TOSHIBA

32 Bit RISC Microcontroller

TX03 Series

TMPM366FDXBG/FYXBG/FWXBG

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		

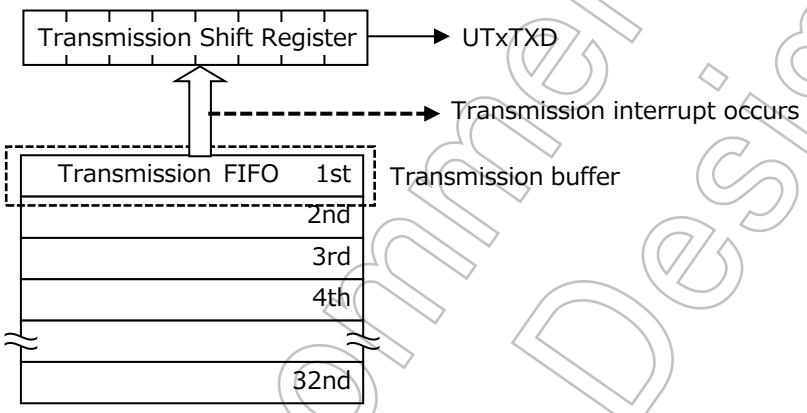
2. Details

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").

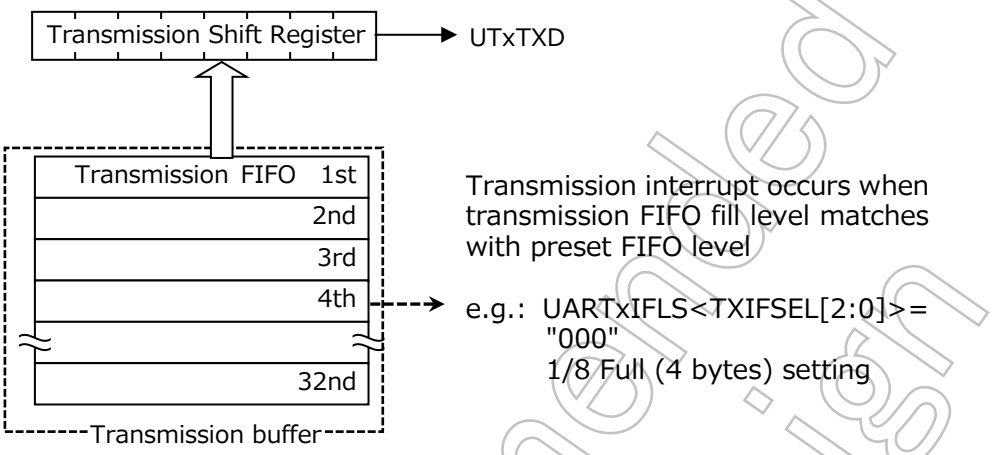
When the transmission is terminated, the final transmission data is transferred to the shift register, and the final transmission interrupt occurs when the transmission buffer becomes empty. If the next data is not written to the transmission buffer, the transmission interrupt can be intentionally cleared 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

ARM

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

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

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

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

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

Base Address = 0x0000_0000

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

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

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

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

1.2.2 SAMCR(Control register)

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

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

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

R / W	READ WRITE
R	READ
W	WRITE

c. Data descriptopn

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

- x:channel numbers/ports
- n,m:bit numbers

d. Register descriptoption

Registers are described as shown below.

- Register name <Bit Symbol>

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

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

- Register name [Bit]

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

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

Revision History

Date	Revision	Comment
2011/05/10	Tentative 1	First Release of Revised
2011/10/19	1	First Release
2011/11/17	2	Contents Revised
2013/06/11	3	Contents Revised
2022/09/30	4	Contents Revised
2023/07/21	5	Contents Revised

TMPM366FDXBG/FYXBG/FWXBG

The TMPM366FDXBG/FYXBG/FWXBG is a 32-bit RISC microprocessor series with an ARM Cortex™-M3 microprocessor core.

Product Name	ROM (FLASH)	RAM	Package
TMPM366FDXBG	512 Kbyte	64 Kbyte	P-TFBGA109-0909-0.65-002
TMPM366FYXBG	256 Kbyte	48 Kbyte	
TMPM366FWXBG	128 Kbyte	32 Kbyte	

Features of the TMPM366FDXBG/FYXBG/FWXBG are as follows:

1.1 Features

1. ARM Cortex-M3 microprocessor core
   a. Improved code efficiency has been realized through the use of Thumb® -2 instruction.
      - New 16-bit Thumb instructions for improved program flow
      - New 32-bit Thumb instructions for improved performance
      - New Thumb mixed 16-/32-bit instruction set can produce faster, more efficient code.
   b. Both high performance and low power consumption have been achieved.

      [High performance]
      - A 32-bit multiplication (32×32=32 bit) can be executed with one clock.
      - Division takes between 2 and 12 cycles depending on dividend and devisor

      [Low power consumption]
      - Optimized design using a low power consumption library
      - Standby function that stops the operation of the micro controller core
   c. High-speed interrupt response suitable for real-time control
      - An interruptible long instruction.
      - Stack push automatically handled by hardware.

2. On chip program memory and data memory
   - On chip SRAM : 64 Kbyte / 48 Kbyte / 32 Kbyte
   - On chip Flash ROM : 512 Kbyte / 256 Kbyte / 128 Kbyte

3. External bus interface (EBIF)
   - Up to 16Mbytes access area (Program / Data)
   - External data bus (Separate / Multiplex): 8 /16bit bus width
   - Chip select / Wait controller: 2 channels

4. DMA controller (DMAC) : 2 units 4 channels

Transfer can support on chip Memory / Peripheral function / External memory

5. 16-bit timer / event counter (TMRB) : 10 channels
   - 16-bit interval timer mode
   - 16-bit event counter mode
   - 16-bit PPG output (can start 4-channels synchronously)
   - Input capture function

6. Watchdog timer (WDT): 1 channel

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

7. Asynchronous serial channel (UART) : 1 channel

   Supports UART with flow control / supports IrDA1.0 mode

8. Serial channel (SIO/UART): 2 channels

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

9. Serial bus interface (I2C/SIO): 2 channels

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

10. Synchronous serial port (SSP): 3 channel
    - Communication protocol that includes SPI: 3 types (SPI/SSI/Microwire)
    - Baud rate: Master mode: 12MHz (max.) (@ fsys=48MHz),
      Slave mode: 4.0MHz (max.) (@ fsys=48MHz)

11. USB Device controller : 1channel
    - USB support
    - Supports full communication speed (12Mbps) (dose not support Low Speed).
    - Supports 8 endpoints.

      End -point 0 : Control 64 bytes × 1-FIFO

      End -point 1 : Bulk (Device → Host : IN transfer) 64 bytes × 2-FIFO

      End -point 2 : Bulk (Host → Device : OUT transfer) 64 bytes × 2-FIFO

      End -point 3 : Bulk (Device → Host : IN transfer) 64 bytes × 2-FIFO

      End -point 4 : Bulk (Host → Device : OUT transfer) 64 bytes × 2-FIFO

      End -point 5 : Bulk (Device → Host : IN transfer) 64 bytes × 2-FIFO

      End -point 6 : Bulk (Host → Device : OUT transfer) 64 bytes × 2-FIFO

      End -point 7 : Interrupt (Device → Host : IN transfer) 64 bytes × 2-FIFO
    - From End-point 1 to 7 can be support 4 transfer modes.

12. 12-bit AD converter (ADC): 12channels
    - Start up with 16-bit timer / Start up with an external trigger input
    - Fixed channel / Channel scan mode
    - Single / repeat mode
    - AD monitoring 2channels
    - Conversion time 1μs (@fsys = 40MHz), 1.67μs (@fsys = 48MHz)

13. Interrupt source
    - Internal 50 factors: The order of precedence can be set over 7 levels (except the watchdog timer interrupt).
    - External 10 factors: The order of precedence can be set over 7 levels.

14. Non-maskable interrupt (NMI)

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

15. Input/ output ports (PORT): 74 pins

    I/O pin: 73 pins (One 5V tolerant pin included)

    Output pin: 1 pin

16. Low power consumption mode

    IDLE, STOP1, STOP2

17. Clock generator (CG)
    - On chip PLL (6 times or 8 times selectable)
    - Clock gear function: The high-speed clock can be divided into 1/1, 1/2, 1/4, 1/8 or 1/16.

18. Endian

    Little endian can be supported.

19. Debug interface

    JTAG / SWD / SWV / TRACE (DATA 4bit)

20. JTAG interface

    Boundary scan

21. Maximum operating frequency: 48 MHz

22. Operating voltage range

    2.7 V to 3.6 V

    3.0 V to 3.45 V (when USB is used)

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

24. Package

    P-TFBGA109-0909-0.65-002 (9mm × 9mm, 0.65mm pitch)

1.2 Block Diagram

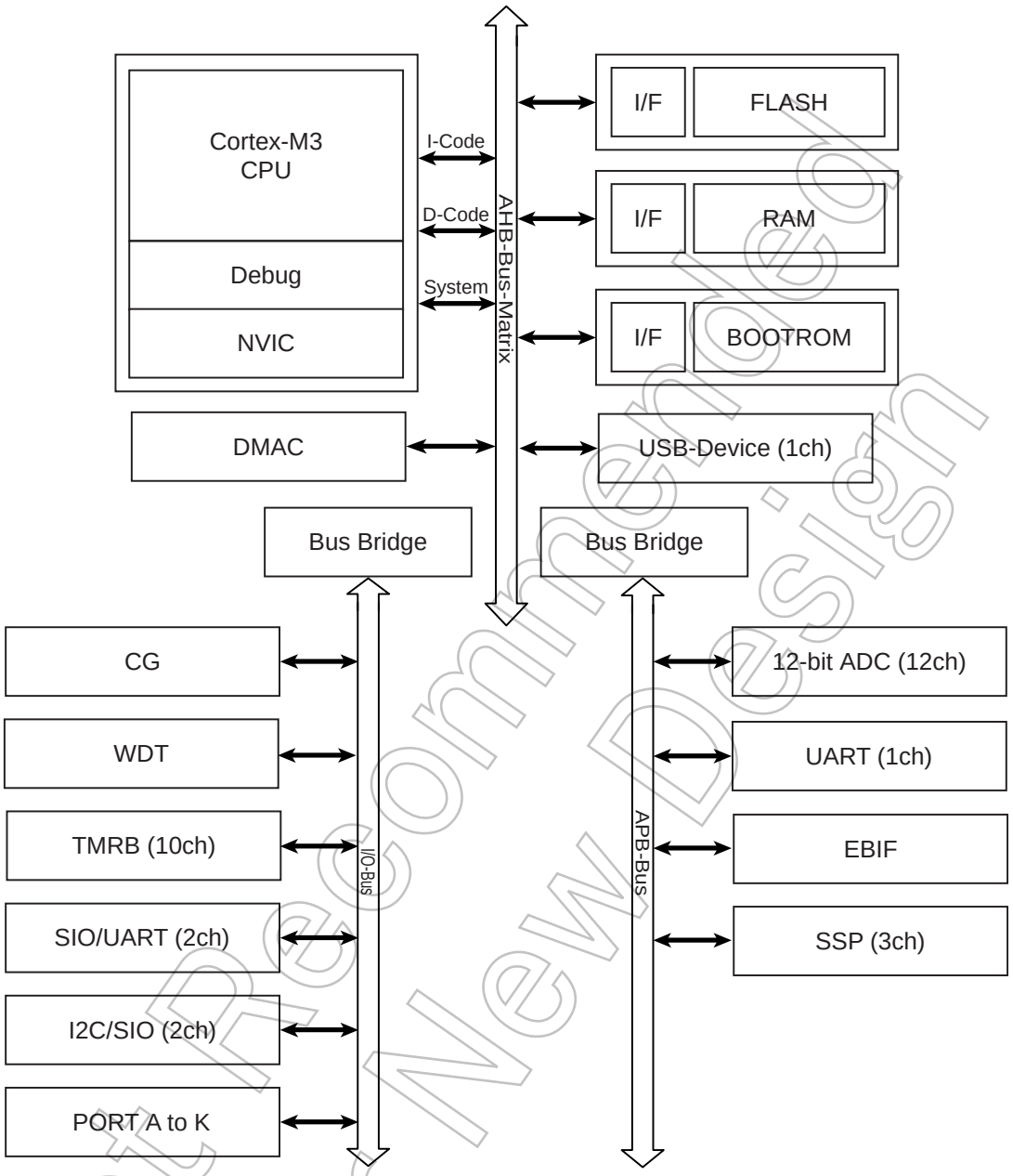


Figure 1-1 TMPM366FDXBG/FYXBG/FWXBGBlock Diagram

1.3 Pin Layout (Top view)

Figure 1-2 shows the pin layout of TMPM366FDXBG/FYXBG/FWXBG.

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12
D1	D2	D3	D4	—	—	—	—	—	D10	D11	D12
E1	E2	E3	—	—	—	—	—	—	E10	E11	E12
F1	F2	F3	—	—	—	—	—	—	F10	F11	F12
G1	G2	G3	—	—	—	—	—	—	G10	G11	G12
H1	H2	H3	—	—	—	—	—	—	H10	H11	H12
J1	J2	J3	—	—	—	—	—	—	J10	J11	J12
K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12
L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12
M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12

Figure 1-2 Pin Layout (BGA109)

1.4 Pin names and Functions

Table 1-1 and Table 1-2 sort the input and output pins of the TMPM366FDXBG/FYXBG/FWXBG by pin or port.

1.4.1 Sorted by Pin

Table 1-1 Pin Names and Functions Sorted by Pin (1/7)

Type	Pin No.	Pin Name	Input/ Out-put	Function
Function	A1	PK3 AIN11 TB6IN1	I/O I I	I/O port Analog input Inputting 16-bit timer / event counter capture trigger
Function	A2	PJ7 AIN07 INT9 TB0IN1	I/O I I I	I/O port Analog input External interrupt pin Inputting 16-bit timer / event counter capture trigger
Function	A3	PJ3 AIN03	I/O I	I/O port Analog input
Function	A4	PJ0 AIN00	I/O I	I/O port Analog input
PS	A5	RVDD3	−	Power supply pin for internal regulator
PS	A6	RVSS	−	GND pin for internal regulator
PS	A7	DVSSA	−	GND pin
PS	A8	DVDD3A	−	Power supply pin
Clock	A9	X1	I	Connected to a high-speed oscillator or input external clock.
PS	A10	DVSSC	−	GND pin
Clock	A11	X2	O	Connected to a high-speed oscillator.
PS	A12	DVSSC	−	GND pin
PS	B1	AVDD3	I	Power supply pin for AD converters (note) AVDD3 must be connected to power supply even if A/D converters are not used.
Function	B2	PK2 AIN10 TB6IN0	I/O I I	I/O port Analog input Inputting 16-bit timer / event counter capture trigger
Function	B3	PJ4 AIN04	I/O I	I/O port Analog input
Function	B4	PJ1 AIN01	I/O I	I/O port Analog input
PS	B5	AVSS	I	GND pin for AD converters (note) AVSS must be connected to GND even if the A/D converters are not used.
Function	B6	PE7 INT4 A11	I/O I O	I/O port External interrupt pin Address bus
Function	B7	PE5 SCL1/SI1 A13	I/O I/O O	I/O port Clock in I2C mode / Data in SIO mode Address bus

Table 1-1 Pin Names and Functions Sorted by Pin (2/7)

Type	Pin No.	Pin Name	Input/ Out-put	Function
PS	B8	DVDD3A	−	Power supply pin
Function	B9	$\overline{NMI}$	I	Non-maskable interrupt (note) With a noise filter (about 30ns (typical value))
PS	B10	DVSSC	−	GND pin
Control	B11	MODE	I	Mode pin (note) MODE pin must be connected to GND.
Function	B12	$\overline{RESET}$	I	Reset input pin (note) With a pull-up and a noise filter (about 30ns (typical value))
PS	C1	AVREFH	I	Power supply pin for AD converters (note) AVREFH must be connected to power supply even if A/D converters are not used.
Function	C2	PK1 AIN09 INT3 TB1IN1	I/O I I I	I/O port Analog input External interrupt pin Inputting 16-bit timer / event counter capture trigger
Function	C3	PJ5 AIN05	I/O I	I/O port Analog input
Function	C4	PJ2 AIN02	I/O I	I/O port Analog input
PS	C5	AVSS	I	GND pin for AD converters (note) AVSS must be connected to GND even if the A/D converters are not used.
Function	C6	PE6 SCK1 A12	I/O I/O O	I/O port Clock in SIO mode Address bus
Function	C7	PE4 SDA1/SO1 A14	I/O I/O O	I/O port Data in I2C mode / Data in SIO mode Address bus
Function	C8	PD0 A16 TB7OUT	I/O O O	I/O port Address bus 16-bit timer / event counter output
Function/ Debug	C9	PD1 A17 TB8OUT	I/O O O	I/O port Address bus 16-bit timer / event counter output
Function	C10	PD2 A18 TB9OUT	I/O O O	I/O port Address bus 16-bit timer / event counter output
Function	C11	PE2 SCLK0 TB2OUT $\overline{CTS0}$ A22	I/O I/O O I O	I/O port Serial channel clock pin 16-bit timer / event counter output Serial channel handshake input pin Address bus

Table 1-1 Pin Names and Functions Sorted by Pin (3/7)

Type	Pin No.	Pin Name	Input/ Out-put	Function
Function	C12	PE3 INT5 A15 TB3OUT A23	I/O I O O O	I/O port External interrupt pin Address bus 16-bit timer / event counter output Address bus
PS	D1	AVSS	I	GND pin for AD converters (note) AVSS must be connected to GND even if the A/D converters are not used.
Function	D2	PK0 AIN08 INT2 TB1IN0	I/O I I I	I/O port Analog input External interrupt pin Inputting 16-bit timer / event counter capture trigger
Function	D3	PJ6 AIN06 TB0IN0	I/O I I	I/O port Analog input Inputting 16-bit timer / event counter capture trigger
PS	D4	AVSS	I	GND pin for AD converters (note) AVSS must be connected to GND even if the A/D converters are not used.
Function	D10	PD3 A19 $\overline{\text{ADTRG}}$	I/O O I	I/O port Address bus ADC trigger input
Function	D11	PE0 TXD0 A20	I/O O O	I/O port Serial channel sending serial data Address bus
Function	D12	PE1 RXD0 A21	I/O I O	GND pin Serial channel receiving serial data Address bus
PS	E1	AVREFL	I	Power supply pin for AD converters (note) AVREFL must be connected to GND even if A/D converters are not used.
PS	E2	AVSS	I	GND pin for AD converters (note) AVSS must be connected to GND even if the A/D converters are not used.
Control	E3	BSC	I	Boundary scan control pin
Function	E10	PD4 SP0DO	I/O O	I/O port SSP DO output
PS	E11	DVDD3C	−	Power supply pin for USB
PS	E12	DVDD3C	−	Power supply pin for USB
PS	F1	DVDD3A	−	Power supply pin
PS	F2	DVSSA	−	GND pin
Function/ Debug	F3	PI7 $\overline{\text{TRST}}$	I/O I	I/O port Debug pin
Function	F10	PD5 SP0DI	I/O I	I/O port SSP DI input
PS	F11	DVSS3C	−	GND pin for USB

Table 1-1 Pin Names and Functions Sorted by Pin (4/7)

Type	Pin No.	Pin Name	Input/ Out-put	Function
Function	F12	D+	I/O	USB pin (D+)
Function/ Debug	G1	PI2 TRACECLK	I/O O	I/O port Debug pin
Function/ Debug	G2	PI6 TDI	I/O I	I/O port Debug pin
Function/ Debug	G3	PI5 TDO/SWV	I/O O	I/O port Debug pin
Function	G10	PD6 SP0CLK	I/O I/O	I/O port SSP clock input/ output
PS	G11	DVSS3C	−	GND pin for USB
Function	G12	D−	I/O	USB pin (D-)
Function/ Debug	H1	PI0 TRACEDA-TA1	I/O O	I/O port Debug pin
Function/ Debug	H2	PI1 TRACEDA-TA0	I/O O	I/O port Debug pin
Function/ Debug	H3	PI4 TMS/SWDIO	I/O I/O	I/O port Debug pin
Function	H10	PD7 SP0FSS SCOUT	I/O I/O O	I/O port SSP FSS input/ output System clock output
PS	H11	DVSSA	−	GND pin
PS	H12	DVDD3A	−	Power supply pin
Function/ Debug	J1	PH0 TRACEDA-TA2	I/O O	I/O port Debug pin
Function/ Debug	J2	PH1 TRACEDA-TA3	I/O O	I/O port Debug pin
Function Debug	J3	PI3 TCK/SWCLK	I/O I	I/O port Debug pin
Function	J10	PB7 D15/AD15 SP2FSS A7	I/O I/O I/O O	I/O port Data bus/Address data bus SSP FSS input/ output Address bus
Function	J11	PB6 D14/AD14 SP2CLK A6	I/O I/O I/O O	I/O port Data bus/Address data bus SSP clock input/ output Address bus

Table 1-1 Pin Names and Functions Sorted by Pin (5/7)

Type	Pin No.	Pin Name	Input/ Out-put	Function
Function	J12	PB5 D13/AD13 SP2DI A5	I/O I/O I O	I/O port Data bus/Address bus SSP DI input Address bus
Function	K1	PH4 A8 INT8 DTR2	I/O O I O	I/O port Address bus External interrupt pin Output modem controlling DTR (Data Terminal Ready).
Function	K2	PH3 A9 TB5OUT DSR2	I/O O O I	I/O port Address bus 16-bit timer / event counter output Modem status signal DSR (Data Set Ready).
Function	K3	PH2 A10 TB4OUT DCD2	I/O O O I	I/O port Address bus 16-bit timer / event counter output Modem status signal DCD (Data Carrier Detect).
Function	K4	PG5 INT1 USBPON	I/O I I	I/O port (5V tolerant input) (note) External interrupt pin USB connection detection pin (VBUS Detect)
Function	K5	PC2 SCLK1 A0 TB0OUT $\overline{CTS1}$	I/O I/O O O I	I/O port Serial channel clock pin Address bus 16-bit timer / event counter output Serial channel handshake input pin
Function	K6	PF2 $\overline{WR}$	I/O O	I/O port Write strobe signal
Function	K7	PF5 $\overline{CS1}$ INT7 TB5IN1	I/O O I I	I/O port Chip select output External interrupt pin Inputting 16-bit timer / event counter capture trigger
Function	K8	PF7 ALE	I/O O	I/O port Address latch enable
Function	K9	PA1 D1/AD1	I/O I/O	I/O port Data bus/Address data bus
Function	K10	PA4 D4/AD4	I/O I/O	I/O port Data bus/Address data bus
Function	K11	PB2 D10/AD10 SP1CLK A2	I/O I/O I/O O	I/O port Data bus/Address data bus SSP clock input/ output Address bus
Function	K12	PB4 D12/AD12 SP2DO A4	I/O I/O O O	I/O port Data bus/Address data bus SSP DO output Address bus

Table 1-1 Pin Names and Functions Sorted by Pin (6/7)

Type	Pin No.	Pin Name	Input/ Out-put	Function
Function	L1	PG2 SCK0 A5 TB3IN1 $\overline{CTS2}$	I/O I/O O I I	I/O port Clock in SIO mode Address bus Inputting 16-bit timer / event counter capture trigger Output modem control line CTS (Clear To Send)
Function	L2	PG3 INT0 A6 TB4IN0 RIN2	I/O I O I I	I/O port External interrupt pin Address bus Inputting 16-bit timer / event counter capture trigger Modem status signal RI (Ring Indicator)
Function	L3	PG4 A7 TB4IN1 $\overline{RTS2}$	I/O O I O	I/O port Address bus Inputting 16-bit timer / event counter capture trigger Output modem control line RTS (Request To Send).
Function	L4	PC0 TXD1 A2 TB2IN0	I/O O O I	I/O port Serial channel sending serial data Address bus Inputting 16-bit timer / event counter capture trigger
Function	L5	PF1 $\overline{RD}$	I/O O	I/O port Read strobe signal
Function	L6	PF3 $\overline{BELL}$	I/O O	I/O port Byte enable signal as an external 8-bit memory access
Function	L7	PF6 $\overline{CS0}$	I/O O	I/O port Chip select output
Function	L8	PA0 D0/AD0	I/O I/O	I/O port Data bus/Address data bus
Function	L9	PA2 D2/AD2	I/O I/O	I/O port Data bus/Address data bus
Function	L10	PA5 D5/AD5	I/O I/O	I/O port Data bus/Address data bus
Function	L11	PB0 D8/AD8 SP1DO A0	I/O I/O O O	I/O port Data bus/Address data bus SSP DO output Address bus
Function	L12	PB3 D11/AD11 SP1FSS A3	I/O I/O I/O O	I/O port Data bus/Address data bus SSP FSS input/ output Address bus
Control	M1	FTEST3	–	TEST pin (note) TEST pin must be left OPEN.

Table 1-1 Pin Names and Functions Sorted by Pin (7/7)

Type	Pin No.	Pin Name	Input/ Output	Function
Function	M2	PG1 SCL0/SI0 A4 TB3IN0 RXD2 IRIN	I/O I/O O I I I	I/O port Clock in I2C mode/Data in SIO mode Address bus Inputting 16-bit timer / event counter capture trigger UART receive data. Data input pin for IrDA1.0.
Function	M3	PG0 SDA0/SO0 A3 TXD2 IROUT	I/O I/O O O O	I/O port Data in I2C mode/Data in SIO mode Address bus UART transmission data. Data output pin for IrDA1.0.
Function	M4	PC1 RXD1 A1 TB2IN1	I/O I O I	I/O port Serial channel receiving serial data Address bus Inputting 16-bit timer / event counter capture trigger
Function/ Control	M5	PF0 $\overline{\text{BOOT}}$ TB6OUT	O I O	Output port Setting a boot mode TMPM366FDXBG/FYXBG/FWXBG goes into single boot mode by sampling "Low" at the rising edge of a $\overline{\text{RESET}}$ pin. 16-bit timer / event counter output
Function	M6	PF4 $\overline{\text{BELH}}$ INT6 TB5IN0	I/O O I I	I/O port Byte enable signal as an external 8-bit memory access External interrupt pin Inputting 16-bit timer / event counter capture trigger
PS	M7	DVSSA	−	GND pin
PS	M8	DVDD3A	−	Power supply pin
Function	M9	PA3 D3/AD3	I/O I/O	I/O port Data bus/Address data bus
Function	M10	PA6 D6/AD6	I/O I/O	I/O port Data bus/Address data bus
Function	M11	PA7 D7/AD7	I/O I/O	I/O port Data bus/Address data bus
Function	M12	PB1 D9/AD9 SP1DI A1	I/O I/O I O	I/O port Data bus/Address data bus SSP DI input Address bus

Note: Only when input is enabled, this pin tolerates 5V input. Note that this pin can not be pulled up over the power supply voltage when using as open-drain output.

1.4.2 Sorted by Port

Table 1-2 Pin Names and Functions Sorted by Port (1/7)

PORT	Type	Pin No.	Pin Name	Input/ Out-put	Function
PORT A	Function	L8	PA0 D0/AD0	I/O I/O	I/O port Data bus/Address data bus
PORT A	Function	K9	PA1 D1/AD1	I/O I/O	I/O port Data bus/Address data bus
PORT A	Function	L9	PA2 D2/AD2	I/O I/O	I/O port Data bus/Address data bus
PORT A	Function	M9	PA3 D3/AD3	I/O I/O	I/O port Data bus/Address data bus
PORT A	Function	K10	PA4 D4/AD4	I/O I/O	I/O port Data bus/Address data bus
PORT A	Function	L10	PA5 D5/AD5	I/O I/O	I/O port Data bus/Address data bus
PORT A	Function	M10	PA6 D6/AD6	I/O I/O	I/O port Data bus/Address data bus
PORT A	Function	M11	PA7 D7/AD7	I/O I/O	I/O port Data bus/Address data bus
PORT B	Function	L11	PB0 D8/AD8 SP1DO A0	I/O I/O O O	I/O port Data bus/Address data bus SSP DO output Address bus
PORT B	Function	M12	PB1 D9/AD9 SP1DI A1	I/O I/O I O	I/O port Data bus/Address data bus SSP DI input Address bus
PORT B	Function	K11	PB2 D10/AD10 SP1CLK A2	I/O I/O I/O O	I/O port Data bus/Address data bus SSP clock input/ output Address bus
PORT B	Function	L12	PB3 D11/AD11 SP1FSS A3	I/O I/O I/O O	I/O port Data bus/Address data bus SSP FSS input/ output Address bus
PORT B	Function	K12	PB4 D12/AD12 SP2DO A4	I/O I/O O O	I/O port Data bus/Address data bus SSP DO output Address bus
PORT B	Function	J12	PB5 D13/AD13 SP2DI A5	I/O I/O I O	I/O port Data bus/Address bus SSP DI input Address bus

Table 1-2 Pin Names and Functions Sorted by Port (2/7)

PORT	Type	Pin No.	Pin Name	Input/ Out-put	Function
PORT B	Function	J11	PB6 D14/AD14 SP2CLK A6	I/O I/O I/O O	I/O port Data bus/Address data bus SSP clock input/ output Address bus
PORT B	Function	J10	PB7 D15/AD15 SP2FSS A7	I/O I/O I/O O	I/O port Data bus/Address data bus SSP FSS input/ output Address bus
PORT C	Function	L4	PC0 TXD1 A2 TB2IN0	I/O O O I	I/O port Serial channel sending serial data Address bus Inputting 16-bit timer / event counter capture trigger
PORT C	Function	M4	PC1 RXD1 A1 TB2IN1	I/O I O I	I/O port Serial channel receiving serial data Address bus Inputting 16-bit timer / event counter capture trigger
PORT C	Function	K5	PC2 SCLK1 A0 TB0OUT $\overline{\text{CTS1}}$	I/O I/O O O I	I/O port Serial channel clock pin Address bus 16-bit timer / event counter output Serial channel handshake input pin
PORT D	Function	C8	PD0 A16 TB7OUT	I/O O O	I/O port Address bus 16-bit timer / event counter output
PORT D	Function	C9	PD1 A17 TB8OUT	I/O O O	I/O port Address bus 16-bit timer / event counter output
PORT D	Function	C10	PD2 A18 TB9OUT	I/O O O	I/O port Address bus 16-bit timer / event counter output
PORT D	Function	D10	PD3 A19 $\overline{\text{ADTRG}}$	I/O O I	I/O port Address bus ADC trigger input
PORT D	Function	E10	PD4 SP0DO	I/O O	I/O port SSP DO output
PORT D	Function	F10	PD5 SP0DI	I/O I	I/O port SSP DI input
PORT D	Function	G10	PD6 SP0CLK	I/O I/O	I/O port SSP clock input/ output
PORT D	Function	H10	PD7 SP0FSS SCOUT	I/O I/O O	I/O port SSP FSS input/ output System clock output

Table 1-2 Pin Names and Functions Sorted by Port (3/7)

PORT	Type	Pin No.	Pin Name	Input/ Output	Function
PORT E	Function	D11	PE0 TXD0 A20	I/O O O	I/O port Serial channel sending serial data Address bus
PORT E	Function	D12	PE1 RXD0 A21	I/O I O	GND pin Serial channel receiving serial data Address bus
PORT E	Function	C11	PE2 SCLK0 TB2OUT $\overline{CTS0}$ A22	I/O I/O O I O	I/O port Serial channel clock pin 16-bit timer / event counter output Serial channel handshake input pin Address bus
PORT E	Function	C12	PE3 INT5 A15 TB3OUT A23	I/O I O O O	I/O port External interrupt pin Address bus 16-bit timer / event counter output Address bus
PORT E	Function	C7	PE4 SDA1/SO1 A14	I/O I/O O	I/O port Data in I2C mode/Data in SIO mode Address bus
PORT E	Function	B7	PE5 SCL1/SI1 A13	I/O I/O O	I/O port Clock in I2C mode/Data in SIO mode Address bus
PORT E	Function	C6	PE6 SCK1 A12	I/O I/O O	I/O port Clock in SIO mode Address bus
PORT E	Function	B6	PE7 INT4 A11	I/O I O	I/O port External interrupt pin Address bus
PORT F	Function/ Control	M5	PF0 TB6OUT $\overline{BOOT}$	I/O O I	I/O port 16-bit timer / event counter output Setting a boot mode TMPM366FDXBG/FYXBG/FWXBG goes into single boot mode by sampling "Low" at the rising edge of a $\overline{RESET}$ pin.
PORT F	Function	L5	PF1 $\overline{RD}$	I/O O	I/O port Read strobe signal
PORT F	Function	K6	PF2 $\overline{WR}$	I/O O	I/O port Write strobe signal
PORT F	Function	L6	PF3 $\overline{BELL}$	I/O O	I/O port Byte enable signal as an external 8-bit memory access
PORT F	Function	M6	PF4 $\overline{BELH}$ INT6 TB5IN0	I/O O I I	I/O port Byte enable signal as an external 8-bit memory access External interrupt pin Inputting 16-bit timer / event counter capture trigger

Table 1-2 Pin Names and Functions Sorted by Port (4/7)

PORT	Type	Pin No.	Pin Name	Input/ Out-put	Function
PORT F	Function	K7	PF5	I/O	I/O port
			$\overline{CS1}$	O	Chip select output
			INT7	I	External interrupt pin
			TB5IN1	I	Inputting 16-bit timer / event counter capture trigger
PORT F	Function	L7	PF6	I/O	I/O port
			$\overline{CS0}$	O	Chip select output
PORT F	Function	K8	PF7	I/O	I/O port
			ALE	O	Address latch enable (output disable for noise-reduction can be selectalbe)
PORT G	Function	M3	PG0	I/O	I/O port
			SDA0/SO0	I/O	Data in I2C mode/Data in SIO mode
			A3	O	Address bus
			TXD2	O	UART transmission data.
			IROUT	O	Data output pin for IrDA1.0.
PORT G	Function	M2	PG1	I/O	I/O port
			SCL0/SI0	I/O	Clock in I2C mode/Data in SIO mode
			A4	O	Address bus
			TB3IN0	I	Inputting 16-bit timer / event counter capture trigger
			RXD2	I	UART receive data.
			IRIN	I	Data input pin for IrDA1.0.
PORT G	Function	L1	PG2	I/O	I/O port
			SCK0	I/O	Clock in SIO mode
			A5	O	Address bus
			TB3IN1	I	Inputting 16-bit timer / event counter capture trigger
			$\overline{CTS2}$	I	Output modem control line CTS (Clear To Send)
PORT G	Function	L2	PG3	I/O	I/O port
			INT0	I	External interrupt pin
			A6	O	Address bus
			TB4IN0	I	Inputting 16-bit timer / event counter capture trigger
			RIN2	I	Modem status signal RI (Ring Indicator)
PORT G	Function	L3	PG4	I/O	I/O port
			A7	O	Address bus
			TB4IN1	I	Inputting 16-bit timer / event counter capture trigger
			RTS2	O	Output modem control line RTS (Request To Send).
PORT G	Function	K4	PG5	I/O	I/O port (5V tolerant input) (note)
			INT1	I	External interrupt pin
			USBPON	I	USB connection detection pin (VBUS Detect)
PORT H	Function/ Debug	J1	PH0	I/O	I/O port
			TRACEDA-TA2	O	Debug pin
PORT H	Function/ Debug	J2	PH1	I/O	I/O port
			TRACEDA-TA3	O	Debug pin
PORT H	Function	K3	PH2	I/O	I/O port
			A10	O	Address bus
			TB4OUT	O	16-bit timer / event counter output
			DCD2	I	Modem status signal DCD (Data Carrier Detect).

Table 1-2 Pin Names and Functions Sorted by Port (5/7)

PORT	Type	Pin No.	Pin Name	Input/ Out-put	Function
PORT H	Function	K2	PH3 A9 TB5OUT DSR2	I/O O O I	I/O port Address bus 16-bit timer / event counter output Modem status signal DSR (Data Set Ready).
PORT H	Function/ Debug	K1	PH4 A8 INT8 DTR2	I/O O I O	I/O port Address bus External interrupt pin Output modem controlling DTR (Data Terminal Ready).
PORT I	Function/ Debug	H1	PI0 TRACEDA-TA1	I/O O	I/O port Debug pin
PORT I	Function/ Debug	H2	PI1 TRACEDA-TA0	I/O O	I/O port Debug pin
PORT I	Function/ Debug	G1	PI2 TRACECLK	I/O O	I/O port Debug pin
PORT I	Function/ Debug	J3	PI3 TCK/SWCLK	I/O I	I/O port Debug pin
PORT I	Function/ Debug	H3	PI4 TMS/SWDIO	I/O I/O	I/O port Debug pin
PORT I	Function/ Debug	G3	PI5 TDO/SWV	I/O O	I/O port Debug pin
PORT I	Function/ Debug	G2	PI6 TDI	I/O I	I/O port Debug pin
PORT I	Function/ Debug	F3	PI7 $\overline{\text{TRST}}$	I/O I	I/O port Debug pin
PORT J	Function	A4	PJ0 AIN00	I/O I	I/O port Analog input
PORT J	Function	B4	PJ1 AIN01	I/O I	I/O port Analog input
PORT J	Function	C4	PJ2 AIN02	I/O I	I/O port Analog input
PORT J	Function	A3	PJ3 AIN03	I/O I	I/O port Analog input
PORT J	Function	B3	PJ4 AIN04	I/O I	I/O port Analog input
PORT J	Function	C3	PJ5 AIN05	I/O I	I/O port Analog input
PORT J	Function	D3	PJ6 AIN06 TB0IN0	I/O I I	I/O port Analog input Inputting 16-bit timer / event counter capture trigger
PORT J	Function	A2	PJ7 AIN07 INT9 TB0IN1	I/O I I I	I/O port Analog input External interrupt pin Inputting 16-bit timer / event counter capture trigger

Table 1-2 Pin Names and Functions Sorted by Port (6/7)

PORT	Type	Pin No.	Pin Name	Input/ Out-put	Function
PORT K	Function	D2	PK0 AIN08 INT2 TB1IN0	I/O I I I	I/O port Analog input External interrupt pin Inputting 16-bit timer / event counter capture trigger
PORT K	Function	C2	PK1 AIN09 INT3 TB1IN1	I/O I I I	I/O port Analog input External interrupt pin Inputting 16-bit timer / event counter capture trigger
PORT K	Function	B2	PK2 AIN10 TB6IN0	I/O I I	I/O port Analog input Inputting 16-bit timer / event counter capture trigger
PORT K	Function	A1	PK3 AIN11 TB6IN1	I/O I I	I/O port Analog input Inputting 16-bit timer / event counter capture trigger
-	Function	F12	D−	I/O	USB DP(D+) pin
-	Function	G12	D+	I/O	USB DM(D-) pin
-	Control	B12	$\overline{\text{RESET}}$	I	Reset input pin (note) With a pull-up and a noise filter (about 30ns (typical value))
-	Function	B9	$\overline{\text{NMI}}$	I	Non-maskable interrupt (note) With a noise filter (about 30ns (typical value))
-	Control	B11	MODE	I	Mode pin (note) MODE pin must be connected to GND.
-	Control	M1	FTEST3	−	TEST pin (note) TEST pin must be left OPEN.
-	Control	E3	BSC	I	Boundary scan control pin
-	Clock	A9	X1	I	Connected to a high-speed oscillator or input external clock.
-	Clock	A11	X2	O	Connected to a high-speed oscillator.
-	PS	F1	DVDD3A	−	Power supply pin
-	PS	M8	DVDD3A	−	Power supply pin
-	PS	H12	DVDD3A	−	Power supply pin
-	PS	A8	DVDD3A	−	Power supply pin
-	PS	B8	DVDD3A	−	Power supply pin
-	PS	F2	DVSSA	−	GND pin
-	PS	M7	DVSSA	−	GND pin
-	PS	H11	DVSSA	−	GND pin
-	PS	A7	DVSSA	−	GND pin

Table 1-2 Pin Names and Functions Sorted by Port (7/7)

PORT	Type	Pin No.	Pin Name	Input/ Out-put	Function
-	PS	A5	RVDD3	−	Power supply pin for internal regulator
-	PS	A6	RVSS	−	GND pin for internal regulator
-	PS	A10	DVSSC	−	GND pin
-	PS	E12	DVDD3C	−	Power supply pin for USB
-	PS	G11	DVSS3C	−	GND pin for USB
-	PS	C1	AVREFH	I	Power supply pin for AD converters. (note) AVREFH must be connected to power supply even if A/D converters are not used.
-	PS	E1	AVREFL	I	Power supply pin for AD converters. (note) AVREFL must be connected to GND even if A/D converters are not used.
-	PS	B1	AVDD3	I	Power supply pin for AD converters. (note) AVDD3 must be connected to power supply even if A/D converters are not used.
-	PS	D1	AVSS	I	GND pin for AD converters. (note) AVSS must be connected to GND even if the A/D converters are not used.
-	PS	A12	DVSSC	−	GND pin
-	PS	B5	AVSS	I	GND pin for AD converters. (note) AVSS must be connected to GND even if the A/D converters are not used.
-	PS	B10	DVSSC	−	GND pin
-	PS	C5	AVSS	I	GND pin for AD converters. (note) AVSS must be connected to GND even if the A/D converters are not used.
-	PS	D4	AVSS	I	GND pin for AD converters. (note) AVSS must be connected to GND even if the A/D converters are not used.
-	PS	E2	AVSS	I	GND pin for AD converters. (note) AVSS must be connected to GND even if the A/D converters are not used.
-	PS	E11	DVDD3C	−	Power supply pin for USB
-	PS	F11	DVSS3C	−	GND pin for USB

Note: Only when input is enabled, this pin tolerates 5V input. Note that this pin can not be pulled up over the power supply voltage when using as open-drain output.

1.5 Pin Numbers and Power Supply Pins

Table 1-3 Pin Numbers and Power Supplies

Power supply	Voltage range	Pin No.	Pin name
DVDD3A	2.7 to 3.6V (When USB is used : 3.0 to 3.45 V)	A8,B8,F1,H12, M8	PA,PB,PC,PD,PE,PF,PG, PH, PI, X1, X2, FTEST3, $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC
AVDD3		B1	PJ, PK
RVDD3		A5	−
DVDD3C		E11,E12	D+,D−

2. Processor Core

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

2.1 Information on the processor core

The following table shows the revision of the processor core in the TMPM366FDXBG/FYXBG/FWXBG.

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

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

Product Name	Core Revision
TMPM366FDXBG/ FYXBG/FWXBG	r2p0

2.2 Configurable Options

The Cortex-M3 core has optional blocks. The optional blocks of the revision r2p0 are ETM™, MPU and WIC. The following table shows the configurable options in the TMPM366FDXBG/FYXBG/FWXBG.

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

2.3 Exceptions/ Interruptions

Exceptions and interruptions are described in the following section.

2.3.1 Number of Interrupt Inputs

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

TMPM366FDXBG/FYXBG/FWXBG has 60 interrupt inputs. The number of interrupt inputs is reflected in <INTLINESNUM[4:0]> bit of NVIC register. In this product, if read <INTLINESNUM[4:0]> bit, 0x00 is read out.

2.3.2 Number of Priority Level Interrupt Bits

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

TMPM366FDXBG/FYXBG/FWXBG has 3 priority level interrupt bits. The number of priority level interrupt bits is used for assigning a priority level in the interrupt priority registers and system handler priority registers.

2.3.3 SysTick

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

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

2.3.4 SYSRESETREQ

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

TMPM366FDXBG/FYXBG/FWXBG provides the same operation when SYSRESETREQ signal are output.

2.3.5 LOCKUP

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

TMPM366FDXBG/FYXBG/FWXBG does not use this signal. To return from LOCKUP status, it is necessary to use non-maskable interruput (NMI) or reset.

2.3.6 Auxiliary Fault Status register

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

However, TMPM366FDXBG/FYXBG/FWXBG is not defined this function. If auxiliary fault status register is read, always "0x0000_0000" is read out.

2.4 Events

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

TMPM366FDXBG/FYXBG/FWXBG does not use event output signals and event input signals. Please do not use SEV instruction and WFE instruction.

2.5 Power Management

The Cortex-M3 core provides power management system which uses SLEEPING signal and SLEEPDEEP 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.

TMPM366FDXBG/FYXBG/FWXBG does not use SLEEPDEEP signal so that <SLEEPDEEP> bit must not be set. And also event signal is not used so that please do not use WFE instruction.

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

2.6 Exclusive access

In Cortex-M3 core, the DCode bus system supports exclusive access. However TMPM366FDXBG/FYXBG/FWXBG does not use this function.

3. Debug Interface

3.1 Specification Overview

TMPM366FDXBG/FYXBG/FWXBG contains the Serial Wire JTAG Debug Port (SWJ-DP) unit for interfacing with the debugging 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 "Cortex-M3 Technical Reference Manual" .

3.2 SWJ-DP

SWJ-DP supports the Serial Wire Debug Port (SWCLK, SWDIO) and the JTAG Debug Port (TDI, TDO, TMS, TCK, $\overline{\text{TRST}}$).

3.3 ETM

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

3.4 Pin Functions

The debug interface pins can also be used as general-purpose ports.

The PI3 and PI4 pins are shared between the JTAG debug port function and the Serial Wire Debug Port function. The PI5 pin is shared between the JTAG debug port function and the SWV trace output function.

Table 3-1 SWJ-DP,ETM Debug Functions

SWJ-DP Pin Name	General-purpose Port Name	JTAG Debug Function		SW Debug Function	
		I / O	Explanation	I / O	Explanation
TMS / SWDIO	PI4	Input	JTAG Test Mode Selection	I / O	Serial Wire Data Input/Output
TCK / SWCLK	PI3	Input	JTAG Test Check	Input	Serial Wire Clock
TDO / SWV	PI5	Output	JTAG Test Data Output	(Output) (Note)	(Serial Wire Viewer Output)
TDI	PI6	Input	JTAG Test Data Input	-	-
$\overline{\text{TRST}}$	PI7	Input	JTAG Test $\overline{\text{RESET}}$	-	-
TRACECLK	PI2	Output	TRACE Clock Output		
TRACEDATA0	PI1	Output	TRACE DATA Output0		
TRACEDATA1	PI0	Output	TRACE DATA Output1		
TRACEDATA2	PH0	Output	TRACE DATA Output2		
TRACEDATA3	PH1	Output	TRACE DATA Output3		

Note: **When SWV function is enabled.**

After reset, PI3, PI4 and , PI5, PI6 and PI7 pins are configured as debug port function pins. The functions of other debug interface pins need to be programmed as required.

When using a low power consumption mode, take note of the following points.

Note: If PI4 and PI5 are configured as TMS/SWDIO and TDO/SWV, output continues to be enabled even in STOP1 mode regardless of the setting of the CGSTBYCR<DRVE> bit.

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

Table 3-2 Debug Interface Pins and Related Port Settings after Reset

Port Name (Bit Name)	Debug Function	Value of Related port settings after reset				
		Function (PxFR)	Input (PxIE)	Output (PxCR)	Pull-up (PxPUP)	Pull-down (PxPDN)
PI4	TMS/SWDIO	1	1	1	1	−
PI3	TCK/SWCLK	1	1	0	−	1
PI5	TDO/SWV	1	0	1	0	−
PI6	TDI	1	1	0	1	−
PI7	$\overline{\text{TRST}}$	1	1	0	1	−
PI2	TRACECLK	0	0	0	0	−
PI1	TRACEDATA0	0	0	0	0	−
PI0	TRACEDATA1	0	0	0	0	−
PH0	TRACEDATA2	0	0	0	0	−
PH1	TRACEDATA3	0	0	0	0	−

− : Don't care

3.5 Peripheral Functions in Halt Mode

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

3.6 Connection with a Debug Tool

3.6.1 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 1: Ensure that to mesure the power-consumption with debug tool connected in STOP1/STOP2 mode is prohibitted.

Note 2: Ensure that not to matain STOP2 mode for a long time with debug tool connected.

3.6.2 Important points of using debug interface pins used as general-purpose ports

When setting a debugging interface terminal to a general-purpose port by a user's program after reset release,after that the control from a debugging tool is impossible.

Please note that it is necessary to prepare for the structure which changes the general-purpose port to the debugging interface function by some kind of methods to connect a debugging tool again..

Table 3-3 Example Table of using debug interface pins

	Debug interface pins						
	$\overline{\text{TRST}}$	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 $\overline{\text{TRST}}$)	× (Note)	o	o	o	o	×	×
JTAG+TRACE	o	o	o	o	o	o	o
SW	×	×	×	o	o	×	×
SW+SWV	×	×	o	o	o	×	×
Debugging function disabled	×	×	×	×	×	×	×

o : Enabled × : Disabled (Usable as general-purpose port)

Note: For the treatment of the pin of which the $\overline{\text{TRST}}$ function is assigned, select the $\overline{\text{TRST}}$ function with the function register and set the pin to OPEN or "High level".

4. JTAG Interface

4.1 Overview

The TMPM366FDXBG/FYXBG/FWXBG provides a boundary-scan interface that is compatible with Joint Test Action Group (JTAG) specifications and uses the industry-standard JTAG protocol (IEEE Standard 1149.1・1990 <Includes IEEE Standard 1449.1a・1993>).

This chapter describes the JTAG interface, with the descriptions of boundary scan and the pins and signals used by the interface.

