrupt 1: interrupt generated. This bit indicates the I2C interrupt status prior to masking by I2CxIE<IE>
		W	Clear the I2C interrupt. 0: Invalid 1: Clear the I2C interrupt. Writing "1" to this bit clears the I2C interrupt output(INTI2Cx). Writing "0" is invalid.

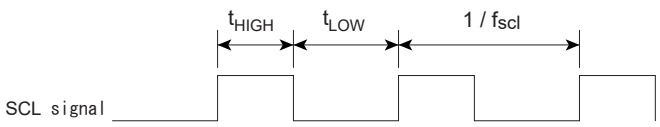
20.4 Control in the I2C Bus Mode

20.4.1 Serial Clock

20.4.1.1 Clock source

I2CxCR1<SCK[2:0]> is used to set the high and low periods of the serial clock to be output in master mode.

<SCK[2:0]>	t_{HIGH} (i / T_{prsck})	t_{LOW} (j / T_{prsck})
	i	j
000	8	12
001	10	14
010	14	18
011	22	26
100	38	42
101	70	74
110	134	138
111	262	266



t_{LOW} = i / Tprsck
t_{HIGH} = j / Tprsck
fscl = 1 / (t_{LOW} + t_{HIGH})

Figure 20-4 Clock source

Note: The tHIGH period may differ from the specified value if the rising edge becomes blunt depending on the combination of bus load capacitance and pull-up register. If the clock synchronization function for synchronizing clocks from multiple clocks is used, the actual clock period may differ from the specified setting.

In master mode, the hold time when a start conditions generated and the setup time when a stop condition is generated are defined as t_{HIGH}[S].

When I2CxCR2<PIN> is set to "1" in slave mode, the time to the release of SCLx is defined as tLOW [S].

In both master and slave modes, the high level period must be 4/Tprsck[s] or longer and the low level period must be 5/Tprsck[s] or longer for externally input serial clocks, regardless of the I2CxCR1<SCK>. setting.

The serial clock rete to be output from the master is set through I2CxCR1<SCK[2:0]> and the I2CxPRS<PRSCK[4:0]>. The prescaler clock which is divided according to I2CxPRS<PRSK[4:0]> is used as the reference clock for generating the serial clock. The prescaler clock is further divided according to I2CxCR1<SCk[2:0]> and used as the serial clock. The default setting of the prescaler clock is divide by 1(=fsys).

<Serial transfer rate>

The serial clock rate (fSCL) is determined by prescaler setting value "p" (I2CxPRS<PRSCK[4:0]>,p=1 to 32) and serial clock setting value "n" (I2CxCR1<SCK[2:0]>,n=0 to 7) based on the operating frequency (fsys) as follows:

$$\text{Serial clock rate : } f_{scl}\ (\text{kHz}) = \frac{\text{fsys(MHz)}}{p \times (2^{n+2}+16)} \times 1000$$

p:	prescaler setting	I2CxPRS<PRSCK[4:0]>,	1 to 32
n:	serial clock setting	I2CxCR1<SCK[2:0]>,	0 to 7

The allowed range of prescaler setting value "p" (I2CxPRS<PRSCK[4:0]>) varies depending on the operating frequency (fsys) and must satisfy the following condition.

50ns	<	Prescaler clock width:Tprsck (ns)	≤	150ns

Note: Setting the prescaler clock width out of this range is prohibited in both master and slave modes.

The serial clock rate may not be constant due to the clock synchronization function.

n: <SCK[2:0]>			p: <PRSCK[4:0]>		
			00001 (divide by1)	01101 (divide by 13)	00000 (divide by 32)
			Ratio to fsys		
0	0	0	20	260	640
0	0	1	24	312	768
0	1	0	32	416	1024
0	1	1	48	624	1536
1	0	0	80	1040	2560
1	0	1	144	1872	4608
1	1	0	272	3536	8704
1	1	1	528	6864	16896

Note: Writing to these bits must be done before a start condition is generated or after a stop condition is benerated.Writing to these bits during transfer will cause unexpected operation.

<Prescaler clock width (=noise cancellation width)

The prescaler clock width (Tprsck)(= noise cancellation width) is determined by prescaler setting value "p" (I2CxPRS<PRSCK[4:0]>,P=1 to 32) based on the operating frequency (fsys) as follows:

$$\text{Prescaler clock width : Tprsck (ns)} = \frac{1}{\text{fsys(MHz)}} \times 1000 \times p$$

(=Noise cancellation width)

20.4.1.2 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.

I2C has a clock synchronization function to ensure proper transfer operation even when multiple master exist on a bus.

For example, the clock synchronization procedure for a bus with two masters is shown below.

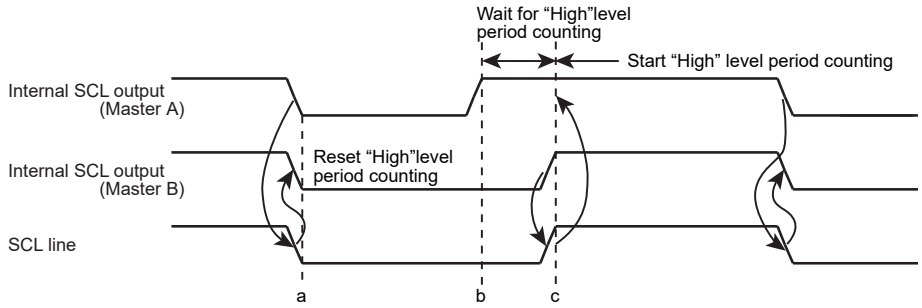


Figure 20-5 Example of Clock Synchronization

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.

20.4.2 Selection for a Slave Address Match Detection or General Call Detection

For a slave device, the I2CxCR1<NOACK> is done to enable or disable for a slave address match detection and general call detection in slave mode.

when I2CxCR1<NOACK>=0,it is enable for a slave address match detection and general call detection.

If the received address matches its slave address specified at I2CxAR or is equal to the general-call address, the I2C pulls the SDA line to the "Low" level during the ninth clock and outputs an acknowledgment signal.

when I2CxCR1<NOACK>=1,it is disable for a slave address match detection and general call detection.

If the received address matches its slave address specified at I2CxAR or is equal to the general-call address, the I2C release the SDA line to the "High" level and outputs an non-acknowledgment signal.

The slave device ignores a slave address and general calls sent from the master and returns non-acknowledgment. the INTI2Cx interrupt request are not generated.

In master mode, the bit of I2CxCR1<NOACK> is ignored and has no effect on operation.

Note: if I2CxCR1<NOACK> is cleared to "0" during data transfer in slave mode, I2CxCR1<NOACK> remains "1" and an acknowledge signal is returned for the transferred data.

20.4.3 Setting the Acknowledgement Mode

Setting I2CxCR1<ACK> to "1" selects the acknowledge mode. When operating as a master, the I2C adds one clock for acknowledgment signal. In slave mode, the clock for acknowledgement signals is counted. In transmitter mode, the I2C releases the SDAx pin during clock cycle to receive acknowledgement signals from the receiver. In receiver mode, the I2C 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 I2C pulls the SDAx pin to the "Low" level during the clock cycle and generates acknowledgement signals.

However, the second byte is necessary to be controlled by software to generate an ACK signal on the contents of the second byte.

By setting <ACK> to "0", the non-acknowledgment mode is activated. When operating as a master, the I2C does not generate clock for acknowledgement signals. In slave mode, the clock for acknowledgement signals is not counted.

20.4.4 Setting the Number of Bits per Transfer

I2CxCR1<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.

Note: A slave address must be trasmitted / received with I2CxCR1<ACK> set to "1". if I2CxCR1<ACK> is cleared, the slave address match detection and direction bit detection cannot be performed properly.

20.4.5 Slave Addressing and Address Recognition Mode

Setting "0" to I2CxAR<ALS> and a slave address in I2CxAR<SA[6:0]> sets addressing format, and then the I2C recognizes a slave address transmitted by the master device and receives data in the addressing format.

If <ALS> is set to "1", the I2C 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.

20.4.6 Configuring the I2C as a Master or a Slave

Setting I2CxCR2<MST> to "1" configures the I2C to operate as a master device.

Setting <MST> to "0" configures the I2C 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.

20.4.7 Configuring the I2C as a Transmitter or a Receiver

Setting I2CxCR2<TRX> to "1" configures the I2C as a transmitter. Setting <TRX> to "0" configures the I2C 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 I2CxCAR.
- when a general-call address is received (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 I2C 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 I2C 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.

Note: When I2CxCR1<NOACK>=1, the slave address detection and general call detection are disabled, and thus I2CxSR<TRX> remains unchanged.

Table 20-2 I2CxSR<TRX> operation

mode	direction bit	condition for state change	Changed <TRX> after state change
Slave mode	0	Received Slave address = I2CxAR<SA>	0
	1		1
Master mode	0	ACK received	1
	1		0

When I2C is used in free data format, the salve address and direction bit are not recognized and bits immediately following a start condition are handled as data. Therefore, <TRX> is not changed by the hardware.

20.4.8 Generating Start and Stop Conditions

When I2CxSR<BB> is "0", writing "1" to I2CxCR2<MST, TRX, BB, PIN> causes the I2C 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.

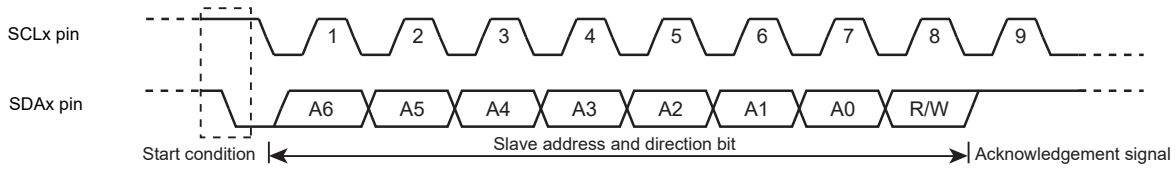


Figure 20-6 Generating the Start Condition and a Slave Address

When <BB> is "1", writing "1" to <MST, TRX, PIN> and "0" to <BB> causes the I2C 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.

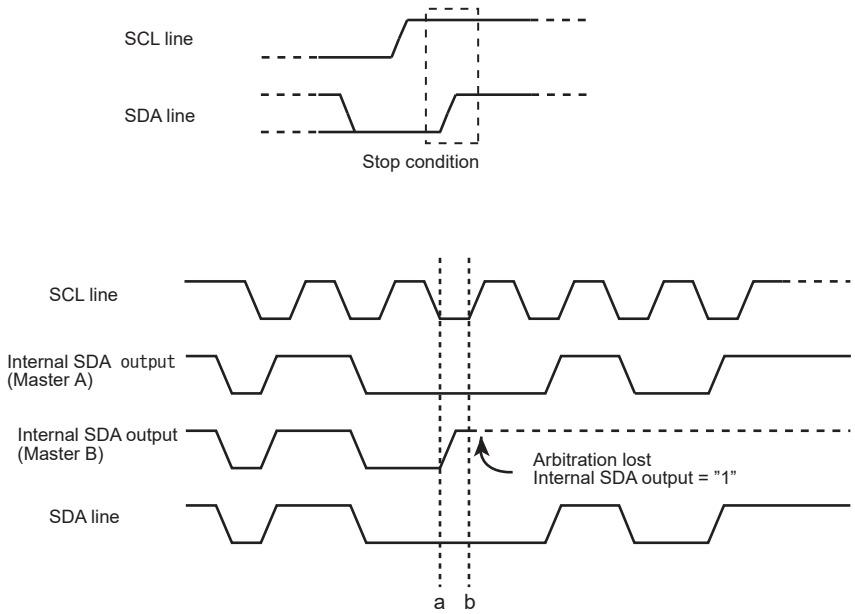


Figure 20-7 Generating the Stop Condition

I2CxSR<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).

The following table shows typical setting examples according to the I2CxSR state.

Although the I2CxCR2<MST>,<TRX>,<BB>,<PIN> bits are given independent functions, they are used in typical combinations, as shown below, according to the I2CxSR setting.

I2CxSR			I2CxCR2				Operation
<MST>	<BB>	<PIN>	<MST>	<TRX>	<BB>	<PIN>	
0	0	1	0	0	0	0	Wait for a start condition as a slave.
			1	1	1	1	Generate a start condition.
1	1	0	1	1	0	1	Generate a stop condition.
			0	0	0	1	Release the internal interrupt for restart.

Note: When writing to these bits, do not change I2CxCR2<I2CM> by mistake.

20.4.9 Interrupt Service Request and Release

In master mode, a I2C bus interface request (INTI2Cx) is generated when the transfer of the number of clock cycles set by I2CxCR1<BC> and I2CxCR1<ACK> is completed.

In slave mode, INTI2Cx is generated under the following conditions.

- When I2CxCR1<NOACK> is "0",after output of the acknowledge signal which is generated when the received slave address matches the slave address set to I2CxAR<SA[6:0]>.
- When I2CxCR1<NOACK> is "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.

When an interrupt request (INTI2Cx) is generated, I2CxCR2<PIN> is cleared to "0". While <PIN> is cleared to "0", the I2C pulls the SCL line to the "Low" level.

<PIN> is set to "1" when data is written to or read from I2CxDBR. 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 lost occurs while a slave address and direction bit are transferred in the master mode, <PIN> is cleared to "0" and INTI2Cx occurs. This does not relate to whether a slave address matches <SA>.

20.4.10 I2C Bus mode

When I2CxCR2<I2CM> is set to "1". I2C bus mode is selected. ensure that the I2C bus interface pins are at "High" level before setting <I2CM> to "1". Also, ensure that the bus is free before switching the operating mode to the port mode.

Note: When I2CxCR2<I2CM> = "0", no value can be written to bits in the I2CxCR2 register other than I2CxCR2<I2CM> bit. Before setting I2CxCR2, write "1" into I2CxCR2<I2CM> to select the I2C bus mode.

20.4.11 Software Reset

If the I2C bus interface circuit locks up due to external noise, it can be initialized by using a software reset.

If the I2C bus interface circuit locks up due to external noise, it can be initialized by using a software reset.Writing "10" followed by "01" into I2CxCR2<SWRES[1:0]> generates a reset signal that initializes the I2C bus interface circuit. After a reset, all control registers and status flags except I2CxCR2<I2CM> and I2CxDBR register are initialized to their reset values. When the I2C bus interface is initialized, <SWRES[1:0]> is automatically cleared to "0".

Note: A software reset causes the I2C operating mode to switch from the I2C mode to the port mode.

20.4.12 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. When a start condition is output under the bus busy state, a start condition is not output to SCL or SDA line, thus arbitration lost occurs.

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.

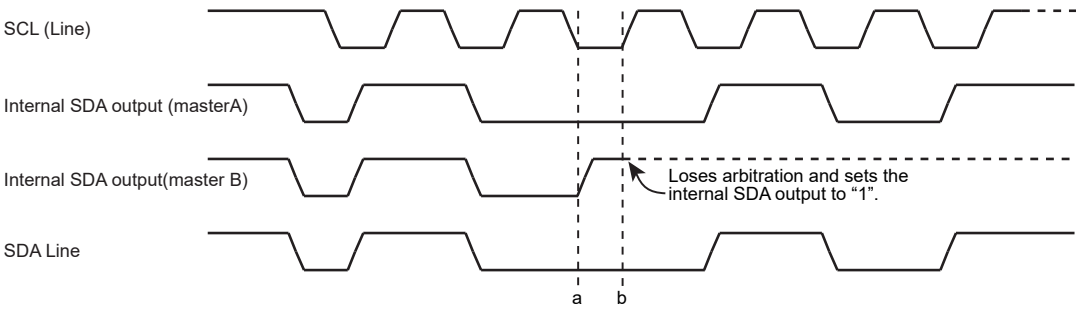


Figure 20-8 Lost Arbitration

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 I2CxSR<AL> is set to "1".

When <AL> is set to "1", I2CxSR<MST, TRX> are cleared to "0", causing the I2C 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 I2CxDBR or data is written to I2CxCR2.

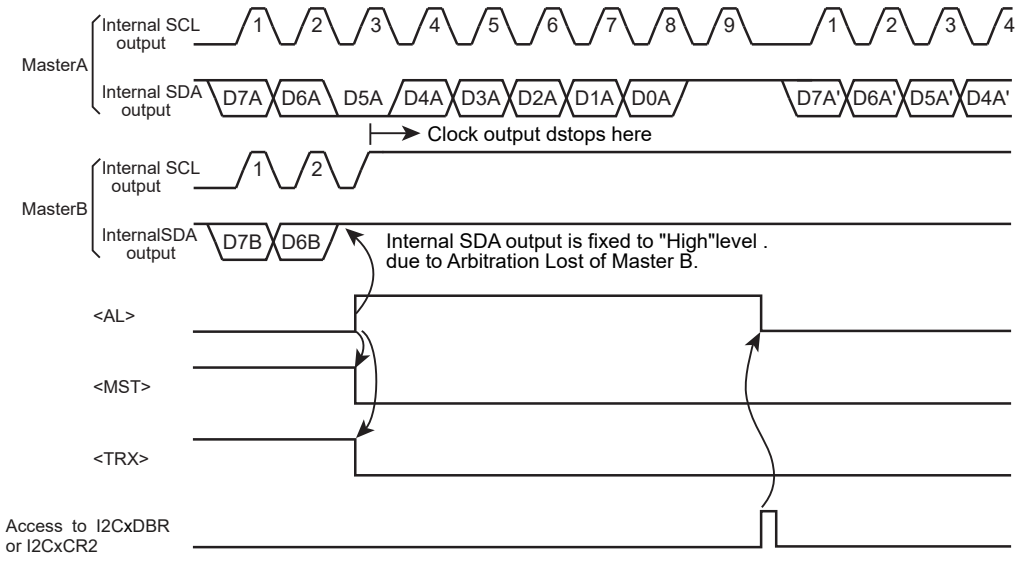


Figure 20-9 Example of Master B Lost Arbitration (D7A = D7B, D6A = D6B)

20.4.13 Slave Address Match Detection Monitor

When the I2C operates as a slave device in the address recognition mode (I2CxAR<ALS>="0"), I2CxSR<AAS> is set to "1" on receiving the general-call address or the slave address that matches the value specified at I2CxAR.

Clearing I2CxCR1<NOACK> to "0" enable the slave address match detection.When a general call is received or the slave address sent form the master matches the slave address set in I2CxAR<SA>,I2CxSR<AAS> is set to "1".

Setting I2CxCR1<NOACK> to "1" disable the slave address match detection. Even if a general call is received or the slave address sent form the master matches the slave address set in I2CxAR<SA>,I2CxSR<AAS> remains "0".

When Free data format mode <ALS> is set to "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 I2CxDBR.

20.4.14 General-call Detection Monitor

When the I2C operates as a slave device, I2CxSR<AD0> is set to "1" when it receives the general-call address (i.e.; the eight bits following the start condition are all zeros).

<AD0> is cleared to "0" when the start or stop condition is detected on the bus.

20.4.15 Last Received Bit Monitor

I2CxSR<LRB> is set to the SDA line value that was read at the rising of the SCL line.

In the acknowledgment mode, reading I2CxSR<LRB> immediately after generation of the INTI2Cx interrupt request causes ACK signal to be read.

20.4.16 Data Buffer Register (I2CxDBR)

Reading or writing I2CxDBR initiates reading received data or writing transmitted data.

When the I2C is acting as a master, setting a slave address and a direction bit to this register generates the start condition.

20.5 Data Transfer Procedure in the I2C Bus Mode

20.5.1 Device Initialization

After ensuring that the SDA and SCL pins are high (Bus free), set I2CxCR2<I2CM> to "1" for enable the I2C.

Next, write "1" to I2CxCR1<ACK]> and "0" to I2CxCR1<NOACK>.Writing "000" to I2CxCR1<BC[2:0]> at the time.

These setting enable acknowledge operation, slave address match detection and general call detection and set the data length to 8 bits.

and set "tHIGH" and "tLOW" in I2CxCR1<SCK>.

then, program I2CxAR 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 I2C Bus Interface as a slave receiver, ensure that the I2C bus interface pin is at "High" first. Finally, write "0" to I2CxCR2<MST, TRX, BB>, "1" to I2CxCR2<PIN>, "00" to I2CxCR2<SWRES[1:0]> to configure the device as a slave receiver.

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	
I2CxCR2	←	0	0	0	1	1	0	0	0	enable I2C
I2CxCR1	←	0	0	0	1	0	1	1	0	Specifies ACK and SCL clock.
I2CxAR	←	X	X	X	X	X	X	X	X	Specifies a slave address and an address recognition mode.
I2CxCR2	←	0	0	0	1	1	0	0	0	Configures the I2C as a slave receiver.

Note: X; Don’t care

20.5.2 Generating the Start Condition and a Slave Address

20.5.2.1 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 I2CxCR1<ACK> to select the acknowledgment mode. Write to I2CxDBR a slave address and a direction bit to be transmitted.

When <BB> = "0", writing "1111" to I2CxCR2<MST, TRX, BB, PIN> generates the start condition on the bus. Following the start condition, the I2C generates nine clocks from the SCL pin. The I2C outputs the slave address and the direction bit specified at I2CxDBR with the first eight clocks, and releases the SDA line in the ninth clock to receive an acknowledgment signal from the slave device.

The INTI2Cx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to "0". In the master mode, the I2C 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 INTI2Cx interrupt request, provided that an acknowledgment signal has been returned from the slave device.

Note 1: To output slave address, check with software that the bus is free before writing to I2CxDBR. If this rule is not followed, data being output on the bus may get ruined.

Note 2: When other master device may transfer while the device generate a start condition after writing the slave address.Therefor,To generate a start condition again after confirmed the bus is free by your program. the above procedure is done within 98.0μs (Minimum transmitting time on the STANDARD mode of I2C standard) or 23.7μs (Minimum transmitting time on the FAST mode of I2C standard) after writing the slave address.

Settings in main routine

		7	6	5	4	3	2	1	0	
Reg.	←	I2CxSR								
Reg.	←	Reg. AND 0x20								
if Reg.	≠	0x00								Ensures that the bus is free.
Then										
I2CxCR1	←	X	X	X	1	0	X	X	X	Selects the acknowledgement mode.
I2CxDBR	←	X	X	X	X	X	X	X	X	Specifies the desired slave address and direction.
I2CxCR2	←	1	1	1	1	1	0	0	0	Generates the start condition.

Example of INTI2C0 interrupt routine

Clears the interrupt request.

Processing

End of interrupt

20.5.2.2 Slave mode

In the slave mode, the I2C receives the start condition and a slave address.

After receiving the start condition from the master device, the I2C 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 I2CxAR or is equal to the general-call address, the I2C pulls the SDA line to the "Low" level during the ninth clock and outputs an acknowledgment signal.

The INTI2Cx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to "0". In the slave mode, the I2C holds the SCL line at the "Low" level while <PIN> is "0".

Note: The I2C bus with DMA transfer is possible as following condition.

- Master and slave, one to one communication
- Sequential and continuous transmission or received

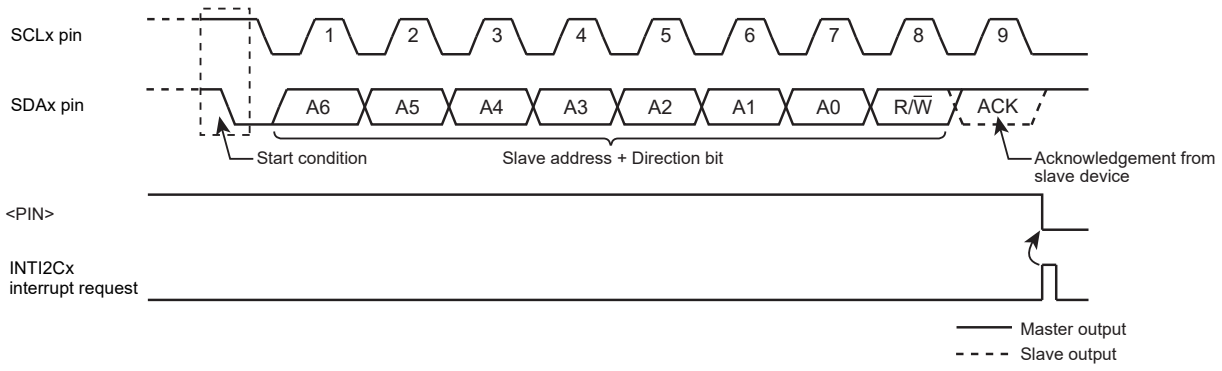


Figure 20-10 Generation of the Start Condition and a Slave Address

20.5.3 Transferring a Data Word

At the end of a data word transfer, the INTI2Cx interrupt is generated to test I2CxSR<MST> to determine whether the I2C is in the master or slave mode.

20.5.3.1 Master mode (<MST> = "1")

Test <TRX> to determine whether the I2C 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 I2CxDBR. If the data has different length, <BC[2:0]> and <ACK> are programmed and the transmit data is written into I2CxDBR. 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 INTI2Cx 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.

INTI2Cx 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.

I2CxCR1	←	X	X	X	X	0	X	X	X	Specifies the number of bits to be transmitted and specify whether ACK is required.
I2CxDBR	←	X	X	X	X	X	X	X	X	Writes the transmit data.

End of interrupt processing.

Note: X; Don't care

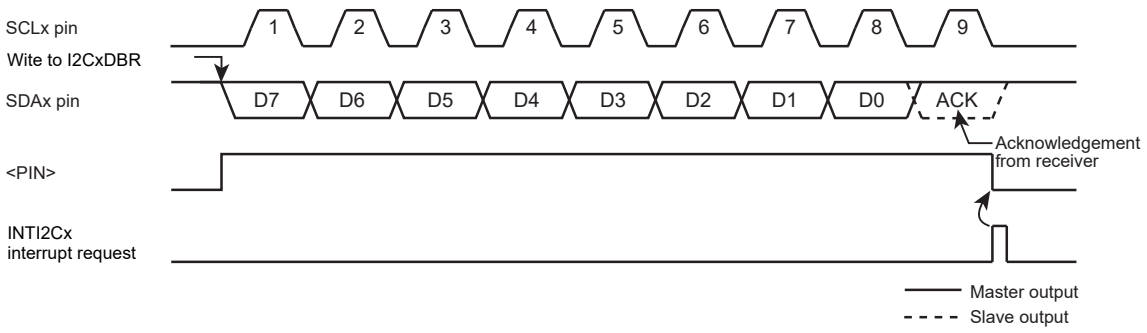


Figure 20-11 <BC[2:0]>= "000",<ACK>= "1" (Transmitter Mode)

(2) Receiver mode (<TRX> = "0")

If the next data to be transmitted has eight bits, the transmit data is written into I2CxDBR.

If the data has different length, <BC[2:0]> and <ACK> are programmed and the received data is read from I2CxDBR 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 INTI2Cx 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 I2CxDBR, one-word transfer clock and an acknowledgement signal are output.

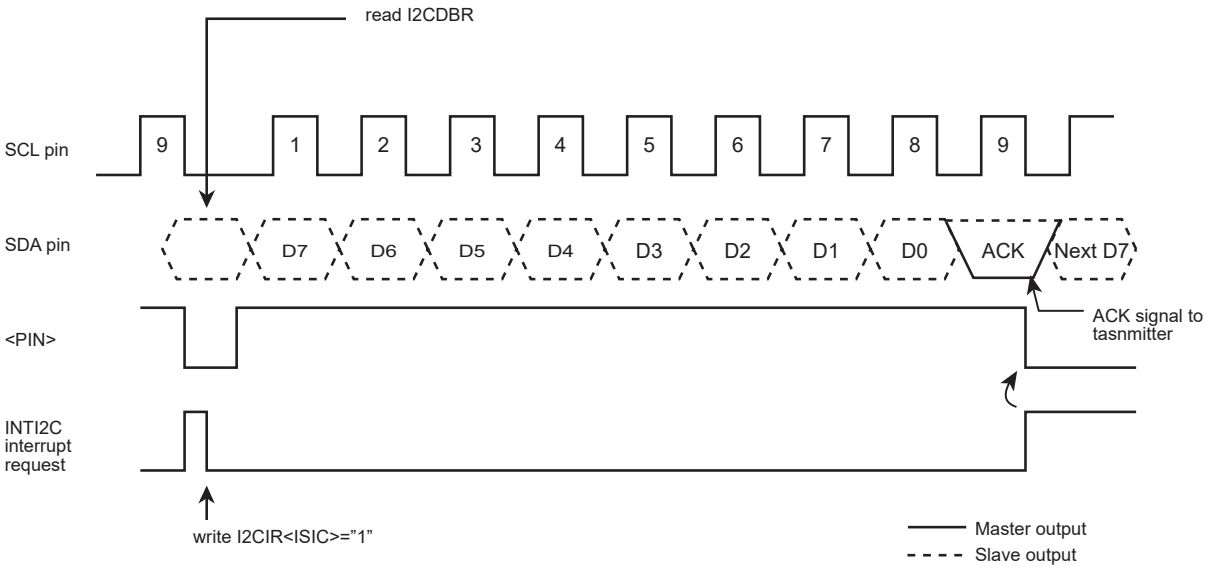


Figure 20-12 <BC[2:0]>= "000",<ACK>= "1" (Receiver Mode)

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.

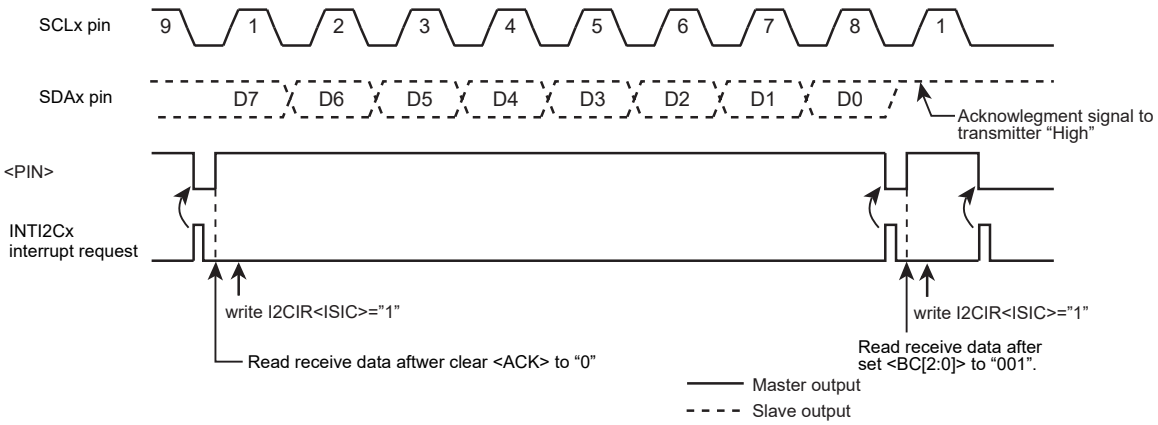


Figure 20-13 Terminating Data Transmission in the Master Receiver Mode

Example: When receiving N data word

INTI2Cx interrupt (after data transmission)

		7	6	5	4	3	2	1	0	
I2CxCR1	←	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.	←	I2CxDBR								Reads dummy data.
End of interrupt										

INTI2Cx interrupt (first to (N-2)th data reception)

		7	6	5	4	3	2	1	0	
Reg.	←	I2CxDBR								Reads the first to (N-2)th data words.
End of interrupt										

INTI2Cx interrupt ((N-1)th data reception)

		7	6	5	4	3	2	1	0	
I2CxCR1	←	X	X	X	0	0	X	X	X	Disables generation of acknowledgement clock.
Reg.	←	I2CxDBR								Reads the (N-1)th data word.
End of interrupt										

INTI2Cx interrupt (Nth data reception)

		7	6	5	4	3	2	1	0	
I2CxCR1	←	0	0	1	0	0	X	X	X	Disables generation of acknowledgement clock.
Reg.	←	I2CxDBR								Reads the Nth data word.
End of interrupt										

INTI2Cx interrupt (after completing data reception)

Processing to generate the stop condition.	Terminates the data transmission.
End of interrupt	

Note: X; Don’t care

20.5.3.2 Slave mode (<MST> = "0")

In the slave mode, the I2C generates the INTI2Cx interrupt request on following status:

1) when I2CxCR1<NOACK> is "0", the I2C has received a general-call address.

2) when I2CxCR1<NOACK> is "0", the received slave address matches I2CxAR<SA> address.

3) when a data transfer has been completed in response to a received slave address matching or a general-call address.

Also, if the I2C 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 INTI2Cx 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 I2CxDBR 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.

I2CxSR<AL>, <TRX>, <AAS> and <AD0> are tested to determine the processing required.

"Table 20-3 Processing in Slave Mode"shows the slave mode states and required processing.

Example: When the received slave address matches the I2C's own address and the direction bit is "1" in the slave receiver mode.