1. JTAG standard version

   IEEE Standard 1149.1・1990 (Includes IEEE Standard 1149.1a・1993)

2. JTAG instructions

   Standard instructions (BYPASS, SAMPLE/PRELOAD, EXTEST)

   HIGHZ instruction

   CLAMP instruction

   However, the SAMPLE/RELOAD instruction doesn't function because internal circuit reset starts as for TMPM366FDXBG/FYXBG/FWXBG while JTAG is operating.

3. IDCODE

   Not available

4. Pins excluded from boundary scan register (BSR)
   a. Oscillator circuit pins (X1, X2)
   b. DAC pin (DA0, DA1)
   c. JTAG control pins (BSC)
   d. Power supply/GND pins (including reference supply pin for ADC)
   e. TEST pins (FTEST3)
   f. Function pins ($\overline{\text{RESET}}$)
   g. Control pins (MODE,)

Note: As for PF0 pin is always pull-up, the pin is output high-level while in HIGHZ instruction.

Note: Please note the input level to the analog input pins.

4.2 Signal Summary and Connection Example

The JTAG interface signals are listed below.

- TDI JTAG serial data input
- TDO JTAG serial data output
- TMS JTAG test mode select
- TCK JTAG serial clock input
- $\overline{\text{TRST}}$ JTAG test reset input
- BSC ICE/JTAG test select input (compatible with the Enable signal)
  0: ICE, 1: JTAG

The TMPM366FDXBG/FYXBG/FWXBG supports debugging by connecting the JTAG interface with a JTAG-compliant development tool.

For information about debugging, refer to the specification of the development tool used.

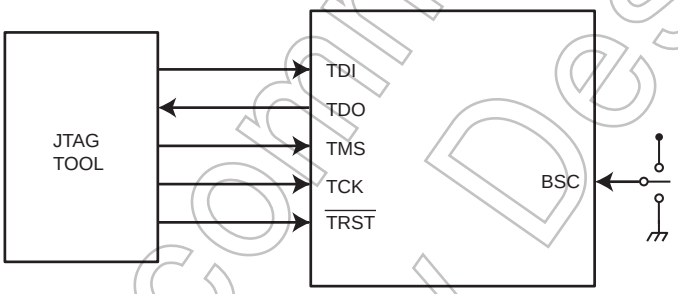


Figure 4-1 Example of connection with a JTAG development tool

Mode Setting Pin (BSC)	Operation mode
0	Set this pin to 0 except for Boundary Scan Mode. The TMPM366FDXBG/FYXBG/FWXBG operates as regular Debug Mode. Note: Debugging is not available if the internal BOOT is running.
1	The TMPM366FDXBG/FYXBG/FWXBG operates in Boundary Scan Mode.

4.3 Outline

With the evolution of ever-denser integrated circuits (ICs), surface-mounted devices, double-sided component mounting on printed-circuit boards (PCBs), and set-in recesses, in-circuit tests that depend upon physical contact like the connection of the internal board and chip has become more and more difficult to use. The more ICs have become complex, the larger and more difficult the test program became.

As one of the solutions, boundary-scan circuits started to be developed. A boundary-scan circuit is a series of shift register cells placed between the pins and the internal circuitry of the IC to which the said pins are connected. Normally, these boundary-scan cells are bypassed; when the IC enters test mode, however, the scan cells can be directed by the test program to pass data along the shift register path and perform various diagnostic tests. To accomplish this, the tests use the five signals, TCK, TMS, TDI, TDO and $\overline{\text{TRST}}$.

The JTAG boundary-scan mechanism (hereinafter referred to as JTAG mechanism in the chapter) allows testing of the connections between the processor, the printed circuit board to which it is attached, and the other components on the circuit board.

The JTAG mechanism cannot test the processor alone.

4.4 JTAG Controller and Registers

The processor contains the following JTAG controller and registers.

- Instruction register
- Boundary scan register
- Bypass register
- Device identification register
- Test Access Port (TAP) controller

JTAG basically operates to monitor the TMS input signal with the TAP controller state machine. When the monitoring starts, the TAP controller determines the test functionality to be implemented. This includes both loading the JTAG instruction register (IR) and beginning a serial data scan through a data register (DR), as shown in Table 4-1. As the data is scanned, the state of the TMS pin signals each new data word and indicates the end of the data stream. The data register is selected according to the contents of the instruction register.

4.5 Instruction Register

The JTAG instruction register includes four shift register-based cells. This register is used to select the test to be performed and/or the test data register to be accessed. As listed in Table 4-1, this instruction codes select either the boundary scan register or the bypass register.

Table 4-1 JTAG Instruction Register Bit Configuration

Instruction code (MSB to LSB)	Instruction	Selected data register
0000	EXTEST	Boundary scan register
0001	SAMPLE/PRELOAD	Boundary scan register
0100 to 1110	Reserved	Reserved
0010	HIGHZ	Bypass register
0011	CLAMP	Bypass register
1111	BYPASS	Bypass register

Figure 4-2 shows the format of the instruction register.

Figure 4-2 Instruction register

The instruction code is shifted out to the instruction register from the LSB.

Figure 4-3 Instruction Register Shift Direction

The bypass register is 1 bit wide. When the TAP controller is in the Shift-DR (bypass) state, the data on the TDI pin is shifted into the bypass register, and the bypass register output shifts to the data out on the TDO output pin.

In essence, the bypass register is an alternative route which allows bypassing of board-level devices in the serial boundary-scan chain, which are not required for a specific test. The logical location of the bypass register in the boundary-scan chain is shown in Figure 4-4.

Use of the bypass register speeds up access to the boundary scan register in the IC that remains active in the board-level test data path.

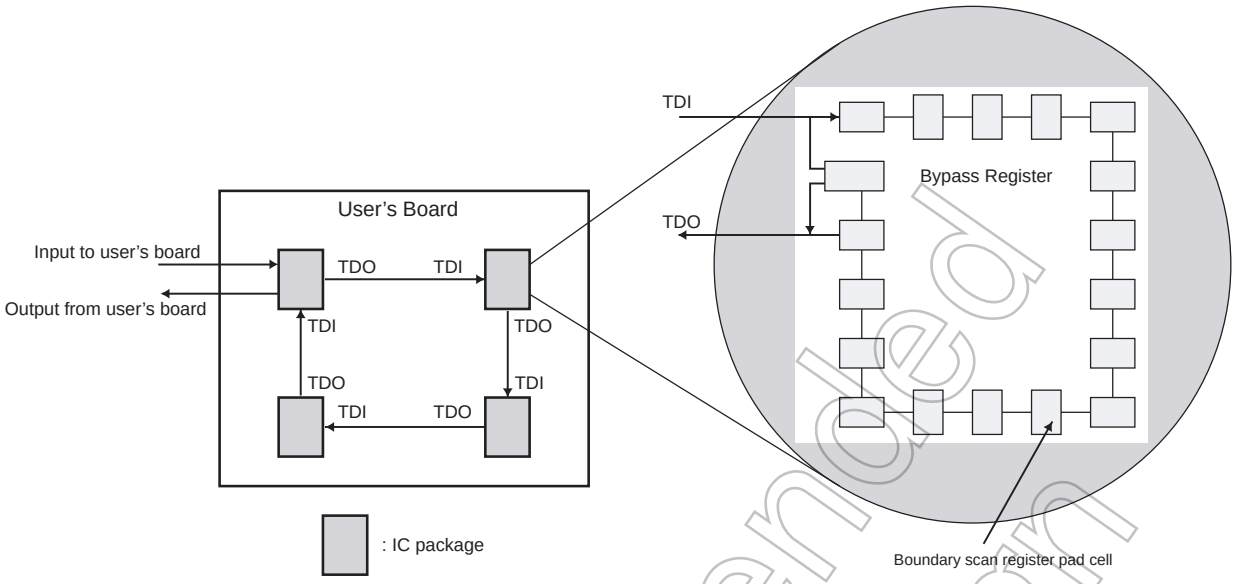


Figure 4-4 Bypass Register Operation

4.6 Boundary Scan Register

The boundary scan register provides all the inputs and outputs of the TMPM366FDXBG/FYXBG/FWXBG processor except some analog outputs and control signals. The pins of the TMPM366FDXBG/FYXBG/FWXBG allow any pattern to be driven by scanning the data into the boundary scan register in the Shift-DR state. Incoming data to the processor is examined by enabling the boundary scan register and shifting the data when the BSR is in the Capture-DR state.

The boundary scan register is a single, 231-bit-wide, shift register-based path containing cells connected to the input and output pads on the TMPM366FDXBG/FYXBG/FWXBG.

The TDI input is loaded to the LSB of the boundary scan register. The MSB of the boundary scan register is shifted out on the TDO output.

4.7 Test Access Port (TAP)

The Test Access Port (TAP) consists of the five signal pins: $\overline{\text{TRST}}$, TDI, TDO, TMS and TCK. These pins control a test by communicating the serial test data and instructions.

As Figure 4-5 shows, data is serially scanned into one of the three registers (instruction register, bypass register or boundary scan register) on the TDI pin, or it is scanned out from one of these three registers on the TDO pin.

The TMS input controls the state transitions of the main TAP controller state machine. The TCK input is a special test clock that allows serial JTAG data to be shifted synchronously, independent of any chip-specific or system clocks.

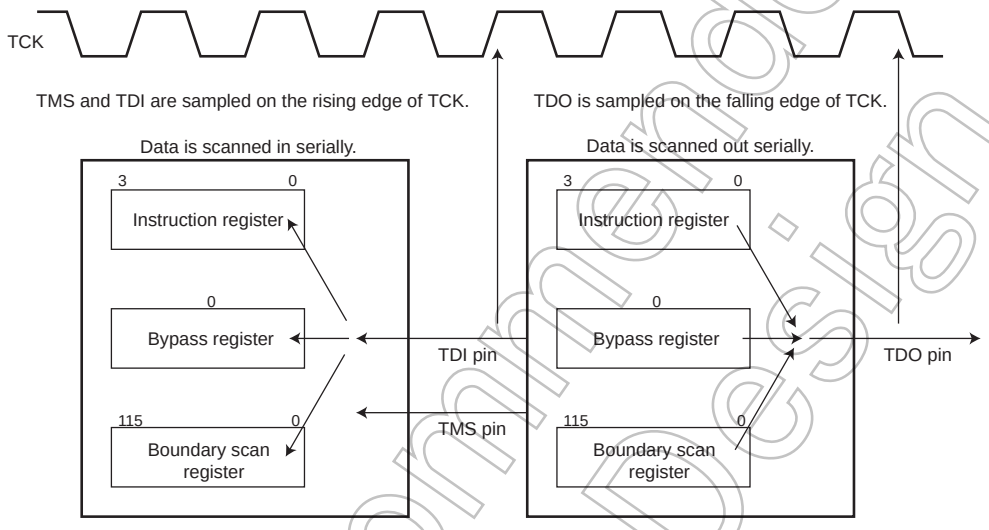


Figure 4-5 JTAG Test Access Port

Data on the TDI and TMS pins are sampled on the rising edge of the TCK input clock signal. Data on the TDO pin changes on the falling edge of the TCK clock signal.

4.8 TAP Controller

The processor incorporates the 16-state TAP controller stipulated in the IEEE JTAG specification.

4.9 Resetting the TAP Controller

The TAP controller state machine can be put into the Reset state by the following method.

Assertion of the $\overline{\text{TRST}}$ signal input (low) resets the TAP controller. After the processor reset state is released, keep the TMS input signal asserted through five consecutive rising edges of TCK input. Keeping TMS asserted maintains the Reset state.

4.10 State Transitions of the TAP Controller

The state transition diagram of the TAP controller is shown in Figure 4-6. Each arrow between states is labeled with a 1 or 0, indicating the logic value of TMS that must be set up before the rising edge of TCK to cause the transition.

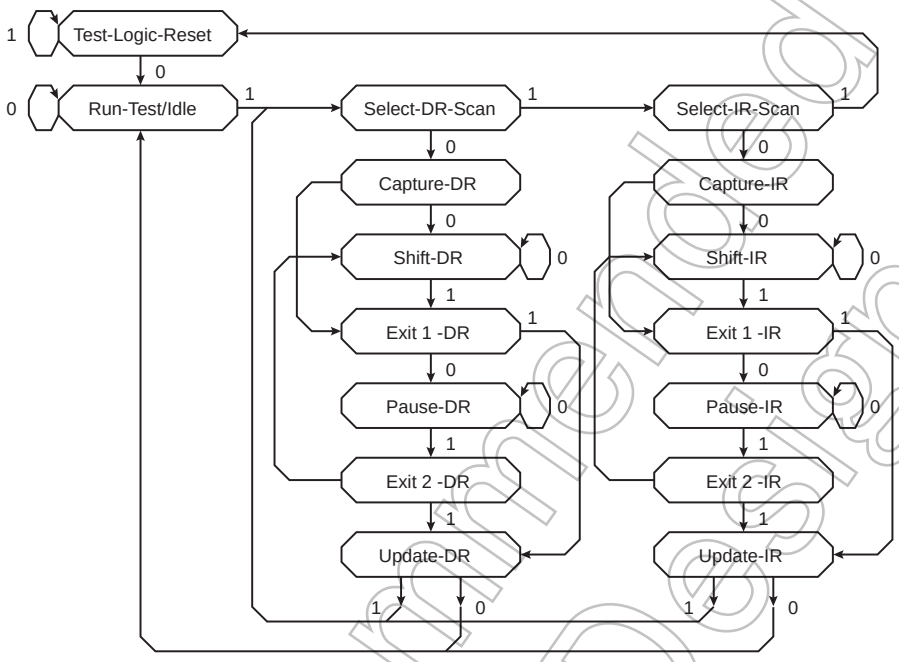


Figure 4-6 TAP Controller State Transition Diagram

The following paragraphs describe each of the controller states. The left column in Figure 4-6 is the data column, and the right column is the instruction column. The data column and instruction column reference the data register (DR) and the instruction register (IR), respectively.

- Test-Logic-Reset

  When the TAP controller is in the Reset state, the device identification register is selected by default. The MSB of the boundary scan register is cleared to 0 which disables the outputs.

  The TAP controller remains in this state while TMS is high. If TMS is held low while the TAP controller is in this state, then the controller moves to the Run-Test/Idle state.

- Run-Test/Idle

  In the Run-Test/Idle state, the IC is put in test mode only when certain instructions such as a built-in self test (BIST) instruction are present. For instructions that do not cause any activities in this state, all test data registers selected by the current instruction retain their previous states.

  The TAP controller remains in this state while TMS is held low. When TMS is held high, the controller moves to the Select-DR-Scan state.

- Select-DR-Scan

  This is a temporary controller state. Here, the IC does not execute any specific functions.

  If TMS is held low when the TAP controller is in this state, the controller moves to the Capture-DR state. If TMS is held high, the controller moves to the Select-IR-Scan state.

- Select-IR-Scan

  This is a temporary controller state. Here, the IC does not execute any specific functions.

  If TMS is held low when the TAP controller is in this state, the controller moves to the Capture-IR state. IF TMS is held high, the controller returns to the Test-Logic-Reset state.

- Capture-DR

  In this state, if the test data register selected by the current instruction has parallel inputs, then data is parallel-loaded into the shift portion of the data register. If the test data register does not have parallel inputs, or if data needs not be loaded into the selected data register, then the data register retains its previous state.

  If TMS is held low when the TAP controller is in this state, the controller moves to the Shift-DR state. If TMS is held high, the controller moves to the Exit 1-DR state.

- Shift-DR

  In this controller state, the test data register connected between TDI and TDO shifts data out serially.

  When the TAP controller is in this state, then it remains in the Shift-DR state if TMS is held low, or moves to the Exit 1-DR state if TMS is held high.

- Exit 1-DR

  This is a temporary controller state.

  If TMS is held low when the TAP controller is in this state, the controller moves to the Pause-DR state. If TMS is held high, the controller moves to the Update-DR state.

- Pause-DR

  This state allows the shifting of the data register selected by the instruction register to be temporarily suspended. Both the instruction register and the data register retain their current states.

  When the TAP controller is in this state, then it remains in the Pause-DR state if TMS is held low, or moves to the Exit 2-DR state if TMS is held high.

- Exit 2-DR

  This is a temporary controller state.

  When the TAP controller is in this state, it returns to the Shift-DR state if TMS is held low, or moves on to the Update-DR state if TMS is held high.

- Update-DR

  In this state, data is latched, on the rising edge of TCK, onto the parallel outputs of the data registers from the shift register path. The data held at the parallel output does not change while data is shifted in the associated shift register path.

  When the TAP controller is in this state, it moves to either the Run-Test/Idle state if TMS is held low, or the Select-DR-Scan state if TMS is held high.

- Capture-IR

  In this state, data is parallel-loaded into the instruction register. The data to be loaded is 0y0001. The Capture-IR state is used for testing the instruction register. Faults in the instruction register, if any, may be detected by shifting out the loaded data.

  When the TAP controller is in this state, it moves to either the Shift-IR state if TMS is held low, or the Exit 1-IR state if TMS is high.

- Shift-IR

  In this state, the instruction register is connected between TDI and TDO and shifts the captured data toward its serial output on the rising edge of TCK.

  When the TAP controller is in this state, it remains in the Shift-IR state if TMS is low, or moves to the Exit 1-IR state if TMS is high.

- Exit 1-IR

  This is a temporary controller state.

  When the TAP controller is in this state, it moves to either the Pause-IR state if TMS is held low, or the Update-IR state if TMS is held high.

- Pause-IR

  This state allows the shifting of the instruction register to be temporarily suspended. Both the instruction register and the data register retain their current states.

  When the TAP controller is in this state, it remains in the Pause-IR state if TMS is held low, or moves to the Exit 2-IR state if TMS is held high.

- Exit 2-IR

  This is a temporary controller state.

  When the TAP controller is in this state, it moves to either the Shift-IR state if TMS is held low, or the Update-IR state if TMS is held high.

- Update-IR

  This state allows the instruction previously shifted into the instruction register to be output in parallel on the rising edge of TCK. Then it becomes the current instruction, setting a new operational mode.

  When the TAP controller is in this state, it moves to either the Run-Test/Idle state if TMS is low, or the Select-DR-Scan state if TMS is high.

4.11 Boundary Scan Order

The below table shows the boundary scan order with respect to the processor signals.

TDI → 1 (PK3)→ 2(PK2)→ − →69(PI1)→70(PI2)→TDO

Table 4-2 JTAG Scan Order of the TMPM366FDXBG/FYXBG/FWXBG Processor Pins

No.	Pin Name	No.	Pin Name	No.	Pin Name	No.	Pin Name
	TDI						
1	PK3	21	PE0	41	PA4	61	PG4
2	PK2	22	PD0	42	PA3	62	PG5
3	PK1	23	PD1	43	PA2	63	PH4
4	PK0	24	PD2	44	PA1	64	PH3
5	PJ7	25	PD3	45	PA0	65	PH2
6	PJ6	26	PD4	46	PF7	66	PH1
7	PJ5	27	PD5	47	PF6	67	PH0
8	PJ4	28	PD6	48	PF5	68	PI0
9	PJ3	29	PD7	49	PF4	69	PI1
10	PJ2	30	PB7	50	PF3	70	PI2
11	PJ1	31	PB6	51	PF2		TDO
12	PJ0	32	PB5	52	PF1		
13	PE7	33	PB4	53	PF0		
14	PE6	34	PB3	54	PC2		
15	PE5	35	PB2	55	PC1		
16	PE4	36	PB1	56	PC0		
17	$\overline{\text{NMI}}$	37	PB0	57	PG0		
18	PE3	38	PA7	58	PG1		
19	PE2	39	PA6	59	PG2		
20	PE1	40	PA5	60	PG3		

4.12 Instructions Supported by the JTAG Controller Cells

This section describes the instructions supported by the JTAG controller cells of the TMPM366FDXBG/FYXBG/FWXBG.

1. EXTEST instruction

The EXTEST instruction is used for external interconnect tests. The EXTEST instruction permits BSR cells at output pins to shift out test patterns in the Update-DR state and those at input pins to capture test results in the Capture-DR state.

Typically, before EXTEST is executed, the initialization pattern is shifted into the boundary scan register using the SAMPLE/PRELOAD instruction. If the boundary scan register is not reset, indeterminate data will be transferred in the Update-DR state and bus conflicts between ICs may occur. Figure 4-7 shows data flow when the EXTEST instruction is selected.

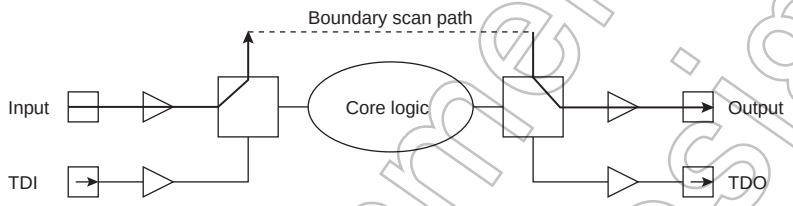


Figure 4-7 Test Data Flow when the EXTEST Instruction is Selected

The following steps describe the basic test procedure of the external interconnect test.

1. Reset the TAP controller to the Test-Logic-Reset state.
2. Load the instruction register with the SAMPLE/PRELOAD instruction. This causes the boundary scan register to be connected between TDI and TDO.
3. Reset the boundary scan register by shifting certain data in.
4. Load the test pattern into the boundary scan register.
5. Load the instruction register with the EXTEST instruction.
6. Capture the data applied to the input pin into the boundary scan register.
7. Shift out the captured data while simultaneously shifting the next test pattern in.
8. Send out the test pattern in the boundary scan register at the output on the output pin.

Repeat steps 6 to 8 for each test pattern.

2. SAMPLE/PRELOAD instruction

This instruction targets the boundary scan register between TDI and TDO. As its name implies, the SAMPLE/PRELOAD instruction provides two functions.

SAMPLE allows the input and output pads of an IC to be monitored. While it does so, it does not disconnect the system logic from the IC pins. SAMPLE is executed in the Capture-DR state. It is mainly used to capture the values of the IC's I/O pins on the rising edge of TCK during normal operation. Figure 4-8 shows the flow of data for the SAMPLE phase of the SAMPLE/PRELOAD instruction.

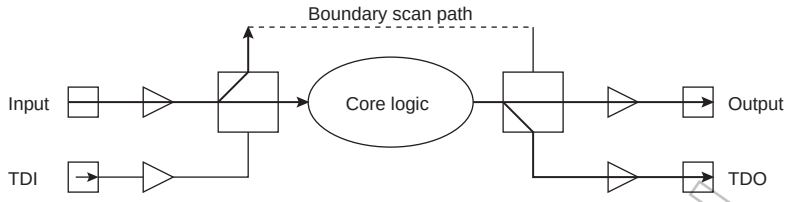


Figure 4-8 Test Data Flow while the SAMPLE is Selected

PRELOAD allows the boundary scan register to be reset before any other instruction is selected. For example, prior to selection of the EXTEST instruction, PRELOAD is used to load reset data into the boundary scan register. PRELOAD permits data shifting of the boundary scan register without interfering with the normal operation of the system logic. Figure 4-9 shows the data flow for the PRELOAD phase of the SAMPLE/PRELOAD instruction.

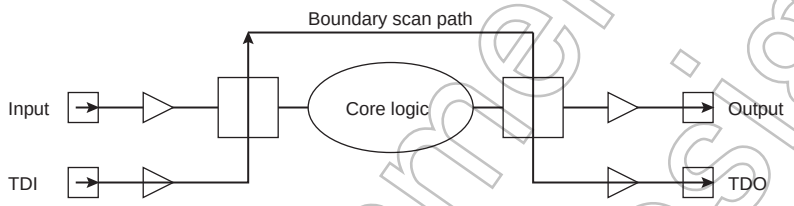


Figure 4-9 Test Data Flow while PRELOAD is Selected

3. BYPASS instruction

This instruction targets the bypass register between JTDI and JTDO. The bypass register provides the shortest serial path that bypasses the IC (between JTDI and JTDO) when the test does not require control or monitoring of the IC. The BYPASS instruction does not cause interference in the normal operation of the on-chip system logic. Figure 4-10 shows the data flow through the bypass register when the BYPASS instruction is selected.

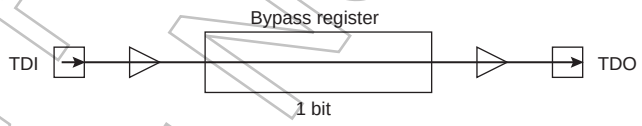


Figure 4-10 Test Data Flow when the BYPASS Instruction is Selected

4. CLAMP instruction

The CLAMP instruction outputs the value that boundary scan register is programmed according to the PRELOAD instruction, and execute Bypass operation.

The CLAMP instruction selects the bypass register between TDI and TDO.

5. HIGHZ instruction

The HIGHZ instruction disables the output of the internal logical circuits. When the HIGHZ instruction is executed, it places the 3-state output pins in the high-impedance state.

The HIGHZ instruction also selects the bypass register between TDI and TDO.

・ Notes

This section describes the cautions of the JTAG boundary-scan operations specific to the processor.

1. As for a PF0 pin is always pull-up, whenever HIGHZ is ordered, High is output.
2. Please note the input level to the analog input pins.
3. The JTAG circuit can be released from the reset state by either of the following two methods:

   Assert $\overline{\text{TRST}}$, initialize the JTAG circuit, and then deassertion $\overline{\text{TRST}}$.

   Supply the TCK signal for 5 or more clock pulses to TCK while pulling the TMS pin High.

5. Memory Map

5.1 Memory map

The memory maps for theTMPM366FDXBG/FYXBG/FWXBG are based on the ARM Cortex-M3 processor core memory map.

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

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

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

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

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

5.1.1 Memory map of the TMPM366FD

Figure 5-1shows the memory map of the TMPM366FD.

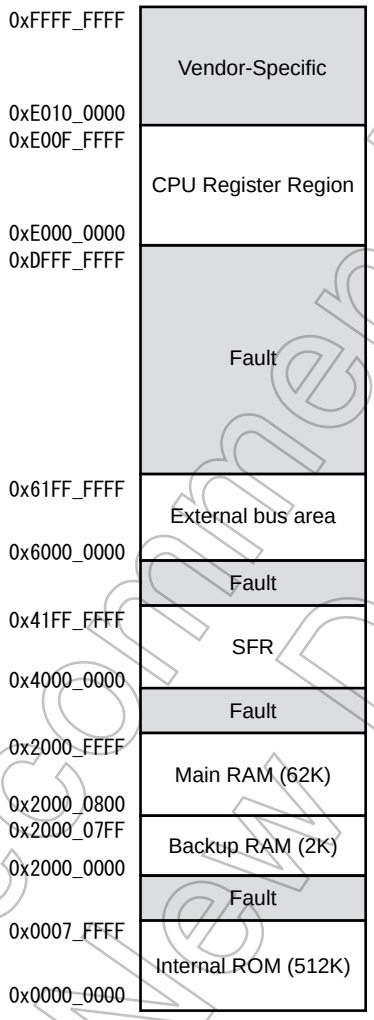


Figure 5-1 Memory Map (TMPM366FD)

5.1.2 Memory map of the TMPM366FY

Figure 5-2shows the memory map of the TMPM366FY.

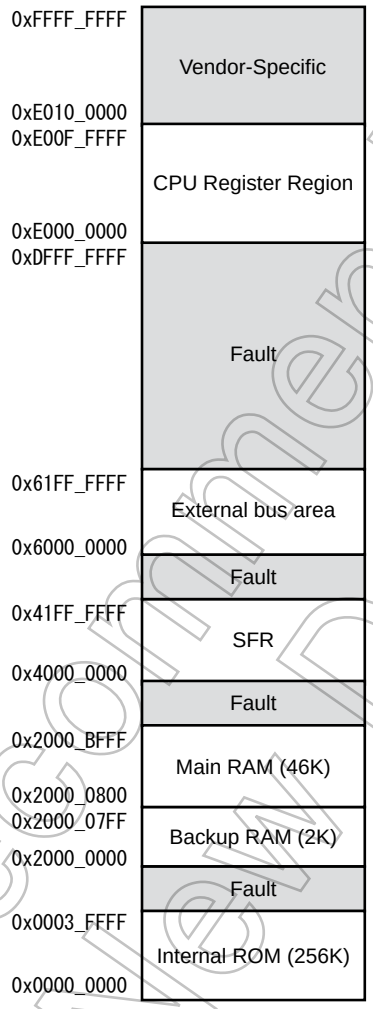


Figure 5-2 Memory Map (TMPM366FY)

5.1.3 Memory map of the TMPM366FW

Figure 5-3shows the memory map of the TMPM366FW.

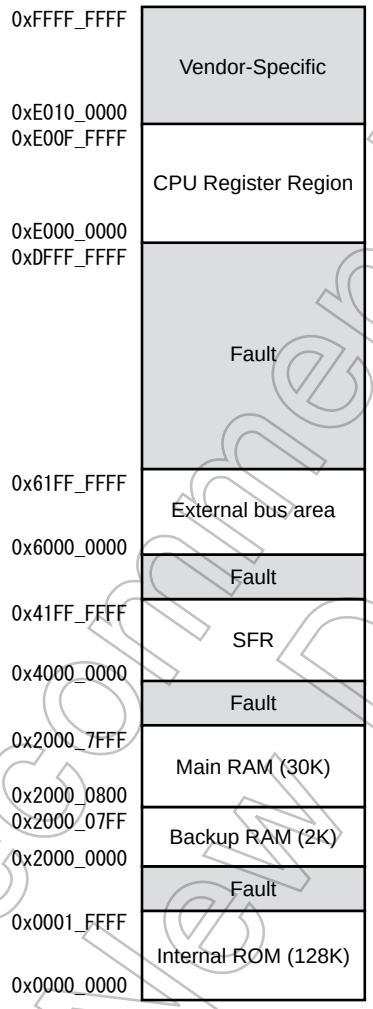


Figure 5-3 Memory Map (TMPM366FW)

5.2 SFR area detail

This section contains the list of addresses in the SFR area (0x4000_0000 through 0x41FF_FFFF) assigned to peripheral function.

Access to the Reserved areas in the Table 5-1, the address not specified and the Reserved area in each chapter are prohibited. As for the SFR area, the areas not specified in each chapter is read an undefined value. Writing this area is ignored.

Table 5-1 SFR area detail

Start Address	End Address	Peripheral	Reserved
0x4000_0000	0x4000_3FFF	DMAC(2 units, 4ch)	0x4000_0028 - 0x4000_002C 0x4000_0034 0x4000_1028 - 0x4000_102C 0x4000_1034
0x4000_4000	0x4000_7FFF	Reserved	
0x4000_8000	0x4000_9FFF	USB Device (1ch)	
0x4000_A000	0x4003_FFFF	Reserved	
0x4004_0000	0x4004_7FFF	SSP (3ch)	
0x4004_8000	0x4004_BFFF	UART (1ch)	
0x4004_C000	0x4004_FFFF	Reserved	
0x4005_0000	0x4005_3FFF	ADC(12ch)	0x4005_0064 - 0x4005_0073 0x4005_0F00 - 0x4005_0F8B
0x4005_4000	0x4005_BFFF	Reserved	
0x4005_C000	0x4005_CFFF	EBIF	0x4004_0230 - 0x4004_023F
0x4005_D000	0x400B_FFFF	Reserved	
0x400C_0000	0x400C_1FFF	PORT	
0x400C_2000	0x400C_3FFF	Reserved	
0x400C_4000	0x400C_5FFF	TMRB (10ch)	
0x400C_6000	0x400D_FFFF	Reserved	
0x400E_0000	0x400E_0FFF	I2C/SIO (2ch)	0x400E_0800 - 0x400E_0FFF
0x400E_1000	0x400E_1FFF	SIO/UART (2ch)	0x400E_1134 - 0x400E_1137
0x400E_2000	0x400F_0FFF	Reserved	
0x400F_1000	0x400F_1FFF	Reserved	
0x400F_2000	0x400F_2FFF	WDT	0x400F_2100 - 0x400F_2FFF
0x400F_3000	0x400F_3FFF	CG	0x400F_3100 - 0x400F_3FFF
0x400F_4000	0x41FF_EFFF	Reserved	
0x41FF_F000	0x41FF_F03F	FLASH	0x41FF_F000 - 0x41FF_F007 0x41FF_F014 - 0x41FF_F017 0x41FF_F024 - 0x41FF_F02B
0x41FF_F040	0x41FF_FFFF	Reserved	

6. Reset

The TMPM366FDXBG/FYXBG/FWXBG has four reset sources: an external reset pin ($\overline{\text{RESET}}$), a watchdog timer (WDT) and the setting <SYSRESETREQ> in the Application Interrupt and Reset Control Register.

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

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

6.1 Initial state

6.1.1 State before input reset

The internal circuits, register settings and pin status of the TMPM366FDXBG/FYXBG/FWXBG are undefined right after the power-on. The state continues until the $\overline{\text{RESET}}$ pin receives low level input after all the power supply voltage (DVDD3A, RVDD3, AVDD3 and DVDD3C) is applied.

6.2 Cold reset

The power-on sequence must include the time for the internal regulator to be stable and the reset time. In the TMPM366FDXBG/FYXBG/FWXBG, the internal regulator requires at least 1ms to be stable.

At cold reset, the external reset pin must be kept "Low" for a duration of time sufficiently long enough for the internal regulator to be stable.

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

Figure 6-1 shows the power-on sequence.

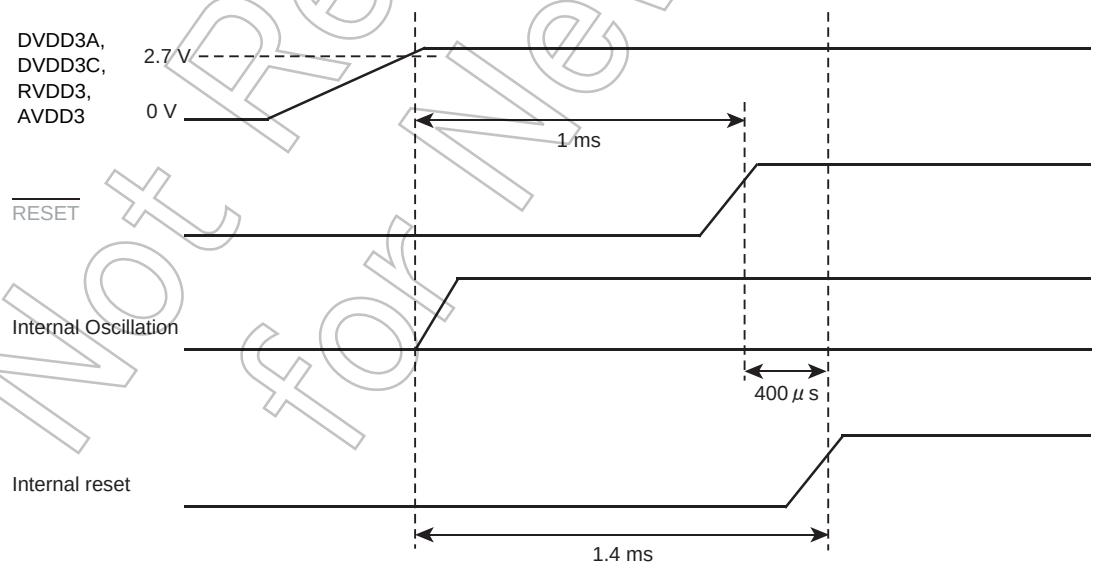


Figure 6-1 Cold Reset Sequence

Note 1: Turn on the power while the $\overline{\text{RESET}}$ pin is fixed to "Low". Release the $\overline{\text{RESET}}$ pin while all the power supplies are stabilized within operating voltage (DVDD3A/DVDD3B/RVDD3/AVDD3) and after an elapse of 1ms or more from while all the power supplies are stabilized within operating voltage.

Note 2: The above sequence is applied as well when restoring power.

6.3 Warm reset

6.3.1 Reset period

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

To reset the TMPM366FDXBG/FYXBG/FWXBG, assert the $\overline{\text{RESET}}$ signal (active low) for a minimum duration of 12 system clocks. And When the $\overline{\text{RESET}}$ signal input at "Low" level while in STOP2 mode, the $\overline{\text{RESET}}$ signal is fixed to "Low" more than 500μs as internal regulator stability time.

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

6.4 After reset

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

The processor core's system debug components (FPB, DWT, ITM) register, the clock generator's CGRSTFLG register and the FCSECBIT register are initialized by following factors. And FCSECBIT register is initialized by STOP2 mode released.

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

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

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

The factor of register initialization

Register	Factors	
CGRSTFLG	Cold reset	External Reset
FCSECBIT	Cold reset	STOP2 mode released
FPB, DWT, ITM	Cold reset	STOP2 mode released (Note

Note: When a debug tool is connecting, these registers are not initialized.

7. Watchdog Timer(WDT)

The watchdog timer (WDT) is for detecting malfunctions (runaways) of the CPU caused by noises or other disturbances and remedying them to return the CPU to normal operation.

If the watchdog timer detects a runaway, it generates a INTWDT interrupt or reset.

Note: INTWDT interrupt is a factor of the non-maskable interrupts (NMI).

Also, the watchdog timer notifies of the detecting malfunction to the external peripheral devices from the watchdog timer pin ($\overline{\text{WDTOUT}}$) by outputting "Low".

Note: This product does not have the watchdog timer out pin ($\overline{\text{WDTOUT}}$).

7.1 Configuration

Figure 7-1shows the block diagram of the watchdog timer.

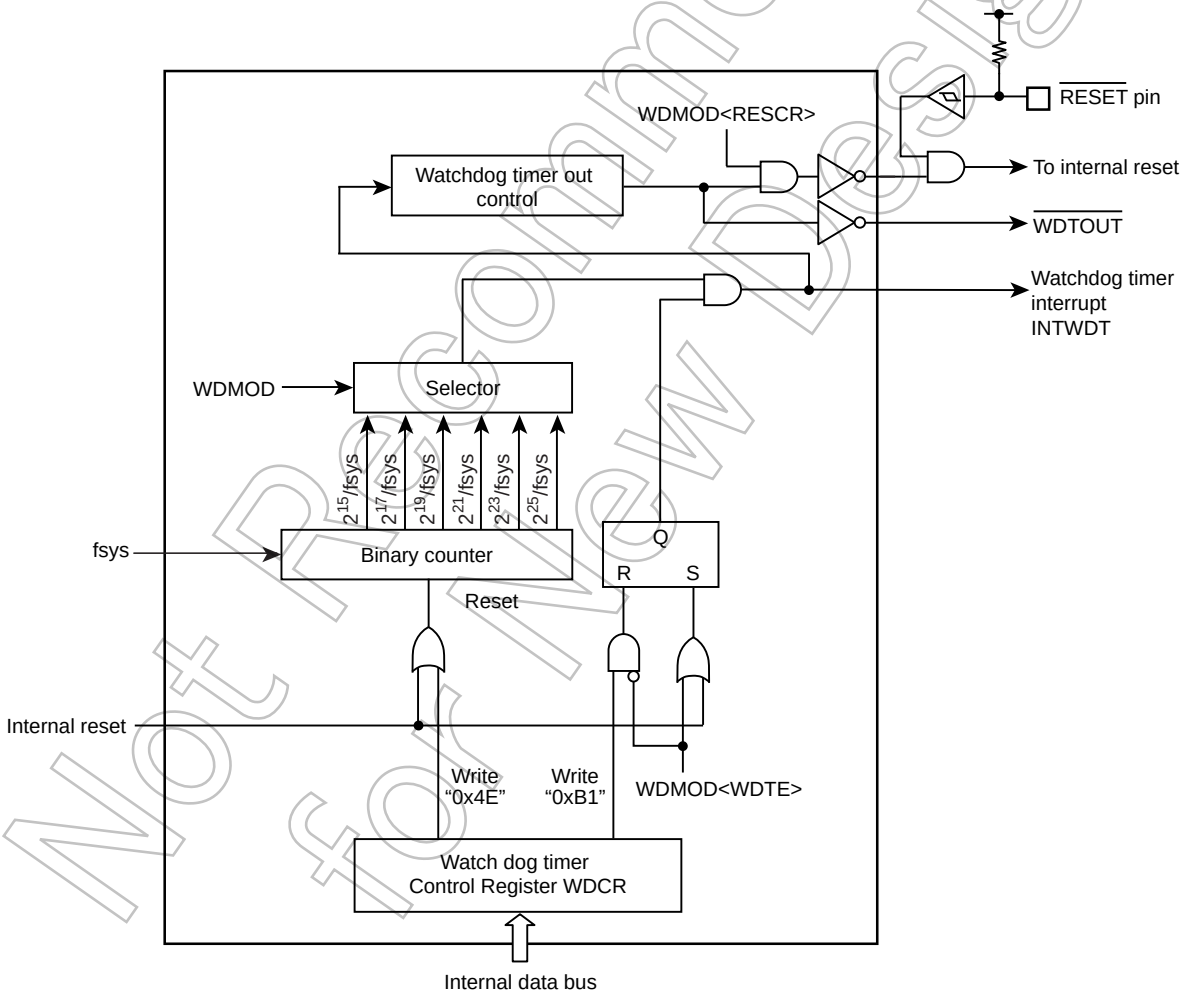


Figure 7-1 Block Diagram of the Watchdog Timer

7.2 Register

The followings are the watchdog timer control registers and addresses.

Base Address = 0x400F_2000

Register name		Address(Base+)
Watchdog Timer Mode Register	WDMOD	0x0000
Watchdog Timer Control Register	WDCR	0x0004

7.2.1 WDMOD(Watchdog Timer Mode Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	WDTE	WDTP			-	I2WDT	RESCR	-
After reset	1	0	0	0	0	0	1	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	WDTE	R/W	Enable/Disable control 0:Disable 1:Enable
6-4	WDTP[2:0]	R/W	Selects WDT detection time(Refer toTable 7-1) 000: 2^{15}/fsys 100: 2^{23}/fsys 001: 2^{17}/fsys 101: 2^{25}/fsys 010: 2^{19}/fsys 110:Setting prohibited. 011: 2^{21}/fsys 111:Setting prohibited.
3	−	R	Read as 0.
2	I2WDT	R/W	Operation when IDLE mode 0: Stop 1:In operation
1	RESCR	R/W	Operation after detecting malfunction 0: INTWDT interrupt request generates. (Note) 1: Reset
0	−	R/W	Write 0.

Note: INTWDT interrupt is a factor of the non-maskable interrupts (NMI).

Table 7-1 Detection time of watchdog timer (fc = 48MHz)

Clock gear value CGSYSCR<GEAR[2:0]>	WDMOD<WDTP[2:0]>					
	000	001	010	011	100	101
000 (fc)	0.68 ms	2.73 ms	10.92 ms	43.69 ms	174.76 ms	699.05 ms
100 (fc/2)	1.37 ms	5.46 ms	21.85 ms	87.38 ms	349.53 ms	1.40 s
101 (fc/4)	2.73 ms	10.92 ms	43.69 ms	174.76 ms	699.05 ms	2.80 s
110 (fc/8)	5.46 ms	21.85 ms	87.38 ms	349.53 ms	1.40 s	5.59 s
111 (fc/16)	10.92 ms	43.69 ms	174.76 ms	699.05 ms	2.80 s	11.18 s

7.2.2 WDCR (Watchdog Timer Control Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7-0	WDCR	W	Disable/Clear code 0xB1:Disable code 0x4E: Clear code Others:Reserved

7.3 Operations

7.3.1 Basic Operation

The Watchdog timer is consists of the binary counters that work using the system clock (fsys) as an input. Detecting time can be selected between 2^{15}, 2^{17}, 2^{19} , 2^{21}, 2^{23} and 2^{25} by the WDMOD<WDTP[2:0]>. The detecting time as specified is elapsed, the watchdog timer interrupt (INTWDT) generates, and the watchdog timer out pin ($\overline{\text{WDTOUT}}$) output "Low".

To detect malfunctions (runaways) of the CPU caused by noise or other disturbances, the binary counter of the watchdog timer should be cleared by software instruction before INTWDT interrupt generates. If the binary counter is not cleared, the non-maskable interrupt generates by INTWDT. Thus CPU detects malfunction (runway), malfunction countermeasure program is performed to return to the normal operation.

Additionally, it is possible to resolve the problem of a malfunction (runaway) of the CPU by connecting the watchdog timer out pin to reset pins of peripheral devices.

Note: This product does not include a watchdog timer out pin ($\overline{\text{WDTOUT}}$).

7.3.2 Operation Mode and Status

The watchdog timer begins operation immediately after a reset is cleared.

If not using the watchdog timer, it should be disabled.

The watchdog timer cannot be used as the high-speed frequency clock is stopped. Before transition to below modes, the watchdog timer should be disabled.In IDLE mode, its operation depends on the WDMOD <I2WDT> setting.

- STOP1 mode
- STOP2 mode

Also, the binary counter is automatically stopped during debug mode.

7.4 Operation when malfunction (runaway) is detected

7.4.1 INTWDT interrupt generation

In the Figure 7-2 shows the case that INTWDT interrupt generates (WDMOD<RESCR>="0").

When an overflow of the binary counter occurs, INTWDT interrupt generates. It is a factor of non-maskable interrupt (NMI). Thus CPU detects non-maskable interrupt and performs the countermeasure program.

The factor of non-maskable interrupt is the plural. CGNMIFLG identifies the factor of non-maskable interrupts. In the case of INTWDT interrupt, CGNMIFLG<NMIFLG0> is set.

When INTWDT interrupt generates, simultaneously the watchdog timer out ($\overline{\text{WDTOUT}}$) output "Low". $\overline{\text{WDTOUT}}$ becomes "High" by the watchdog timer clearing that is writing clear code 0x4E to the WDCR register.

Note: This product does not have the watchdog timer output pin($\overline{\text{WDTOUT}}$).

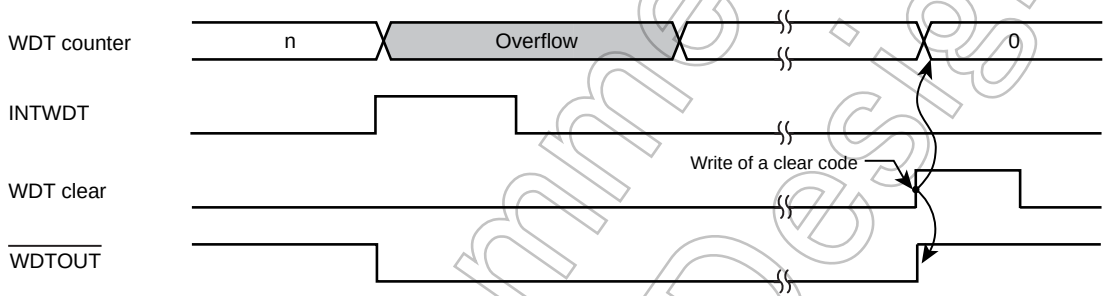


Figure 7-2 INTWDT interrupt generation

7.4.2 Internal reset generation

Figure 7-3 shows the internal reset generation (WDMOD<RESCR>="1").

MCU is reset by the overflow of the binary counter. In this case, reset status continues for 32 states. A clock is initialized so that input clock (fsys) is the same as a internal high-speed frequency clock (fosc). This means fsys = fosc.

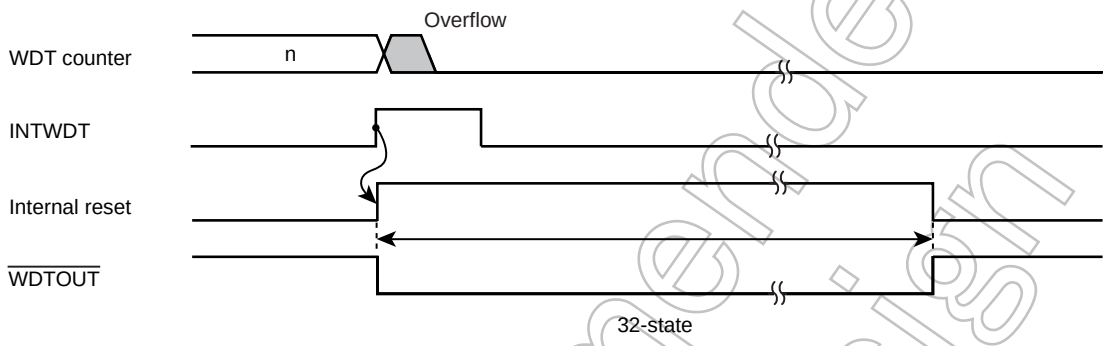


Figure 7-3 Internal reset generation

7.5 Control register

The watchdog timer (WDT) is controlled by two control registers WDMOD and WDCR.

7.5.1 Watchdog Timer Mode Register (WDMOD)

1. Specifying the detection time of the watchdog timer <WDTP[2:0]>.

   Set the watchdog timer detecting time to WDMOD<WDTP[2:0]>. After reset, it is initialized to WDMOD<WDTP[2:0]> = "000".

2. Enabling/disabling the watchdog timer <WDTE>.

   When resetting, WDMOD <WDTE> is initialized to "1" and the watchdog timer is enabled.

   To disable the watchdog timer to protect from the error writing by the malfunction, first <WDTE> bit is set to "0", and then the disable code (0xB1) must be written to WDCR register.

   To change the status of the watchdog timer from "disable" to "enable," set the <WDTE> bit to "1".