INTI2Cx 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.
I2CxCR1   ←  X  X  X  1  0  X  X  X   Sets the number of bits to be transmitted.
I2CxDBR   ←  X  X  X  X  X  X  X  X   Sets the transmit data.
```

Note:X; Don't care

Table 20-3 Processing in Slave Mode

<TRX>	<AL>	<AAS>	<AD0>	State	Processing
1	1	1	0	Arbitration Lost is detected while the slave address was being transmitted and the I2C 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 I2CxDBR.
	0	1	0	In the slave receiver mode, the I2C received a slave address with the direction bit "1" transmitted by the master.	
		0	0	In the slave transmitter mode, the I2C 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 I2CxDBR.
0	1	1	1/0	Arbitration Lost is detected while a slave address is being transmitted, and the I2C receives either a slave address with the direction bit "0" or a general-call address transmitted by another master.	Read the I2CxDBR (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 I2C 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 I2C 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 I2CxDBR.

Note: In salve mode, if I2CxAR<SA[6:0]> is set to "0x00" a START byte (0x01) of the I2C bus standard is received, a slave address match is detected and I2CxSR<TRX> is set to "1". Not set I2CxAR<SA[6:0]> to "0x00".

20.5.4 Generating the Stop Condition

When I2CxSR<BB> is "1", writing "1" to I2CxCR2<MST, TRX, PIN> and "0" to <BB> causes the I2C 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 I2C waits until the SCL line is released.

After that, the SDA pin goes "High", and then causing the stop condition to be generated for a "t_{HIGH}".

		7	6	5	4	3	2	1	0	
I2CxCR2	←	1	1	0	1	1	0	0	0	Generates the stop condition.

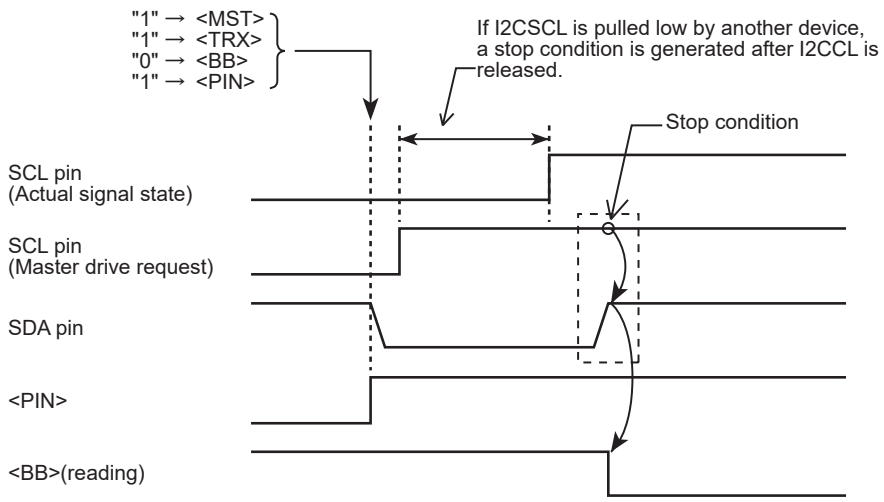


Figure 20-14 Generating the Stop Condition

20.5.5 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 I2CxCR2<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 I2CxSR<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 "20.5.2 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	
I2CxCR2	←	0	0	0	1	1	0	0	0	Releases the bus.
if I2CxSR<BB> ≠ 0										Checks that the SCL pin is released.
Then										
if I2CxSR<LRB> ≠ 1										Checks that no other device is pulling the SCL pin to the "Low".
Then										
4.7 μs Wait										
I2CxCR1	←	X	X	X	1	0	X	X	X	Selects the acknowledgment mode.
I2CxDBR	←	X	X	X	X	X	X	X	X	Sets the desired slave address and direction.
I2CxCR2	←	1	1	1	1	1	0	0	0	Generates the start condition.

Note: X; Don't care

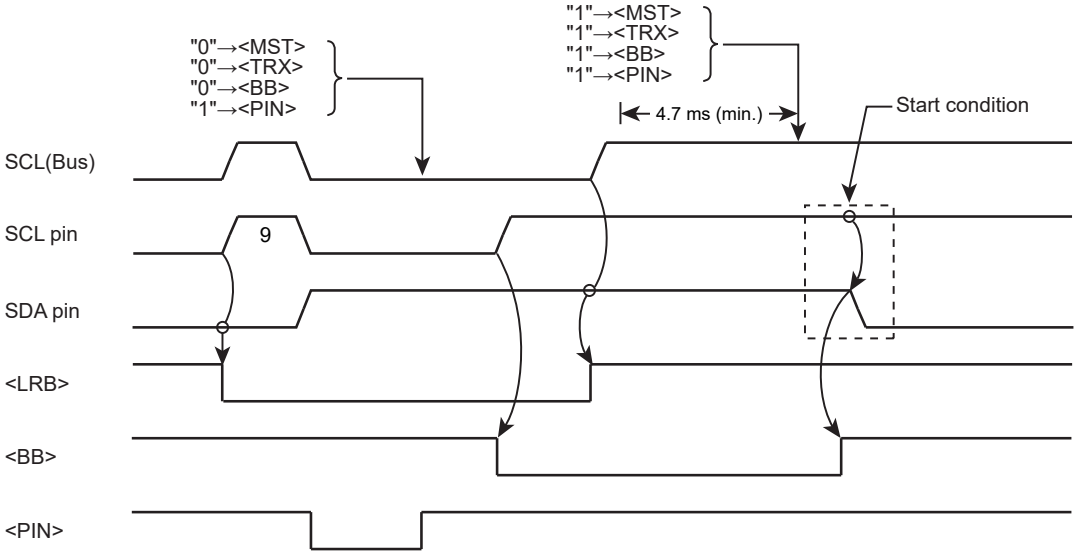


Figure 20-15 Timing Chart of Generating a Restart

20.6 Notice on usage

20.6.1 Register values after a Software Reset

A software reset initializes the I2C register other than I2CxCR2<I2CM> and internal circuit and releases the SCL and SDA lines. (refer to section "20.4.1.2 Clock Synchronization".)

However, depending on read timing after a software reset, reading I2CxSR<LRB> may return a value other than the initial value as "0".

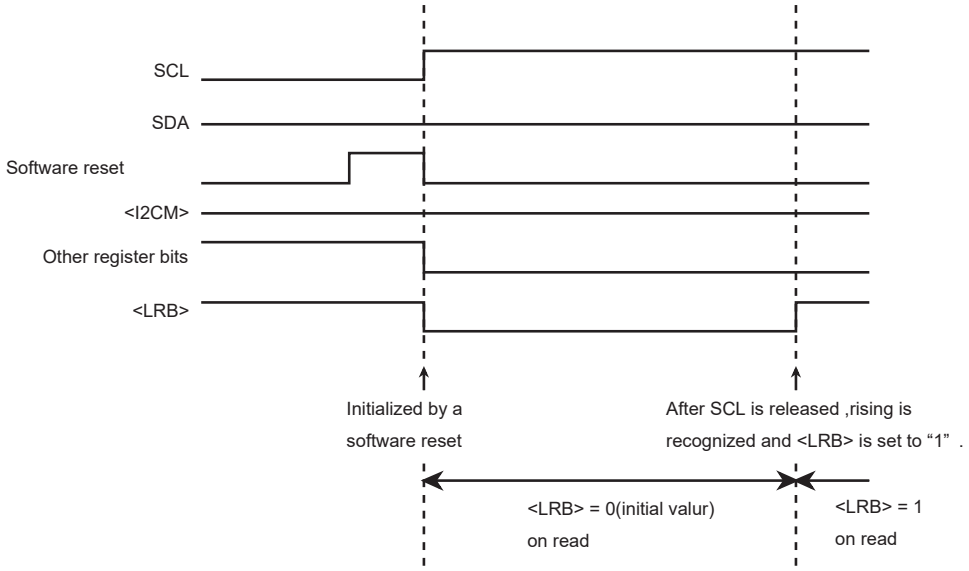


Figure 20-16 Software reset release to the SCL0 "0" to "1" while SDA="1"

21. Synchronous Serial Port (SSP)

21.1 Overview

This LSI contains the SSP (Synchronous Serial Port) with channels. These channels have 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 (note)	In master mode	fsys/2 to fsys/65024
	In slave mode	fsys/12 to fsys/65024
DMA		Supported
Internal test function		The internal loopback test mode is available.

21.2 Block Diagram

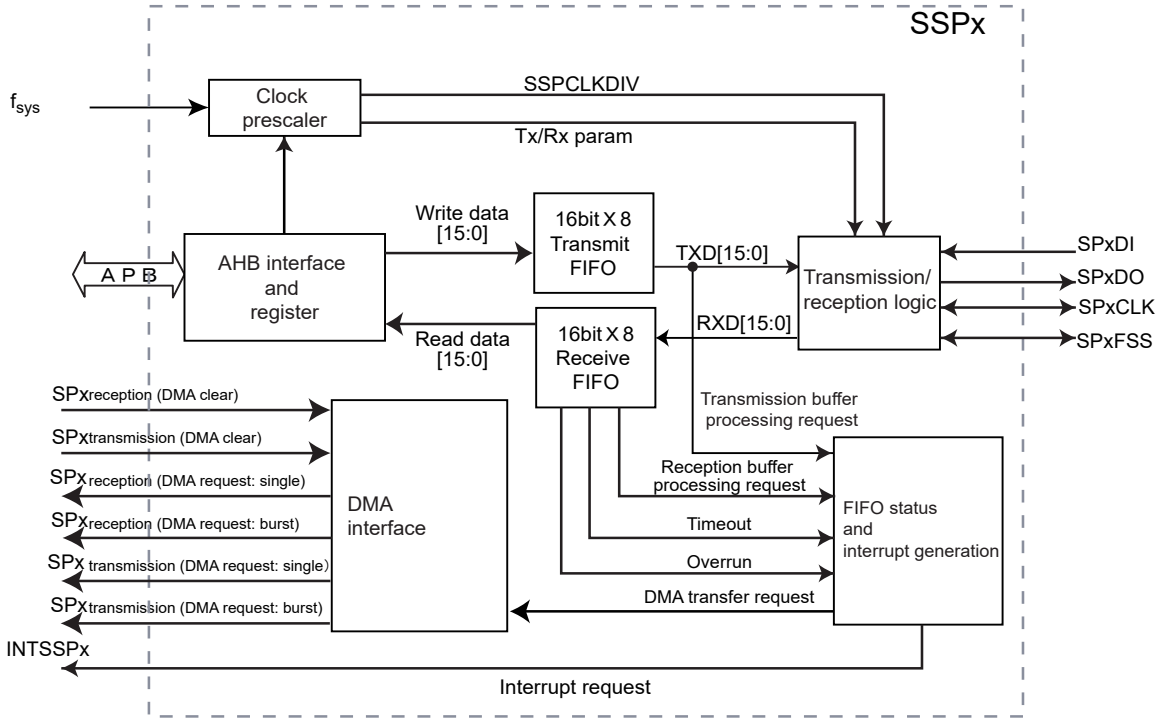


Figure 21-1 SSP block diagram

21.3 Register

21.3.1 Register List

The table shows control registers and their addresses.

For detail of the base address, refer to "Address lists of peripheral functions" of "Memory Map" chapter.

Register Name		Address (Base+)
Control register 0	SSPxCR0	0x0000
Control register 1	SSPxCR1	0x0004
Receive FIFO (read) and transmit FIFO (write) data register	SSPxDR	0x0008
Status register	SSPxSR	0x000C
Clock prescale register	SSPxCPSR	0x0010
Interrupt enable/disable register	SSPxIMSC	0x0014
Pre-enable interrupt status register	SSPxRIS	0x0018
Post-enable interrupt status register	SSPxMIS	0x001C
Interrupt clear register	SSPxICR	0x0020
DMA control register	SSPxDMACR	0x0024
Reserved	-	0x0028 to 0x0FFC

Note1: These registers in the above table allows to access only word (32 bits) basis.
Note2: Access to the "Reserved" area is prohibited.

21.3.2 SSPxCR0(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 SSPxCPSR register, and <SCR> takes a value between 0 to 255.
7	SPH	R/W	SPxCLK 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	SPxCLK polarity: 0:SPxCLK is in Low state. 1:SPxCLK 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 SSPxCR0<SCR[7:0]> = 0x00, SSPxCPSR<CPSDVSR[7:0]> = 0x02, when slave mode is selected.

21.3.3 SSPxCR1(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 SPxDO 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

21.3.4 SSPxDR(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.

21.3.5 SSPxSR(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.

21.3.6 SSPxCPSR (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 SSPxCR0<SCR[7:0]> = 0x00, SSPxCPSR<CPSDVSR[7:0]> = 0x02, when slave mode is selected.

21.3.7 SSPxIMSC (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.

21.3.8 SSPxRIS (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

21.3.9 SSPxMIS (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

21.3.10 SSPxICR (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

21.3.11 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

21.4 Overview of SSP

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 SPxDO and received via SPxDI.

The SSP contains a programmable prescaler to generate the serial output clock SPxCLK from the input clock fsys. The operation mode, frame format, and data size of the SSP are programmed in the control registers SSPxCR0 and SSPxCR1.

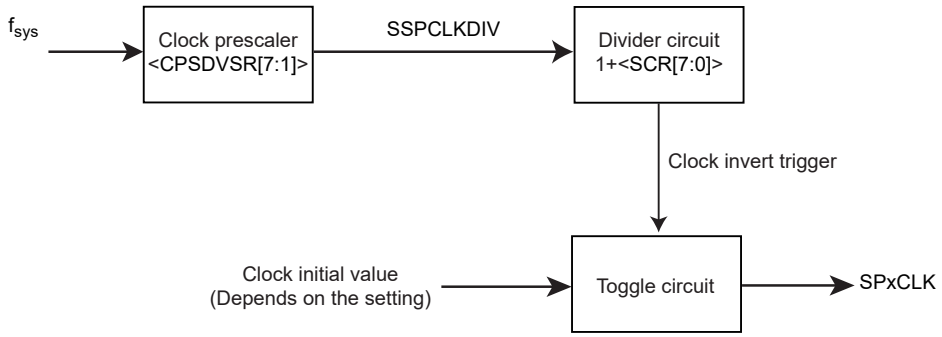
21.4.1 Clock prescaler

When configured as a master, a clock prescaler comprising two free-running serially linked counters is used to provide the serial output clock SPxCLK.

You can program the clock prescaler through the SSPxCPSR register, to divide fsys by a factor of 2 to 254 in steps of two. Because the least significant bit of the SSPxCPSR 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 SSPxCR0 register, to give the master output clock SPxCLK.

Bitrate = fsys / (<CPSDVSR>×(1+<SCR>))



21.4.2 Transmit FIFO

This is a 16-bit wide, 8-layered transmit FIFO buffer, which is shared in master and slave modes.

21.4.3 Receive FIFO

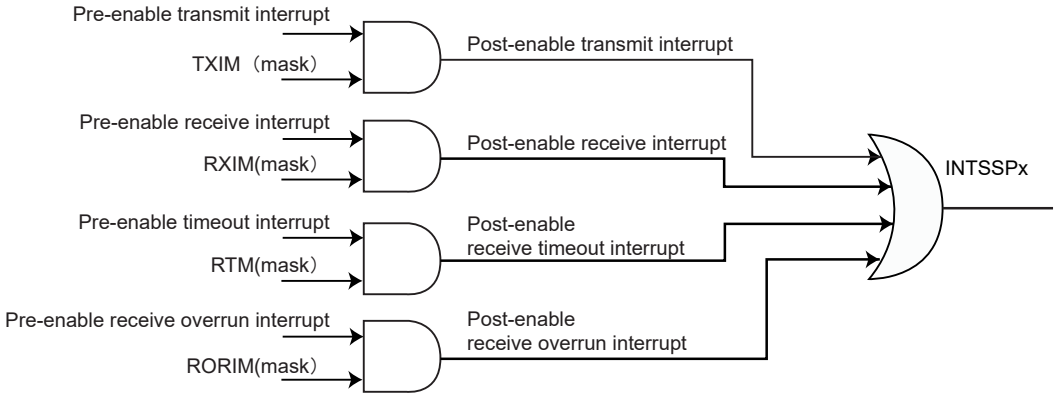
This is a 16-bit wide 8-layered receive FIFO buffer, which is shared in master and slave modes.

21.4.4 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 INTSSPx 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 (SSPxCR1 <SSE> = "0").

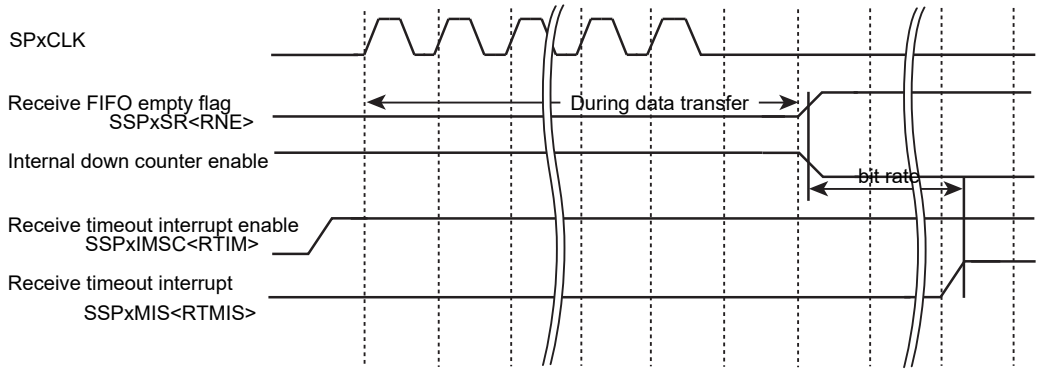
The first transmitted data can be written in the FIFO by using this interrupt.

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.



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 SSPxICR<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.

21.4.5 DMA Interface

SSP supports DMA burst and single transfers. When DMA transfer is enabled by SSPxDMACR, both burst and single transfers are enabled. All transfer requests are cleared if SSP operation or DMA transfers is disabled.

21.4.5.1 Burst Transfer

When data is increased over the watermark level (half of the FIFO) in the receive FIFO, a receive burst DMA transfer request is asserted.

When data is reduced to less than the watermark level (half of the FIFO) in the transmit FIFO, a transmit burst DMA transfer request is asserted.

Set the DMA burst length to 4 words.

Note:For the remaining three words, the SSP does not assert the burst request.

The following table shows the trigger points for DMABREQ, for both transmit and receive FIFOs.

	Burst length	
Watermark level	Transmit (Number of empty locations)	Receive (Number of filled locations)
1/2	4	4

21.4.5.2 Single Transfer

If at least one data is stored in the receive FIFO, a receive single DMA transfer request is asserted.

If at least one empty space exists in the transmit FIFO, a transmit single DMA transfer request is asserted.

Both burst and single transfers can be used simultaneously. In the case of reception, when data is increased over the watermark level, both burst and single transfer requests are asserted. When data is reduced to less than the watermark level, only single transfer request is asserted. In the case of transmission, when data is reduced to less than the watermark level, both burst and single transfer requests are asserted. When data is increased over the watermark level, only single transfer request is asserted.

For example, when 19-word is received, 4-word burst transfer is performed 4 times. After this transfer, DMA asserts a transfer completion signal and the burst transfer is finished. A single transfer request is asserted for left 3 words. Through three times single transfers all data can be transferred.

21.5 SSP operation

21.5.1 Initial setting for SSP

Settings for the SSP communication protocol must be made with the SSP disabled.

Control registers SSPxCR0 and SSPxCR1 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 SSPxCPSR and SSPxCR0 <SCR>.

This SSP supports the following protocols:

- SPI
- SSI
- Microwire

21.5.2 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 SPxFSS 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.

21.5.3 Clock ratios

When setting a frequency for fsys, the following conditions must be met.

If there are further conditions with a product, refer to the "product information " chapter.

- In master mode

  f_{SPxCLK} (maximum) → f_{sys} /2

  f_{SPxCLK} (minimum) → f_{sys} /(254×256)
- In slave mode

  f_{SPxCLK} (maximum) → f_{sys} /12

  f_{SPxCLK} (minimum) → f_{sys} /(254×256)

21.6 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 (SPxCLK)

  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 (SPxFSS)

  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 SPxCLK and the input data is received at its falling edge.

Refer to Section "21.6.1" to "21.6.3" for details of each frame format.

21.6.1 SSI frame format

In this mode, the SSP is in idle state, SPxCLK and SPxFSS are forcedly set to "Low", and the transmit data line SPxDO becomes Hi-Z. When data is written in the transmit FIFO, the master outputs "High" pulses of 1 SPxCLK to the SPxFSS 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 SPxDO pin at the next rising edge of SPxCLK.

Likewise, the received data will be input starting from the MSB to the SPxDI pin at the falling edge of SPxCLK. The received data will be transferred from the serial shift register into the receive FIFO at the rising edge of SPxCLK after its LSB data is latched.

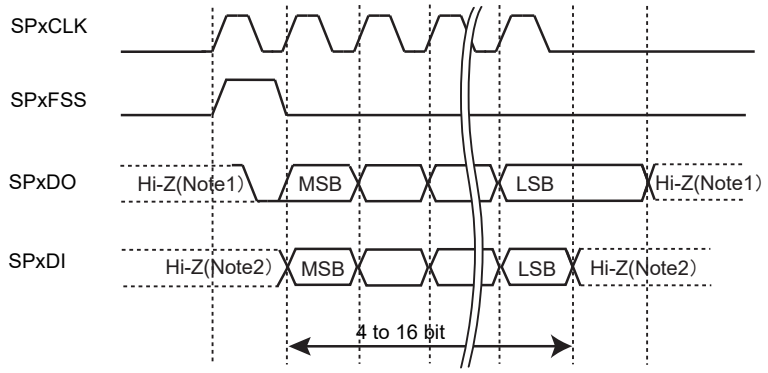


Figure 21-2 SSI frame format (transmission/reception during single transfer)

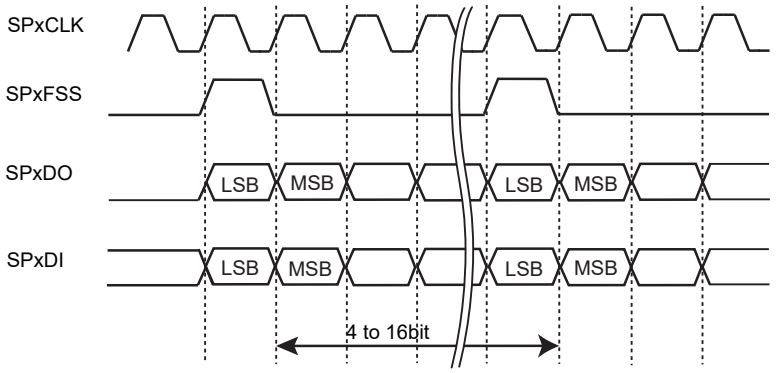


Figure 21-3 SSI frame format (transmission/reception during continuous transfer)

Note 1: When transmission is disable, SPxDO 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: SPxDI 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.

21.6.2 SPI frame format

The SPI interface has 4 lines. SPxFSS is used for slave selection. One of the main features of the SPI format is that the <SPO> and <SPH> bits in the SSPxCR0 register can be used to set the SPxCLK operation timing.

SSPxCR0 <SPO> is used to set the level at which SPxCLK in idle state is held.

SSPxCR0 <SPH> is used to select the clock edge at which data is latched.

	SSPxCR0<SPO>	SSPxCR0<SPH>
0	"Low" state	Capture data at the 1st clock edge.
1	"High" state	Capture data at the 2nd clock edge.

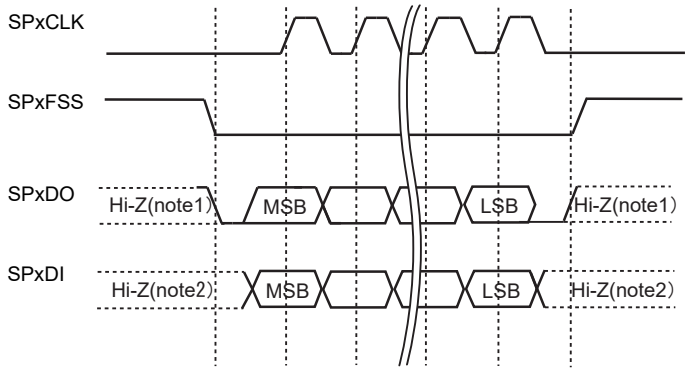


Figure 21-4 SPI frame format (single transfer, <SPO>="0" & <SPH>="0")

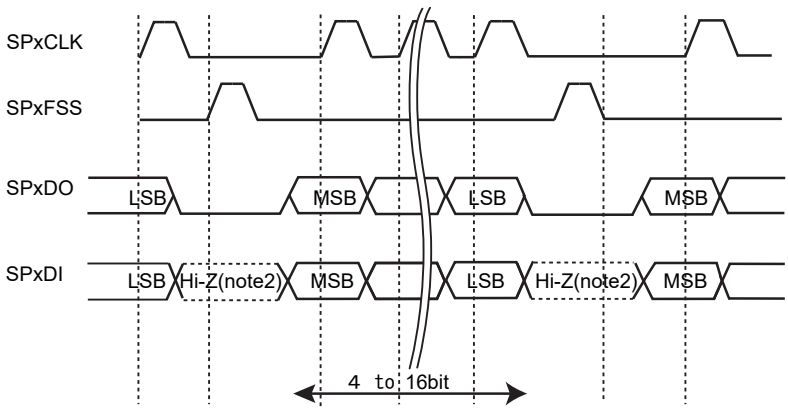


Figure 21-5 SPI frame format (continuous transfer,<SPO>="0" & <SPH>="0")

Note 1: When transmission is disable, SPxDO 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: SPxDI 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 SPxCLK signal is set to "Low".
- SPxFSS is set to "High".
- The transmit data line SPxDO is set to "Low".

If the SSP is enabled and valid data exists in the transmit FIFO, the SPxFSS master signal driven by "Low" notifies of the start of transmission. This enables the slave data in the SPxDI input line of the master.

When a half of the SPxCLK period has passed, valid master data is transferred to the SPxDO pin. Both the master data and slave data are now set. When another half of SPxCLK has passed, the SPxCLK master clock pin becomes "High". After that, the data is captured at the rising edge of the SPxCLK signal and transmitted at its falling edge.

In the single transfer, the SPxFSS line will return to the idle "High" state when all the bits of that data word have been transferred, and then one cycle of SPxCLK has passed after the last bit was captured.

However, for continuous transfer, the SPxFSS 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 SPxFSS pin of the slave device between individual data transfers. When the continuous transfer is completed, the SPxFSS pin will return to the idle state when one cycle of SPxCLK has passed after the last bit is captured.

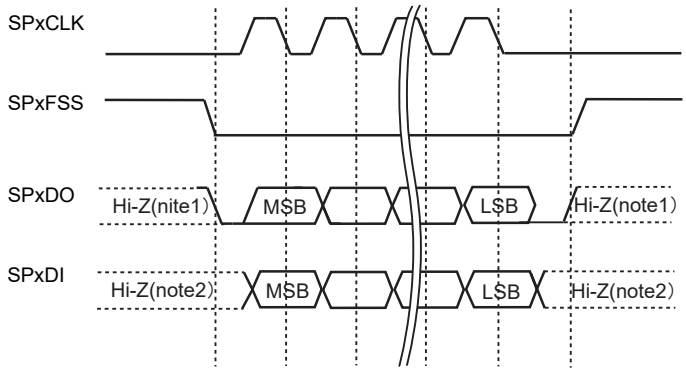


Figure 21-6 SPI frame format (Single & continuous transfer,<SPO>="0" & <SPH>="1")

Figure 21-6 show the SPI frame format with <SPO>=0 & <SPH>=1,which is covers both single and continuous transfer.

Note 1: When transmission is disable, SPxDO 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: SPxDI 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.

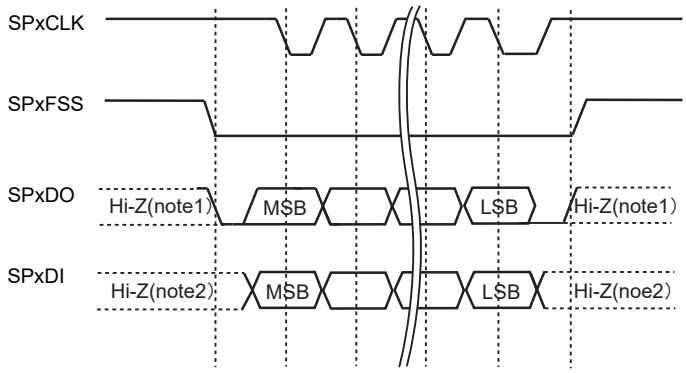


Figure 21-7 SPI frame format (single transfer,<SPO>="1" & <SPH>="0")

Figure 21-7 shows the SPI frame format with <SPO>=1 &<SPH>=0, which is for single transmission signal sequence.

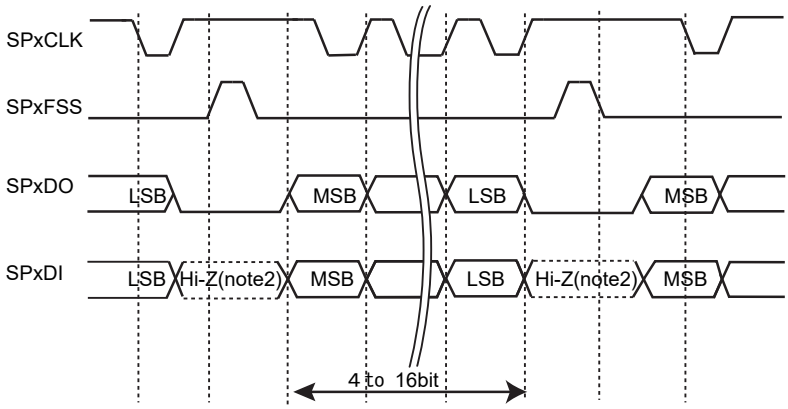


Figure 21-8 SPI frame format (continuous transfer,<SPO>="1" & <SPH>="0")

Figure 21-8 shows the SPI frame format with <SPO>=1 &<SPH>=0, which is for continuous transmission signal sequence.

Note 1: When transmission is disable, SPxDO 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: SPxDI 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.

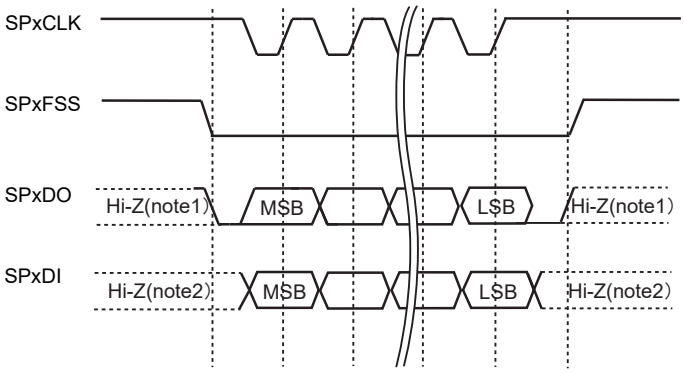


Figure 21-9 SPI frame format (Single & continuous transfer,<SPO>="1" & <SPH>="1")

Figure 21-9 show the SPI frame format with <SPO>=1 & <SPH>=1,which is covers both single and continuous transfer.

Note 1: When transmission is disable, SPxDO 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: SPxDI 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.

21.6.3 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.

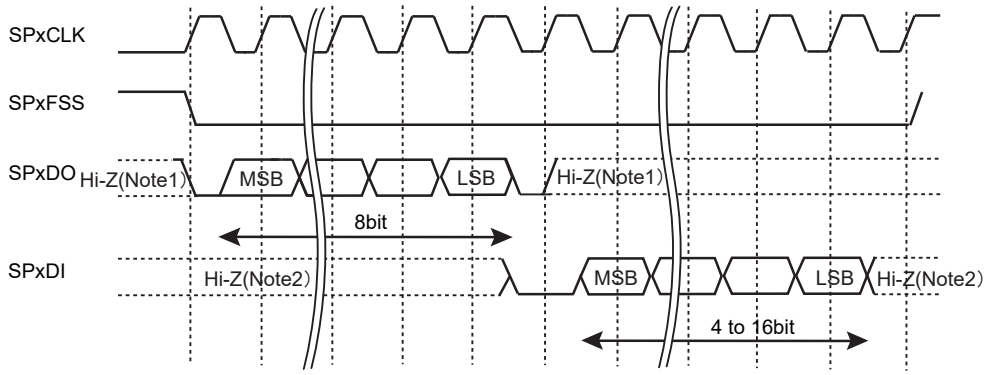


Figure 21-10 Microwire frame format (single transfer)

Note 1: When transmission is disabled, SPxDO 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: SPxDI 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 SPxCLK signal is set to "Low".
- SPxFSS is set to "High".
- The transmit data line SPxDO is set to "Low".

A transmission is triggered by writing a control byte to the transmit FIFO. The falling edge of SPxFSS 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 SPxDO pin.

SPxFSS remains "Low" and the SPxDI pin remains tristate during this transmission. The off-chip serial slave device latches each control bit into its serial shifter on the rising edge of each SPxCLK.

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 SPxDI line on the falling edge of SPxCLK.

The SSP in turn latches each bit on the rising edge of SPxCLK. At the end of the frame, for single transfers, the SPxFSS 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 SPxCLK after the LSB has been latched by the receive shifter, or when the SPxFSS pin goes "High".

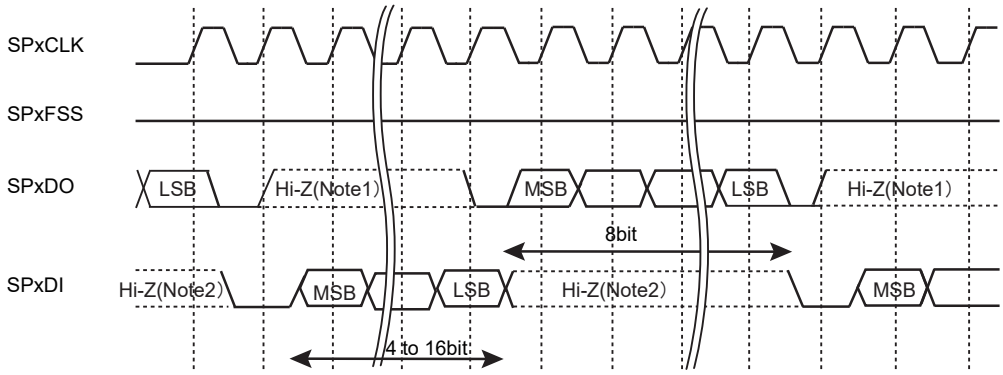


Figure 21-11 Microwire frame format (continuous transfer)

Note 1: When transmission is disabled, SPxDO 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: SPxDI 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.

For continuous transfers, data transmission begins and ends in the same manner as a single transfer. However, the SPxFSS 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 SPxCLK, 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.

22. Analog / Digital Converter (ADC)

22.1 Features

TMPM46BF10FG contain one unit of 12-bit sequential-conversion analog/digital converters (ADC) with 8 analog input channels.

These 8 analog input channels (AIN0 to AIN7) are also used as input/output ports.

A 12-bit A/D converter has the features shown below.

- Starting normal AD conversion and top-priority AD conversion

  Software activation

  Start-up by an external trigger ($\overline{\text{ADTRG}}$)

  Start-up by internal triggers
- Operation modes of Normal AD conversion

  Fixed-channel single conversion mode

  Channel scan single conversion mode

  Fixed-channel repeat conversion mode

  Channel scan repeat conversion mode
- Operation modes of top-priority AD conversion

  Fixed-channel single conversion mode
- Normal AD conversion end interrupt and top-priority AD conversion end interrupt
- Normal AD conversion function and top-priority AD conversion contain the below status flags.

  A flag that indicates AD conversion results are valid and a flag that indicates overwrite.

  AD conversion completion flag and AD conversion busy flag.
- AD monitor function

  If an arbitrary condition for comparison is satisfied, an interrupt occurs.
- AD conversion clock is controllable from fc to fc/16.
- Current reduction Function of VREF Reference is supported.

22.2 Configuration

Figure 22-1 shown the block diagram of the AD converter.

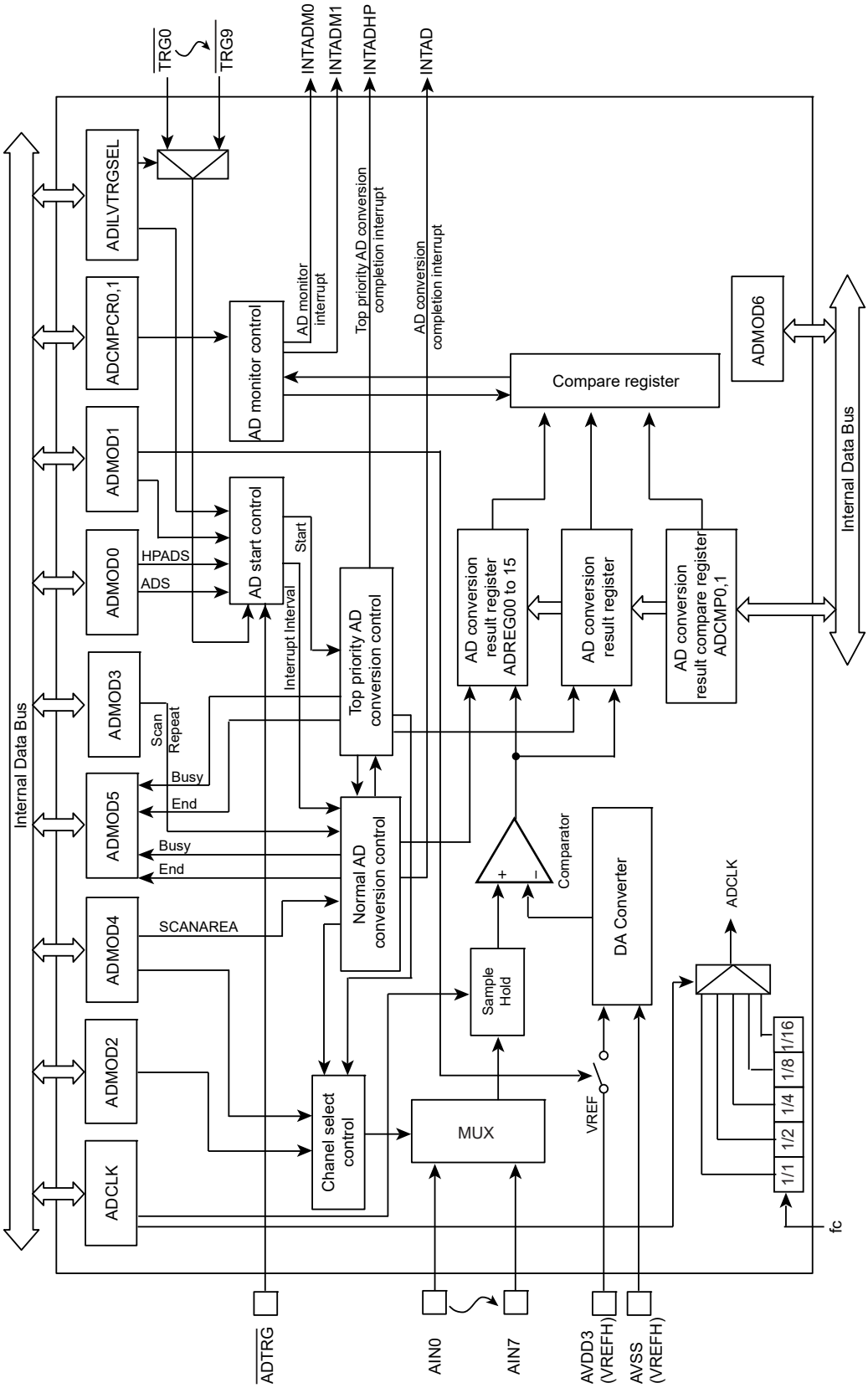


Figure 22-1 Block Diagram

22.3 Registers

22.3.1 Register list

The AD converter is controlled by the Mode Setting Registers (ADMOD0 through ADMOD6).

For the base address, refer to "Address lists of peripheral functions" of Chapter "Memory Map".

Registers		Address (Base+)
Clock Setting Register	ADCLK	0x0000
Mode Setting Register 0	ADMOD0	0x0004
Mode Setting Register 1	ADMOD1	0x0008
Mode Setting Register 2	ADMOD2	0x000C
Mode Setting Register 3	ADMOD3	0x0010
Mode Setting Register 4	ADMOD4	0x0014
Mode Setting Register 5	ADMOD5	0x0018
Mode Setting Register 6	ADMOD6	0x001C
Monitoring Setting Register 0	ADCMPCR0	0x0024
Monitoring Setting Register 1	ADCMPCR1	0x0028
AD Conversion Result Compare Register 0	ADCMP0	0x002C
AD Conversion Result Compare Register 1	ADCMP1	0x0030
AD Conversion Result Register 0	ADREG00	0x0034
AD Conversion Result Register 1	ADREG01	0x0038
AD Conversion Result Register 2	ADREG02	0x003C
AD Conversion Result Register 3	ADREG03	0x0040
AD Conversion Result Register 4	ADREG04	0x0044
AD Conversion Result Register 5	ADREG05	0x0048
AD Conversion Result Register 6	ADREG06	0x004C
AD Conversion Result Register 7	ADREG07	0x0050
Top-priority Conversion Result Register	ADREGSP	0x0074

Registers		Address (Base+)
Trigger Selection Register	ADILVTRGSEL	0x0010

Note:Access the registers by using word (32 bit) reads and word writes.

22.3.2 ADCLK (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	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	ADSH				-	ADCLK		
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as zero.
7-4	ADSH[3:0]	R/W	Select the ADC sample hold time. 0000: 10 × <ADCLK> 0001: 20 × <ADCLK> 0010: 30 × <ADCLK> 0011: 40 × <ADCLK> 0100: 80 × <ADCLK> 0101: 160 × <ADCLK> 0110: 320 × <ADCLK> 0111 to 1111: Reserved
3	–	R	Read as zero.
2-0	ADCLK[2:0]	R/W	Select the ADC prescaler output. 000: fc 001: fc/2 010: fc/4 011: fc/8 100: fc/16 101 to 111: Reserved

Note 1: Use in the range of 4MHz ≤ ADCLK ≤ 40MHz. For example, if the settings are fosc = 12MHz and PLL = multiply-by-4, the value is fc = 48MHz. In this case, do not use the condition ADCLK<ADCLK[2:0]> = "000".

Note 2: To select the ADC prescaler output <ADCLK >, stop AD conversion and the condition must be as follows: ADMOD1<DACON>="0".

Note 3: ADC sample hold time is used the following condition :Normal type channels(250 ns over)

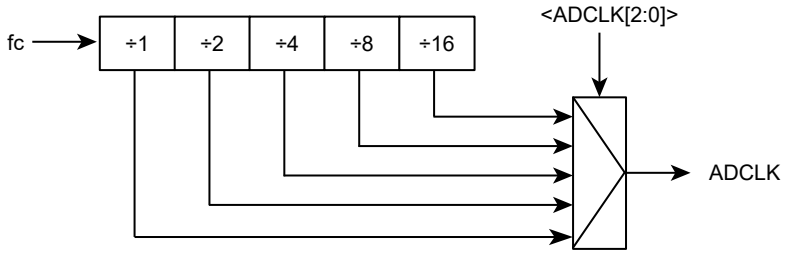


Figure 22-2 AD Conversion Clock (ADCLK)

The total ADC conversion time is caluculated as follows:

t_{cov}=ADCLK×(ADSH(Sampling clock) + 30clocks)

Example: fc=40MHz ,ADCLK=40MHz(=fc) ,ADSH=10 × <ADCLK>

t_{cov}=ADCLK× (10 + 30) = 25(ns)×(10 + 30)=1000(ns)=1.0 μs

22.3.3 ADMOD0 (Mode Setting 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	-	-	-	-	-	-	HPADS	ADS
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	–	R	Read as zero.
1	HPADS	W	Starts a top-priority AD conversion 0: Don't care 1: Start conversion "0" is always read.
0	ADS	W	Starts a normal AD conversion 0: Don't care 1: Start conversion "0" is always read.

Note: If the top-priority AD conversion <HPADS> and the normal AD conversion <ADS> start at the same time, the top-priority AD conversion receives preference to start. The normal AD conversion does not start.

22.3.4 ADMOD1 (Mode Setting 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	DACON	I2AD	RCUT	-	HPADHWS	HPADHWE	ADHWS	ADHWE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as zero.
7	DACON	R/W	Circuit ON/OFF control 0: OFF 1: ON
6	I2AD	R/W	ADC operation control in IDLE mode. 0: Stop 1: Operate
5	RCUT	R/W	Controls the reference current between VREFH and VREFL. 0: Energizing only during conversion. 1: Always energized excepting the reset time.
4	–	R	Read as zero.
3	HPADHWS	R/W	Selects a hardware activation source of top-priority AD conversion 0: $\overline{\text{ADTRG}}$ pin 1: Internal trigger (selected by ADILVTRGSEL<HPTRGSEL>)
2	HPADHWE	R/W	Controls top-priority AD conversion triggered by hardware factors. 0: Disable 1: Enable
1	ADHWS	R/W	Selects a hardware activation source of normal AD conversion 0: $\overline{\text{ADTRG}}$ pin 1: Internal trigger (selected by ADILVTRGSEL<TRGSEL>)
0	ADHWE	R/W	Activate normal AD conversion triggered by hardware factors. 0: Disable 1: Enable

Note 1: To reduce consumption current used when shifting to IDLE mode with setting <I2AD>="0", or in STOP1/STOP2 mode, set "0" to <DACON> and <RCUT> after AD conversion, and then execute an instruction to move on to stand-by mode.

Note 2: When using external trigger is used as a hardware activating source for the top-priority AD conversion, external trigger cannot be selected for the normal AD conversion hardware start.

22.3.5 ADMOD2 (Mode Setting 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	HPADCH				ADCH			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as zero.
7-4	HPADCH[3:0]	R/W	Select an analog input channel for top-priority AD conversion.(refer toTable 22-1).
3-0	ADCH[3:0]	R/W	Select an analog input channel for normal AD conversion (refer to Table 22-1).

Table 22-1 Select input channels for normal AD conversion and top-priority AD conversion

<HPADCH[3:0]>	Analog input channel for top-priority AD conversion	<ADCH[3:0]>	Analog input channel for normal AD conversion
0000	AIN0	0000	AIN0
0001	AIN1	0001	AIN1
0010	AIN2	0010	AIN2
0011	AIN3	0011	AIN3
0100	AIN4	0100	AIN4
0101	AIN5	0101	AIN5
0110	AIN6	0110	AIN6
0111	AIN7	0111	AIN7

22.3.6 ADMOD3 (Mode Setting 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	-	ITM			-	-	REPEAT	SCAN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	–	R	Read as zero.
6-4	ITM[2:0]	R/W	Set interrupt generation timing in channel fixed repeated conversion mode.(refer to Table 22-2)
3-2	–	R	Read as zero.
1	REPEAT	R/W	Sets repeat mode. 0 : Single conversion 1 : Repeat conversion
0	SCAN	R/W	Sets scan mode. 0 : Fixed channel mode 1 : Channel scan mode

Table 22-2 Interrupt generation timing in fixed channel mode

<ITM[2:0]>	Fixed channel repeat conversion mode <SCAN> = "0", <REPEAT> = "1"
000	Each time one conversion is completed.
001	Each time when conversion is completed twice.
010	Each time when conversion is completed three times.
011	Each time when conversion is completed four times.
100	Each time when conversion is completed five times.
101	Each time when conversion is completed six times.
110	Each time when conversion is completed seven times.
111	Each time when conversion is completed eight times.

Note 1: <ITM[2:0]> is valid only in fixed channel repeat mode (<REPEAT>=1,<SCAN>=0).

Note 2: To stop the conversion during repeat conversion (when <REPEAT>=1, fixed channel and channel scan), zero clear the <REPEAT> (<REPEAT>=0). Do not change any bits excepting the <REPEAT> bit.

22.3.7 ADMOD4 (Mode Setting 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	SCANAREA				SCANSTA			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as zero.
7-4	SCANAREA [3:0]	R/W	Sets the range of channel scan (refer to Table 22-3).
3-0	SCANSTA[3:0]	R/W	Sets the beginning channel of the channel scan (refer to Table 22-3).

The following setting sets the channel scan single mode: ADMOD3<SCAN> = "1" and <REPEAT> = "0". Also, the following setting sets the channel scan repeat mode: ADMOD3<SCAN> = "1" and <REPEAT> = "1". Then, select the channel for channel scan. For example, if you would like to set ADMOD4<SCANSTA> = "0001"(AIN01), <SCANAREA>="0010"(3 channel scan), perform channel scan from AIN01 to AIN03 (for 3 channels).

Table 22-3 shows the <SCANSTA> setting in relation to the range of assignable value of <SCANAREA>.

Table 22-3 The range of assignable channel scan values (ADMOD4)

<SCANSTA[3:0]>	Start channel	<SCANAREA[3:0]>	The range of assignable channel scan value
0000	AIN0	0000 to 0111	1ch to 8ch
0001	AIN1	0000 to 0110	1ch to 7ch
0010	AIN2	0000 to 0101	1ch to 6ch
0011	AIN3	0000 to 0100	1ch to 5ch
0100	AIN4	0000 to 0011	1ch to 4ch
0101	AIN5	0000 to 0010	1ch to 3ch
0110	AIN6	0000 to 0001	1ch to 2ch
0111	AIN7	0000	1ch

Note: Settings other than above is prohibited. Use assignable <SCANAREA>.

22.3.8 ADMOD5 (Mode Setting 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	-	-	-	-	HPEOCF	HPADBF	EOCF	ADBF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	–	R	Read as zero.
3	HPEOCF	R	Top-priority A/D conversion completion flag (note 1) 0: Before or during conversion 1: Completion
2	HPADBF	R	Top-priority A/D conversion BUSY flag 0: Conversion stop 1: During conversion
1	EOCF	R	Normal A/D conversion end flag (note 1) 0: Before or during conversion 1: Completion
0	ADBF	R	Normal A/D conversion BUSY flag 0: Conversion stop 1: During conversion

Note 1: <EOCF> and <HPEOCF> zero cleared by reading them.

Note 2: To reduce consumption current used when shifting to IDLE mode with setting <I2AD>="0", or in STOP1/STOP2 mode, set "0" to <DACON> and <RCUT> after AD conversion, and then execute an instruction to move on to stand-by mode.

22.3.9 ADMOD6 (Mode Setting Register 6)

	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	-	-	-	-	-	-	ADRST	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	–	R	Read as zero.
1-0	ADRST[1:0]	W	Overwriting 10 with 01 allows ADC to be software reset. All registers excepting <ADCLK> bit are initialized.

Note: To perform software, initialization takes 3μs.

22.3.10 ADCMPCR0 (Monitor 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	-	-	-	-	CMPCNT0			
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CMP0EN	-	CMPCOND0	ADBIG0	AINS0			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-12	–	R	Read as zero.
11-8	CMPCNT0[3:0]	R/W	The number of counting is configured. 0000 : 1 count 0001 : 2 counts 0010 : 3 counts 0011 : 4 counts 0100 : 5 counts 0101 : 6 counts 0110 : 7 counts 0111 : 8 counts 1000 : 9 counts 1001 : 10 counts 1010 : 11 counts 1011 : 12 counts 1100 : 13 counts 1101 : 14 counts 1110 : 15 counts 1111 : 16 counts
7	CMP0EN	R/W	A/D monitor function 0 0: Disable (the number of counting for judgement is cleared) 1: Enable (if condition is satisfied, an AD monitor interrupt INTADM0 is generated)
6	–	R	Read as zero.
5	CMPCOND0	R/W	Sets the condition for judgement count. 0: Serial 1: Cumulative Using serial method, an AD monitor interrupt occurs when the condition set to the <ADBIG0> continues and counts up to the number set to the <CMPCNT0>. After exceeding the setting value, an AD monitor interrupt occurs every time when the judgement condition is true. If the condition is different from the condition set to the <ADBIG0>, the counter is cleared. Using cumulative method, an AD monitor interrupt occurs and the counter is cleared when the condition set to the <ADBIG0> is accumulated and reaches the number set to the <CMPCNT0>. Even if the condition is different from the value set to the <ADBIG0>, the value of the counter is held.
4	ADBIG0	R/W	Sets judging large and small. 0: Larger than the value of the comparison register (ADCMP0) 1: Smaller than the value of the comparison register (ADCMP0) Sets the conversion results of the target analog input larger/smaller than the value of the comparison register. Every time when the AD conversion set to AINS0[3:0] ends, judges large and small. If the result of the judgement matches to the setting of the <ADBIG0>, the counter is counted up.
3-0	AINS0[3:0]	R/W	Sets analog input as the comparison target. 0000 : AIN0 0001 : AIN1 0010 : AIN2 0011 : AIN3 0100 : AIN4 0101 : AIN5 0110 : AIN6 0111 : AIN7 Other than the above-mentioned is prohibited.

Note 1: AD monitor function is used in the fixed repeat conversion mode and the scan repeat mode.

22.3.11 ADCMPCR1 (Monitor 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	-	-	-	-	CMPCNT1			
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CMP1EN	-	CMPCOND1	ADBIG1	AINS1			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-12	–	R	Read as zero.
11-8	CMPCNT1[3:0]	R/W	Sets judging large and small. 0000 : 1 count 0001 : 2 counts 0010 : 3 counts 0011 : 4 counts 0100 : 5 counts 0101 : 6 counts 0110 : 7 counts 0111 : 8 counts 1000 : 9 counts 1001 : 10 counts 1010 : 11 counts 1011 : 12 counts 1100 : 13 counts 1101 : 14 counts 1110 : 15 counts 1111 : 16 counts
7	CMP1EN	R/W	A/D monitor function 1 0: Disable (the number of counting for judgement is cleared) 1: Enable (if condition is satisfied, an AD monitor interrupt INTADM1 is generated)
6	–	R	Read as zero.
5	CMPCOND1	R/W	Sets the condition for judgement count. 0: Serial 1: Cumulative Using serial method, an AD monitor interrupt occurs when the condition set to the <ADBIG1> continues and counts up to the number set to the <CMPCNT1>. After exceeding the setting value, an AD monitor interrupt occurs every time when the judgement condition is true. If the condition is different from the condition set to the <ADBIG1>, the counter is cleared. Using cumulative method, an AD monitor interrupt occurs and the counter is cleared when the condition set to the <ADBIG1> is accumulated and reaches the number set to the <CMPCNT1>. Even if the condition is different from the value set to the <ADBIG1>, the value of the counter is held.
4	ADBIG1	R/W	Sets judging large and small. 0: Larger than the value of the comparison register (ADCMP1). 1: Smaller than the value of the comparison register (ADCMP1). Sets the conversion results of the target analog input larger/smaller than the value of the comparison register. Every time when the AD conversion set to AINS1[3:0] ends, judges large and small. If the result of the judgement matches to the setting of the <ADBIG1>, the counter is counted up.
3-0	AINS0[3:0]	R/W	Sets analog input as the comparison target. 0000 : AIN0 0001 : AIN1 0010 : AIN2 0011 : AIN3 0100 : AIN4 0101 : AIN5 0110 : AIN6 0111 : AIN7 Other than the above-mentioned is prohibited.

Note: AD monitor function is used in the fixed repeat conversion mode and the scan repeat mode.

22.3.12 ADCMP0 (AD Conversion Result Compare 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	-	-	-	-	AD0CMP			
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	AD0CMP							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-12	–	R	Read as zero.
11-0	AD0CMP[11:0]	R/W	Sets the comparison value of A/D conversion.

Note: To write a value to this register or to change the settings, disable the AD monitor function first (ADCMPCR0<CMP0EN> = "0", ADCMPCR1<CMP1EN> = "0").

22.3.13 ADCMP1 (AD Conversion Result Compare 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	-	-	-	-	AD1CMP			
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	AD1CMP							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-12	–	R	Read as zero.
11-0	AD1CMP[11:0]	R/W	Sets the comparison value of A/D conversion.

Note: To write a value to this register or to change the settings, disable the AD monitor function first (ADCMPCR0<CMP0EN> = "0", ADCMPCR1<CMP1EN> = "0").

22.3.14 ADREG00 to ADREG07 (AD Conversion Result Register)

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

Bit	Bit Symbol	Type	Function
31-20	ADR_MIR [11:0]	R	12-bit normal A/D conversion results are stored. If you read the ADREGx register during AD conversion, the previous conversion result is read.
19-18	–	R	Read as zero.
17	ADOVRF_MIR	R	Overrun flag 0: Not generated 1: Generated If AD conversion result is over written before reading the AD conversion result register (ADREGx), this bit is set to "1". This flag is zero cleared when reading the ADREGx register.
16	ADRF_MIR	R	AD conversion result storage flag 0: Conversion result is not stored. 1: Conversion result is stored. If the conversion result is stored, this bit is set to "1". This flag is zero cleared when the ADREGx register is read.
15-14	–	R	Read as zero.
13	ADOVRF	R	Overrun flag 0: Not generated. 1: Generated. If the conversion result is overwritten before reading the AD conversion result register (ADREGx),this bit is set to "1". This flag is zero cleared when the ADREGx register is read.
12	ADRF	R	AD conversion result storage flag 0: Conversion result is NOT stored. 1: Conversion result is stored. If a conversion result is stored, this bit is set to "1". This flag is zero cleared when the ADREGx register is read.
11-0	ADR[11:0]	R	12-bit normal A/D conversion result is stored. If you read the ADREGx register during AD conversion, the previous conversion result is read.

Note: Since <ADR_MIR>, <ADOVRF_MIR> and <ADRF_MIR> are read as same as <ADR>, <ADOVRF> and <ADRF>. Use one of them.

22.3.15 ADREGSP (Top-priority AD Conversion Result Register)

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

Bit	Bit Symbol	Type	Function
31-20	ADSPR_MIR [11:0]	R	12-bit top-priority A/D conversion results are stored. If you read the ADREGSP register during AD conversion, the previous conversion result is read.
19-18	–	R	Read as zero.
17	ADOVRSPF _MIR	R	Overrun flag 0: Not generated 1: Generated If AD conversion result is over written before reading the AD conversion result register (ADREGSP), this bit is set to "1". This flag is zero cleared when reading the ADREGSP register.
16	ADSPRF_MIR	R	Top-priority AD conversion result storage flag 0: Conversion result is NOT stored. 1: Conversion result is stored. If a result of top-priority conversion is stored, this bit is set to "1". This flag is zero cleared when reading the ADREGSP register.
15-14	–	R	Read as zero.
13	ADOVRSPF	R	Overrun flag 0: Not generated. 1: Generated. If the result of top-priority conversion is overwritten before reading the top-priority AD Conversion Result Register (ADREGSP), this bit is set to "1". This flag is zero cleared when reading the ADREGSP register.
12	ADSPRF	R	Top-priority AD conversion storage flag 0: Conversion result is NOT stored. 1: Conversion result is stored. When a conversion result of top-priority AD conversion is stored, this bit is set to "1". This flag is zero cleared when reading the ADREGSP register.
11-0	ADSPR[11:0]	R	12-bit top-priority A/D conversion result is stored. If you read the ADREGSP register during AD conversion, the previous conversion result is read.

Note: Since <ADSPR_MIR>, <ADOVRSPF_MIR> and <ADSPRF_MIR> are read as same as <ADSPR>, <ADOVRSPF> and <ADSPRF>. Use one of them.

22.3.16 ADILVTRGSEL (Trigger 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	HPTRGSEL				TRGSEL			
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	TRGSELEN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as zero.
15-12	HPTRGSEL[3:0]	R/W	Selects a trigger for start-up of top-priority AD conversion 0000 : $\overline{TRG0}$ 0001 : $\overline{TRG1}$ 0010 : $\overline{TRG2}$ 0011 : $\overline{TRG3}$ 0100 : $\overline{TRG4}$ 0101 : $\overline{TRG5}$ 0110 : $\overline{TRG6}$ 0111 : $\overline{TRG7}$ 1000 : $\overline{TRG8}$ 1001 : $\overline{TRG9}$ Other than the above-mentioned is prohibited.
11-8	TRGSEL[3:0]	R/W	Selects a trigger for start-up of normal AD conversion 0000 : $\overline{TRG0}$ 0001 : $\overline{TRG1}$ 0010 : $\overline{TRG2}$ 0011 : $\overline{TRG3}$ 0100 : $\overline{TRG4}$ 0101 : $\overline{TRG5}$ 0110 : $\overline{TRG6}$ 0111 : $\overline{TRG7}$ 1000 : $\overline{TRG8}$ 1001 : $\overline{TRG9}$ Other than the above-mentioned is prohibited.
7-1	–	R	Read as zero.
0	TRGSELEN	R/W	Controls a selected trigger operation 0: Trigger is disabled 1: Trigger is enabled

Note: For detail a trigger for start-up , refer to "ADC " of Chapter "Product Information".

22.4 Description of Operations

22.4.1 Usage note about the activation of analog conversion

To start AD conversion, write "1" to the ADMOD1<DACON>, and wait for 3 μs until the internal circuit stabilizes. Then, write "1" to the ADMOD0<ADS>. If you do not use this function, write "0" to the ADMOD1<DACON>. The consumption current of the analog circuit is reduced.

22.4.2 AD conversion mode

Two types of A/D conversion are supported: normal AD conversion and top-priority AD conversion.

22.4.2.1 Normal AD conversion

Normal AD conversion supports the following four operation types. Operation modes are selected by setting ADMOD3<REPEAT>, <SCAN>.

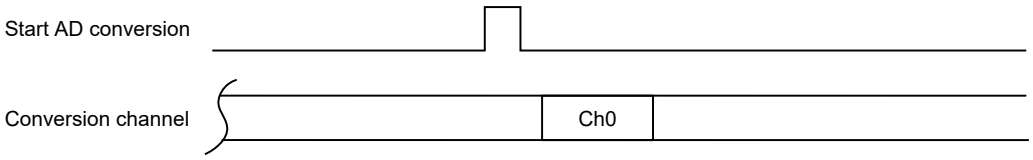
- Fixed channel single conversion mode
- Channel scan single conversion mode
- Fixed channel repeat conversion mode
- Channel scan repeat conversion mode

(1) Fixed channel single conversion mode

If ADMOD3<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 which is selected by ADMOD2<ADCH>. After the AD conversion, ADMOD5<EOCF> is set to "1", ADMOD5<ADBF> is zero cleared, and an interrupt request of INTAD is generated. <EOCF> is cleared to zero upon read.

The figure below shows an example of converting AIN0 in fixed channel single conversion mode.

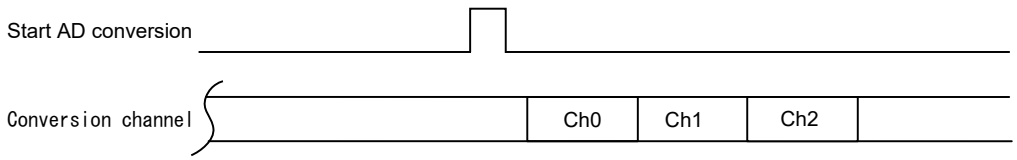


(2) Channel scan single conversion mode

If ADMOD3<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 the scan channel area selected by ADMOD4<SCANAREA> from the start channel selected by ADMOD4<SCANSTA>. After AD scan conversion, ADMOD5<EOCF> is set to "1", ADMOD5<ADBF> is zero cleared, and an interrupt request of INTAD is generated. <EOCF> is zero cleared upon read.

The figure below shows an example of converting from AIN0 to AIN2 in channel scan single conversion mode.

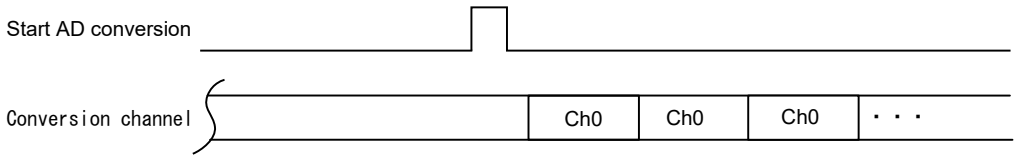


(3) Fixed channel repeat conversion mode

If ADMOD3<REPEAT>, <SCAN> is set to "10", AD conversion is performed in fixed channel repeat conversion mode.

In this mode, AD conversion is repeated for the number of times set to ADMOD3<ITM> (interrupt request generating timing for INTAD can be selected) for the one channel selected by ADMOD2<ADCH> . After the number of AD conversion set to <ITM> is repeated, ADMOD5<EOCF> is set to "1", but ADMOD5<ADBF> is not zero cleared and keeps "1". <EOCF> is zero cleared upon read.

The figure below is an conversion example of AIN0 in fixed channel repeat conversion mode.

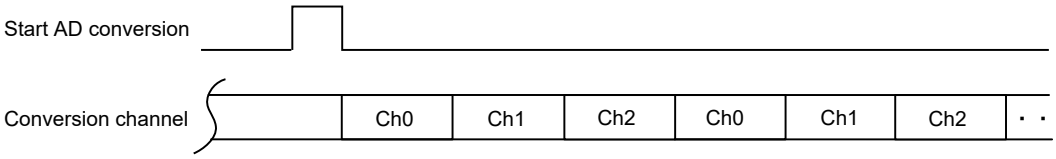


(4) Channel scan repeat conversion mode

If ADMOD3<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 the scan channel area in the channel selected by ADMOD4<SCANSTA> from the start channel selected by ADMOD4<SCANAREA>. Each time one AD scan conversion is completed, ADMOD5<EOCF> is set to "1" and the INTAD interrupt request is generated. ADMOD5<ADBF> is not zero cleared and remains at "1". <EOCF> is cleared to "0" upon read.

The figure below shows an operation example of converting AIN0 to AIN2 in the channel scan repeat conversion mode.



22.4.2.2 Top-priority AD conversion

Top-priority AD conversion can be performed by interrupting ongoing normal AD conversion. If the top-priority AD conversion interrupts into normal AD conversion, the normal AD conversion starts from the stopped channel after the top-priority conversion.

The fixed channel single conversion is the only operation mode. The settings to ADMOD3<REPEAT>, <SCAN> are invalid. When conditions to start operation are met, a conversion is performed just once for a channel selected by ADMOD2<HPADCH>. When conversion is completed, the top-priority AD conversion completion interrupt (INTADHP) is generated, ADMOD5<HPEOCF> is set to "1" and <HPADBF> is zero cleared. <HPEOCF> returns to "0" upon read.

If a top-priority AD conversion is activated during the operation of another top-priority conversion, the previous conversion becomes invalid and the new operation becomes valid.

22.4.3 AD monitor function

This function is used for configuring the fixed channel repeat mode and the scan repeat mode.

If ADCMPCR0<CMP0EN> and ADCMPCR1<CMP1EN> are set to "1", AD monitor function is enabled. Two monitor function can be enabled concurrently.

Here is an example of ADCMPCR0 (same as ADCMPCR1).

Analog input for comparison is set to ADCMPCR0<AINS0[3:0]>. Large or small judgement is set to <ADBIG0>. Conditions of this comparison count is set to <CMPCOND0>. The number of comparison count is set to <CMPCNT0[3:0]>.

Once AD conversion starts, the AD converter checks comparison conditions (smaller/larger than values of the compare register) for each AD conversion. If the result of the comparison meets the settings of <ADBIG0>, the AD converter counts up the counter value.

There are two types of comparison condition: sequential method and cumulative method.

In the sequential method, an AD monitor interrupt (INTADM0) occurs if the condition set to <ADBIG0> continues and reaches the number of counts! set to <CMPCNT0[3:0]>. An interrupt occurs without clearing the counter if the result of comparison meets the condition even after reaching the number of counts set to <CMPCNT0[3:0]>. The value of the counter is zero cleared only when the condition does not meet the set condition of <ADBIG0>.

In the cumulative method, the counter is zero cleared when the total number of times that the condition set to <ADBIG0> meets reaches the number set to <CMPCNT0[3:0]>. An AD monitor interrupt (INTADM0) occurs at this time. The counter value is kept even if the result of the comparison is different from the set value of the counter. If values of the Conversion Result Register configured by the ADCMPCR0 register are the same as those of the tar get register, the AD converter does not count up. An AD conversion interrupt (INTADM0) does not occur either.

This comparison operation is performed each time a result is stored in a corresponding conversion result register. An interrupt (INTADM0) occurs when conditions meet (including counting). Since the conversion result register assigned to perform the AD monitor function is usually read by software, the conversion result storage flag ADREG<ADRF> and the overrun flag ADREG<ADOVRF> remain being set. Do not use the conversion result register when you use the AD monitor function.

1. AIN0 input is set to the fixed channel repeat conversion mode and compare values of the AD Conversion Result Compare Register (0x0888).
   - ADMOD3=0x0002: fixed channel repeat conversion

     AD conversion completion interrupt (INTAD) is disabled.
   - ADCMPCR0 =0x0280: compare target channel: AIN0, size determination: larger than a value of the compare register. Compare counting condition: sequential method. AD monitor function: enabled. Size determination count: 3 counts.
   - ADCMP0=0x0888: AD Conversion Result Compare Register (compared value 0x0888)

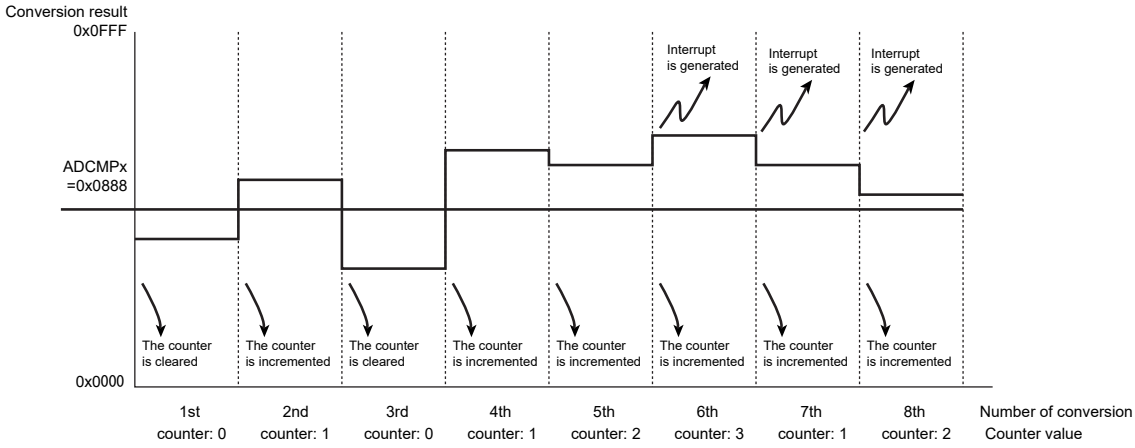


Figure 22-3 AD monitor function (fixed channel repeat and sequential method)

2. AIN0 is set to fixed channel repeat conversion and compare the value of the AD Conversion Result Compare Register (0x0888).
   - ADMOD3=0x0002: fixed channel repeat conversion

     AD conversion completion interrupt (INTAD) is disabled.
   - ADCMPCR0 =0x02A0: compare target channel: AIN0, size determination: larger than a value of the compare register. compare counting condition: cumulative method. AD monitor function: enabled. size determination count: 3 counts
   - ADCMP0=0x0888: AD Conversion Result Compare Register (compared value 0x0888)

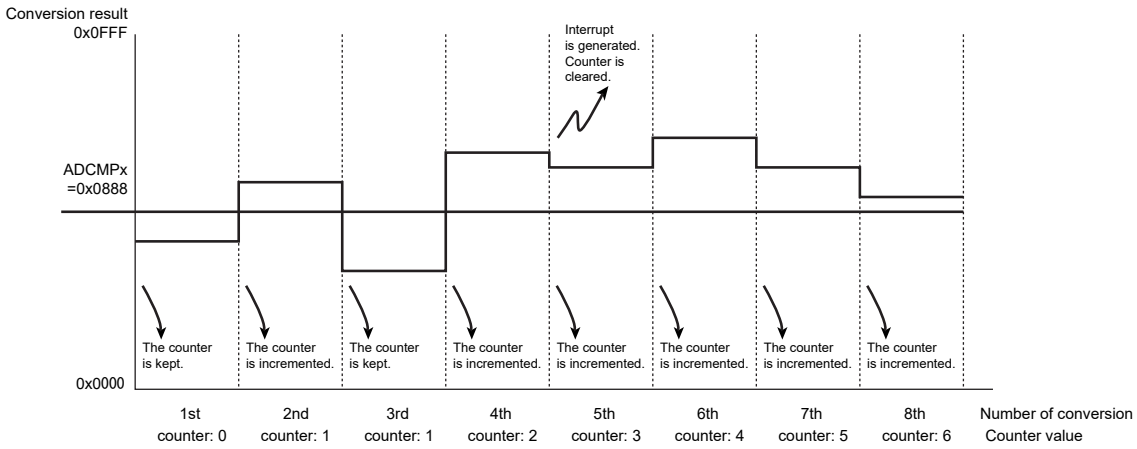


Figure 22-4 AD Monitor Function (fixed channel repeat and cumulative method)

22.4.4 Selecting the input channel

After reset, ADMOD3<REPEAT>, <SCAN> is initialized to "00" and ADMOD2<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 (ADMOD3<SCAN> = "0")

     One channel is selected from analog input pins AIN0 through AIN7 by setting ADMOD2<ADCH> to an appropriate setting.
   - If the analog input channel is used in a scan state (ADMOD3<SCAN> = "1")

     One scan mode is selected from the scan modes by setting ADMOD4<SCANSTA> and ADMOD4<SCANAREA> 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. In this mode, 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. The normal AD conversion restarts from the channel previously stopped after the top-priority AD conversion is completed.

22.4.5 Details of AD Conversion

22.4.5.1 Starting AD conversion

The normal AD conversion is enabled by setting "1" to ADMOD0<ADS>. The top-priority AD conversion is enabled by setting "1" to ADMOD0<HPADS>. Starting by setting values is called software start.

Also, hardware activation can be selected.

Hardware activation sources are selected by ADMOD1<HPADHWS>, <ADHWS> and ADILVTRGSEL<HPTRGSEL>, <TRGSEL>. Setting "0" to ADMOD1<HPADHWS>, <ADHWS>, an AD converstion is started up by an external trigger. Setting "1" to ADMOD1 <HPADHWS>,<ADHWS>, an AD conversion is started up by internal triggers. Internal trigger events are selected by ADILVTRGSEL <HPTRGSEL> ,<TRGSEL>. If "1" is set to ADILVTRGSEL <TRGSELEN>, a selected internal trigger is enabled.

To enable hardware activation, set "1" to ADMOD1<ADHWE> for normal AD conversion and to ADMOD1<HPADHWE> for top-priority AD conversion.

Even if hardware start is enabled, start with software can be done.

Note: When using external trigger is used as a hardware activating source for the top-priority AD conversion, external trigger cannot be selected for the normal AD conversion hardware start.

22.4.5.2 AD conversion

When normal AD conversion starts, the AD conversion Busy flag (ADMOD5<ADBF>) which indicates that AD conversion is under way is set to "1".

When top-priority AD conversion starts, the top-priority AD conversion Busy flag (ADMOD5<HPADBF>) showing that AD conversion is underway is set to "1". At that time, the value of the Busy flag ADMOD5<EOCF> and <ADBF> for normal AD conversion before the start of top-priority AD conversions are retained.

Note: Do not execute the normal AD conversion when the top-priority AD conversion is underway. A top-priority AD conversion completion flag is not set. Also, a previous normal AD conversion flag is not cleared.

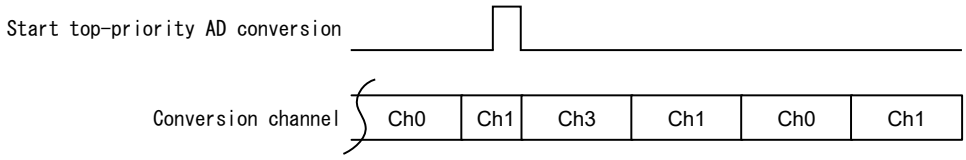
22.4.5.3 Top-priority AD conversion during normal AD conversion

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 ADMOD0<HPADS> 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 &! nbsp; by <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 AIN2 and if <HPADS> is set to "1" during AIN1 conversion, AIN1 conversion is suspended, and conversion is performed for a channel designated by <HPADCH> (AIN3 in the case shown below). After the result of conversion is stored in ADREGSP, channel repeat conversion is resumed, starting from AIN1.



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 ADMOD3<REPEAT>.When ongoing AD conversion is completed, the repeat conversion mode terminates, and ADMOD5<ADBF> is set to "0".

22.4.5.5 Reactivating normal AD conversion

Setting "1" to ADMOD0<ADS> during normal AD conversion reactivates the normal AD conversion. The previous normal AD conversion is immediately discontinued when a new normal AD conversion is reactivated. At this time, the normal AD conversion busy flag ADMOD5<ADBF>, the normal AD conversion completion flag ADMOD5<EOCF> and the AD conversion result storage flag ADREG00 to 15<ADOVRF> <ADRF> are zero cleared.

If H/W activation of top-priority AD conversion is authorized during normal AD conversion, ongoing AD conversion is reactivated when requirements for activation using a H/W activation resource are met. At this time, the previous normal AD conversion is immediately discontinued. <ADBF>, <EOCF>, <AOVRF> and <ADRF> are zero cleared.

22.4.5.6 Conversion completion

(1) Completing normal AD conversion

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, and values of two registers change: the register ADMOD0<EOCFN> which indicates the completion of AD conversion and the register ADMOD0<ADBFN> which indicates conversion is ongoin.Interrupt requests, 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 the AD conversion result registers (ADREG00 to 07) corresponding to a channel.

In fixed-channel repeat conversion mode, the conversion results are sequentially stored in storage registers ADREG through ADREG07, according to the interrupt condition set to ADMOD3<ITM>.

Interrupt requests, flag changes and conversion result registers in each mode are as shown below.

- <u>Fixed-channel single conversion mode</u>

  After AD conversion completed, <EOCF> is set to "1", <ADBF> is cleared to "0", and the interrupt request is generated.

  Conversion results are stored a conversion result register correspond to a channel.

- <u>Channel scan single conversion mode</u>

  After the channel scan conversion is completed, <EOCF> is set to "1", <ADBF> is set to "0", and an interrupt request is generated.

  Conversion results are stored a conversion result register correspond to a channel.

- <u>Fixed-channel repeat conversion mode</u>

  The <ADBF> is not cleared to "0". It remains at "1". The timing with which the interrupt request INTAD is generated can be selected by setting <ITM> to an appropriate setting. ADREG04-ADREG07 can be used only in the fixed channel repeat conversion mode.

  a. One-time conversion

     With <ITM> set to "00", an interrupt request is generated each time one AD conversion set to <ADCH> is completed. In this case, the conversion results are always stored in the storage register s ADREG00 in sequential order. After the conversion result is stored, <EOCF> changes to "1".

  b. Eight-time conversions

With <ITM> set to "111", an interrupt request is generated each time eight AD conversions set to <ADCH> are completed. In this case, the conversion results are always stored in the storage registers ADREG00 through ADREG07 in sequential order. After the conversion results are stored in ADREG07, <EOCF> is set to "1", and storage of subsequent conversion results from ADREG00.

- Fixed-channel repeat conversion mode

With <EOCF> set to "1", a INTAD interrupt request is generated each time one scan conversion is completed. <ADBF> is not cleared to zero and remains at "1".

With ADMOD4<SCANAREA> set to "0011" (4-channel scan), four channel scans are performed from the Start Channel designated by ADMOD4<SCANSTA>. Each time when a conversion of the final channel is completed, <EOCF> is set to "1", an interrupt request is generated, and four channel scanning starts from the Start Channel again. Since this mode is a repeat mode, <ADBF> is not zero cleared and maintains "1".

Conversion results are stored in a conversion result register corresponding to the channel.

(2) Completing top-priority AD conversion

After the top-priority AD conversion is completed, the top-priority AD conversion completion interrupt (INTADHP) is generated. ADMOD5<HPEOCF> which indicates the completion of top-priority AD conversion is set to "1".

Conversion results are stored in the conversion result register ADREGSP.

(3) Data polling

To confirm the completion of AD conversion without using interrupts, data polling can be used. When AD conversion is completed, ADMOD5<EOCF> are 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 word access. If <ADOVRF> = "0" and <ADRF> = "1" in ADREG00 to ADREG07, a correct conversion result has been successfully obtained.

A top-priority AD conversion can use data polling, too.

22.4.5.7 Interrupt Timing and Conversion Result Register

Table 22-4 shows the correlation between AD conversion modes, interrupt timing and flags. Table 22-5, Table 22-6 and Table 22-7 show the correlation between analog input channels and conversion result registers.

Table 22-4 Correlation between AD conversion mode, interrupt timing and flag operation

Conversion mode		Scan / repeat mode setting (ADMOD3)			Interrupt generation timing	Conversion Status Flag (ADMOD5)		
		<REPEAT>	<SCAN>	<ITM[2:0]>		<EOCF>/ <HPEOCF> set timing (note 1)	<ADBF> (after interrupt)	<HPADBF> (after interrupt)
Normal conversion	Fixed channel single conversion	0	0	–	After conversion	After conversion	0	–
	Fixed channel repeat conversion	1	0	000	Each 1 conversion	After 1 conversion is completed	1	–
				001	Each 2 conversions	After 2 conversions are completed	1	–
				010	Each 3 conversions	After 3 conversions are completed	1	–
				011	Each 4 conversions	After 4 conversions are completed	1	–
				100	Each 5 conversions	After 5 conversions are completed	1	–
				101	Each 6 conversions	After 6 conversions are completed	1	–
				110	Each 7 conversions	After 7 conversions are completed	1	–
				111	Each 8 conversions	After 8 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	–	One scan conversion is completed.	One scan conversion is completed.	1	–
Top-priority conversion		–	–	–	After completion of conversion	After completion of conversion	–	0

Note 1: ADMOD5<EOCF><HPEOCF> are cleared to zero upon read.

Note 2: In repeat mode, ADMOD5<ADBF> are not zero cleared even if the interrupt is generated. To stop repeat mode, write zero to ADMOD3<REPEAT>, then <ADBF> is zero cleared when the AD conversion is completed.

Table 22-5 Analog input channels and AD conversion result registers (Fixed channel single mode)

Fixed channel single mode	
Channel	Storage registers
AIN0	ADREG00
AIN1	ADREG01
AIN2	ADREG02
AIN3	ADREG03
AIN4	ADREG04
AIN5	ADREG05
AIN6	ADREG06
AIN7	ADREG07

Table 22-6 Analog input channels and AD conversion result registers (Fixed channel repeat mode)

Fixed channel repeat mode		
ADMOD3<ITM[2:0]>		Storage register
000	An interrupt occurs each time.	ADREG00
001	An interrupt occurs every twice.	ADREG00 to ADREG01
010	An interrupt occurs every 3 times.	ADREG00 to ADREG02
011	An interrupt occurs every 4 times.	ADREG00 to ADREG03
100	An interrupt occurs every 5 times.	ADREG00 to ADREG04
101	An interrupt occurs every 6 times.	ADREG00 to ADREG05
110	An interrupt occurs every 7 times.	ADREG00 to ADREG06
111	An interrupt occurs every 8 times.	ADREG00 to ADREG07

Table 22-7 Analog input channels and AD conversion result registers (Channel scan single mode / repeat mode)

Channel scan single mode / repeat mode				
<SCANSTA> (Start Channel)		<SCANAREA> (Scan channel width)		Storage registers
0000	AIN0	0000 to 0111	1ch to 8ch	ADREG00 to ADREG07
0001	AIN1	0000 to 0110	1ch to 7ch	ADREG01 to ADREG07
0010	AIN2	0000 to 0101	1ch to 6ch	ADREG02 to ADREG07
0011	AIN3	0000 to 0100	1ch to 5ch	ADREG03 to ADREG07
0100	AIN4	0000 to 0011	1ch to 4ch	ADREG04 to ADREG07
0101	AIN5	0000 to 0010	1ch to 3ch	ADREG05 to ADREG07
0110	AIN6	0000 to 0001	1ch to 2ch	ADREG06 to ADREG07
0111	AIN7	0000	1ch	ADREG07

Notes on designing for AD converter inputs

<An output impedance of the external signal source which is connected with AIN pin>

An output impedance of the external signal source which is connected with AIN pin is equal or less than R_{EXAIN} shown below formula.