3. Watchdog timer out reset connection <RESCR>

   This register specifies whether WDTOUT is used for internal reset or interrupt. After reset, WDMOD<RESCR> is initialized to "1", the internal reset is generated by the overflow of binary counter.

7.5.2 Watchdog Timer Control Register(WDCR)

This is a register for disabling the watchdog timer function and controlling the clearing function of the binary counter.

7.5.3 Setting example

7.5.3.1 Disabling control

By writing the disable code (0xB1) to this WDCR register after setting WDMOD <WDTE> to "0," the watchdog timer can be disabled and the binary counter can be cleared.

		7	6	5	4	3	2	1	0	
WDMOD	←	0	−	−	−	−	−	−	−	Set <WDTE> to "0".
WDCR	←	1	0	1	1	0	0	0	1	Writes the disable code (0xB1).

7.5.3.2 Enabling control

Set WDMOD <WDTE> to "1".

		7	6	5	4	3	2	1	0	
WDMOD	←	1	−	−	−	−	−	−	−	Set <WDTE> to "1".

7.5.3.3 Watchdog timer clearing control

Writing the clear code (0x4E) to the WDCR register clears the binary counter and it restarts counting.

		7	6	5	4	3	2	1	0	
WDCR	←	0	1	0	0	1	1	1	0	Writes the clear code (0x4E).

7.5.3.4 Detection time of watchdog timer

In the case that 2^{21}/fsys is used, set "011" to WDMOD<WDTP[2:0]>.

		7	6	5	4	3	2	1	0
WDMOD	←	1	0	1	1	−	−	−	−

8. Clock/Mode control

8.1 Features

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.

The clock/mode control block has the following functions:

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

In addition to NORMAL mode, the TMPM366FDXBG/FYXBG/FWXBG can operate in three types of low power mode to reduce power consumption according to its usage conditions.

8.2 Registers

8.2.1 Register List

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

Base Address = 0x400F_3000

Register name		Address(Base+)
System control register	CGSYSCR	0x0000
Oscillation control register	CGOSCCR	0x0004
Standby control register	CGSTBYCR	0x0008
PLL selection register	CGPLLSEL	0x000C
Reserved	-	0x0010
Reserved	-	0x0014
USB clock control register	CGUSBCTL	0x0038
Protect register	CGPROTECT	0x003C

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

8.2.2 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).
19-18	−	R	Read as 0.
17-16	SCOSEL[1:0]	R/W	SCOUT out 00: Reserved 01: fsys/2 10: fsys 11: φT0 Enables to output the specified clock from SCOUT pin.
15-14	−	R	Read as 0.
13	−	R/W	Write "0"
12	FPSEL	−	fperiph 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 I/O.
7-3	−	R	Read as 0.
2-0	GEAR[2:0]	R/W	High-speed clock gear (fc) gear 000: fc 100: fc/2 001: Reserved 101: fc/4 010: Reserved 110: fc/8 011: Reserved 111: fc/16

8.2.3 CGOSCCR (Oscillation control register)

	31	30	29	28	27	26	25	24
bit symbol	WUODR							
After reset	1	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	WUODR				HWUPSEL	EHOSCSEL	OSCSEL	XEN2
After reset	0	0	0	0	0	0	0	1
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	XEN1
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	PLLON	WUEF	WUEON
After reset	0	0	1	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-20	WUODR[11:0]	R/W	Warm-up counter setup value. Setup the 16-bit timer for warm-up timer of upper 12-bits counter value.
19	HWUPSEL	R/W	High-speed warm-up clock. 0: internal (IHOSC) 1: external (f_{eosc}) Selects the source clock for warm-up counter. The selected clock is counted by warm-up counter.
18	EHOSCSEL	R/W	External oscillator. 0: input external clock (EHCLKIN) 1: external oscillator (EHOSC)
17	OSCSEL	R/W	High-speed oscillator (Note2) 0: internal high-speed oscillator (IHOSC) 1: external high-speed oscillator (feosc)
16	XEN2	R/W	Internal high-speed oscillator operation 0: Stop 1: Oscillation
15-12	−	R/W	Write "0".
11-10	−	R	Read as 0.
9	−	R/W	Write "0".
8	XEN1	R/W	External high-speed oscillator operation 0: Stop 1: Oscillation
7-3	−	R/W	Write "00110"
2	PLLON	R/W	PLL (multiplying circuit) operation 0: Stop 1: Oscillation
1	WUEF	R	Operation of warm-up timer (WUP) for oscillator 0: WUP finish 1: WUP active Enables to monitor the status of the warm-up timer.
0	WUEON	W	Operation of warm-up timer (WUP) for oscillator 0: don't care 1: WUP start Enables to start the warm-up timer. Read as 0.

Note 1: Refer to Section "8.3.4 Warm-up function" about the Warm-up setup.

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

Note 3: After setting CGOSCCR<PLLON>=1, operate Warm-up operation and then set CGPLLSEL<PLLSEL>=1.

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

Note 5: When using internal high-speed oscillator (IHOSC) as system clock, do not use PLL multiplying.

Note 6: When using internal high-speed oscillator (IHOSC), do not use it as system clock which high accuracy assurance is required.

8.2.4 CGSTBYCR (Standby control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	PTKEEP	DRVE
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
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
31-20	−	R	Read as 0.
19-18	−	R/W	Write "0" after reset.
17	PTKEEP	R/W	Keeps IO control signal in STOP2 mode 0: Control by port 1: Keep status when setting 0->1 (<PTKEEP> must be set before entering STOP2 mode)
16	DRVE	R/W	Pin status in STOP1 mode. 0: Inactive in STOP1 mode 1: Active in STOP1 mode
15-3	−	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 1: The reserved value is not set.

8.2.5 CGPLLSEL (PLL Selection Register)

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

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-1	PLLSET[14:0]	R/W	PLL multiplying value (Do not use except below) 0x381E: 8 multiplying 0x3816: 6 multiplying
0	PLLSEL	R/W	Use of PLL 0: f_{osc} 1: f_{PLL} / 2 (PLL use) Specifies use or disuse of the clock multiplied by the PLL. "fosc (internal high-speed oscillator)" is automatically set after reset.Resetting is required when using the PLL.

Note 1: Select PLL multiplying value which is shown in Table 8-1.

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

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

Note 4: When using internal high-speed oscillator (IHOSC) as system clock, do not use PLL multiplying.

8.2.6 CGUSBCTL (USB clock 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	-	-	-	-	-	-	USBCLKSEL	USBCLKEN-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	1	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-10	−	R	Read as 0.
9	USBCLKSEL	R/W	USB Source clock selection 0 : PLL clock 1 : External input clcok Selects source clock to USB device block.
8	USBCLKEN	R/W	USB Souce clock control 0: Clock disable 1: Clock enable
7-1	−	R	Read as 0
0		R/W	Write as Zero

Note 1: When <USBCLKSEL> is modified, it should be doing in source clock disable.
Note 2: <USBCLKSEL> and <USBCLLEN It can not be midified at the same time.

8.2.7 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 regiser can not be written.

8.3 Clock control

8.3.1 Clock Type

Each clock is defined as follows:

fosc	: Clock generated by internal oscillator. Clock input from the X1 and X2 pins.
f_{PLL}	: Clock multiplied by 8 or 16 by PLL.
fc	: Clock specified by CGPLLSEL<PLLSEL> (high-speed clock)
fgear	: Clock specified by CGSYSCR<GEAR[2:0]> (gear clock)
fsys	: Clock specified by same as fgear clock (system clock)
fperiph	: Clock specified by CGSYSCR<FPSEL>
φT0	: Clock specified by CGSYSCR<PRCK[2:0]> (prescaler clock)

The gear clock fgear 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

8.3.2 Initial Values after Reset

Reset operation initializes the clock configuration as follows.

internal high-speed oscillator	: oscillating
external high-speed oscillator	: stop
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.

fc = fosc
fsys = fosc
φT0 = fosc

8.3.3 Clock system Diagram

Figure 8-1 shows the clock system diagram.

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

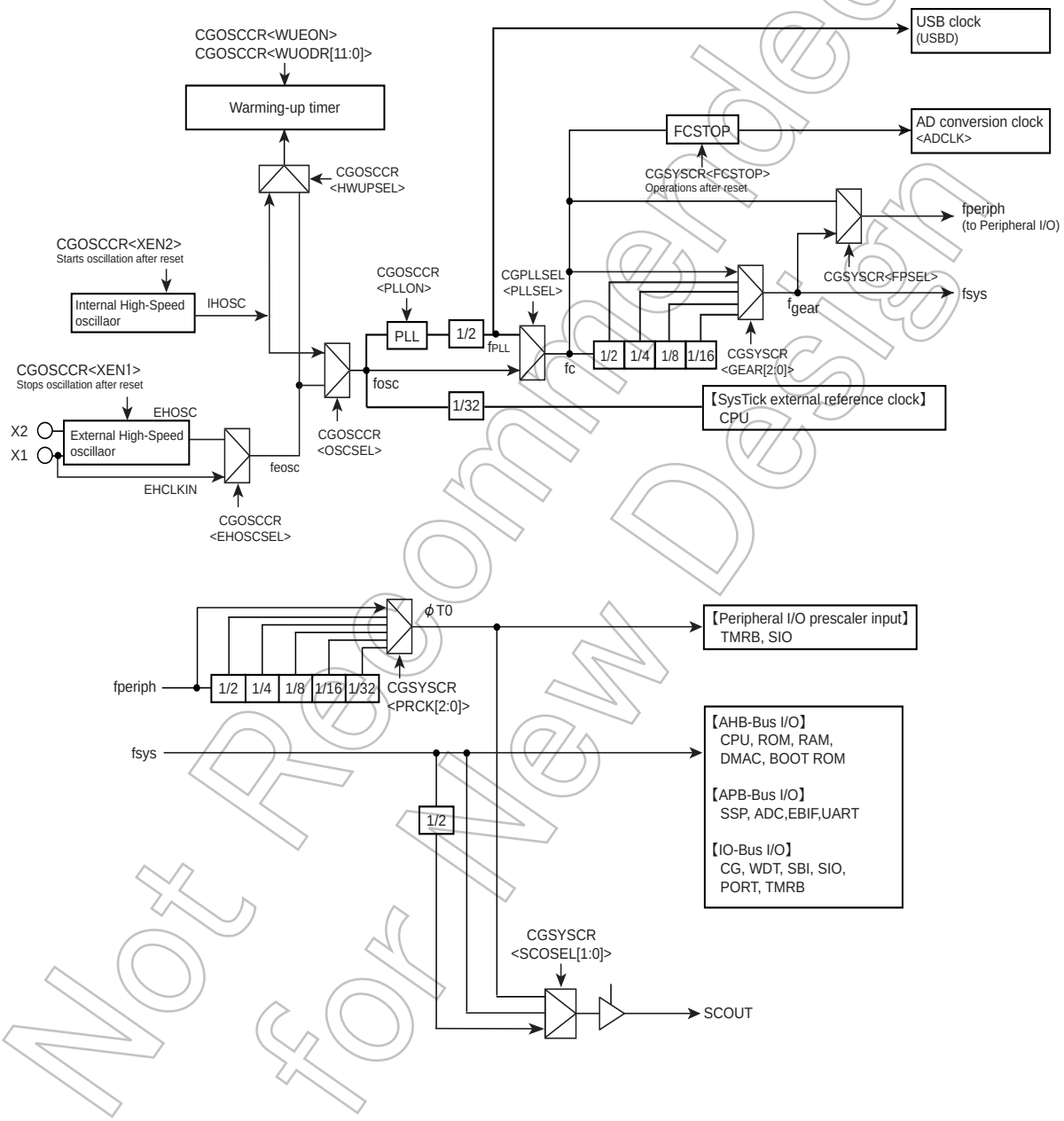


Figure 8-1 Clock Block Diagram

8.3.4 Warm-up function

The warm-up function secures the stability time for the oscillator and the PLL with the warm-up timer. When using external clock inputting, Warm-up function is not necessary when using stable external clock.

How to configure the warm-up function.

1. Specify the count up clock

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

2. Specify the warm-up counter value

   The warm-up time can be selected by setting the CGOSCCR<WUODR[11:0]>. The value can be calculated by following formula with round lower 4 bit off, set to the bit of <WUODR[11:0]>.

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

   <example 1> 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<WUODR[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>.

Note 1: Setting warm-up count value to CGOSCCR<WUDOR[11:0]>, wait until this value is reflected, then transit to low power consumption mode by executing a command "WFI"

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.

The example of warm-up function setup.

Table 8-1 <example> from STOP mode to NORMAL mode transition (internal high-speed oscillator is selected)

	CGOSCCR<WUODR[11:0]> = "0x9C4"	: Specify the warm-up time
↻	CGOSCCR<WUODR[11:0]> read	: Confirm warm-up time reflecting Repaet until the read data is "0x9C4".
	CGOSCCR<XEN2> = "1"	: Internal 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: It is not required the warm-up time in using the external clock to be stabled.

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

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

Note 4: When returning from STOP1/STOP2 mode, related bits CGPLLSEL<PLLSEL>, CGOSCCR<HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1> and <PLLON> are initialized in order to start internal high-speed oscillator and CGOSCCR<WUDOR[11:0]> is not initialized.

8.3.5 Clock Multiplication Circuit (PLL)

This circuit outputs the fpll clock that is multiplied by 6 or 8 of the high-speed oscillator output clock (fosc). As a result, the input frequency to oscillator can be low frequency, and the internal clock be made high-speed.

8.3.5.1 How to configure the PLL function

The PLL is disabled after reset.

To enable the PLL, set CGPLLSEL<PLLSET> to multiplying value. And set <PLLON> to "1" after 100μs for initialize time of PLL. After 100μs for lock-up time elapses, set CGPLLSEL<PLLSEL> to "1", f_{PLL} which is multiplied by 8 or 16 from fosc is used.

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

Note: When using internal high-speed oscillator (IHOSC) as system clock, do not use PLL multiplying.

As for the 8 or 16 multiplying value, only the following setting are permitted.

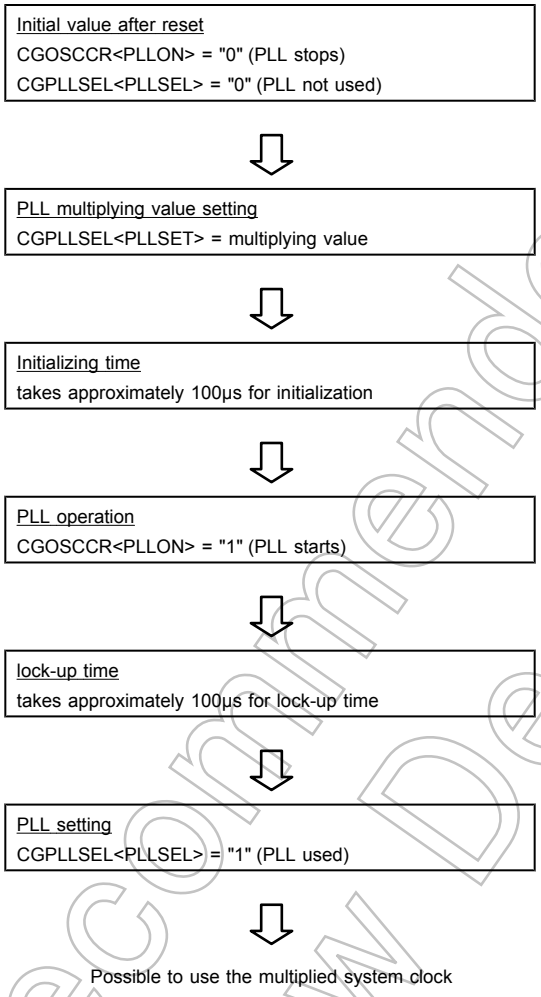
Multiplying	<PLLSEL>
8	0x381E
6	0x3816

8.3.5.2 Change PLL multiplying

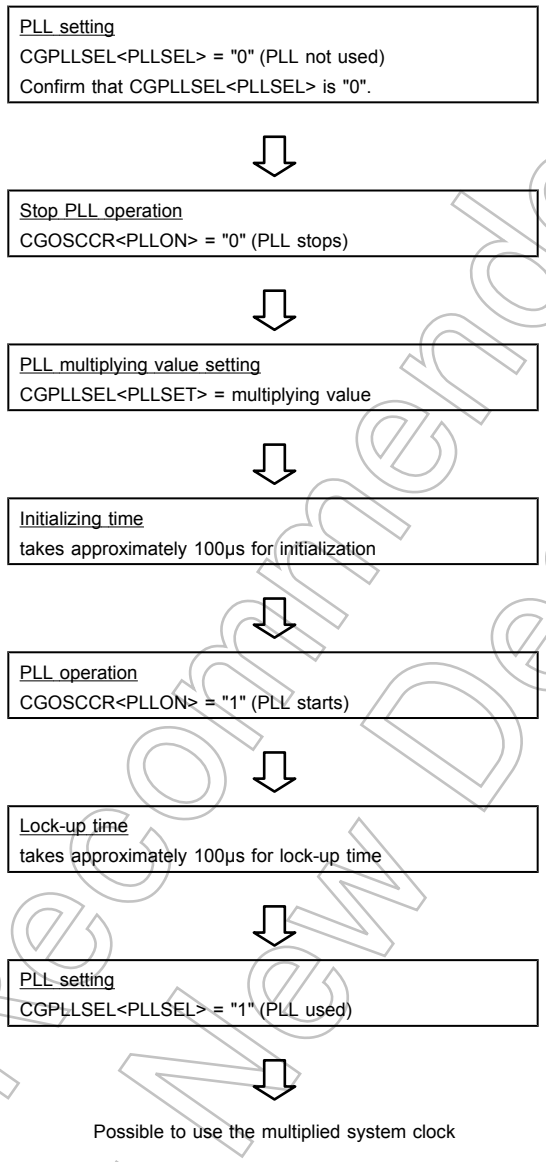
When number of multiplication is changed, firstly set "0" to CGPLLSEL<PLLSEL>. Secondly read CGPLLSEL<PLLSEL> to check the setting in which multiplication clock is not used (CGPLLSEL<PLLSEL>="0). Thirdly, set "0" to <PLLN> to stop PLL.

Modify CGPLLSEL<PLLSEL> to multiplying value. And set <PLLON> to "1" after 100μs for initialize time of PLL. After 100μs for lock-up time elapses, set CGPLLSEL<PLLSEL> to "1".

8.3.5.3 PLL setup sequence



8.3.5.4 Change PLL multiplying sequence



8.3.6 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.

When using internal high-speed oscillation clock, do not use it as system clock which high accuracy assurance is required.

When using external high-speed oscilltion clock, the PLL function can be used by multiplying.

Source clock		Frequency	Using PLL
Internal high-speed oscillation (IHOSC)		10MHz	can not use
External high-speed oscillation	Oscillator (EHOSC)	8 to 16MHz	Not use, 6 or 8 multiplying
	Input clock (EHCLKIN)	8 to 16, 48MHz	

Note: About multiplying of PLL and the external high-speed oscillation, refer to Table 8-2.

The clock two dividing can be used as a system clock and an ADC clock. The frequency that can be used respectively is as follows.

	System clock	ADC clock
Operation frequency (MHz)	1 to 48	40 (Max.)

The system clock can be divided by CGSYSCR<GEAR[2:0]>. Although the setting can be changed while operating, the actual switching takes place after a slight delay.

Table 8-2 shows the example of the operation frequency by the setting of PLL and the clock gear.

Table 8-2 The setting example of operation frequency depended on PLL multiplying and the clock gear setting

External oscillator (MHz)	External clock input (MHz)	PLL multiplying	Max. operation freq. (fc) (MHz)	A/DC Max. operation freq. (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	8	32	32	32	16	8	4	2	8	4	2	1	-
10	10		40	40	40	20	10	5	2.5	10	5	2.5	1.25	-
12	12		48	24 Note1)	48	24	12	6	3	12	6	3	1.5	-
-	48		48	24 Note1)	-	-	-	-	-	48	24	12	6	3
16	16	6	48	24 Note1)	48	24	12	6	3	16	8	4	2	1

↑ Initial value after reset

Note 1: Maximum Operating Frequency of A/DC(A/D convertor) is 40MHz, which is 2 dividing fc/2 frequency specified by ADCLK<ADCLK.> register.

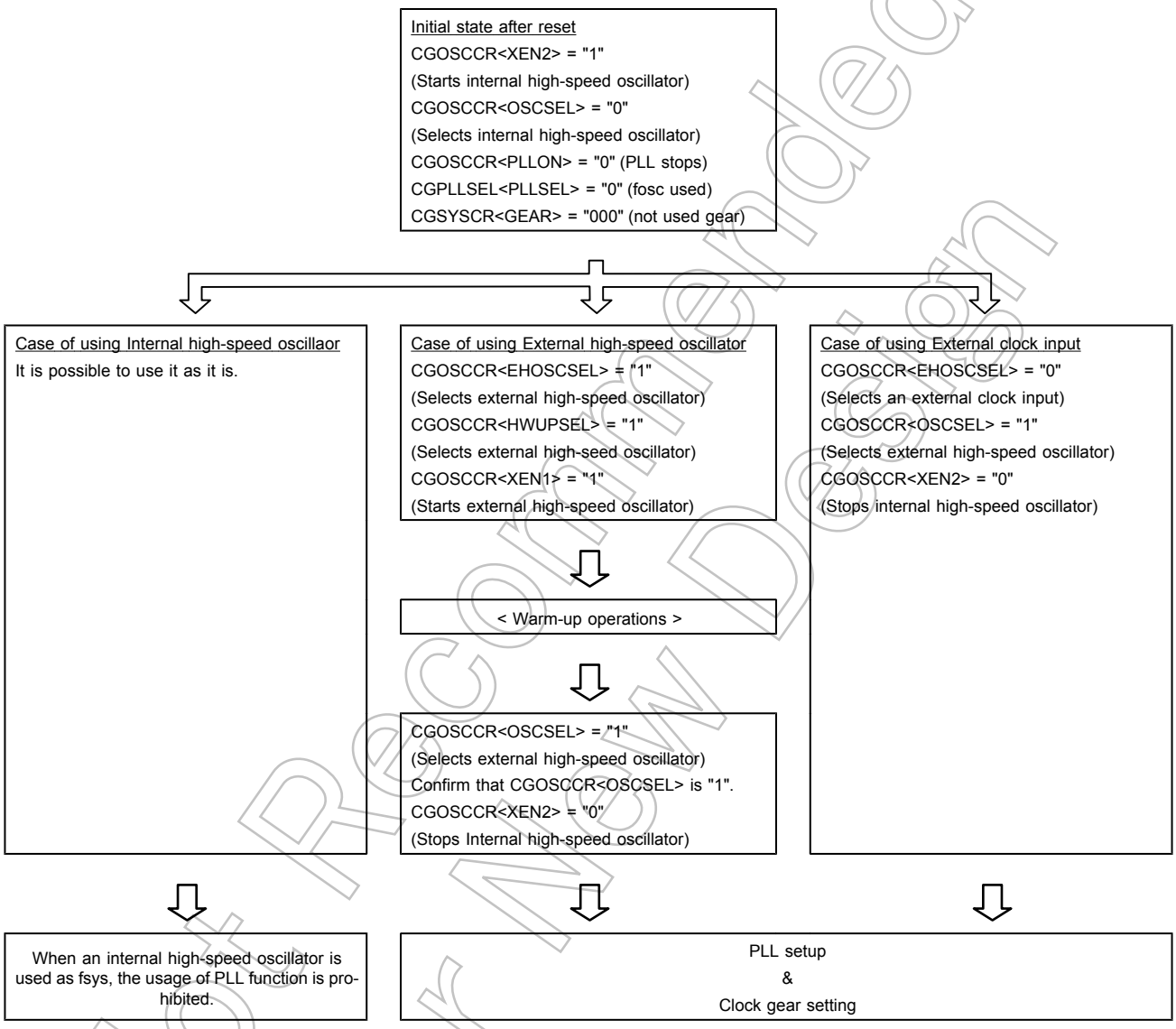
Note 2: When SysTick is used, do not use 1/16.

8.3.6.1 System Clock setting

The system clock can be selected by CGOSCCR. After the clock is selected, the PLL setting is done if necessary with CGPLLSEL and CGOSCCR. And, the clock gear is set with CGSYSCR.

The clock setup sequence is shown as follow.

Clock setup sequence



8.3.7 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.

8.3.8 System Clock Pin Output Function

The TX03 enables to output the system clock from a pin. The SCOUT pin can output the system clock fsys and fsys/2, and the prescaler input clock for peripheral function φT0.

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

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

The setting to use port as SCOUT pin, refer to "Input/Output port". The output clock is selected by setting the CGSYSCR<SCOSEL[1:0]>.

Table 8-3 shows the pin status in each mode when the SCOUT pin is set to the SCOUT output.

Table 8-3 SCOUT Output Status in Each Mode

Mode / SCOUT selection CGSYSCR	NORMAL	Low power consumption mode	
		IDLE	STOP1/STOP2 (Note)
<SCOSEL[1:0]> = "00"	Reserved		
<SCOSEL[1:0]> = "01"	Output the fsys/2 clock		
<SCOSEL[1:0]> = "10"	Output the fsys clock		
<SCOSEL[1:0]> = "11"	Output the φT0 clock	Fixed to "0" or "1".	

Note: To transit mode to STOP2, set port keep by CGSTBYCR<PTKEEP>="1" at first.

8.4 Modes and Mode Transitions

8.4.1 Mode Transitions

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

And TMPM366FDXBG/FYXBG/FWXBG has STOP2 mode that enables to reduce power consumption significantly by halting main voltage supply, retaining some functional operations.

Figure 8-2 shows a mode transition diagram.

For a detail of sleep-on-exit, refer to "Cortex-M3 Technical Reference Manual."

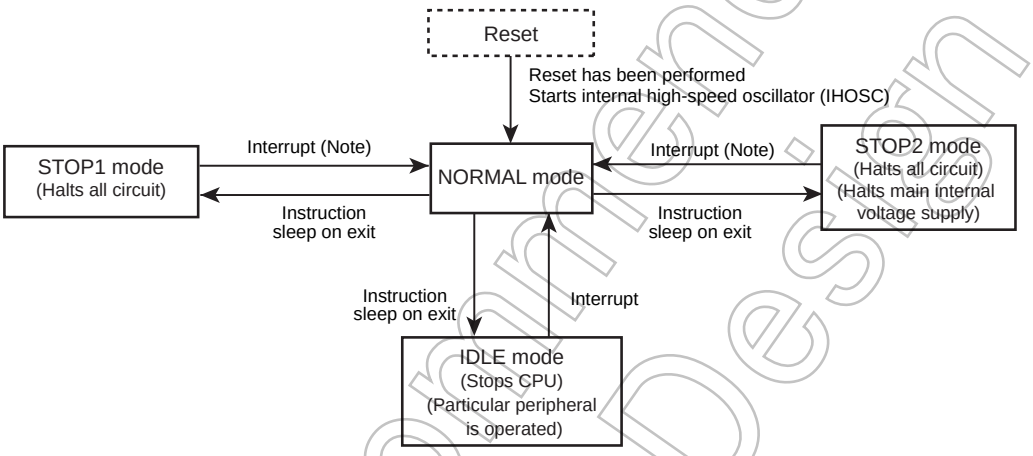


Figure 8-2 Mode Transition Diagram

Note 1: Returning from the STOP1/STOP2 mode, related bits <HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1>, <PLLON> of CGOSCCR and CGPLLSEL<PLLSEL> are initialized in order to start internal high-speed oscillator.

Note 2: 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.

Note 3: The warm-up is needed. The warm-up time must be set in NORMAL mode before entering to STOP mode. Regarding warm-up time, refer to "8.6.7 Warm-up".

8.5 Operation mode

8.5.1 NORMAL mode

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

It is shifted to the NORMAL mode after reset.

8.6 Low Power Consumption Modes

The TX03 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 TX03 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 TX03 does not support the low power consumption mode configured with the SLEEPDEEP bit in the Cortex-M3 core. Setting the <SLEEPDEEP> bit of the system control register is prohibited.

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

8.6.1 IDLE mode

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

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

- 16-bit timer/event counter (TMRB)
- Serial channel (SIO/UART)
- Serial bus interface (I2C/SIO)
- Analog Digital converter (ADC)
- Watch dog timer (WDT)

Note 1: Pay attention that the counter of watch dog timer function can not be cleared by CPU while in IDLE mode.

Note 2: Before entering IDLE mode, stop supplying the source clock for USBD (CGUSBCTL<USBCLKEN>="0").

8.6.2 STOP1 mode

All the internal circuits including the internal oscillator are brought to a stop in STOP1 mode.

The STOP1 mode enables to select the pin status by setting the CGSTBYCR<DRVE>. Table 8-3 shows the pin status in the STOP1 mode.

When returning from STOP1 mode, related bits CGPLLSEL<PLLSEL>, CGOSCCR<HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1> and <PLLON> are initialized and internal oscillator starts. After elapsed warm-up time, the NORMAL mode is operated.

The warm-up time is needed approximately 3μs elapses as stability time for an internal high-speed oscillator and the warm-up time set before entering the STOP1 mode.

When releasing the STOP1 mode by reset, the power-on counter activates a usual reset operation instead of the warm-up counter.

8.6.3 STOP2 mode

This mode halts main voltage supply, retaining some function operation. This enables to reduce power consumption significantly compares to STOP1 mode. After releasing the STOP2 mode, voltage is supplied to the halted blocks then an internal high-speed oscillator starts, and returns to NORMAL 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. Table 8-3 shows the pin status in the STOP2 mode

When returning from STOP2 mode, related bits CGPLLSEL<PLLSEL>, CGOSCCR<HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1> and <PLLON> are initialized and internal oscillator starts. After elapsed warm-up time, the NORMAL mode is operated.

The warm-up time is needed approximately 3μs elapses as stability time for an internal high-speed oscillator and the warm-up time set before entering the STOP2 mode.

When releasing the STOP2 mode by reset, the power-on counter activates a usual reset operation instead of the warm-up counter.

Note 1: Returning from the STOP1/STOP2 mode, it is required to set the warm-up time in NORMAL mode before entering STOP1/STOP2 mode. Regarding the warm-up time, refer to 8.6.8.1 and 8.6.8.2.

Note 2: Returning from the STOP1/STOP2 mode, related bits <HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1>, <PLLON> of CGOSCCR and CGPLLSEL<PLLSEL> are initialized in order to start internal high-speed oscillator. CGOSCCR<WUODR[11:0]> is not initialized.

Note 3: Because STOP2 mode halts internal voltage supply, more than 45μs time is required from transition to release. If the STOP2 mode is released during this time, internal voltage cannot operate normally.

Table 8-4 Pin States in the STOP1/STOP2 mode

Function	Pin Name	I/O	STOP1		STOP2	
			<DRVE> = 0	<DRVE> = 1	<PTKEEP> = 0	<PTKEEP> = 1
Control Pin	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC	Input	o	o	×	o
Oscillator	X1/EHCLKIN	Input (Note1	×	×	×	×
	X2	Output (Note1	"High" level output.	"High" level output.	×	×
PORT	PI3 to PI5 ($\overline{\text{TRST}}$, TDI, SWCLK/TCK) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Input	Depends on (PxIE[m])		×	Input holding, but Depends on (PxIE[m])
	PI4 (SWDIO/TMS) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Input	Depends on (PxIE[m])		×	Input holding, but Depends on (PxIE[m])
		Output	Enabled when data is valid. Disabled when data is invalid.		×	Output holding, but Depends on (PxCR[m])
	PI5, PI2, PI1, PI0, PH0, PH1 (TDO/SWV, TRACECLK, TRACEDATA0 to 3) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Output	Depends on (PxCR[m])		×	Output holding, but Depends on (PxCR[m])
	PG3, PG5, PK0, PK1, PE7, PE3, PF4, PF5, PH4, PJ7 (INT0 to 9) (Interrupt setting, case of PxFRn<PxmFn>="1" and PxIE<PxmIE>="1")	Input	o	o	×	o
	If using other than listed above	Input	×	Depends on (PxIE[m])	×	Input holding, but Depends on (PxIE[m])
		Output	×	Depends on (PxCR[m])	×	Output holding, but Depends on (PxCR[m])

o : Valid input or output.

× : Invalid input or output.

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

8.6.4 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 8-5 shows the mode setting in the <STBY[2:0]>.

Table 8-5 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]>.

8.6.5 Operational Status in Each Mode

Table 8-6 show the operational status in each mode.

Table 8-6 Operational Status in Each Mode

Block	NORMAL Internal high-speed oscillator use (IHOSC)	NORMAL External high-speed oscillator use (EHOSC)	IDLE Internal high-speed oscillator use (IHOSC)	IDLE External high-speed oscillator use (EHOSC)	STOP1 (Note 1)	STOP2 (Note 1)
Processor core	o	o	−	−	−	×
DMAC	o	o	o	o	−	×
EBIF	o	o	o	o	−	×
I/O port	o	o	o	o	o(Note 2)	Δ(Note 3)
SIO/UART	o	o	Δ	Δ	−	×
I2C/SIO	o	o	Δ	Δ	−	×
TMRB	o	o	Δ	Δ	−	×
WDT	o	o	Δ(Note 5)	Δ(Note 5)	−	×
SSP	o	o	o	o	−	×
USB	o	o	o	o	−	×
ADC	o	o	Δ	Δ	−	×
DAC	o	o	Δ	Δ	−	×
CG	o	o	o	o	o	o
PLL	o	o	Δ	Δ	−	×
External high-speed oscillator (EHOSC)	Δ	o	Δ	o	−	×
Internal high-speed oscillator (IHOSC)	o	o(Note 4)	o	o(Note 4)	−	×
Main RAM	o	o	o	o	o	×
Backup RAM	o	o	o	o	o	o

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: Before transit to STOP1/STOP2 mode, stop peripheral functions of "−"and "×". It is available to reduce leakage current by stopping reference voltage for AD converter or DA converter.

Note 2: The status depends on the CGSTBYCR<DRVE> bit.

Note 3: The status depends on the CGSTBYCR<PTKEEP> bit.

Note 4: After reset or STOP1/STOP2 mode released, clock is provided from internal high-speed oscillator.

Note 5: Pay attention that the counter of watch dog timer function can not be cleared by CPU while in IDLE mode.

8.6.6 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 8-7.

Table 8-7 Release Source in Each Mode

Low power consumption mode			IDLE	STOP1	STOP2 (Note3)
Release source	Interrupt	INT0 to 9 (Note2)	o	o	・Note3
		INTTB0 to 9	o	×	×
		INTCAP00 to 91	o	×	×
		INTRX0 to 1, INTTX0 to 1	o	×	×
		INTUART	o	×	×
		INTSBI0 to 1	o	×	×
		INTUSB	o	o	×
		INTUSBWKUP	o	o	×
		INTAD/INTADHP/INTADM0 to 1	o	×	×
		INTDMAC0TC, INTDMAC1TC, INTDMAC0ERR, INTDMAC1ERR	o	×	×
		INTSSP0 to 2	o	×	×
	SysTick Interrupt		o	×	×
	Non-Maskable Interrupt (INTWDT)		o	×	×
	Non-Maskable Interrupt ($\overline{\text{NMI}}$ pin)		o	o	o
	RESET ($\overline{\text{RESET}}$ pin)		o	o	o

o : Starts the interrupt handling after the mode is released. (The reset initializes the TMPM366FDXBG/FYXBG/FWXBG).
・ : Starts the reset interrupt handling after tje mode is released. (The reset initializes the TMPM366FDXBG/FYXBG/FWXBG).
× : Unavailable

Note 1: When STOP2 mode is released, set the active request of corresponding interrupt control register CGIMCGA,B,F to "rising edge". When "High" pulse width over than 500μs is detected, STOP2 mode is released. And releasing by NMI pin, input pulse which "Low" width is over than 500μs.

Note 2: When releasing from IDLE,STOP1 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.

Note 3: After STOP2 mode is released, reset operation initializes the internal supply voltage cut off block. But back-up module is not initialized.

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

- Release by interrupt request

  To release the low power consumption mode by an interrupt, the CPU must be set in advance to detect the interrupt. In addition to the setting in the CPU, the clock generator must be set to detect the interrupt to be used to release the STOP1 and STOP2 modes.

- Release by Non-Maskable Interrupt (NMI)

  INTWDT can only be used in the IDLE mode.

  The $\overline{\text{NMI}}$ pin can be used to release all the lower power consumption modes.

- Release by reset

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

Note that returning to the STOP1 and STOP2 mode by reset does not induce the automatic warm-up. Keep the reset signal valid until the oscillator operation becomes stable.

- Release by SysTick interrupt

  SysTick interrupt is used in only IDLE mode.

Refer to "Interrupts" for details.

8.6.7 Warm-up

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

In the mode transition from STOP1/STOP2 to the NORMAL, the warm-up counter and the internal oscillator are activated automatically. And 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<WUODR[11:0]> before executing the instruction to enter the STOP1/STOP2 mode.

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

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

Table 8-8 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 (Note)
STOP2 → NORMAL	Auto-warm-up (Note)

Note 1: Returning to NORMAL mode by reset does not induce the automatic warm-up. The reset as same as cold reset must be inputted.

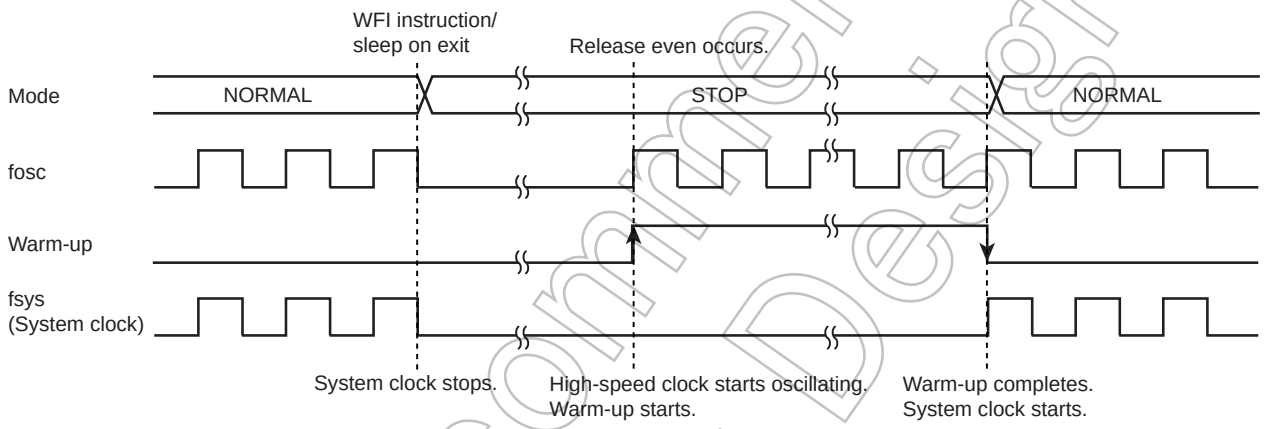
8.6.8 Clock Operations in Mode Transition

The clock operations in mode transition are described as follows.

8.6.8.1 Transition of operation modes: NORMAL → STOP1 → NORMAL

When returning to the NORMAL mode from the STOP1 mode, the warm-up is activated automatically. It is necessary for the warm-up to be set CGOSCCR<WUODR[11:0]>=0x119 in this case as a stability time (450μs) of on chip Flash ROM before entering the STOP1 mode.

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.



8.6.8.2 Transition of operation modes: NORMAL → STOP2 → NORMAL

When returning to the NORMAL mode from the STOP2 mode, the warm-up is activated automatically. It is necessary for the warm-up to be set CGOSCCR<WUODR[11:0]>=0x270 in this case as a stability time (1ms) of on chip Flash ROM before entering the STOP2 mode.

When releasing STOP2 mode by external interrupt input pin, the active request of CGIMGGA, B, C are set to rising edge. The rising edge of the corresponding external interrupt input pin releases STOP2 mode. At the falling edge of the corresponding external interrupt input pin after keeping "High" width over described belows, an internal oscillator starts. After the warm-up of a stability time (1m) for on chip Flash ROM, entering to NORMAL mode.

When releasing STOP2 mode by NMI, The falling edge of $\overline{\text{NMI}}$ pin releases STOP2 mode. At the rising edge of the $\overline{\text{NMI}}$ pin after keeping "Low" width over the period described below , an internal oscillator starts. After the warm-up of a stability time (1ms) for on chip Flash ROM, entering to NORMAL mode.

The pulse width which keeps an external interrupt pin or $\overline{\text{NMI}}$ pin is shown below.

- When he time (T1 and T3) from entering STOP2 to releasing STOP2 is more than 50μs

  Keep the width of sinal (T2 and T4) equal or more than 500μs.
- When he time (T1 and T3) from entering STOP2 to releasing STOP2 is less than 50μs

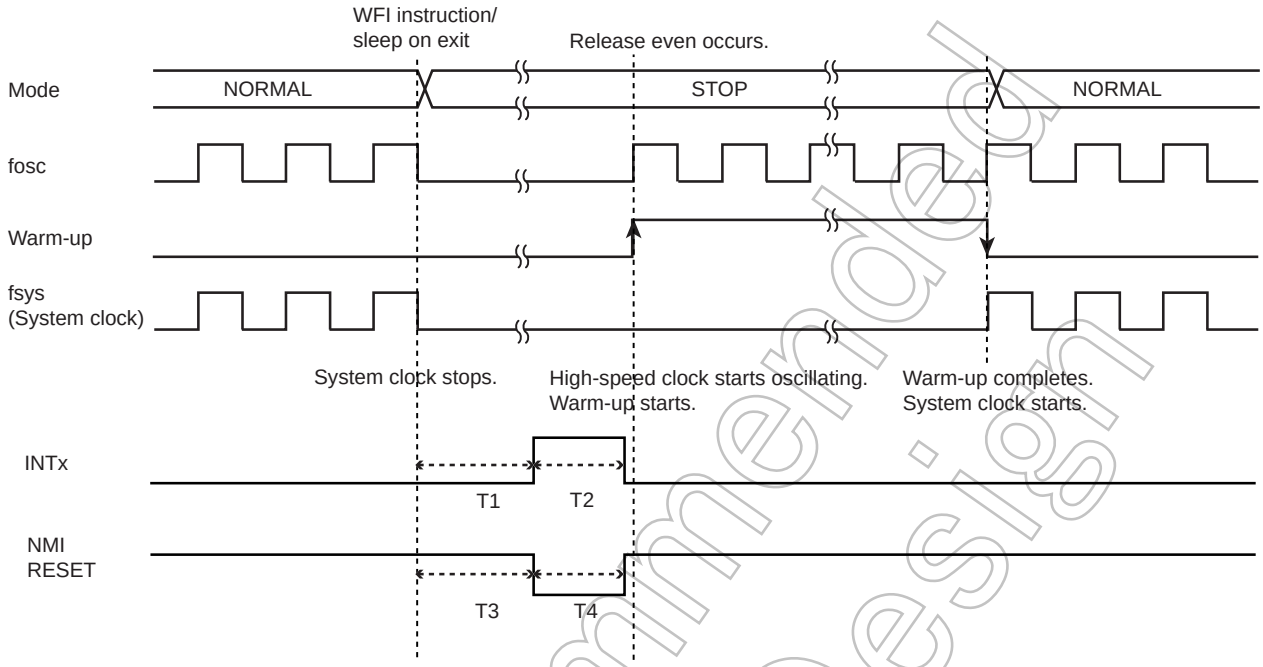
  Keep the width of sinal (T2 and T4) equal or more than 1500μs.

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

Even returning to the NORMAL mode by without reset, it would be branched to interrupt service routine of reset. After STOP2 mode is released, reset operation initializes the internal supply voltage cut off block. But back-up module is not initialized.

Note 1: When entering STOP2 mode, the level of corresponding interrupt must be "Low". And the level of $\overline{\text{NMI}}$ pin must be "High".

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



9. Exceptions

This chapter describes features, types and handling of exceptions.

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

9.1 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.

9.1.1 Exception Types

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

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

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

9.1.2 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 9.1.2.1
Handling by CPU	The CPU handles the exception request.	Section 9.1.2.2
Branch to ISR	The CPU branches to the corresponding interrupt service routine (ISR).	
Execution of ISR	Necessary processing is executed.	Section 9.1.2.3
Return from exception	The CPU branches to another ISR or returns to the previous program.	Section 9.1.2.4

9.1.2.1 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 "9.5 Interrupts".

(2) Exception detection

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

Table 9-1 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 9-1 Exception Types and Priority

No.	Exception type	Priority	Description
1	Reset	−3 (highest)	Reset pin, WDT or SYSRETREQ
2	Non-Maskable Interrupt	−2	$\overline{\text{NMI}}$ pin or WDT
3	Hard Fault	−1	Fault that cannot activate because a higher-priority fault is being handled or it is disabled
4	Memory Management	Configurable	Exception from the Memory Protection Unit (MPU) (Note 1) Instruction fetch from the Execute Never (XN) region
5	Bus Fault	Configurable	Access violation to the Hard Fault region of the memory map
6	Usage Fault	Configurable	Undefined instruction execution or other faults related to instruction execution
7~10	Reserved	−	
11	SVCall	Configurable	System service call with SVC instruction
12	Debug Monitor	Configurable	Debug monitor when the CPU is not faulting
13	Reserved	−	
14	PendSV	Configurable	Pendable system service request
15	SysTick	Configurable	Notification from system timer
16~	External Interrupt	Configurable	External interrupt pin or peripheral function (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 "9.5.1.5 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.

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

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

- Priority grouping

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

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

The Table 9-2 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 9-2 Priority grouping setting

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

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

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

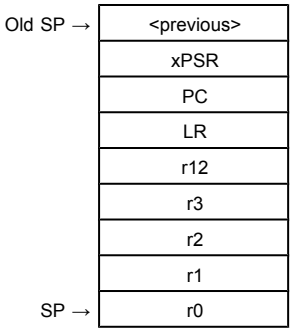
When an exception occurs, the CPU suspends the currently executing process and branches to the interrupt service routine. This is called "pre-emption".

(1) Stacking

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

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

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



(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 ~ 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

9.1.2.3 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 "9.5 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.

9.1.2.4 Exception exit

(1) Execution after returning from an ISR

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

- Tail-chaining

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

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

- Returning to the last stacked ISR

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

- Returning to the previous program

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

(2) Exception exit sequence

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

- Pop eight registers

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

- Load current active interrupt number

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

- Select SP

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

9.2 Reset Exceptions

Reset exceptions are generated from the following three sources.

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

- External reset pin

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

- Reset exception by WDT

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

- Reset exception by SYSRESETREQ

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

9.3 Non-Maskable Interrupts (NMI)

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

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

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

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

- Non-maskable interrupt by WDT

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

9.4 SysTick

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

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

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

Note: **In this product, fosc which is selected by CGOSCCR <OSCSEL> <EHOSCSEL> by 32 is used as external reference clock.**

9.5 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.

9.5.1 Interrupt Sources

9.5.1.1 Interrupt Route

Figure 9-1 shows an interrupt request route.

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

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

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

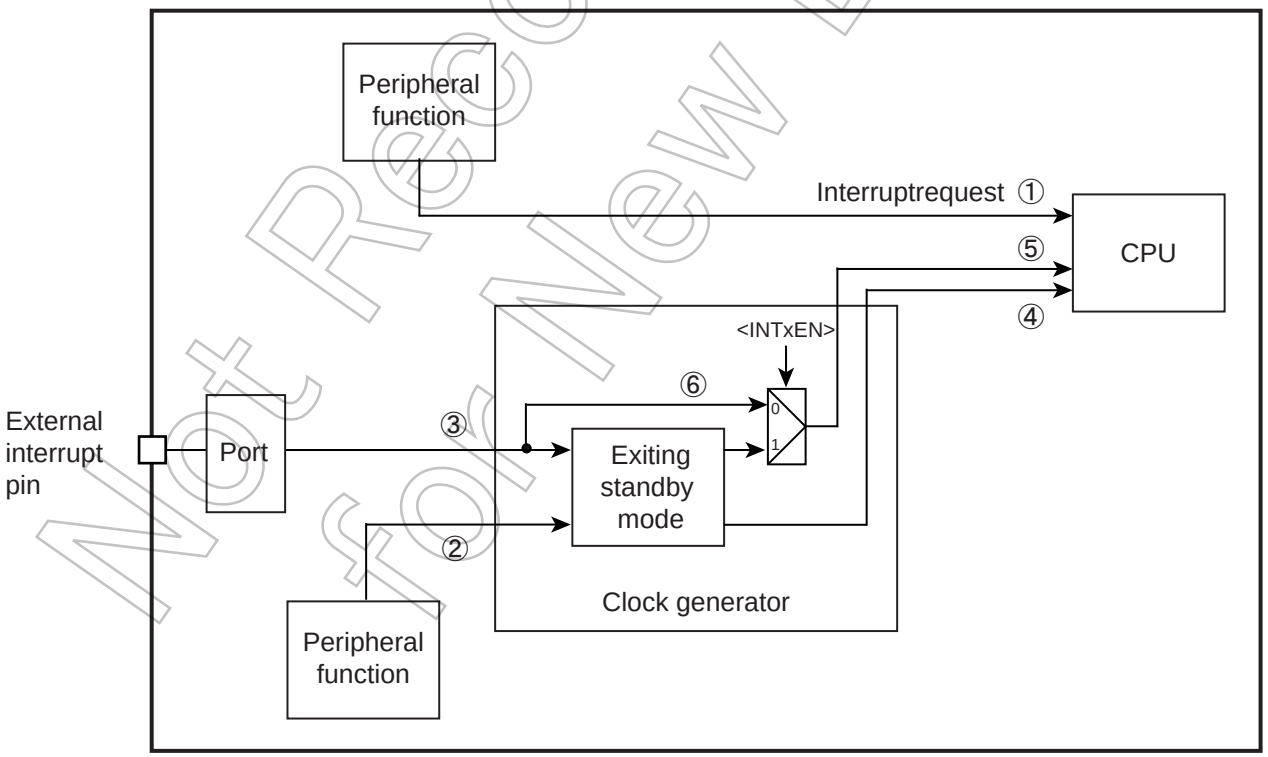


Figure 9-1 Interrupt Route

9.5.1.2 Generation

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

- From external pin

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

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

  See the chapter of each peripheral function for details.
- By setting Interrupt Set-Pending Register (forced pending)

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

9.5.1.3 Transmission

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

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

9.5.1.4 Precautions when using external interrupt pins

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

If input disabled (PxIE<PxmIE>="0"), inputs from external interrupt pins are "High". Also, if external interrupts are not used as a trigger to release standby (route 6 of "Figure 9-1 Interrupt Route"), input signals from the external interrupt pins are directly sent to the CPU. Since the CPU recognizes "High" input as an interrupt, interrupts occur if corresponding interrupts are enabled by the CPU as inputs are being disabled.