- Calculating formula of allowable value of output impedance of the external signal source -

The maxmum value of an output impedance connected with AIN pin: $R_{EXAIN} < Tscyc \div (ADCLK \times C_{ADC} \times \ln(2^{14})) - R_{AIN}$

MCU information	Symbol	Min	Typ	Max	Unit
ADC clock frequency	ADCLK	4	-	40	MHz
Total AIN input capacity in MCU	C_{ADC}	-	-	12.2	pF
AIN resistance in MCU	R_{AIN}	-	-	1	kΩ
Cycle number in the sample hold period	Tscyc	10	-	320	Cycle

R_{EXAIN} maximum value list (ADCLK = 40MHz)

Tcsyc	R_{EXAIN}	Unit
10	1.1	kΩ
20	3.2	kΩ
30	5.3	kΩ
40	7.5	kΩ
80	15.9	kΩ
160	32.8	kΩ
320	66.6	kΩ

< Addition of stabilizing capacity >

If high-speed AD conversion is required and the sample hold period cannot meet the conditions of calculating formula of allowable values of output impedance of external signal source, add stabilizing capacity to the AIN pin. The additional capacity depends on external circuit. Although the capacity depended on the external circuit is different from the each board set, add the capacity from about 0.1μF to 1μF, appropriate amount for your circuit board.

Set the capacity to be added next to the AIN pin.

< Adjustment of sample hold period>

Generally, by setting the sample hold period long, you can make the input voltage of the comparator in the ADC circuit as same as the input voltage of the AIN pin can reduce the error of an AD conversion.

Although, in case that the sample hold period is too long, the error of an AD conversion may be increased because the voltage held in sample hold circuit is changed.

Because the suitable sample hold period is depended on the each board set, please decided the suitable sample hold period on your board set.

Notes of the use of the AD converter

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.

Please keep the following restrictions and use.

(1) Do not change clock gear during AD converting. And condition is below.
Gear ratio down from 1 :1 to 1 :N (N=2,4,8,16)

(2) Keep frequency ratio between f_{ADCLK} and f_{SYS} as below condition.
$f_{ADCLK} \div f_{SYS} < 5.5$ times

(3) For detection of end of AD conversion, do not use BUSY flag (ADxMOD5 <HPADBF> ,<ADBF>). Please use end of conversion flag (ADxMOD5<HPEOCF>, <EOCF>) or use end of conversion interrupt (INTADxHP, INTADx).

(4) When using Top-priority AD conversion with repeat conversion mode (Fixed channel, Channel scan) in parallel , please make the change AD conversion completion interrupt request (INTADx) prohibited before stopping repeat conversion.

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 function (available only in the products that have $\overline{\text{ALARM}}$ pin.)
6. Alarm interrupt
7. Clock correction function
8. 1 Hz clock output

23.2 Block Diagram

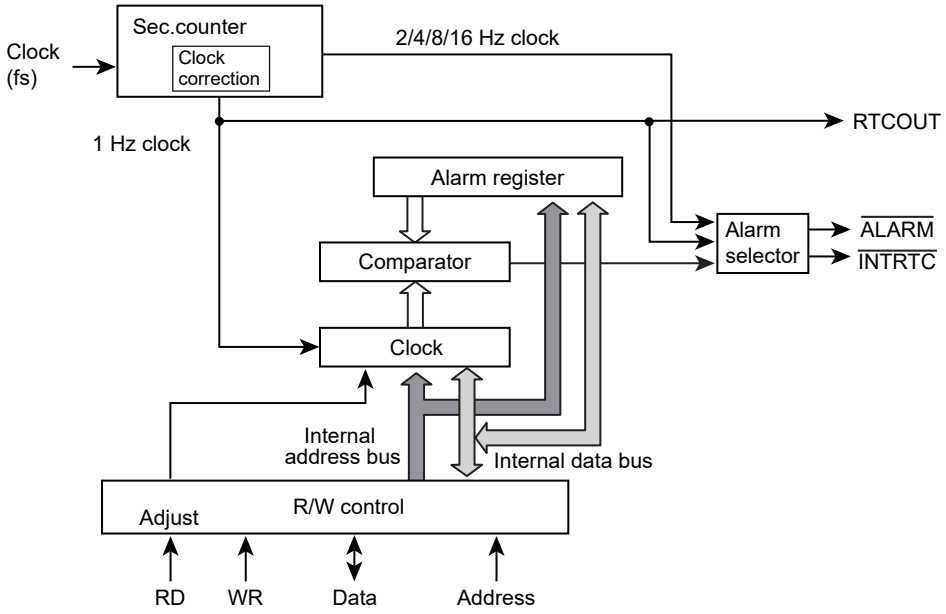


Figure 23-1 Block Diagram

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 table below shows control registers and their addresses.

For detail of the base address, refer to "Address lists of peripheral functions" of "Memory Map" chapter.

RTC has two functions, PAGE0 (clock) and PAGE1 (alarm), which share some parts of registers.

The PAGE can be selected by setting RTCPAGER<PAGE >.

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) Selection register of 24-hour,12-hour (PAGE1)	RTCMONTHR	0x0006
Year column register (PAGE0) Leap year register (PAGE1)	RTCYEARR	0x0007
PAGE register	RTCPAGER	0x0008
- (note 1)	-	0x0009
- (note 1)	-	0x000A
- (note 1)	-	0x000B
Reset register	RTCRESTR	0x000C
- (note 1)	-	0x000D
Protect register	RTCPROTECT	0x000E
Correction Function Control Register	RTCADJCTL	0x000F
Correction Value Register	RTCADJDAT	0x0010, 0x0011

Note:"0" is read by reading the address. Writing is disregarded.

23.3.2 Control Register

Reset operation initializes the following registers.

- RTCPAGER<PAGE>, <ADJUST>, <INTENA>
- RTCRESTR
- RTCPROTECT
- RTCADJCTL
- RTCADJDAT

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	Bit8	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	–	–	Adjust-ment function	Clock en-able	Alarm en-able	–	PAGE setting	PAGE register
RTCRESTR		1 Hz enable	16 Hz enable	Clock reset	Alarm reset	–	2Hz enable	4 Hz enable	8 Hz enable	Reset register
RTCPROTECT		Protect code								Clock correction function register protection
RTCADJCTL		–	–	–	–	Correction reference time			Correction enable	Correction function control
RTCADJDAT	Correction value									Correction value

Note: Reading RTCSECR, RTCMINR, RTCHOURR, RTCDAYR, RTCMONTHR, RTCYEARR of PAGE0 captures the current state.

Table 23-2 PAGE1 (alarm function) registers

Symbol	Bit8	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	–	–	Adjust-ment function	Clock en-able	Alarm en-able	–	PAGE setting	PAGE register
RTCRESTR		1 Hz Enable	16 Hz Enable	Clock reset	Alarm reset	–	2 Hz enable	4 Hz enable	8 Hz enable	Reset register
RTCPROTECT		Protect code								Clock correction function register protection
RTCADJCTL		–	–	–	–	Correction reference time			Correc-tion ena-ble	Correction function control
RTCADJDAT	Correction value									Correction value

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. 111_1111 : Don’t compare Minutes at alarm function.

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 11_1111 : Don't compare Hour at alarm function.

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 11_1111 : Don't compare hour at alarm function.

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 111: Don't compare day of the week at alarm function.

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 11_1111 : Don’t compare day at alarm function.

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 performed.

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	-	DIS2HZ	DIS4HZ	DIS8HZ
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	DIS2HZ	R/W	2 Hz 0:Enable 1: Disable
1	DIS4HZ	R/W	4 Hz 0:Enable 1: Disable
0	DIS8HZ	R/W	8 Hz 0:Enable 1: Disable

Note: A read-modify-write operation cannot be performed.

The setting of <DIS1HZ>, <DIW2HZ>, <DIS4HZ> and <DIS16MHZ>, RTCPAGER<ENAALM> used for alarm, 1Hz, 2Hz, 4Hz, 8Hz and 16Hz interrupt is shown as below.

<DIS1HZ>	<DIS2HZ>	<DIS4HZ>	<DIS8HZ>	<DIS16HZ>	RTCPAGER <ENAALM>	Interrupt source signal
1	1	1	1	1	1	Alarm
0	1	1	1	1	0	1 Hz
1	0	1	1	1	0	2 Hz
1	1	0	1	1	0	4Hz
1	1	1	0	1	0	8Hz
1	1	1	1	0	0	16 Hz
Others						Interrupt not generated.

23.3.3.12 RTCPROTECT(Protect register)

	7	6	5	4	3	2	1	0
Bit symbol	RTCPROTECT							
After reset	1	1	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
7-0	RTCPROTECT	R/W	Clock correction function register protection 0xC1: Write enable. 0xC1: Write disable. In the initial state, RTCPROTECT is "0xC1" and write enable. If RTCPROTECT is set to a value other than "0xC1", RTCADJCTL and RTCADJDAT will be write disable.

23.3.3.13 RTCADJCTL (Correction Function Control Register)

	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	AJSEL			AJEN
After reset	0	0	0	0	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
7-4	–	R	Read as "0".
3-1	AJSEL	R/W	Correction reference time setting 000: 1 second 001: 10 seconds 010: 20 seconds 011: 30 seconds 100: 1 minute 101 - 111: Reserved Set a correction reference time.
0	AJEN	R/W	Correction function control 0: Disabled 1: Enabled

23.3.3.14 RTCADJDAT (Correction Value Register)

	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	ADJDAT
After reset	0	0	0	0	0	0	0	Undefined
	7	6	5	4	3	2	1	0
bit symbol	ADJDAT							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
15-9	–	R	Read as "0".
8-0	ADJDAT	R/W	Correction value 0_0000_0000 : No correction 0_0000_0001 : 32768 + 1 0_0000_0010 : 32768 + 2 ・ 0_1111_1110 : 32768 + 254 0_1111_1111 : 32768 + 255 1_0000_0000 : 32768 - 256 1_0000_0001 : 32768 - 255 ・ 1_1111_1110 : 32768 - 2 1_1111_1111 : 32768 - 1 Sets a correction value per second. The 8th is a sign bit. If the 8th bit is "0", a plus correction is applied. If the bit is "1", a minus correction is applied. Specifies a correction value using bit 7 to 0.

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.

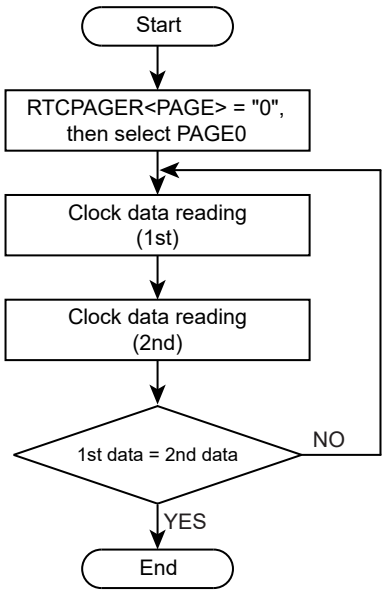


Figure 23-2 Flowchart of the clock data reading

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.

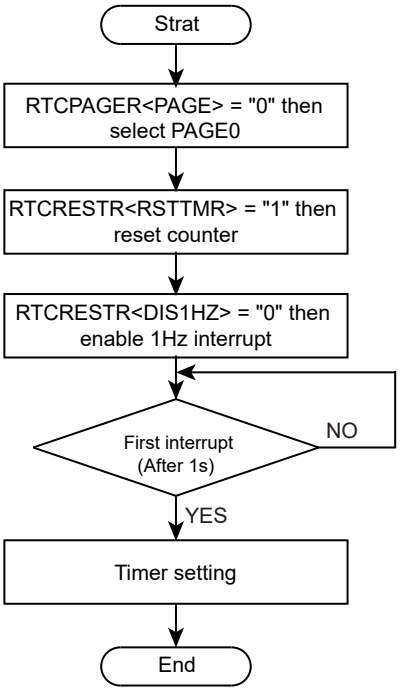


Figure 23-3 Flowchart of the clock data writing

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

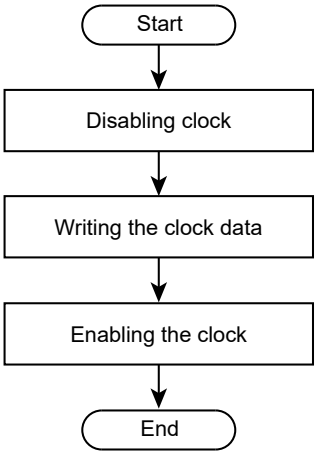


Figure 23-4 Flowchart of the disabling clock

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 signals is output to the $\overline{\text{ALARM}}$ pin if the product provide the ALARM pin.

1. "Low" pulse (when the alarm register corresponds with the clock)
2. 1, 2, 4, 8 or 16Hz cycle "Low" pulse

In any cases shown above, the RTC 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 Usage of alarm function

"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 $\overline{\text{ALARM}}$ 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

If some alarm registers are set to "1", RTC doesn't compare the term. For example, if RTCDATER is set to "11_1111" and RTCDAYR is set to "111", the alarm will be output at noon (12:00) every day.

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.

23.5.2 1, 2, 4, 8 or 16 Hz cycle "Low" pulse

The RTC outputs a "Low" pulse cycle to the $\overline{\text{ALARM}}$ pin by setting RTCPAGER<INTENA>="1" after setting RTCPAGER<ENAALM>= "0" and RTCRESTR. It is required that one of RTCRESTR<DIS1HZ>, <DIS2HZ>, <DIS4HZ>, <DIS8HZ> or <DIS16HZ> is set to "0".

The RTC outputs one cycle pulse of low-speed clock which correspond to RTCRESTR setting. It generates an INTRTC interrupt simultaneously.

23.6 Clock Correction Function

The clock correction function can precisely adjust the deviation of the clock.

In the Figure 23-5, T1 indicates one second. One second is generated by counting fs (32768Hz) 32768 times. The clock correction function adjusts the number of counts of T2 that is an one second of the correction reference time (Tall). The correction reference time is selected either among 1, 10, 20, 30 seconds or 1 minute with RTCADJCTL<AJSEL>. A count value of T2 can be adjustable from 32768-255 to 32768+256 with RTCADJDAT<ADJDAT>.

Symbol	Item	Description
Tall	Correction reference time	Selects either among 1, 10, 20, 30 seconds or 1 minute with RTCADJCTL<AJSEL>.
T1	1 second	Counts fs 32768 times
T2	Count correction	Adjust a count value with RTCADJDAT<ADJDAT> by plus/minus 32768 counts

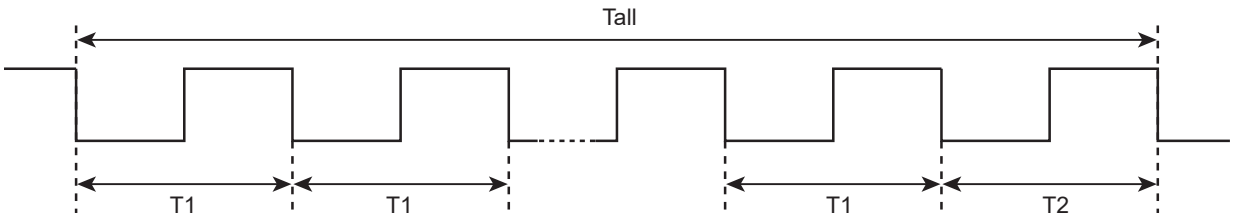


Figure 23-5 Clock Correction

The correction function related register, RTCADJCTL and RTCADJDAT, can be disabled with the RTCPROTECT register. In the initial state, RTCPROTECT is “0xC1” and write enable. If RTCPROTECT is set to a value other than “0xC1”, RTCADJCTL and RTCADJDAT will be write disable.

23.7 1Hz Clock Output Function

RTCOUT pin outputs 1Hz clock. This clock is adjusted to operate on a 50% duty ratio. If the clock correction function is used, a duty ratio may be varied due to the error corrections.

24. Low Voltage detection circuit (LVD)

The low voltage detection circuit generates reset or an interrupt (INTLVD) by detecting a decreasing voltage.

Note: INTLVD is a factor of non-maskable interrupts (NMI).

24.1 Configuration

The low voltage detection circuit consists of a reference voltage generation circuit, a detection voltage selection circuit, a comparator and control registers.

The supply voltage is divided by the ladder resistor and input to the detection voltage selection circuit.

The detection voltage selection circuit selects a voltage according to the specified detection voltage, and the comparator compares it with the reference voltage.

When the supply voltage becomes lower than the detection voltage, a voltage detection reset signal (an interrupt can be also selected) is generated.

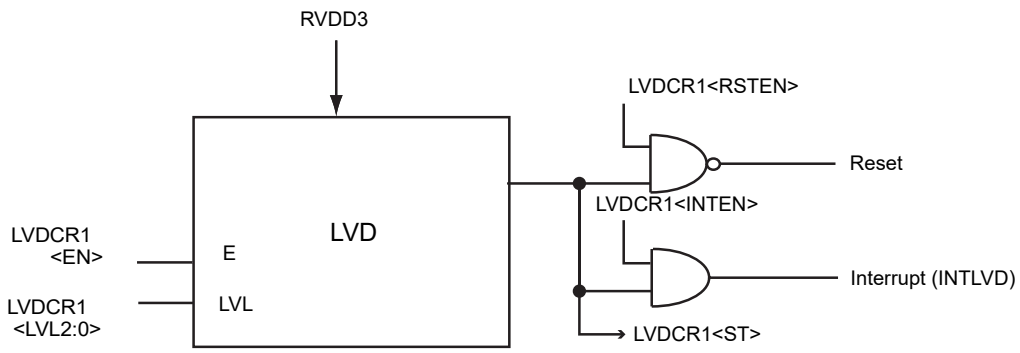


Figure 24-1 LVD Block Diagram

24.2 Registers

For the base address, refer to "Address lists of peripheral functions" of Chapter "Memory Map".

24.2.1 Register list

Register name		Address (Base+)
LVD detection control 1	LVDCR1	0x0004

24.2.2 LVDCR1 (LVD detection 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	ST	RSTEN	INTEN	–	LVL			EN
after reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31 - 8	–	R	Read as "0"
7	ST	R	LVL voltage detection status 0: Power-supply voltage is the same as detection voltage or higher. 1: Power-supply voltage is the same as detection voltage or lower.
6	RSTEN	R/W	Controls RESET output 0: Disable 1: Enable
5	INTEN	R/W	Controls INTLVD output 0: Disable 1: Enable
4	–	R	Read as "0"
3 - 1	LVL[2:0]	R/W	3V Power supply detection voltage 1 000: 2.8 ± 0.1V 001: 2.85 ± 0.1V 010: 2.9 ± 0.1V 011: 2.95 ± 0.1V 100: 3.0 ± 0.1V 101: 3.05 ± 0.1V 110: 3.1 ± 0.1V 111: 3.15 ± 0.1V
0	EN	R/W	Voltage detection operation 0: Disable 1: Enable

Note 1: LVDCR1 is initialized by the terminal reset.

Note 2: There is not hysteresis between the detection of LVD and release voltage. Chattering may occur during detection, depending on the slope of power supply.

24.3 Operation

24.3.1 Selecting detection voltage and enabling voltage detection operation

Selecting detection voltage, enabling detection operation, selecting output condition and enabling output are configured by setting the LVDCR1.

LVDCR1 is initialized by a terminal reset.

Voltage detection is enabled when voltage to be detected is selected by setting the register LVDCR1<LVL [2:0]> and "1" is set to the LVDCR1<EN>.

Note: Before entering into STOP2 mode, LVD operation should be disable.

24.3.2 Detecting

Reset or an interrupt(INTLVD) occurs if the supply voltage falls under the set detection voltage level.

Reset occurs when "1" is set to the LVDCR1<RSTEN> and the supply voltage falls under the set detection voltage. An interrupt occurs when "1" is set to the LVDCR1<INTEN> and the supply voltage falls under the set detection voltage.

It needs approximately 100 μs to detect voltage reduction and generate reset or an interrupt. If the period that the supply voltage falls under the detected voltage is short, reset or an interrupt may not occur.

25. 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 INTWDTx interrupt or reset.

Note: INTWDTx interrupt is a factor of the non-maskable interrupts (NMI).

25.1 Configuration

Figure 25-1shows the block diagram of the watchdog timer.

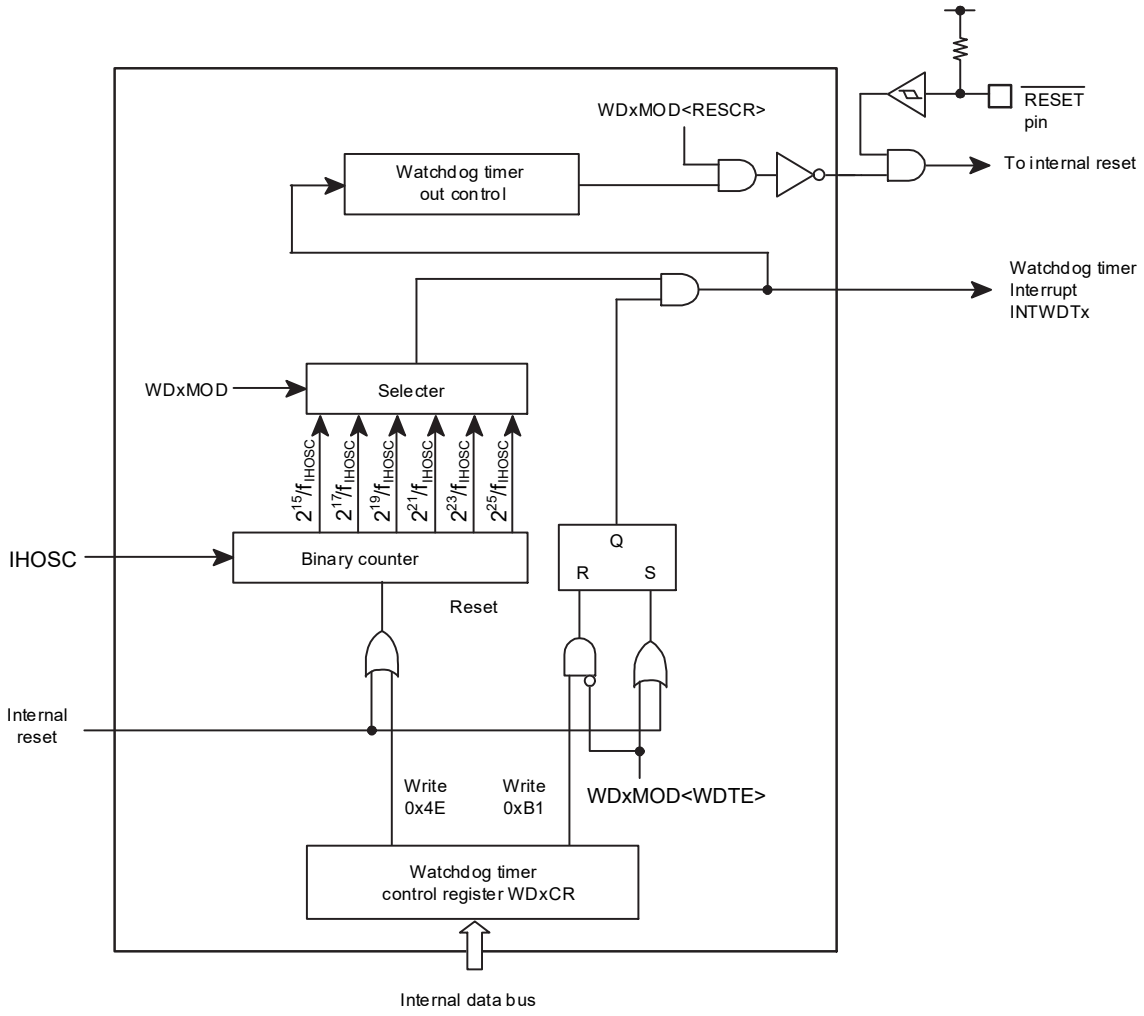


Figure 25-1 Block Diagram of the Watchdog Timer

25.2 Register

25.2.1 Register List

The table below shows control registers and their addresses.

For detail of the base address, refer to "Address lists of peripheral functions" of "Memory Map" chapter.

Peripheral function: WDT

Register name		Address(Base+)
Watchdog Timer Mode Register	WDxMOD	0x0000
Watchdog Timer Control Register	WDxCR	0x0004
Watchdog Flag register	WDxFLG	0x0008

25.2.2 WDxMOD(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 000: $2^{15}/f_{IHOSC}$ 100: $2^{23}/f_{IHOSC}$ 001: $2^{17}/f_{IHOSC}$ 101: $2^{25}/f_{IHOSC}$ 010: $2^{19}/f_{IHOSC}$ 110:Setting prohibited. 011: $2^{21}/f_{IHOSC}$ 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: INTWDTx interrupt request generates. (Note) 1: Reset
0	–	R/W	Write 0.

Note: INTWDTx interrupt is a factor of the non-maskable interrupts (NMI).

25.2.3 WDxCR (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	0	0	0	0	0	0	0	0

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

25.2.4 WDxFLG (Watchdog 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	-	-	-	-	-	-	-	FLG
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	–	R	Read as 0.
0	FLG	R	Flag for writing to registers 0: Enable 1: Disable When writing to WDxMOD or WDxCR, confirm "0" of this bit.

25.3 Description of Operation

25.3.1 Basci Operation

The Watchdog timer is consists of the binary counters that work using the count clock (IHOSC) as an input. Detecting time can be selected between 2^{15}, 2^{17}, 2^{19} , 2^{21}, 2^{23} and 2^{25} by the WDxMOD<WDTP[2:0]>.

The detecting time as specified is elapsed, the watchdog timer interrupt (INTWDTx) generates.

To detect malfunctions (runaways) of the CPU caused by noise, stopping system clock or other disturbances, the binary counter of the watchdog timer should be cleared by software instruction before INTWDTx interrupt generates. If the binary counter is not cleared, the non-maskable interrupt generates by INTWDTx. 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.

25.3.2 Operation Mode and Status

The watchdog timer begins operation immediately after a reset is released. 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 operation modes, the watchdog timer should be disabled.

In IDLE mode, its operation depends on the WDxMOD <I2WDT> setting.

- STOP1 mode
- STOP2 mode

Also,the binary counter is automatically stopped during debug mode.

25.3.3 Operation when malfunction(runway) is detected.

25.3.3.1 INTWDTx interrupt generation

In the Figure 25-2 shows the case that INTWDTx interrupt generates (WDxMOD<RESCR>="0").

When an overflow of the binary counter occurs, INTWDTx 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 INTWDTx, CGNMIFLG<NMIFLG0> is set

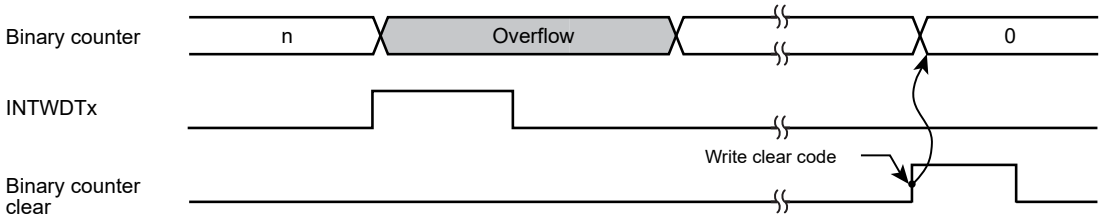


Figure 25-2 INTWDTx interrupt generaiton

25.3.3.2 Internal Resetgeneration

Figure 25-3 shows the internal reset generation (WDxMOD<RESCR>="1").

MCU is reset by the overflow of the binary counter. In this case, reset status continues for 32 states.

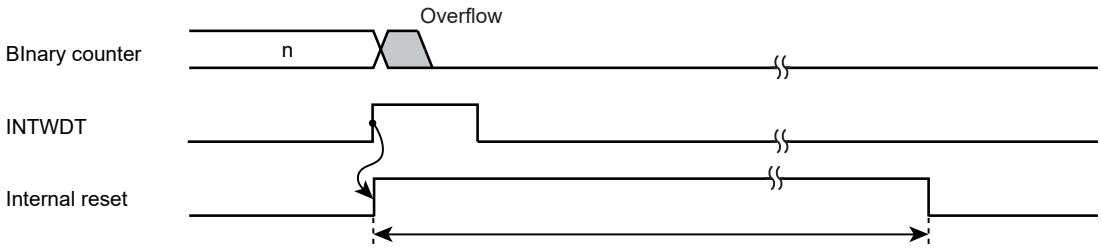


Figure 25-3 Internal reset genaration

25.4 Control of the watchdog timer

25.4.1 Register access

When writing to WDxMOD and WDxCR, confirm whether WDxFLG<FLG> is "0".

However, the consecutive writing to WDxMOD and WDxCR is available for the case of the disable control. Confirming WDxFLG<FLG> is required only before writing to WDxMOD.

25.4.2 Disable control

By writing the disable code (0xB1) to WDxCR after setting WDxMOD<WDTE> to "0", the watchdog timercan be disabled and the binary counter can be cleared

25.4.3 Enable control

Set WDxMOD<WDTE> to "1"

25.4.4 Watchdog timer clearing control

Writing the clear code (0x4E) to WDxCR clears the binary counter and it restarts counting.

25.4.5 Detection time of watchdog timer

Set WDxMOD<WDTP[2:0]> depend on the detection time.

For example, in the case that $2^{21}/f_{IHOSC}$ is used, set "011" to WDxMOD<WDTP[2:0]>.

26. Flash Memory (FLASH)

This section describes the hardware configuration and operation of Flash memory. In this section, "1-word" means 32 bits.

26.1 Features

26.1.1 Memory Size and Configuration

Table 26-1,Table 26-2 and show a built-in memory size and configuration of TMPM46BF10FG.

Table 26-1 Memory size and configuration

Product	Memory size (KB)	Area information		Block information		Page information		Program time (s) (Note)	Erase time (ms) (Note)			
		Size (KB)	# of memory	Size (KB)	# of memory	Size (Byte)	# of memory		1 page	1 block	1 area	Chip
TMPM46BF10FG	1024	512	2	32	32	4096	256	10.7	115	920	115	230

Note: Above time is assumed as an initial value of each register after reset. Data transfer time is not included. Flash programming time per chip varied depending on users' programming method.

Table 26-2 Block Configuration

Area No.	Block-No.	Address (Single chip mode)	Address (Single boot mode) (Single chip mode(mirror))	Size (KByte)	Number of pages
0	0	0x0000_0000 to 0x0000_7FFF	0x5E00_0000 to 0x5E00_7FFF	32	8
	1	0x0000_8000 to 0x0000_FFFF	0x5E00_8000 to 0x5E00_FFFF	32	8
	2	0x0001_0000 to 0x0001_7FFF	0x5E01_0000 to 0x5E01_7FFF	32	8
	3	0x0001_8000 to 0x0001_FFFF	0x5E01_8000 to 0x5E01_FFFF	32	8
	4	0x0002_0000 to 0x0002_7FFF	0x5E02_0000 to 0x5E02_7FFF	32	8
	5	0x0002_8000 to 0x0002_FFFF	0x5E02_8000 to 0x5E02_FFFF	32	8
	6	0x0003_0000 to 0x0003_7FFF	0x5E03_0000 to 0x5E03_7FFF	32	8
	7	0x0003_8000 to 0x0003_FFFF	0x5E03_8000 to 0x5E03_FFFF	32	8
	8	0x0004_0000 to 0x0004_7FFF	0x5E04_0000 to 0x5E04_7FFF	32	8
	9	0x0004_8000 to 0x0004_FFFF	0x5E04_8000 to 0x5E04_FFFF	32	8
	10	0x0005_0000 to 0x0005_7FFF	0x5E05_0000 to 0x5E05_7FFF	32	8
	11	0x0005_8000 to 0x0005_FFFF	0x5E05_8000 to 0x5E05_FFFF	32	8
	12	0x0006_0000 to 0x0006_7FFF	0x5E06_0000 to 0x5E06_7FFF	32	8
	13	0x0006_8000 to 0x0006_FFFF	0x5E06_8000 to 0x5E06_FFFF	32	8
	14	0x0007_0000 to 0x0007_7FFF	0x5E07_0000 to 0x5E07_7FFF	32	8
	15	0x0007_8000 to 0x0007_FFFF	0x5E07_8000 to 0x5E07_FFFF	32	8
1	16	0x0008_0000 to 0x0008_7FFF	0x5E08_0000 to 0x5E08_7FFF	32	8
	17	0x0008_8000 to 0x0008_FFFF	0x5E08_8000 to 0x5E08_FFFF	32	8
	18	0x0009_0000 to 0x0009_7FFF	0x5E09_0000 to 0x5E09_7FFF	32	8
	19	0x0009_8000 to 0x0009_FFFF	0x5E09_8000 to 0x5E09_FFFF	32	8
	20	0x000A_0000 to 0x000A_7FFF	0x5E0A_0000 to 0x5E0A_7FFF	32	8
	21	0x000A_8000 to 0x000A_FFFF	0x5E0A_8000 to 0x5E0A_FFFF	32	8
	22	0x000B_0000 to 0x000B_7FFF	0x5E0B_0000 to 0x5E0B_7FFF	32	8
	23	0x000B_8000 to 0x000B_FFFF	0x5E0B_8000 to 0x5E0B_FFFF	32	8
	24	0x000C_0000 to 0x000C_7FFF	0x5E0C_0000 to 0x5E0C_7FFF	32	8
	25	0x000C_8000 to 0x000C_FFFF	0x5E0C_8000 to 0x5E0C_FFFF	32	8
	26	0x000D_0000 to 0x000D_7FFF	0x5E0D_0000 to 0x5E0D_7FFF	32	8
	27	0x000D_8000 to 0x000D_FFFF	0x5E0D_8000 to 0x5E0D_FFFF	32	8
	28	0x000E_0000 to 0x000E_7FFF	0x5E0E_0000 to 0x5E0E_7FFF	32	8
	29	0x000E_8000 to 0x000E_FFFF	0x5E0E_8000 to 0x5E0E_FFFF	32	8
	30	0x000F_0000 to 0x000F_7FFF	0x5E0F_0000 to 0x5E0F_7FFF	32	8
	31	0x000F_8000 to 0x000F_FFFF	0x5E0F_8000 to 0x5E0F_FFFF	32	8

Flash memory configuration units ares described as "block" and "page".

- Page

  Used in erase function and protect function.

  One page is fixed to 4096 bytes.
- Block

  Used in erase function and protect function.

  One block is fixed to 32K bytes.
- Area

Used in erase function.

There are 512K byte areas.

Write operation is performed in the units of 16 bytes (4 bytes x 4 times). The write time per 16 bytes is 163µs (Typ.).

Erase is performed per block or performed on entire flash memory. Erase time varies on commands. If auto block command is used, the erase time will be 920 ms. per block (Typ.). If the auto chip erase command is used to erase entire area, the time will be 230 ms. (Typ.). If the other commands is used, the time will be 115 ms (Typ.).

In addition, the protect function is set by a page from page 0 to 7. The other blocks are set by a block. When the protection function is erased, the entire Flash memory is erased once. For detail of the protect function, refer to "26.1.5 Protect/Security Function".

26.1.2 Function

Flash memory built-in this device is generally compliant with the JEDEC standards except for some specific functions. Therefore, if a user is currently using a Flash memory as an external memory, it is easy to implement the functions into this device. Furthermore, to provide easy write or erase operation, this product contains a dedicated circuit to perform write or chip erase automatically.

JEDEC compliant functions	Modified, added, or deleted functions
・Automatic programming ・Automatic chip erase ・Automatic block erase ・Data polling/toggle bit	<Addition> Auto area erase, auto page erase, auto memory swap <Modified> Block write/erase protect (only software protection is supported) <Deleted> Erase resume - suspend function

26.1.3 Operation Mode

26.1.3.1 Mode Description

This device provides the single chip mode and single boot mode. The single chip mode contains the normal mode and user boot mode. Figure 26-1 shows the mode transition.

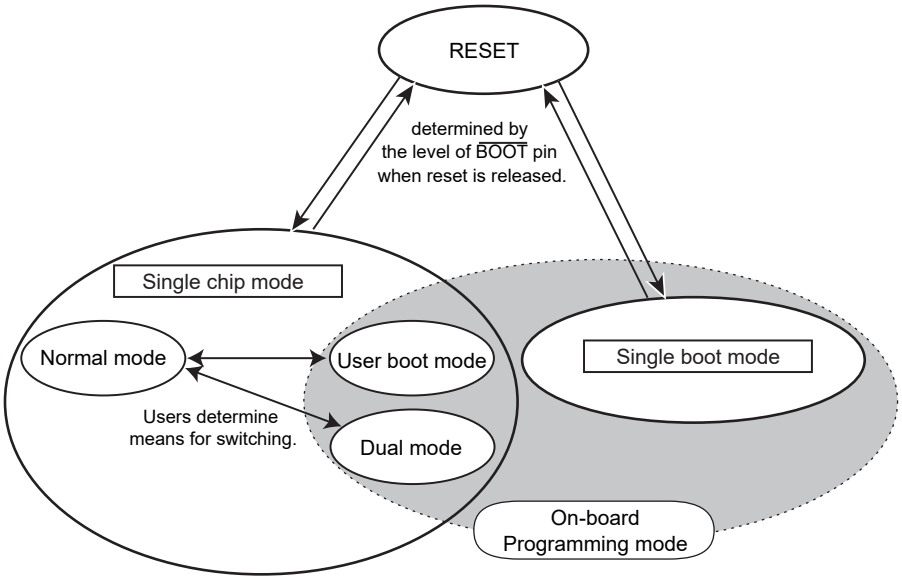


Figure 26-1 Mode transition

(1) Single chip mode

The single chip mode is a mode where the device can boot-up from Flash memory after reset. The mode contains 3 sub-modes in below.

- Normal mode

  The mode where user application program is executed.

- User boot mode

  User boot mode is the mode in which Flash memory externally located in the Flash memory can be re-programmed.

  For details of Flash reprogramming, refer to "26.4 Reprogramming in the User Boot Mode". Users can switch the normal mode to user boot mode freely. For example, a user can set if PA0 of port A is "1", the mode is the normal mode. If PA0 of port A is "0", the mode is the user boot mode.

  The user must prepare a routine program in the application program to determine the switching.

- Dual mode

  Dual mode is the mode in which Flash memory internally located in the Flash memory can be re-programmed.

  For details of Flash reprogramming, refer to "26.5 How to Reprogramming using Dual Mode".

Switching each mode is determined freely by users. For example, when PA0 of port A is set to "1", the mode is normal mode; when PA0 of port A is set to "0", the mode is user boot mode. Users should be provide the routine in the part of application program to switch the mode.

(2) Single boot mode

Single boot mode is the mode in which Flash memory externally located in the Flash memory can boot-up from the built-in BOOT ROM (Mask ROM) after reset.

Flash memory can be reprogrammed by Flash memory reprogramming program located in outside of Flash memory such as single internal memory. For details about reprogramming Flash memory, refer to "26.3 How to Reprogram Flash in Single Boot Mode".

The BOOT ROM contains the algorithm that can rewrite Flash memory via serial port of this device on the users' set. With connecting the serial port to external host, data transfer is performed in above-mentioned protocol and re-programmed Flash memory

(3) On-board programming mode

The user boot mode, dual mode and single boot mode are the modes where Flash memory can be re-programmable on the users' set. These two modes are called "on-board programming mode".

26.1.3.2 Mode Determination

Either the single chip or single boot operation mode can be selected by the level of the $\overline{\text{BOOT}}$ pin when reset is released.

Table 26-3 Operation mode setting

Operation mode	Pin	
	$\overline{\text{RESET}}$	$\overline{\text{BOOT}}$
Single chip mode / dual mode	0 → 1	1
Single boot mode	0 → 1	0

26.1.4 Memory Map

Figure 26-2 and show comparisons of the memory map in the single chip mode and single boot mode. In the single boot mode, built-in Flash memory is mapped to 0x5E00_0000 and subsequent addresses, and the built-in BOOT ROM is mapped to 0x0000_0000 through 0x0000_0FFF.

Flash memory and RAM addresses are shown below.

Product	Flash size	RAM size	Flash address	RAM address
TMPM46BF10FG	1024KB	514KB	0x0000_0000 to 0x000F_FFFF (Single chip mode) 0x5E00_0000 to 0x5E0F_FFFF (Single dhip mode(mirror)) 0x5E00_0000 to 0x5E0F_FFFF (Single boot mode)	0x2000_0000 to 0x2008_07FF

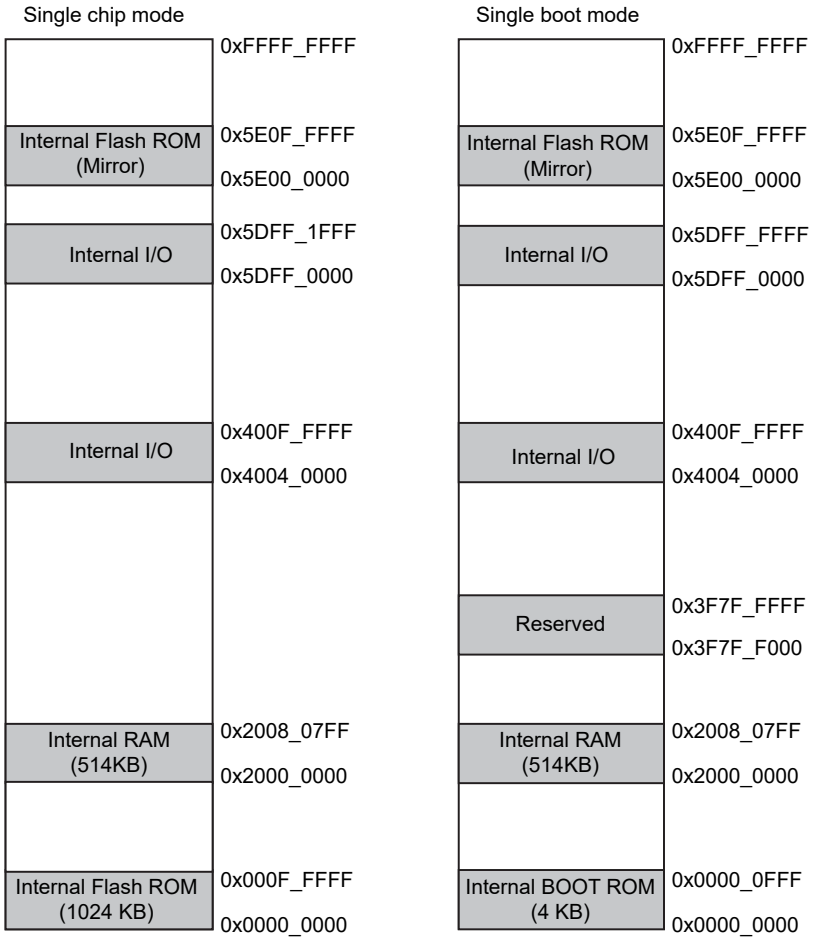


Figure 26-2 Comparison of memory map

26.1.5 Protect/Security Function

This device has the protect and security functions for Flash memory.

1. Protect function

   The write/erase operation can be prohibited per block.

2. Security function

   The read operation from a flash writer can be prohibited.

   Usage restrictions on debug functions

26.1.5.1 Protect Function

Block 0 can use the protect function by page. From Block1 through the last block, these blocks can use the protect function by block.

To enable the protect function, a protect bit corresponding to a block is set to "1" using the protect bit program command. If a protect bit is set to "0" using the protect bit erase command, a block protect can be cancelled. The protect bit can be monitored in each FCPSR register.

For detail of programming/erasing of protect bits, refer to Chapter "26.2.7 Command Description".

26.1.5.2 Security Function

Table 26-4 shows operations when the security function is enabled.

Table 26-4 Operations when the security function is enabled.

Item	Description
Read flash memory	CPU can read flash memory.
Debug port	Serial wire or trace communication is disabled.
Command execution to Flash memory	Command write to flash memory is not accepted. If a user tries to erase a protect bit, chip erase is executed and all protect bits are erased.

The security function is enabled under the following conditions:

1. FCSECBIT<SECBIT> is set to "1".
2. All protect bits (all bits in each FCPSR register) are set to "1".

FCSECBIT<SECBIT> is set to "1" by the cold reset. Rewriting of FCSECBIT<SECBIT> is described in below.

Note: Use a 32-bit transfer instruction to the following writing operations, item1 and 2.

1. Write the specified code (0xa74a9d23) to FCSECBIT
2. Write data within 16 clocks after the operation of item 1.

Note: When FCSECBIT<SECBIT>="0" is executed, a protect bit is masked to "0" and it becomes write-enabled. Note that the protect bit is not changed. (Security and protection conditions have been being released.)

Table 26-5 SECBIT and protection bit

	<SECBIT>=1	<SECBIT>=0
protect bits <All>=1	Secured condition	Connectable to the tools. Read/Write enabled.
protect bits <any>=1	Connectable to the tools. Block-write is protected.	Connectable to the tools. Read/Write enabled.
protect bits <All>=0	Connectable to the tools. Read/Write enabled.	Connectable to the tools. Read/Write enabled.

26.1.6 Memory Swap Function

26.1.6.1 Outline

If the power is off in the middle of Flash memory reprogramming, programming may not be executed. Suppose you came upon a situation in which the power becomes OFF after a program is erased. In this case, you cannot finish the programming. To avoid such case, use this memory swap function to save your program.

26.1.6.2 Operation Description

A swap region is determined by the region starting from address 0 and the next region. A swap size is determined by FCSWPSR<SIZE>. To change this size, set "1" to FCSWPSR<SIZE> bit using automatic memory swap command.

In order to perform memory swap, set "1" to FCSWPSR[0] using automatic memory swap command. To release the swap condition, set "1" to FCSWPSR[1] using automatic memory swap command. Swap condition can be checked by FCSWPSR<SWP>.

For details of automatic memory swap command, "26.2.7 Command Description".

26.1.6.3 Usage of Memory Swap

This section describes the basic flow of memory swap procedure. For an example of memory swap, refer to "26.6 How to Reprogram Flash using User Boot Mode".

1. Release the security function if the security function is enabled.

   For details of security releasing, refer to "26.1.5.2 Security Function".

   If the security function is not released, Flash memory is erased according to the command execution.

2. If the security function is enabled, erase the protect bit.

   For details of erasing protect bit, refer to "26.1.5.1 Protect Function".

   If the protect bit is not erased, command execution is not performed according to the procedure.

3. Confirm if the next region next to the region starting from address 0 is blank. (Hereafter, a region starting from address 0 is called Page0; the next region is called Page1) If this region is not blank, erase the data.

   Page0:Old original data

   Page1:Blank

4. Write the original data starting from address 0 to the next region (Both regions become the same data.)

   Page0:Old original data

   Page1:Copy data (Old original data)

5. Perform memory swap

   Page0:Copy data (Old original data)

   Page1:Old original data

6. Erase old original data to be blank

   Page0:Copy data (Old original data)

   Page1:Blank

7. Write new data to the blank region

   Page0:Copy data (Old original data)

   Page1:New original data

8. Release swap conditions

Page0:New original data

Page1:Copy data (old original data)

9. Execute automatic protect bit erase command

10. Options if required
    - Erase copy data (old original data).
    - Reprogramming Flash memory data except swap regions
    - Validate the protect function
    - Validate the security function

Procedure		3	4	5	6	7	8
On-chip RAM		Erase routine	Programming routine	Swap routine	Erase routine	Programming routine	Swap routine
Flash memory	Page0	Old original	Old original	Copy of old original	Copy of old original	Copy of old original	New original
	Page1	Blank	Copy of old original	Old original	Blank	New original	Copy of old original

Erase routine:	A program is to erase Flash memory.
Programming routine:	A program is to program Flash memory.
Swap routine:	A program is to swap Flash memory.

26.1.7 Register

26.1.7.1 Register List

The following table shows control registers and addresses.

For details of base address, refer to "Address lists of peripheral functions" of "Memory Map" Chapter.

Register name		Address (Base+)
Security bit register	FCSECBIT	0x0010
Protect status register 0	FCPSR0	0x0020
Protect status register 1	FCPSR1	0x0030
Status register	FCSR	0x0100
Swap status register	FCSWPSR	0x0104
Area selection register	FCAREASEL	0x0140
Control register	FCCR	0x0148
Status clear register	FCSTSCLR	0x014C
WCLK setting register	FCWCLKCR	0x0150
Count setting register for program	FCPROGCR	0x0154
Count setting register for erase	FCERASECR	0x0158

26.1.7.2 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: Security function setting is disabled. 1: Security function setting is enabled.

Note: This register is initialized by system reset and the Release trigger from the STOP2 mode.

26.1.7.3 FCPSR0 (Protect Status Register 0)

	31	30	29	28	27	26	25	24
bit symbol	BLK15	BLK14	BLK13	BLK12	BLK11	BLK10	BLK9	BLK8
After reset	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)
	23	22	21	20	19	18	17	16
bit symbol	BLK7	BLK6	BLK5	BLK4	BLK3	BLK2	BLK1	-
After reset	(Note1)	(Note1)	(Note1)	(Note1)	(Note)	(Note1)	(Note1)	0
	15	14	13	12	11	10	9	8
bit symbol	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
After reset	(Note1)	(Note1)	(Note1)	(Note1)	(Note)	(Note1)	(Note1)	(Note1)
	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
31-17	BLK15 to BLK1	R	Protection status of Block1 to 15 0: No protection 1: Protected Protect bit values correspond to protect status of each block. If corresponding bit indicates "1", corresponding block is in the protection status. A block in the protection status cannot be re-programmable.
16	–	R	Read as "0".
15-8	PG7 to PG0	R	Protection status of Page0 to 7 0: No protection 1: Protected Protect bit values correspond to protect status of each block. If corresponding bit indicates "1", corresponding block is in the protection status. A block in the protection status cannot be re-programmable.
7-1	–	R	Read as "0".
0	RDY_BSY	R	Ready/Busy flag when automatic program or automatic chip erase command is executing. (Note2) 0: In automatic operation 1:Automatic operation ends This bit is a function bit to monitor flash memory from CPU. While flash memory is in auto operation, this bit outputs "0" to indicate that flash memory is busy. Once auto operation is finished, this bit becomes ready state and outputs "1". Then next command is accepted. If a result of auto operation is failed, this bit outputs "0" continuously. The bit returns to "1" by aborting auto operation. For details, refer to "26.2.5 Auto Operation Abort".

Note 1: Depend on the Protection status

Note 2: Make sure that Flash memory is ready before commands are issued. If a command is issued during busy, not only the command is not sent but also subsequent commands may not be accepted. In that case, use FCCR<WEABORT> to release the busy state. For details, refer to "26.2.5 Auto Operation Abort".

26.1.7.4 FCPSR1 (Protect Status Register 1)

	31	30	29	28	27	26	25	24
bit symbol	BLK31	BLK30	BLK29	BLK28	BLK27	BLK26	BLK25	BLK24
After reset	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)
	23	22	21	20	19	18	17	16
bit symbol	BLK23	BLK22	BLK21	BLK20	BLK19	BLK18	BLK17	BLK16
After reset	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)	(Note1)
	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	1

Bit	Bit Symbol	Type	Function
31-17	BLK31 to BLK16	R	Protection status of Block16 to 31 0: No protection 1: Protected Protect bit values correspond to protect status of each block. If corresponding bit indicates "1", corresponding block is in the protection status. A block in the protection status is not re-programmable.
16-1	–	R	Read as "0".
0	–	R	Read as "1".

Note 1: Depend on the Protection status

26.1.7.5 FCSR (Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	WEABORT
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	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-25	–	R	Read as "0".
24	WEABORT	R	Once the auto operation is aborted by FCCR<WEABORT>, "1" is set to this bit. For details, refer to "26.2.5 Auto Operation Abort".
23-0	–	R	Read as "0".

26.1.7.6 FCSWPSR (Swap 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	-	-	-	-	-	SIZE		
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	FLG						SWP	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-11	–	R	Read as "0".
10-8	SIZE[2:0]	R	Swap size 000: 4K bytes 001: 8K bytes 010: 16K bytes 011: 32K bytes 100: 512K byte Other than the above: Setting is prohibited.
7-2	FLG	R	Use as a flag bit for software management(refer to the "26.6 How to Reprogram Flash using User Boot Mode" for a example.)
1-0	SWP[1:0]	R	Swap state 11: Release the swap 10: Setting is prohibited. 01: In swapping 00: Release the swap (Initial state)

Note:This register is initialized by auto protect bit erase command.

26.1.7.7 FCAREASEL (Area 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	-	AREA1			-	AREA0		
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	–	R	Read as "0".
6-4	AREA1	R/W	Specifies an "area" in the Flash memory that is targeted by Flash memory operation command. 111: Selects area 1. Other than the above: non-selective area 1.
3	–	R	Read as "0".
2-0	AREA0	R/W	Specifies an "area" in the Flash memory that is targeted by Flash memory operation command. 111: Selects area 0. Other than the above: non-selective area 0.

Note 1: If area selection bit (<AREAn>) is written, wait for written data to become read by polling.

Note 2: If Flash memory operation command is executed when area selection bit is other than "111" (0x7), the command execution is cancelled.

Note 3: When auto chip erase command is executed, set "111" (0x7) to all area selection bit.

26.1.7.8 FCCR (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	-	-	-	-	-	WEABORT		
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	–	R	Read as "0".
2-0	WEABORT	R/W	Abort of auto operation command 111: Abort Read as the setting value. For details, refer to "26.2.5 Auto Operation Abort".

26.1.7.9 FCSTSCLR (Status 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	-	-	-	-	-	WEABORT		
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	–	R	Read as "0".
2-0	WEABORT	R/W	Clears FCSR<WEABORT> to "0". 111: Clear Read as the setting value. For details, refer to "26.2.5 Auto Operation Abort".

26.1.7.10 FCWCLKCR (WCLK 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	-	-	-	DIV				
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	–	R	Read as "0".
4-0	DIV	R/W	Frequency division ratio to change the clock (WCLK: fc/(DIV+1)) in automatic operation to 8 to 12MHz 00000: Divide-by-1 00001: Divide-by-2 : 11110: Divide-by-31 11111: Divide-by-32

Note 1: When Flash operation is performed with changing gear, re-set WCLK to be within 8 to 12MHz according to the operation frequency (fc). Table 26-6 shows the setting values in each range of operation frequency (fc).

26.1.7.11 FCPROGCR (Count Setting Register for Program)

	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	-	-	-	-	-	-	CNT	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	–	R	Read as "0".
1-0	CNT	R/W	Set the number of counts that makes a programming time (CNT/WCLK) by automatic program execution command be within the range of 20 to 40 μs. 00: Number of counts 250 01: Number of counts 300 Other than the above: Number of counts 350

Note 1: When WCLKCR<DIV> is re-set, re-set a programming time by automatic program execution command within the range of 20 to 40μs if required. Table 26-6 shows the setting values in each range of operation frequency (fc).

26.1.7.12 FCERASECR (Count Setting Register for Erase)

	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	-	-	-	-	CNT			
After reset	0	0	0	0	0	1	1	0

Bit	Bit Symbol	Type	Function
31-4	–	R	Read as "0".
3-0	CNT	R/W	Number of counts until erase time (CNT/WCLK) will be 100 to 130ms using each auto erase command 0000: Number of counts 850000 0001: Number of counts 900000 0010: Number of counts 950000 0011: Number of counts 1000000 0100: Number of counts 1050000 0101: Number of counts 1100000 0110: Number of counts 1150000 0111: Number of counts 1200000 1000: Number of counts 1250000 1001: Number of counts 1300000 1010: Number of counts 1350000 Other than the above: Number of counts 1400000

Note 1: If WCLKCR<DIV> is re-set, set a erase time within 100 to130 ms using auto erase command if required. Table 26-6 shows the setting values in each range of operation frequency (fc).

Table 26-6 Setting values in each operation frequency (fc)

Operation frequency (fc) range	FCWCLKCR<DIV>	FCPROGCR<CNT>	FCERASECR<CNT>
115.0 < fc ≤ 120.0 MHz	01011	01	0110
107.8 < fc ≤ 115.0 MHz	01010	01	0110
102.8 < fc ≤ 107.8 MHz	01000	10	1010
96.9 < fc ≤ 102.8 MHz	01001	01	0110
91.9 < fc ≤ 96.9 MHz	00111	10	1010
86.3 < fc ≤ 91.9 MHz	01000	01	0110
81.3 < fc ≤ 86.3 MHz	01001	00	0010
77.2 < fc ≤ 81.3 MHz	00111	01	0110
72.2 < fc ≤ 77.2 MHz	01000	00	0010
68.1 < fc ≤ 72.2 MHz	00110	01	0110
62.9 < fc ≤ 68.1 MHz	00111	00	0010
58.9 < fc ≤ 62.9 MHz	00101	01	0110
53.8 < fc ≤ 58.9 MHz	00110	00	0010
48.5 < fc ≤ 53.8 MHz	00100	01	0110
44.0 < fc ≤ 48.5 MHz	00011	10	1010
40.6 < fc ≤ 44.0 MHz	00100	00	0010
37.5 < fc ≤ 40.6 MHz	00011	01	0110
34.1 < fc ≤ 37.5 MHz	00010	10	1010
31.5 < fc ≤ 34.1 MHz	00011	00	0010
27.3 < fc ≤ 31.5 MHz	00010	01	0110
24.1 < fc ≤ 27.3 MHz	00010	00	0010
21.7 < fc ≤ 24.1 MHz	00001	10	1010
18.2 < fc ≤ 21.7 MHz	00001	01	0110
15.6 < fc ≤ 18.2 MHz	00001	00	0110
13.4 < fc ≤ 15.6 MHz	00001	00	0000
11.9 < fc ≤ 13.4 MHz	00000	10	1011
10.8 < fc ≤ 11.9 MHz	00000	10	1010
9.1 < fc ≤ 10.8 MHz	00000	01	0110
8.0 < fc ≤ 9.1 MHz	00000	00	0010

26.2 Detail of Flash Memory

In on-board programming, the CPU executes commands for reprogramming or erasing Flash memory. This reprogramming/erase control program should be prepared by the user beforehand.

Furthermore, for example while a program is executing on area 0, the other area (such as area 1) of Flash memory, on which instructions are not executed, can be written/erased. (The opposite case is possible.)

26.2.1 Function

Flash memory is generally compliant with the JEDEC standards except for some specific functions. However, a method of address designation of operation command is different from standard commands.

If write/erase operation is executed, commands are input to flash memory using 32-bit (1-word) store instruction command. After command input, write or erase operation is automatically executed in inside.

Table 26-7 Flash memory function

Main function	Description
Auto program	Writes data in units of 4 word (16 bytes) automatically.
Auto chip erase	Erases the entire area of Flash memory automatically.
Auto area erase	Erases a selected area automatically.
Auto block erase	Erases a selected block automatically.
Auto page erase	Erases a selected page automatically.
Write/erase protect	The write or erase operation can be prohibited.
Auto memory swap	Automatically performs swap/swap release/swap size designation.

26.2.2 Operation Mode of Flash Memory

Flash memory provides main two types of operation modes:

- The mode to read memory data (Read mode)
- The mode to erase or rewrite memory data automatically (Automatic operation mode)

After power-on, after rest or after automatic operation mode is finished normally, Flash memory becomes read mode. Instruction stored in Flash memory or data read is executed in the read mode.

If commands is input during the read mode, the operation mode becomes the automatic operation. If the command process is normally finished, the operation mode returns to the read mode except the ID-Read command. For details of command execution, refer to "26.2.4 How to Execute Command".During the automatic operation, data read and instruction execution stored in Flash memory cannot be performed.

If command process is abnormally finished then the operation mode should forcibly return to read mode. In this case, use FCCR<WEABORT>. For details, refer to "26.2.5 Auto Operation Abort".

26.2.3 Hardware Reset

For details of the reset operation, refer to "Reset" Chapter.

26.2.4 How to Execute Command

The command execution is performed by writing command sequences to Flash memory with a store instruction. Flash memory executes each automatic operation command according to the combination of input addresses and data. For detail of the command execution, refer to "26.2.7 Command Description".

An execution of store instruction to the Flash memory is called "bus write cycle". Each command consists of some bus write cycles. In Flash memory, when address and data of bus write cycle are performed in the specified order, the automatic command operation is performed. When the cycle is performed in non-specified order, Flash memory stops command execution and returns to the read mode.

If you cancel the command during the command sequence or input a different command sequence, execute the read command or read/reset command. Then Flash memory stops command execution and returns to the read mode. The read command and read/reset command are called "software reset".

When write command sequence ends, the automatic operation starts and FCPSR0<RDY_BSY> is set to "0". When the automatic operation normally ends, FCPSR0<RDY_BSY> = "1" is set and Flash memory returns to the read mode.

New command sequences are not accepted during the automatic operation. If you want to stop the command operation, refer to "26.2.5 Auto Operation Abort". In case that the automatic operation abnormally ends (FCPSR0<RDY_BSY> remains "0"), Flash memory remains locked and will not return to the read mode. To returns to the read mode, the command should be aborted. If the hardware reset stops the command operation, commands are not normally executed.

Notes on the command execution

1. The following operation are prohibited while auto operation.
   - Shut down
   - All interrupt request

2. To recognize command, command sequencer need to be in the read mode before command starting. Confirm if FCPSR0<RDY_BSY> ="1" is set prior to the first bus write cycle of each command. It is recommended that the read command is consecutively executed.

3. The following command sequences must be executed on the on-chip RAM.
   - Automatic chip erase command
   - ID-Read command
   - Automatic protect bit program command
   - Automatic protect bit erase command
   - Automatic memory swap command

   All command sequences except above command sequences are re programmable using dual mode Thus, A program can be executed on the other area such as area 0 in which is not a Flash memory area in area 1 used for programming/erasing. (The opposite case is also possible.)

4. Set an area selection bit (write "111" (0x7) to <AREAn>) in FCAREASEL register before each command is executed.

   In addition, if the following commands are executed, write "111" (0x7) to all area selection bit.
   - Automatic chip erase command
   - ID-Read command
   - Automatic protect bit program command
   - Automatic protect bit erase command
   - Automatic memory swap command

5. Set each bus write cycle using consecutive 1-word (32-bit) data transfer instruction.

6. If an access is performed to the target Flash memory in each command sequence, an bus fault occurs.

7. When issuing commands, wrong addresses or data is written, make sure to issue software reset then return to the read mode.

8. Confirmation procedure after each command completion is as follow.
   1) Execute the final bus write cycle.
   2) Wait for FCPSR0<RDY_BSY> to become read "0" (Busy) by polling.
   3) Wait for FCPSR0<RDY_BSY> to become read "1" (Ready) by polling.

9. When a data is read from Flash memory, clear the area selection bit in FCAREASEL register. (Write "000" (0x0) to <AREAn>)

Note: Each command execution is under the internal Flash ROM (Mirror) address area.

26.2.5 Auto Operation Abort

The following are the actions for returning to the read mode when forced abort or abnormal termination occurred in the automatic program/erase.

1. Write "0x7" to FCCR<WEABORT>.
2. Write "0x0" to FCCR<WEABORT>.
3. Wait for FCPSR0<RDY_BSY> to become read "1" (Ready) by polling.
4. Wait for FCSR<WEABORT> to become read "1" by polling.
5. Execute Read/reset command.
6. Write "0x7" to FCSTSCLR<WEABORT>.
7. Write "0x0" to FCSTSCLR<WEABORT>.
8. Wait for FCSR<WEABORT> to become read "0".
9. Execute automatic operation command if necessary.

26.2.6 Completion Notice of Automatic Operation

TMPM46BF10FG provides the interrupt function that detects a completion notice of Flash programming/erase operation.

26.2.6.1 Procedure

The following is a procedure of how to set a completion notice of automatic operation.

For details of interrupt service routines, refer to "Interrupts" in Chapter "Exception".

1. After a programming/erase command is issued to Flash, check the automatic operation state (BUSY state) by FCPSR0<RDY_BSY>. After the confirmation of automatic operation, enable INTFLRDY interrupt.
2. After Flash automatic operation has been complete, INTFLRDY interrupt occurs.
3. In the INTFLRDY interrupt service routine, disable the event of INTFLRDY interrupt.

26.2.7 Command Description

This section explains each command content. For details of specific command sequences, refer to "26.2.8 Command Sequence".

26.2.7.1 Automatic Page Program

(1) Operation Description

The automatic page program writes data in the units of 4-word (16 bytes). When the program writes data to multiple pages, a page command need to be executed in page by page. Writing across pages is not possible.

Writing to Flash memory means that data cell of "1" becomes those of "0". It is not possible to become data cell of "1" from data of "0". To become data cell of "1" from "0", the erase operation is required.

The automatic page program is allowed only once to each page already erased. Either data cell of "1" or "0" cannot be written data twice or more. If rewriting to a page that has already been written once, the automatic page program is needed to be set again after the automatic block erase or automatic chip erase command is executed.

Another command sequence is not accepted during automatic program. Refer to "26.2.5 Auto Operation Abort" to stop this operation. At this time, data writing may not be executed normally. Thus, automatic program must be executed again after erase operation.

Note 1: Page program execution to the same page twice or more without erasing operation may damage the device.

Note 2: Writing to the protected block is not possible.

Note 3: For programming into the Flash memory after entering into Normal mode from Stop2 mode, it is required to confirm that CGRSTFLG<OSCFLF> is read as "1".

(2) How to Set

The 1st to 3rd bus write cycles indicate the automatic page program command.

In the 4th bus write cycle, the first address and data of the page are written. On and after 5th bus cycle, one page data will be written sequentially. Data is written in the units of one-word (32-bit).

If a part of 16 bytes needs to be written, write "0xFFFFFFFF" of 16 bytes as data which is not required to write.

No automatic verify operation is performed internally in the device. So, be sure to read the data programmed to confirm that it has been correctly written.

26.2.7.2 Automatic Chip Erase

(1) Operation Description

The automatic chip erase is executed to the memory cell of all addresses. If protected blocks are contained, these blocks will not be erased and return to the read mode after a command sequence is input.

Another command sequence is not accepted during automatic program. Refer to "26.2.5 Auto Operation Abort" to stop this operation. At this time, data writing may not be executed normally. Thus, automatic program must be executed again after erase operation.

(2) How to Set

The 1st to 6th bus write cycles indicate the automatic chip erase command. After the command sequence is input, the automatic chip erase operation starts.

No automatic verify operation is performed internally in the device. So, be sure to read the data to confirm that it has been correctly erased.

26.2.7.3 Automatic Area Erase

(1) Operation Description

The automatic erase command performs erase operation to the specified area. If the specified area contains protected blocks, erase operation is not executed and returns to the read mode after a command sequence is input.

Another command sequence is not accepted during automatic erase operation. Refer to "26.2.5 Auto Operation Abort" to stop this operation. At this time, data writing may not be executed normally. Thus, automatic program must be executed again after erase operation.

(2) How to Set

The 1st to 5th bus write cycles indicate the automatic block erase command. In the 6th bus write cycle, the block to be erased is specified. After the command sequence is input, the automatic block erase operation starts.

No automatic verify operation is performed internally in the device. So, be sure to read the data to confirm that it has been correctly erased.

26.2.7.4 Automatic Block Erase

(1) Operation Description

The automatic erase command performs erase operation to the specified block. If the specified block contains protected pages or blocks, erase operation is not executed and returns to the read mode after a command sequence is input.

Another command sequence is not accepted during automatic erase operation. Refer to "26.2.5 Auto Operation Abort" to stop this operation. At this time, data writing may not be executed normally. Thus, automatic program must be executed again after erase operation.

(2) How to Set

The 1st to 5th bus write cycles indicate the automatic block erase command. In the 6th bus write cycle, the block to be erased is specified. After the command sequence is input, the automatic block erase operation starts.

No automatic verify operation is performed internally in the device. So, be sure to read the data to confirm that it has been correctly erased.

26.2.7.5 Automatic Page Erase

(1) Operation Description

The automatic page command performs erase operation to the specified page. If the specified page is protected, erase operation is not executed and returns to the read mode after a command sequence is input.

Another command sequence is not accepted during automatic erase operation. Refer to "26.2.5 Auto Operation Abort" to stop this operation. At this time, data writing may not be executed normally. Thus, automatic program must be executed again after erase operation.

(2) How to Set

The 1st to 5th bus write cycles indicate the automatic page erase command sequence. In the 6th bus write cycle, the page to be erased in the bus write cycle is specified. After the command sequence is input, the automatic page erase operation starts.

No automatic verify operation is performed internally in the device. So, be sure to read the data to confirm that it has been correctly erased.

26.2.7.6 Automatic Protect Bit Program

(1) Operation Description

The automatic protect bit program writes "1" to a protect bit at a time. To set "0" to a protect bit, use the automatic protect bit erase command.

For detail of the protect function, refer to "26.1.5 Protect/Security Function".

Another command sequence is not accepted during automatic erase operation. Refer to "26.2.5 Auto Operation Abort" to stop this operation. At this time, data writing may not be executed normally. Thus, automatic program must be executed again after erase operation.

(2) How to Set

The 1st to 3th bus write cycles indicate the automatic protect bit program command sequence. In the 4th bus write cycle, the protect bit to be written is specified. After the command sequence is input, the automatic protect bit program starts. Check whether protect bits in each FCPSR register, are written properly.

26.2.7.7 Automatic Protect Bit Erase

(1) Operation Description

The automatic protect bit erase command operation depends on the security status on the exetutions.

- Non-security status

  Clear all specified protect bits to "0".
- Security status