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

9.5.1.5 List of Interrupt Sources

Table 9-3 shows the list of interrupt sources.

Table 9-3 List of Interrupt Sources

No.	Interrupt Source		active level (Releasing standby and interrupt)	CG interrupt mode control register
0	INT0	Interrupt pin 0	Selectable	CGIMCGA
1	INT1	Interrupt pin 1		
2	INT2	Interrupt pin 2		
3	INT3	Interrupt pin 3		
4	INT4	Interrupt pin 4		CGIMCGB
5	INT5	Interrupt pin 5		
6	INT6	Interrupt pin 6		
7	INT7	Interrupt pin 7		
8	INTRX0	Serial channel 0 reception interrupt		
9	INTTX0	Serial channel 0 transmission interrupt		
10	INTRX1	Serial channel 1 reception interrupt		
11	INTTX1	Serial channel 1 transmission interrupt		
12	INTUSBWKUP	USB Wake-up interrupt	Falling	CGIMCGC
13	-	Reserved		
14	INTSBI0	Serial bus interface 0 interrupt		
15	INTSBI1	Serial bus interface 1 interrupt		
16	INTADHP	Highest priority AD conversion complete interrupt		
17	INTAD	AD conversion completion interrupt		
18	INTADM0	AD conversion monitoring function 0 interrupt		
19	INTADM1	AD conversion monitoring function 1 interrupt		
20	INTTB0	16-bit timer / event counter 0 match detection interrupt		
21	INTTB1	16-bit timer / event counter 1 match detection interrupt		
22	INTTB2	16-bit timer / event counter 2 match detection interrupt		
23	INTTB3	16-bit timer / event counter 3 match detection interrupt		
24	INTTB4	16-bit timer / event counter 4 match detection interrupt		
25	INTTB5	16-bit timer / event counter 5 match detection interrupt		
26	INTTB6	16-bit timer / event counter 6 match detection interrupt		
27	INTTB7	16-bit timer / event counter 7 match detection interrupt		
28	INTTB8	16-bit timer / event counter 8 match detection interrupt		
29	INTTB9	16-bit timer / event counter 9 match detection interrupt		
30	INTUSB	USB interrupt		
31	INTSSP2	SSP 2 interrupt		
32	-	Reserved		
33	-	Reserved		
34	INTUSBPON	USB Power On connection detection interrupt	Selectable	CGIMCGC
35	INTUART	UART interrupt		
36	INTCAP00	16-bit timer / event counter 0 input capture interrupt 0		
37	INTCAP01	16-bit timer / event counter 0 input capture interrupt 1		
38	INTCAP10	16-bit timer / event counter 1 input capture interrupt 0		
39	INTCAP11	16-bit timer / event counter 1 input capture interrupt 1		
40	INTCAP20	16-bit timer / event counter 2 input capture interrupt 0		
41	INTCAP21	16-bit timer / event counter 2 input capture interrupt 1		
42	INTCAP30	16-bit timer / event counter 3 input capture interrupt 0		

Table 9-3 List of Interrupt Sources

No.	Interrupt Source		active level (Releasing standby and interrupt)	CG interrupt mode control register
43	INTCAP31	16-bit timer / event counter 3 input capture interrupt 1		
44	INTCAP40	16-bit timer / event counter 4 input capture interrupt 0		
45	INTCAP41	16-bit timer / event counter 4 input capture interrupt 1		
46	INTCAP50	16-bit timer / event counter 5 input capture interrupt 0		
47	INTCAP51	16-bit timer / event counter 5 input capture interrupt 1		
48	INTCAP60	16-bit timer / event counter 6 input capture interrupt 0		
49	INTCAP61	16-bit timer / event counter 6 input capture interrupt 1		
50	INTCAP70	16-bit timer / event counter 7 input capture interrupt 0		
51	INTCAP71	16-bit timer / event counter 7 input capture interrupt 1		
52	INTCAP80	16-bit timer / event counter 8 input capture interrupt 0		
53	INTCAP81	16-bit timer / event counter 8 input capture interrupt 1		
54	INTCAP90	16-bit timer / event counter 9 input capture interrupt 0		
55	INTCAP91	16-bit timer / event counter 9 input capture interrupt 1		
56	INT8	Interrupt pin 8	Selectable	CGIMCGC
57	INT9	Interrupt pin 9		
58	INTSSP1	SSP 1 interrupt		
59	INTSSP0	SSP0 interrupt		
60	INTDMAC0TC	DMA terminal count status interrupt 0		
61	INTDMAC1TC	DMA terminal count status interrupt 1		
62	INTDMAC0ERR	DMA error status interrupt 0		
63	INTDMAC1ERR	DMA error status interrupt 1		

9.5.1.6 Active level

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

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

If an interrupt source is used for clearing a standby mode, setting the relevant clock generator register is also required. Enable the CGIMCGx<INTxEN> bit and specify the active level in the CGIMCGx<EMCGx> bits. You must set the active level for interrupt requests from each peripheral function as shown in Table 9-3.

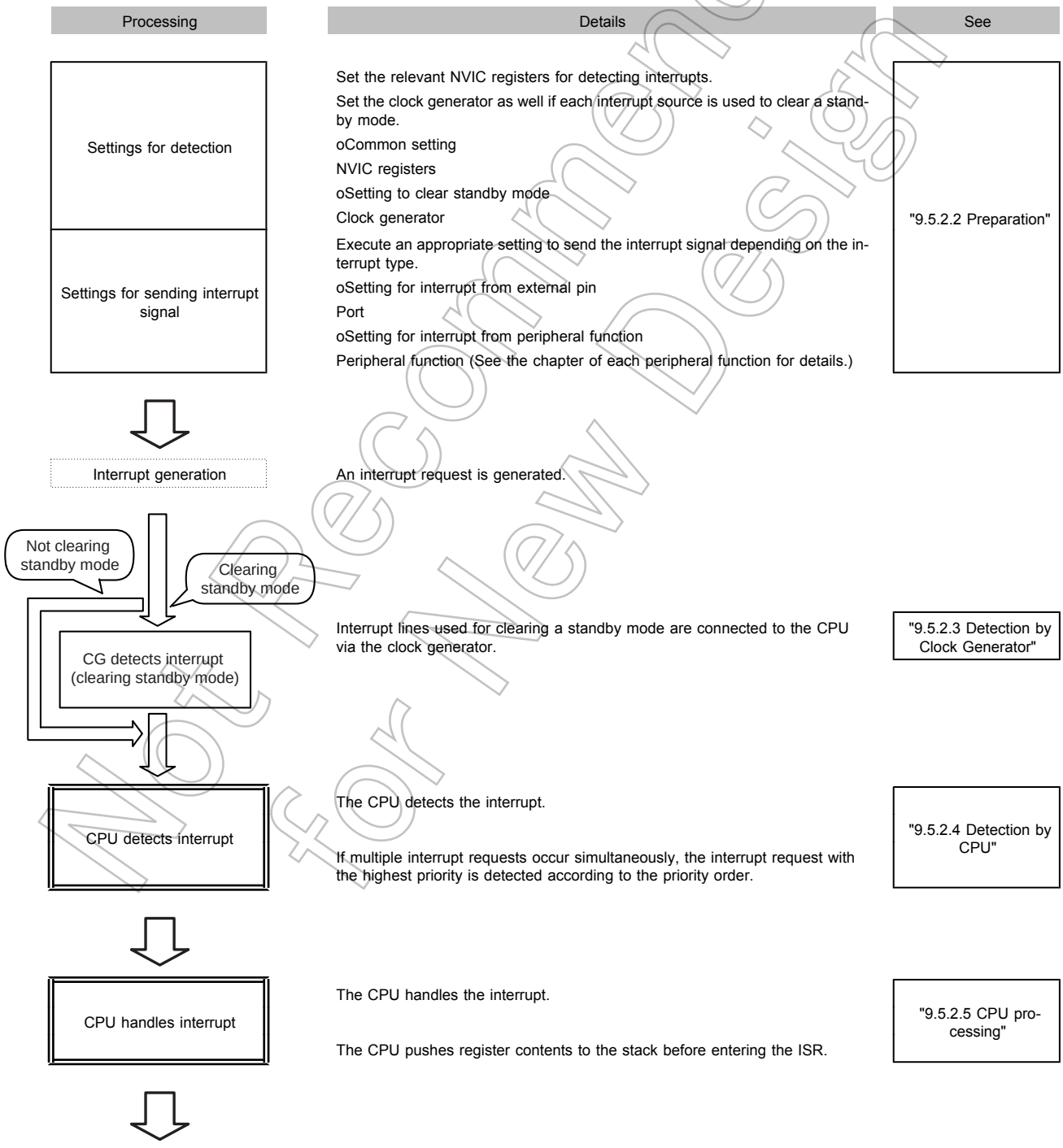
An interrupt request detected by the clock generator is notified to the CPU with a signal in "High" level.

9.5.2 Interrupt Handling

9.5.2.1 Flowchart

The following shows how an interrupt is handled.

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



Processing	Details	See
ISR execution	Program for the ISR. Clear the interrupt source if needed.	"9.5.2.6 Interrupt Service Routine (ISR)"
Return to preceding program	Configure to return to the preceding program of the ISR.	

9.5.2.2 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. Preconfiguration (1) (Interrupt from external pin)
4. Preconfiguration (2) (Interrupt from peripheral function)
5. Preconfiguration (3) (Interrupt Set-Pending Register)
6. Configuring the clock generator
7. Enabling interrupt by CPU

(1) Disabling interrupt by CPU

To make the CPU for not accepting any interrupt, write "1" to the corresponding bit of the 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) Preconfiguration (1) (Interrupt from external pin)

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

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

Note: **x: port number / m: corresponding bit / n: function register number**
In modes other than STOP mode, setting PxIE to enable input enables the corresponding interrupt input regardless of the PxFR setting. Be careful not to enable interrupts that are not used. Also, be aware of the description of "9.5.1.4 Precautions when using external interrupt pins".

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

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

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

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

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

Note: **m: corresponding bit**

(6) Configuring the clock generator

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

Before enabling an interrupt, clear the corresponding interrupt request already held. This can avoid unexpected interrupt.To clear corresponding interrupt request, write a value corresponding to the interrupt to be used to the CGICRCG register.See "9.6.3.4 CGICRCG(CG Interrupt Request Clear Register)" for each value.

Interrupt requests from external pins can be used without setting the clock generator if they are not used for exiting a standby mode. However, an "High" pulse or "High"-level signal must be input so that the CPU can detect it as an interrupt request. Also, be aware of the description of"9.5.1.4 Precautions when using external interrupt pins".

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

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

(7) Enabling interrupt by CPU

Enable the interrupt by the CPU as shown below.

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

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

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

At the end, PRIMASK register is zero cleared.

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

9.5.2.3 Detection by Clock Generator

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

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

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

9.5.2.4 Detection by CPU

The CPU detects an interrupt request with the highest priority.

9.5.2.5 CPU processing

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

9.5.2.6 Interrupt Service Routine (ISR)

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

(1) Pushing during ISR

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

Push the contents of other registers if needed.

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

(2) Clearing an interrupt source

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

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

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

9.6 Exception/Interrupt-Related Registers

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

9.6.1 Register List

NVIC registers — Base Address = 0xE000_E000

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

Clock generator registers — Base Address = 0x400F_3000

Register name		Address
CG Interrupt Mode Control Register A	CGIMCGA	0x0040
CG Interrupt Mode Control Register B	CGIMCGB	0x0044
CG Interrupt Mode Control Register C	CGIMCGC	0x0048
CG Interrupt Request Clear Register	CGICRCG	0x0060
Reset Flag Register	CGRSTFLG	0x0064
NMI Flag Register	CGNMIFLG	0x0068

9.6.2 NVIC Registers

9.6.2.1 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> <EHOSCSEL> by 32 is used as external reference clock.**

9.6.2.2 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".

9.6.2.3 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.

9.6.2.4 SysTick Calibration Value Register

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

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 Reload value to use for 10 ms timing (0xC35) by external reference clock (Note).

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

9.6.2.5 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

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

Note: **For descriptions of interrupts and interrupt numbers, see Section "9.5.1.5 List of Interrupt Sources".**

9.6.2.6 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

Bit	Bit Symbol	Type	Function
31-0	SETENA	R/W	Interrupt number [63:32] [Write] 1: Enable [Read] 0: Disabled 1: Enable Each bit corresponds to the specified number of interrupts. Writing "1" to a bit in this register enables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.

Note: **For descriptions of interrupts and interrupt numbers, see Section "9.5.1.5 List of Interrupt Sources".**

9.6.2.7 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

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

Note: **For descriptions of interrupts and interrupt numbers, see Section "9.5.1.5 List of Interrupt Sources".**

9.6.2.8 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

Bit	Bit Symbol	Type	Function
31-0	CLRENA	R/W	Interrupt number [63:32] [Write] 1: Disabled [Read] 0: Disabled 1: Enable Each bit corresponds to the specified number of interrupts. It can be performed to enable interrupts and to check if interrupts are disabled. Writing "1" to a bit in this register disables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.

Note: **For descriptions of interrupts and interrupt numbers, see Section "9.5.1.5 List of Interrupt Sources".**

9.6.2.9 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

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

Note: **For descriptions of interrupts and interrupt numbers, see Section "9.5.1.5 List of Interrupt Sources".**

9.6.2.10 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

Bit	Bit Symbol	Type	Function
31-0	SETPEND	R/W	Interrupt number [63:32] [Write] 1: Pend [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register pends the corresponding interrupt. However, writing "1" has no effect on an interrupt that is already pending or is disabled. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts. Clear and Interrupt Set-Pending Register bit by writing "1" to the corresponding bit in the Interrupt Clear-Pending Register.

Note: **For descriptions of interrupts and interrupt numbers, see Section "9.5.1.5 List of Interrupt Sources".**

9.6.2.11 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

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

Note: **For descriptions of interrupts and interrupt numbers, see Section "9.5.1.5 List of Interrupt Sources".**

9.6.2.12 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

Bit	Bit Symbol	Type	Function
31-0	CLRPEND	R/W	Interrupt number [63:32] [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register clears the corresponding pending interrupt. However, writing "1" has no effect on an interrupt that is already being serviced. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts.

Note: **For descriptions of interrupts and interrupt numbers, see Section "9.5.1.5 List of Interrupt Sources".**

9.6.2.13 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

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.

9.6.2.14 Vector Table Offset Register

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

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

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

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

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

9.6.2.16 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.

9.6.2.17 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.**

9.6.3 Clock generator registers

9.6.3.1 CGIMCGA(CG Interrupt Mode Control Register A)

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

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

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

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

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

9.6.3.2 CGIMCGB(CG Interrupt Mode Control Register B)

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

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

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

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

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

9.6.3.3 CGIMCGC(CG Interrupt Mode Control Register C)

	31	30	29	28	27	26	25	24
bit symbol	-	EMCGB			EMSTB		-	INTBEN
After reset	0	0	1	0	0	0	Undefined	0
	23	22	21	20	19	18	17	16
bit symbol	-	EMCGA			EMSTA		-	INTAEN
After reset	0	0	1	0	0	0	Undefined	0
	15	14	13	12	11	10	9	8
bit symbol	-	EMCG9			EMST9		-	INT9EN
After reset	0	0	1	0	0	0	Undefined	0
	7	6	5	4	3	2	1	0
bit symbol	-	EMCG8			EMST8		-	INT8EN
After reset	0	0	1	0	0	0	Undefined	0

Bit	Bit Symbol	Type	Function
31	−	R	Read as 0.
30-28	EMCGB[2:0]	R/W	INTUSBWKUP sets the detection state of the demand. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
27-26	EMSTB[1:0]	R	Reads as undefined.
25	−	R	Reads as undefined.
24	INTBEN	R/W	INTUSBWKUP detection 0:Disable 1: Enable
23	−	R	Read as 0.
22-20	EMCGA[2:0]	R/W	INTUSBON sets the detection state of the demand. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
19-18	EMSTA[1:0]	R	Reads as undefined.
17	−	R	Reads as undefined.
16	INTAEN	R/W	INTUSBPON detection 0:Disable 1: Enable
15	−	R	Read as 0.
14-12	EMCG9[2:0]	R/W	active level setting of INT9 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
11-10	EMST9[1:0]	R	active level of INT9 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges
9	−	R	Reads as undefined.
8	INT9EN	R/W	INT9 clear input 0:Disable 1: Enable

Bit	Bit Symbol	Type	Function
7	−	R	Read as 0.
6-4	EMCG8[2:0]	R/W	active level setting of INT8 standby clear request. (101 to 111: setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
3-2	EMST8[1:0]	R	active level of INT8 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges
1	−	R	Reads as undefined.
0	INT8EN	R/W	INT8 clear input 0:Disable 1: Enable

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

9.6.3.4 CGICRCG(CG Interrupt Request Clear Register)

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

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as 0.
4-0	ICRCG[4:0]	W	Clear interrupt requests. 0_0000: INT0 0_1000: INT8 0_0001: INT1 0_1001: INT9 0_0010: INT2 0_1010:INTUSBPON 0_0011: INT3 0_1011:INTUSBWKUP 0_0100: INT4 0_0101: INT5 0_0110: INT6 0_0111: INT7 0_1100 to 1_1111: Reserved. Read as 0.

9.6.3.5 CGNMIFLG(NMI Flag Register)

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

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

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

9.6.3.6 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	-	-	-	DBGRSTF	STOP2RSTF	WDTRSTF	-	PINRSTF
After pin reset	0	0	0	0	0	0	Undefined	1

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as 0.
4	DBGRSTF	R/W	Debug reset flag (Note1) 0: "0" is written 1: Reset from <SYSRESETREQ>
3	STOP2RSTF	R/W	STOP2 reset flag 0: "0" is written 1: Reset flag by STOP2 mode release
2	WDTRSTF	R/W	WDT reset flag 0: "0" is written 1: Reset from WDT
1	−	R/W	Read as undefined. Write as 0.
0	PINRSTF	R/W	$\overline{\text{RESET}}$ pin flag 0: "0" is written 1: Reset from $\overline{\text{RESET}}$ pin.

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

Note: **This product is initialized by the external reset.**

10. Input/Output Ports

10.1 Port Functions

10.1.1 Function Lists

TMPM366FDXBG/FYXBG/FWXBG has 74 ports. Besides the ports function, these ports can be used as I/O pins for peripheral functions.

Table 10-1, Table 10-2 and Table 10-3 show the port function table.

Table 10-1 Port Function List (Port A to Port C)

Port	Pin	Input/ Output	Programmable Pull-up Pull-down	Schmitt input	Noise Filter	Program-mable Open-drain	Function pin
Port A	PA0	I/O	Pull-up	-	-	o	D0/AD0
	PA1	I/O	Pull-up	-	-	o	D1/AD1
	PA2	I/O	Pull-up	-	-	o	D2/AD2
	PA3	I/O	Pull-up	-	-	o	D3/AD3
	PA4	I/O	Pull-up	-	-	o	D4/AD4
	PA5	I/O	Pull-up	-	-	o	D5/AD5
	PA6	I/O	Pull-up	-	-	o	D6/AD6
	PA7	I/O	Pull-up	-	-	o	D7/AD7
Port B	PB0	I/O	Pull-up	-	-	o	D8/AD8/SP1D0/A0
	PB1	I/O	Pull-up	-	-	o	D9/AD9/SP1DI/A1
	PB2	I/O	Pull-up	-	-	o	D10/AD10/SP1CLK/A2
	PB3	I/O	Pull-up	-	-	o	D11/AD11/SP1FSS/A3
	PB4	I/O	Pull-up	-	-	o	D12/AD12/SP2D0/A4
	PB5	I/O	Pull-up	-	-	o	D13/AD13/SP2DI/A5
	PB6	I/O	Pull-up	-	-	o	D14/AD14/SP2CLK/A6
	PB7	I/O	Pull-up	-	-	o	D15/AD15/SP2FSS/A7
Port C	PC0	I/O	Pull-up	o	-	o	TXD1/A2/TB2IN0
	PC1	I/O	Pull-up	o	-	o	RXD1/A1/TB2IN1
	PC2	I/O	Pull-up	o	-	o	SCLK1/A0/TB0OUT/$\overline{\text{CTS1}}$

o : Exist
- : Not exist

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

Table 10-2 Port Function List (Port D to Port G)

Port	Pin	Input/ Output	Programmable Pull-up Pull-down	Schmitt Input	Noise Filter	Program-mable Open-drain	Function pin
Port D	PD0	I/O	Pull-up	o	−	o	A16/TB7OUT
	PD1	I/O	Pull-up	o	−	o	A17/TB8OUT
	PD2	I/O	Pull-up	o	−	o	A18/TB9OUT
	PD3	I/O	Pull-up	o	−	o	A19/$\overline{\text{ADTRG}}$
	PD4	I/O	Pull-up	o	−	o	SP0DO
	PD5	I/O	Pull-up	o	−	o	SP0DI
	PD6	I/O	Pull-up	o	−	o	SP0CLK
	PD7	I/O	Pull-up	o	−	o	SP0FSS/SCOUT
Port E	PE0	I/O	Pull-up	o	−	o	TXD0/A20
	PE1	I/O	Pull-up	o	−	o	RXD0/A21
	PE2	I/O	Pull-up	o	−	o	SCLK0/TB2OUT/ $\overline{\text{CTS0}}$/A22
	PE3	I/O	Pull-up	o	o(INT5 only)	o	INT5/A15/TB3OUT/A23
	PE4	I/O	Pull-up	o	−	o	SDA1/A14
	PE5	I/O	Pull-up	o	−	o	SCL1/A13
	PE6	I/O	Pull-up	o	−	o	SCK1/A12
	PE7	I/O	Pull-up	o	−	o	INT4/A11
Port F	PF0	Output	After Reset, Pull-up	o	−	o	$\overline{\text{BOOT}}$/TB6OUT
	PF1	I/O	Pull-up	o	−	o	$\overline{\text{RD}}$
	PF2	I/O	Pull-up	o	−	o	$\overline{\text{WR}}$
	PF3	I/O	Pull-up	o	−	o	$\overline{\text{BELL}}$
	PF4	I/O	Pull-up	o	o(INT6 only)	o	$\overline{\text{BELH}}$/INT6/TB5IN0
	PF5	I/O	Pull-up	o	o(INT7 only)	o	$\overline{\text{CS1}}$/INT7/TB5IN1
	PF6	I/O	Pull-up	o	−	o	$\overline{\text{CS0}}$
	PF7	I/O	Pull-up	o	−	o	ALE
Port G	PG0	I/O	Pull-up	o	−	o	SDA0/A3/TXD2/IROUT
	PG1	I/O	Pull-up	o	−	o	SCL0/A4/TB3IN0/RXD2/IRIN
	PG2	I/O	Pull-up	o	−	o	SCK0/A5/TB3IN1/$\overline{\text{CTS2}}$
	PG3	I/O	Pull-up	o	o(INT0 only)	o	INT0/A6/TB4IN0/RIN2
	PG4	I/O	Pull-up	o	−	o	A7/TB4IN1/$\overline{\text{RTS2}}$
	PG5	I/O	Pull-up	o	−	o	INT1/USBPON

o : Exist
- : Not exist

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

Table 10-3 Port Function List (Port H to Port K)

Port	Pin	Input/ Output	Programmable Pull-up Pull-down	Schmitt input	Noise Filter	Program-mable Open-drain	Function pin
Port H	PH0	I/O	Pull-up	o	−	o	TRACEDATA2
	PH1	I/O	Pull-up	o	−	o	TRACEDATA3
	PH2	I/O	Pull-up	o	−	o	A10/TB4OUT/DCD2
	PH3	I/O	Pull-up	o	−	o	A9/TB5OUT/DSR2
	PH4	I/O	Pull-up	o	−	o	A8/INT8/DTR2
Port I	PI0	I/O	Pull-up	o	−	o	TRACEDATA1
	PI1	I/O	Pull-up	o	−	o	TRACEDATA0
	PI2	I/O	Pull-up	o	−	o	TRACECLK
	PI3	I/O	After Reset, Pull-down	o	−	−	TCK/SWCLK
	PI4	I/O	After Reset, Pull-up	o	−	−	TMS/SWDIO
	PI5	I/O	Pull-up	o	−	−	TDO/SWV
	PI6	I/O	After Reset, Pull-up	o	−	−	TDI
	PI7	I/O	Pull-up	o	o	−	$\overline{\text{TRST}}$
Port J	PJ0	I/O	Pull-up	o	−	−	AINA00
	PJ1	I/O	Pull-up	o	−	−	AINA01
	PJ2	I/O	Pull-up	o	−	−	AINA02
	PJ3	I/O	Pull-up	o	−	−	AINA03
	PJ4	I/O	Pull-up	o	−	−	AINA04
	PJ5	I/O	Pull-up	o	−	−	AINA05
	PJ6	I/O	Pull-up	o	−	−	AINA06/TB0IN0
	PJ7	I/O	Pull-up	o	o(INT9 only)	−	AINA07/INT9/TB0IN1
Port K	PK0	I/O	Pull-up	o	o(INT2 only)	−	AINA08/INT2/TB1IN0
	PK1	I/O	Pull-up	o	o(INT3 only)	−	AINA09/INT3/TB1IN1
	PK2	I/O	Pull-up	o	−	−	AINA10/TB6IN0
	PK3	I/O	Pull-up	o	-	−	AINA11/TB6IN1

o : Exist
- : Not exist

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

10.1.2 Port Registers Outline

The following registers need to be configured to use ports.

- PxDATA: Port x data register

  To read/ write port data.

- PxCR: Port x output control register

  To control output.

  PxIE needs to be configured to control input.

- PxFRn: Port x function register n

  To set functions.

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

- PxOD: Port x open drain control register

  To control the programmable open drain.

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

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

- PxPUP: Port x pull-up control register

  To control program pull ups.

- PxPDN: Port x pull-down control register

  To control programmable pull downs.

- PxIE: Port x input control register

  To control inputs.

  For avoided through current, default setting prohibits inputs.

10.1.3 Port States in STOP1/STOP2 Mode

On the one hand, input and output in STOP1 mode are enabled/disabled by the CGSTBYCR<DRVE>; on the other hand input and output in STOP2 mode are enabled/disabled by the CGSTBYCR<PTKEEP>.

If PxIE or PxCR is enabled with <DRVE>=1 or <PTKEEP>="0" → "1", input or output is enabled respectively in STOP1/STOP2 mode. If <DRVE>=0, both input and output are disabled in STOP1 mode except for some ports even if PxIE or PxCR are enabled. However, to transmit from Normal mode to STOP2 mode, the <PTKEEP> need to be set "0" to "1".

Table 10-4 shows the pin conditions in STOP mode.

Table 10-4 Pin States in the STOP1/STOP2 mode

Function	Pin Name	I/O	STOP1		STOP2	
			<DRVE> = 0	<DRVE> = 1	<PTKEEP> = 0	<PTKEEP> = 1
Control Pin	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC	Input	o	o	×	o
Oscillator	X1/EHCLKIN	Input (Note1	×	×	×	×
	X2	Output (Note1	"High" level output.	"High" level output.	×	×
PORT	PI3 to PI5 ($\overline{\text{TRST}}$, TDI, SWCLK/TCK) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Input	Depends on (PxIE[m])		×	Input holding, but Depends on (PxIE[m])
	PI4 (SWDIO/TMS) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Input	Depends on (PxIE[m])		×	Input holding, but Depends on (PxIE[m])
		Output	Enabled when data is valid. Disabled when data is invalid.		×	Output holding, but Depends on (PxCR[m])
	PI5, PI2, PI1, PI0, PH0, PH1 (TDO/SWV, TRACECLK, TRACEDATA0 to 3) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Output	Depends on (PxCR[m])		×	Output holding, but Depends on (PxCR[m])
	PG3, PG5, PK0, PK1, PE7, PE3, PF4, PF5, PH4, PJ7 (INT0 to 9) (Interrupt setting, case of PxFRn<PxmFn>="1" and PxIE<PxmIE>="1")	Input	o	o	×	o
	If using other than listed above	Input	×	Depends on (PxIE[m])	×	Input holding, but Depends on (PxIE[m])
		Output	×	Depends on (PxCR[m])	×	Output holding, but Depends on (PxCR[m])

o : Valid input or output.

× : Invalid input or output.

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

10.2 Port functions

This chapter describes the port registers detail.

This chapter describes only "circuit type" reading circuit configuration.For detailed circuit diagram, refer to "10.3 Block Diagrams of Ports".

10.2.1 Port A (PA0 to PA7)

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

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

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

10.2.1.1 Port A register

Base Address = 0x400C_0000

Register name		Address (Base+)
Port A data register	PADATA	0x0000
Port A output control register	PACR	0x0004
Port A function register 1	PAFR1	0x0008
Reserved	-	0x000C
Reserved	-	0x0010
Reserved	-	0x0014
Reserved	-	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Port A open drain control register	PAOD	0x0028
Port A pull-up control register	PAPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port A input control register	PAIE	0x0038

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

10.2.1.2 PADATA (Port A data register)

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

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

10.2.1.3 PACR (Port A output control register)

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

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

10.2.1.4 PAFR1 (Port A function register 1)

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

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

10.2.1.5 PAOD (Port A open drain control register)

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

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

10.2.1.6 PAPUP (Port A pull-up control register)

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

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

10.2.1.7 PAIE (Port A input control register)

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

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

10.2.2 Port B (PB0 to PB7)

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

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

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

10.2.2.1 Port B Register

Base Address = 0x400C_0100

Register name		Address (Base+)
Port B data register	PBDATA	0x0000
Port B output control register	PBCR	0x0004
Port B function register 1	PBFR1	0x0008
Port B function register 2	PBFR2	0x000C
Port B function register 3	PBFR3	0x0010
Reserved	-	0x0014
Reserved	-	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Port B open drain control register	PBOD	0x0028
Port B pull-up control register	PBPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port B input control register	PBIE	0x0038

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

10.2.2.2 PBDATA (Port B data register)

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

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

10.2.2.3 PBCR (Port B output control register)

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

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

10.2.2.4 PBFR1 (Port B function register 1)

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

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

10.2.2.5 PBFR2 (Port B function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PB7F2	PB6F2	PB5F2	PB4F2	PB3F2	PB2F2	PB1F2	PB0F2
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PB7F2	R/W	0: PORT 1: SP2FSS
6	PB6F2	R/W	0: PORT 1: SP2CLK
5	PB5F2	R/W	0: PORT 1: SP2DI
4	PB4F2	R/W	0: PORT 1: SP2DO
3	PB3F2	R/W	0: PORT 1: SP1FSS
2	PB2F2	R/W	0: PORT 1: SP1CLK
1	PB1F2	R/W	0: PORT 1: SP1DI
0	PB0F2	R/W	0: PORT 1: SP1DO

10.2.2.6 PBFR3 (Port B function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PB7F3	PB6F3	PB5F3	PB4F3	PB3F3	PB2F3	PB1F3	PB0F3
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PB7F2	R/W	0: PORT 1: A7
6	PB6F2	R/W	0: PORT 1: A6
5	PB5F2	R/W	0: PORT 1: A5
4	PB4F2	R/W	0: PORT 1: A4
3	PB3F2	R/W	0: PORT 1: A3
2	PB2F2	R/W	0: PORT 1: A2
1	PB1F2	R/W	0: PORT 1: A1
0	PB0F2	R/W	0: PORT 1: A0

10.2.2.7 PBOD (Port B open drain control register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PB7OD-PB0OD	R/W	0: Push-pull output 1: Open drain

10.2.2.8 PBPUP (Port B pull-up control register)

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

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

10.2.2.9 PBIE (Port B input control register)

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

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

10.2.3 Port C (PC0 to PC2)

The port C is a general-purpose, 3-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input/output function, the port C performs the external bus interface, the serial channel and the 16-bit timer / event counter.

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

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

10.2.3.1 Port C Register

Base Address = 0x400C_0200

Register name		Address (Base+)
Port C data register	PCDATA	0x0000
Port C output control register	PCCR	0x0004
Port C function register 1	PCFR1	0x0008
Port C function register 2	PCFR2	0x000C
Port C function register 3	PCFR3	0x0010
Port C function register 4	PCFR4	0x0014
Reserved	-	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Port C open drain control register	PCOD	0x0028
Port C pull-up control register	PCPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port C input control register	PCIE	0x0038

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

10.2.3.2 PCDATA (Port C data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	PC2	PC1	PC0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2-0	PC2-PC0	R/W	Port C data register.

10.2.3.3 PCCR (Port C 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	-	-	-	-	-	PC2C	PC1C	PC0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2-0	PC2C-PC0C	R/W	Output 0: Disable 1: Enable

10.2.3.4 PCFR1 (Port C function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	PC2F1	PC1F1	PC0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	PC2F1	R/W	0: PORT 1: SCLK1
1	PC1F1	R/W	0: PORT 1: RXD1
0	PC0F1	R/W	0: PORT 1: TXD1

10.2.3.5 PCFR2 (Port C function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	PC2F2	PC1F2	PC0F2
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	PC2F2	R/W	0: PORT 1: A0
1	PC1F2	R/W	0: PORT 1: A1
0	PC0F2	R/W	0: PORT 1: A2

10.2.3.6 PCFR3 (Port C function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	PC2F3	PC1F3	PC0F3
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	PC2F3	R/W	0: PORT 1: TB0OUT
1	PC1F3	R/W	0: PORT 1: TB2IN1
0	PC0F3	R/W	0: PORT 1: TB2IN0

10.2.3.7 PCFR4 (Port C function 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	-	-	-	-	-	PC2F4	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	PC2F4	R/W	0: PORT 1: $\overline{CTS1}$
1-0	−	R	Read as "0".

10.2.3.8 PCOD (Port C open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	PC2OD	PC1OD	PC0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2-0	PC2OD- PC0OD	R/W	0: Push-pull output 1: Open drain

10.2.3.9 PCPUP (Port C pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	PC2UP	PC1UP	PC0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2-0	PC2UP-PC0UP	R/W	Pull-up 0: Disable 1: Enable

10.2.3.10 PCIE (Port C 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	-	-	-	-	-	PC2IE	PC1IE	PC0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2-0	PC2IE-PC0IE	R/W	input 0: Disable 1: Enable

10.2.4 Port D (PD0 to PD7)

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

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

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

10.2.4.1 Port D Register

Base Address = 0x400C_0300

Register name		Address (Base+)
Port D data register	PDDATA	0x0000
Port D output control register	PDCR	0x0004
Reserved	-	0x0008
Port D function register 2	PDFR2	0x000C
Port D function register 3	PDFR3	0x0010
Reserved	-	0x0014
Reserved	-	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Port D open drain control register	PDOD	0x0028
Port D pull-up control register	PDPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port D input control register	PDIE	0x0038

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

10.2.4.2 PDDATA (Port D data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PD7-PD0	R/W	Port D data register.

10.2.4.3 PDCR (Port D 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	PD7C	PD6C	PD5C	PD4C	PD3C	PD2C	PD1C	PD0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PD7C-PD0C	R/W	Output 0: Disable 1: Enable

10.2.4.4 PDFR2 (Port D function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PD7F2	PD6F2	PD5F2	PD4F2	PD3F2	PD2F2	PD1F2	PD0F2
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PD7F2	R/W	0: PORT 1: SP0FSS
6	PD6F2	R/W	0: PORT 1: SP0CLK
5	PD5F2	R/W	0: PORT 1: SP0DI
4	PD4F2	R/W	0: PORT 1: SP0DO
3	PD3F2	R/W	0: PORT 1: A19
2	PD2F2	R/W	0: PORT 1: A18
1	PD1F2	R/W	0: PORT 1: A17
0	PD0F2	R/W	0: PORT 1: A16

10.2.4.5 PDFR3 (Port D function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PD7F3	-	-	-	PD3F3	PD2F3	PD1F3	PD0F3
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PD7F3	R/W	0: PORT 1: SCOUT
6-4	−	R	Read as "0".
3	PD3F3	R/W	0: PORT 1: $\overline{\text{ADTRG}}$
2	PD2F3	R/W	0: PORT 1: TB9OUT
1	PD1F3	R/W	0: PORT 1: TB8OUT
0	PD0F3	R/W	0: PORT 1: TB7OUT

10.2.4.6 PDOD (Port D open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PD7OD	PD6OD	PD5OD	PD4OD	PD3OD	PD2OD	PD1OD	PD0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PD7OD-PD0OD	R/W	0: Push-pull output 1: Open drain

10.2.4.7 PDPUP (Port D pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PD7UP	PD6UP	PD5UP	PD4UP	PD3UP	PD2UP	PD1UP	PD0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PD7UP-PD0UP	R/W	Pull-up 0: Disable 1: Enable

10.2.4.8 PDIE (Port D 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	PD7IE	PD6IE	PD5IE	PD4IE	PD3IE	PD2IE	PD1IE	PD0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PD7IE-PD0IE	R/W	Input 0: Disable 1: Enable

10.2.5 Port E (PE0 to PE7)

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

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

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

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

10.2.5.1 Port E Register

Base Address = 0x400C_0400

Register name		Address (Base+)
Port E data register	PEDATA	0x0000
Port E output control register	PECR	0x0004
Port E function register 1	PEFR1	0x0008
Port E function register 2	PEFR2	0x000C
Port E function register 3	PEFR3	0x0010
Port E function register 4	PEFR4	0x0014
Port E function register 5	PEFR5	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Port E open drain control register	PEOD	0x0028
Port E pull-up control register	PEPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port E input control register	PEIE	0x0038

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

10.2.5.2 PEDATA (Port E data register)

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

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

10.2.5.3 PECR (Port E output control register)

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

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

10.2.5.4 PEFR1 (Port E function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PE7F1	PE6F1	PE5F1	PE4F1	PE3F1	PE2F1	PE1F1	PE0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PE7F1	R/W	0: PORT 1: INT4
6	PE6F1	R/W	0: PORT 1: SCK1
5	PE5F1	R/W	0: PORT 1: SCL1 / SI1
4	PE4F1	R/W	0: PORT 1: SDA1 / SO1
3	PE3F1	R/W	0: PORT 1: INT5
2	PE2F1	R/W	0: PORT 1: SCLK0
1	PE1F1	R/W	0: PORT 1: RXD0
0	PE0F1	R/W	0: PORT 1: TXD0

10.2.5.5 PEFR2 (Port E function register 2)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PE7F2	R/W	0: PORT 1: A11
6	PE6F2	R/W	0: PORT 1: A12
5	PE5F2	R/W	0: PORT 1: A13
4	PE4F2	R/W	0: PORT 1: A14
3	PE3F2	R/W	0: PORT 1: A15
2-0	−	R	Read as "0".

10.2.5.6 PEFR3 (Port E function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PE3F3	PE2F3	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3	PE3F3	R/W	0: PORT 1: TB3OUT
2	PE2F3	R/W	0: PORT 1: TB2OUT
1-0	−	R	Read as "0".

10.2.5.7 PEFR4 (Port E function 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	-	-	-	-	-	PE2F4	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	PE2F4	R/W	0: PORT 1: $\overline{\text{CTS0}}$
1-0	−	R	Read as "0".

10.2.5.8 PEFR5 (Port E function 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	-	-	-	-	PE3F5	PE2F5	PE1F5	PE0F5
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3	PE3F1	R/W	0: PORT 1: A23
2	PE2F1	R/W	0: PORT 1: A22
1	PE1F1	R/W	0: PORT 1: A21
0	PE0F1	R/W	0: PORT 1: A20

10.2.5.9 PEOD (Port E open drain control register)

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

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

10.2.5.10 PEPUP (Port E pull-up control register)

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

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

10.2.5.11 PEIE (Port E input control register)

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

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

10.2.6 Port F (PF0 to PF7)

The port F is a general-purpose, 1-bit output port and 7-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose port function, the port F performs the functions of the external bus interface, the external interrupt input, 16-bit timer / event counter and the mode setting function ($\overline{\text{BOOT}}$).

Reset initializes from bit7 to bit1 of the port F as general-purpose ports with input, output and pull-up disabled. Reset initializes bit0 of PF as output port with output disabled and pull-up enabled.

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

While $\overline{\text{RESET}}$ pin is "Low", input and pull-up of PF0 ($\overline{\text{BOOT}}$) are enabled. At the rising edge of $\overline{\text{RESET}}$ pin, if PF0 ($\overline{\text{BOOT}}$) is "High", TMPM366FDXBG/FYXBG/FWXBG enters the single chip mode and boots from the on chip Flash ROM. If PF0 ($\overline{\text{BOOT}}$) is "Low", TMPM366FDXBG/FYXBG/FWXBG enters the single boot mode and boots from the on chip BOOTROM. For details of the single boot mode, refer to "Flash memory operation".

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

10.2.6.1 Port F Register

Base Address = 0x400C_0500

Register name		Address (Base+)
Port F data register	PFDATA	0x0000
Port F output control register	PFCR	0x0004
Port F function register 1	PFFR1	0x0008
Port F function register 2	PFFR2	0x000C
Port F function register 3	PFFR3	0x0010
Reserved	-	0x0014
Reserved	-	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Port F open drain control register	PFOD	0x0028
Port F pull-up control register	PFPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port F input control register	PFIE	0x0038

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

10.2.6.2 PFDATA (Port F data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PF7-PF0	R/W	Port F data register

10.2.6.3 PFCR (Port F output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PF7C	PF6C	PF5C	PF4C	PF3C	PF2C	PF1C	PF0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PF7C-PF0C	R/W	Output 0: Disable 1: Enable

10.2.6.4 PFFR1 (Port F function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PF7F1	PF6F1	PF5F1	PF4F1	PF3F1	PF2F1	PF1F1	-
After reset	0	0	0	0	0	0	0	-

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PF7F1	R/W	0: PORT 1: ALE
6	PF6F1	R/W	0: PORT 1: $\overline{CS0}$
5	PF5F1	R/W	0: PORT 1: $\overline{CS1}$
4	PF4F1	R/W	0: PORT 1: $\overline{BELH}$
3	PF3F1	R/W	0: PORT 1: $\overline{BELL}$
2	PF2F1	R/W	0: PORT 1: $\overline{WR}$
1	PF1F1	R/W	0: PORT 1: $\overline{RD}$
0	−	R	Read as "0".

10.2.6.5 PFFR2 (Port F function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	PF5F2	PF4F2	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5	PF5F2	R/W	0: PORT 1: INT7
4	PF4F2	R/W	0: PORT 1: INT6
3-0	−	R	Read as "0".

10.2.6.6 PFFR3 (Port F function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	PF5F3	PF4F3	-	-	-	PF0F3
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5	PF5F3	R/W	0: PORT 1: TB5IN1
4	PF4F3	R/W	0: PORT 1: TB5IN0
3-1	-	R	Read as "0".
0	PF0F3	R/W	0: PORT 1: TB6OUT

10.2.6.7 PFOD (Port F open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PF7OD	PF6OD	PF5OD	PF4OD	PF3OD	PF2OD	PF1OD	PF0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PF7OD-PF0OD	R/W	0: Push-pull output 1: Open-drain output

10.2.6.8 PFPUP (Port F pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PF7UP	PF6UP	PF5UP	PF4UP	PF3UP	PF2UP	PF1UP	PF0UP
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PF7UP-PF1UP	R/W	Pull-up 0: Disable 1: Enable

10.2.6.9 PFIE (Port F input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PF7IE	PF6IE	PF5IE	PF4IE	PF3IE	PF2IE	PF1IE	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-1	PF7IE-PF1IE	R/W	Input 0: Disable 1: Enable
0	-	R	Read as "0".

10.2.7 Port G (PG0 to PG5)

The port G is a general-purpose, 6-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input/output function, the port G performs the external bus interface, the serial bus interface, the asynchronous serial channel, the external interrupt input,16-bit timer / event counter and the USB.

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

The port G has 5 types of function register. If you use the port G as a general-purpose port, set "0" to the corresponding bit of the three registers. If you use the port G as other than a general-purpose port, set "1" to the corresponding bit of the function register. Do not set "1" to the some function registers at the same time.

Only when input is enabled, PG5 tolerates 5V inputs. Note that these pins cannot be pulled up over the power supply voltage when using as open-drain output.

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

10.2.7.1 Port G Register

Base Address = 0x400C_0600

Register name		Address (Base+)
Port G data register	PGDATA	0x0000
Port G output control register	PGCR	0x0004
Port G function register 1	PGFR1	0x0008
Port G function register 2	PGFR2	0x000C
Port G function register 3	PGFR3	0x0010
Port G function register 4	PGFR4	0x0014
Port G function register 5	PGFR5	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Port G open drain control register	PGOD	0x0028
Port G pull-up control register	PGPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port G input control register	PGIE	0x0038

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

10.2.7.2 PGDATA (Port G data register)

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

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5-0	PG5-PG0	R/W	Port G data register.

10.2.7.3 PGCR (Port G output control register)

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

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5-0	PG5C-PG0C	R/W	Output 0: Disable 1: Enable

10.2.7.4 PGFR1 (Port G function register 1)

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

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5	PG5F1	R/W	0: PORT 1: INT1
4	−	R	Read as "0".
3	PG3F1	R/W	0: PORT 1: INT0
2	PG2F1	R/W	0: PORT 1: SCK0
1	PG1F1	R/W	0: PORT 1: SCL0 / SI0
0	PG0F1	R/W	0: PORT 1: SDA0 / SO0

10.2.7.5 PGFR2 (Port G function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PG4F2	PG3F2	PG2F2	PG1F2	PG0F2
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4	PG4F2	R/W	0: PORT 1: A7
3	PG3F2	R/W	0: PORT 1: A6
2	PG2F2	R/W	0: PORT 1: A5
1	PG1F2	R/W	0: PORT 1: A4
0	PG0F2	R/W	0: PORT 1: A3

10.2.7.6 PGFR3 (Port G function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PG4F3	PG3F3	PG2F3	PG1F3	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4	PG4F3	R/W	0: PORT 1: TB4IN1
3	PG3F3	R/W	0: PORT 1: TB4IN0
2	PG2F3	R/W	0: PORT 1: TB3IN1
1	PG1F3	R/W	0: PORT 1: TB3IN0
0	−	R	Read as "0".

10.2.7.7 PGFR4 (Port G function 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	-	-	PG5F4	PG4F4	PG3F4	PG2F4	PG1F4	PG0F4
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5	PG5F4	R/W	0: PORT 1: USBPON
4	PG4F4	R/W	0: PORT 1: $\overline{\text{RTS2}}$
3	PG3F4	R/W	0: PORT 1: RIN2
2	PG2F4	R/W	0: PORT 1: $\overline{\text{CTS2}}$
1	PG1F4	R/W	0: PORT 1: RXD2
0	PG0F4	R/W	0: PORT 1: TXD2

10.2.7.8 PGFR5 (Port G function 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	-	-	-	-	-	-	PG1F5	PG0F5
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as "0".
1	PG1F5	R/W	0: PORT 1: IRIN
0	PG0F5	R/W	0: PORT 1: IROUT

10.2.7.9 PGOD (Port G open drain control register)

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

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5-0	PG5OD- PG0OD	R/W	0: Push-pull output 1: Open-drain output

Note: Only when input is enabled, PG5 tolerates 5V inputs. Note that this pin cannot be pulled up over the power supply voltage when using as open-drain output.

10.2.7.10 PGPUP (Port G pull-up control register)

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

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5-0	PG5UP- PG0UP	R/W	Pull-up 0: Disable 1: Enable

10.2.7.11 PGIE (Port G input control register)

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

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as "0".
5-0	PG5IE-PG0IE	R/W	Input 0: Disable 1: Enable

10.2.8 Port H (PH0 to PH4)

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

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

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

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

10.2.8.1 Port H Register

Base Address = 0x400C_0700

Register name		Address (Base+)
Port H data register	PHDATA	0x0000
Port H output control register	PHCR	0x0004
Port H function register 1	PHFR1	0x0008
Port H function register 2	PHFR2	0x000C
Port H function register 3	PHFR3	0x0010
Port H function register 4	PHFR4	0x0014
Reserved	-	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Port H open drain control register	PHOD	0x0028
Port H pull-up control register	PHPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port H input control register	PHIE	0x0038

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

10.2.8.2 PHDATA (Port H data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PH4	PH3	PH2	PH1	PH0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4-0	PH4-PH0	R/W	Port H data register.