  Erase all protect bits after all addresses of Flash memory are erased.

For detail of security status, refer to "26.1.5 Protect/Security Function".

Another command sequence is not accepted during automatic erase operation. Refer to "26.2.5 Auto Operation Abort" to stop this operation. At this time, data writing may not be executed normally. Thus, automatic program must be executed again after erase operation.

(2) How to Set

The 1st to 3th bus write cycles indicate the automatic protect bit erase sequence. In the 4th bus write cycle, specify 0x5E000000. After the command sequence is input, the automatic protect bit erase operation starts.

In the non-security status, all protect bits are erased. Check whether the protect bits in each FCPSR register are erased properly.

In the security status, all addresses and all protect of Flash memory bits are erased. Confirm if data and protect bits are erased normally, and then execute the automatic protect bit erase, automatic chip erase or automatic block erase again if necessary.

26.2.7.8 ID-Read

(1) Operation Description

The ID-Read command can read information including Flash memory type and three types of codes such as a maker code, device code and macro code.

(2) How to Set

The 1st to 3rd bus write cycles indicate the ID-Read command sequence. In the 4th bus write cycle, the code to be read is specified. After the 4th bus write cycle, a code can be acquired by read operation in the arbitrary Flash area.

The ID-Read can be executed successively. The 4th bus write cycle and reading ID value can be executed repeatedly.

The ID-Read command does not automatically return to the read mode. To return to the read mode, execute the read/reset command.

26.2.7.9 Read/Reset Command (Software Reset)

(1) Operation Description

The read/reset command is a command to return Flash memory to the read mode.

When the ID-Read command is executed, Flash memory stops at the current status without automatically returning to the read mode. To return to the read mode from this situation, use the read/reset command. It is also used to cancel the command when commands are input to the middle.

(2) How to Set

The 1st bus cycle indicates the read command sequence. After either command sequence is executed, Flash memory returns to the read mode.

26.2.7.10 Automatic Memory Swap

(1) Operation Description

Automatic memory swap is a command to write "1" to each bit of FCSWPSR[10:0] in the units of 1-bit. A bit cannot be set to "0" independently. All bits should be cleared to "0" using automatic protect bit erase command.

Another command sequence is not accepted during automatic memory swap operation. Refer to "26.2.5 Auto Operation Abort" to stop this operation. At this time, this operation may not be executed normally. Thus, automatic memory swap must be executed again.

(2) How to Set

The 1st to 4th bus write cycles indicate the automatic memory swap command sequence. After the command sequence is input, "1" is written to the specified bit of FCSWPSR register.

26.2.8 Command Sequence

26.2.8.1 Command Sequence List

Table 26-8 shows addresses and data of bus write cycle in each command.

All command cycles except the 5th bus cycle of ID-Read command are bus write cycles. A bus write cycle is performed by 32-bit (1-word) data transfer instruction. (Following table shows only lower 8 bits of data.)

For detail of addresses, refer to Table 26-9. Use below values to "command" described in a column of Addr[15:9] in the Table 26-9.

Note1) Use the following values for address bit [19] depending on Flash memory size.
Memory size is 512KB or less: Always set to "0".
Memory size is over 512KB: If bus write to 512KB area or less, the bit is set to "0".
If bus write to over 512KB area, the bit is set to "1".

Note2) When a programing into the Flash memory after shinfting from STOP2 mode to Normal mode, it should be confirmed the CGRSTFLG<OSCFLF> status is as "1".

Table 26-8 Flash memory access by internal CPU

Command sequence	1st bus cycle	2nd bus cycle	3rd bus cycle	4th bus cycle	5th bus cycle	6th bus cycle	7th bus cycle
	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.
	Data	Data	Data	Data	Data	Data	Data
Read/reset	0xXXXXXX	–	–	–	–	–	–
	0xF0	–	–	–	–	–	–
ID-Read	0xXX54XX	0xXXAAXX	0xXX54XX	IA	0xXX	–	–
	0xAA	0x55	0x90	0x00	ID	–	–
Automatic program	0xXX54XX	0xXXAAXX	0xXX54XX	PA	PA	PA	PA
	0xAA	0x55	0xA0	PD0	PD1	PD2	PD3
Automatic chip erase	0xXX54XX	0xXXAAXX	0xXX54XX	0xXX54XX	0xXXAAXX	0xXX54XX	–
	0xAA	0x55	0x80	0xAA	0x55	0x10	–
Automatic area erase	0xXX54XX	0xXXAAXX	0xXX54XX	0xXX54XX	0xXXAAXX	AA	–
	0xAA	0x55	0x80	0xAA	0x55	0x20	–
Automatic block erase	0xXX54XX	0xXXAAXX	0xXX54XX	0xXX54XX	0xXXAAXX	BA	–
	0xAA	0x55	0x80	0xAA	0x55	0x30	–
Automatic page erase	0xXX54XX	0xXXAAXX	0xXX54XX	0xXX54XX	0xXXAAXX	PGA	–
	0xAA	0x55	0x80	0xAA	0x55	0x40	–
Automatic protect bit program	0xXX54XX	0xXXAAXX	0xXX54XX	PBA	–	–	–
	0xAA	0x55	0x9A	0x9A	–	–	–
Automatic protect bit erase	0xXX54XX	0xXXAAXX	0xXX54XX	0x0000XX	–	–	–
	0xAA	0x55	0x6A	0x6A	–	–	–
Automatic memory swap	0xXX54XX	0xXXAAXX	0xXX54XX	MSA	–	–	–
	0xAA	0x55	0x9A	0x9A	–	–	–

Supplementary explanation

- IA: ID Address
- ID: ID Data
- PA: Program Address
- PD: Program Data (32-bit data)

  After the 4th bus cycle, input 16 bits data in the order of the addresses
- AA: Area Address (see Table 26-2)
- BA: Block Address (see Table 26-2)
- PGA: Page Address
- PBA: Protect Bit Address (see Table 26-10)
- MSA: Memory Swap Address (see Table 26-12)

26.2.8.2 Address Bit Configuration in the Bus Cycle

Use Table 26-9 in conjunction with "Table 26-8 Flash memory access by internal CPU".

Set an address according to the normal bus write cycle address configuration in the first bus cycle and later.

Table 26-9 Address bit configuration in the bus write cycle

[Normal command]

Address	Addr [31:20]	Addr [19]	Addr [18:16]	Addr [15:9]	Addr [8:0]
Normal command	Setting of the bus write cycle address for normal command				
	0x5E0	Note	"0" is recommended.	Command	"0" is recommended.

[Read/reset and ID-READ]

Address	Addr [31:20]	Addr [19]	Addr [18:16]	Addr [15:14]	Addr [13:0]
Read /reset	Setting of the 1st bus write cycle address for READ/Reset				
	0x5E0	Fixed to "0".	"0" is recommended.		
ID-READ	IA: ID address (Setting of the 4th bus write cycle address for ID-READ)				
	0x5E0	Fixed to "0".	"0" is recommended.	ID Address (Table 26-11)	"0" is recommended.

[Automatic area erase]

Address	Addr [31:20]	Addr [19]	Addr [18:0]
Area erase	AA: Area Address (Setting of the 6th bus write cycle address for area erase)		
	0x5E0	Note	"0" is recommended.

[Automatic block erase]

Address	Addr [31:20]	Addr [19]	Addr [18:13]	Addr [12:0]
Block erase	BA: Block Address (Setting of the 6th bus write cycle address for block erase)			
	0x5E0	Note	Block address (Table 26-2)	"0" is recommended.

[Automatic page erase]

Address	Addr [31:20]	Addr [19]	Addr [18:12]	Addr [11:0]
Page erase	PGA: Page Address (Setting of the 6th bus write cycle address for page erase)			
	0x5E0	Note	Page address	"0" is recommended.

[Automatic program]

Address	Addr [31:20]	Addr [19]	Addr [18:3]	Addr [2:0]
Program	PA: Program page Address (Setting of the 4th bus write cycle address for page program)			
	0x5E0	Note	Page Address	"0" is recommended.

[Protect bit program and memory swap]

Address	Addr [31:20]	Addr [19]	Addr [18:12]	Addr [11:8]	Addr [7:4]	Addr [3:0]
Protect bit erase	Setting of the 4th bus write cycle address for protect bit erase					
	0x5E0	Fixed to "0".			"0" is recommended.	
Protect bit program	PBA: Protect bit address (Setting of the 4th bus write cycle address for protect bit program)					
	0x5E0	Fixed to "0".	"0" is recommended.	Protect bit selection (Table 26-10)		"0" is recommended.
Memory swap	MSA: Memory Swap Address (Setting of the 4th bus write cycle address for memory swap)					
	0x5E0	Fixed to "0".	"0" is recommended.	Memory swap bit selection (Table 26-12)		"0" is recommended.

Note) Use the following values for address bit [19] depending on Flash memory size..
Memory size is 512KB or less: Always set to "0".
Memory size is over 512KB: If bus write to 512KB area or less, the bit is set to "0".
If bus write to over 512KB area, the bit is set to "1"

26.2.8.3 Area Address (AA) and Block Address (BA)

Table 26-2 shows area and block addresses. Specify any address included in the area/block to be erased in the 6th bus write cycle of the automatic block erase command. In the single chip mode, specify using the address of mirror area.

26.2.8.4 How to Specify Protect Bit (PBA)

A protect bit is specified in the units of 1-bit in operation.

Table 26-10 shows a protect bit selection table of the automatic protect bit program.

Table 26-10 Protect bit program address

Block	Page	Register	Protect bit	PBA[11:4]							Example of address [31:0]
				Address [11:10]	Address [9]	Address [8]	Address [7]	Address [6]	Address [5]	Address [4]	
0	0	FCPSR0	<PG0>	0	0	0	0	0	0	0	0x5E00_0000
	1		<PG1>	0	0	0	0	0	0	1	0x5E00_0010
	2		<PG2>	0	0	0	0	0	1	0	0x5E00_0020
	3		<PG3>	0	0	0	0	0	1	1	0x5E00_0030
	4		<PG4>	0	0	0	0	1	0	0	0x5E00_0040
	5		<PG5>	0	0	0	0	1	0	1	0x5E00_0050
	6		<PG6>	0	0	0	0	1	1	0	0x5E00_0060
	7		<PG7>	0	0	0	0	1	1	1	0x5E00_0070
1	8 to 15		<BLK1>	0	0	0	1	0	0	0	0x5E00_0080
2	16 to 23		<BLK2>	0	0	0	1	0	0	1	0x5E00_0090
3	24 to 31		<BLK3>	0	0	0	1	0	1	0	0x5E00_00A0
4	32 to 39		<BLK4>	0	0	0	1	0	1	1	0x5E00_00B0
5	40 to 47		<BLK5>	0	0	0	1	1	0	0	0x5E00_00C0
6	48 to 55		<BLK6>	0	0	0	1	1	0	1	0x5E00_00D0
7	56 to 63		<BLK7>	0	0	0	1	1	1	0	0x5E00_00E0
8	64 to 71		<BLK8>	0	0	0	1	1	1	1	0x5E00_00F0
9	72 to 79		<BLK9>	0	0	1	0	0	0	0	0x5E00_0100
10	80 to 87		<BLK10>	0	0	1	0	0	0	1	0x5E00_0110
11	88 to 95		<BLK11>	0	0	1	0	0	1	0	0x5E00_0120
12	96 to 103		<BLK12>	0	0	1	0	0	1	1	0x5E00_0130
13	104 to 111		<BLK13>	0	0	1	0	1	0	0	0x5E00_0140
14	112 to 119		<BLK14>	0	0	1	0	1	0	1	0x5E00_0150
15	120 to 127		<BLK15>	0	0	1	0	1	1	0	0x5E00_0160

Block	Page	Register	Protect bit	PBA[11:4]							Example of address [31:0]
				Address [11:10]	Address [9]	Address [8]	Address [7]	Address [6]	Address [5]	Address [4]	
16	128 to 135	FCPSR1	<BLK16>	0	0	1	0	1	1	1	0x5E00_0170
17	136 to 143		<BLK17>	0	0	1	1	0	0	0	0x5E00_0180
18	144 to 151		<BLK18>	0	0	1	1	0	0	1	0x5E00_0190
19	152 to 159		<BLK19>	0	0	1	1	0	1	0	0x5E00_01A0
20	160 to 167		<BLK20>	0	0	1	1	0	1	1	0x5E00_01B0
21	168 to 175		<BLK21>	0	0	1	1	1	0	0	0x5E00_01C0
22	176 to 183		<BLK22>	0	0	1	1	1	0	1	0x5E00_01D0
23	184 to 192		<BLK23>	0	0	1	1	1	1	0	0x5E00_01E0
24	193 to 199		<BLK24>	0	0	1	1	1	1	1	0x5E00_01F0
25	200 to 207		<BLK25>	0	1	0	0	0	0	0	0x5E00_0200
26	208 to 215		<BLK26>	0	1	0	0	0	0	1	0x5E00_0210
27	216 to 223		<BLK27>	0	1	0	0	0	1	0	0x5E00_0220
28	224 to 231		<BLK28>	0	1	0	0	0	1	1	0x5E00_0230
29	232 to 239		<BLK29>	0	1	0	0	1	0	0	0x5E00_0240
30	240 to 247		<BLK30>	0	1	0	0	1	0	1	0x5E00_0250
31	248 to 255		<BLK31>	0	1	0	0	1	1	0	0x5E00_0260

26.2.8.5 ID-Read Command (IA, ID)

Table 26-11 shows how to specify a code and the content using ID-Read command.

Table 26-11 ID-Read Command codes and contents

Code	ID[7:0]	IA[15:14]	Example of address [31:0]
Manufacture code	0x0098	00	0x5E00_0000
Device code	0x005A	01	0x5E00_4000
–	Reserved	10	–
Macro code	0x012D	11	0x5E00_C000

26.2.8.6 Memory Swap Bit Assignment (MSA)

Table 26-12 shows setting values of FCSWPSR[10:0] assigned in the 4th bus write cycle of auto memory swap command.

Table 26-12 Setting values of FCSWPSR[10:0] by memory swap command and example of address

FCSWPSR[10:0]	MSA[11:4]							Example of address [31:0]
	Address [11]	Address [10:9]	Address [8]	Address [7]	Address [6]	Address [5]	Address [4]	
FCSWPSR[0]	1	Fixed to "0".	0	1	0	0	0	0x5E00_0880
FCSWPSR[1]	1	Fixed to "0".	0	1	0	0	1	0x5E00_0890
FCSWPSR[2]	1	Fixed to "0".	0	1	0	1	0	0x5E00_08A0
FCSWPSR[3]	1	Fixed to "0".	0	1	0	1	1	0x5E00_08B0
FCSWPSR[4]	1	Fixed to "0".	0	1	1	0	0	0x5E00_08C0
FCSWPSR[5]	1	Fixed to "0".	0	1	1	0	1	0x5E00_08D0
FCSWPSR[6]	1	Fixed to "0".	0	1	1	1	0	0x5E00_08E0
FCSWPSR[7]	1	Fixed to "0".	0	1	1	1	1	0x5E00_08F0
FCSWPSR[8]	1	Fixed to "0".	1	0	0	0	0	0x5E00_0900
FCSWPSR[9]	1	Fixed to "0".	1	0	0	0	1	0x5E00_0910
FCSWPSR[10]	1	Fixed to "0".	1	0	0	1	0	0x5E00_0920

26.2.8.7 Example of Command Sequence

Example of Command Sequence

(Block0to31)

Command		Bus cycle						
		1	2	3	4	5	6	7
Read/Reset	Address	0x5E00_0000	–	–	–	–	–	–
	Data	0x0000_00F0	–	–	–	–	–	–
ID-Read	Address	0x5E00_5400	0x5E00_AA00	0x5E00_5400	IA	0x5E00_0000	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0090	0x0000_0000	ID	–	–
Auto chip erase	Address	0x5E00_5400	0x5E00_AA00	0x5E00_5400	0x5E00_5400	0x5E00_AA00	0x5E00_5400	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0010	–
Auto protect bit program	Address	0x5E00_5400	0x5E00_AA00	0x5E00_5400	PBA	–	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_009A	–	–	–
Auto protect bit erase	Address	0x5E00_5400	0x5E00_AA00	0x5E00_5400	0x5E00_0000	–	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_006A	–	–	–
Auto memory swap	Address	0x5E00_5400	0x5E00_AA00	0x5E00_5400	MSA	–	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_009A	–	–	–

(Block0to15)

Command		Bus write						
		1	2	3	4	5	6	7
Automatic program	Address	0x5E00_5400	0x5E00_AA00	0x5E00_5400	PA	Hereafter,16 bits data are consecutively written.		
	Data	0x0000_00AA	0x0000_0055	0x0000_00A0	PD			
Automatic area erase	Address	0x5E00_5400	0x5E00_AA00	0x5E00_5400	0x5E00_5400	0x5E00_AA00	0x5E00_0000	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0020	–
Automatic block erase	Address	0x5E00_5400	0x5E00_AA00	0x5E00_5400	0x5E00_5400	0x5E00_AA00	BA	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0030	–
Automatic page erase	Address	0x5E00_5400	0x5E00_AA00	0x5E00_5400	0x5E00_5400	0x5E00_AA00	PGA	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0040	–

(Block15to31)

Command		Bus write						
		1	2	3	4	5	6	7
Automatic program	Address	0x5E08_5400	0x5E08_AA00	0x5E08_5400	PA	Hereafter,16 bits data are consecutively written.		
	Data	0x0000_00AA	0x0000_0055	0x0000_00A0	PD			
Automatic area erase	Address	0x5E08_5400	0x5E08_AA00	0x5E08_5400	0x5E08_5400	0x5E08_AA00	0x5E08_0000	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0020	–
Automatic block erase	Address	0x5E08_5400	0x5E08_AA00	0x5E08_5400	0x5E08_5400	0x5E08_AA00	BA	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0030	–
Automatic page erase	Address	0x5E08_5400	0x5E08_AA00	0x5E08_5400	0x5E08_5400	0x5E08_AA00	PGA	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0040	–

26.2.9 Flowchart

26.2.9.1 Automatic Program

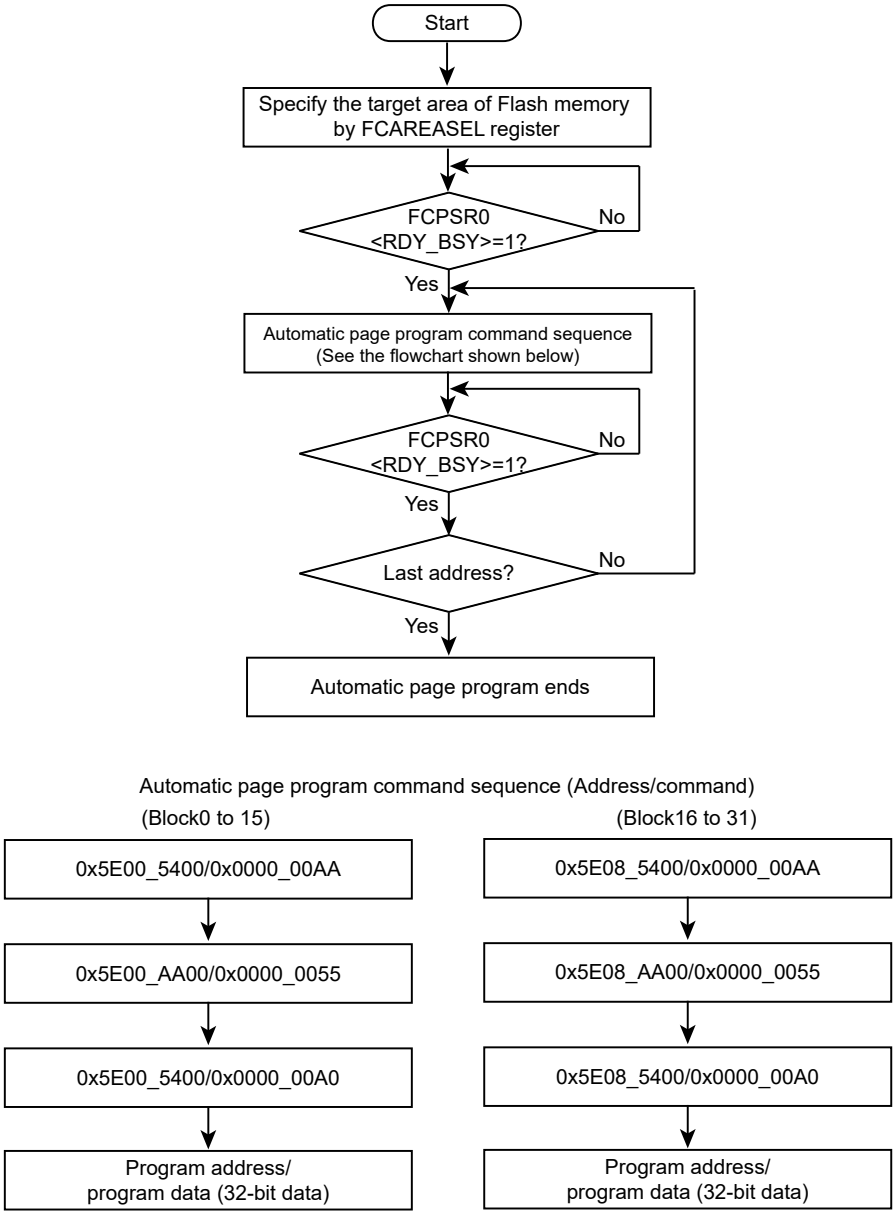


Figure 26-3 Flowchart of Automatic Program

26.2.9.2 Automatic Erase

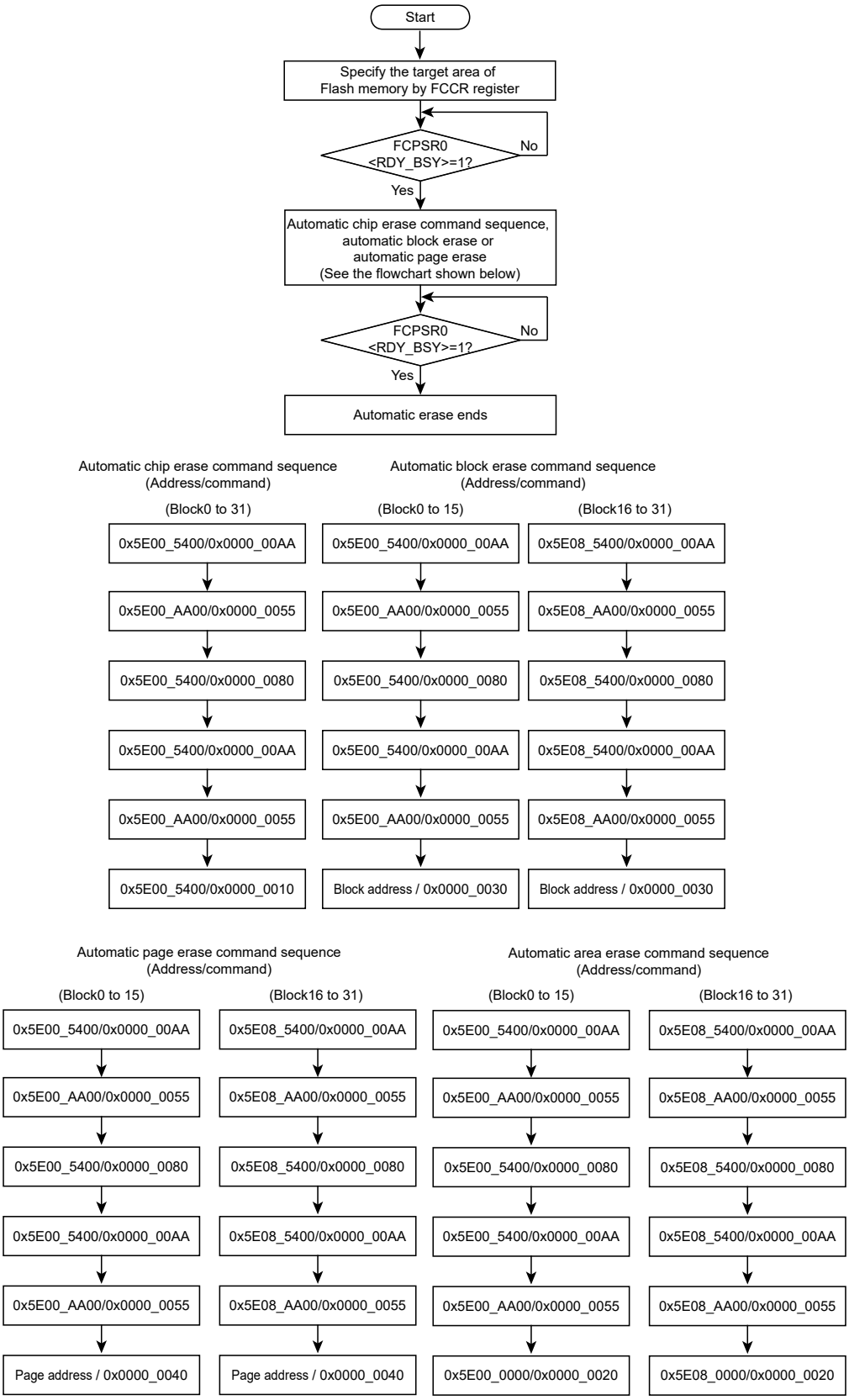


Figure 26-4 Flowchart of automatic erase

26.3 How to Reprogram Flash in Single Boot Mode

The single boot mode utilizes a program contained in built-in BOOT ROM for reprogramming Flash memory. In this mode, BOOT ROM is mapped to the area containing interrupt vector tables and Flash memory is mapped to another address area other than BOOT ROM area.

In the single boot mode, Flash memory is reprogrammed using serial command/data transfer. With connecting serial channel (SIO/UART) of this device to the external host, a reprogramming program is copied from the external host to the built-in RAM. A reprogramming routine in the RAM is executed to reprogram Flash memory. For details of communication with host, follow the protocol described later.

Even in the single boot mode, do not generate exception to avoid abnormal program termination.

To secure the contents of Flash memory in the single chip mode (normal operation mode), it is recommended to protect relevant Flash blocks against accidental erasure during subsequent single chip operations after re-programming is complete.

26.3.1 Mode Setting

In order to execute the on-board programming, this device is booted-up in the single boot mode. Below setting is for the single boot mode setting.

$\overline{BOOT}$ = 0
$\overline{RESET}$ = 0 → 1

While $\overline{BOOT}$ pin is set to the above conditions in advance, set $\overline{RESET}$ pin to "0". Then release $\overline{RESET}$ pin to boot-up in the single boot mode.

26.3.2 Interface Specification

This section describes the serial communication format in the single boot mode. The serial operation supports UART (asynchronous communication) mode. In order to execute the on-board programming, set the communication format of the programming controller as well.

Communication channel:	channel 0 (SIO/UART)
Serial transfer mode:	UART (asynchronous), half-duplex, LSB first
Data length:	8-bit
Parity bit:	Note
STOP bit:	1-bit
Baud rate:	Arbitrary baud rate

The internal boot program operates on the clock/mode control block setting as an initial condition. For detail of the initial setting of the clock, refer to "Clock/Mode control".

As explained in the "26.3.5.1 Serial Operation Mode Determination", a baud rate is determined by the 16-bit timer (TMRB). Since communications are executed by 1/16 of a desired baud rate when determining the baud rate; therefore the communication baud rate must be within the measurable range. The timer count clock operates at ΦT1(fc/2).

Table 26-13 shows the pins used in the boot program. Other than these pins are not used by the boot program.

Table 26-13 Pin connection

Pin		Connection is used orunused.
Mode setting pin	$\overline{BOOT}$	o
Reset pin	$\overline{RESET}$	o
Communication-pin	TXD0	o
	RXD0	o

o: used ×: unused

26.3.3 Restrictions on Built-in Memories

Note that the single boot mode places restrictions on the built-in RAM and built-in flash memory as shown in Table 26-14.

Table 26-14 Restrictions on the memories in the single boot mode

Memory	Restrictions
Internal RAM	Boot program uses the memory as a work area through 0x2000_0000 to 0x2000_03FF. Store the program 0x2000_0400 through the end address of RAM.
Internal Flash memory	The following addresses are assigned for storing software ID information and passo-werds. Should not use the following address for program storage. 0x5E0F_FFF0 to 0x5E0F_FFFF

Note: If a password is erased data (0xFF), it is difficult to protect data secure due to an easy-to-guess pass-word. Even if the single boot mode is not used, it is recommended to set a unique value as a password

26.3.4 Operation Command

The boot program provides the following operation commands

Table 26-15 Operation command data

Operation command da-ta	Operation mode
0x10	RAM transfer
0x40	Flash memory chip erase and protect bit erase

26.3.4.1 RAM Transfer

The RAM transfer is to store data from the controller to the internal RAM. When the transfer is com-plete normally, a user program starts. User program can use the memory address of 0x2000_0400 or later (except 0x2000_0000 to 0x2000_03FF) for the boot program. CPU will start at RAM store start address.

This RAM transfer function enables user-specific on-board programming control. In order to execute the on-board programming by a user program, use Flash memory command sequence explained in 26.2.8.

26.3.4.2 Flash Memory Chip Erase and Protect Bit Erase

"Flash memory chip erase and protect bit erase command" erases the entire blocks of Flash memory. This command erases all blocks regardless of write/erase protect condition or security status.

26.3.5 Common Operation regardless of Command

This section describes common operation under the boot program execution.

26.3.5.1 Serial Operation Mode Determination

The controller sends 0x86 of the 1st byte at the desired baud rate. Figure 32-6 shows waveforms in each case.

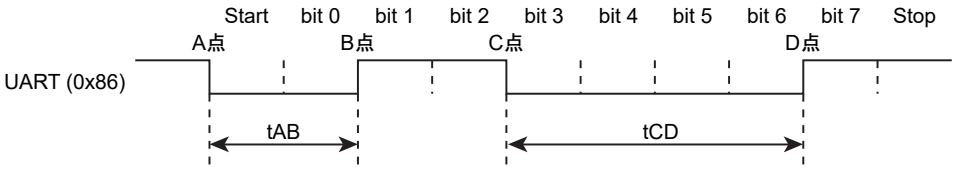


Figure 26-5 Serial operation mode determination data

Figure 26-6 shows a flow chart of internal boot program. Using 16-bit timer (TMRB) with the time of tAB, tAC and tAD, the 1st byte of serial operation mode determination data (0x86) after reset is provided. In Figure 26-6, the CPU monitors the level of the receive pin, and obtains a timer value at the moment when the receive pin's level is changed. Therefore, the timer values of tAB, tAC and tAD have a margin of error. In addition, note that if a transfer goes at a high baud rate, the CPU may not be able to determine the level of receive pin.

The flow chart in Figure 26-7 shows the serial operation mode is determined by whether the time length of the receive pin is long or short. If the length is tAB ≤ tCD, the serial operation mode is determined as UART mode. The time of tAD is used for whether the automatic baud rate setting is enable or not. If the length is tAB > tCD, the serial operation mode is not determined as UART Interface mode. Note that timer values of tAB, tAC and tAD have a margin of error. If the baud rate is high but operation frequency is low, each timer value becomes small. This may generate unexpected determination. (To prevent this problem, re-set UART within the programming routine.)

For example, the serial operation mode may not be determined to be UART Interface mode when the controller attempts to use UART mode. To avoid such situation, when UART mode is utilized, the controller should allow for a time-out period where the time is expected to receive an echo-back (0x86) from the target board. If it fails to obtain that echo-back within the allowed time, the controller is incapable of communications.

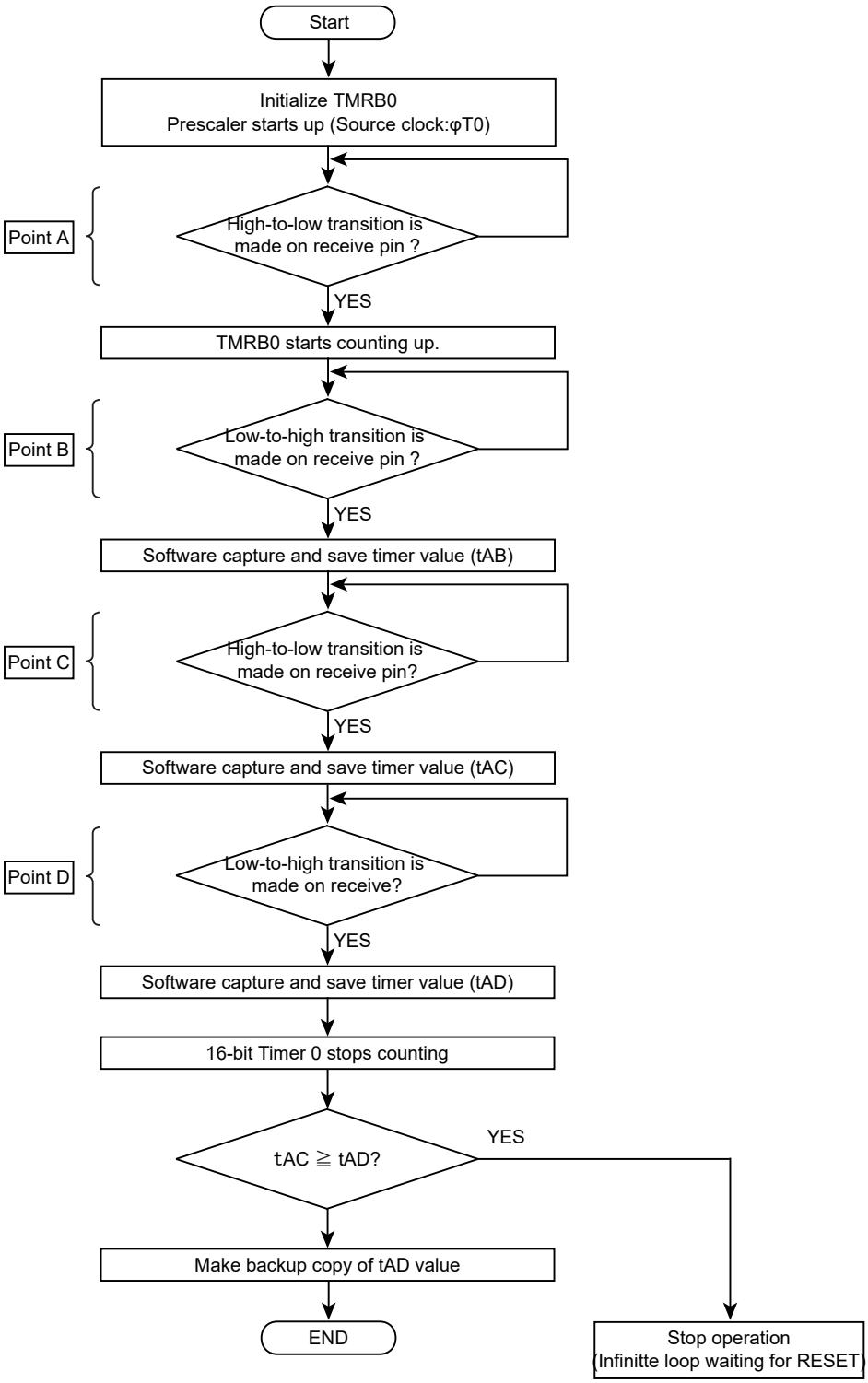


Figure 26-6 Serial operation mode receive flow chart

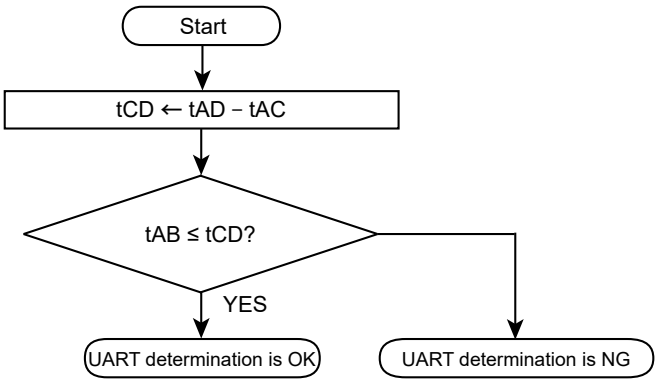


Figure 26-7 Serial operation mode determination flow chart

26.3.5.2 Acknowledge Response Data

The internal boot program represents processing states in specific codes and sends them to the controller. Table 26-16 to Table 26-19 show the values of acknowledge responses to each receive data.

In Table 26-17 to Table 26-19, the upper four bits of the acknowledge response are equal to those of upper 4 bits of the operation command data. The 3rd bit indicates a receive error. The 0th bit indicates an invalid operation command error, a checksum error or a password error. The 1st bit and 2nd bit are always "0".

Table 26-16 ACK response to the serial operation determination data

Transmit data	Description
0x86	Determined that UART communication is possible. (Note)

Note: When the serial operation is determined as UART, if the baud rate setting is determined as unacceptable, the boot program aborts without sending back any response.

Table 26-17 ACK response to the operation command data

Transmit data	Description
0x?8 (Note)	A receive error occurs in the operation command data.
0x?1 (Note)	An undefined operation command data is received normally.
0x10	Determined as a RAM transfer command.
0x40	Determined as a flash memory chip erase command.

Note: The upper 4 bits of the ACK response data are the same as those of the previous command data.

Table 26-18 ACK response to the CHECK SUM data

Transmit data	Description
0xN8 (Note)	A receive error occurred.
0xN1 (Note)	A CHECK SUM or password error occurred.
0xN0 (Note)	The CHECK SUM value was determined as correct.

Note: The upper 4 bits of the ACK response data are the same as those of the previous command data.

Table 26-19 ACK response to Flash memory chip erase and protect bit erase operation

Transmit data	Description
0x54	Determined as erase enable command.
0x4F	Erase command ended
0x4C	Erase command ended illegally.
0x47	Flash operation command was aborted.

26.3.5.3 Password Determination

The boot program uses the below area as the examination of necessity of password and password data area.

Area	Address
Examination of necessity of password	0x5E0F_FFF0 (1byte)
Password	0x5E0F_FFF4 to 0x5E0F_FFFF (12byte)

The RAM Transfer command performs a password verification regardless of necessity judging data.

Flash memory chip erase or protect bit erase command performs a password verification only when necessity judging is determined as "required".

Password requirement setting	Data
Need password	Other than 0xFF
No password	0xFF

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.

(1) Password verification using RAM transfer command

This item describes the password determination described in No.5 of "Table 26-21 Communication rules of RAM transfer".

If all these address locations contain the same bytes of data other than 0xFF, this condition is determined as a password area error as shown in Figure 26-8. In this case, the boot program returns an error acknowledge (0x11) in response to the checksum value regardless of the password verification.

The boot program verifies receive data (password data). A password error occurs if all 12 bytes do not match. If the password error is determined, an ACK response becomes a password error.

The password verification is performed even when the security function is enabled.

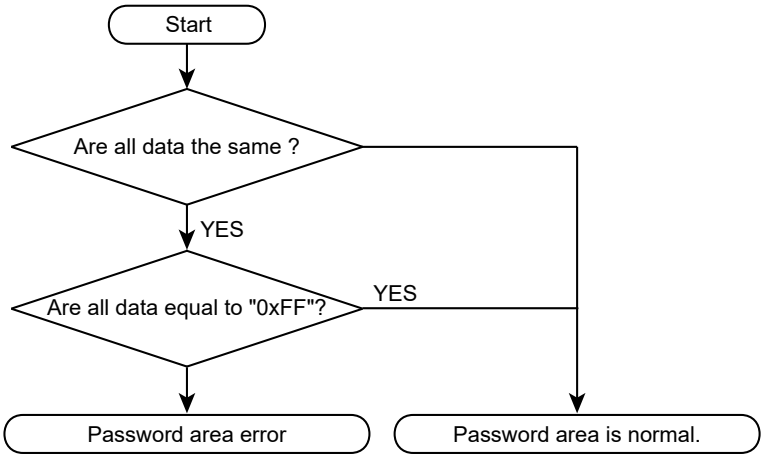


Figure 26-8 Password area check flow chart

(2) Password verification to Flash memory chip erase and protect bit erase command

This item describes the password determination described in No.5 of "Table 26-22 Communication Rules of Flash memory Chip Erase and Protect Bit Erase".

When a password is valid in the erase password necessity determination area as shown in Figure 26-9. If the passwords are identical data, a password area error occurs. If a password area error is determined, an ACK response to CHECK SUM value sends 0x41 regardless of the password verification.

The boot program verifies receive data (password data). A password error occurs if all 12 bytes do not match. If the password error is determined, an ACK response data to the CHECK SUM data is a password error. The password verification is performed even when the security function is enabled.

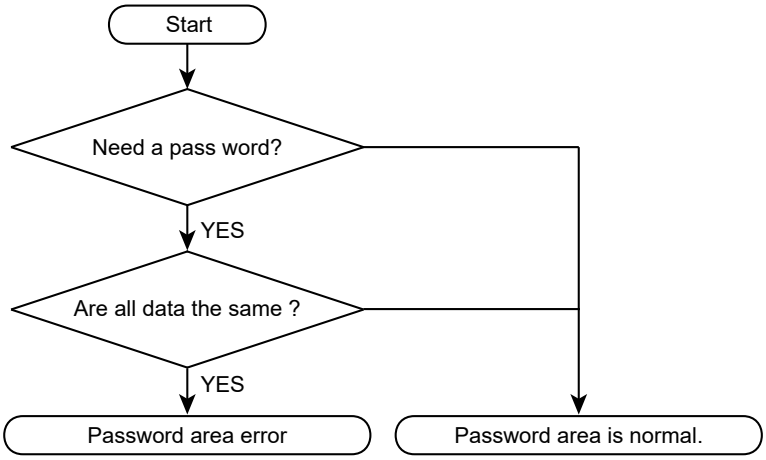


Figure 26-9 Password area check flow chart

26.3.5.4 CHECK SUM Calculation

The CHECK SUM is calculated by 8-bit addition (ignoring the overflow) to transmit data and taking the two's complement of the sum of lower 8 bits. Use this calculation when the controller transmits the CHECK SUM value.

Example calculation of CHECK SUM

To calculate the CHECK SUM for 2 bytes data (0xE5 and 0xF6), perform 8-bit addition.

0xE5 + 0xF6 = 0x1DB

Take the two's complement of the sum to the lower 8-bit, and that is a checksum value. So the boot program sends 0x25 to the controller.

0 − 0xDB = 0x25

26.3.6 Communication Rules for Determination of Serial Operation Mode

This section describes the communication rule for determination of serial operation mode. Transfer directions in the table are indicated as follows:

Transfer direction (C→T): Controller to TMPM46BF10FG

Transfer direction (C←T): TMPM46BF10FG to Controller

Table 26-20 Communication rules for determination of serial operation mode

No	Transfer direction	Transfer data	Description
1	C→T	Serial operation mode and baud rate setting	Controller transmits data to determine the serial operation mode. For details of mode determination of the target, refer to "26.3.5.1 Serial Operation Mode Determination".
		0x86	Controller transmits 0x86. If the target determines UART mode is OK, the target successively determines whether baud-rate setting is possible or not. If not, the program stops and communication is shutdown.
2	C←T	ACK response to serial operation mode	Receive data from the controller is the ACK response data responding to 1st byte of serial operation mode setting data. If the target determines the setting is possible, the target sets UART. A receive enable timing is set before transmit buffer is written to the data.
		Normal state: 0x86	If the target determines the setting is possible, the target transmits 0x86. If the target determines the setting is not possible, the target does not transmit anything and stops the operation. Set a timeout time (5 sec.) after the controller finished transmitting 1st byte of data. If the controller does not receive data (0x86) properly within the timeout time, it should be determined as a communication failure. If data (0x86) is not normally received within a time-out time, communications are not possible.
3	–	–	Controller transmits operation command data. For details of transfer format of each operation command, refer to "26.3.7 Communication Rules at RAM Transfer" or "26.3.8 Communication Rules of Flash memory Chip Erase and Protect Bit Erase".

26.3.7 Communication Rules at RAM Transfer

This section shows a communication rules of RAM transfer. Transfer directions in the table are indicated as follows:

Transfer direction (C→T): Controller to TMPM46BF10FG

Transfer direction (C←T): TMPM46BF10FG to Controller

Table 26-21 Communication rules of RAM transfer

No	Transfer direction	Transfer data	Description
1	C→T	Operation command data (0x10)	Controller transmits RAM transfer command data (0x10).
2	C←T	ACK response to the operation command Normal: 0x10 Abnormal: 0x11 Communication error: 0x18	Target checks received data and sends ACK response data. If a receive error exists, the target responses ACK response data 0x18 indicating communication error, and then returns to the initial state waiting for operation command data. If a receive error does not exist, the target checks the data against operation command data described in Table 26-15. If checking is failed, the target responses ACK response data 0x11 indicating abnormal state, and then returns to the initial stat waiting for operation command data. If checking is succeeded, the target responses ACK response data 0x10 indicating normal state, and then it waits for next data.
3	C→T	Password data (12 bytes)	The controller transmits data which is the same area as the password data of Flash memory. For details of password data area, refer to "26.3.5.3 Password Determination".
4	C→T	CHECK SUM value of transmit data (No.3)	The controller transmits a CHECKSUM value of transmit data (No.3). For details of CHECK SUM calculation, refer to "26.3.5.4 CHECK SUM Calculation".
5	C←T	ACK response to CHECK SUM value Normal: 0x10 Abnormal: 0x11 Communication error: 0x18	The target checks receive data and responses ACK response data. If a receive error exists, the target responses ACK response data 0x18 indicating communication error, and then returns to the initial state waiting for operation command data. If a receive error does not exist, the target checks a CHECK SUM value and password. For details of password determination, refer to "26.3.5.3 Password Determination". If password determination is failed, the target responses ACK response data 0x11 indicating abnormal state, and then returns to the initial state waiting for operation command data. If password determination is succeeded, the target responses ACK response data 0x10 indicating normal state, and then it waits for next transmit data.
6	C→T	RAM store start address 31 to 24	Transmit the RAM start address to be stored in RAM store data by dividing into 4 times as a next transmit data from the controller. Transmission order is as follows:1st byte corresponds to 31 bit to 24 bit and 4th byte corresponds to 7th bit to 0th bit of transfer address. These addresses should be placed in 0x2000_0400 through the last address of RAM address. The target checks receive data. If a receive error exists, the target responses ACK response data 0x18 indicating communication error, and then returns to the initial state waiting for operation command data. If a receive error does not exist, the target does not transmit anything, and it waits for next transmit data.
7	C→T	RAM store start address 23 to 16	
8	C→T	RAM store start address 15 to 8	
9	C→T	RAM store start address 7 to 0	
10	C→T	Number of RAM store bytes 15 to 8	Transmit the number of byte to be block-transferred from the controller. Transmission order is as follows: 1st byte corresponds to 15 bit to 8 bit and 2nd byte corresponds to 7th bit to 0th bit of transfer address. These addresses should be placed in 0x2000_0400 through the last address of RAM address. The target checks receive data. If a receive error exists, the target responses ACK response data 0x18 indicating communication error, and then returns to the initial state waiting for operation command data. If a receive error does not exist, the target does not transmit anything, and it waits for next transmit data.
11	C→T	Number of RAM store byte 7 to 0	
12	C→T	CHECK SUM value of transmit data (No.6 to 11)	Transmit a CHECK SUM value of transmit data (No.6 to 11) from the controller.

No	Transfer direction	Transfer data	Description
13	C←T	ACK response to CHECK SUM value Normal: 0x10 Abnormal: 0x11 Communication error: 0x18	The target checks receive data and responses ACK response data. If a receive error exists, the target responses ACK response data 0x18 indicating communication error, and then returns to the initial state waiting for operation command data. If a receive error does not exist, the target checks a CHECK SUM value. If checking is failed, the target responses ACK response data 0x11 indicating abnormal state, and then returns to the initial state waiting for operation command data. If checking is succeeded, the target responses ACK response data 0x10 indicating normal state, and then it waits for next data.
14	C→T	RAM stored data	Transmit data to be stored in RAM from the controller. The target receives data to be stored in RAM.
15	C→T	CHECK SUM value of transmit data (No. 14)	Transmit a CHECK SUM value of transmit data (No.14) from the controller.
16	C←T	ACK response to CHECK SUM value Normal:0x10 Abnormal: 0x11 Communication error: 0x18	The target checks receive data and responses ACK response data. If a receive error exists, the target responses ACK response data 0x18 indicating communication error, and then returns to the initial state waiting for operation command data. If a receive error does not exist, the target checks a CHECK SUM value. If checking is failed, the target responses ACK response data 0x11 indicating abnormal state, and then returns to the initial state waiting for operation command data. If checking is succeeded, the target responses ACK response data 0x10 indicating normal state and jumps to RAM store start address (No.6 to 9) as a branch address.

26.3.8 Communication Rules of Flash memory Chip Erase and Protect Bit Erase

This section shows a communication format of Flash memory chip erase and protect bit erase commands. Transfer directions in the table are indicated as follows:

Transfer direction "C→T": Controller ->TMPM46BF10FG

Transfer direction "C←T": Controller <-TMPM46BF10FG

Table 26-22 Communication Rules of Flash memory Chip Erase and Protect Bit Erase

No	Transfer direction	Transfer data	Description
1	C→T	Operation command data (0x40)	Sends Flash memory chip erase and protect bit erase command data (0x40).
2	C←T	ACK response to operation command Normal: 0x40 Abnormal: 0x41 Communication error: 0x48	ACK response data to the operation command. First, checks if 3rd byte of receive data has errors. (UART mode only) If receive errors exist, sends a ACK response data 0xX8 that means abnormal communication sand waits for a next operation command (3rd byte). Upper 4 bits of transmit data are undefined. (same as upper 4 bits of immediately before operation command data.) Note that in the I/O interface, receive error check is not performed. Then, if the 3rd byte of receive data corresponds to either operation command data in Table 26-11, receive data is echoed back. If the data does not correspond to the command in Table 26-11, sends a ACK response data 0xX1 that means operation command errors, and waits for next operation command. (3rd byte) Upper 4 bits of transmit data are undefined. (Upper 4 bits of immediate before operation command data are used.)
3	C→T	Password data (12 bytes)	Transmit data which is the same as password data area of Flash memory from the controller. However; if password requirement of Flash memory is set to "no" (data: 0xFF), the target does not conduct password verification, so that dummy data can be used as a password. For details of password data area, refer to "26.3.5.3 Password Determination".
4	C→T	CHECK SUM value of transmit data (No.3)	Transmit a CHECK SUM value of transmit data (No.3) from the controller. For a method of CHECK SUM calculation, refer to "26.3.5.4 CHECK SUM Calculation".
5	C←T	ACK response to CHECK SUM value Normal: 0x40 Abnormal: 0x41 Communication error: 0x48	The target checks receive data and responses ACK response data. If receive error exists, the target responses ACK response data 0x48 indicating abnormal communication, and then returns to the initial state waiting for operation command data. If receive error does not exist, the target checks a CHECK SUM value. If checking is failed, the target responses ACK response data 0x41 indicating abnormal communication, and then returns to the initial state waiting for operation command data. If checking is succeeded, the target performs password checking. If password requirement is set to "no", the target transmits ACK response data 0x40 indicating normal. If password requirement is set to "need password", the target checks the password. If password checking is failed, the target responses ACK response data 0x41 indicating abnormal communication, and then returns to the initial state waiting for operation command data. If password checking is succeeded, the target responses ACK response data 0x40 indicating normal and then it waits next data.
6	C→T	Erase enable command data (0x54)	Transmit erase enable command data (0x54) from the controller.
7	C←T	ACK response to erase enable command Normal: 0x54 Abnormal: 0x51 Communication error: 0x58	The target checks receive data and responses ACK response data. If receive error exists, the target responses ACK response data 0x58 indicating communication error, and then returns to the initial state waiting for operation command data. If receive error does not exist, the target checks erase enable command (0x54). If checking is failed, the target responses ACK response data 0x51 indicating abnormal communication, and then returns to the initial state waiting for operation command data. If checking is succeeded, the target responses ACK response data 0x54 indicating normal, and then chip erase process is performed.
8	C←T	ACK response to erase command Normal: 0x4F Abnormal: 0x4C Abort chip erase command: 0x47	The target responses the result of chip erase process. If any problems occur, the target responses ACK response data (0x4F) indicating normal. If an blank check error occurs, the target responses ACK response data (0x4C) indicating abnormal. If chip erase command is aborted, the target responses ACK response data (0x47) indicating abort and then returns to the initial state waiting for operation command data.

26.3.9 Boot Program Whole Flowchart

This section shows a boot program whole flow chart.

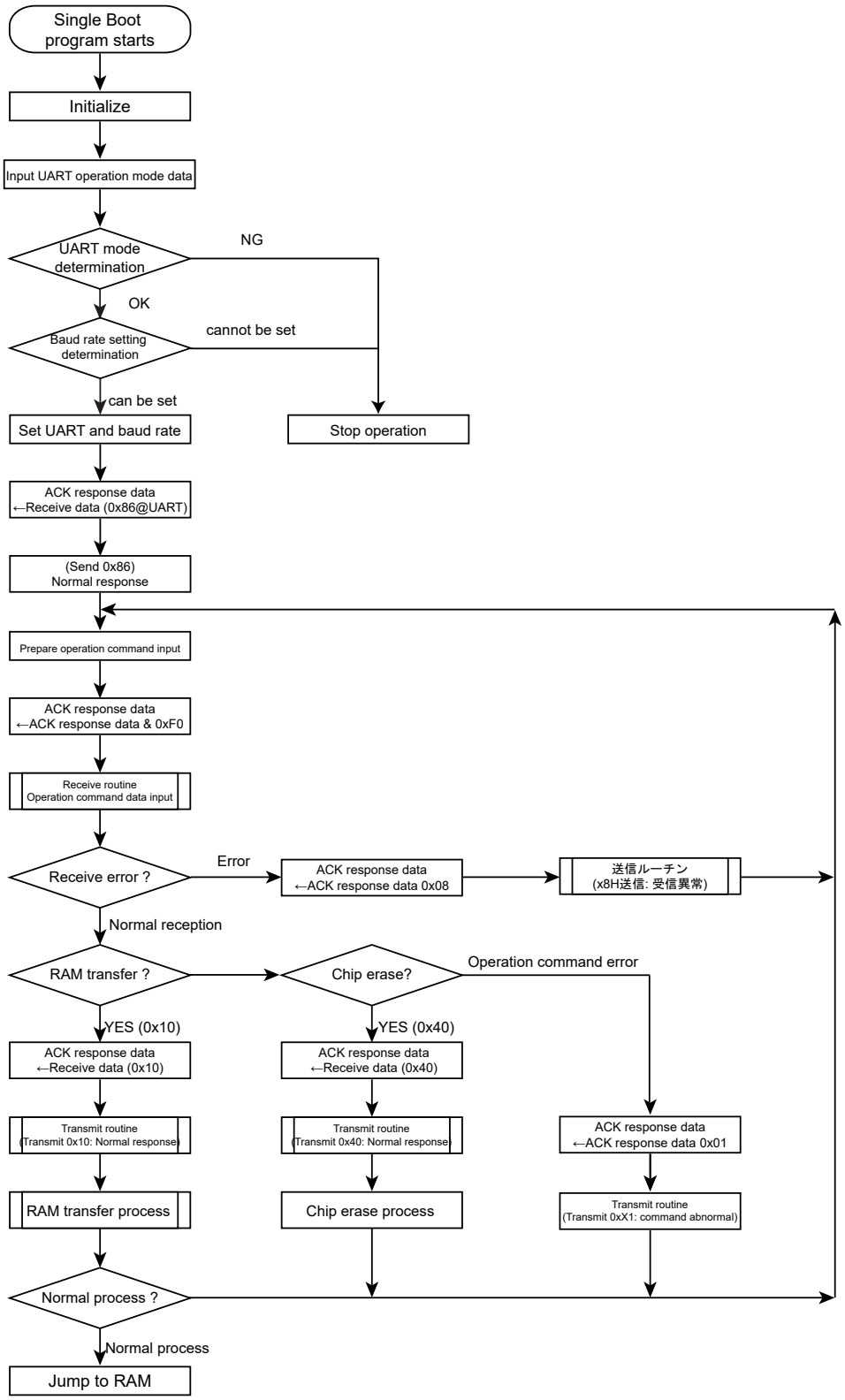


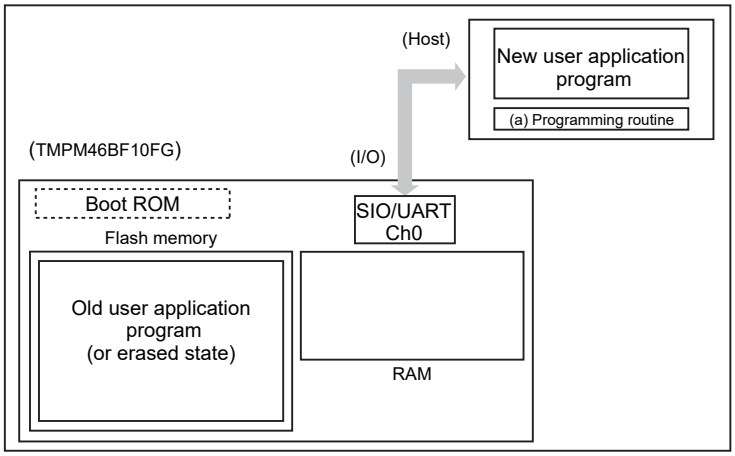
Figure 26-10 Boot Program Whole Flowchart

26.3.10 Reprogramming Procedure of Flash Using Reprogramming Algorithm in BOOT ROM

This section describes the reprogramming procedure of the flash using reprogramming algorithm in the on-chip boot ROM.(The Following example is using a ch0 of UART/SIO channel)

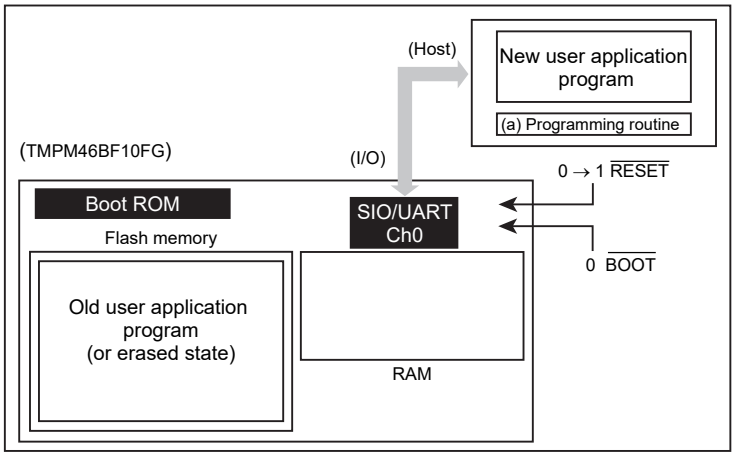
26.3.10.1 Step-1

The condition of Flash memory does not need to care whether a former user program has been written or erased. Since a programming routine and programming data are transferred via the SIO/UART, the SIO/UART must be connected to an external host. A programming routine (a) is prepared on the host.



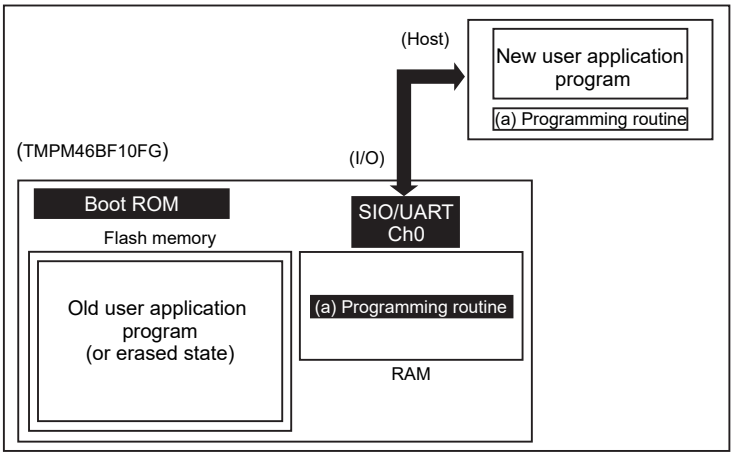
26.3.10.2 Step-2

Release the reset by the pin condition setting in the boot mode and boot-up on the BOOT ROM. According to the procedure of boot mode, transfer the programming routine (a) via SIO/UART from the source (host). A password verification in the user application program is performed first. (If Flash memory is erased, erase data (0xFF) is dealt as a password.)



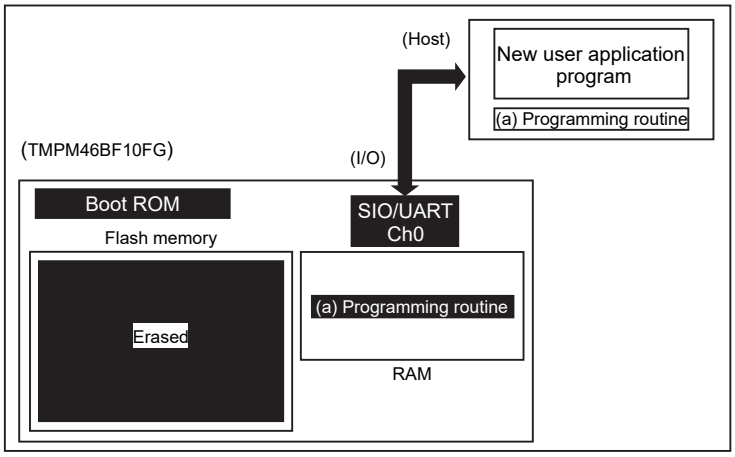
26.3.10.3 Step-3

If the password verification is complete, the boot program transfers a programming routine (a) from the host into the on-chip RAM. The programming routine must be stored in the range from 0x2000_0400 to the end address of RAM.



26.3.10.4 Step-4

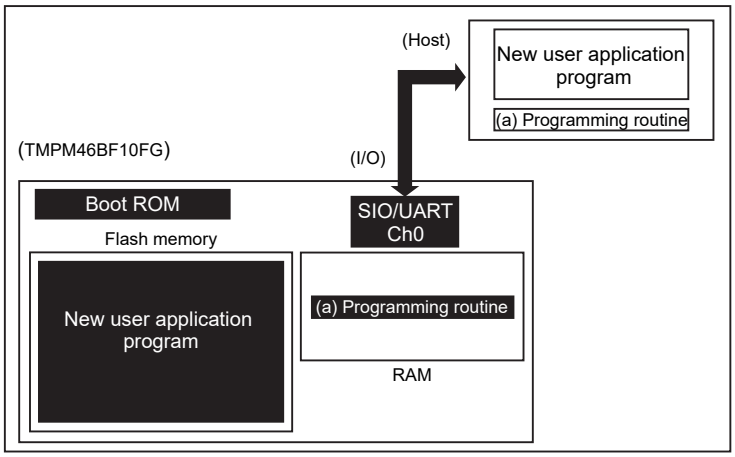
The boot program jumps to the programming routine (a) in the on-chip RAM to erase the Flash block containing old application program codes (The units of erase can be anything).



26.3.10.5 Step-5

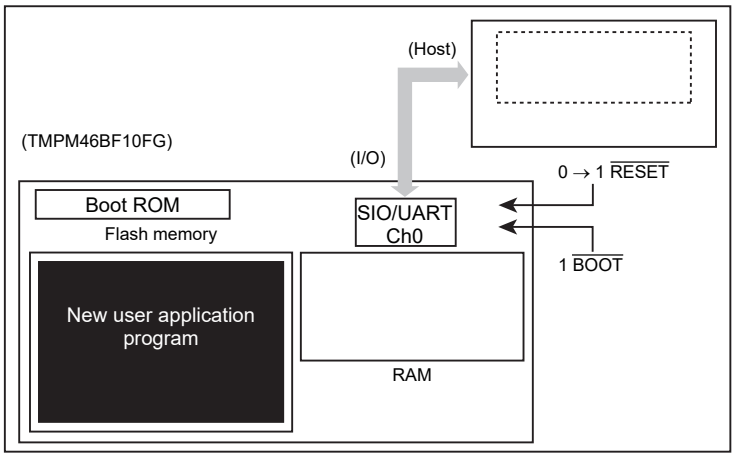
The boot program executes the programming routine (a) to download new application program codes from the host and programs it into the erased flash area. When the programming is complete, the writing or erase protection of that Flash area in the user's program must be ON.

In the example below, new program codes come from the same host via the same SIO/UART channel as for the programming routine. However, once the programming routine has begun to execute, it is free to change the transfer path and the source of the transfer. Create a hardware board and programming routine to suit your particular needs.



26.3.10.6 Step-6

When programming of Flash memory is complete, power off the board and disconnect the cable leading from the host to the target board. Turn on the power again, so that the device re-boots in the single-chip (Normal) mode to execute the new program.



26.4 Reprogramming in the User Boot Mode

A user boot mode is to use Flash memory programming on the internal RAM of users' set. It is used when the data transfer bus for Flash memory program code on the user application is different from the serial I/O. It operates in the single chip mode; therefore, normal mode, in which user application is activated in the single chip mode, is required to switch to the user boot mode for programming Flash memory. Consequently, add a mode judgment routine to the reset service routine in the user application program.

The condition to switch the modes needs to be set according to the user's system setup conditions. Also, a 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 the user boot mode. Once re-programming is complete, write/erase protection to the necessary block is recommended to avoid from accidental modification in the single chip mode (normal operation mode). Make sure not to generate exception to avoid abnormal termination even in the user boot mode.

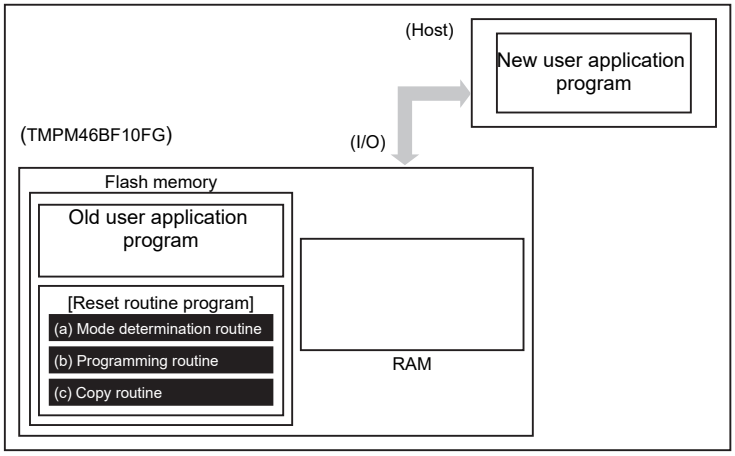
Taking examples from two cases such as the method that reprogramming routine stored in Flash memory (1-A) and transferred from the external device (1-B), the following section explains the procedure. For details of the program/erase Flash memory, refer to "26.2 Detail of Flash Memory".

26.4.1 (1-A) Procedure that a Programming Routine Stored in Flash memory

26.4.1.1 Step-1

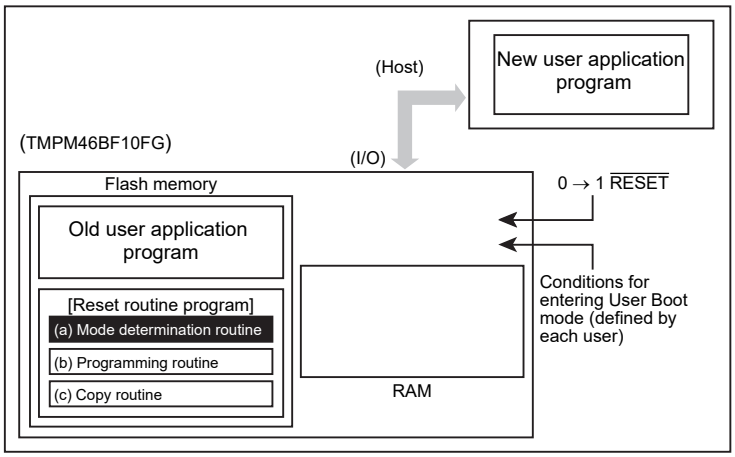
A user determines the conditions (e.g., pin status) to enter the user boot mode and the I/O bus to be used to transfer data. Then suitable circuit design and program are created. Before installing the device on a printed circuit board, write the following three program routines into an arbitrary Flash block using programming equipment such as a Flash writer.

(a) Mode determination routine:	A program to determine to switch to user boot mode or not
(b) Flash reprogramming routine:	A program to download new program from the external device and re-program Flash memory
(c) Copy routine:	A program to copy the data described in (a) to the built-in RAM.



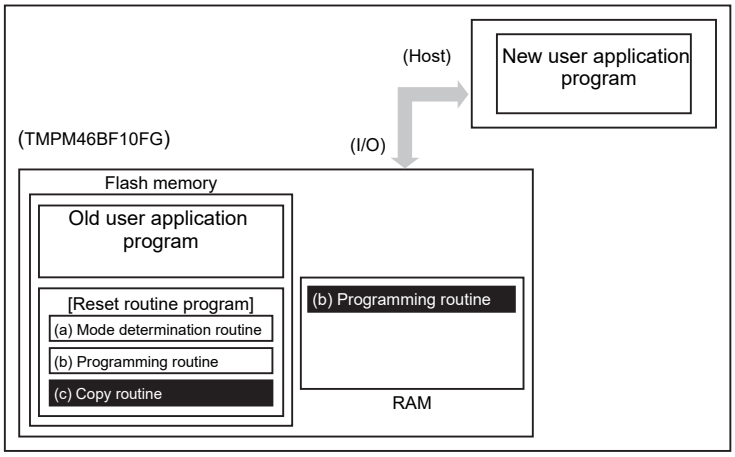
26.4.1.2 Step-2

This section explains the case that a programming routine stored in the reset routine. First, the reset routine determines to enter the user boot mode. If mode switching conditions are met, the device enters the user boot mode to reprogram data. (Prohibit to generate all exception after enter in user boot mode)



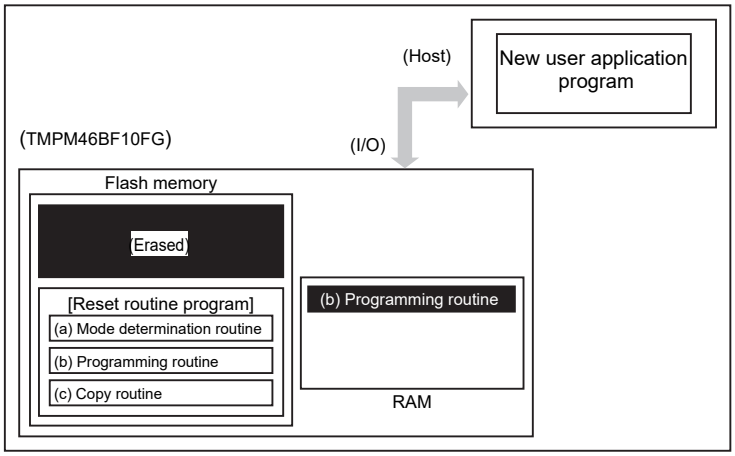
26.4.1.3 Step-3

Once the device enters the user boot mode, execute the copy routine (C) to download the Flash programming routine (b) from the host controller to the internal RAM.



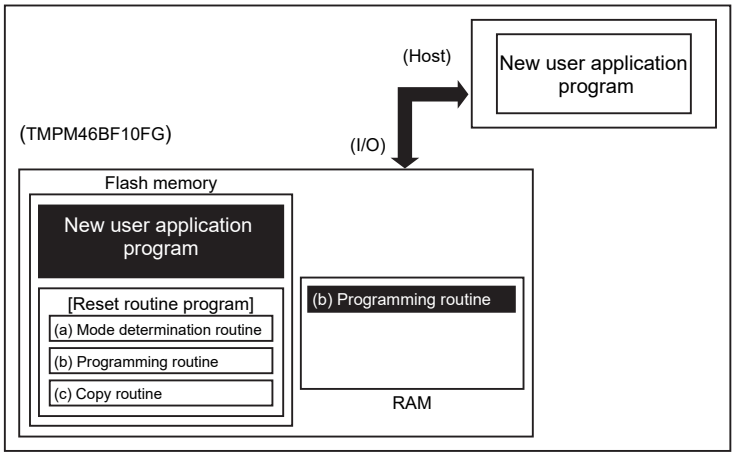
26.4.1.4 Step-4

Jump to the reprogramming routine on the RAM to release the write/erase protection for the old application program, and to erase Flash. (The units of erase can be anything)



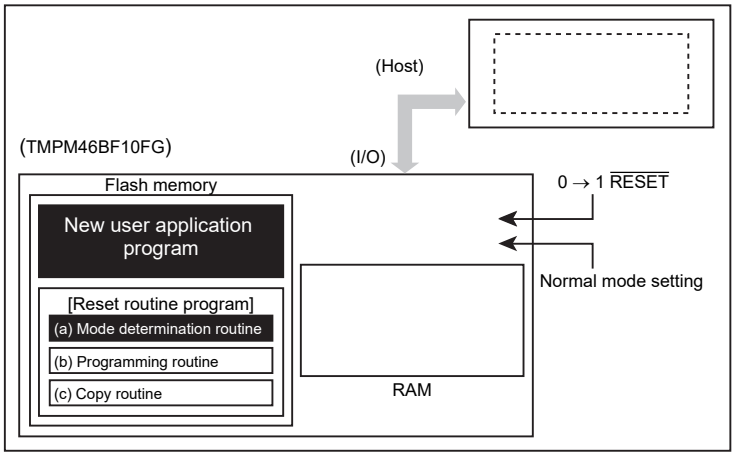
26.4.1.5 Step-5

Continue to execute the flash programming routine to download new program data from the host controller and program it into the erased Flash block. When the programming is complete, the write/erase protection of that flash block in the user program area must be ON.



26.4.1.6 Step-6

Do reset by setting "0" to $\overline{\text{RESET}}$. Upon reset, Flash memory is set to the normal mode. After reset, the CPU will start along with the new application program.



26.4.2 (1-B) Procedure that a Programming Routine is transferred from External Host

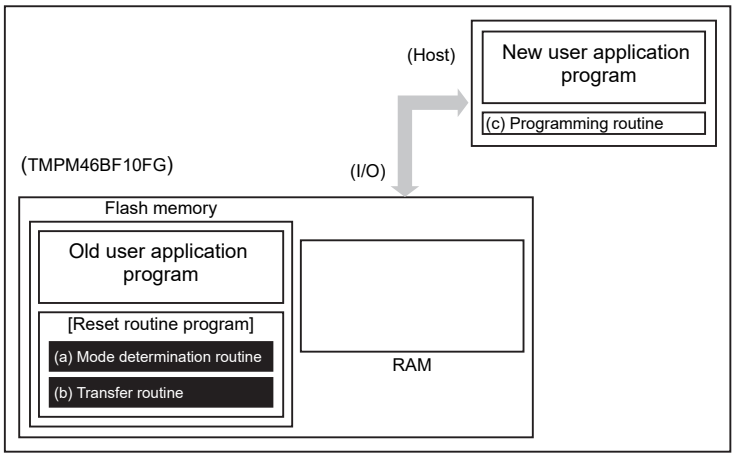
26.4.2.1 Step-1

A user determines the conditions (e.g., pin status) to enter the user boot mode and the I/O bus to be used to transfer data. Then suitable circuit design and program are created. Before installing the device on a printed circuit board, write the following two program routines into an arbitrary Flash block using programming equipment such as a Flash writer.

(a) Mode determination routine: A program to determine to switch to reprogramming operation
(b) Transfer routine: A program to obtain a reprogramming program from the external device.

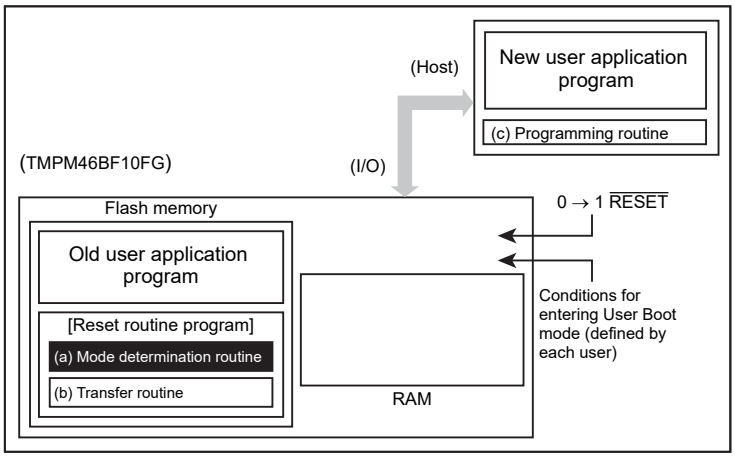
In addition, prepare a reprogramming routine shown below must be stored on the host controller.

(c) Reprogramming routine: A program to reprogram data



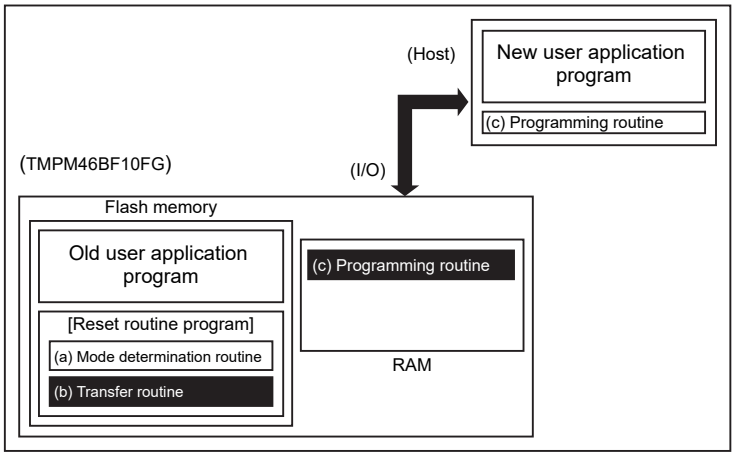
26.4.2.2 Step-2

This section explains the case that a programming routine stored in the reset routine. First, the reset routine determines to enter the user boot mode. If mode switching conditions are met, the device enters the user boot mode to reprogram data. (Prohibit to generate all exception after enter in user boot mode)



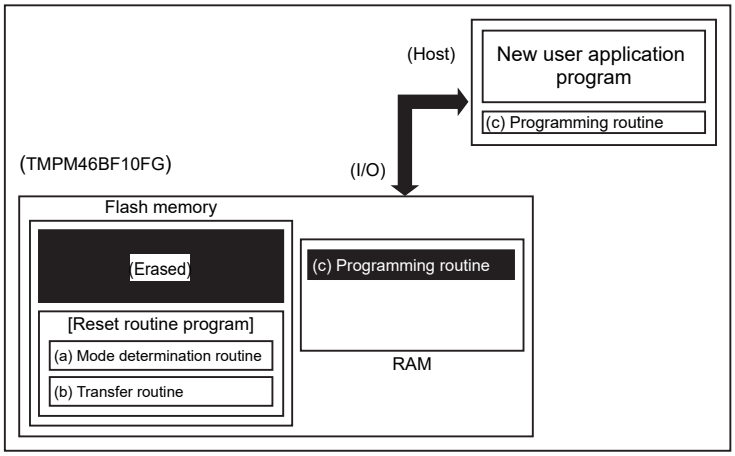
26.4.2.3 Step-3

Once the device enters the user boot mode, execute the transfer routine (b) to download the programming routine (c) from the host controller to the internal RAM.



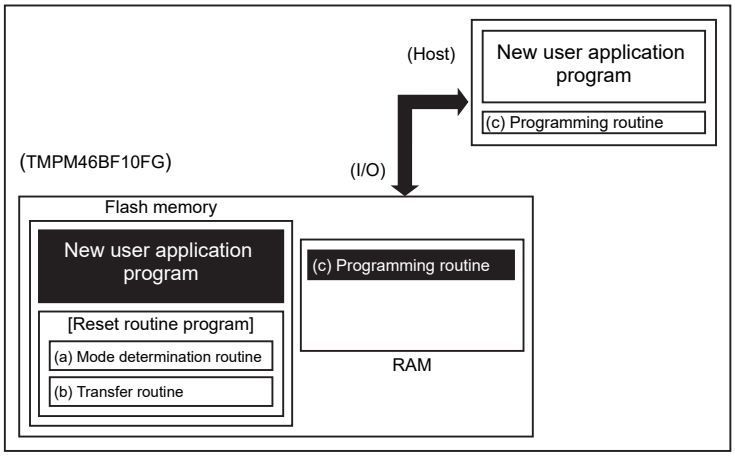
26.4.2.4 Step-4

Jump to the reprogramming routine in the internal RAM to release the write/erase protection for the old application program, and to erase Flash. (The units of erase can be anything)



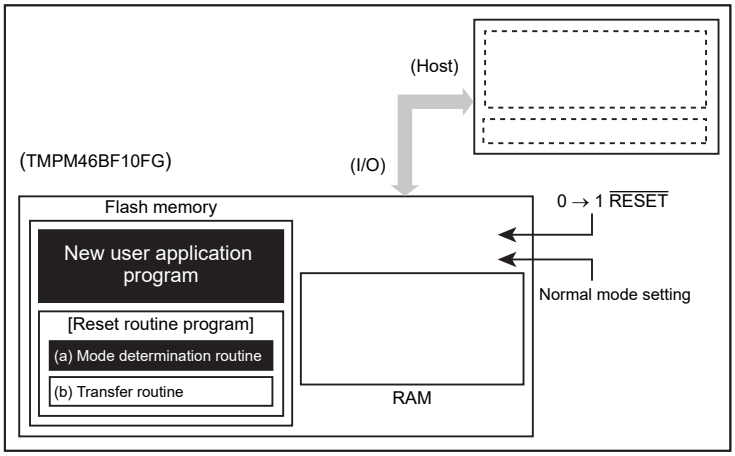
26.4.2.5 Step-5

Continue to execute the Flash programming routine (c) to download new program data from the host controller and program it into the erased Flash blocks. When the programming is complete, the write/erase protection of that Flash block in the user program area must be ON.



26.4.2.6 Step-6

Do reset by setting "0" to $\overline{\text{RESET}}$. Upon reset, Flash memory is set to the normal mode. After reset, the CPU will start along with the new application program.



26.5 How to Reprogramming using Dual Mode

The dual mode executes Flash reprogramming using the Flash memory reprogramming routine located in specified block on the users' set.

For example, while a program is executing on area 0, the other area (such as area 1) of Flash memory, on which instructions are not executed, can be written/erased. (The opposite case is possible.) Writing/erasing of Flash memory cannot be executed on the same area of Flash memory. Use different areas for writing/erasing of Flash memory.

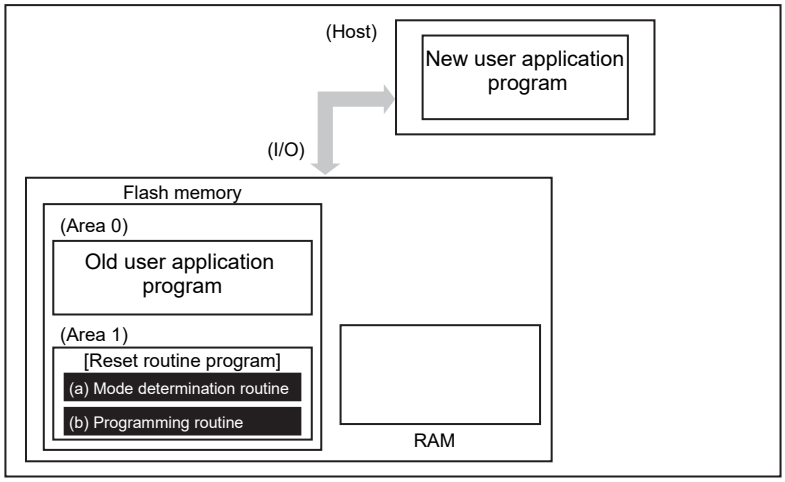
If an exception is generated in the dual mode, make sure not to read the target area in Flash memory.

26.5.1 Example of Flash Reprogramming Procedure

26.5.1.1 Step-1

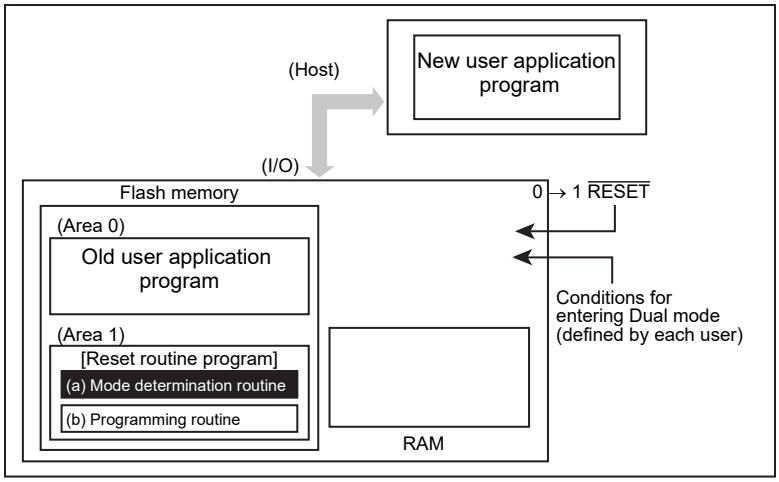
A user determines the conditions (e.g., pin status) to enter the on-board programming mode and the target area in Flash memory to be programmed or erased. Then suitable circuit design and program are created along to the users' conditions.

(a) Mode determination routine:	A program to determine to switch to user boot mode or not
(b) Flash programming routine:	A program to download new program from the host controller and re-program Flash memory



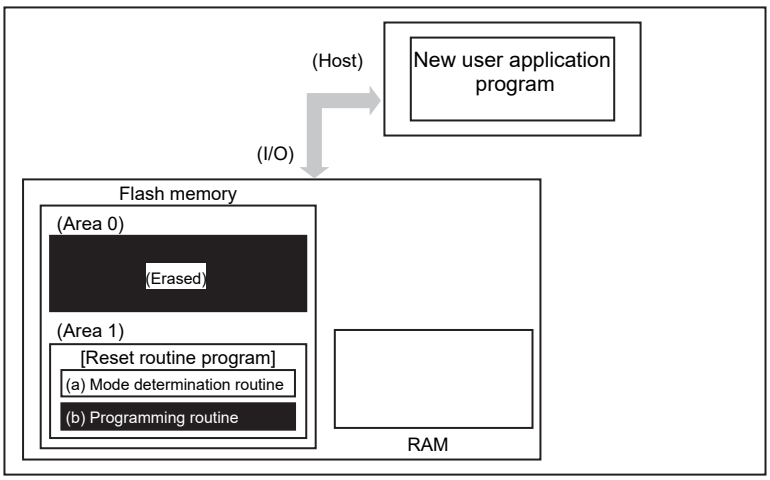
26.5.1.2 Step-2

This section explains the case that a programming routine stored in the reset routine. First, the reset routine determines to enter the dual mode.If mode switching conditions are met, the program jumps to the Flash reprogramming routine (transfers to dual mode).



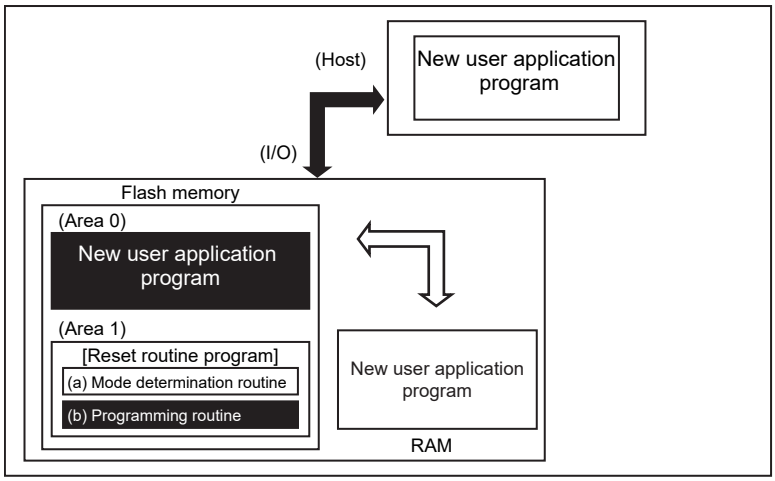
26.5.1.3 Step-3

Once the program jumps to the Flash reprogramming routine, the program releases the write/erase protection in the old user program area and erase areas in the units of areas or blocks or pages.



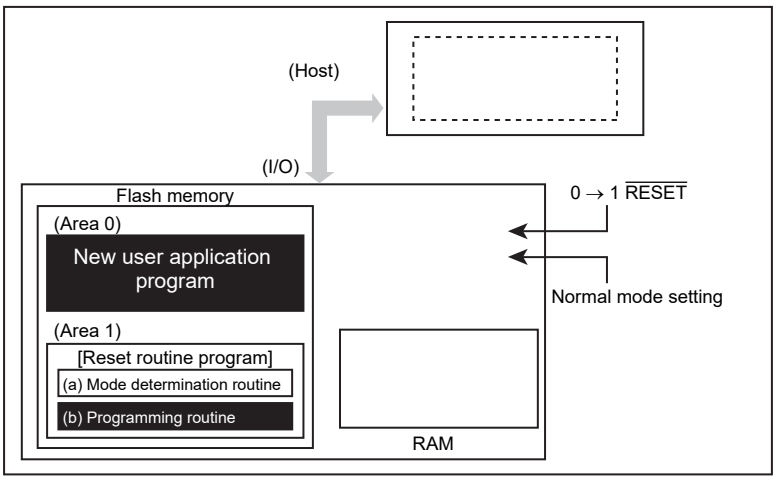
26.5.1.4 Step-4

Subsequently, confirm if the erased area of Flash are blank and then download a new users' application program data from the transfer source (host) to develop on the RAM. Developed data on the RAM is written to the erased area of Flash memory. When all data programming is complete, the write/erase protection of that Flash block in the user program area must be ON.



26.5.1.5 Step-5

Do reset by setting "0" to $\overline{\text{RESET}}$. Upon reset, Flash memory is set to the normal mode. After reset, the CPU will start along with the new application program



26.6 How to Reprogram Flash using User Boot Mode

This method switches the Page 0 area to Page 1 area to leave a user boot program using swap function when Flash memory is reprogrammed.

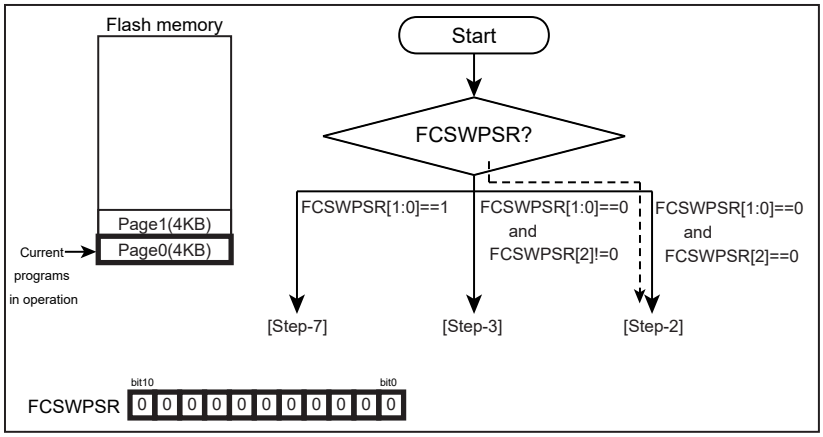
The following is an example of reprogramming procedure of user boot program.

(Assumed conditions in the following explanations: Swap size is 4K bytes. Page 1 program is copied from Page 0.)

26.6.1 Example of Flash Memory Reprogramming Procedure

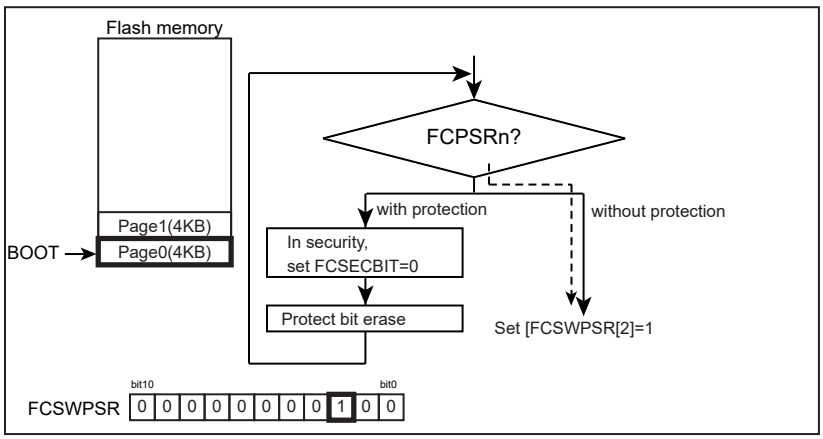
26.6.1.1 Step-1

Confirm if 0x0 is read from FCSWPSR[2:0].



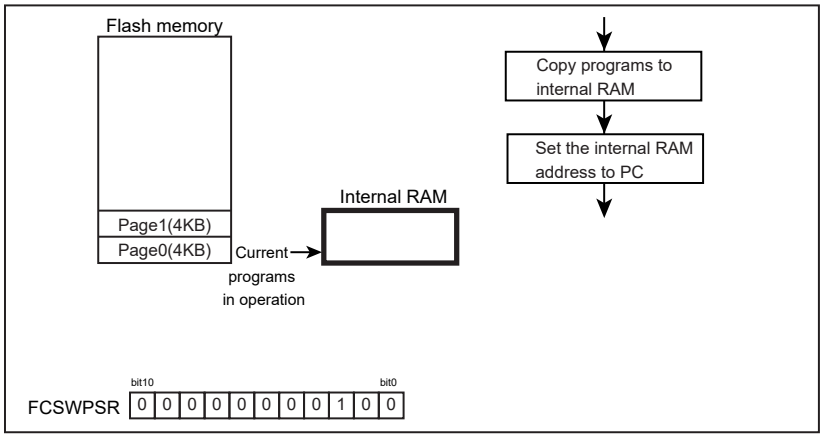
26.6.1.2 Step-2

Check each bit of FCPSR to confirm if the protect status is released. Then "1" is set to FCSWPSR[2] using automatic memory swap command.



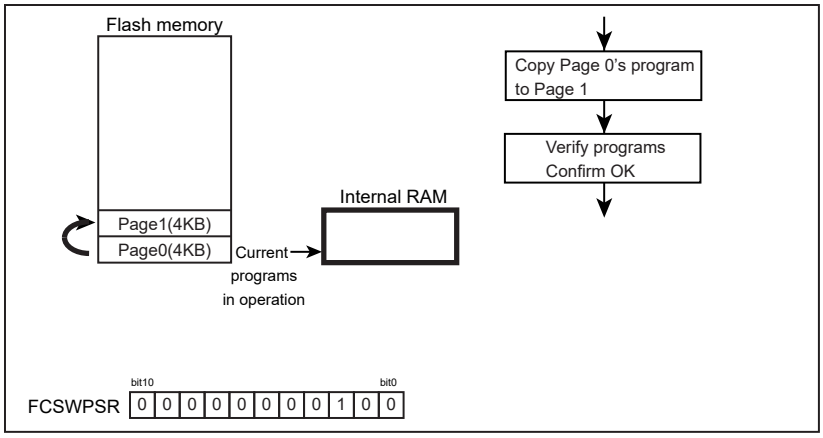
26.6.1.3 Step-3

Transfer the reprogramming routine to the internal RAM. Move PC (Program Counter) to the transferred program.



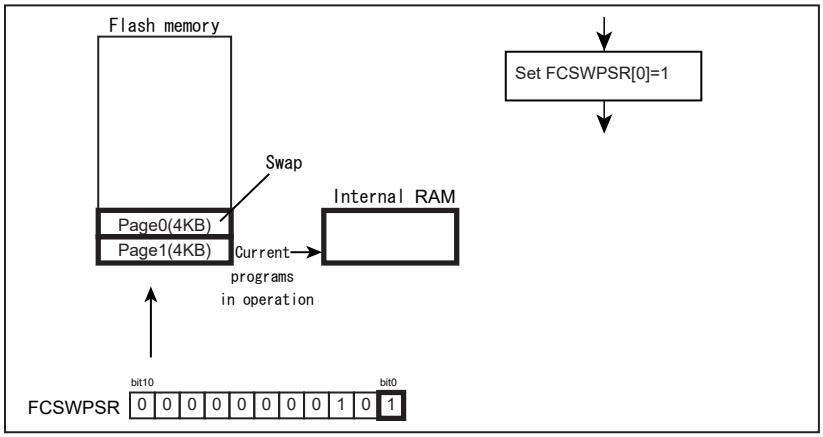
26.6.1.4 Step-4

Erase Page 1. Then write a program of Page0 to those of Page 1.



26.6.1.5 Step-5

Automatic memory swap command set "1" to FCSWPSR[0] to swap Page 0 with Page 1.

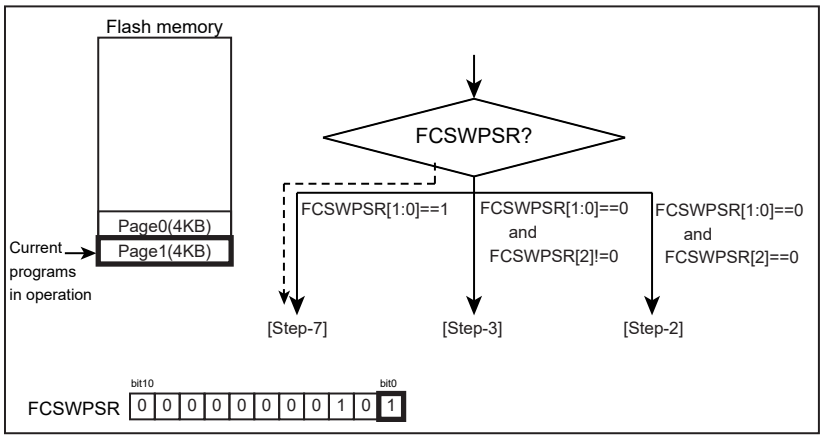


26.6.1.6 Step-6

Perform or release reset.

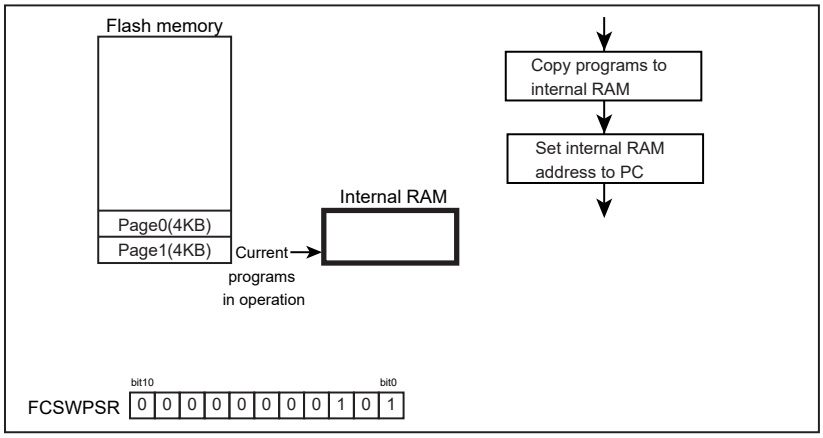
Page 1 is assigned to address 0 and Flash memory boots-up at Page 1.

A program branches to the conditioning routine that FCSWPSR[1:0] is set to "1". (To [Step-7])



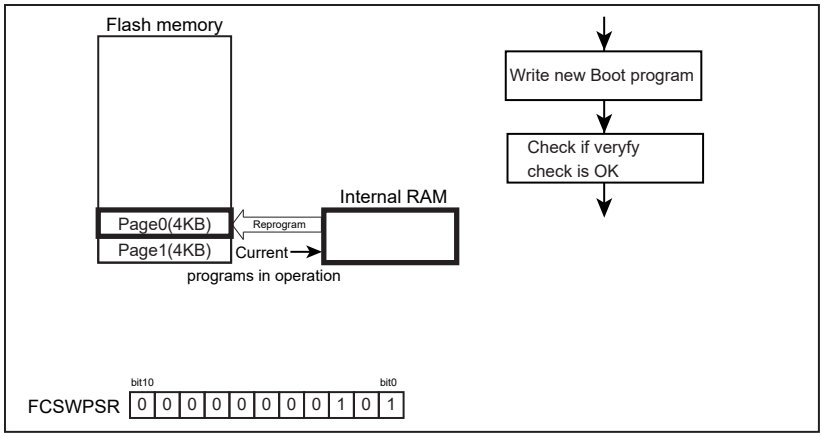
26.6.1.7 Step-7

Transfer the Flash reprogramming routine to the internal RAM then set the internal RAM address to PC (Program Counter).



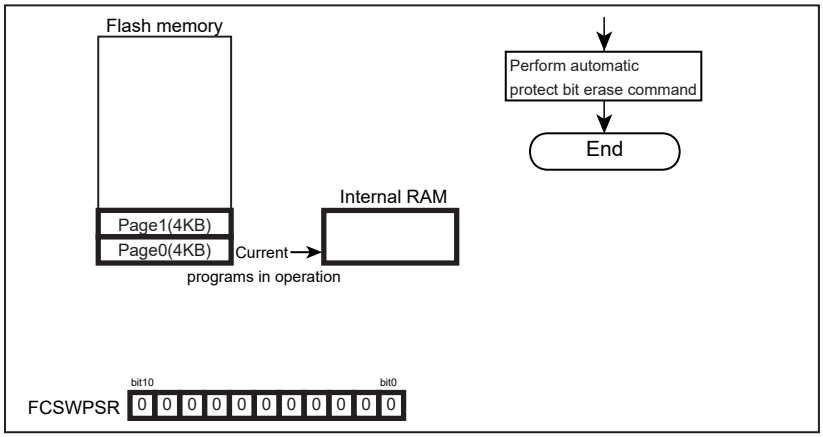
26.6.1.8 Step-8

Write a new boot program to Page0.



26.6.1.9 Step-9

Execute automatic protect bit erase command.



27. Debug Interface

27.1 Specification Overview

TMPM46BF10FG contains the Serial Wire JTAG Debug Port (SWJ-DP) unit for interfacing with the debug-ging tools and the Embedded Trace Macrocell™(ETM) unit for instruction trace output.Trace data is output to the dedicated pins(TRACEDATA[3:0], SWV) for the debugging via the on-chip Trace Port Interface Unit (TPIU).

For details about SWJ-DP, ETM and TPIU, refer to the Arm manual "Arm documentations set for the Cortex-M4".

27.2 SWJ-DP

SWJ-DP supports the Serial Wire Debug Port (SWCLK, SWDIO) and the JTAG Debug Port (TDI, TDO, TMS, TCK, $\overline{\text{TRST}}$).

Pin name	Function	Description	I/O
TMS	JTAG	JTAG Test Mode Selection	Input
SWDIO	SW	Serial Wire Data Input/Output	I/O
TCK	JTAG	JTAG Test Clock	Input
SWCLK	SW	Serial Wire Clock	Input
TDO	JTAG	JTAG Test Data Output	Output
SWV	SW	(Serial Wire Viewer Output)	(Output) (Note)
TDI	JTAG	JTAG Test Data Input	Input
$\overline{\text{TRST}}$	JTAG	JTAG Test $\overline{\text{RESET}}$	Input

Note:When SWV function is used, this pin is used as output pin.

27.3 ETM

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

27.4 Peripheral Functions in Halt Mode

When the Cortex-M4 processor with FPU core enters in the halt mode, the watchdog-timer (WDT) automatically stops. Other periph-eral functions continue to operate.

27.5 Connection with a Debug Tool

27.5.1 About connection with 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.

Note: Ensure that to measure the power-consumption with debug tool connected in STOP1/STOP2 mode is prohibitted.

27.5.2 Important points of using debug interface pins used as general-purpose ports

The debug interface pins can also be used as general-purpose ports.

After releasing reset, the particular pins of the debug interface pins are initialized as the debug interface pins. The other debug interface pins should be changed to the debug interface pins if needed.

If the debug interface pins are used as the general I/O port, please prepare the way to change the general I/O port to the debug interface pins beforehand.

Table 27-1 Example Table of using debug interface pins

	Debug interface pins						
	$\overline{TRST}$	TDI	TDO / SWV	TCK / SWCLK	TMS / SWDIO	TRACE DATA[3:0]	TRACE CLK
JTAG+SW (After reset)	o	o	o	o	o	×	×
JTAG+SW (without $\overline{TRST}$)	× (Note)	o	o	o	o	×	×
JTAG+TRACE	o	o	o	o	o	o	o
SW	×	×	×	o	o	×	×
SW+SWV	×	×	o	o	o	×	×

o : Enabled × : Disabled (Usable as general-purpose port)

Note: For the treatment of the pin of which the $\overline{TRST}$ function is assigned, select the $\overline{TRST}$ function with the function register and set the pin to OPEN or "High level".

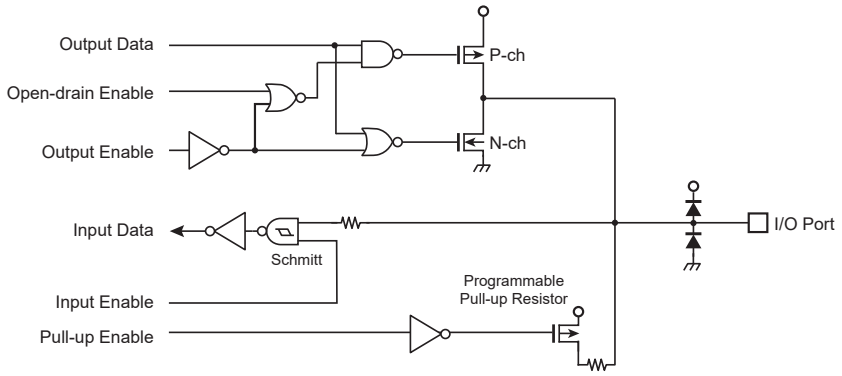
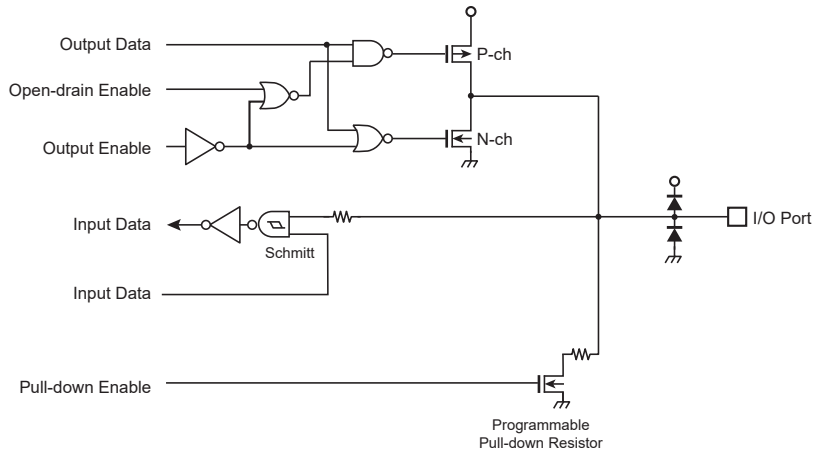
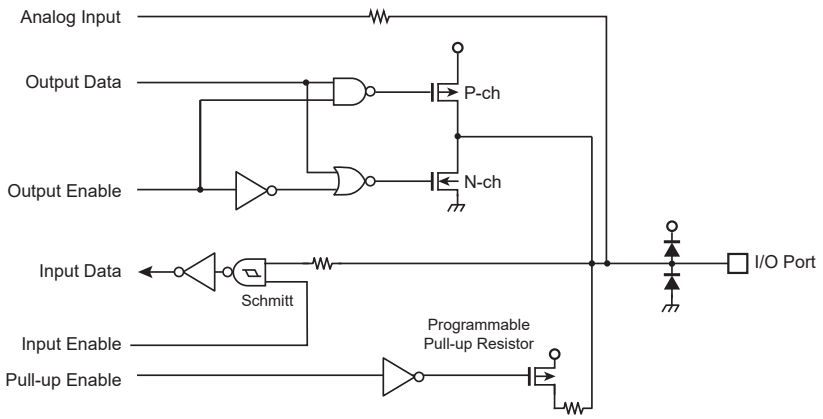
28. Port Section Equivalent Circuit Schematic

Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

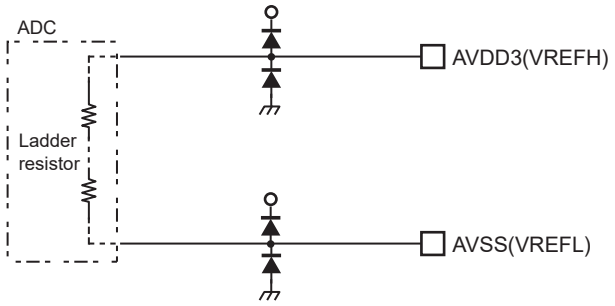
The input protection resistance ranges from several tens of Ω to several hundreds of Ω. Damping resistors X2 and XT2 are shown with a typical value.

Note: Resistors without values in the figure show input protection resistors.

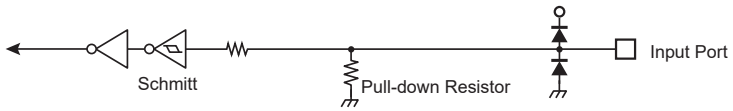
28.1 PORT pin

PA0,PA1, PA3 toPA7, PB0 toPB7, PC0 toPC5 , PD0 toPD4, PE0 toPE7, PF0 toPF7, PG0 toPG7, PH0 toPH3 , PK0 toPK4, PL0 toPL3	(Programmable open drain output, Programmable pull up , Schimitt input) 
PA2	(Programmable open drain output, Programmable pull down , Schimitt input) 
PJ0 to PJ7	(Analog input, Programmable pull up , Schimitt input) 

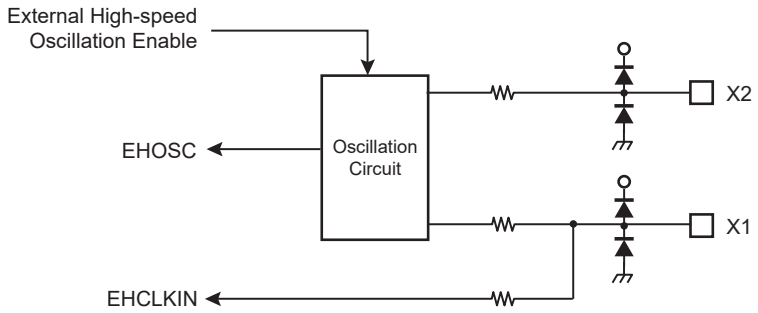
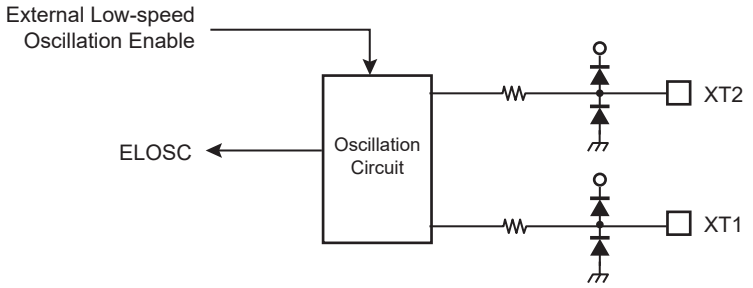
28.2 Analog pin

AVDD3, AVSS	

28.3 Control pin

MODE	 Mode pin must be connected to GND.
$\overline{\text{RESET}}$	

28.4 Clock pin

X1、X2	
XT1, XT2	

29. Electrical Characteristics

29.1 Absolute Maximum Ratings

Parameter		Symbol	Rating	Unit
Supply voltage		DVDD3	−0.3 to 3.9	V
		RVDD3	−0.3 to 3.9	
		AVDD3	−0.3 to 3.9	
Input voltage	Digital input pins	V_{IN1}	−0.3 to DVDD3 + 0.3	V
	Analog input pins	V_{IN2}	−0.3 to AVDD3 + 0.3	
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}	-55 to 125	°C
Operating Temperature		T_{OPR}	−40 to 85	°C

Note: **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 blow up and/or burning.**

29.2 DC Electrical Characteristics (1/2)

DVDD3 = RVDD3 = AVDD3 = 2.7 V to 3.6 V
DVSS = RVSS = AVSS = 0V
Ta = −40 to 85 °C

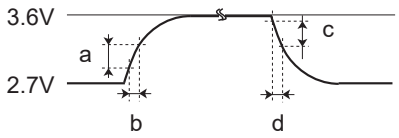
Parameter		Symbol	Condition	Min	Typ. (Note 1)	Max	Unit
Supply voltage	DVDD3 RVDD3 AVDD3	VDD	f_{OSC} = 8 to 40 MHz fsys = 1 to 120 MHz fs = 30 to 34 kHz	2.7	–	3.6	V
Low-level Input voltage	PA0 to 7, PB0 to 7, PC0 to 5, PD0 to 4, PE0 to 7, PF0 to 7, PG0 to 7, PH0 to 3, PK0 to 4, PL0~3	V_{IL1}	–	-0.3	–	0.25 DVDD3	V
	PJ0 to 7	V_{IL2}				0.25 AVDD3	
	X1, MODE, $\overline{RESET}$	V_{IL3}				0.2 DVDD3	