10.2.8.3 PHCR (Port H 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	-	-	-	PH4C	PH3C	PH2C	PH1C	PH0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4-0	PH4C-PH0C	R/W	Output 0: Disable 1: Enable

10.2.8.4 PHFR1 (Port H function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	PH1F1	PH0F1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as "0".
1	PH1F1	R/W	0: PORT 1: TRACEDATA3
0	PH0F1	R/W	0: PORT 1: TRACEDATA2

10.2.8.5 PHFR2 (Port H function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PH4F2	PH3F2	PH2F2	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4	PH4F2	R/W	0: PORT 1: A8
3	PH3F2	R/W	0: PORT 1: A9
2	PH2F2	R/W	0: PORT 1: A10
1-0	−	R	Read as "0".

10.2.8.6 PHFR3 (Port H function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PH4F3	PH3F3	PH2F3	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4	PH4F3	R/W	0: PORT 1: INT8
3	PH3F3	R/W	0: PORT 1: TB5OUT
2	PH2F3	R/W	0: PORT 1: TB4OUT
1-0	−	R	Read as "0".

10.2.8.7 PHFR4 (Port H function 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	-	-	-	PH4F4	PH3F4	PH2F4	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4	PH4F4	R/W	0: PORT 1: DTR2
3	PH3F4	R/W	0: PORT 1: DSR2
2	PH2F4	R/W	0: PORT 1: DCD2
1-0	−	R	Read as "0".

10.2.8.8 PHOD (Port H open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PH4OD	PH3OD	PH2OD	PH1OD	PH0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4-0	PH4OD-PH0OD	R/W	0: Push-pull output 1: Open-drain output

10.2.8.9 PHPUP (Port H pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	PH4UP	PH3UP	PH2UP	PH1UP	PH0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4-0	PH4UP-PH0UP	R/W	Pull-up 0: Disable 1: Enable

10.2.8.10 PHIE (Port H 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	-	-	-	PH4IE	PH3IE	PH2IE	PH1IE	PH0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as "0".
4-0	PH4IE-PH0IE	R/W	Input 0: Disable 1: Enable

10.2.9 Port I (PI0 to PI7)

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

Reset initializes PI3, PI4, PI5, PI6 and PI7 as debug interface. PI7 is initialized as the TRST function with input and pull-up enabled. PI6 is initialized as the TDI function wuth input and pull-up enabled. PI3 initialized as the TCK / SWCLK function with input and pull-down enabled. PI4 is initialized as the TMS / SWDIO function with input, output and pull-up enabled. PI5 is initialized as the TDO / SWV function with output enabled.

Reset initialize the other bits as general-purpose ports with input, output and pull-up disabled.

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

Note: If PI4 or PI5 is configured as the TMS/SWDIO or TDO/SWV pin, output is enabled even in STOP1/STOP2 mode regardless of the CGSTBYCR<DRVE>/<PTKEEP> setting

10.2.9.1 Port I Register

Base Address = 0x400C_0800

Register name		Address (Base+)
Port I data register	PIDATA	0x0000
Port I output control register	PICR	0x0004
Port I function register 1	PIFR1	0x0008
Reserved	-	0x000C
Reserved	-	0x0010
Reserved	-	0x0014
Reserved	-	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Port I open drain control register	PIOD	0x0028
Port I pull-up control register	PIPUP	0x002C
Port I pull-down control register	PIPDN	0x0030
Reserved	-	0x0034
Port I input control register	PIIE	0x0038

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

10.2.9.2 PIDATA (Port I data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PI7-PI0	R/W	Port I data register.

10.2.9.3 PICR (Port I output control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PI7C	PI6C	PI5C	PI4C	PI3C	PI2C	PI1C	PI0C
After reset	0	0	1	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PI7C-PI0C	R/W	Output 0: Disable 1: Enable

10.2.9.4 PIFR1(Port I function register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PI7F1	PI6F1	PI5F1	PI4F1	PI3F1	PI2F1	PI1F1	PI0F1
After reset	1	1	1	1	1	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PI7F1	R/W	0: PORT 1: $\overline{\text{TRST}}$
6	PI6F1	R/W	0: PORT 1: TDI
5	PI5F1	R/W	0: PORT 1: TDO/SWV
4	PI4F1	R/W	0: PORT 1: TMS/SWDIO
3	PI3F1	R/W	0: PORT 1: TCK/SWCLK
2	PI2F1	R/W	0: PORT 1: TRACECLK
1	PI1F1	R/W	0: PORT 1: TRACEDATA0
0	PI0F1	R/W	0: PORT 1: TRACEDATA1

10.2.9.5 PIOD (Port I open drain control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	PI2OD	PI1OD	PI0OD
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2-0	PI2OD- PI0OD	R/W	0: Push-pull output 1: Open-drain output

10.2.9.6 PIPUP (Port I pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PI7UP	PI6UP	PI5UP	PI4UP	PI3UP	PI2UP	PI1UP	PI0UP
After reset	1	1	0	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PI7UP	R/W	Pull-up 0: Disable 1: Enable, Always set to "1" as the debug interface.
6	PI6UP	R/W	Pull-up 0: Disable 1: Enable, Always set to "1" as the debug interface.
5	PI5UP	R/W	Pull-up 0: Disable 1: Enable
4	PI4UP	R/W	Pull-up 0: Disable 1: Enable, Always set to "1" as the debug interface.
3-0	PI3UP-PI0UP	R/W	Pull-up 0: Disable 1: Enable

10.2.9.7 PIPDN (Port I pull-down 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	-	-	-	-	PI3DN	-	-	-
After reset	0	0	0	0	1	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3	PI3DN	R/W	Pull-up 0: Disable 1: Enable, Always set to "1" as the debug interface.
2-0	-	R	Read as "0".

10.2.9.8 PIIE (Port I input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PI7IE	PI6IE	PI5IE	PI4IE	PI3IE	PI2IE	PI1IE	PI0IE
After reset	0	1	0	1	1	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PI7IE-PI0IE	R/W	Input 0: Disable, Always clear to "0" as the debug interface. 1: Enable
7-0	PI7IE-PI0IE	R/W	Input 0: Disable 1: Enable, Always set to "1" as the debug interface.
7-0	PI7IE-PI0IE	R/W	Input 0: Disable, Always clear to "0" as the debug interface. 1: Enable
7-0	PI7IE-PI0IE	R/W	Input 0: Disable 1: Enable, Always set to "1" as the debug interface.
7-0	PI7IE-PI0IE	R/W	Input 0: Disable 1: Enable

10.2.10 Port J (PJ0 to PJ7)

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

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

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

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

Note 1: Unless you use all the bits of port J and port K as analog input pins, conversion accuracy may be reduced. Be sure to verify that this causes no problem on your system.

Note 2: In modes other than STOP mode, interrupt input is enabled regardless of the PxFR register setting if input is enabled in PxIE. Make sure to disable unused interrupts when programming the device.

10.2.10.1 Port J Register

Base Address = 0x400C_0900

Register name		Address (Base+)
Port J data register	PJDATA	0x0000
Port J output control register	PJCR	0x0004
Reserved	-	0x0008
Port J function register 2	PJFR2	0x000C
Port J function register 3	PJFR3	0x0010
Reserved	-	0x0014
Reserved	-	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Reserved	-	0x0028
Port J pull-up control register	PJPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port J input control register	PJIE	0x0038

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

10.2.10.2 PJDATA (Port J data register)

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

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

10.2.10.3 PJCR (Port J 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	PJ7C	PJ6C	PJ5C	PJ4C	PJ3C	PJ2C	PJ1C	PJ0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	PJ7C-PJ0C	R/W	Output 0: Disable 1: Enable

10.2.10.4 PJFR2 (Port J function register 2)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PJ7F2	R/W	0: PORT 1: INT9
6-0	-	R	Read as "0".

10.2.10.5 PJFR3 (Port J function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PJ7F3	PJ6F3	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	PJ7F3	R/W	0: PORT 1: TB0IN1
6	PJ6F3	R/W	0: PORT 1: TB0IN0
5-0	-	R	Read as "0".

10.2.10.6 PJPUP (Port J pull-up control register)

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

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

10.2.10.7 PJIE (Port J input control register)

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

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

10.2.11 Port K (PK0 to PK3)

The port K is a general-purpose, 4-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose input/output function, the port K performs the AD converter, the external interrupt input and the 16-bit timer / event counter.

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

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

To use the Port K as an analog input of the AD converter, disable input on PKIE and disable pull-up on PJPUP.

Note 1: Unless you use all the bits of port J and port K as analog input pins, conversion accuracy may be reduced. Be sure to verify that this causes no problem on your system.

Note 2: In modes other than STOP mode, interrupt input is enabled regardless of the PxFR register setting if input is enabled in PxIE. Make sure to disable unused interrupts when programming the device.

10.2.11.1 Port K Register

Base Address = 0x400C_0A00

Register name		Address (Base+)
Port K data register	KIDATA	0x0000
Port K output control register	PKCR	0x0004
Reserved	-	0x0008
Port K function register 2	PKFR2	0x000C
Port K function register 3	PKFR3	0x0010
Reserved	-	0x0014
Reserved	-	0x0018
Reserved	-	0x001C
Reserved	-	0x0020
Reserved	-	0x0024
Reserved	-	0x0028
Port K pull-up control register	PKPUP	0x002C
Reserved	-	0x0030
Reserved	-	0x0034
Port K input control register	PKIE	0x0038

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

10.2.11.2 PKDATA (Port K data register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PK3	PK2	PK1	PK0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-0	PK3-PK0	R/W	Port K data register.

10.2.11.3 PKCR (Port K 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	-	-	-	-	PK3C	PK2C	PK1C	PK0C
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-0	PK3C-PK0C	R/W	Output 0: Disable 1: Enable

10.2.11.4 PKFR2 (Port K function register 2)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	PK1F2	PK0F2
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as "0".
1	PK1F2	R/W	0: PORT 1: INT3
0	PK0F2	R/W	0: PORT 1: INT2

10.2.11.5 PKFR3 (Port K function register 3)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PK3F3	PK2F3	PK1F3	PK0F3
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3	PK3F3	R/W	0: PORT 1: TB6IN1
2	PK2F3	R/W	0: PORT 1: TB6IN0
1	PK1F3	R/W	0: PORT 1: TB1IN1
0	PK0F3	R/W	0: PORT 1: TB1IN0

10.2.11.6 PKPUP (Port K pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	PK3UP	PK2UP	PK1UP	PK0UP
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-0	PK3UP-PK0UP	R/W	Pull-up 0: Disable 1: Enable

10.2.11.7 PKIE (Port K 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	-	-	-	-	PK3IE	PK2IE	PK1IE	PK0IE
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-0	PK3IE-PK0IE	R/W	Input 0: Disable 1: Enable

10.3 Block Diagrams of Ports

10.3.1 Port Types

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

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

Table 10-5 Function Lists

Type	GP Port	Function	Analog	Pull-up	Pull-down	Programmable open-drain	Note
FT1	I/O	I/O	−	R	−	o	
FT2	I/O	I/O	−	EnR	EnR	o	Function output triggered by enable signal
FT3	I/O	I/O	−	R	−	o	Function output triggered by enable signal
FT4	I/O	Input (int)	−	R	−	o	with Noise filter
FT5	I/O	Input	o	R	-	-	ADC pin
FT6	Output	Output	−	EnR	−	o	$\overline{\text{BOOT}}$ input enabled during reset
FT7	I/O	I/O	−	R	−	−	Function output triggered by enable signal
FT8	I/O	Input	−	R	−	o	
FT9	I/O	I/O	−	R	R	o	
FT10	I/O	I/O	−	R	R	o	Function output triggered by enable signal

int: Interrupt input
−: Not exist
o: Exist
R: Forced disable during reset.
EnR: Force enable during reset.

10.3.2 Type FT1

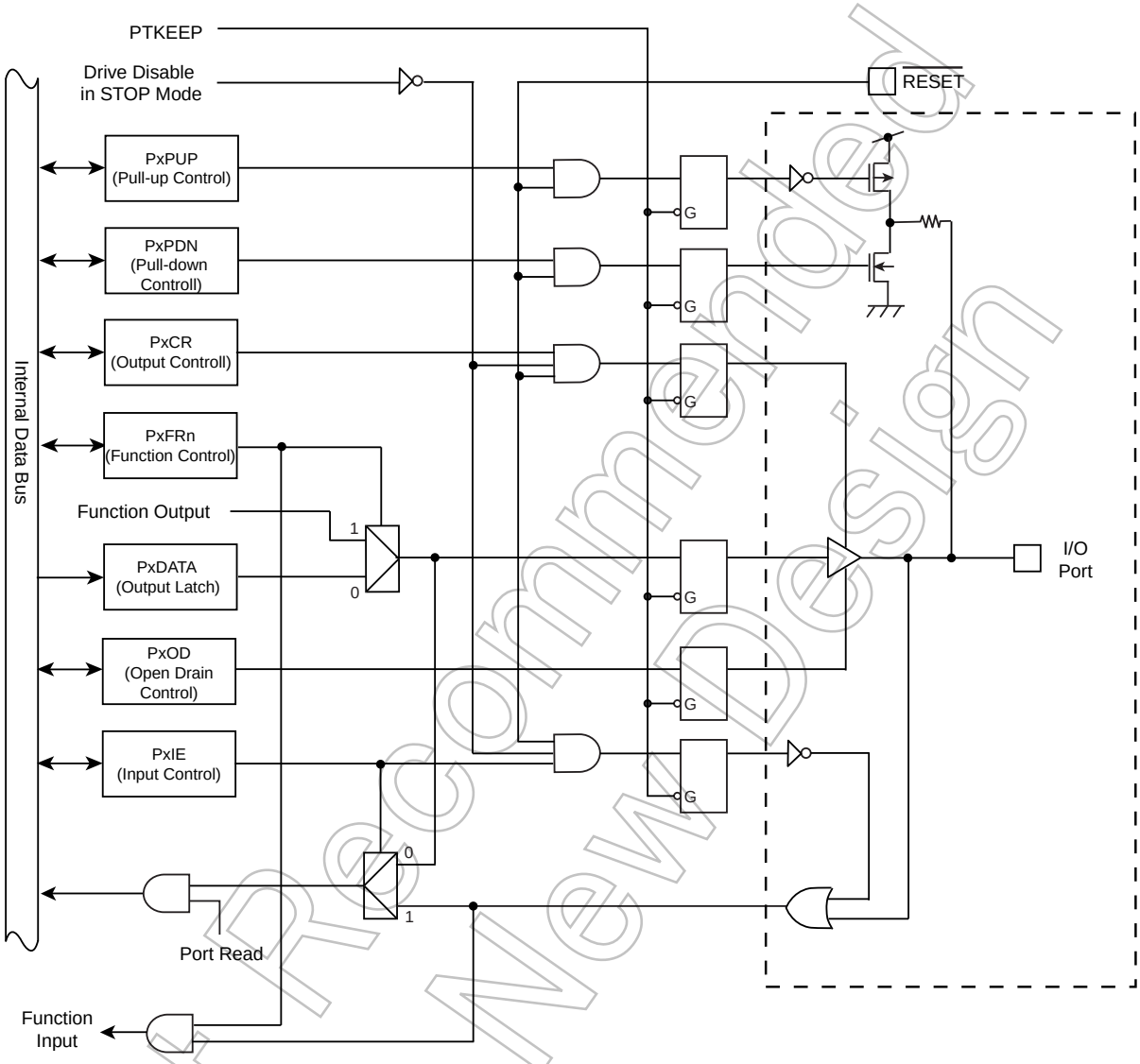


Figure 10-1 Port Type FT1

10.3.3 Type FT2

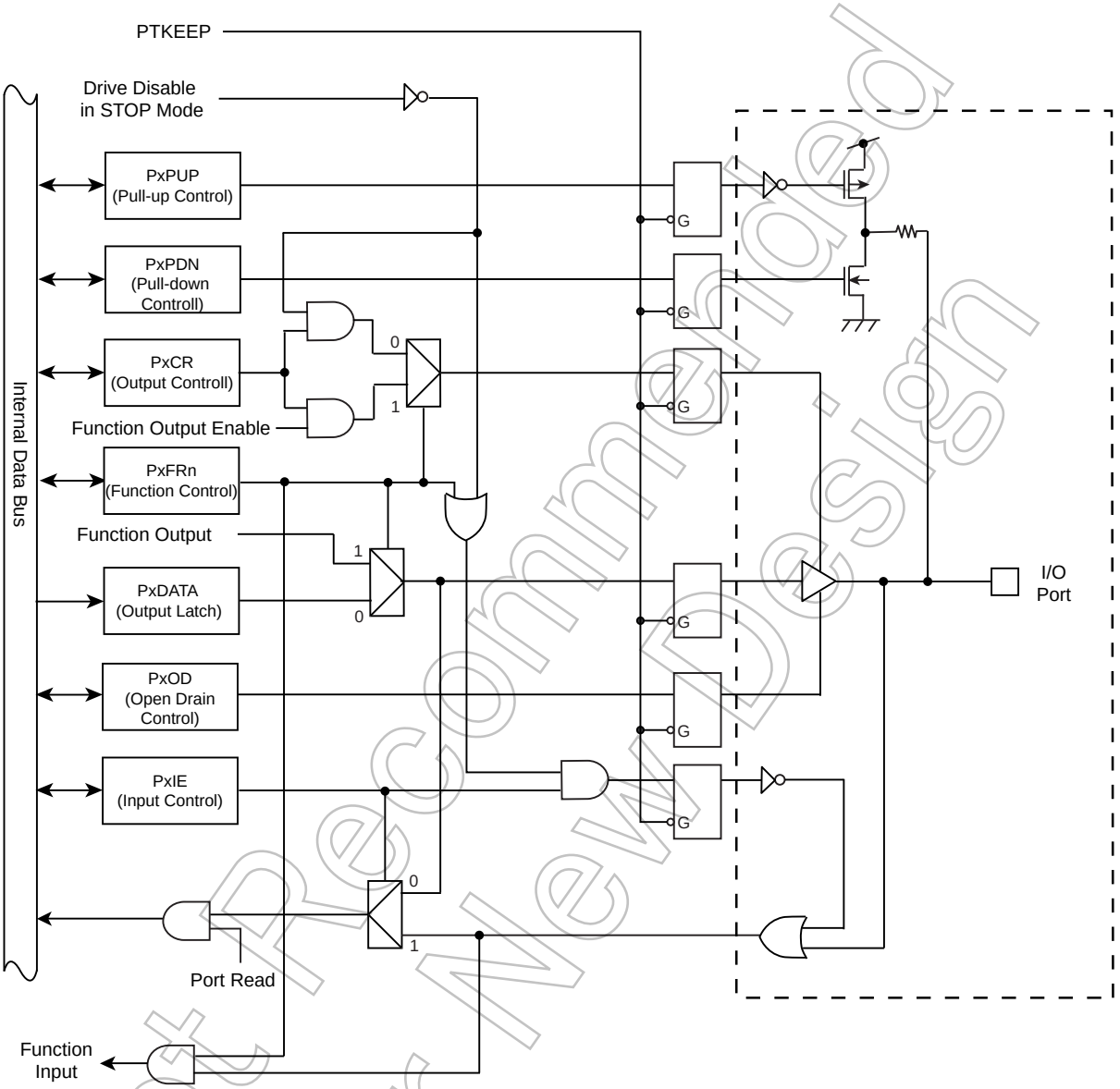


Figure 10-2 Port type FT2

Note: TRST has noise filter(30ns Typ.).

10.3.4 Type FT3

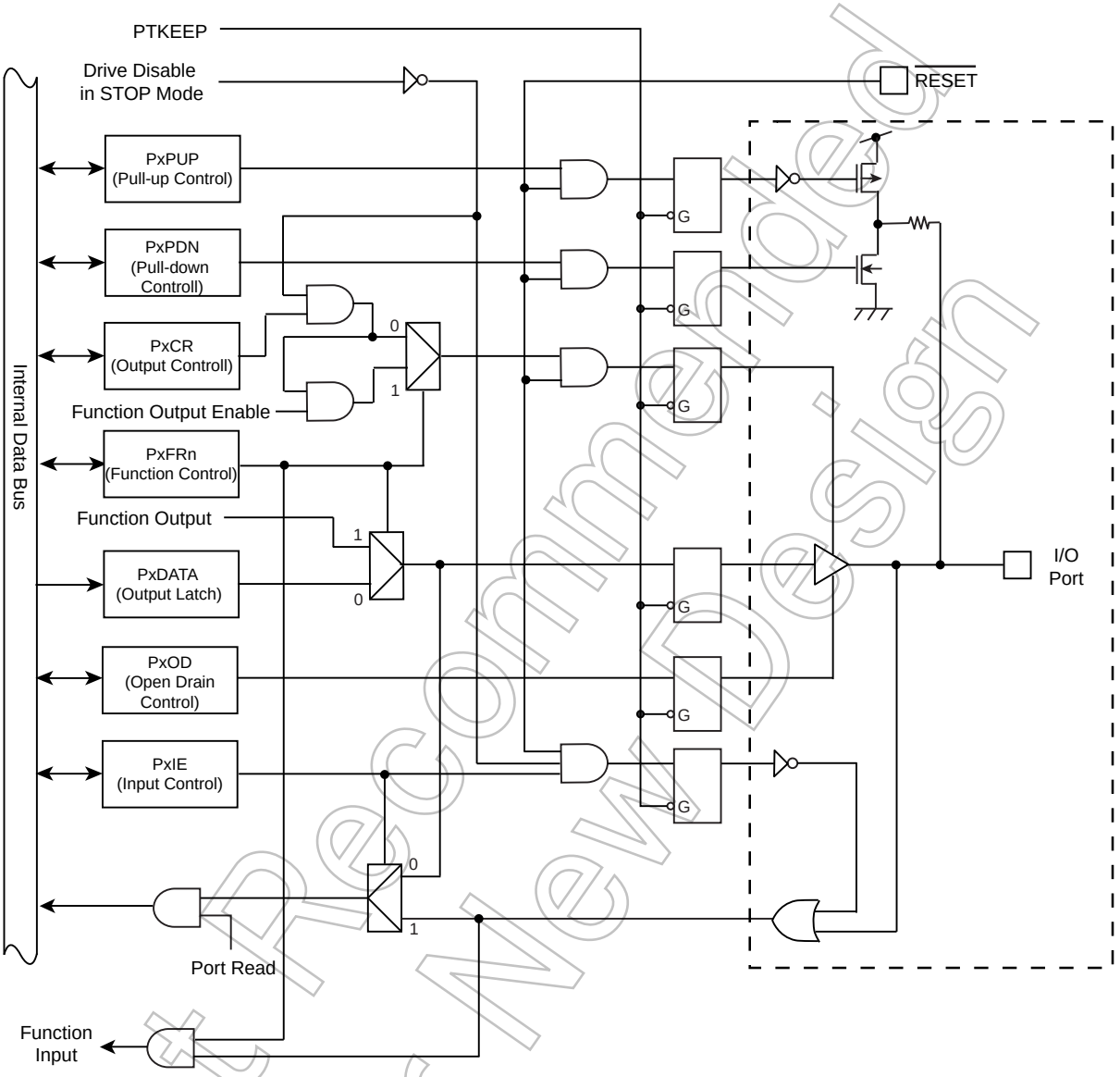


Figure 10-3 Port Type FT3

10.3.5 Type FT4

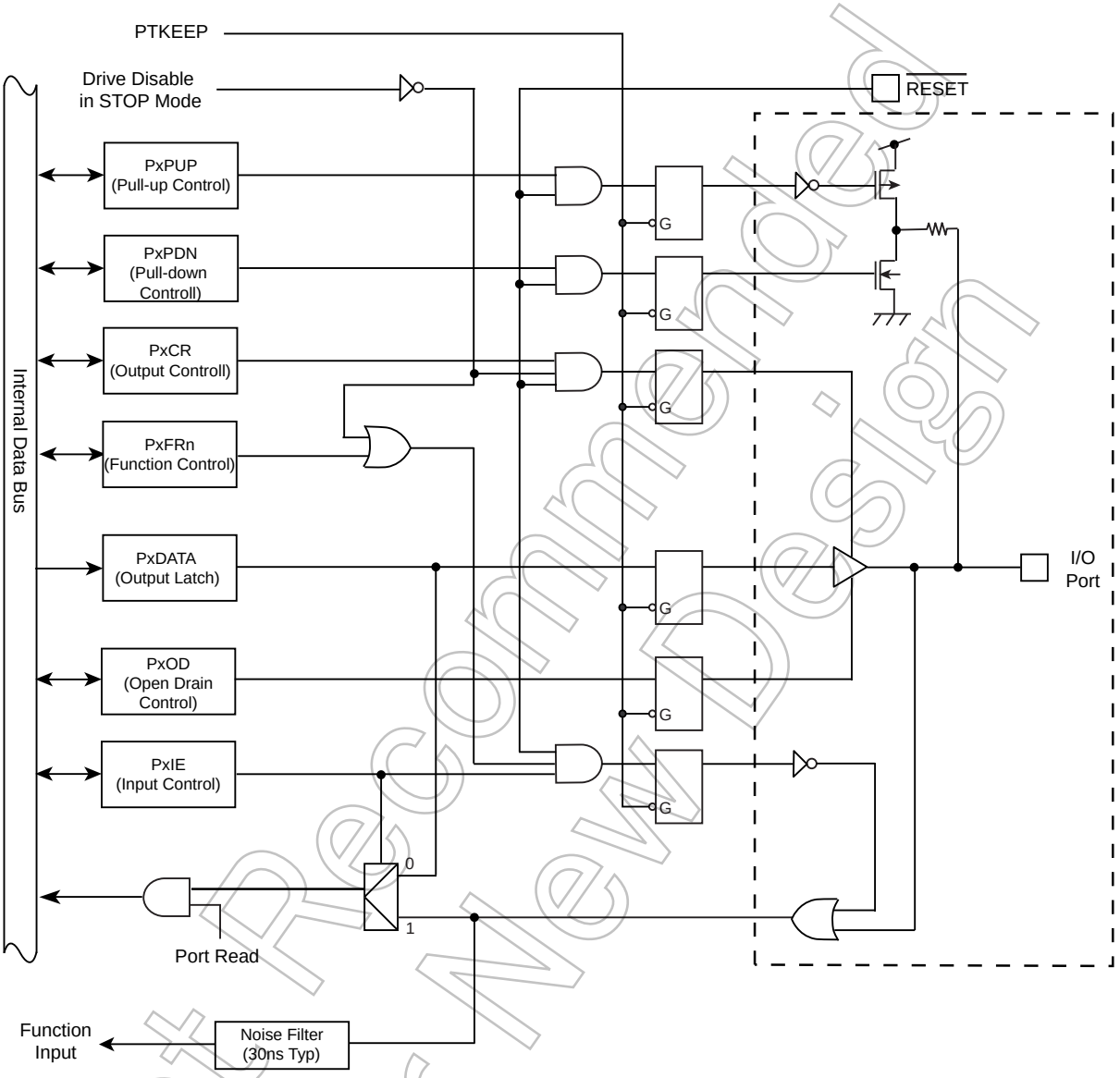


Figure 10-4 Port Type FT4

10.3.6 Type FT5

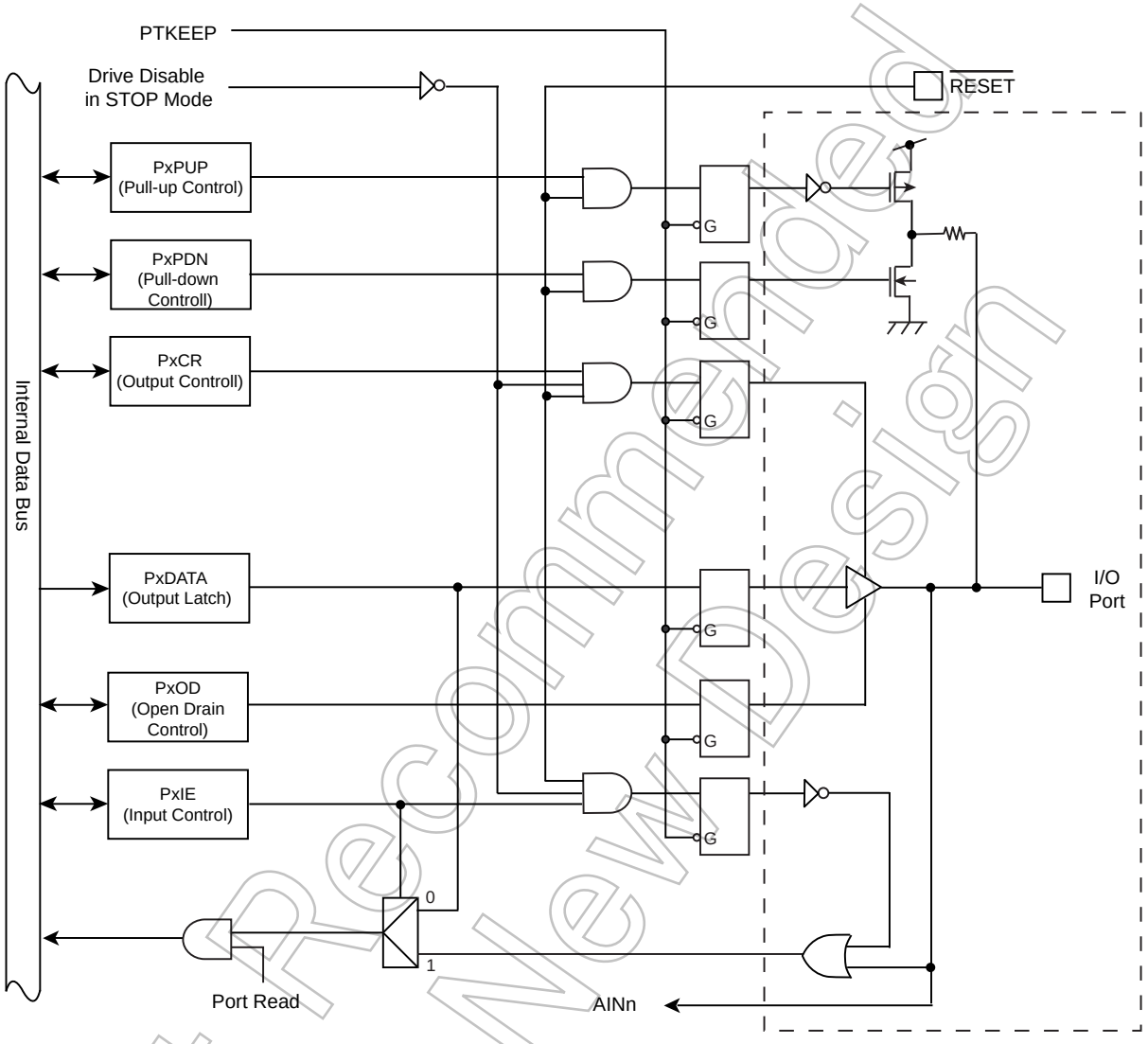


Figure 10-5 Port Type FT5

10.3.7 Type FT6

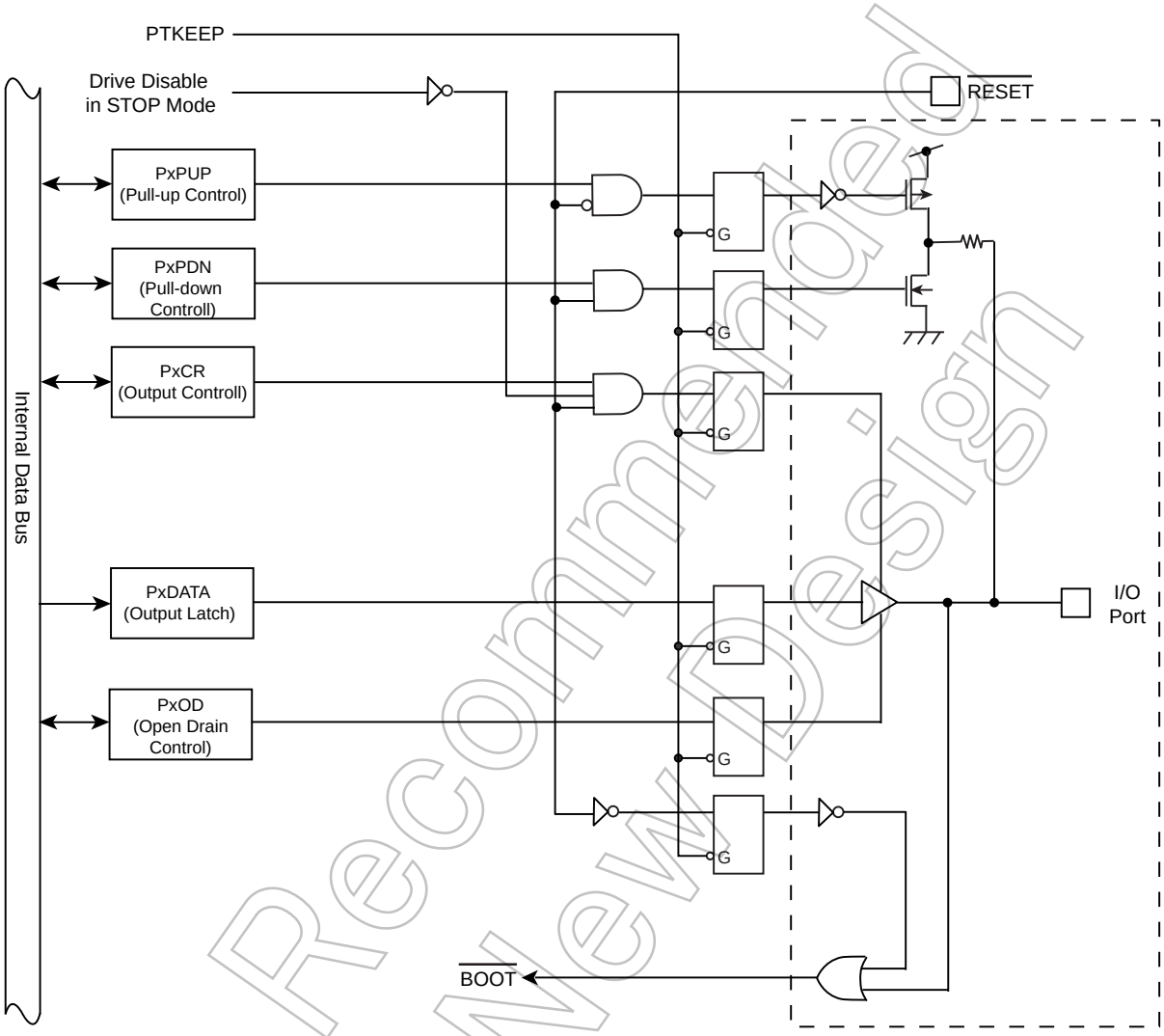


Figure 10-6 Port Type FT6

10.3.8 Type FT7

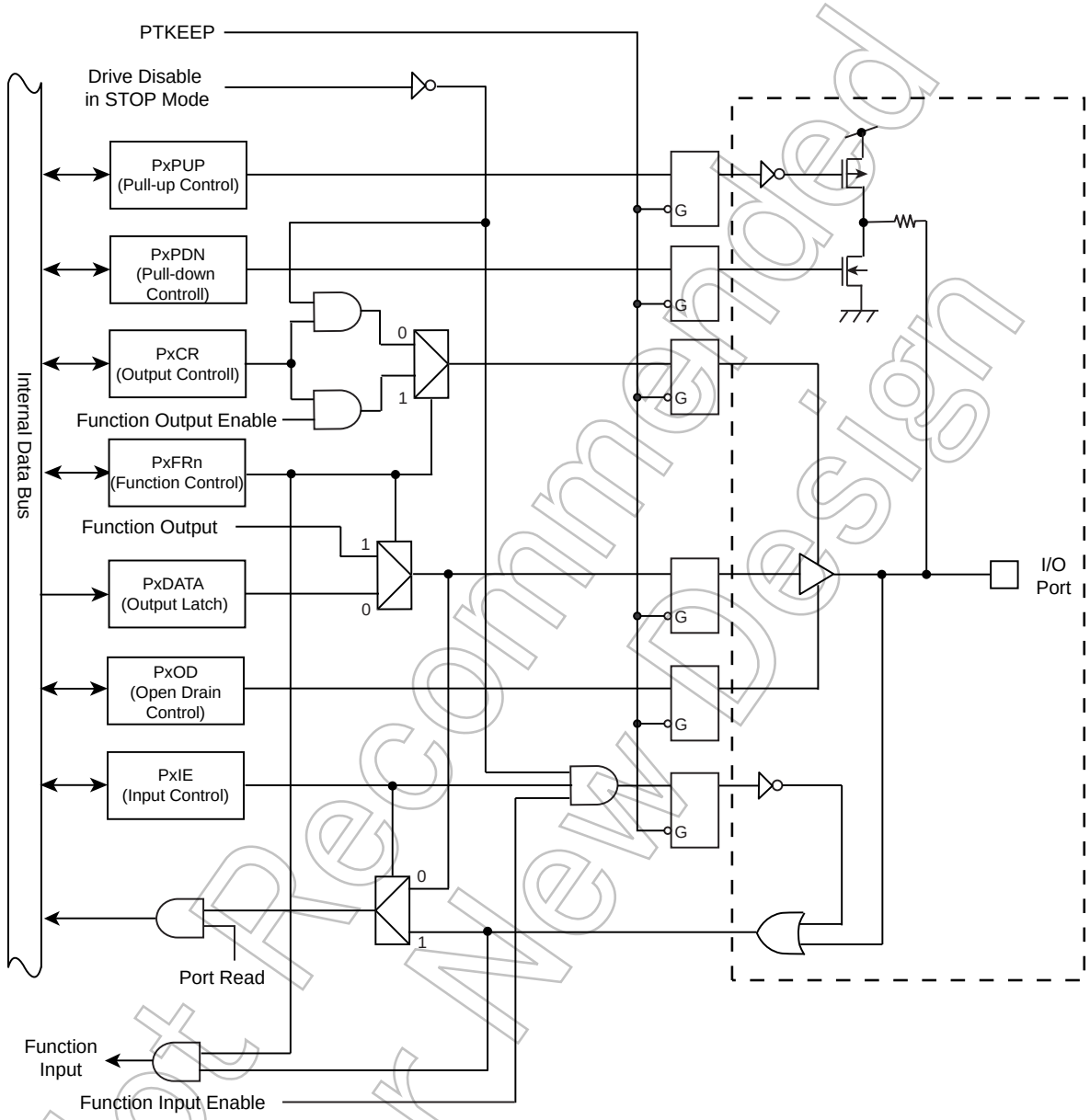


Figure 10-7 Port Type FT7

10.3.9 Type FT8

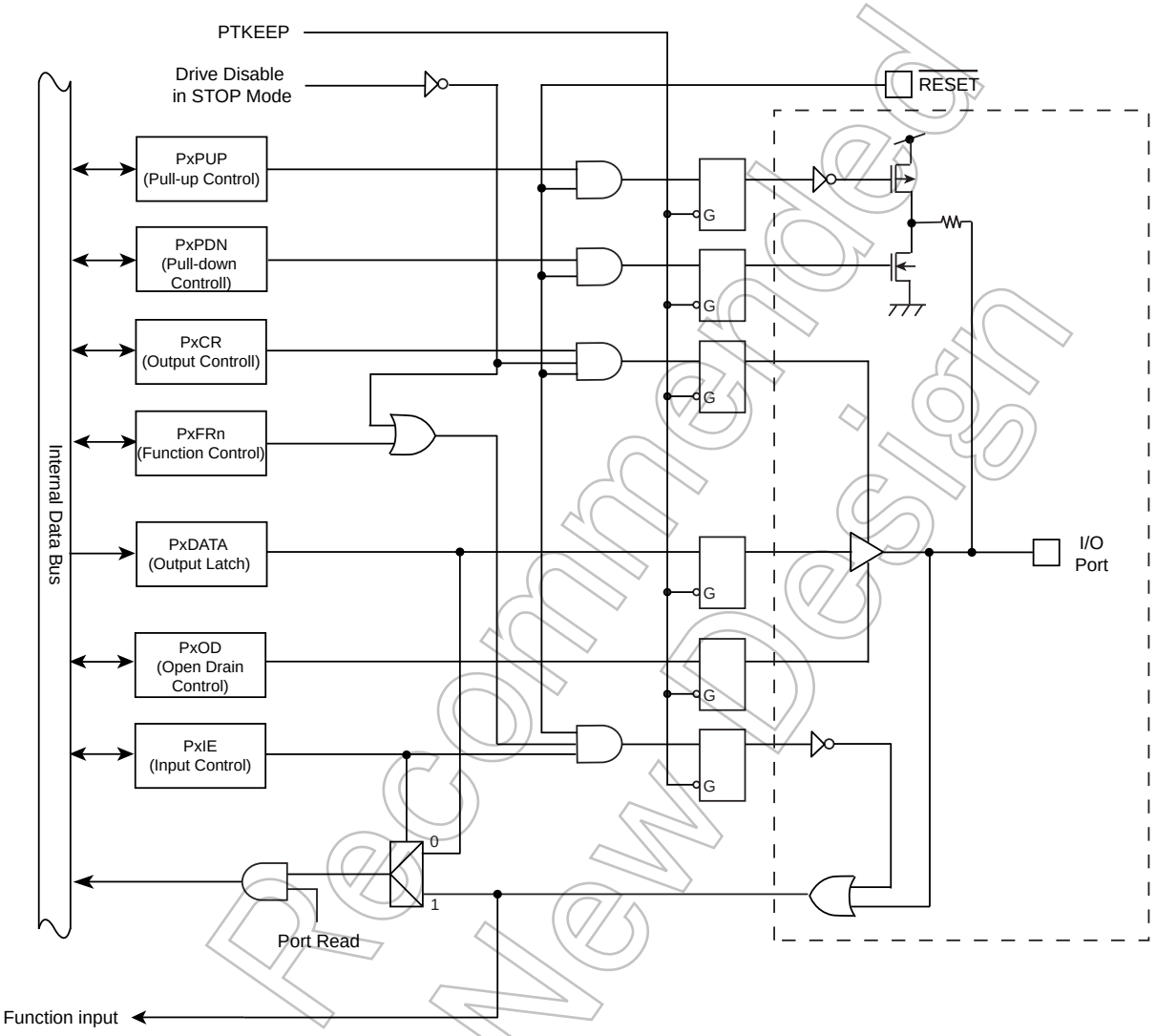


Figure 10-8 Port Type FT8

10.3.10 Type FT9

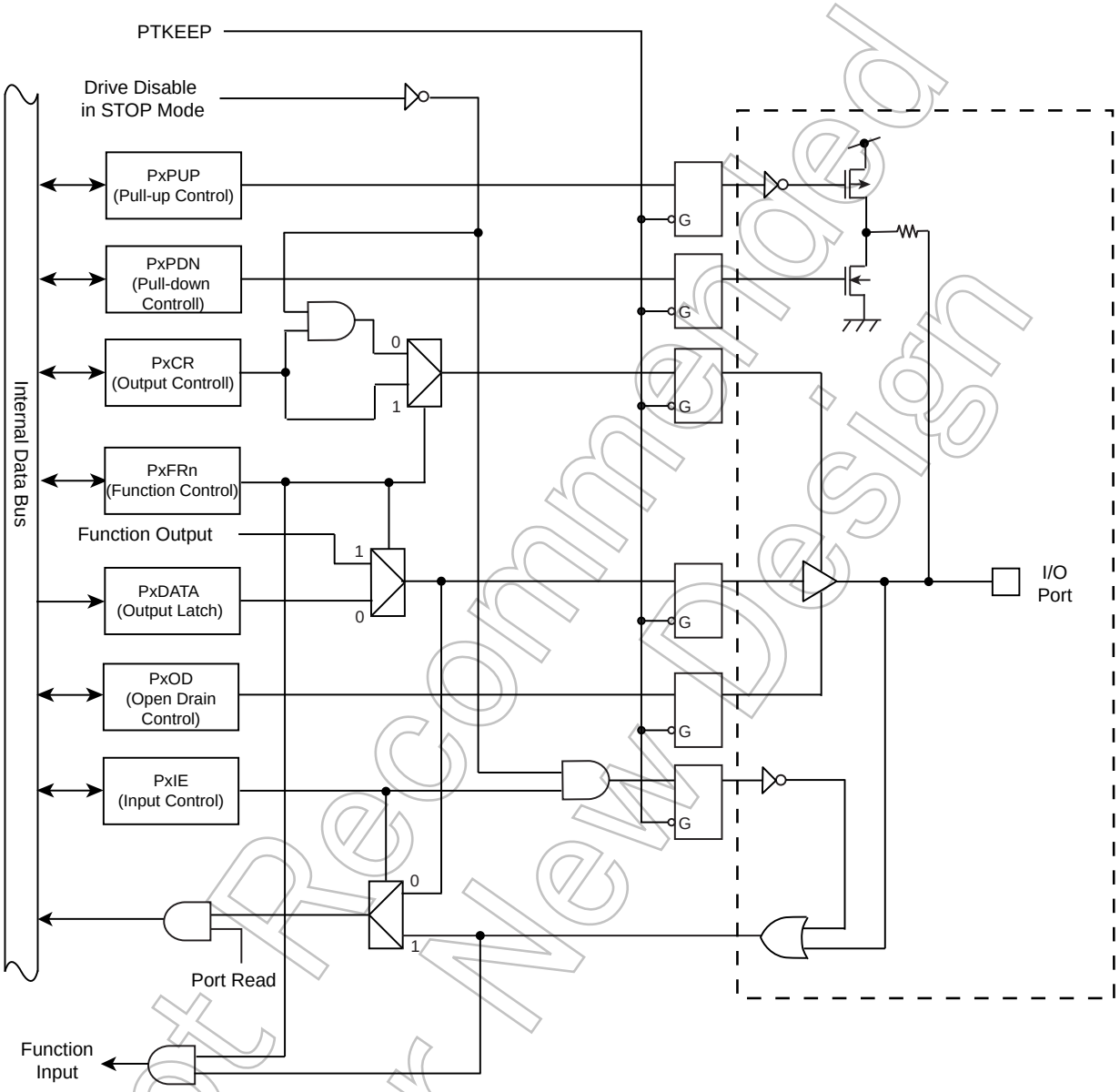


Figure 10-9 Port Type FT9

10.3.11 Type FT10

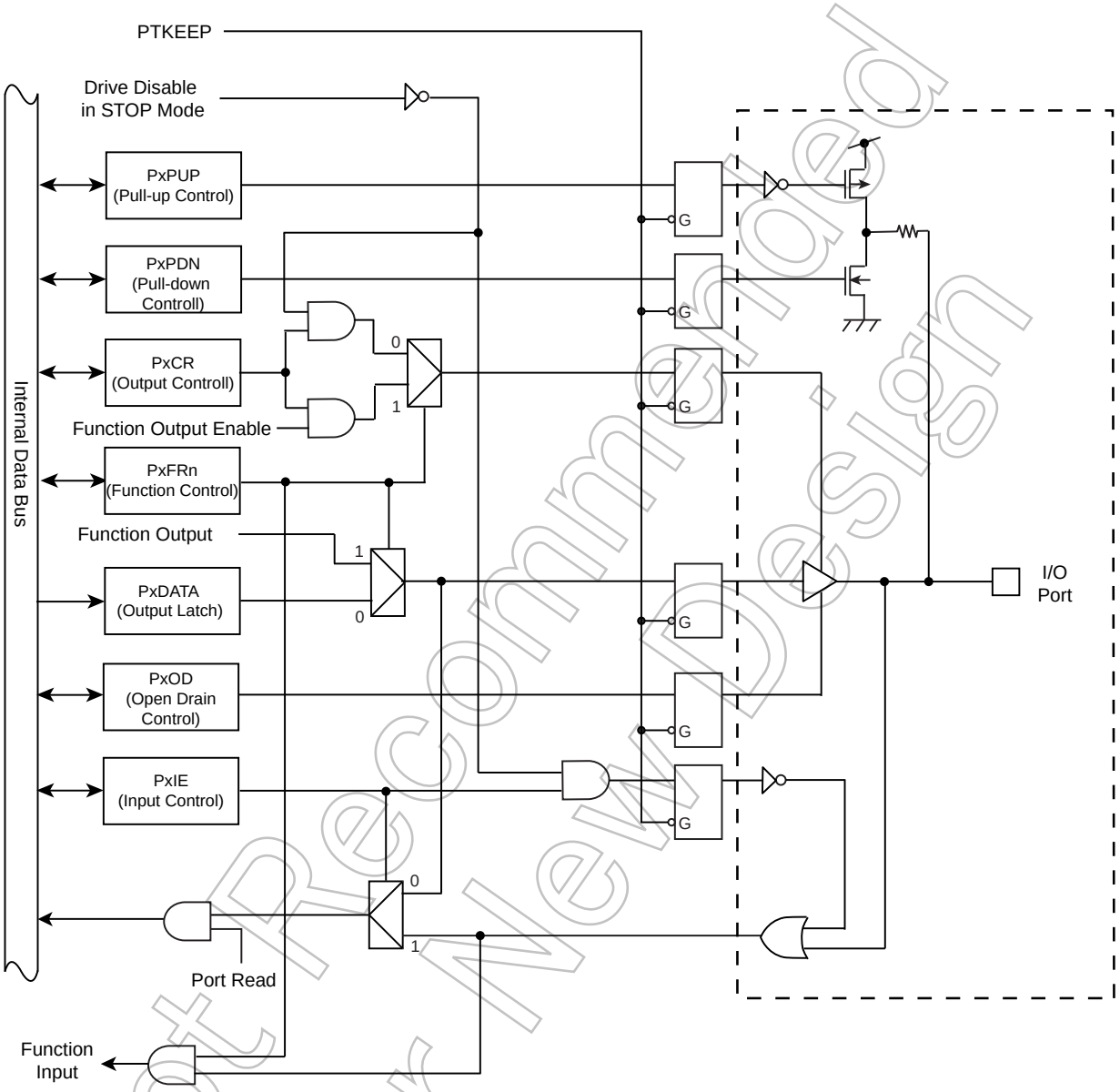


Figure 10-10 Port Type FT10

10.4 Appendix (Port setting List)

The following table shows the register setting for each function.