High-level Input voltage	PA0 to 7, PB0 to 7, PC0 to 5, PD0 to 4, PE0 to 7, PF0 to 7, PG0 to 7, PH0 to 3, PK0 to 4, PL0 to 3	V_{IH1}	–	0.75 DVDD3	–	DVDD3+0.3	V
	PJ0 to 7	V_{IH2}		0.75 AVDD3		AVDD3+0.3	
	X1, MODE, $\overline{RESET}$	V_{IH3}		0.8 DVDD3		DVDD3+0.3	
Low-level output voltage		V_{OL1}	I_{OL} =1mA	–	–	0.2 DVDD3	V
High-level output voltage		V_{OH1}	I_{OH} = −1 mA	0.8 DVDD3	–	DVDD3	V

Note 1: open-drain output
Note 2: CMOS output

DVDD3 = RVDD3 = AVDD3 = 2.7 V to 3.6 V
DVSS = RVSS = AVSS = 0V
Ta = −40 to 85 °C

Parameter	Symbol	Condition	Min	Typ. (Note 1)	Max	Unit
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	
Schmitt trigger input width	VTH	2.7 V ≤ DVDD3 ≤ 3.6 V	0.075DVDD3	–	–	V
Pull-up resistor at Reset	RRST	–	40	80	180	kΩ
Programmable pull-up/pull-down resistor	PKH	–	40	80	180	kΩ
Power supply variation rate in operation range	VRS	RVDD3 = DVDD3	–	–	5	mV/μs
	VFS		–	–	-5	
Pin capacitance (Except power supply pins)	C_{IO}	fc = 1 MHz	–	–	10	pF
Low-level output current	I_{OL}	Per pin	–	–	2	mA
	ΣI_{OL}	・Per Port GrA : FPA0-7 GrE : FPE0-7 GrF : FPF0-7 GrG : FPG0-7 GrJ : FPJ0-7	–	–	10	mA
		・Per Port GrB : FPB2-7,PD0-4,PH0-3 GrC : FPB0-1,PC0-5,PK0-4,PL0-3	–	–	20	mA
	ΣI_{OL}	Total, all ports	–	–	35	mA
High-level output current	I_{OH}	Per pin	–	–	-2	mA
	ΣI_{OH}	・Per Port GrA : FPA0-7 GrE : FPE0-7 GrF : FPF0-7 GrG : FPG0-7 GrJ : FPJ0-7	–	–	-10	mA
		・Per Port GrB : FPB2-7,PD0-4,PH0-3 GrC : FPB0-1,PC0-5,PK0-4,PL0-3	–	–	-20	mA
	ΣI_{OH}	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, RVDD3 and AVDD3.
Note 3: VRS(Rising), VFS(Falling) should be measured at a strict level against a characteristics.



VRS = a / b , VFS = c / d

29.3 DC Electrical Characteristics (2/2)

Ta = −40 to 85 °C

Parameter	Symbol	condition				Min	Typ. (Note1)	Max	Unit
		Operation Voltage	High-speed oscillation	Low-speed oscillation	Operation condition				
NORMAL	I_{DD}	DVDD3 = RVDD3 = AVDD3 = 3.6V	Refer to Table 29-1, Table 29-2 regarding to the operation condition			–	44.8	67.3	mA
			Enabled	Enabled	CPU only	–	24.0	44.0	
IDLE			Refer to Table 29-1, Table 29-2 regarding to the operation condition			–	19.4	37.5	
STOP1			Disabled	Enabled	Refer to Table 29-1, Table 29-2 regarding to the operation condition	–	0.45	16.0	
STOP2						–	8.9	111.0	μA
				Disabled		–	7.3	108.0	

Note: Ta = 25 °C, DVDD3 = RVDD3 = AVDD3 = 3.3 V, unless otherwise noted.

Table 29-1 I_{DD} Measurement Condition (Pin condition, Oscillator)

		NORMAL	IDLE	STOP1	STOP2	
				Low-speed oscillator enabled		Low-speed oscillator disabled
Pin condition	DVDD3 = RVDD3 = AVDD3 =	3.3 V(typ),3.6V(Max)				
	X1, X2 pin	Connected to the high-speed oscillator (10MHz)				
	XT1, XT2 pin	Connected to the low-speed oscillator (32.768kHz)				
	Input pin	Fixed				
	Output pin	Open				
Operation condition (Oscillator)	System clock (fsys)	120MHz		Disabled		
	External high-speed oscillator (EHOSC)	Enabled		Disabled		
	Internal high-speed oscillator (IHOSC)	Disabled				
	PLL for fsys	Enabled (by 12)		Disabled		
	Low-speed oscillator (ELOSC)	Enabled				Disabled

Table 29-2 I_{DD} Measurement condition (CPU, peripheral functions)

Circuit	Channel / Unit	NORMAL	IDLE	STOP1	STOP2	
				Low-speed oscillator enabled		Low-speed Disabled
CPU	1	Enabled (Dhrystone Ver. 2.1)	Disabled			
DMAC	3	Unit A (Transfer source:UART0 transmit, destination: EBIF) Unit B/C (Transfer source: TMRB0/8, destination: RAM)	Disabled			
ADC	1	Enabled (1.33μs, Repeat mode)	Disabled			
EBIF	1	Enabled	Disabled			
TMRB	8	All ch :Enabled	Disabled			
MPT	4	Enabled	Disabled			
RTC	1	Enabled				Disabled
WDT	1	Enabled	Disabled			
UART/SIO	4	All ch:UART, transmit	Disabled			
UART	2	All ch :transmit (7.38Mbps)	Disabled			
I2C	3	Disabled				
SSP	3	Ch0:SPI ,transmit ,20MHz ch1,ch2: SPI,transimit,10MHz	Disabled			
I/O port	-	Disabled				
LVD	1	Disabled				

29.4 12-bit AD Converter Electrical Characteristics

DVDD3 = RVDD3 = AVDD3 = 2.7 V to 3.6 V
DVSS = RVSS = AVSS = 0V
Ta = −40 to 85 °C

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage (+)	AVDD3	−	DVDD3-0.3	−	DVDD3	V
Analog input voltage	VAIN	−	AVSS	−	AVDD3	V
Consumption current	I_{DD}	−	−	2.9	4.0	mA
INL error	−	AIN resistance ≤ 300 Ω AIN load capacitance ≥ 0.1 μF Conversion time ≥ 1.0 μs	−	−	±8	LSB
DNL error			−	−	±7	
Zero-scale error			−	−	±8	
Full-scale error			−	−	±8	
Total error			−	−	±9	
Stable time	Tsta	After setting ADMOD1<DA-CON> to "1"	-	-	3.0	μs
Conversion time	Tconv	−	1.0	−	10	μs

Note 1: 1LSB = (AVDD3(VREFH) − AVSS(VREFL)) / 4096 [V]
Note 2: This characteristics is shown in operating only ADC.

29.5 AC Electrical Characteristics

29.5.1 Serial Channel (SIO/UART)

29.5.1.1 AC Measurement Condition

The AC characteristics data of this chapter is measured under the following conditions.

- Output levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Input levels: High = 0.75 × DVDD3, Low = 0.25 × DVDD3
- Load capacity: CL = 30pF

29.5.1.2 AC Electrical Characteristics (I/O Interface Mode)

In the table below, the letter x represents the SIO operation clock SCLK Clock Low width (input) cycle time which is identical to the fc cycle time.

(1) SCLK input mode

[Data Input]

Parameter	Symbol	Equation		fsys = 60 MHz		fsys = 120MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK Clock High width (input)	t_{SCH}	4x	–	66.7	–	33.3	–	ns
SCLK Clock Low width (input)	t_{SCL}	4x	–	66.7	–	33.3	–	
SCLK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	–	133.4	–	66.7	–	
Valid Data Input ← SCLK rise or fall (Note 1)	t_{SRD}	30	–	30	–	30	–	
SCLK rise or fall (Note 1) → Input Data hold	t_{HSR}	x + 30	–	46.7	–	38.3	–	

[Data Output]

Parameter	Symbol	Equation		fsys = 60 MHz		fsys = 120 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK Clock High width (input)	t_{SCH}	4x	–	95 (Note3)	–	70.0 (Note3)	–	ns
SCLK Clock Low width (input)	t_{SCL}	4x	–	95 (Note3)	–	70.0 (Note3)	–	
SCLK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	–	190	–	140	–	
Output Data → SCLK rise or fall (Note 1)	t_{OSS}	$t_{SCY}/2 - 3x - 45$	–	0 (Note2)	–	0 (Note2)	–	
SCLK rise or fall → Output Data hold (Note 1)	t_{OHS}	$t_{SCY}/2$	–	95	–	70.0	–	

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

[Data Input]

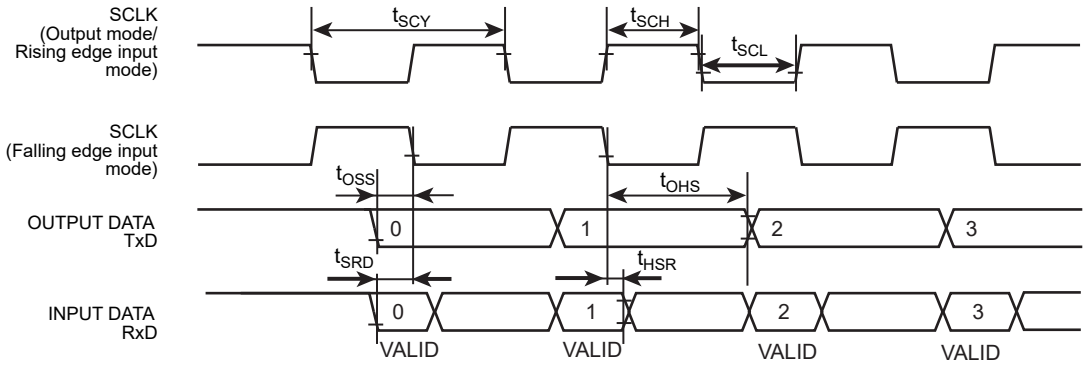
Parameter	Symbol	Equation		fsys = 60 MHz		fsys = 120 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK cycle (programmable)(note3)	t_{SCY}	2x	–	100 (note1)		100 (note1)	–	ns
Valid Data Input → SCLK rise	t_{SRD}	45	–	45	–	45	–	
SCLK rise → Input Data hold	t_{HSR}	0	–	0	–	0	–	

Note 1: This value is the minimum value of the SCLK cycle in the range where $t_{SCY} \geq t_{SRD} \times 2$.

[Data Output]

Parameter	Symbol	Equation		fsys = 60 MHz		fsys = 120 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK cycle (programmable)(note3)	t_{SCY}	2x	–	66.7 (note1)	–	66.7 (note1)	–	ns
Output Data → SCLK rise	t_{OSS}	$t_{SCY}/2$ - 30	–	3.4	–	3.4	–	
SCLK rise → Output Data hold	t_{OHS}	$t_{SCY}/2$ - 30	–	3.4	–	3.4	–	

Note 1: This value is the minimum value of the SCLK cycle in the range where $t_{OSS} = t_{OHS} \geq 0$.



29.5.2 I2C Interface (I2C)

29.5.2.1 AC Measurement Condition

The AC characteristics data of this chapter is measured under the following conditions.

- Output levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Input levels: High = 0.75 × DVDD3, Low = 0.25 × DVDD3
- Load capacity: CL = 30pF

29.5.2.2 AC Electrical Characteristics

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.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 I2CxCR1.

p denotes the value of p programmed into the PRSCK(dividing clock select)field in the I2CxPRS.

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 START condition	$t_{HD;\ STA}$	–	–	4.0	–	0.6	–	μs
SCL Low width (Input) (Note 1)	t_{LOW}	–	–	4.7	–	1.3	–	μs
SCL High width (Input) (Note 2)	t_{HIGH}	–	–	4.0	–	0.6	–	μs
Setup time for a repeated START condition	$t_{SU;\ STA}$	(Note 5)	–	4.7	–	0.6	–	μs
Data hold time (Input) (Note 3, 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}	(Note 5)	–	4.7	–	1.3	–	μs

Note 1: SCL clock Low width (output): $p(2^{n+1} + 10)/x$

Note 2: SCL clock High width (output): $p(2^{n+1} + 6)/x$

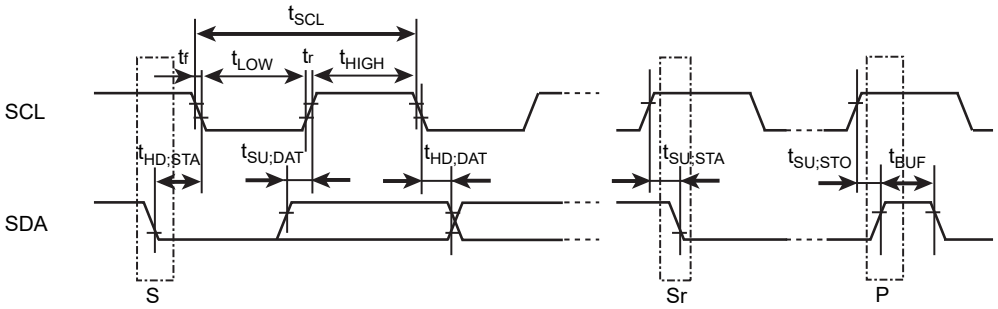
On I2C-bus specification, maximum Speed of Standard Mode/fast mode is 100kHz/400khz. Internal SCL Frequency setting should comply with fsys and Note1 & Note2 shown above.

Note 3: The output data hold time is equal to 4 cycle of Prescaler clock(Tprsck) from the edge 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 I2C 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 I2C does not satisfy this requirement.



S: Start condition
Sr: Re-start condition
P: Stop condition

29.5.3 Synchronous serial Interface (SSP)

29.5.3.1 AC measurement conditions

The letter "T" used in the equations in the table represents the period of internal bus frequency (fsys).

- Output levels: High = 0.7 × DVDD3, Low = 0.3 × DVDD3
- Input levels: High = 0.9 × DVDD3, Low = 0.1 × DVDD3

29.5.3.2 AC Electrical Characteristics

Baud rate clock is set below condition.

Master Mode

m=(<CPSDVSR>×(1+<SCR>))= fsys / f_{SPxCLK}

<CPSDVR> is set only even number. 65024 ≥ m ≥ 2

Slave mode

n=(<CPSDVSR>×(1+<SCR>))= fsys / f_{SPxCLK} 65024 ≥ n ≥ 12

10MHz Type @ch1/2

Parameter	Symbol	Equation		fsys=60MHz (m=6, n=18)		fsys=120MHz (m=12, n=36)		Unit
		Min	Max	Min	Max	Min	Max	
SPxCLK Period (Master)	T_m	(m)T However more than 100ns	–	100 (10MHz)	–	100 (10MHz)	-	ns
SPxCLK Period (Slave)	T_s	(n)T However more than 300ns	–	300 (3.3MHz)	–	300 (3.3MHz)	–	ns
SPxCLK rise up time	t_r	–	15	–	15	–	15	ns
SPxCLK fall down time	t_f	–	15	–	15	–	15	ns
Master mode: SPxCLK low level pulse width	t_{WLM}	(m)T/2 - 15	–	35	–	35	–	ns
Master mode: SPxCLK high level pulse width	t_{WHM}	(m)T/2 - 15	–	35	–	35	–	ns
Slave mode: SPxCLK low level pulse width	t_{WLS}	(n)T/2 - 15	–	135	–	135	–	ns
Slave mode: SPxCLK high level pulse width	t_{WHS}	(n)T/2 - 15	–	135	–	135	–	ns
Master mode: SPxCLK rise/fall → output data valid	t_{ODSM}	–	15	–	15	–	15	ns
Master mode: SPxCLK rise/fall → output data hold	t_{ODHM}	(m)T/2 - 13	–	35	–	35	–	ns
Master mode: SPxCLK rise/fall → input data valid delay time	t_{IDSM}	30	–	30	–	30	–	ns
Master mode: SPxCLK rise/fall → input data hold	t_{IDHM}	0	–	0	–	0	–	ns
Master mode: SPxFSS valid → SPxCLK rise/fall	t_{OFSM}	(m)T - 15	(m)T + 15	85	115	85	115	ns
Slave mode: SPxCLK rise/fall → output data valid delay time	t_{ODSS}	–	(3T) + 40	–	90	–	65	ns
Slave mode: SPxCLK rise/fall → output data hold	t_{ODHS} (Note1)	(n)T/2 + (2T)	–	183.3	–	166.7	–	ns
Slave mode: SPxCLK rise/fall → input data valid delay time	t_{IDSS}	10	–	10	–	10	–	ns
Slave mode: SPxCLK rise/fall → input data hold	t_{IDHS}	(3T) + 15	–	65	–	40	–	ns
Slave mode: SPxFSS valid → SPxCLK rise/fall	t_{OFSS}	(n)T + 10	–	310	–	310	–	ns

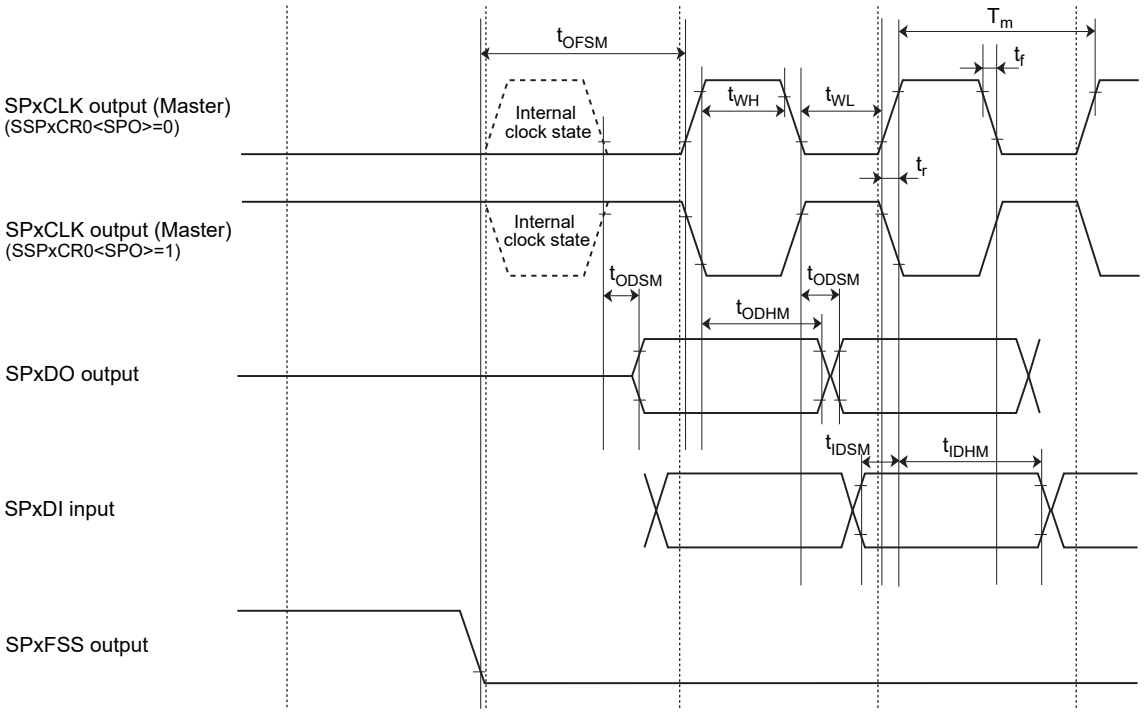
20MHz Type @ch0

Parameter	Symbol	Equation		fsys=60MHz (m=4, n=12)		fsys=120MHz (m=6, n=18)		Unit
		Min	Max	Min	Max	Min	Max	
SPxCLK Period (Master)	T_m	(m)T However more than 50ns	–	66.6 (15MHz)	–	50 (20MHz)	-	ns
SPxCLK Period (Slave)	T_s	(n)T However more than 150ns	–	200 (5MHz)	–	150 (6.6MHz)	–	ns
SPxCLK rise up time	t_r	–	10	–	10	–	10	ns
SPxCLK fall down time	t_f	–	10	–	10	–	10	ns
Master mode: SPxCLK low level pulse width	t_{WLM}	(m)T/2 - 10	–	23.3	–	15	–	ns
Master mode: SPxCLK high level pulse width	t_{WHM}	(m)T/2 - 10	–	23.3	–	15	–	ns
Slave mode: SPxCLK low level pulse width	t_{WLS}	(n)T/2 - 10	–	90	–	65	–	ns
Slave mode: SPxCLK high level pulse width	t_{WHS}	(n)T/2 - 10	–	90	–	65	–	ns
Master mode: SPxCLK rise/fall → output data valid	t_{ODSM}	–	10	–	10	–	10	ns
Master mode: SPxCLK rise/fall → output data hold	t_{ODHM}	(m)T/2 - 10	–	23.3	–	15	–	ns
Master mode: SPxCLK rise/fall → input data valid delay time	t_{IDSM}	15	–	15	–	15	–	ns
Master mode: SPxCLK rise/fall → input data hold	t_{IDHM}	0	–	0	–	0	–	ns
Master mode: SPxFSS valid → SPxCLK rise/fall	t_{OFSM}	(m)T - 15	(m)T + 15	51.6	81.6	35	65	ns
Slave mode: SPxCLK rise/fall → output data valid delay time	t_{ODSS}	–	(3T) + 30	–	80.0	–	55	ns
Slave mode: SPxCLK rise/fall → output data hold	t_{ODHS}	(n)T/2 + (2T)	–	133.3	–	91.7	–	ns
Slave mode: SPxCLK rise/fall → input data valid delay time	t_{IDSS}	10	–	10	–	10	–	ns
Slave mode: SPxCLK rise/fall → input data hold	t_{IDHS}	(3T) + 15	–	65	–	40	–	ns
Slave mode: SPxFSS valid → SPxCLK rise/fall	t_{OFSS}	(n)T + 10	–	210	–	160	–	ns

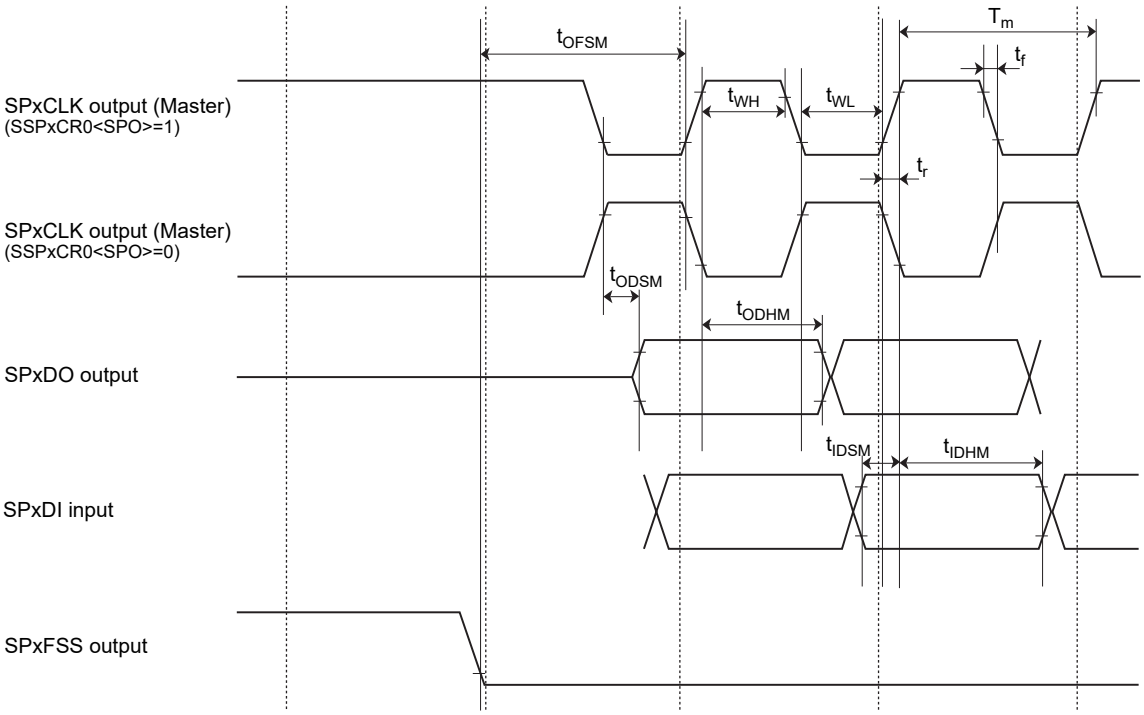
SSP SPI mode (Master)

- fsys ≥ 2 × f_{SPxCLK} (Max)
- fsys ≤ 65024 × f_{SPxCLK} (Min)

(1)Master SSPCR0<SPH>="0"(Data is latched on the first edge.)



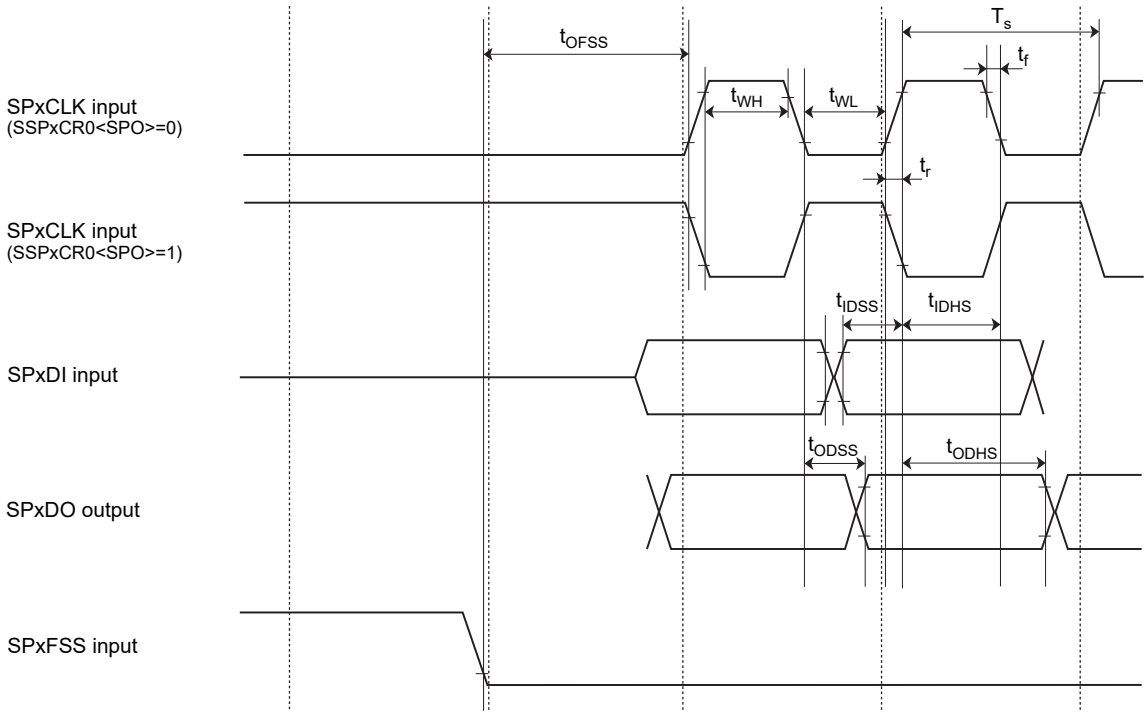
(2) Master SSPCR0<SPH>="1" (2nd Data is latched on the second edge.)



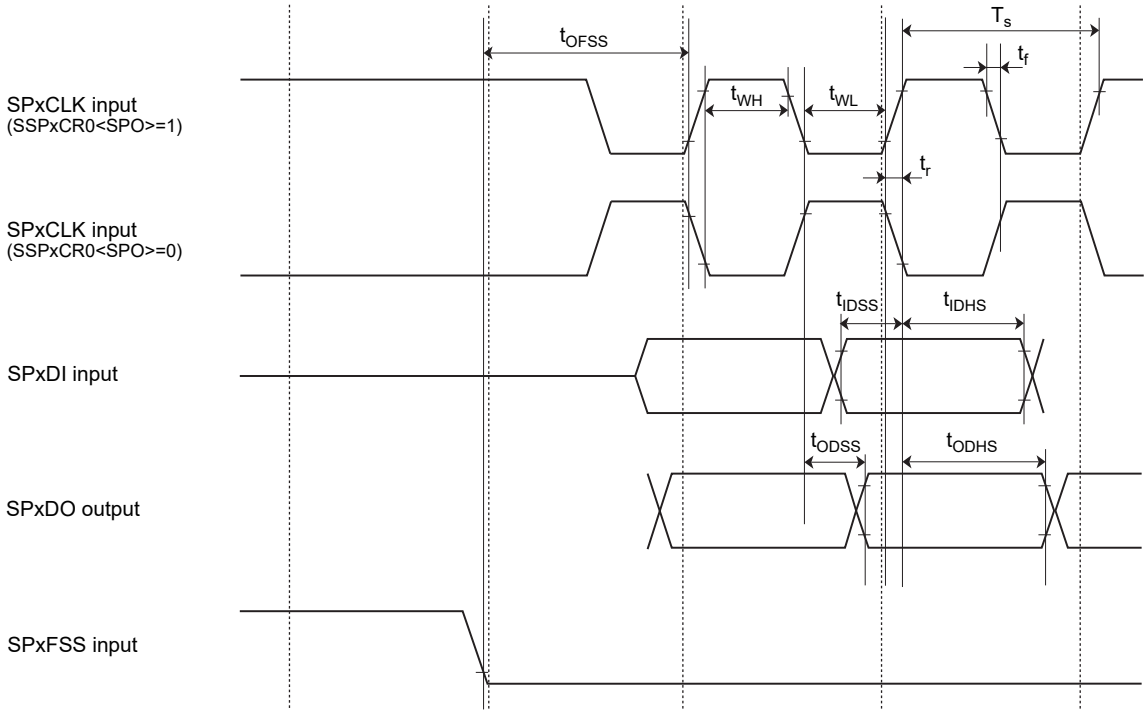
SSP SPI mode(Slave)

- fsys ≥ 12 × f_{SPxCLK} (Max)
- fsys ≤ 65024 × f_{SPxCLK} (Min)

(3) Slave SSPCR0<SPH>="0"(Data is latched on the first edge.)



(4) Slave SSPCR0<SPH> = "1" (Data is latched on the second edge.)



29.5.4 External Bus Interface AC Characteristics

29.5.4.1 AC Measurement Condition

- DVDD3 = 2.7 to 3.6V
- Output levels: High = 0.7 × DVDD3,Low = 0.3 × DVDD3
- Input levels: High = 0.7 × DVDD3,Low = 0.3 × DVDD3
- Load capacitance: CL = 30pF

29.5.4.2 Variable condition

- ALE : Conditional variable (ALE = 1 + n ; n = 0, 1, 2, 4)
- RWS :Number of setup cycle insertion before $\overline{\text{RD}}$, WR asserted (RWS = 0, 1, 2, 4)
- TW : Number of internal wait insertion (TW = 0 to 15)
- RWH: Number of RD, WR hold cycle insertion (RWH = 0 to 6 or 8)
- CSH: Number of CSx hold cycle insertion (CSH = 0, 1, 2 or 4)

29.5.4.3 AC Characteristics (BCLK asynchronous mode multiplex Bus mode)

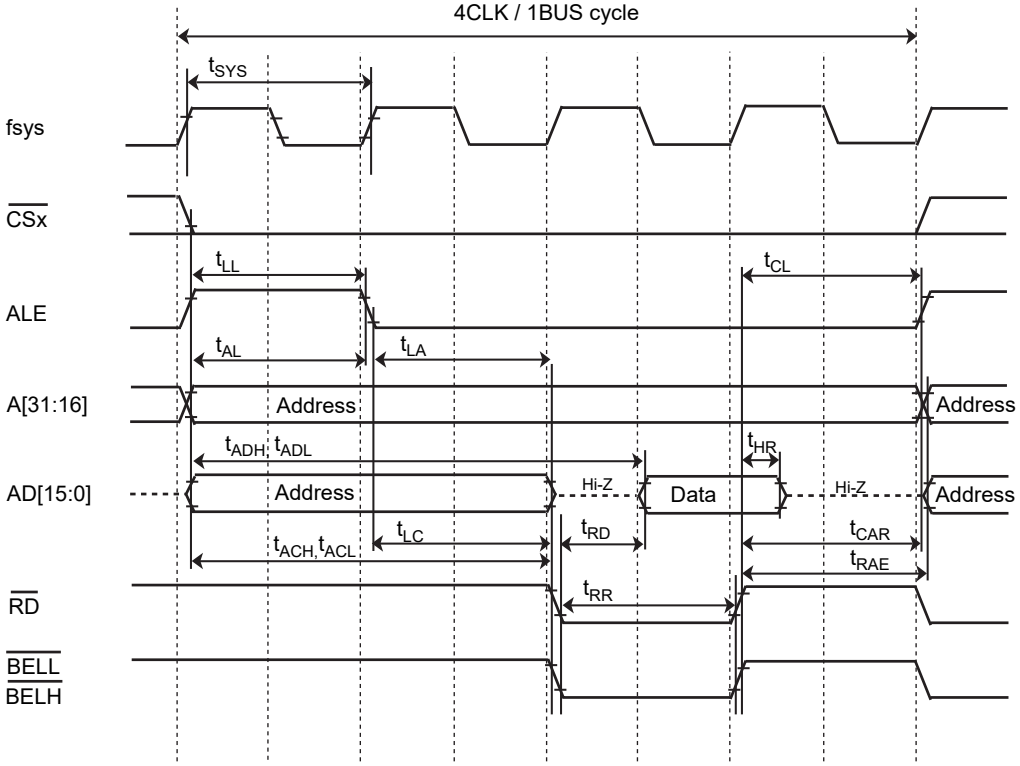
Conditional variable (60MHz) : ALE=1 ,RWS = 1 ,TW = 3, RWH = 1, CSH =1

Conditional variable (120MHz) : ALE=3,RWS = 3, TW = 7, RWH = 3, CSH =2

Parameter	Sym-bol	Equation		fsys = 80MHz		fsys = 120MHz		Unit
		Min	Max	Min	Max	Min	Max	
System clock period (x)	t_{SYS}	x	–	16.7	–	8.4	–	ns
A[31:0] valid →ALE negated	t_{AL}	x (1+ALE)-15	–	18.4	–	18.4	–	
ALE negated →A[31:0] hold	t_{LA}	x (1+RWS)-10	–	23.4	–	23.4	–	
ALE high pulse width	t_{LL}	x (1+ALE)-15	–	18.4	–	18.4	–	
ALE negated →$\overline{RD}$ or $\overline{WR}$ asserted	t_{LC}	x (1+RWS)-10	–	23.4	–	23.4	–	
$\overline{RD}$ or $\overline{WR}$ negated → ALE asserted	t_{CL}	x (1+RWH+CSH)-15	–	35.0	–	35.0	–	
A[15:0] valid →$\overline{RD}$ or $\overline{WR}$ asserted A[23:16] valid →$\overline{RD}$ or $\overline{WR}$ asserted	t_{ACL} t_{ACH}	x (2+ALE+RWH)-15	–	51.7	–	51.7	–	
$\overline{RD}$ or $\overline{WR}$ negated →A[31:16] hold	t_{CAR}	x (1+RWH+CSH)-15	–	35.0	–	35.0	–	
A[15:0] valid →D[15:0] input A[31:16] valid →D[15:0] input	t_{ADL} t_{ADH}	–	x (3+ALE+RWS +TW)-35	–	98.4	–	98.4	
$\overline{RD}$ asserted →D[15:0] input	t_{RD}	–	x (1+TW)-30	–	36.7	–	36.7	
$\overline{RD}$ low pulse width	t_{RR}	x (1+TW)-15	–	51.7	–	51.7	–	
$\overline{RD}$ negated →D[15:0] hold	t_{HR}	0	–	0	–	0	–	
$\overline{RD}$ negated →A[31:0] output	t_{RAE}	x (1+RWH+CSH)-15	–	35	–	35	–	
$\overline{WR}$ low pulse width	t_{WW}	x (1+TW)-15	–	51.7	–	51.7	–	
D[15:0] valid →$\overline{WR}$ negated	t_{DW}	x (1+TW)-15	–	51.7	–	51.7	–	
$\overline{WR}$ negated →D[15:0] hold	t_{WD}	x (1+RWH)-10	–	23.4	–	23.4	–	

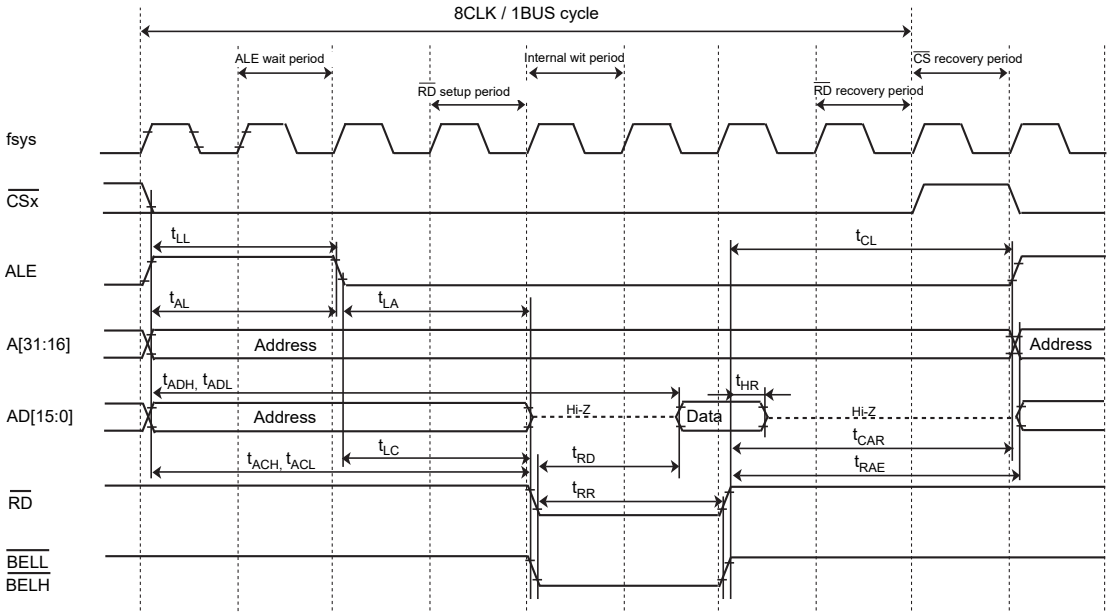
1. Read cycle timing (minimum cycle)

(Neither Cycle expander, ALE wait, RD setup, Internal wait, CS recovery nor RD recovery function are used.)



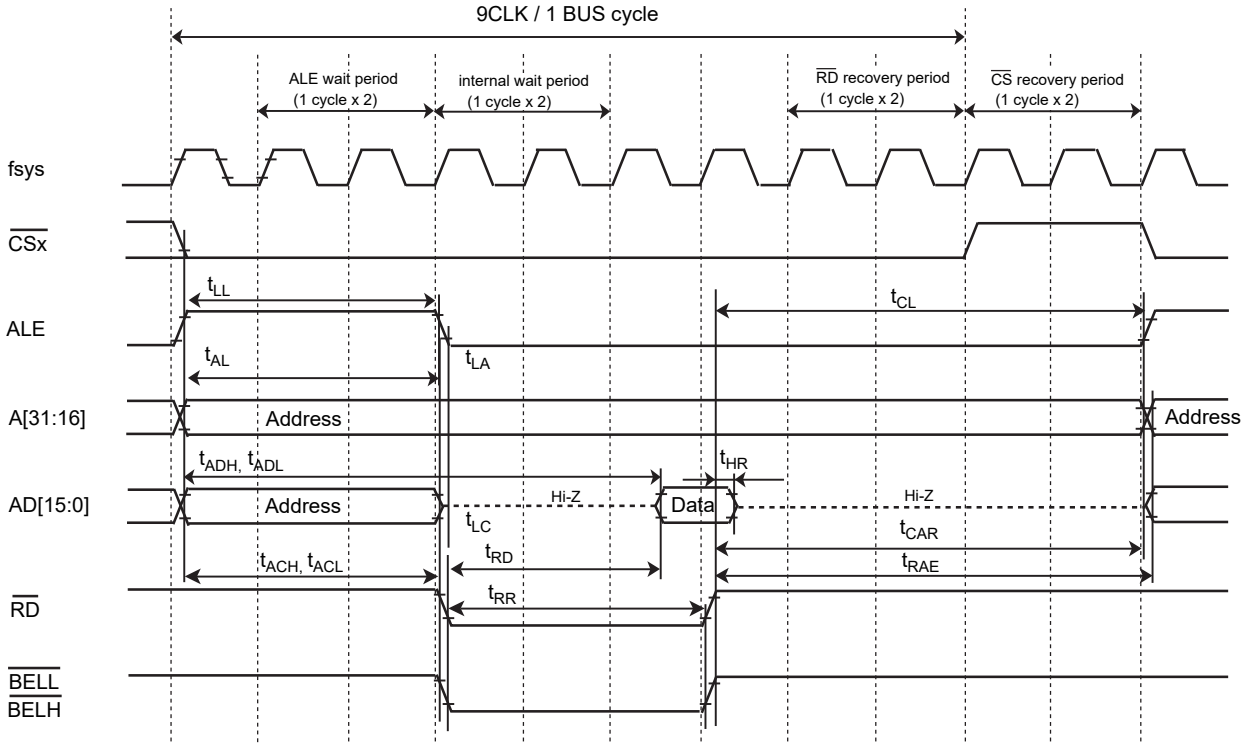
2. Read cycle timing (1 bus cycle per 8 clock)

(ALE wait, RD setup, Internal wait, CS recovery and RD recovery function are set to 1 cycle though Cycle expander function is not used.)



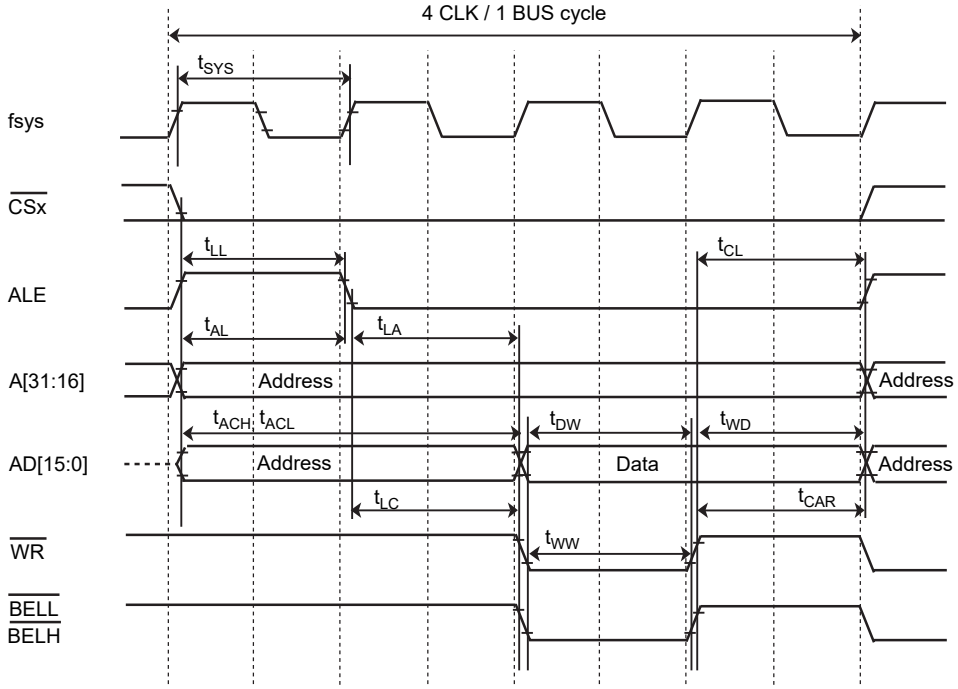
3. Read cycle timing (1 bus cycle per 9 clock)

(ALE wait, RD setup, Internal wait, CS recovery and RD recovery function are set to 1 cycle though Cycle expander function is set double.)



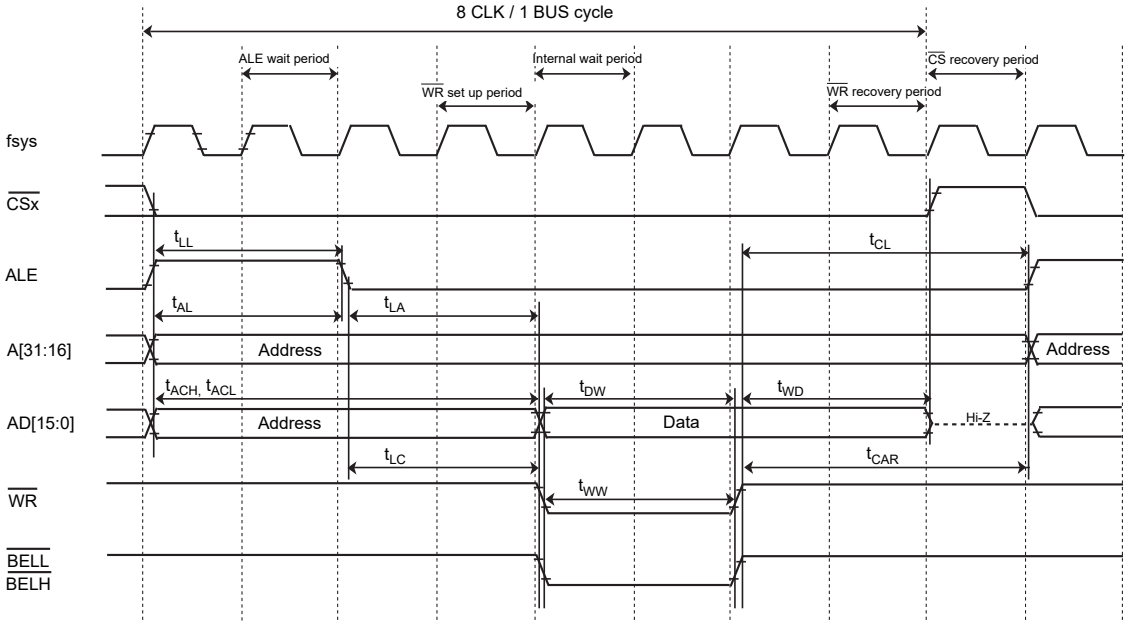
4. Write cycle timing (minimum bus cycle)

(Neither Cycle expander, ALE wait, WR setup, Internal wait, CS recovery nor WR recovery function are used.)



5. Write cycle timing (1 bus cycle per 8 clock)

(ALE wait, WR setup, Internal wait, CS recovery and WR recovery function are set to 1 cycle though Cycle expander function is not used.)



29.5.5 SLC NAND Flash Controller AC Characteristics

29.5.5.1 AC Measurement Condition

- DVDD3 = 2.7 to 3.6V
- Output levels: High = 0.5 × DVDD3,Low = 0.5 × DVDD3
- Input levels: High = 0.5 × DVDD3,Low = 0.5 × DVDD3
- Load capacitance: CL = 30pF

29.5.5.2 Variable condition

- CLES :Sets the setup clock of SNFCCLE. (CLES=n; n=0, 1, 2, or 3)
- CLEH :Sets the hold clock of SNFCCLE. (CLEH=n; n=0, 1, 2, or 3)
- ALES :Sets the setup clock of SNFCALE. (ALES = n; n=0, 1, 2, or 3)
- ALEH :Sets the hold clock of SNFCALE. (ALEH = n; n=0, 1, 2, or 3)
- WES :Sets the setup clock of $\overline{\text{SNFCWE}}$. (WES = 1 + n; n = 0, 1, 2, or 3)
- WEW :Sets the wait clock of $\overline{\text{SNFCWE}}$. (WEW = 1 + n; n = 0, 1, 2, 3, 4, 5, 6, or 7)
- WEH :Sets the hold clock of $\overline{\text{SNFCWE}}$. (WEH = n; n = 0, 1, 2, 3, 4, 5, 6, or 7)
- RES :Sets the setup clock of $\overline{\text{SNFCRE}}$. (RES = 1 + n; n = 0, 1, 2, or 3)
- REW :Sets the wait clock of $\overline{\text{SNFCRE}}$. (REW = 1 + n; n = 0, 1, 2, 3, 4, 5, 6, or 7)
- REH :Sets the hold clock of $\overline{\text{SNFCRE}}$. (REH = n; n = 0, 1, 2, 3, 4, 5, 6, or 7)
- DMYA :Selects the dummy period. (DMYA=n; n=0, 1, 2, or 3)
- DMYC1 :Sets the number of clocks for the dummy period 1. (DMYC1 = 1 + n; n = 0, 1, 2, 3, 4, 5, 6, or 7)
- DMYB :Selects the wait period. (DMYB=n; n=0, 1, 2, or 3)
- DMYC2 :Sets the number of clocks for the wait period 2. (DMYC2 = 1 + n; n = 0, 1, 2, 3, 4, 5, 6, or 7)

29.5.5.3 AC Characteristics

- Conditional variable (60MHz) :

  CLES = 0, CLEH = 0, ALES = 0, ALEH = 0,

  WES = 0, WEW = 1, WEH = 1, RES = 0, REW = 2, REH = 0,

  DMYA = 0, DMYC1 = 0, DMYB = 1, DMYC2 = 3
- Conditional variable (120MHz) :

  CLES = 0, CLEH = 1, ALES = 0, ALEH = 1,

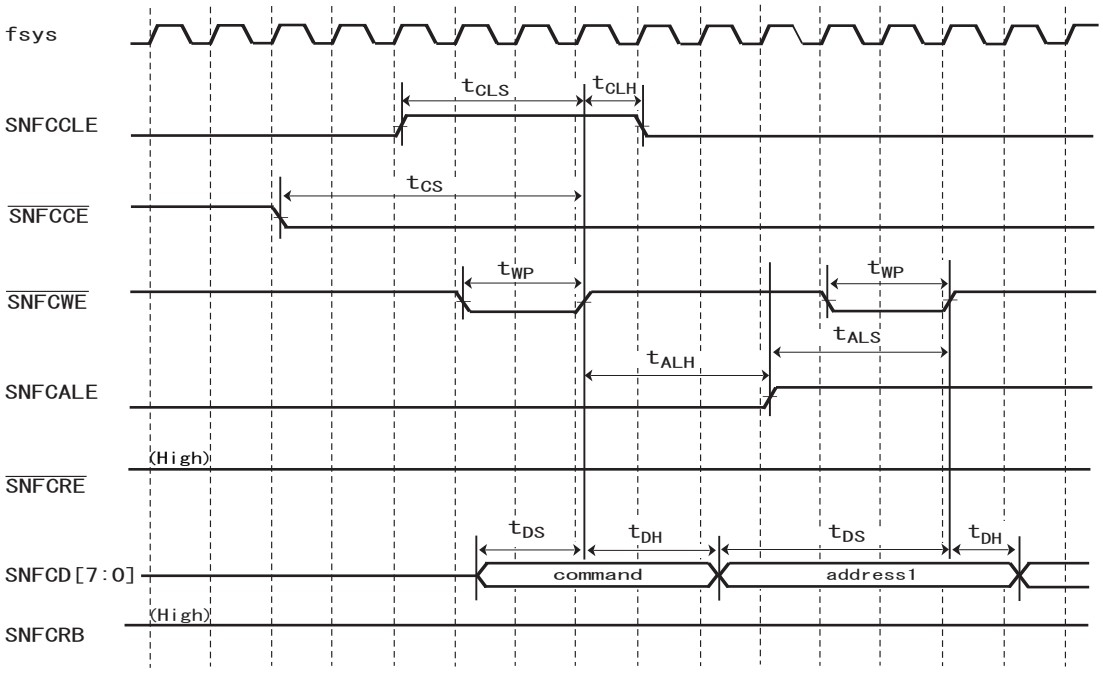
  WES = 0, WEW = 2, WEH = 2, RES = 0, REW = 4, REH = 1,

  DMYA = 0, DMYC1 = 0, DMYB = 1, DMYC2 = 5

Parameter	Sym-bol	Equation	fsys = 60MHz		fsys = 120MHz		Unit
			Min	Max	Min	Max	
System clock period (x)	t_{SYS}	x	16.7	–	8.3	–	ns
$\overline{\text{SNFCCE}}$ setup time‘	t_{CS}	x (3+CLES+WES+WEW) - 15	51.7	–	26.7	–	
SNFCCLE setup time‘	t_{CLS}	x (2+WES+WEW) - 13	37.0	–	20.3	–	
SNFCCLE hold time	t_{CLH}	x (WEH) - 5	11.7	–	11.7	–	
SNFCALE setup time	t_{ALS}	(ALE ↑→ WE ↑) : x (2+WES+WEW) - 13	37.0	–	20.3	–	
		(ALE ↓→ WE ↑) : x (2+ALEH+CLES+WES+WEW) - 13	37.0	–	28.7	–	
SNFCALE hold time	t_{ALH}	(WE ↑→ ALE ↑) : x (WEH+CLEH+ALE) - 5	11.7	–	20.0	–	
		(WE ↑→ ALE↓) : x (WEH) - 5	11.7	–	11.7	–	
$\overline{\text{SNFCWE}}$ cycle time	t_{WC}	x (2+WES+WEW+WEH)	66.7	–	50.0	–	
$\overline{\text{SNFCWE}}$ low level pulse width	t_{WP}	x (1+WEW) - 6	27.3	–	19.0	–	
$\overline{\text{SNFCWE}}$ high level pulse width	t_{WH}	x (1+WEH+WES) - 6.6	26.7	–	18.4	–	
write data setup time	t_{DS}	x (1+WEW) - 8	25.3	–	17.0	–	
write data hold time‘	t_{DH}	x (WEH) - 5	11.7	–	11.7	–	
$\overline{\text{SNFCRE}}$ cycle time	t_{RC}	x (2+RES+REW+REH)	66.7	–	58.3	–	
$\overline{\text{SNFCRE}}$ low level pulse width	t_{RP}	x (1+REW) - 5	45.0	–	36.7	–	
$\overline{\text{SNFCRE}}$ high level pulse width	t_{REH}	x (1+REH+RES) - 6.6	10.1	–	10.1	–	
SNFCALE negated → $\overline{\text{SNFCRE}}$ asserted	t_{AR}	x (1+ALEH+RES) - 6.6	10.1	–	10.1	–	
$\overline{\text{SNFCWE}}$ negated → $\overline{\text{SNFCRE}}$ asserted	t_{WHR}	x (2+WEH+CLEH+(DMYC2)*DMYB+RES) - 15	85.0	–	68.3	–	
read data setup time	t_{RES}	–	–	16.7	–	16.7	
read data hold time	t_{RHOH}	–	0.0	–	0.0	–	
SNFCRB negated → $\overline{\text{SNFCRE}}$ asserted asserted	t_{RR}	x (<1+DMYC1>*<DMYA>) + 4 +<RES>) - 5	61.7	–	28.3	–	
$\overline{\text{SNFCRE}}$ access time	t_{REA}	x (1+REW) - 21.6	–	28.4	–	20.1	

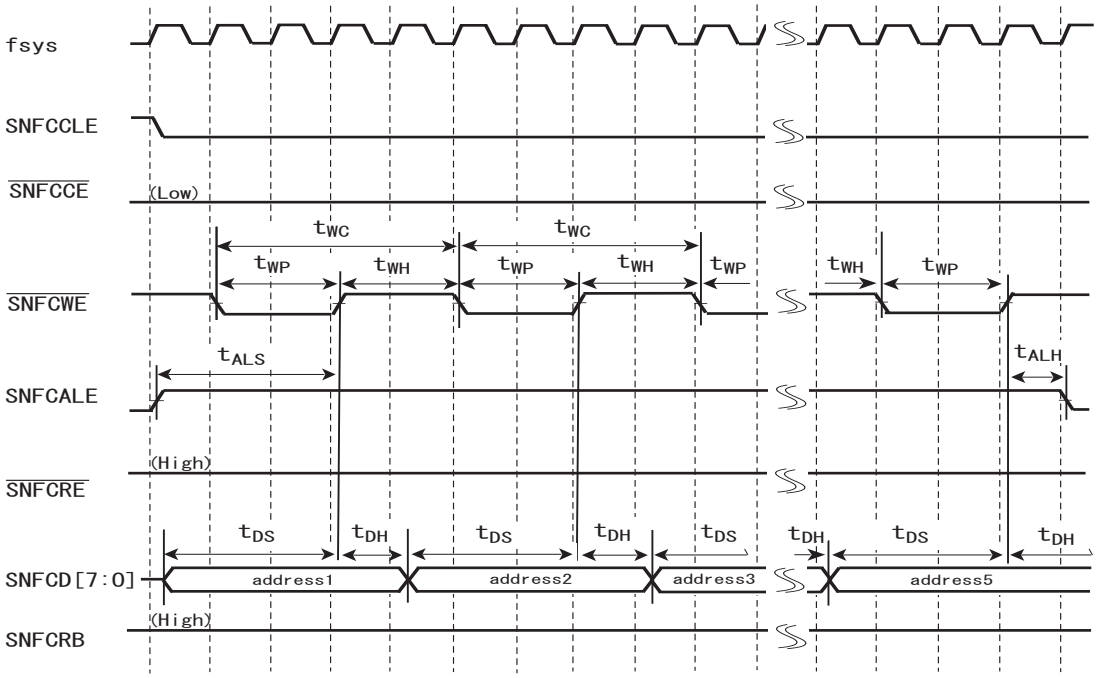
1. Command timing

Conditional variable : CLES = 1, CLEH = 1, ALES = 1, WES = 0, WEW = 1, WEH = 1



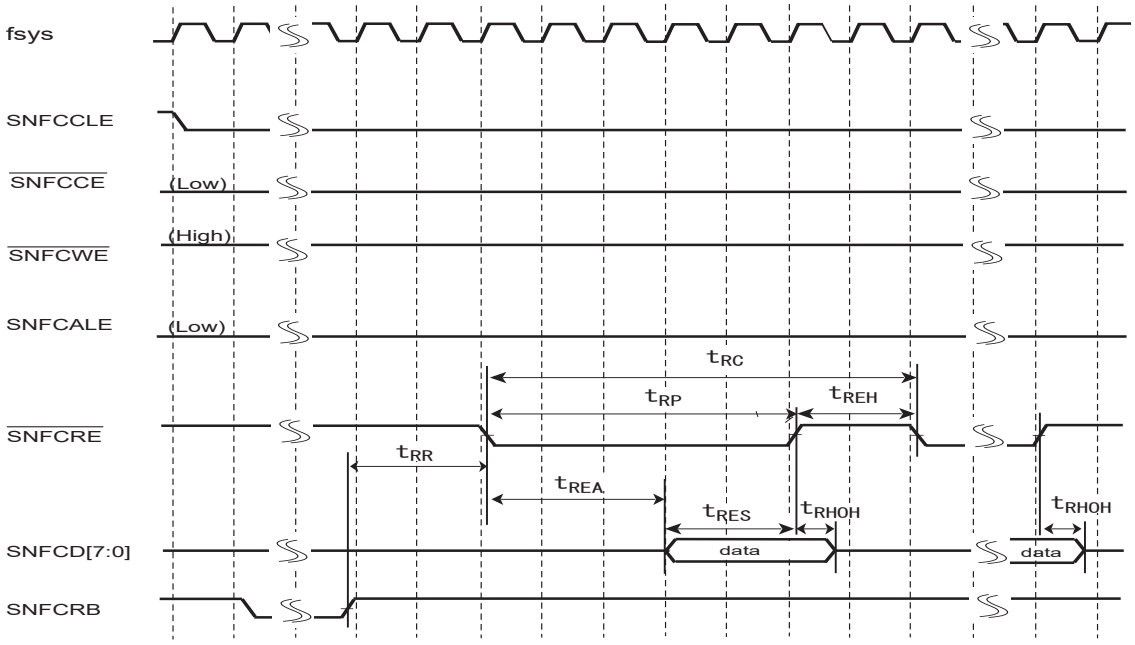
2. Address timing

Conditional variable : CLEH = 1, ALES = 1, ALEH = 2, WES = 0, WEW = 0, WEH = 0



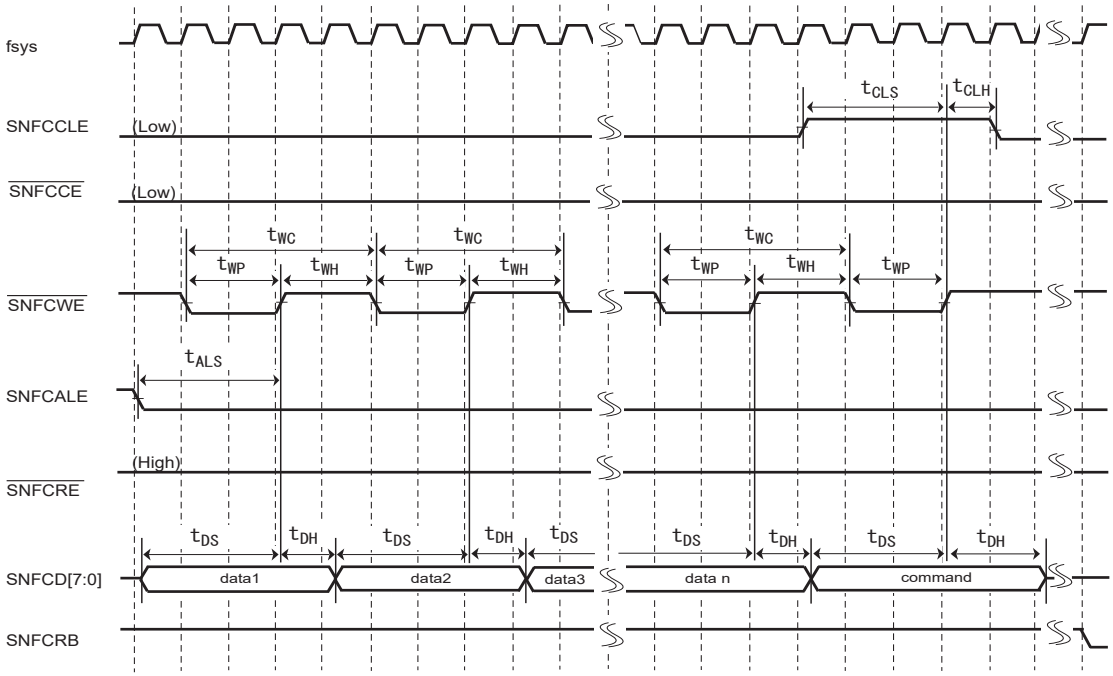
3. Lead timing

Conditional variable : ALEH =1 RES = 0, REW = 0, REH = 0, DMYA = 1, DMYC1 = 0



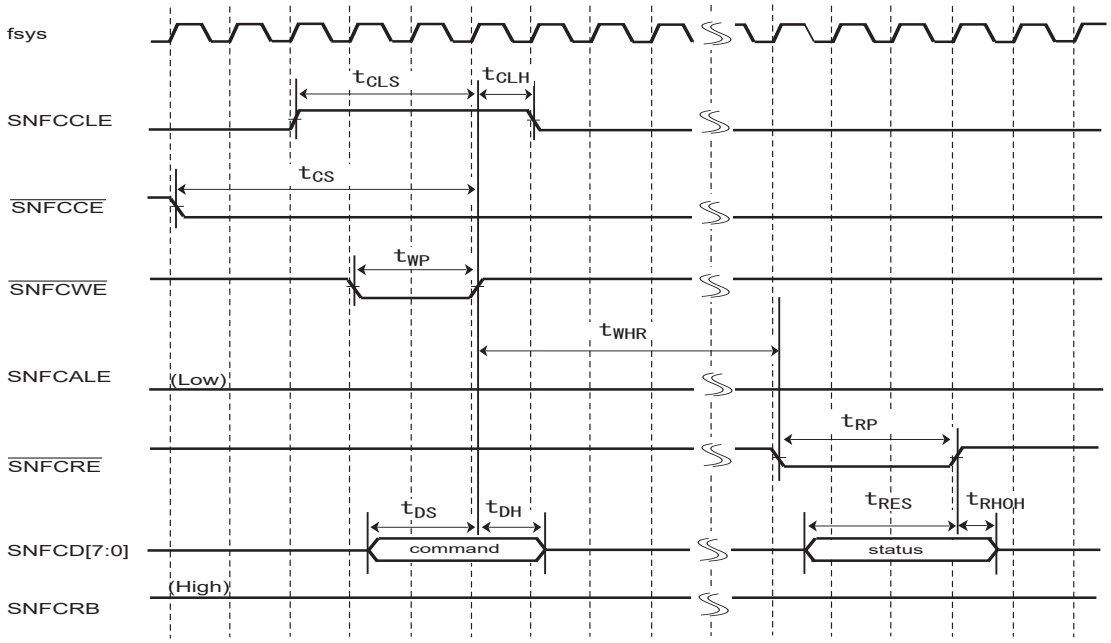
4. Write timing

Conditional variable : CLES = 1, ALEH = 0, WES = 0, WEW = 0, WEH = 0



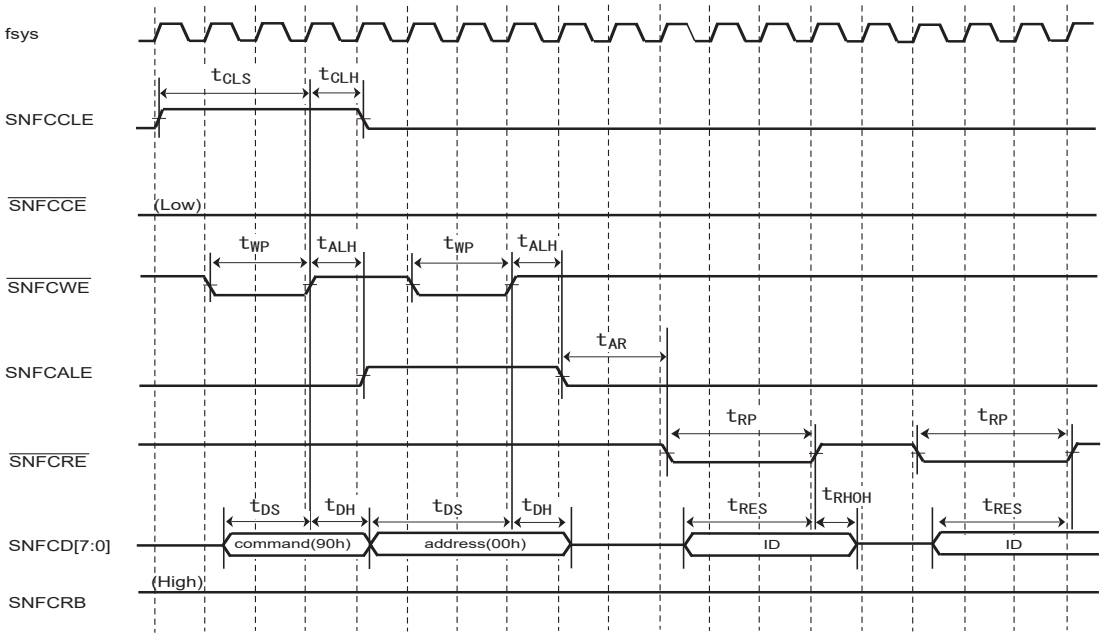
5. Status read timing

Conditional variable : CLES = 1, CLEH = 1, WES = 0, WEW = 1, WEH = 1,
RES = 0, REW = 1, REH = 1, DMYB = *, DMYC2 = *



6. ID read timing

Conditional variable : CLES = *, CLEH = 0, ALES = 0, ALEH = 1,
WES = 0, WEW = 1, WEH = 1, RES = 0, REW = 1, REH =0



29.5.6 16-bit Timer / Event counter (TMRB)

29.5.6.1 Event Counter

(1) AC Measurement Condition

The AC characteristics data of this chapter is measured under the following conditions.

- Input levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Load capacity: CL = 30pF

(2) AC Electrical Characteristics

In the table below, the letter x represents the TMRB operation clock cycle time. TMRB use the clock cycle as same as one of fsys. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		60 MHz		120 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Clock low pulse width	t_{VCKL}	2x + 100	–	133.3	–	116.7	–	ns
Clock high pulse width	t_{VCKH}	2x + 100	–	133.3	–	116.7	–	

29.5.6.2 Capture

(1) AC Measurement Condition

The AC characteristics data of this chapter is measured under the following conditions.

- Input levels: High = 0.75 × DVDD3, Low = 0.25 × DVDD3
- Load capacity: CL = 30pF

(2) AC Electrical Characteristics

In the table below, the letter x represents the TMRB operation clock cycle time.TMRB use the clock cycle as same as one of fsys. It varies depending on the programming of the clock gear function

Parameter	Symbol	Equation		60 MHz		120 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Low pulse width	t_{CPL}	2x + 100	–	133.3	–	116.7	–	ns
High pulse width	t_{CPH}	2x + 100	–	133.3	–	116.7	–	

29.5.7 External Interrupt

29.5.7.1 AC Measurement Condition

The AC characteristics data of this chapter is measured under the following conditions.

- Input levels: High = 0.75 × DVDD3, Low = 0.25 × DVDD3
- Load capacity: CL = 30pF

29.5.7.2 AC Electrical Characteristics

In the table below, the letter x represents the fsys cycle time.

1. Except STOP1 and STOP2 release interrupts

Parameter	Symbol	Equation		fsys = 60 MHz		fsys = 120 MHz		Unit
		Min	Max	Min	Max	Min	Max	
low-level pulse width	t_{INTAL}	x + 100	–	116.6	–	108.3	–	ns
high-level pulse width	t_{INTAH}	x + 100	–	116.6	–	108.3	–	ns

2. STOP1 and STOP2 release interrupts

Parameter	Symbol	Equation		fsys = 60 MHz		fsys = 120 MHz		Unit
		Min	Max	Min	Max	Min	Max	
INT0 to 15 low-level pulse width	t_{INTBL}	100	–	500	–	500	–	ns
INT0 to 15 high-level pulse width	t_{INTBH}	100	–	500	–	500	–	ns

29.5.8 ADC Trigger Input pin AC Characteristics

29.5.8.1 AC Measurement condition

The AC characteristics data of this chapter is measured under the following conditions.

- Input levels: High = 0.75 × DVDD3, Low = 0.25 × DVDD3
- Load capacity: CL = 30pF

29.5.8.2 AC Electrical Characteristics

In the table below, the letter x represents the fc cycle time.

Parameter	Symbol	Formula		fsys = 60 MHz		fsys = 120 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Low-level pulse width	T_{adl}	2 x + 20	–	53.3	–	36.7	–	ns
High-level pulse interval	T_{adh}	2 x + 20	–	53.3	–	36.7	–	

29.5.9 SCOUT pin AC Characteristics

29.5.9.1 AC Measurement condition

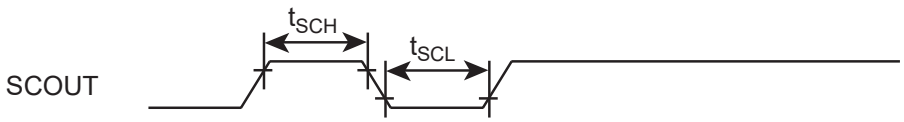
The AC characteristics data of this chapter is measured under the following conditions.

- Output levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Load capacity: CL = 30pF(SCOUT)

29.5.9.2 AC Electrical Characteristics

In the below table, the letter T represents the cycle time of the SCOUT output clock.

Parameter	Symbol	Equation		When the frequency of SCOUT is 25MHz		When the frequency of SCOUT is 30MHz		Unit
		Min	Max	Min	Max	Min	Max	
High-level pulse width	t_{SCH}	0.5T − 8	–	12	–	8.6	–	ns
Low-level pulse width	t_{SCL}	0.5T − 8	–	12	–	8.6	–	ns



29.5.10 Debug Communication

29.5.10.1 AC Measurement Condition

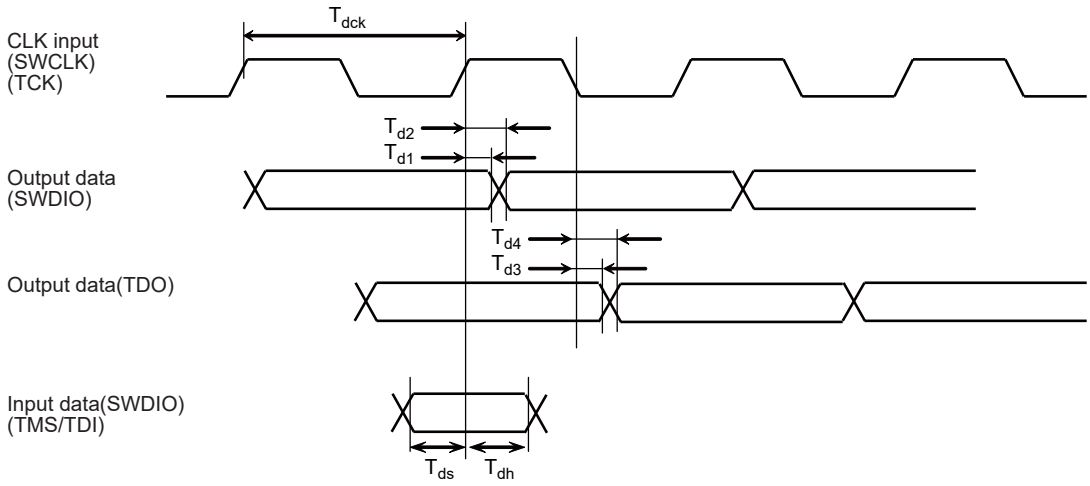
- Output levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Input levels: Low = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Load capacitance: CL = 30pF (SWDIO, TRACECLK, TRACEDATA0 to 3)

29.5.10.2 SWD Interface

Parameter	Symbol	Min	Max	Unit
CLK cycle	T_{dck}	100	–	ns
CLK rise → Output data hold	T_{d1}	4	–	
CLK rise → to output data valid	T_{d2}	–	30	
Input data valid → CLK rise	T_{ds}	20	–	
CLK rise → Input data hold	T_{dh}	15	–	

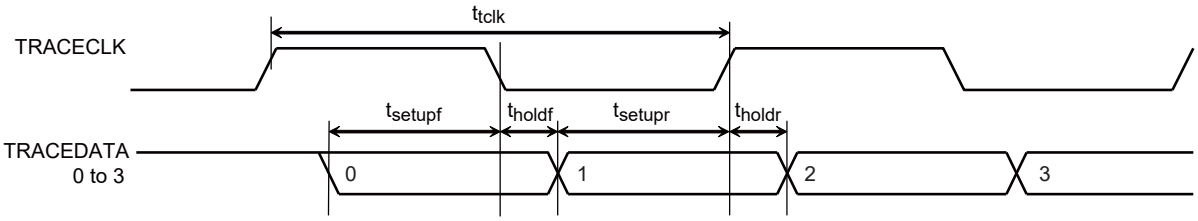
29.5.10.3 JTAG Interface

Parameter	Symbol	Min	Max	Unit
CLK cycle	T_{dck}	100	–	ns
CLK fall→ Output data hold	T_{d3}	4	–	
CLK fall→ to output data valid	T_{d4}	–	50	
Input data valid → CLK rise	T_{ds}	20	–	
CLK rise → Input data hold	T_{dh}	15	–	



29.5.11 ETM Trace

Parameter	Symbol	Min	Max	Unit
TRACECLK cycle	t_{tclk}	33.3	–	ns
TRACEDATA valid ← TRACECLK rise	t_{setupr}	2	–	
TRACECLK rise → TRACEDATA hold	t_{holdr}	1	–	
TRACEDATA valid ← TRACECLK fall	t_{setupf}	2	–	
TRACECLK fall→ TRACEDATA hold	t_{holdf}	1	–	



29.5.12 On-chip Oscillator Characteristic

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Oscillation frequency	IHOSC	Ta = -40 to 85°C	9.0	10.0	11.0	MHz

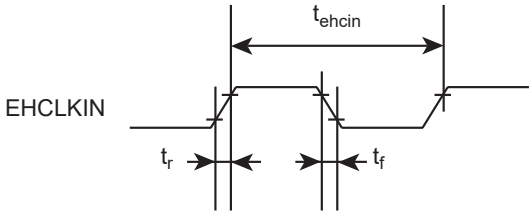
Note: Do not use an on-chip oscillator as a system clock (fsys) when high-accuracy oscillation frequency is required.

29.5.13 External Oscillator

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
High-frequency oscillation	EHOSC	Ta = -40 to 85°C	8	–	16	MHz

29.5.14 External Clock Input

Parameter	Symbol	Min	Typ.	Max	Unit
External clock frequency	t_{ehcin}	8	–	40	MHz
External clock duty	–	45	–	55	%
External clock input rise time	t_r	–	–	10	ns
External clock input fall time	t_f	–	–	10	ns



29.5.15 Flash Characteristic

Parameter	Condition	Min	Typ.	Max	Unit
Guaranteed number of Flash memory programming	DVDD3 = RVDD3 = AVDD3 = 2.7 V to 3.6 V Ta = -40 to 85°C	–	–	10000	times

29.5.16 Noise Filter Characteristic

Parameter	Condition	Min	Typ.	Max	Unit
The width of noise filter	–	15	30	60	ns

29.6 Recommended Oscillation Circuit

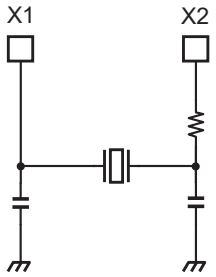


Figure 29-1 High-frequency oscillation connection

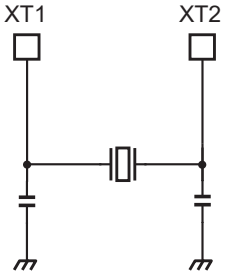


Figure 29-2 Low-frequency oscillation connection

Note: To obtain a stable oscillation, load capacity and the position of the oscillator must be configured properly. Since these factors are strongly affected by substrate patterns, please evaluate oscillation stability using the substrate you use.

The TMPM46BF10FG has been evaluated by the oscillator vendor below. Please refer this information when selecting external parts

29.6.1 Ceramic Oscillator

This product has been evaluated by the ceramic oscillator by Murata Manufacturing Co., Ltd. Please refer to the company's website for details.

29.6.2 Crystal Oscillator

This product has been evaluated by the ceramic oscillator by KYOCERA Crystal Device Corpora-tion. Please refer to the company's website for details.

29.6.3 Precautions for designing printed circuit board

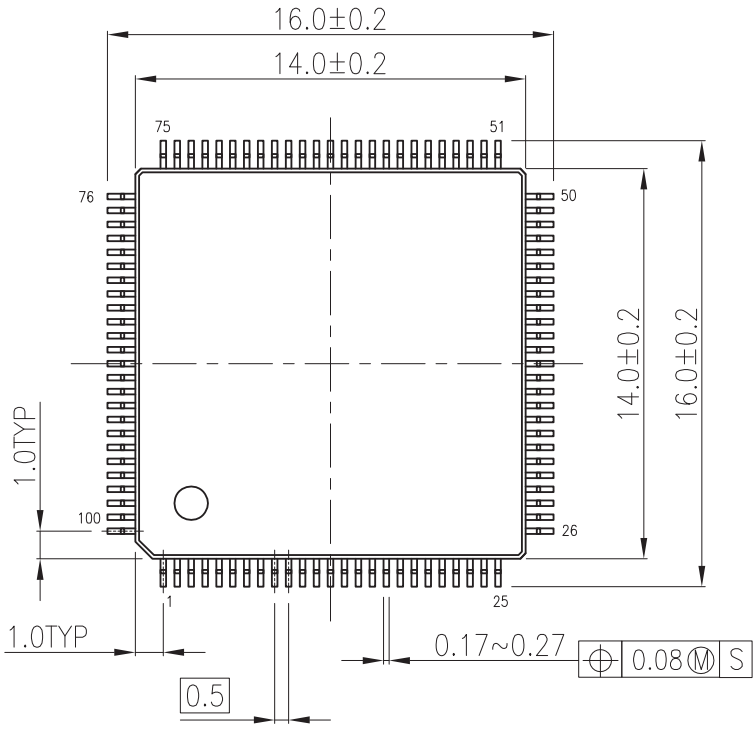
Be sure to design printed circuit board patterns that connect a crystal unit with other oscillation elements so that the length of such patterns become shortest possible to prevent deterioration of characteristics due to stray capacitances and wiring inductance. For multi-layer circuit boards, it is important 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.

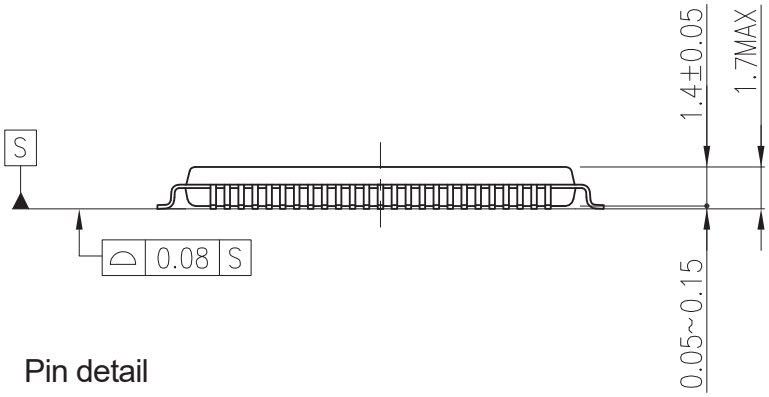
30. Package Dimensions

Type: P-LQFP100-1414-0.50-002

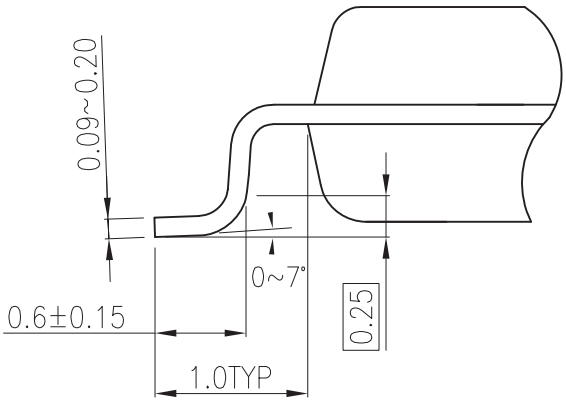
Unit: mm

Dimensions





Pin detail



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 all 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, Class 3 medical devices, equipment used for automobiles, and military vehicles and munitions. **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.**