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

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

10.4.1 Port A Setting

Table 10-6 Port Setting List (Port A)

Pin	Port Type	Function	After reset	PACR	PAFR1	PAOD	PAPUP	PAIE
PA0	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT7	D0/AD0 (Input/Output)		1	1	0	x	1
PA1	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT7	D1/AD1 (Input/Output)		1	1	0	x	1
PA2	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT7	D2/AD2 (Input/Output)		1	1	0	x	1
PA3	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT7	D3/AD3 (Input/Output)		1	1	0	x	1
PA4	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT7	D4/AD4 (Input/Output)		1	1	0	x	1
PA5	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT7	D5/AD5 (Input/Output)		1	1	0	x	1
PA6	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT7	D6/AD6 (Input/Output)		1	1	0	x	1
PA7	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT7	D7/AD7 (Input/Output)		1	1	0	x	1

10.4.2 Port B Setting

Table 10-7 Port Setting List (Port B)

Pin	Port Type	Function	After re-set	PBCR	PBFR1	FBFR2	FBFR3	PBOD	PBPUP	PBIE
PB0	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT7	D8/AD8 (Input/Output)		1	1	0	0	x	x	1
	FT3	SP1DO (Output)		1	0	1	0	0	x	1
	FT9	A0 (Output)		1	0	0	1	0	x	0
PB1	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT7	D9/AD9 (Input/Output)		1	1	0	0	0	x	1
	FT3	SP1DI (Input)		0	0	1	0	0	x	1
	FT9	A1 (Output)		1	0	0	1	0	x	0
PB2	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT7	D10/AD10 (Input/Output)		1	1	0	0	0	x	1
	FT3	SP1CLK (Input/Output)		1	0	1	0	0	x	1
	FT9	A2 (Output)		1	0	0	1	0	x	0
PB3	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT7	D11/AD11 (Input/Output)		1	1	0	0	0	x	1
	FT3	SP1FSS(Input/Output)		1	0	1	0	0	x	1
	FT9	A3 (Output)		1	0	0	1	0	x	0
PB4	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT7	D12/AD12 (Input/Output)		1	1	0	0	0	x	1
	FT3	SP2DO (Output)		1	0	1	0	0	x	1
	FT9	A4 (Output)		1	0	0	1	0	x	0
PB5	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT7	D13/AD13 (Input/Output)		1	1	0	0	0	x	1
	FT3	SP2DI (Input)		0	0	1	0	0	x	1
	FT9	A5 (Output)		1	0	0	1	0	x	0
PB6	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT7	D14/AD14 (Input/Output)		1	1	0	0	0	x	1
	FT3	SP2CLK (Input/Output)		1	0	1	0	0	x	1
	FT9	A6 (Output)		1	0	0	1	0	x	0
PB7	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT7	D15/AD15 (Input/Output)		1	1	0	0	0	x	1
	FT3	SP2FSS (Input/Output)		1	0	1	0	0	x	1
	FT9	A7 (Output)		1	0	0	1	0	x	0

10.4.3 Port C Setting

Table 10-8 Port Setting List (Port C)

Pin	Port Type	Function	After re-set	PCCR	PCFR1	PCFR2	PCFR3	PCFR4	PCOD	PCPUP	PCIE
PC0	FT1	Input Port		0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	x	x	0
	FT1	TXD1 (Output)		1	1	0	0	0	x	x	0
	FT9	A2 (Output)		1	0	1	0	0	x	x	0
	FT1	TB2IN0 (Input)		0	0	0	1	0	x	x	1
PC1	FT1	Input Port		0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	x	x	0
	FT1	RXD1 (Input)		0	1	0	0	0	x	x	1
	FT9	A1 (Output)		0	0	1	0	0	x	x	0
	FT1	TB2IN1 (Input)		0	0	0	1	0	x	x	1
PC2	FT1	Input Port		0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	x	x	0
	FT1	SCLK1 (Input)		0	1	0	0	0	x	x	1
		SCLK1 (Output)		1	1	0	0	0	x	x	0
	FT9	A0 (Output)		1	0	1	0	0	x	x	0
	FT1	TB0OUT (Output)		1	0	0	1	0	x	x	0
		$\overline{\text{CTS1}}$ (Input)		0	0	0	0	1	x	x	1

10.4.4 Port D Setting

Table 10-9 Port Setting List (Port D)

Pin	Port Type	Function	After re-set	PDCR	PDFR2	PDFR3	PDOD	PDPUP	PDIE
PD0	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT9	A16 (Output)		1	1	0	x	x	0
	FT1	TB7OUT (Output)		1	0	1	x	x	0
PD1	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT9	A17 (Output)		1	1	0	x	x	0
	FT1	TB8OUT (Output)		1	0	1	x	x	0
PD2	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT9	A18 (Output)		1	1	0	x	x	0
	FT1	TB9OUT (Output)		1	0	1	x	x	0
PD3	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT9	A19 (Output)		1	1	0	x	x	0
	FT1	$\overline{\text{ADTRG}}$ (Input)		0	0	1	x	x	1
PD4	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT3	SP0DO (Output)		1	1	0	x	x	0
PD5	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT3	SP0DI (Input)		0	1	0	x	x	1
PD6	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT3	SP0CLK (Input)		0	1	0	x	x	1
		SP0CLK (Output)		1	1	0	x	x	0
PD7	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT3	SP0FSS (Input)		0	1	0	x	x	1
		SP0FSS (Output)		1	1	0	x	x	0
	FT1	SCOUT (Output)		1	0	1	x	x	0

10.4.5 Port E Setting

Table 10-10 Port Setting List (Port E)

Pin	Port Type	Function	After reset	PECR	PEFR1	PEFR2	PEFR3	PEFR4	PEFR5	PEOD	PEPUP	PEIE
PE0	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT1	TXD0 (Output)		1	1	0	0	0	0	x	x	0
	FT9	A20 (Output)		1	0	0	0	0	1	x	x	0
PE1	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT1	RXD0 (Input)		0	1	0	0	0	0	0	x	1
	FT9	A21 (Output)		1	0	0	0	0	1	x	x	0
PE2	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT1	SCLK0 (Input)		0	1	0	0	0	0	0	x	1
		SCLK0 (Output)		1	1	0	0	0	0	x	x	0
		TB2OUT (Output)		1	0	0	1	0	0	x	x	0
		$\overline{\text{CTS0}}$ (Input)		0	0	0	0	1	0	x	x	1
	FT9	A22 (Output)		1	0	0	0	0	1	x	x	0
PE3	FT1	Input Port		0	0	0	0	0	0	0	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT4	INT5 (Input)		0	1	0	0	0	0	0	x	1
	FT9	A15 (Output)		1	0	1	0	0	0	x	x	0
	FT1	TB3OUT (Output)		1	0	0	1	0	0	x	x	0
	FT9	A23 (Output)		1	0	0	0	0	1	x	x	0
PE4	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT1	SDA1 (Input/Output)		1	1	0	0	0	0	1	x	1
		SO1 (Output)		1	1	0	0	0	0	x	x	0
	FT9	A14 (Output)		1	0	1	0	0	0	x	x	0
PE5	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT1	SCL1 (Input/Output)		1	1	0	0	0	0	1	x	1
		SI1 (Input)		0	1	0	0	0	0	x	x	1
	FT9	A13 (Output)		1	0	1	0	0	0	x	x	0
PE6	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT1	SCK1 (Input)		0	1	0	0	0	0	x	x	1
		SCK1 (Output)		1	1	0	0	0	0	x	x	0
	FT9	A12 (Output)		1	0	1	0	0	0	x	x	0
PE7	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT4	INT4 (Input)		0	1	0	0	0	0	x	x	1
	FT9	A11 (Output)		1	0	1	0	0	0	x	x	0

10.4.6 Port F Setting

Table 10-11 Port Setting List (Port F)

Pin	Port Type	Function	After reset	PFCR	PFFR1	PFFR2	PFFR3	PFOD	PFPUP	PFIE
PF0	FT6	Output Port		1	0	0	0	x	1	0
	FT1	TB6OUT (output)		1	0	0	1	x	1	0
PF1	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT9	$\overline{RD}$ (output)		1	1	0	0	0	x	0
PF2	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT9	$\overline{WR}$ (output)		1	1	0	0	0	x	0
PF3	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT9	$\overline{BELL}$ (output)		1	1	0	0	x	x	0
PF4	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT9	$\overline{BELH}$ (output)		1	1	0	0	0	x	0
	FT4	INT6 (Input)		0	0	1	0	0	x	1
	FT1	TB5IN0 (Input)		0	0	0	1	0	x	1
PF5	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT9	$\overline{CS1}$ (output)		1	1	0	0	0	x	0
	FT4	INT7 (Input)		0	0	1	0	0	x	1
	FT1	TB5IN1 (Input)		0	0	0	1	0	x	1
PF6	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT9	$\overline{CS0}$ (output)		1	1	0	0	0	x	0
PF7	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT9	ALE (output)		1	1	0	0	0	x	0

Note: The PF0 input and pull-up are enabled and act as $\overline{BOOT}$ input pin while a $\overline{RESET}$ pin is "Low".

10.4.7 Port G Setting

Table 10-12 Port Setting List (Port G)

Pin	Port Type	Function	After reset	PGCR	PGFR1	PGFR2	PGFR3	PGFR4	PGFR5	PGOD	PGPUP	PGIE
PG0	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT1	SDA0 (Input/Output)		1	1	0	0	0	0	1	x	1
		SO0 (Output)		1	1	0	0	0	0	x	x	0
	FT9	A3 (Output)		1	0	1	0	0	0	x	x	0
	FT1	TX02 (Output)		1	0	0	0	1	0	x	x	0
		IROUT (Output)		1	0	0	0	0	1	x	x	0
PG1	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT1	SCL0 (Input/Output)		1	1	0	0	0	0	1	x	1
		SI0 (Input)		0	1	0	0	0	0	x	x	1
	FT9	A4 (Output)		1	0	1	0	0	0	x	x	0
	FT1	TB3IN0 (Input)		0	0	0	1	0	0	x	x	1
		RX02 (Input)		0	0	0	0	1	0	x	x	1
		IRIN (Input)		0	0	0	0	0	1	x	x	1
PG2	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT1	SCK0 (Input)		0	1	0	0	0	0	x	x	1
		SCK0 (Output)		1	1	0	0	0	0	x	x	0
	FT9	A5 (Output)		1	0	1	0	0	0	x	x	0
	FT1	TB3IN1 (Input)		0	0	0	1	0	0	x	x	1
		$\overline{\text{CTS2}}$ (Input)		0	0	0	0	1	0	x	x	1
PG3	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT4	INT0 (Input)		0	1	0	0	0	0	x	x	1
	FT9	A6 (Output)		1	0	1	0	0	0	x	x	0
	FT1	TB4IN0 (Input)		0	0	0	1	0	0	x	x	1
	FT1	RIN2 (Input)		0	0	0	0	1	0	x	x	1
PG4	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT9	A7 (Output)		0	0	1	0	0	0	x	x	0
	FT1	TB4IN1 (Input)		0	0	0	1	0	0	x	x	1
		$\overline{\text{RTS2}}$ (Input)		1	0	0	0	1	0	x	x	0
PG5	FT1	Input Port		0	0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	0	x	x	0
	FT4	INT1 (Input)		0	1	0	0	0	0	x	x	1
	FT4	USBPON (Input)		0	0	0	0	1	0	x	x	1

10.4.8 Port H Setting

Table 10-13 Port Setting List (Port H)

Pin	Port Type	Function	After reset	PHCR	PHFR1	PHFR2	PHFR3	PHFR4	PHOD	PHPUP	PHIE
PH0	FT1	Input Port		0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	x	x	0
	FT9	TRACEDATA2 (Output)		1	1	0	0	0	0	0	0
PH1	FT1	Input Port		0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	x	x	0
	FT9	TRACEDATA3 (Output)		0	1	0	0	0	0	0	1
PH2	FT1	Input Port		0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	x	x	0
	FT9	A10 (Output)		1	0	1	0	0	x	x	0
	FT1	TB4OUT (Output)		1	0	0	1	0	x	x	0
		DCD2 (Input)		0	0	0	0	1	x	x	1
PH3	FT1	Input Port		0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	x	x	0
	FT9	A9 (Output)		1	0	1	0	0	x	x	0
	FT1	TB5OUT (Output)		1	0	0	1	0	x	x	0
		DSR2 (Input)		0	0	0	0	1	x	x	1
PH4	FT1	Input Port		0	0	0	0	0	x	x	1
		Output Port		1	0	0	0	0	x	x	0
	FT9	A8 (Output)		1	0	1	0	0	x	x	0
	FT4	INT8 (Input)		0	0	0	1	0	x	x	1
	FT1	DTR2 (Output)		1	0	0	0	1	x	x	0

10.4.9 Port I Setting

Table 10-14 Port Setting List (Port I)

Pin	Port Type	Function	After re-set	PICR	PIFR1	PIOD	PIPUP	PIPDN	PIIE
PI0	FT1	Input Port		0	0	x	x	x	1
		Output Port		1	0	x	x	x	0
	FT9	TRACEDATA1 (Output)		1	1	0	0	0	0
PI1	FT1	Input Port		0	0	x	x	x	1
		Output Port		1	0	x	x	x	0
	FT9	TRACEDATA0 (Output)		1	1	0	0	0	0
PI2	FT1	Input Port		0	0	x	x	x	1
		Output Port		1	0	x	x	x	0
	FT9	TRACECLK (Output)		1	1	0	0	0	0
PI3	FT1	Input Port		0	0	x	x	x	1
		Output Port		1	0	x	x	x	0
	FT2	TCK (Input)/ SWCLK (Input)	o	0	1	0	0	1	1
PI4	FT1	Input Port		0	0	x	x	x	1
		Output Port		1	0	x	x	x	0
	FT2	TMS (Input)/ SWDIO (Input/Output)	o	1	1	0	1	0	1
PI5	FT1	Input Port		0	0	x	x	x	1
		Output Port		1	0	x	x	x	0
	FT2	TDO (Output)/ SWV (Output)	o	1	1	0	0	0	0
PI6	FT1	Input Port		0	0	x	1	x	1
		Output Port		1	0	x	1	x	0
	FT2	TDI (Input)	o	0	1	0	1	0	1
PI7	FT1	Input Port		0	0	x	x	x	1
		Output Port		1	0	x	x	x	0
	FT2	$\overline{\text{TRST}}$ (Input)	o	0	1	0	1	0	1

10.4.10 Port J Setting

Table 10-15 Port Setting List (Port J)

pin	Port Type	Function	After re-set	PJCR	PJFR2	PJFR3	PJPUP	PJIE
PJ0	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA00 (Input)		0	0	0	0	0
PJ1	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA01 (Input)		0	0	0	0	0
PJ2	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA02 (Input)		0	0	0	0	0
PJ3	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA03 (Input)		0	0	0	0	0
PJ4	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA04 (Input)		0	0	0	0	0
PJ5	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA05 (Input)		0	0	0	0	0
PJ6	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA06 (Input)		0	0	0	0	0
	FT1	TB0IN0 (Input)		0	0	1	x	1
PJ7	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA07 (Input)		0	0	0	0	0
	FT4	INT9 (Input)		0	1	0	x	1
	FT1	TB0IN1 (Input)		0	0	1	x	1

10.4.11 Port K Setting

Table 10-16 Port Setting List (Port K)

Pin	Port Type	Function	After re-set	PKCR	PKFR2	PKFR3	PKPUP	PKIE
PK0	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA08 (Input)		0	0	0	0	0
	FT4	INT2 (Input)		0	1	0	x	1
	FT1	TB1IN0 (Input)		0	0	1	x	1
PK1	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA09 (Input)		0	0	0	0	0
	FT4	INT3 (Input)		0	1	0	x	1
	FT1	TB1IN1 (Input)		0	0	1	x	1
PK2	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA10 (Input)		0	0	0	0	0
	FT1	TB6IN0 (Input)		0	0	1	x	1
PK3	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
	FT5	AINA11 (Input)		0	0	0	0	0
	FT1	TB6IN1 (Input)		0	0	1	x	1

11. DMA Controller(DMAC)

11.1 Overview

The table below lists its major functions.

Table 11-1 DMA controller functions (1 Unit)

Item	Function		Description
Number of channels	2ch		-
Number of DMA request	16		-
DMA Start up trigger	Hardware start		Started with DMA request for peripheral circuit.
	Software start		Started with a write to the DMACxSoftBReq register.
Bus master	32bit × 1 (AHB)		-
Priority	High : Unit A CH0 Unit A CH1 Unit B CH0 Low : Unit B CH1		Fixed
FIFO	4word × 2ch (1word = 32bit)		-
Bus width	8/16/32bit		Settable individually for transfer source and destination.
Burst size	1/4/8/16/32/64/128/256		-
Number of transfers	up to 4095		-
Address	Transfer source address	increment not increment	It is possible to specify whether Source and Destination addresses should increment or should not increment. (Address wrapping is not supported.)
	Transfer destination address	increment not increment	
Endian	Littleendian is supported.		-
Transfer type	Peripheral to Memory Memory to Peripheral Memory to Memory Peripheral to Peripheral		When "Memory to Memory" is selected, hardware start for DMA startup is not supported. Refer to the DMACxCnConfiguration for more information. Particular peripheral can be assigned as Source or Distination when "Peripheral to Peripheral" is selected. Regarding to peripheral assigned, refer to "11.4.1 Peripheral function supported with Peripheral to Peripheral Transfer".
Interrupt function	Transfer end interrupt (INTDMACxTC) Error interrupt (INTDMACxERR)		-
Special Function	Scatter/gather function		-

11.2 DMA transfer type

Table 11-2 DMA transfer type

<table>
<tr><th>No.</th><th>DMA transfer type</th><th>Circuit generated DMA request</th><th>DMA request type</th><th>Description</th></tr>
<tr><td>1</td><td>Memory to Peripheral</td><td>Peripheral (Destination)</td><td>Burst request</td><td>In case of 1word transmission, set to the "1" for burst size of DMA controller.</td></tr>
<tr><td>2</td><td>Peripheral to Memory</td><td>Peripheral (Source)</td><td>Burst request / single request</td><td>If the amount of transfer data is not an integral multiple of the burst size, both burst and single request can be used.
If amount of transfer data is more or equal than burst size, the single request is ignored and the burst transfer is used.
If it becomes less than burst size, the single transfer is used.</td></tr>
<tr><td>3</td><td>Memory to Memory</td><td>DMAC</td><td>None</td><td>Enabling the DMAC starts data transfer without DMAC request.
(Select Memory to Memory mode, set DMACxCnConfiguration<E> to "1")
When All transfer data is transferred completely or when the DMAC channel is disabled, DMAC is stopped.</td></tr>
<tr><td rowspan="2">4</td><td rowspan="2">Peripheral to Peripheral</td><td>Peripheral (Source)</td><td>Burst request / single request</td><td rowspan="2"><table><tr><th>Transfer size</th><th>Source</th><th>Destination</th></tr><tr><td>(1)An integral multiple of the burst size</td><td>Burst request</td><td>Burst request</td></tr><tr><td>(2)Not an integral multiple of the burst size</td><td>Burst request / single request</td><td>-</td></tr></table></td></tr>
<tr><td>Peripheral (Destination)</td><td>Burst request</td></tr>
</table>

Note: When much data is transferred in memory to memory, we recommend that a lower priority channel is used. If a lower priority channel is used, a higher priority channel can be started to transfer during a lower priority channel is transferring. If a higher priority channel is used, a lower priority channel can not be started to transfer during a higher priority channel is transferring.

11.3 Block diagram

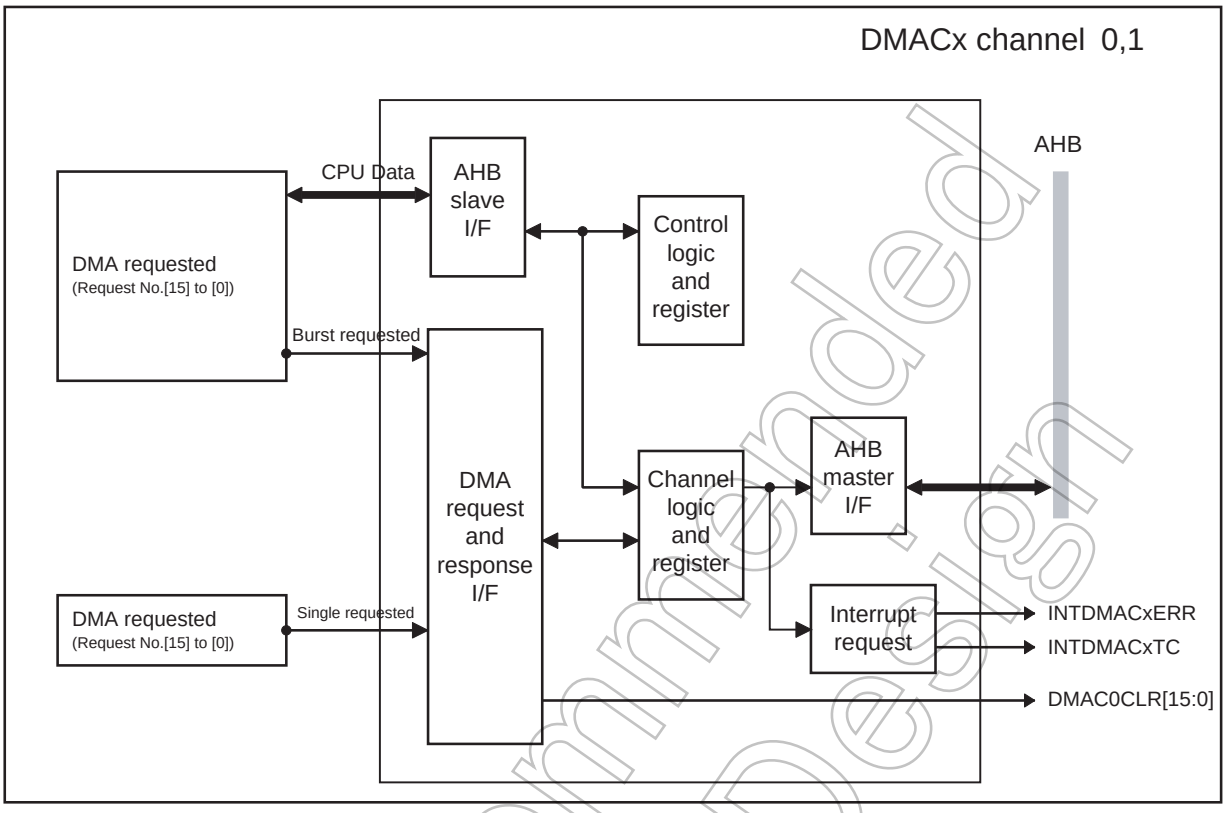


Figure 11-1 DMAC Block diagram

11.4 Product information of TMPM366FDXBG/FYXBG/FWXBG

11.4.1 Peripheral function supported with Peripheral to Peripheral Transfer

Peripheral functions (Register) supported with Peripheral to Peripheral Transfer are shown below.

Source	Destination
Peripheral register	SCxBUF (x=0 to 1)
	TBxREG0 to 1 (x=0 to 9)
	TBxCP0 to 1 (x=0 to 9)
SCxBUF (x=0 to1)	Peripheral register
TBxREG0 to 1 (x=0 to 9)	
TBxCP0 to 1 (x=0 to 9)	

11.4.2 DMA request

DMA request against each DMA request no. are shown as bellows.

Table 11-3 DMA request factor (Unit A)

DMA request No.	Corresponding peripheral	
	ch0,ch1	
	Burst request	Single request
0	SIO0/UART0 Reception	-
1	SIO0/UART0 Transmission	-
2	SIO1/UART1 Reception	-
3	SIO1/UART1 Transmission	-
4	TMRB8 compare match	-
5	TMRB9 compare match	-
6	TMRB0 input capture 0	-
7	TMRB4 input capture 0	-
8	TMRB4 input capture 1	-
9	TMRB5 input capture 0	-
10	TMRB5 input capture 1	-
11	High priority AD conversion end	-
12	-	-
13	-	-
14	UART reception	UART reception
15	UART transmission	UART transmission

Table 11-4 DMA request factor (Unit B)

DMA request No.	Corresponding peripheral	
	ch0,ch1	
	Burst request	Single request
0	TMRB6 compare match	-
1	TMRB7 compare match	-
2	TMRB0 input capture 1	-
3	TMRB2 input capture 0	-
4	TMRB2 input capture 1	-
5	TMRB3 input capture 0	-
6	TMRB3 input capture 1	-
7	TMRB6 input capture 0	-
8	TMRB6 input capture 1	-
9	Normal AD conversion end	-
10	SSP0 transmission	SSP0 transmission
11	SSP0 reception	SSP0 reception
12	SSP1 transmission	SSP1 transmission
13	SSP1 reception	SSP1 reception
14	SSP2 transmission	SSP2 transmission
15	SSP2 reception	SSP2 reception

11.4.3 Interrupt request

11.4.4 Base address of registers

Unit	Base Address
Unit A	0x4000_0000
Unit B	0x4000_1000

11.5 Description of Registers

11.5.1 DMAC register list

The function and address for each register are shown bellow.

Register Name		Address (Base+)
DMAC Interrupt Status Register	DMACxIntStaus	0x0000
DMAC Interrupt Terminal Count Status Register	DMACxIntTCStatus	0x0004
DMAC Interrupt Terminal Count Clear Register	DMACxIntTCClear	0x0008
DMAC Interrupt Error Status Register	DMACxIntErrorStatus	0x000C
DMAC Interrupt Error Clear Register	DMACxIntErrClr	0x0010
DMAC Raw Interrupt Terminal Count Status Register	DMACxRawIntTCStatus	0x0014
DMAC Raw Error Interrupt Status Register	DMACxRawIntErrorStatus	0x0018
DMAC Enabled Channel Register	DMACxEnbldChns	0x001C
DMAC Software Burst Request Register	DMACxSoftBReq	0x0020
DMAC Software Single Request Register	DMACxSoftSReq	0x0024
Reserved	-	0x0028
Reserved	-	0x002C
DMAC Configuration Register	DMACxConfiguration	0x0030
Reserved	-	0x0034
DMAC Channel0 Source Address Register	DMACxC0SrcAddr	0x0100
DMAC Channel0 Destination Address Register	DMACxC0DestAddr	0x0104
DMAC Channel0 Linked List Item Register	DMACxC0LLI	0x0108
DMAC Channel0 Control Register	DMACxC0Control	0x010C
DMAC Channel0 Configuration Register	DMACxC0Configuration	0x0110
DMAC Channel1 Source Address Register	DMACxC1SrcAddr	0x0120
DMAC Channel1 Destination Address Register	DMACxC1DestAddr	0x0124
DMAC Channel1 Linked List Item Register	DMACxC1LLI	0x0128
DMAC Channel1 Control Register	DMACxC1Control	0x012C
DMAC Channel 1 Configuration Register	DMACxC1Configuration	0x0130

Note 1: Access the registers by using word (32bit) reads and word writes.

Note 2: Access to the "Reserved" area is prohibited.

Note 3: For the registers prepared for every channel, if the channel structure is the same, unit number is expressed as "x" and channel number is expresses as "n".

Note 4: When the register which is not assigned with an each channel is read after the register which is assigned with an each channel is written, one machine cycle is inserted between the instructions or read the register which is not assigned with an each channel twice.

11.5.2 DMACxIntStatus (DMAC Interrupt Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntStatus1	IntStatus0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntStatus1	R	Status of DMAC channel 1 transfer end interrupt. 0 : Interrupt not requested 1 : Interrupt requested Status of the DMAC interrupt generation after passing through the transfer end interrupt enable register and error interrupt enable register. An interrupt is requested when there is a transfer error or when the counter completes counting.
0	IntStatus0	R	Status of DMAC channel 0 interrupt generation. 0 : Interrupt not requested 1 : Interrupt requested Status of the DMAC interrupt generation after passing through the transfer end interrupt enable register and error interrupt enable register. An interrupt is requested when there is a transfer error or when the counter completes counting.

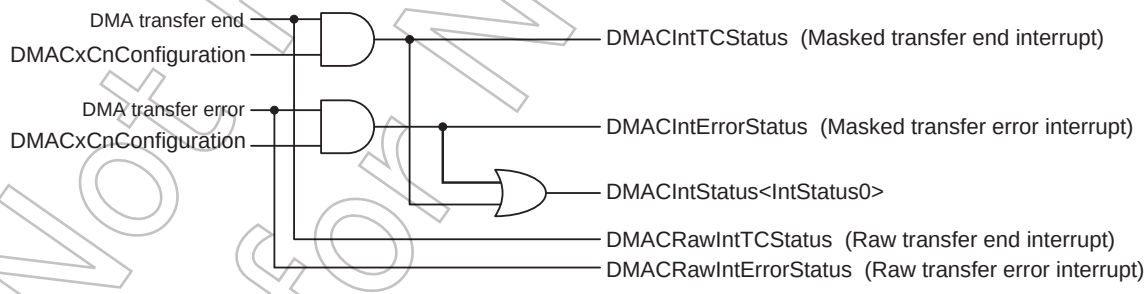


Figure 11-2 Interrupt-related block diagram

11.5.3 DMACxIntTCStatus (DMAC Interrupt Terminal Count Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntTCStatus1	IntTCStatus0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntTCStatus1	R	Status of DMAC channel 1 transfer end interrupt. 0 : Interrupt not requested 1 : Interrupt requested The status of post-enable transfer end interrupt generation.
0	IntTCStatus0	R	Status of DMAC channel 0 transfer end interrupt. 0 : Interrupt not requested 1 : Interrupt requested The status of post-enable transfer end interrupt generation.

11.5.4 DMACxIntTCClear (DMAC Interrupt Terminal Count Clear Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntTCClear1	IntTCClear0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntTCClear1	W	Clear DMAC channel 1 transfer end interrupt. 0 : Do nothing 1 : Clear The DMACxIntTCStatus<IntTCStatus1> will be cleared when "1" is written.
0	IntTCClear0	W	Clear DMAC channel 0 transfer end interrupt. 0 : Do nothing 1 : Clear The DMACxIntTCStatus<IntTCStatus0> will be cleared when "1" is written.

11.5.5 DMACxIntErrorStatus (DMAC Interrupt Error Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntErrStatus1	IIntErrStatus0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntErrStatus1	R	Status of DMAC channel 1 error interrupt generation. 0 : Interrupt not requested 1 : Interrupt requested Shows error interrupt status after enabled.
0	IntErrStatus0	R	Status of DMAC channel 0 error interrupt generation. 0 : Interrupt not requested 1 : Interrupt requested Shows error interrupt status after enabled.

11.5.6 DMACxIntErrClr (DMAC Interrupt Error Clear Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	IntErrClr1	IntErrClr0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	IntErrClr1	W	Clear DMAC channel 1 transfer end interrupt. 0 : Do nothing 1 : Clear The DMACxIntErrorStatus<IntErrStatus1> will be cleared when "1" is written.
0	IntErrClr0	W	Clear DMAC channel 0 transfer end interrupt. 0 : Do nothing 1 : Clear The DMACxIntErrorStatus<IntErrStatus0> will be cleared when "1" is written.

11.5.7 DMACxRawIntTCStatus (DMAC Raw Interrupt Terminal Count Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	RawIntTCS1	RawIntTCS0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	RawIntTCS1	R	Status of DMAC channel 1 pre-enable transfer end interrupt generation 0 : Interrupt not requested 1 : Interrupt requested
0	RawIntTCS0	R	Status of DMAC channel 0 pre-enable transfer end interrupt generation 0 : Interrupt not requested 1 : Interrupt requested

11.5.8 DMACxRawIntErrorStatus (DMAC Raw Error Interrupt Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	RawIntErrS1	RawIntErrS0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	RawIntErrS1	R	Status of DMAC channel 1 pre-enable error interrupt. 0 : Interrupt not requested 1 : Interrupt requested
0	RawIntErrS0	R	Status of DMAC channel 0 pre-enable error interrupt. 0 : Interrupt not requested 1 : Interrupt requested

11.5.9 DMACxEnbldChns (DMAC Enabled Channel Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	EnabledCH1	EnabledCH0
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	EnabledCH1	R	DMA channel 1 enable status. 0 : Disable 1 : Enable After finishing all the total transfer number of times in DMACxCnControl register (the value becomes the zero), the this flag is clearded.
0	EnabledCH0	R	DMA channel 0 enable status. 0 : Disable 1 : Enable After finishing all the total transfer number of times in DMACxCnControl register (the value becomes the zero), the this flag is clearded.

11.5.10 DMACxSoftBReq (DMAC Software Burst Request Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	SoftBReq15	SoftBReq14	SoftBReq13	SoftBReq12	SoftBReq11	SoftBReq10	SoftBReq9	SoftBReq8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SoftBReq7	SoftBReq6	SoftBReq5	SoftBReq4	SoftBReq3	SoftBReq2	SoftBReq1	SoftBReq0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	-	Write as zero.
15	SoftBReq15	R/W	DMA burst request by software (Request No. [15]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
14	SoftBReq14	R/W	DMA burst request by software (Request No. [14]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
13	SoftBReq13	R/W	DMA burst request by software (Request No. [13]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
12	SoftBReq12	R/W	DMA burst request by software (Request No. [12]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
11	SoftBReq11	R/W	DMA burst request by software (Request No. [11]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
10	SoftBReq10	R/W	DMA burst request by software (Request No. [10]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
9	SoftBReq9	R/W	DMA burst request by software (Request No. [9]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested

Bit	Bit Symbol	Type	Function
8	SoftBReq8	R/W	DMA burst request by software (Request No. [8]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
7	SoftBReq7	R/W	DMA burst request by software (Request No. [7]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
6	SoftBReq6	R/W	DMA burst request by software (Request No. [6]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
5	SoftBReq5	R/W	DMA burst request by software (Request No. [5]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
4	SoftBReq4	R/W	DMA burst request by software (Request No. [4]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
3	SoftBReq3	R/W	DMA burst request by software (Request No. [3]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
2	SoftBReq2	R/W	DMA burst request by software (Request No. [2]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
1	SoftBReq1	R/W	DMA burst request by software (Request No. [1]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested
0	SoftBReq0	R/W	DMA burst request by software (Request No. [0]) Read : 0 :Stopping DMA burst transfer 1 : running DMA burst transfer Write: 0 : invaild 1 : DMA burst requested

Note 1: Do not execute DMA requests by software and hardware at the same time.

Note 2: Refer to "11.4.2 DMA request" for DMA request number. Clear "0" to bit corresponded with the DMA request number which has no burst request.

11.5.11 DMACxSoftSReq (DMAC Software Single Request Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	15	14	13	12	11	10	9	8
bit symbol	SoftSReq15	SoftSReq14	SoftSReq13	SoftSReq12	SoftSReq11	SoftSReq10	SoftSReq9	SoftSReq8
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	SoftSReq7	SoftSReq6	SoftSReq5	SoftSReq4	SoftSReq3	SoftSReq2	SoftSReq1	SoftSReq0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	-	-	Write as zero.
15	SoftSReq15	R/W	DMA single request by software (Request No. [15]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
14	SoftSReq14	R/W	DMA single request by software (Request No. [14]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
13	SoftSReq13	R/W	DMA single request by software (Request No. [13]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
12	SoftSReq12	R/W	DMA single request by software (Request No. [12]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
11	SoftSReq11	R/W	DMA single request by software (Request No. [11]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
10	SoftSReq10	R/W	DMA single request by software (Request No. [10]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
9	SoftSReq9	R/W	DMA single request by software (Request No. [9]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
8	SoftSReq8	R/W	DMA single request by software (Request No. [8]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested

Bit	Bit Symbol	Type	Function
7	SoftSReq7	R/W	DMA single request by software (Request No. [7]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
6	SoftSReq6	R/W	DMA single request by software (Request No. [6]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
5	SoftSReq5	R/W	DMA single request by software (Request No. [5]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
4	SoftSReq4	R/W	DMA single request by software (Request No. [4]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
3	SoftSReq3	R/W	DMA single request by software (Request No. [3]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
2	SoftSReq2	R/W	DMA single request by software (Request No. [2]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
1	SoftSReq1	R/W	DMA single request by software (Request No. [1]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested
0	SoftSReq0	R/W	DMA single request by software (Request No. [0]) Read : 0 :Stopping DMA single transfer 1 : running DMA single transfer Write: 0 : invaild 1 : DMA single requested

Note 1: Do not execute DMA requests by software and hardware at the same time.

Note 2: Refer to "11.4.2 DMA request" for DMA request number. Clear "0" to bit corresponded with the DMA request number which has no single request.

11.5.12 DMACxConfiguration (DMAC Configuration Register)

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

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	M	R/W	Write as zero.
0	E	R/W	DMA circuit control 0 : Stop 1 : Operate When circuit stops, the registers for the DMA circuit cannot be written or read. When operating the DMA, always set <E>="1".

11.5.13 DMACxCnSrcAddr (DMAC Channelx Source Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	SrcAddr[31:0]	R/W	Sets a DMA transfer source address. Make sure to confirm the source address and the bit width before setting. The below are the restrictions in setting of source address bit width.

Source address bit width DMACxCnControl<Swidth[2:0]>	Setting of least significant address
000 : Byte (8 bits)	no restriction
001 : Half word (16 bits)	Setting as multiples of 2, (0x0,0x02,0x4,0x06,0x8,0xA,0xC…)
010 : Word (32 bits)	Setting as multiples of 4, (0x0,0x4,0x8,0xC…)

Because enabling channel "n" (DMACxCnConfiguration<E>="1") updates the data written in the registers, set DMACxCnSrcAddr before enabling the channels.

When the DMA is operating, the value in the DMACxCnSrcAddr register sequentially changes, so the read values are not fixed.

And do not update DMACxCnSrcAddr during transfer. To change DMACxCnSrcAddr, be sure to disable the channel "n" (DMACxCnConfiguration<E>="0") before change.

11.5.14 DMACxCnDestAddr (DMAC Channelx Destination Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	DestAddr[31:0]	R/W	Sets a DMA transfer destination address. Make sure to confirm the destination address and the bit width before setting. The below are the restrictions in setting of destination address bit width.

Destination address bit width DMACxCControl<Dwidth[2:0]>	Setting of least significant address
000 : Byte (8 bits)	no restriction
001 : Half word (16 bits)	Setting as multiples of 2, (0x0,0x02,0x4,0x06,0x8,0xA,0xC…)
010 : Word (32 bits)	Setting as multiples of 4, (0x0,0x4,0x8,0xC…)

Do not update DMACxCnDestAddr during transfer. To change DMACxCnDestAddr, be sure to disable the channel "n" (DMACxCnConfiguration<E>="0") before change.

11.5.15 DMACxCnLLI (DMAC Channelx Linked List Item Register)

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

Bit	Bit Symbol	Type	Function
31-2	LLI[29:0]	R/W	Sets the first address of the next transfer information. Set a value smaller than 0xFFFF_FFF0. When <LLI> = 0, LLI is the last chain. After DMA transfer finishes, the DMA channel is disabled.
1-0	-	R/W	Write as zero.

Note: For <LLI> detailed operation, see "11.6 Special Functions".

11.5.16 DMACxCnControl (DMAC Channeln Control Register)

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

Bit	Bit Symbol	Type	Function
31	I	R/W	Register for enabling a transfer interrupt. 0 : Disable 1 : Enable The transfer end interrupt is generated by setting <I>="1" and DMACxCnConfiguration<ITC>="1". When the scatter/gather function is used in the last transfer DMAC setting flow and by setting this bit to enable, to generate the transfer end interrupt is enable only at the last transfer. To generate interrupt during normal transfer, set this bit to "1" and change to enable mode.
30-28	-	W	Write as zero.
27	DI	R/W	Increment the transfer destination address 0 : Do not increment 1 : Increment
26	SI	R/W	Increment the transfer source address 0 : Do not increment 1 : Increment
25-24	-	W	Write as zero.
23-21	Dwidth[2:0]	R/W	Transfer destination bit width. 000 : Byte (8 bits) 001 : Half-word (16 bits) 010 : Word (32 bits) other: Reserved Refer to Table 11-5 for the setting value.
20-18	Swidth[2:0]	R/W	Transfer source bit width 000: Byte (8 bits) 001: Half-word (16 bits) 010 : Word (32 bits) other: Reserved Refer to Table 11-5 for the setting value.
17-15	DBSize[2:0]	R/W	Transfer destination burst size: (Note 1) 000: 1 beat 100: 32 beats 001: 4 beats 101: 64 beats 010: 8 beats 110: 128 beats 011: 16 beats 111: 256 beats Refer to Table 11-5 for the setting value.

Bit	Bit Symbol	Type	Function
14-12	SBSize[2:0]	R/W	Transfer source burst size: (Note 1) 000: 1 beat 100: 32 beats 001: 4 beats 101: 64 beats 010: 8 beats 110: 128 beats 011: 16 beats 111: 256 beats Refer to Table 11-5 for the setting value.
11-0	TransferSize [11:0]	R/W	Set the total number of transfers. Amount of transfer data per a bit width specified by transfer source bit witdth (4 bytes / 2 bytes / 1byte) is set into <TransderSize[11:0]>. Because the burst size shows amount of transfer data at every DMA request internally, amount of transfer data is never changed even if any burst size is set when transfer source bit width and total number of transfers are not changed. The value of <TransferSize[11:0]> is decremented to 0 by the DMA transferring. If this is read, the value which is the number of data not to transfer. The total number of transfers is used as the unit for the transfer source bit width. For examples: When <Swidth>="000" (8bit), the number of transfers is expressed in the units of byte. When <Swidth>="001" (16bit), the number of transfers is expressed in the units of half word. When <Swidth>="010" (32bit), the number of transfers is expressed in the units of word.

Note: The burst size to be set with DBsize and SBsize has no connections with the HBURST for the AHB bus.

Table 11-5 How to decide the value of <Dwidth[2:0]>,<Swidth[2:0]>, <DBSize[2:0]>, <SBSize[2:0]>

<Dwidth[2:0]> / <Swidth[2:0]>	Set the number so that the following expression is satisfied: Transfer source bit width × Total number of transfers = Transfer destination bit width × N (N : Integer number)
	(ex.1) Bit width of transfer source:8 bit, bit width of transfer destination:32 bit, total number of transfers:25 times 8 bit × 25 times = 200 bit (25 byte) N = 200 ÷ 32 = 6.25 word Since 6.25 is not an integer number, the above setting is invalid. If the transfer source bit width is smaller than the transfer destination bit width, care must be taken when setting the total number of transfers.
	(ex.2) Bit width of transfer source :32 bit, bit width of transfer destination:16 bit, total number of transfers: 13 times 32 bit × 13 times = 416 bit (13 word) N = 416 ÷ 16 = 26 half_word Since 26 is an integer number, the above setting is valid.
<DBSize[2:0]> / <SBSize[2:0]>	When "Peripheral to Memory" or "Memory to Peripheral" is performed, peripheral circuits generates DMA request signal to indicate the preparation is ready. This signal triggers to execute data transfers. (In the case of "Memory to Memory", only software start is used.) Set the burst size to define the amount of data transferred from peripherals per DMA request signal. This register is used with FIFO buffer that can be contained multiple data.

11.5.17 DMACxCnConfiguration (DMAC Channel n Configuration Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	Halt	Active	Lock
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ITC	IE	FlowCntrl			-	DestPeripheral	
After reset	0	0	0	0	0	Undefined	0	0
	7	6	5	4	3	2	1	0
bit symbol	DestPeripheral		-	SrcPeripheral				E
After reset	0	0	Undefined	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-19	-	W	Write as zero.
18	Halt	R/W	Controls accepting a DMA request 0 : Accept a DMA request 1 : Ignore a DMA request
17	Active	R	Indicates whether data is present in the channel FIFO. 0 : No data exists in the FIFO 1 : Data exists in the FIFO
16	Lock	R/W	Sets a locked transfer (Non-divided transfer). 0 : Disable locked transfer 1: Enable locked transfer (Note3) When locked transfer is enabled, as many burst transfers as specified are consecutively executed without re-leasing the bus.
15	ITC	R/W	Transfer end interrupt enable register. 0 : Disable interrupt 1 : Enable interrupt
14	IE	R/W	Error interrupt enable register 0 :Disable interrupt 1: Enable interrupt
13-11	FlowCntrl[2:0]	R/W	Sets transfer method <FlowCntrl[2:0]> setting value / Transfer method 000: Memory to Memory (Note) 001: Memory to Peripheral 010: Peripheral to Memory 011: Peripheral to Peripheral 100 to 111: Reserved
10	-	W	Write as zero.
9-6	DestPeripheral [3:0]	R/W	Sets transfer destination peripheral Refer to "11.4.2 DMA request". When a memory is the transfer destination, this setting is ignored.
5	-	W	Write as zero.
4-1	SrcPeripheral [3:0]	R/W	Sets transfer source peripheral Refer to "11.4.2 DMA request". When a memory is the transfer source, this setting is ignored.

Bit	Bit Symbol	Type	Function
0	E	R/W	Channel enable 0 : Disable 1 : Enable This bit can be used to enable/disable the channels. (This bit works as start bit when "Memory to Memory" is selected.) Amount of transfer data specified by DMACxCnControl <TransferSize> is completed, the corresponding <E> is cleared to "0" automatically. Disabling channels during transfer loses the data in the FIFO. Initialize all the channels before restart. To pause the transfer, stop the DMA request by using the <HALT>, and poll the data until the <Active> becomes "0" and then disable the channel with the <E> bit.

Note 1: When "Memory to Memory" is selected, hardware start for DMA startup is not supported. Write "1" <E> for starting transfer.

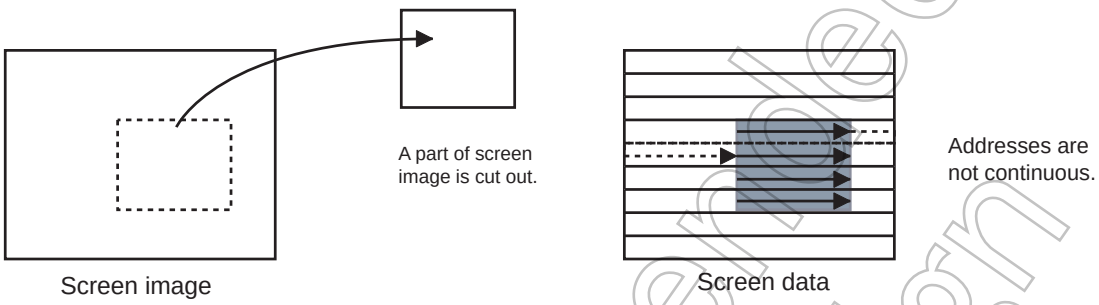
Note 2: When DMACxENableChns<EnabledCHx> is enabled and the corresponding DMACxCnConfiguration<Halt> is set to "1", write them after channel enable bit (E:bit0) is clear to "0". Without this, in the case of the slave error is occurred when writing them, the error is recovered by reset. Regarding slave error, when the width and address of transfer have mismatch, this error is occurred.

Note 3: When a lock transfer is enabled, satisfy the following conditions.
a) The bit widths of DMA transfer source and destination are same.
b) A burst size of DMA transfer source is four or more.

11.6 Special Functions

11.6.1 Scatter/gather function

When removing a part of image data and transferring it, image data cannot be handled as consecutive data, and the address changes dramatically depending on the special rule. Since DMA can transfer data only by using consecutive addresses, it is necessary to make required settings at locations where addresses changes.



The scatter/gather function can consecutively operate DMA settings (transfer source address, destination address, number of transfers, and transfer bus width) by re-loading them each time a specified number of DMA executions have completed via a pre-set "Linked List" where the CPU does not need to control the operation.

Setting "1" in the DMACxCnLLI register enables/disables the operation.

The items that can be set with Linked List are configured with the following 4 words:

1. DMACxCnSrcAddr
2. DMACxCnDestAddr
3. DMACxCnLLI
4. DMACxCnControl

They can be used with the interrupt operation.

An interrupt depends on the count end interrupt enable bit of the DMACxCnControl register, and can be generated at the end of each LLI. When this bit is used, a condition can be added even during transfer using LLI to perform branch operation, etc. To clear the interrupt, control the appropriate bit of the DMACxIntTCClear register.

11.6.2 Linked list operation

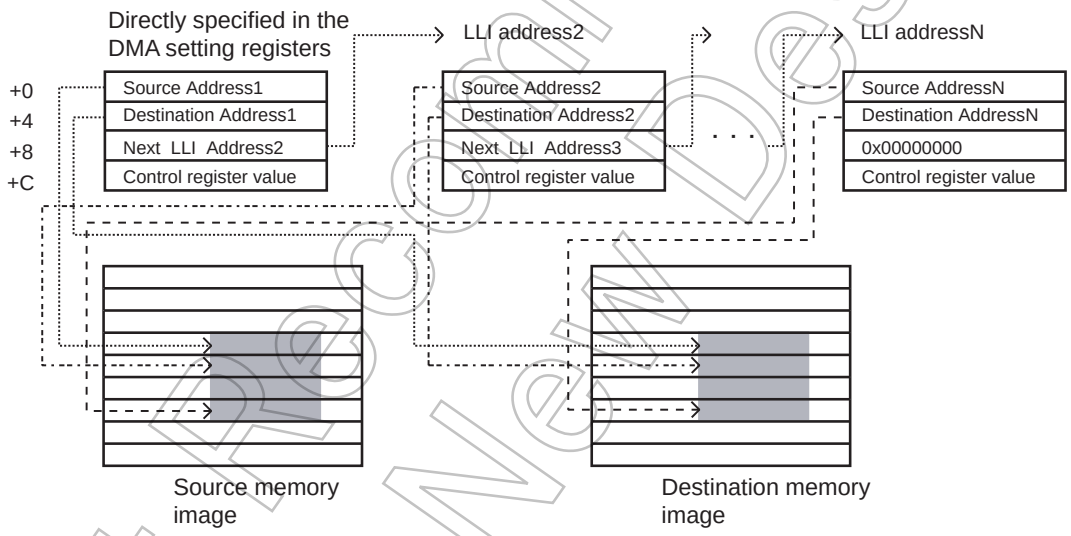
To operate the scatter/gather function, a transfer source and source data areas need to be defined by creating a set of Linked Lists first.

Each setting is called LLI (LinkedList).

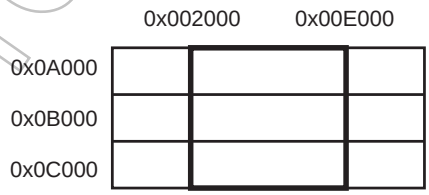
Each LLI controls the transfer of one block of data. Each LLI indicates normal DMA setting and controls transfer of successive data. Each time each DMA transfer is complete, the next LLI setting will be loaded to continue the DMA operation (Daisy Chain).

An example of the setting is shown below.

1. The first DMA transfer setting should be made directly in the DMA register.
2. The second and subsequent DMA transfer settings should be written in the addresses of the memory set in "next LLI AddressX."
3. To stop up to N'th DMA transfer, set "next LLI AddressX" to 0x0000_0000.

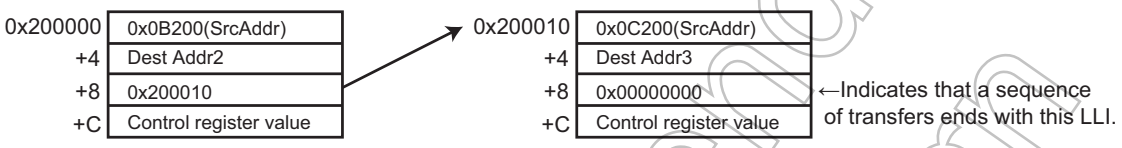


When transferring data in the area enclosed by the square



	Setting register	Setting parameter
+0	DMACxCnSrcAddr	:0x0A200
+4	DMACxCnDestAddr	:Destination address 1
+8	DMACxCnLLI	:0x200000
+C	DMACxCnControl	:Set the number of burst transfers and the number of transfers, etc.

Linked List



12. External bus interface (EBIF)

12.1 Overview

The TMPM366FDXBG/FYXBG/FWXBG has a built-in external bus interface function to connect to external memory, I/Os, etc. This interface consists of an external bus interface circuit (EBIF), a chip selector (CS) and a wait controller.

The chip selector and wait controller designate mapping addresses in a 2-block address space and also control wait states and data bus widths (8- or 16-bit) in these and other external address spaces.

The external bus interface circuit (EBIF) controls the timing of external buses based on the chip selector and wait controller settings.

Their features are given in the following.

Table 12-1 Features of External bus interface

features	
Memory supports	Asynchronous memory (NOR Flash memory, SRAM, Peripheral I/O and e.t.c.) Selectable separate bus mode or multiplex bus mode.
Data bus width	Either an 8- or 16-bit width can be set for each channel.
Chip select	2 channels ($\overline{CS0}$, $\overline{CS1}$)
Address access spaces	Supports up to 16MB memory spaces CS0: 0x6000_0000 to 0x61FF_FFFF (Max. 16MB) CS1: 0x6000_0000 to 0x61FF_FFFF (Max. 16MB)
Internal wait function	This function can be enabled for each channel. A wait of up to 15 cycles can be automatically inserted.
ALE wait function	This function can be enabled for each channel. An ALE high pulse of up to 4 cycles can be automatically inserted.
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	When an external bus is selected, a dummy cycle of up to 8 clocks can be inserted and this dummy cycle can be specified for each channel. (tCAR, tRAE cycles expanded)
Bus expansion function	The internal wait, the ALE wait, the Setup wait and the Recovery cycle can be expanded double or quadruple.
Control pins	Separate bus mode: D[15:0], A[19:0], $\overline{RD}$, $\overline{WR}$, $\overline{BELL}$, $\overline{BELH}$, $\overline{CS0}$, $\overline{CS1}$
	Multiplex bus mode: AD[15:0], A[23:16], $\overline{RD}$, $\overline{WR}$, $\overline{BELL}$, $\overline{BELH}$, $\overline{CS0}$, $\overline{CS1}$, ALE

12.2 Address and Data Pins

12.2.1 Address and Data pin setting

The TMPM366FDXBG/FYXBG/FWXBG can be set to either separate bus or multiplexed bus mode. Setting the bit <EXBSEL> of EXBMOD register to "1" as the separate bus mode, and setting to "0" as the multiplexed bus mode. Port pins A to E which are to be connected to external devices (memory), are used as address buses, data buses and address/data buses. Table 12-2 shows these.

Table 12-2 Bus Mode, Address and Data Pins

PORT	Separate EXBMOD<EXBSEL> = "1"	Multiplex EXBMOD<EXBSEL> = "0"
Port A (PA0 to PA7)	D0 to D7	AD0 to AD7
Port B (PB0 to PB7)	D8 to D15 A0 to A7	AD8 to AD15
Port C(PC0~PC2)	A0~A2	-
Port D(PD0~PD7)	A16~A19	A16~A19
Port E(PE0~PE7)	A11~A15	A20~A23
Port G(PG0~PG5)	A3~A7	-
Port H(PH0~PH4Åj	A8~A10	-

Each port is put into input mode after a reset. To access an external device, set the address and data bus functions by using the port control register (PxCR) and the port function register (PxFC), and set the input enable register (PxIE).

When the access changing from the external area to internal area, the address buses are kept the previously external area address output and the data buses will be high impedance.

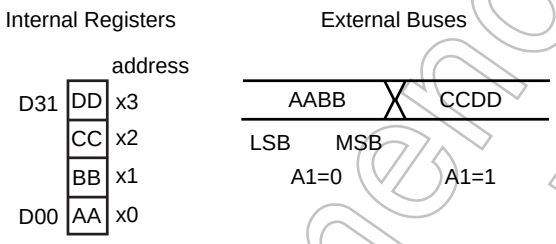
12.3 Data Format

Internal registers and external bus interfaces of the TMPM366FDXBG/FYXBG/FWXBG are configured as described below.

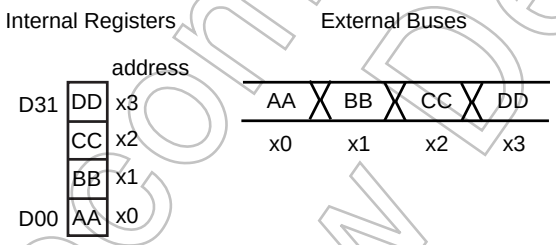
12.3.1 Little-endian mode

12.3.1.1 Word access

- 16-bit bus width

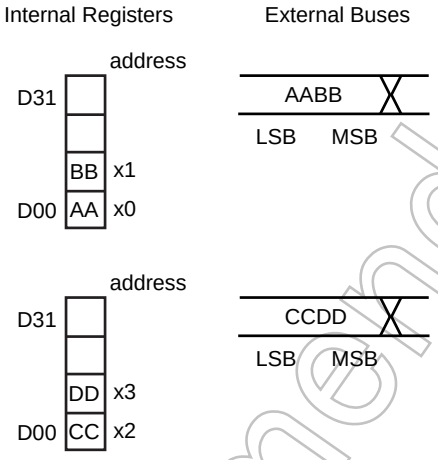


- 8-bit bus width

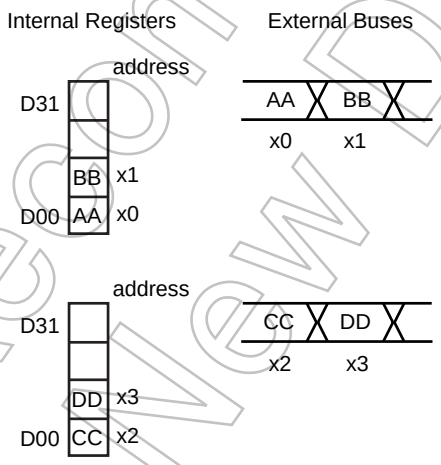


12.3.1.2 Half word access

- 16-bit bus width

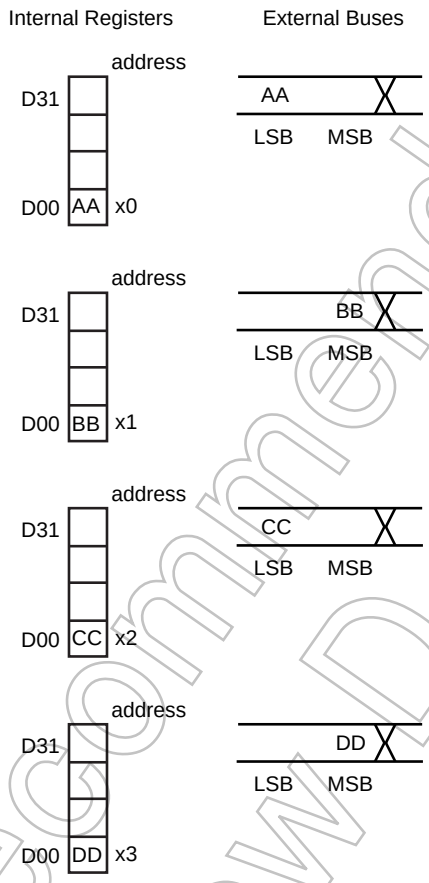


- 8-bit bus width

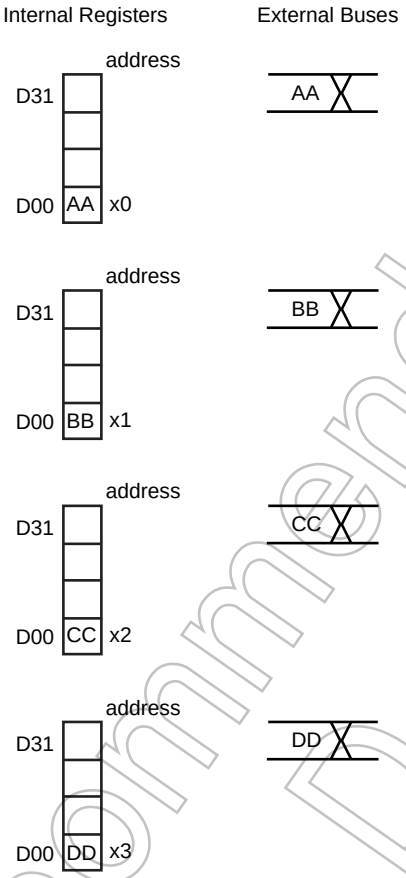


12.3.1.3 Byte access

- 16-bit bus width



- 8-bit bus width



12.4 External Bus Operations (Separete Bus Mode)

This section describes various bus timing values. The timing diagram shown below assumes that the address buses are A23 through A0 and that the data buses are D15 through D0.

12.4.1 Basic bus operation

The external bus cycle of the TMPM366FDXBG/FYXBG/FWXBG basically consists of three clock pulses. The basic clock of an external bus cycle is the same as the internal system clock.Figure 12-1 shows read bus timing and Figure 12-2 shows write bus timing. If internal areas are accessed, address buses remain unchanged as shown in these figures. Additionally, data buses are in a state of high impedance and control signals such as $\overline{RD}$ and $\overline{WR}$ do not become active.

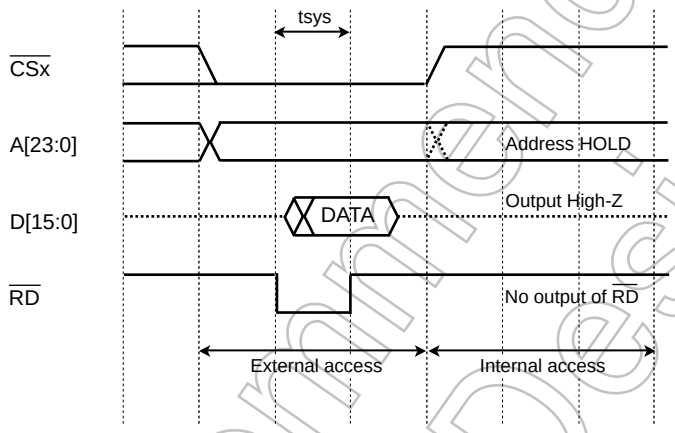


Figure 12-1 Read Operation Timing

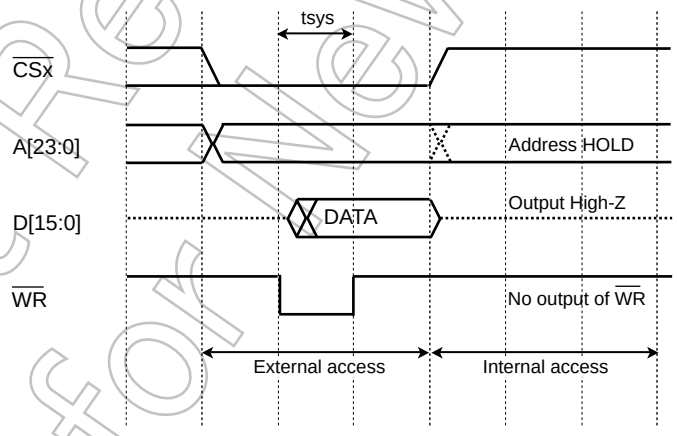


Figure 12-2 Write Operation Timing

12.4.2 Wait timing

A wait cycle can be inserted for each channel by using the chip selector (CS) and wait controller.

The following wait can be inserted.

- A wait of up to 15 clocks can be automatically inserted.

The setting of the number of waits to be automatically inserted and the setting can be made using the chip select control registers, EXBCSx<CSIW[4:0]>.

Figure 12-3 through Figure 12-4 show the timing diagrams in which waits have been inserted.

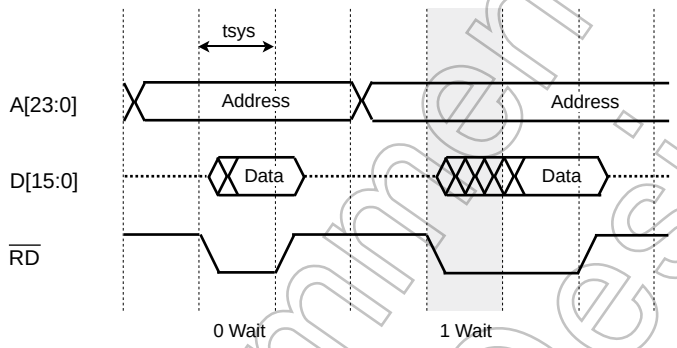


Figure 12-3 Read Operation Timing (0 Wait and 1 Wait Automatically Inserted)

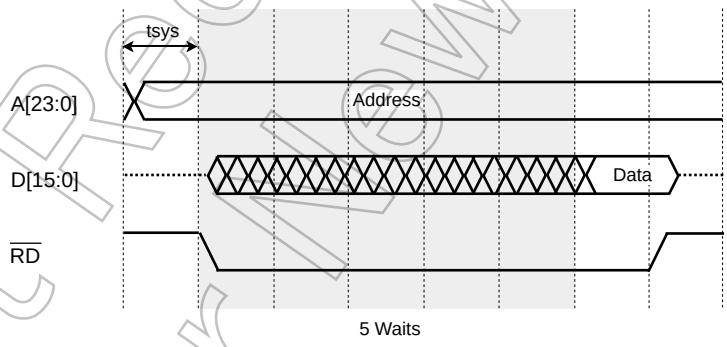


Figure 12-4 Read Operation Timing (5 Waits Automatically Inserted)

Figure 12-5 through Figure 12-6 shows the read and write operation timing when 0 wait and 2 waits automatically inserted in the separate bus mode.

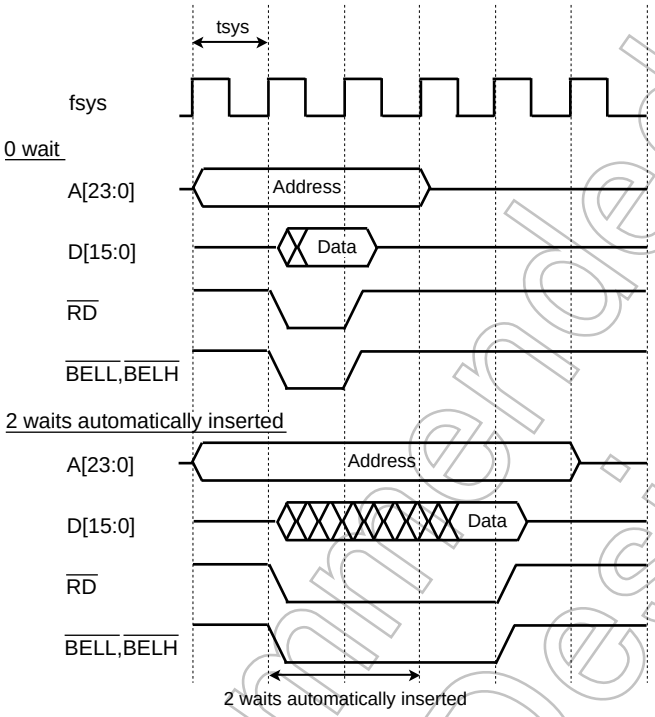


Figure 12-5 Read Operation Timing

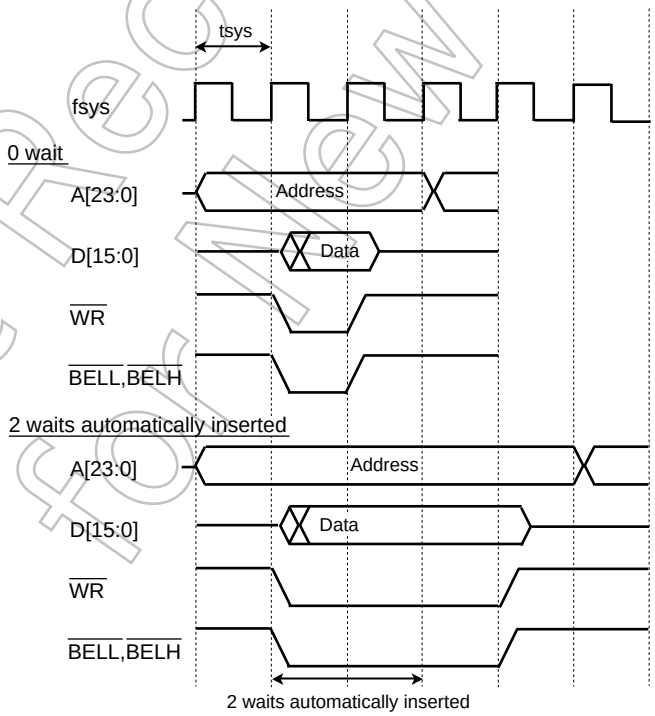


Figure 12-6 Write Operation Timing

12.4.3 Read and Write Recovery time

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

A dummy cycle can be inserted in both a read and a write cycle. The dummy cycle insertion setting can be made in the chip select control registers, EXBCSx<WRR[2:0]> (write recovery cycle) and <RDR[2:0]> (read recovery cycle). As for dummy cycle, none, one to six or eight system clocks (internal) can be specified for each channel. Figure 12-7 shows the timing of recovery time insertion.

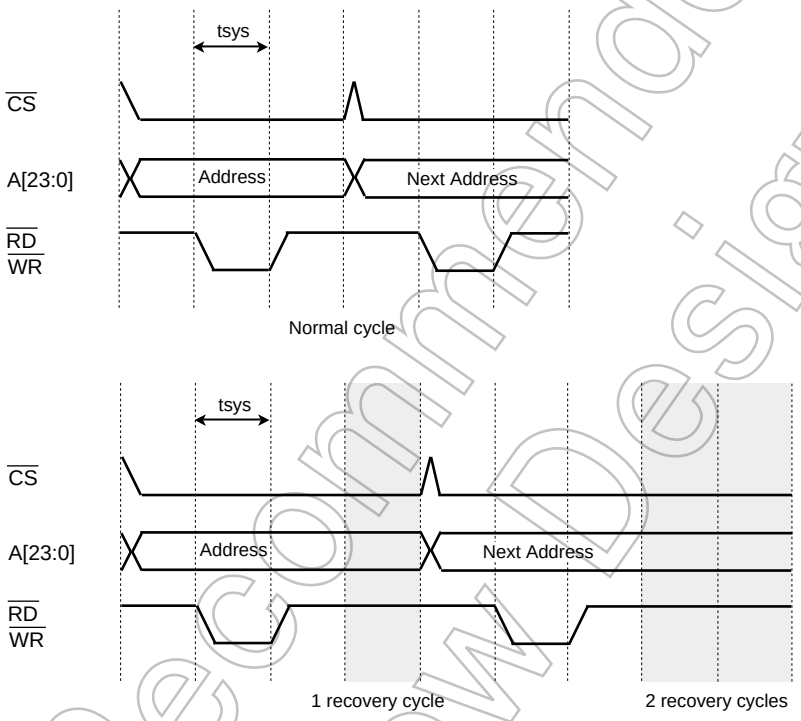


Figure 12-7 Timing of Recovery Time insertion

12.4.4 Chip select recovery time

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

The dummy cycle insertion setting can be made in the chip select control registers, EXBCSx<CSR[1:0]>. As for the number of dummy cycles, none, one, two and four system clocks (internal) can be specified for each channel. Figure 12-8 shows the timing of recovery time insertion.

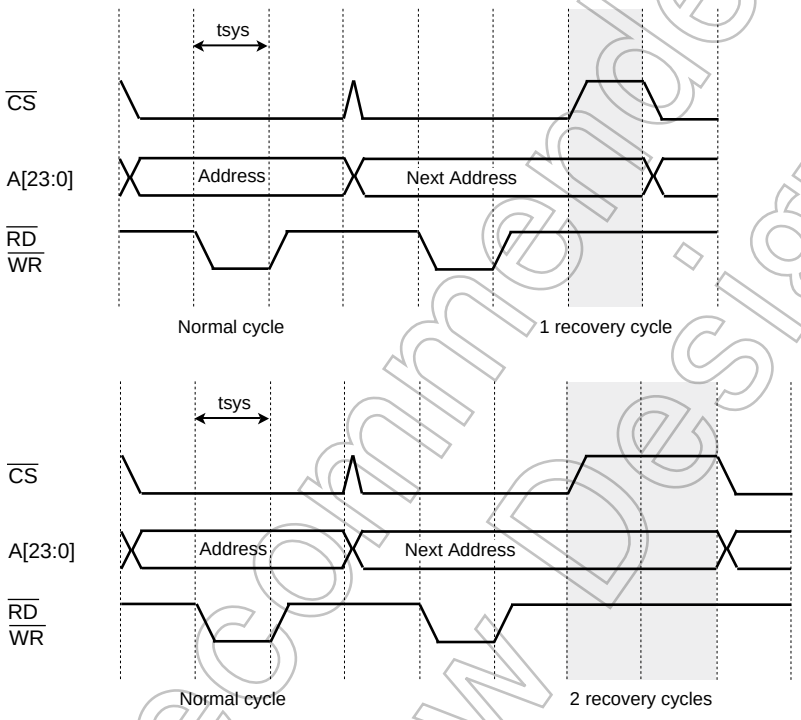


Figure 12-8 Timing of Chip Select Recovery Time Insertion

12.4.5 Read and Write setup cycle

A read and a write setup cycle can be inserted for each channel by using the chip selector (CS) and wait controller.

The following can be inserted.

- A read and a write setup cycle of up to 4 clocks can be automatically inserted.

The setting of the number of setup cycles to be automatically inserted and the setting can be made using the chip select control registers, EXBCSx<WRS[1:0]> and <RDS[1:0]>.

Figure 12-9 show the timing diagrams in which setup cycle have been inserted.

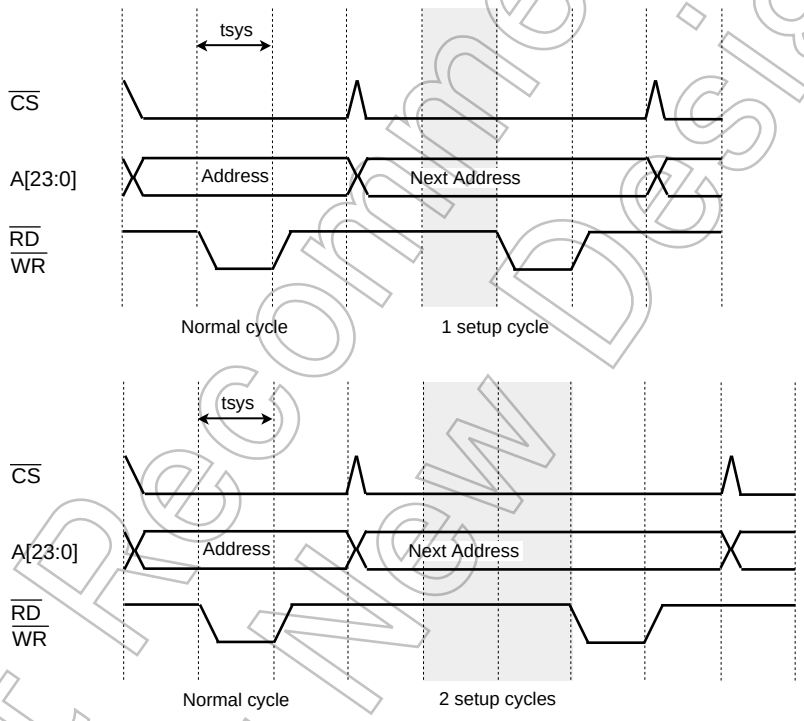


Figure 12-9 Timing of Read and Write Setup Time Insertion

12.5 External Bus Operations (Multiplexed Bus Mode)

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

12.5.1 Basic bus operation

The external bus cycle of the TMPM366FDXBG/FYXBG/FWXBG basically consists of four clock pulses. The basic clock of an external bus cycle is the same as the internal system clock.Figure 12-10 shows read bus timing and Figure 12-11 shows write bus timing. If internal areas are accessed, address buses remain unchanged and the ALE does not output latch pulse 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.

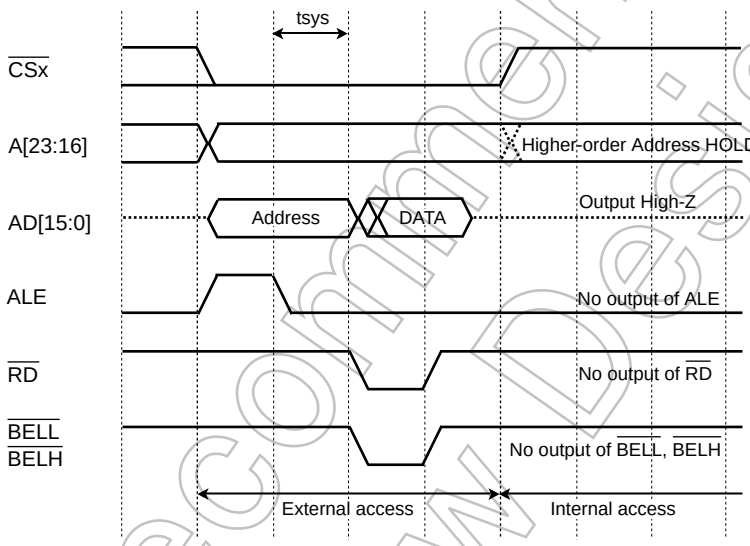


Figure 12-10 Read Operation Timing

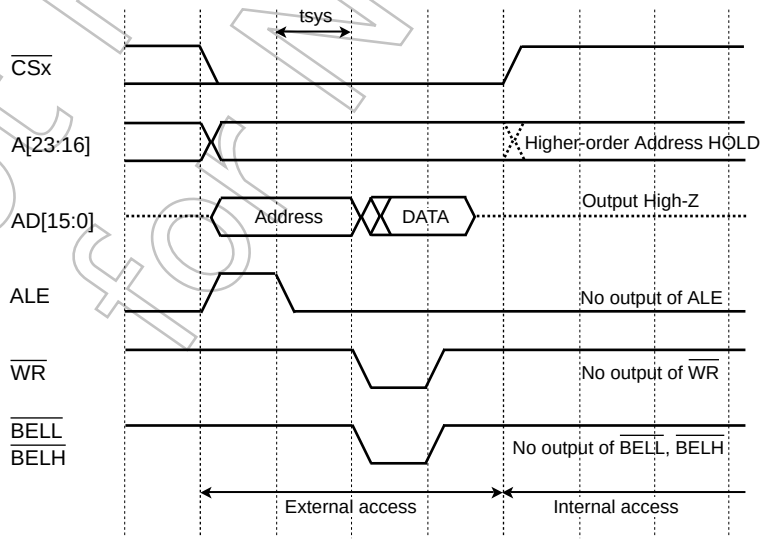


Figure 12-11 Write Operation Timing

12.5.2 Wait timing

A wait cycle can be inserted for each channel by using the chip selector (CS) and wait controller.

The following wait can be inserted.

- A wait of up to 15 clocks can be automatically inserted.

The setting of the number of waits to be automatically inserted and the setting can be made using the chip select control registers, EXBCSx<CSIW[4:0]>.

Figure 12-12 through Figure 12-13 show the timing diagrams in which waits have been inserted.

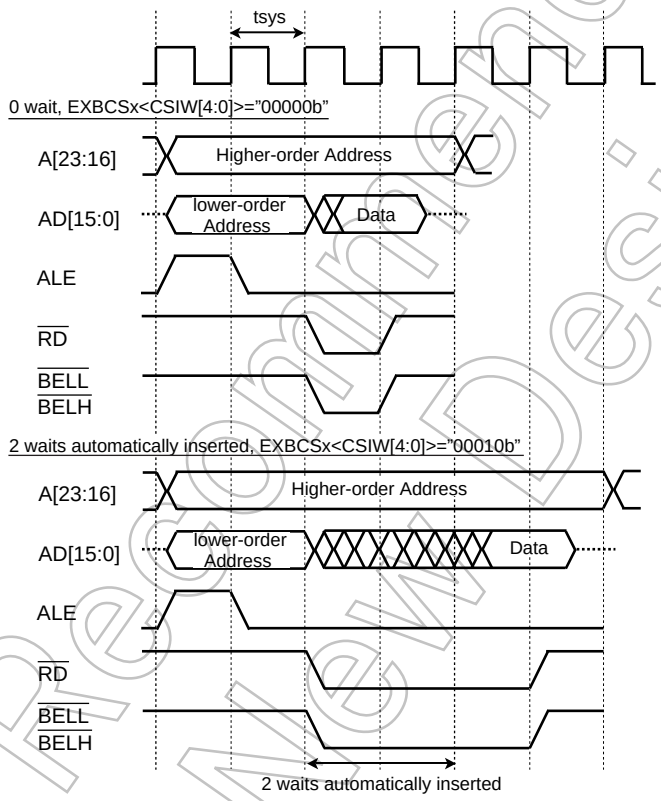


Figure 12-12 Read Operation Timing

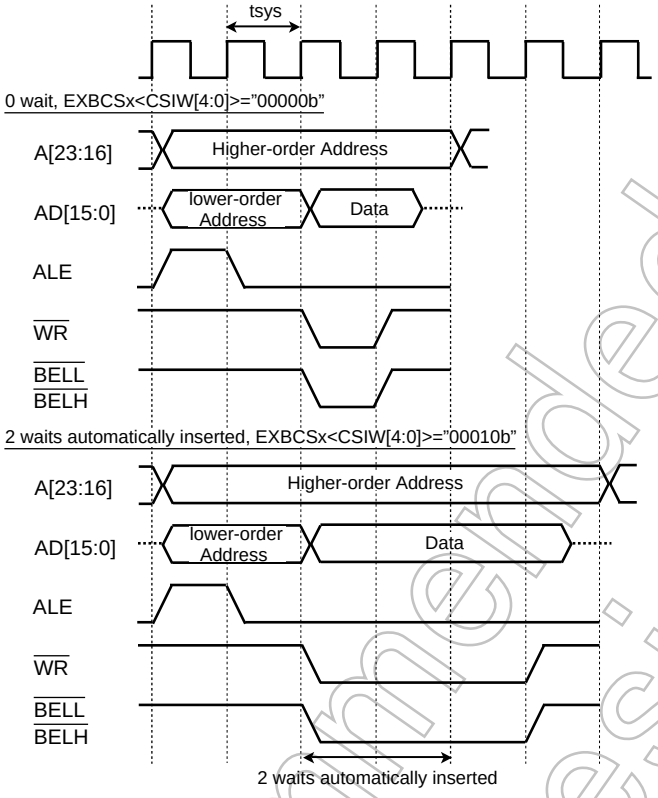


Figure 12-13 Write Operation Timing

12.5.3 Time that it takes before ALE is asserted

One of system clocks of 1, 2 or 4 can be selected as the time that it takes before ALE is asserted. The setting can be made using the chip select control registers, EXBCSx<ALEW[1:0]>. The default is asserted the $\overline{RD}$ or $\overline{WR}$ signal from the address is generated after 2 clocks.

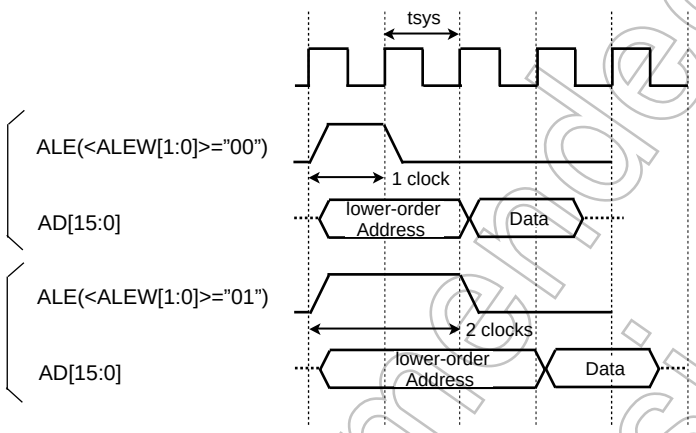


Figure 12-14 Time that it takes before ALE is asserted

Figure 12-15 shows the timing when the ALE is 1 clock or 2 clocks.

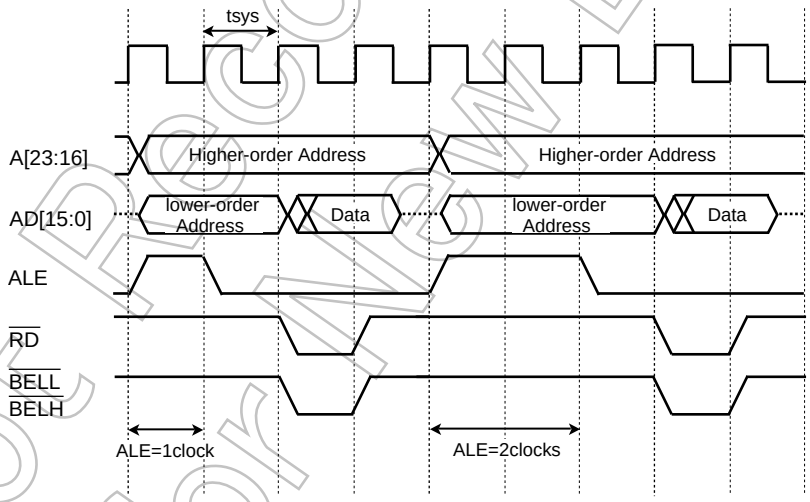


Figure 12-15 Read Operation Timing (When the ALE is 1 Clock or 2 Clocks)

12.5.4 Read and Write Recovery Time

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

A dummy cycle can be inserted in both a read and a write cycle. The dummy cycle insertion setting can be made in the chip select control registers, EXBCSx<WRR[2:0]> (write recovery cycle) and <RDR[2:0]> (read recovery cycle). As for the number of dummy cycles, none, one to six or eight system clocks (internal) can be specified for each channel. Figure 12-16 shows the timing of recovery time insertion.

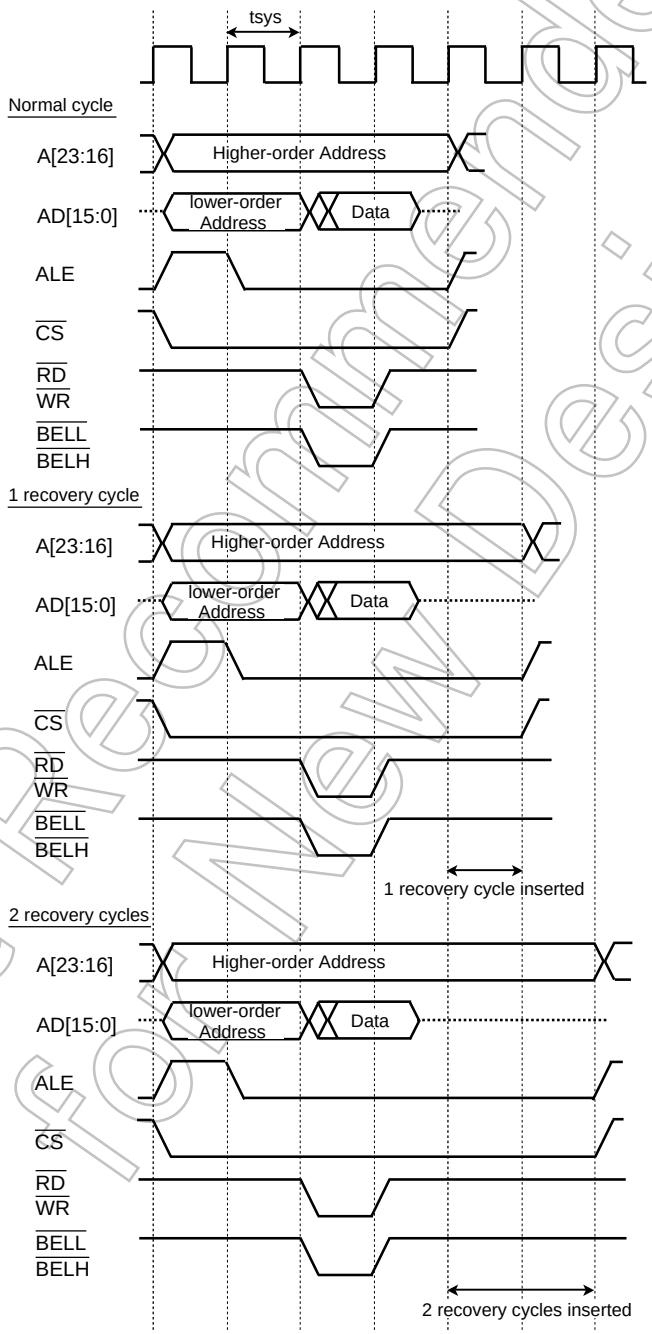


Figure 12-16 Timing of Recovery Time Insertion

12.5.5 Chip select recovery time

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

The dummy cycle insertion setting can be made in the chip select control registers, EXBCSx<CSR[1:0]>. As for the number of dummy cycles, none, one, two and four system clocks (internal) can be specified for each channel. Figure 12-17 shows the timing of recovery time insertion.

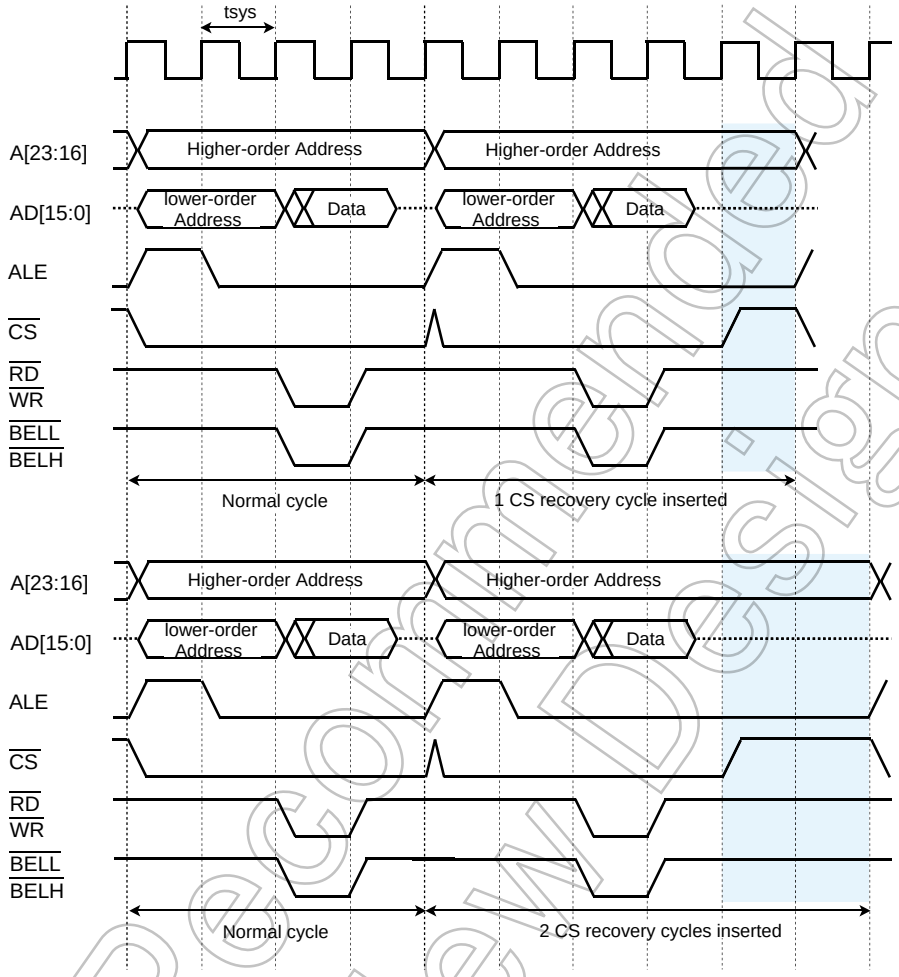


Figure 12-17 Timing of CS Recovery Time Insertion (ALE width: 1 clock)

12.5.6 Read and Write setup cycle

A read and a write setup cycle can be inserted for each channel by using the chip selector (CS) and wait controller.

The following can be inserted.

- A read and a write setup cycle of up to 4 clocks can be automatically inserted.

The setting of the number of setup cycles to be automatically inserted and the setting can be made using the chip select control registers, EXBCSx<WRS[1:0]> and <RDS[1:0]>.

Figure 12-18 shows the timing diagrams in which the read or write setup cycle have been inserted.

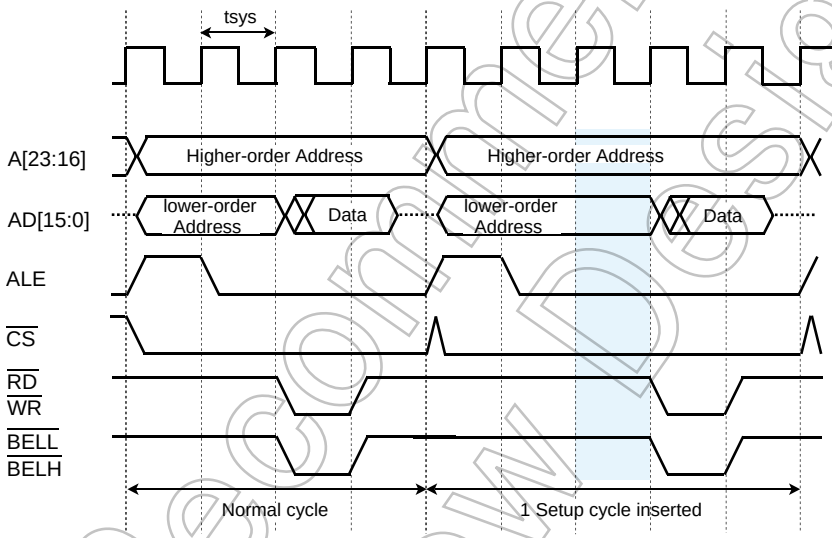


Figure 12-18 Timing of Read or Write Setup Time Insertion

12.6 Registers

12.6.1 Registers List

Address and names of EBIF control registers are shown below.

Base Address = 0x4005_C000

Register name		Address (Base+)
External Bus Mode Control Register	EXBMOD	0x0000
Reserved	-	0x0004 to 0x000C
External Bus Area and Start Address Configuration Register 0	EXBAS0	0x0010
External Bus Area and Start Address Configuration Register 1	EXBAS1	0x0014
Reserved	-	0x0018 to 0x003C
External Bus Chip Select Control Regiser 0	EXBCS0	0x0040
External Bus Chip Select Control Regiser 1	EXBCS1	0x0044
Reserved	-	0x0048 to 0x0FFC

Note 1: Access the registers by using word reads and word writes.
Note 2: Access to the "Reserved" area is prohibited.

12.6.2 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) 0: Multiplex bus 1: Separate bus

Note: Do not change the setting of external bus mode in operating the external bus access.

12.6.3 EXBASx (External Bus Area and Start Address 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	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
31-24	−	R/W	Always write to 0110_0000
23-16	SA23-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 CS0 and CS1, the chip selector will be given priority to CS0.

Note: If the access address is exceeded space in 0x6000_0000 to 0x61FF_FFFF, a hard fault error will be generated.

12.6.4 EXBCSx (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, 001: 1 cycle, 010: 2 cycles, 011: 3 cycles, 100: 4 cycles, 101: 5 cycles, 110: 6 cycles, 111: 8 cycles
26-24	RDR[2:0]	R/W	Read ($\overline{RD}$) Recovery cycle 000: None, 001: 1 cycle, 010: 2 cycles, 011: 3 cycles, 100: 4 cycles, 101: 5 cycles, 110: 6 cycles, 111: 8 cycles
23-22	−	R	Read as 0.
21-20	ALEW[1:0]	R/W	ALE wait cycle for multiplex bus 000: None, 001: 1 cycle, 010: 2 cycles, 011: 4 cycles
19-18	WRS[1:0]	R/W	Write ($\overline{WR}$) Setup cycle 000: None, 001: 1 cycle, 010: 2 cycles, 011: 4 cycles
17-16	RDS[1:0]	R/W	Read ($\overline{RD}$) Setup cycle 000: None, 001: 1 cycle, 010: 2 cycles, 011: 4 cycles
15-13	−	R	Read as 0.
12-8	CSIW[4:0]	R/W	Internal Wait (Automatically insertion) 0000: 0 wait, 0001: 1 wait, 0010: 2 waits, 0011: 3 waits, 0100: 4 waits, 0101: 5 waits, 0110: 6 waits, 0111: 7 waits, 1000: 8 waits, 1001: 9 waits, 1010: 10 waits, 1011: 11 waits, 1100: 12 waits, 1101: 13 waits, 1110: 14 waits, 1111: 15 waits
7-4	−	R	Read as 0.
3	−	R/W	Always write to "0"
2-1	CSW[2:1]	R/W	Data bus width 00: 8-bit, 01: 16-bit, Others: Prohibited
0	CSW0	R/W	CS Enable 0: Disable, 1: Enable

12.7 Connection example for external memory

12.7.1 Connection Example for external 16-bit SRAM and NOR-Flash (Sparate bus)

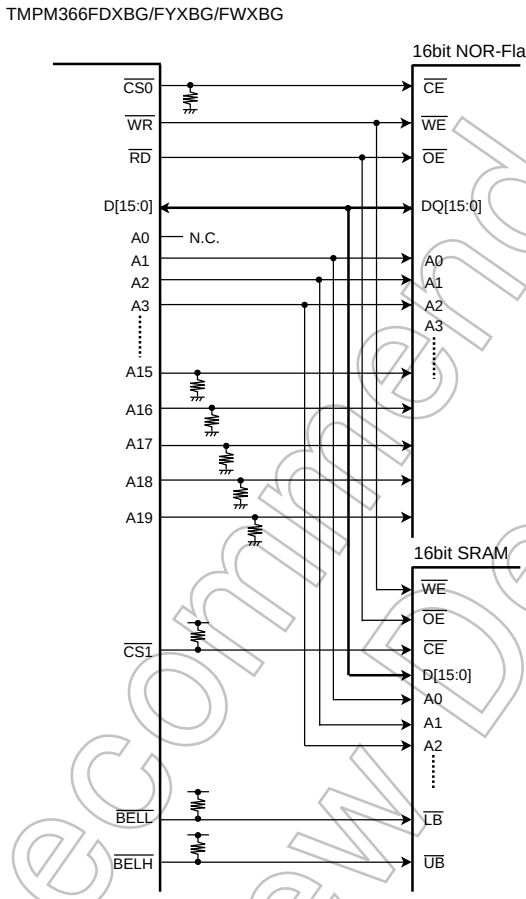


Figure 12-19 Connection Example for external 16-bit SRAM and NOR-Flash (Separate bus)

12.7.2 Connection Example for external 16-bit SRAM and NOR-Flash (Multiplex bus)

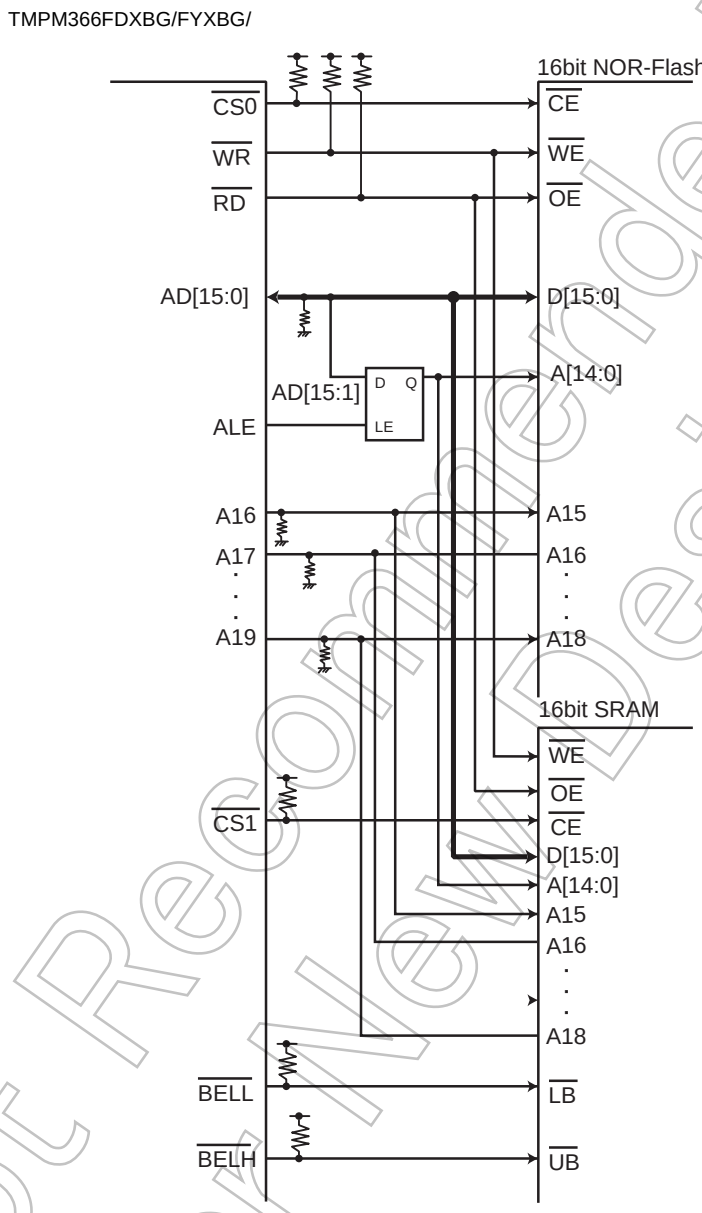


Figure 12-20 Connection Example for external 16-bit SRAM and NOR-Flash (Multiplex bus)

13. 16-bit Timer/Event Counters(TMRB)

13.1 Outline

TMRB operate in the following four operation modes:

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable pulse generation mode (PPG)
- Timer synchronous mode

The use of the capture function allows TMRB to perform the following three measurements.

- Frequency measurement
- Pulse width measurement
- Time difference measurement

In the following explanation of this section, "x" indicates a channel number.

13.2 Differences in the Specifications

TMPM366FDXBG/FYXBG/FWXBG contains 10-channel of TMRB.

Each channel functions independently and the channels operate in the same way except for the differences in their specification as shown in Table 13-1.

Some of the channels can put the capture trigger and the synchronous start trigger on other channels.

1. The flip-flop output of TMRB 7 through TMRB 9 can be used as the capture trigger of other channels.
   - TB7OUT → available for TMRB0 through TMRB1
   - TB8OUT → available for TMRB2 through TMRB3
   - TB9OUT → available for TMRB4 through TMRB6

2. The start trigger of the timer synchronous mode (with TBxRUN)
   - TMRB0 → can start TMRB0 through TMRB3 synchronously
   - TMRB4 → can start TMRB4 through TMRB7 synchronously

3. The start trigger of the timer prescaler synchronous mode (with TBxPRUN)
   - TMRB0 → can start TMRB0 through TMRB3 synchronously
   - TMRB4 → can start TMRB4 through TMRB7 synchronously

Table 13-1 Differences in the Specifications of TMRB Modules

Specification	External pins		Trigger function between timers		Interrupt		Internal connection		
Channel	Timer flip-flop output pin	External clock/capture trigger input pins	Capture trigger	Synchronous start trigger	Capture interrupt	TMRB interrupt	A/DC high priority conversion start	ADC normal conversion start	Timer flip-flop output TBxOUT to SIO/UART (TXTRG :transfer clock)
TMRB0	TB0OUT	TB0IN0 TB0IN1	TB7OUT	–	INTCAP00 INTCAP01	INTTB0	–	–	–
TMRB1	TB1OUT	TB1IN0 TB1IN1	TB7OUT	TB0PRUN,TB0RUN	INTCAP10 INTCAP11	INTTB1	–	–	–
TMRB2	TB2OUT	TB2IN0 TB2IN1	TB8OUT	TB0PRUN,TB0RUN	INTCAP20 INTCAP21	INTTB2	–	–	–
TMRB3	TB3OUT	TB3IN0 TB3IN1	TB8OUT	TB0PRUN,TB0RUN	INTCAP30 INTCAP31	INTTB3	–	–	–
TMRB4	TB4OUT	TB4IN0 TB4IN1	TB9OUT	–	INTCAP40 INTCAP41	INTTB4	INTCAP40	–	–
TMRB5	TB5OUT	TB5IN0 TB5IN1	TB9OUT	TB4PRUN,TB4RUN	INTCAP50 INTCAP51	INTTB5	–	INTCAP50	–
TMRB6	TB6OUT	TB6IN0 TB6IN1	TB9OUT	TB4PRUN,TB4RUN	INTCAP60 INTCAP61	INTTB6	–	–	–
TMRB7	TB7OUT	TB7IN0 TB7IN1	–	TB4PRUN,TB4RUN	INTCAP70 INTCAP71	INTTB7	–	–	–
TMRB8	TB8OUT	TB8IN0 TB8IN1	–	–	INTCAP80 INTCAP81	INTTB8	–	–	SIO0, SIO1
TMRB9	TB9OUT	TB9IN0 TB9IN1	–	–	INTCAP90 INTCAP91	INTTB9	–	–	

13.3 Configuration

Each channel consists of a 16-bit up-counter, two 16-bit timer registers (double-buffered), two 16-bit capture registers, two comparators, a capture input control, a timer flip-flop and its associated control circuit. Timer operation modes and the timer flip-flop are controlled by a register.

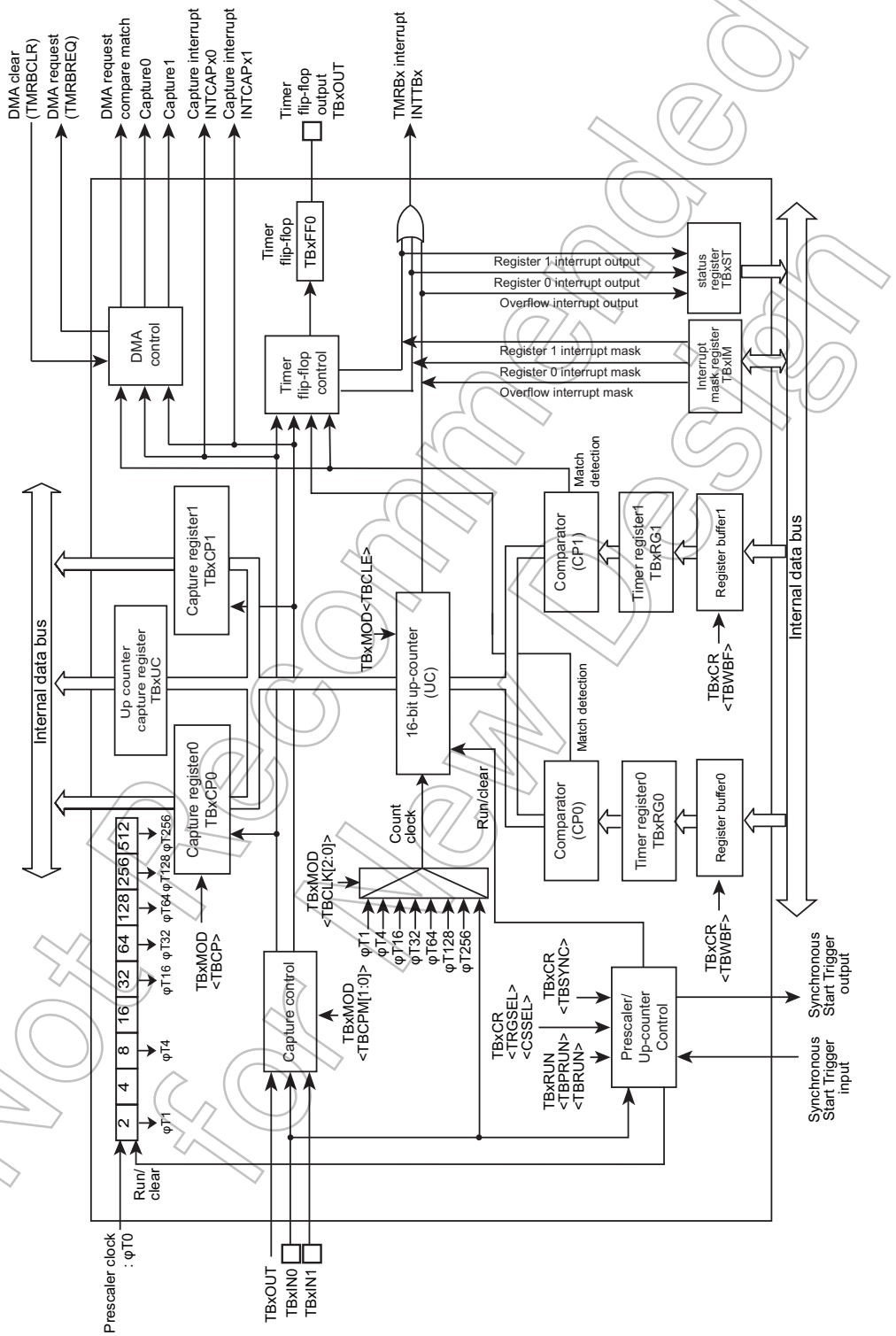


Figure 13-1 TMRBx Block Diagram(x= 0 to 9)

13.4 Registers

13.4.1 Register list according to channel

The following table shows the register names and addresses of each channel.

Channel x	Base Address
Channel0	0x400C_4000
Channel1	0x400C_4100
Channel2	0x400C_4200
Channel3	0x400C_4300
Channel4	0x400C_4400
Channel5	0x400C_4500
Channel6	0x400C_4600
Channel7	0x400C_4700
Channel8	0x400C_4800
Channel9	0x400C_4900

Register name(x=0~9)		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
DMA request enable register	TBxDMA	0x0030

Note: Timer control register, timer mode register and timer flip-flop control register can not be changed during timer operation. Make any changes of the above regsters after timer stopped.

13.4.2 TBxEN (Enable register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBEN	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 TMRB operation. When the operation is disabled, no clock is supplied to the other registers in the TMRB module. This can reduce power consumption. (This disables reading from and writing to the other registers except TBxEN register.) To use the TMRB, enable the TMRB operation (set to "1") before programming each register in the TMRB module. If the TMRB operation is executed and then disabled, the settings will be maintained in each register.
6	TBHALT	R/W	Clock operation during debug HALT 0: run 1: stop Specifies the TMRB clock setting to run or stop when the debug tool transits to HALT mode while in use.
5-0	−	R	Read as 0.

13.4.3 TBxRUN(RUN register)

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

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as 0.
2	TBPRUN	R/W	Prescaler operation 0: Stop & clear 1: Count
1	−	R	Read as 0.
0	TBRUN	R/W	Count operation 0: Stop & clear 1: Count

Note: When the counter is stopped (<TBRUN>="0") and TBxUC<TBUC[15:0]> is read, the value which was captured when the counter was operated is read.

13.4.4 TBxCR (Control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBWBF	-	TBSYNC	-	I2TB	TBINSEL	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 (unit of channel) 1: synchronous
4	−	R	Read as 0.
3	I2TB	R/W	Operation at IDLE mode 0: Stop 1:Operation
2	TBINSEL	R/W	Selects the external inputs. 0: TBxIN0/1
1	TRGSEL	R/W	Selects the external triggers. 0: rising 1: falling Controls the edge selection (of signal to TBxIN0 pin) when the external triggers is selected.
0	CSSEL	R/W	Selects the count start 0: starts by software 1: starts by external trigger

13.4.5 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	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	TBCP	TBCPM		TBCLE	TBCLK		
After reset	0	1	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
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) takes count value. Read as 1.
5-4	TBCPM[1:0]	R/W	Capture timing 00: Disable Capture timing 01: TBxIN0↑, TBxIN1↑ Takes count values into capture register 0 (TBxCP0) upon rising of TBxIN0 pin input. Takes count values into capture register 1 (TBxCP1) upon rising of TBxIN1 pin input. 10: TBxIN0↑, TBxIN0↓ Takes count values into capture register 0 (TBxCP0) upon rising of TBxIN0 pin input. Takes count values into capture register 1 (TBxCP1) upon falling of TBxIN0 pin input. 11: TBxOUT↑, TBxOUT↓ Takes count values into capture register 0 (TBnCP0) upon rising of 16-bit timer match output (TBxOUT) and into capture register 1 (TBnCP1) upon falling of TBxOUT. (x = 7, n = 0,1), (x = 8, n = 2,3), (x = 9, n = 4,5,6) (TMRB0 to 1: TB7OUT, TMRB2 to 3: TB8OUT, TMRB4 to 6: TB9OUTÅj
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 there is a match with Timer Regsiter1 (TBxRG1).
2-0	TBCLK[2:0]	R/W	Selects the TMRBx source clock. 000: TBxIN0 pin input 001: φT1 010: φT4 011: φT16 100: φT32 101: φT64 110: φT128 111: φT256

Note: When the channel of TBxMOD register is 7, 8 or 9, the setting of <TBCPM[1:0]>="11" is prohibited.

Note: Do not make any changes of TBxMOD register while corresponding TMRBx is running.

13.4.6 TBxFFCR (Flip-flop control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	TBC1T1	TBC0T1	TBE1T1	TBE0T1	TBFF0C	
After reset	1	1	0	0	0	0	1	1

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7-6	−	R	Read as 1.
5	TBC1T1	R/W	TBxFF0 reverse trigger when the up-counter value is taken into the TBxCP1. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when the up-counter value is taken into the Capture register 1 (TBxCP1).
4	TBC0T1	R/W	TBxFF0 reverse trigger when the up-counter value is taken into the TBxCP0. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when the up-counter value is taken into the Capture register 0 (TBxCP0).
3	TBE1T1	R/W	TBxFF0 reverse trigger when the up-counter value is matched with TBxRG1. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when the up-counter value is matched with the Timer register 1 (TBxRG1).
2	TBE0T1	R/W	TBxFF0 reverse trigger when the up-counter value is matched with TBxRG0. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when an up-counter value is matched with the Timer register 0 (TBxRG0).
1-0	TBFF0C[1:0]	R/W	TBxFF0 control 00: Invert Reverses the value of TBxFF0 (reverse by using software). 01: Set Sets TBxFF0 to "1". 10: Clear Clears TBxFF0 to "0". 11: Don't care * This is always read as "11".

13.4.7 TBxST (Status register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	INTTBOF	INTTB1	INTTB0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as 0.
2	INTTBOF	R	Overflow flag 0:No overflow occurs 1:Overflow occurs When an up-counter is overflow, "1" is set.
1	INTTB1	R	Match flag (TBxRG1) 0:No detection of a mach 1:Detects a match with TBxRG1 When a match with the timer register 1 (TBxRG1) is detected,"1" is set.
0	INTTB0	R	Match flag (TBxRG0) 0:No match is detected 1:Detects a match with TBxRG0 When a match with the timer register 0 (TBxRG0) is detected, "1" is set.

Note 1: To clear the flag, TBxST register should be read.

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: Even if mask configuration by TBxIM register is valid, the status is set to TBxST register.

13.4.8 TBxIM (Interrupt mask register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	TBIMOF	TBIM1	TBIM0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as 0.
2	TBIMOF	R/W	Overflow interrupt mask 0:Disable 1:Enable Sets the up-counter overflow interrupt to disable or enable.
1	TBIM1	R/W	Match interrupt mask (TBxRG1) 0:Disable 1:Enable Sets the match interrupt mask with the Timer register 1 (TBxRG1) to enable or disable.
0	TBIM0	R/W	Match interrupt mask (TBxRG0) 0:Disable 1:Enable Sets the match interrupt mask with the Timer register 0 (TBxRG0) to enable or disable.

Note: Even if mask configuration by TBxIM register is valid, the status is set to TBxST register.

13.4.9 TBxUC (Up counter capture register)

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

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-0	TBUC[15:0]	R	Captures a value by reading up-counter out. If TBxUC is read, current up-counter value can be captured.

Note: When the counter is operated and TBxUC is read, the value of the up counter is captured and read.

13.4.10 TBxRG0 (Timer register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBRG0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBRG0							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-0	TBRG0[15:0]	R/W	Sets a value comparing to the up-counter.

13.4.11 TBxRG1 (Timer register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBRG1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBRG1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-0	TBRG1[15:0]	R/W	Sets a value comparing to the up-counter.

13.4.12 TBxCP0 (Capture register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBCP0							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	TBCP0							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-0	TBCP0[15:0]	R	A value captured from the up-counter is read.

13.4.13 TBxCP1 (Capture register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBCP1							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	TBCP1							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-0	TBCP1[15:0]	R	A value captured from the up-counter is read.

13.4.14 TBxDMA(DMA request 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	-	-	-	-	-	TBDMAEN2	TBDMAEN1	TBDMAEN0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as 0.
2	TBDMAEN2	R/W	Selects DMA request:compare match. 0: Disabled 1 :Enabled
1	TBDMAEN1	R/W	Selects DMA request:input capture 1 0: Disabled 1: Enabled
0	TBDMAEN0	R/W	Selects DMA request:input capture 0 0: Disabled 1: Enabled

Note: When mask configuration by TBxIM register is valid, DMA request does not issue even if it is enabled.

Note: The assignment of DMA request factor differs by channel, TMRB0 to 9 . Refer to Chapter "DMAC " for details.

13.5 Description of Operations for Each Circuit

The channels operate in the same way, except for the differences in their specifications as shown in Table 13-2 .

13.5.1 Prescaler

There is a 4-bit prescaler to generate the source clock for up-counter UC.

The prescaler input clock φT0 is fperiph/1, fperiph/2, fperiph/4, fperiph/8, fperiph/16 or fperiph/32 selected by CGSYSCR<PRCK[2:0]> in the CG.The peripheral clock, fperiph, is either fgear, a clock selected by CGSYSCR<FPSEL> in the CG, or fc, which is a clock before it is divided by the clock gear.

The operation or the stoppage of a prescaler is set with TBxRUN<TBPRUN> where writing "1" starts counting and writing "0" clears and stops counting. Table 13-2 and Table 13-3 show prescaler output clock resolutions.

Table 13-2 Prescaler Output Clock Resolutions(fc = 48MHz)

Select peripheral clock CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function		
			φT1	φT4	φT16
0 (fgear)	000 (fc)	000 (fperiph/1)	fc/2^1 (0.04 μs)	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)
		001 (fperiph/2)	fc/2^2 (0.08 μs)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)
		010 (fperiph/4)	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)
		011 (fperiph/8)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		100 (fperiph/16)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		101 (fperiph/32)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)
	100 (fc/2)	000 (fperiph/1)	fc/2^2 (0.08 μs)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)
		001 (fperiph/2)	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)
		010 (fperiph/4)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		011 (fperiph/8)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		100 (fperiph/16)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)
		101 (fperiph/32)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)	fc/2^{11} (42.67 μs)
	101 (fc/4)	000 (fperiph/1)	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)
		001 (fperiph/2)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		010 (fperiph/4)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		011 (fperiph/8)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)
		100 (fperiph/16)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)	fc/2^{11} (42.67 μs)
		101 (fperiph/32)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)	fc/2^{12} (85.33 μs)
	110 (fc/8)	000 (fperiph/1)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		001 (fperiph/2)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		010 (fperiph/4)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)
		011 (fperiph/8)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)	fc/2^{11} (42.67 μs)
		100 (fperiph/16)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)	fc/2^{12} (85.33 μs)
		101 (fperiph/32)	fc/2^9 (10.67 μs)	fc/2^{11} (42.67 μs)	fc/2^{13} (170.67 μs)
	111 (fc/16)	000 (fperiph/1)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		001 (fperiph/2)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)
		010 (fperiph/4)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)	fc/2^{11} (42.67 μs)
		011 (fperiph/8)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)	fc/2^{12} (85.33 μs)
		100 (fperiph/16)	fc/2^9 (10.67 μs)	fc/2^{11} (42.67 μs)	fc/2^{13} (170.67 μs)
		101 (fperiph/32)	fc/2^{10} (21.34 μs)	fc/2^{12} (85.34 μs)	fc/2^{14} (341.34 μs)

Table 13-2 Prescaler Output Clock Resolutions(fc = 48MHz)

Select peripheral clock CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function		
			φT1	φT4	φT16
1 (fc)	000 (fc)	000 (fperiph/1)	fc/2^1 (0.04 μs)	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)
		001 (fperiph/2)	fc/2^2 (0.08 μs)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)
		010 (fperiph/4)	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)
		011 (fperiph/8)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		100 (fperiph/16)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		101 (fperiph/32)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)
	100 (fc/2)	000 (fperiph/1)	−	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)
		001 (fperiph/2)	fc/2^2 (0.08 μs)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)
		010 (fperiph/4)	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)
		011 (fperiph/8)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		100 (fperiph/16)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		101 (fperiph/32)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)
	101 (fc/4)	000 (fperiph/1)	−	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)
		001 (fperiph/2)	−	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)
		010 (fperiph/4)	fc/2^3 (0.17 μs)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)
		011 (fperiph/8)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		100 (fperiph/16)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		101 (fperiph/32)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)
	110 (fc/8)	000 (fperiph/1)	−	−	fc/2^5 (0.67 μs)
		001 (fperiph/2)	−	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)
		010 (fperiph/4)	−	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)
		011 (fperiph/8)	fc/2^4 (0.33 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		100 (fperiph/16)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		101 (fperiph/32)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)
	111 (fc/16)	000 (fperiph/1)	−	−	fc/2^5 (0.67 μs)
		001 (fperiph/2)	−	−	fc/2^6 (1.33 μs)
		010 (fperiph/4)	−	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)
		011 (fperiph/8)	−	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		100 (fperiph/16)	fc/2^5 (0.67 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.67 μs)
		101 (fperiph/32)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.33 μs)

Note 1: The prescaler output clock φTn must be selected so that φTn < fsys is satisfied (so that φTn is slower than fsys).
Note 2: Do not change the clock gear while the timer is operating.
Note 3: "−" denotes a setting prohibited.

Table 13-3 Prescaler Output Clock Resolutions(fc = 48MHz)

Select peripheral clock CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function			
			φT32	φT64	φT128	φT256
0 (fgear)	000 (fc)	000 (fperiph/1)	fc/2^6 (1.33 μs)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)
		001 (fperiph/2)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)
		010 (fperiph/4)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)
		011 (fperiph/8)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)
		100 (fperiph/16)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		101 (fperiph/32)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)
	100 (fc/2)	000 (fperiph/1)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)
		001 (fperiph/2)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)
		010 (fperiph/4)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)
		011 (fperiph/8)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		100 (fperiph/16)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)
		101 (fperiph/32)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)	fc/2^{15} (682.67 μs)
	101 (fc/4)	000 (fperiph/1)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)
		001 (fperiph/2)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)
		010 (fperiph/4)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		011 (fperiph/8)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)
		100 (fperiph/16)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)	fc/2^{15} (682.67 μs)
		101 (fperiph/32)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)	fc/2^{15} (682.67 μs)	fc/2^{16} (1365.33 μs)
	110 (fc/8)	000 (fperiph/1)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)
		001 (fperiph/2)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		010 (fperiph/4)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)
		011 (fperiph/8)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)	fc/2^{15} (682.67 μs)
		100 (fperiph/16)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)	fc/2^{15} (682.67 μs)	fc/2^{16} (1365.33 μs)
		101 (fperiph/32)	fc/2^{14} (341.33 μs)	fc/2^{15} (682.67 μs)	fc/2^{16} (1365.33 μs)	fc/2^{17} (2730.67 μs)
	111 (fc/16)	000 (fperiph/1)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		001 (fperiph/2)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)
		010 (fperiph/4)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)	fc/2^{15} (682.67 μs)
		011 (fperiph/8)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)	fc/2^{15} (682.67 μs)	fc/2^{16} (1365.33 μs)
		100 (fperiph/16)	fc/2^{14} (341.33 μs)	fc/2^{15} (682.67 μs)	fc/2^{16} (1365.33 μs)	fc/2^{17} (2730.67 μs)
		101 (fperiph/32)	fc/2^{14} (682.66μs)	fc/2^{15} (1365.33 μs)	fc/2^{16} (2730.67 μs)	fc/2^{17} (5461.34μs)

Table 13-3 Prescaler Output Clock Resolutions(fc = 48MHz)

Select peripheral clock CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function			
			φT32	φT64	φT128	φT256
1 (fc)	000 (fc)	000 (fperiph/1)	fc/2^6 (1.33 μs)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)
		001 (fperiph/2)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)
		010 (fperiph/4)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)
		011 (fperiph/8)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)
		100 (fperiph/16)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		101 (fperiph/32)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)
	100 (fc/2)	000 (fperiph/1)	fc/2^6 (1.33 μs)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)
		001 (fperiph/2)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)
		010 (fperiph/4)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)
		011 (fperiph/8)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)
		100 (fperiph/16)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		101 (fperiph/32)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)
	101 (fc/4)	000 (fperiph/1)	fc/2^6 (1.33 μs)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)
		001 (fperiph/2)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)
		010 (fperiph/4)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)
		011 (fperiph/8)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)
		100 (fperiph/16)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		101 (fperiph/32)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)
	110 (fc/8)	000 (fperiph/1)	fc/2^6 (1.33 μs)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)
		001 (fperiph/2)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)
		010 (fperiph/4)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)
		011 (fperiph/8)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)
		100 (fperiph/16)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		101 (fperiph/32)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)
	111 (fc/16)	000 (fperiph/1)	fc/2^6 (1.33 μs)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)
		001 (fperiph/2)	fc/2^7 (2.67 μs)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)
		010 (fperiph/4)	fc/2^8 (5.33 μs)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)
		011 (fperiph/8)	fc/2^9 (10.67 μs)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)
		100 (fperiph/16)	fc/2^{10} (21.33 μs)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)
		101 (fperiph/32)	fc/2^{11} (42.67 μs)	fc/2^{12} (85.33 μs)	fc/2^{13} (170.67 μs)	fc/2^{14} (341.33 μs)

Note 1: The prescaler output clock φTn must be selected so that φTn < fsys is satisfied (so that φTn is slower than fsys).
Note 2: Do not change the clock gear while the timer is operating.
Note 3: "−" denotes a setting prohibited.

13.5.2 Up-counter (UC)

UC is a 16-bit binary counter.

- Source clock

  UC source clock, specified by TBxMOD<TBCLK[2:0]>, can be selected from either seven types - φT1, φT4, φT16, φT32, φT64, φT128 and φT256 - of prescaler output clock or the external clock of the TBxIN0 pin.

- Count start/ stop

  Counter operation is specified by TBxRUN<TBRUN>. UC starts counting if <TBRUN> = "1", and stops counting and clears counter value if <TBRUN> = "0".

- Timing to clear UC

  1. When a match is detected

     By setting TBxMOD<TBCLE> = "1", UC is cleared if when the comparator detects a match between counter value and the value set in TBxRG1. UC operates as a free-running counter if TBxMOD<TBCLE> = "0".

  2. When UC stops

     UC stops counting and clears counter value if TBxRUN<TBRUN> = "0".

- UC overflow

  If UC overflow occurs, the INTTBx overflow interrupt is generated.

13.5.3 Timer registers (TBxRG0, TBxRG1)

TBxRG0 and TBxRG1 are registers for setting values to compare with up-counter values and two registers are built into each channel. If the comparator detects a match between a value set in this timer register and that in a UC up-counter, it outputs the match detection signal.

TBxRG0 and TBxRG1 are consisted of the double-buffered configuration which are paired with register buffers. The double buffering is disabled in the initial state.

Controlling double buffering disable or enable is specified by TBxCR<TBWBF> bit. If <TBWBF> = "0", the double buffering becomes disable. If <TBWBF> = "1", it becomes enable. When the double buffering is enabled, a data transfer from the register buffer to the timer register (TBxRG0/1) is done in the case that UC is matched with TBxRG1.When the counter is stopped even if double buffering is enabled, the double buffering operates as a single buffer, and an immediate data can be written to the TBxRG0 and TBxRG1.

13.5.4 Capture

This is a circuit that controls the timing of latching values from the UC up-counter into the TBxCP0 and TBxCP1 capture registers. The timing with which to latch data is specified by TBxMOD<TBCPM[1:0]>.

Software can also be used to import values from the UC up-counter into the capture register; specifically, UC values are taken into the TBxCP0 capture register each time "0" is written to TBxMOD<TBCP>.

13.5.5 Capture registers (TBxCP0, TBxCP1)

This register captures an up-counter (UC) value.

13.5.6 Up-counter capture register (TBxUC)

Other than the capturing functions shown above, the current count value of the UC can be captured by reading the TBxUC registers.

13.5.7 Comparators (CP0, CP1)

This register compares with the up-counter (UC) and the value setting of the Timer Register (TBxRG0 and TBxRG1) to detect whether there is a match or not. If a match is detected, INTTBx is generated.

13.5.8 Timer Flip-flop (TBxFF0)

The timer flip-flop (TBxFF0) is reversed by a match signal from the comparator and a latch signal to the capture registers. It can be enabled or disabled to reverse by setting the TBxFFCR<TBC1T1, TBC0T1, TBE1T1, TBE0T1>.

The value of TBxFF0 becomes undefined after a reset. The flip-flop can be reversed by writing "00" to TBxFFCR<TBFF0C[1:0]>. It can be set to "1" by writing "01," and can be cleared to "0" by writing "10."

The value of TBxFF0 can be output to the Timer output pin (TBxOUT). If the timer output is performed, the corresponding port settings must be programmed beforehand.

13.5.9 Capture interrupt (INTCAPx0, INTCAPx1)

Interrupts INTCAPx0 and INTCAPx1 can be generated at the timing of latching values from the UC up-counter into the TBxCP0 and TBxCP1 capture registers. The interrupt timing is specified by the CPU.

13.6 Description of Operations for Each Mode

13.6.1 16-bit Interval Timer Mode

In the case of generating constant period interrupt, set the interval time to the Timer register (TBxRG1) to generate the INTTBx interrupt.

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops count operation.
Interrupt Set-Enable Register	←	*	*	*	*	*	*	*	*	Permits INTTBx interrupt by setting corresponding bit to "1".
TBxFFCR	←	X	X	0	0	0	0	1	1	Disable to TBxFF0 reverse trigger
TBxMOD	←	0	1	0	0	1	*	*	*	Changes to prescaler output clock as input clock. Specifies Capture function to disable.
						(*** = 001 to 111)				
TBxRG1	←	*	*	*	*	*	*	*	*	Specifies a time interval. (16 bits)
	←	*	*	*	*	*	*	*	*	
TBxRUN	←	*	*	*	*	*	1	X	1	Starts TMRBx.

Note: X; Don't care −; No change

13.6.2 16-bit Event Counter Mode

It is possible to make it the event counter by using an input clock as an external clock (TBxIN0 pin input).

The up-counter counts up on the rising edge of TBxIN0 pin input. It is possible to read the count value by capturing value using software and reading the captured value.

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops count operation.
PxIE[m]	←								1	Allocates corresponding port toTBxIN0.
PxFR1[m]	←								1	
TBxFFCR	←	X	X	0	0	0	0	1	1	Disables to TBxFF0 reverse trigger
TBxMOD	←	0	1	0	0	0	0	0	0	Changes toTBxIN0 as an input clock
TBxRUN	←	*	*	*	*	*	1	X	1	Starts TMRBx.
TBxMOD	←	X	0	0	0	0	0	0	0	Software capture is done.

Note 1: m: corresponding bit of port
Note 2: X; Don't care
−; No change

13.6.3 16-bit PPG (Programmable Pulse Generation) Output Mode

Square waves with any frequency and any duty (programmable square waves) can be output. The output pulse can be either low-active or high-active.

Programmable square waves can be output from the TBxOUT pin by triggering the timer flip-flop (TBxFF) to reverse when the set value of the up-counter (UC) matches the set values of the timer registers (TBxRG0 and TBxRG1). Note that the set values of TBxRG0 and TBxRG1 must satisfy the following requirement:

(Set value of TBxRG0) < (Set value of TBxRG1)

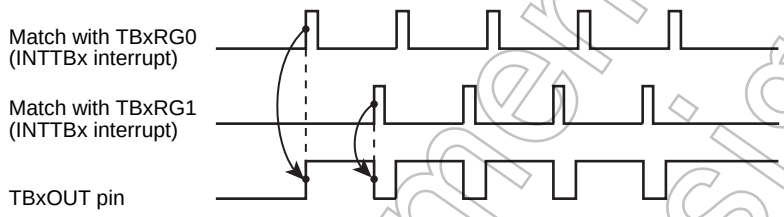


Figure 13-2 Example of Output of Programmable Pulse Generation (PPG)

In this mode, by enabling the double buffering of TBxRG0, the value of register buffer 0 is shifted into TBxRG0 when the set value of the up-counter matches the set value of TBxRG1. This facilitates handling of small duties.

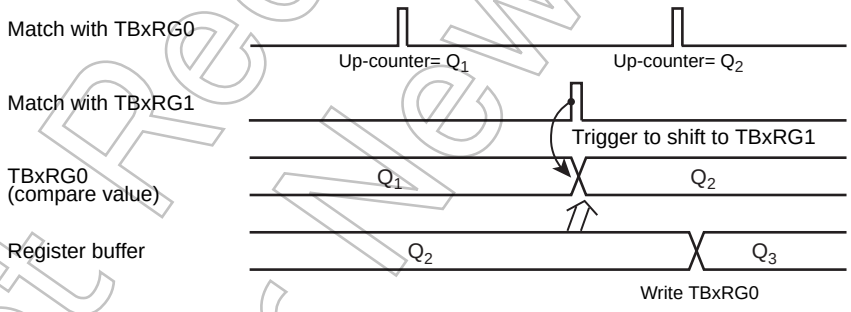


Figure 13-3 Register Buffer Operation

The block diagram of this mode is shown below.

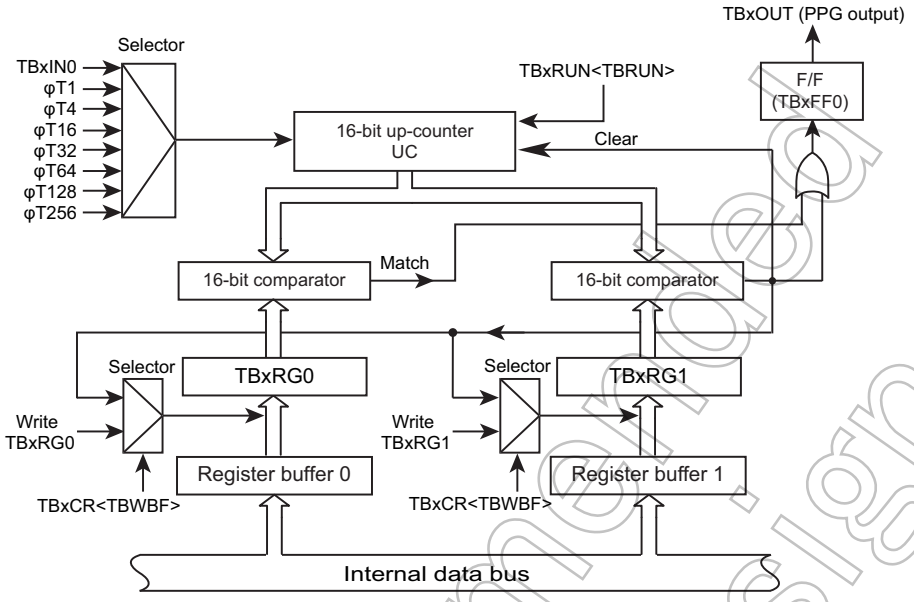


Figure 13-4 Block Diagram of 16-bit PPG Mode

Each register in the 16-bit PPG output mode must be programmed as listed below.

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops count operation.
TBxCR	←	0	0	−	X	−	X	X	X	Disables double buffering.
TBxRG0	←	*	*	*	*	*	*	*	*	Specifies a duty. (16 bits)
	←	*	*	*	*	*	*	*	*	
TBxRG1	←	*	*	*	*	*	*	*	*	Specifies a cycle. (16 bits)
	←	*	*	*	*	*	*	*	*	
TBxCR	←	1	0	X	0	0	0	0	0	Enables the TBxRG0 double buffering.
										(Changes the duty/cycle when the INTTBx interrupt is generated)
TBxFFCR	←	X	X	0	0	1	1	1	0	Specifies to trigger TBxFF0 to reverse when a match with TBxRG0 or TBxRG1 is detected,and sets the initial value of TBxFF0 to "0."
TBxMOD	←	0	1	0	0	1	*	*	*	Designates the prescaler output clock as the input clock,and disables the capture function.
							(** = 001 to 111)			
PxCR[m]	←						1			Allocates corresponding port to TBxOUT.
PxFR1[m]	←						1			
TBxRUN	←	*	*	*	*	*	1	X	1	Starts TMRBx.

Note 1: m: corresponding bit of port

Note 2: X; Don't care

−; No change

13.6.4 Timer synchronous mode

This mode enables the timers to start synchronously.

If the mode is used with PPG output, the output can be applied to drive a motor.

TMRB is consisted of two pairs of 4-channel TMRB. If one channel starts, remaining 3 channels can be start synchronously. In the TMPM366FDXBG/FYXBG/FWXBG, the following combinations allow to use.

Start trigger channel (Master channel)	Synchronous operation channel (Slave channel)
TMRB0	TMRB1, TMRB2, TMRB3
TMRB4	TMRB5, TMRB6, TMRB7

Use of the timer synchronous mode is specified in TBxCR<TBSYNC> bit.

- <TBSYNC> = "0" : Timer operates individually.
- <TBSYNC> = "1" : Timers operates synchronously.

Set "0" to the <TBSYNC> bit in the master channel.

If <TBSYNC>= "1" is set in the slave channel, the start timing is synchronized with master channel start timing. Setting of start timing for TBxRUN<TBPRUN, TBRUN> bit in the slave channel is not required.

Note 1: The channels designated for synchronous output must be started by TBxRUN<TBPRUN,TBRUN>="1,1" before the start triggered by TMRB0 and TMRB4.

Note 2: Set TBxCR<TBSYNC> to "0" unless the timer synchronous mode is used. The timer synchronous mode keeps the other channels operation waiting until TMRB0 and TMRB4 start operation.

Note 3: TMRB0 and TMRB4 are the master clocks of the timer synchronous mode. Therefore, their <TBSYNC> bit must be set to "0".

Note 4: This mode cannot be applied to TMRB8 and TMRB9.

13.6.5 External trigger count start mode

The external trigger count start mode can be set to start counting by an external signal.

Set TBxCR<CSSEL> to select the count start mode.

- <CSSEL> = "0" : Start counting according to the timing of each timer channel.
- <CSSEL> = "1" : Start counting by an external signal.

Set TBxCR<TRGSEL> to select the active edge of the external trigger signal.

- <TRGSEL> = "0" : The rising edge of TBxIN0 is selected.
- <TRGSEL> = "1" : The falling edge of TBxIN0 is selected.

Timer synchronous mode has a priority if its mode is specified.

13.7 Applications using the Capture Function

The capture function can be used to develop many applications, including those described below:

1. One-shot pulse output triggered by an external pulse
2. Frequency measurement
3. Pulse width measurement
4. Time difference measurement

13.7.1 One-shot pulse output triggered by an external pulse

One-shot pulse output triggered by an external pulse is carried out as follows:

The 16-bit up-counter is made to count up by putting it in a free-running state using the prescaler output clock. An external pulse is input through the TBxIN0 pin. A trigger is generated at the rising of the external pulse by using the capture function and the value of the up-counter is taken into the capture registers (TBxCP0).

The CPU must be programmed so that an interrupt INTCAPx0 is generated at the rising of an external trigger pulse. This interrupt is used to set the timer registers (TBxRG0) to the sum of the TBxCP0 value (c) and the delay time (d), (c + d), and set the timer registers (TBxRG1) to the sum of the TBxRG0 values and the pulse width (p) of one-shot pulse, (c + d + p).[TBxRG1 change must be completed before the next match.]

In addition, the timer flip-flop control registers(TBxFFCR<TBE1T1, TBE0T1>) must be set to "11". This enables triggering the timer flip-flop (TBxFF0) to reverse when TBxUC matches TBxRG0 and TBxRG1. This trigger is disabled by the INTTBx interrupt after a one-shot pulse is output.

Symbols (c), (d) and (p) used in the text correspond to symbols c, d and p in "Figure 13-5 One-shot Pulse Output (With Delay)".

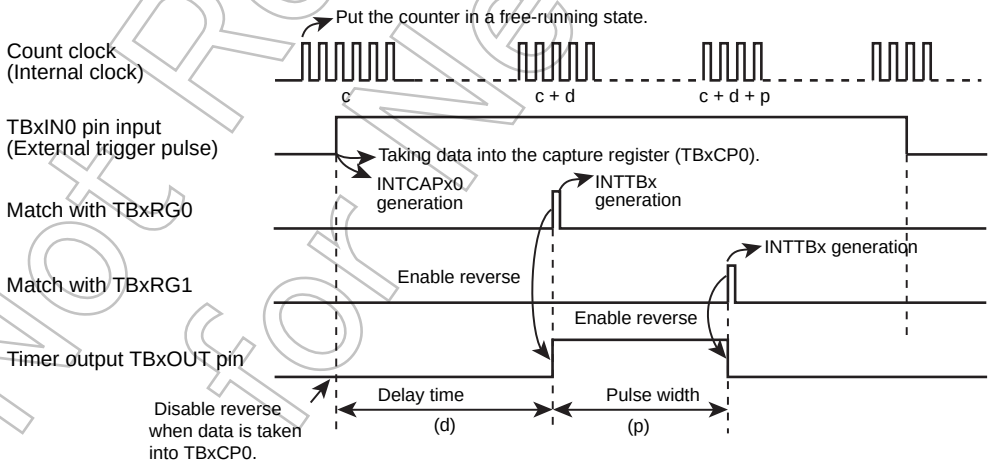


Figure 13-5 One-shot Pulse Output (With Delay)

The followings show the settings in the case that 2 ms width one-shot pulse is output after 3ms by triggering TBxIN0 input at the rising edge. (ΦT1 is selected for counting.)

Changes source clock to ΦT1. Fetches a count value into the TBxCP0 at the rising edge of TBxIN0.

		7	6	5	4	3	2	1	0	
[Main processing] Capture setting by TBxIN0										
PxIE[m]	←								1	Allocates corresponding port to TBxIN0.
PxFR1[m]	←								1	
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops TMRBx operation
TBxMOD	←	0	1	0	1	0	0	0	1	Changes source clock to ΦT1. Fetches a count value into the TBxCP0 at the rising edge of TBxIN0.
TBxFFCR	←	X	X	0	0	0	0	1	0	Clears TBxFF0 reverse trigger and disables.
PxCR[m]	←								1	Allocates corresponding port toTBxOUT.
PxFR1[m]	←								1	
Interrupt Set-Enable Register	←	*	*	*	*	*	*	*	*	Permits to generate interrupts specified by INTCAPx0 interrupt corresponding bit by setting to "1".
TBxRUN	←	*	*	*	*	*	1	X	1	Starts the TMRBx module.
[Processing of INTCAPx0 interrupt service routine] Pulse output setting										
TBxRG0	←	*	*	*	*	*	*	*	*	Sets count value. (TBxCAP0 + 3ms/ΦT1)
TBxRG1	←	*	*	*	*	*	*	*	*	Sets count value.(TBxCAP0 + (3+2)ms/ΦT1)
TBxFFCR	←	X	X	−	−	1	1	−	−	Reverses TBxFF0 if TBxRG0 consistent with TBxRG1.
TBxIM	←	X	X	X	X	X	1	0	1	Masks except TBxRG1 correspondence interrupt.
Interrupt Set-Enable Register	←	*	*	*	*	*	*	*	*	Permits to generate interrupt specified by INTTBx interrupt corresponding bit setting to "1".
[Processing of INTTBx interrupt service routine] Output disable										
TBxFFCR	←	X	X	−	−	0	0	−	−	Clears TBxFF0 reverse trigger setting.
Interrupt enable clear register	←	*	*	*	*	*	*	*	*	Prohibits interrupts specified by INTTBx interrupt corresponding bit by setting to "1".

Note 1: m: corresponding bit of port

Note 2: X; Don't care

−; No change

If a delay is not required, TBxFF0 is reversed when data is taken into TBxCP0, and TBxRG1 is set to the sum of the TBxCP0 value (c) and the one-shot pulse width (p), (c + p), by generating the INTCAPx0 interrupt. (TBxRG1 change must be completed before the next match.)

TBxFF0 is enabled to reverse when UC matches with TBxRG1, and is disabled by generating the INTTBx interrupt.

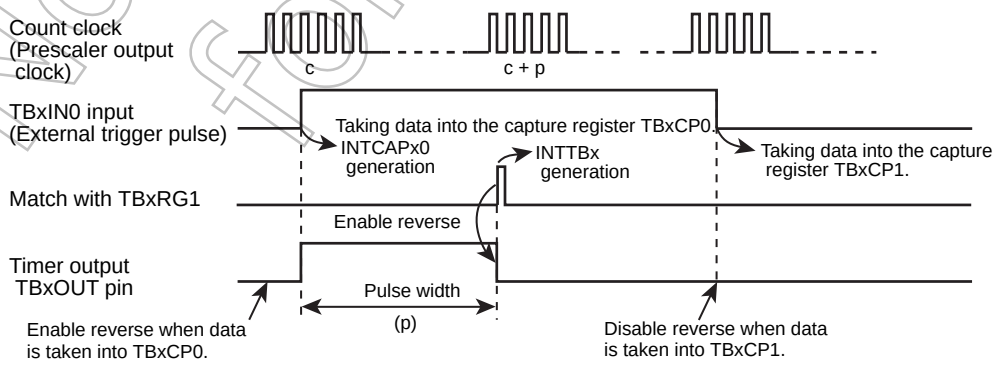


Figure 13-6 One-shot Pulse Output Triggered by an External Pulse (Without Delay)

13.7.2 Frequency measurement

The frequency of an external clock can be measured by using the capture function.

To measure frequency, another 16-bit timer is used in combination with the 16-bit event counter mode. As an example, we explain with TMRB3 and TMRB8. TB8OUT of the 16-bit timer TMRB8 is used to specify the measurement time.

TMRB3 count clock selects TB3IN0 input and performs count operation by using external clock input. If TB3MOD<TBCPM[1:0]> is set "11", TMRB3 count clock takes the counter value into the TB3CP0 at the rising edge of TB8OUT and takes the counter value into TB3CP1 at the falling edge of TB8OUT.

This setting allows a count value of the 16-bit up-counter UC to be taken into the capture register (TB3CP0) upon rising of a timer flip-flop output (TB8OUT) of the 16-bit timer (TMRB8), and an UC counter value to be taken into the capture register (TB3CP1) upon falling of TB8OUT of the 16-bit timer (TMRB8).

A frequency is then obtained from the difference between TB3CP0 and TB3CP1 based on the measurement, by generating the INTTB8 16-bit timer interrupt.

For example, if the difference between TB3CP0 and TB3CP1 is 100 and the level width setting value of TB8OUT is 0.5 s, the frequency is 200 Hz (100 ÷ 0.5 s = 200 Hz).

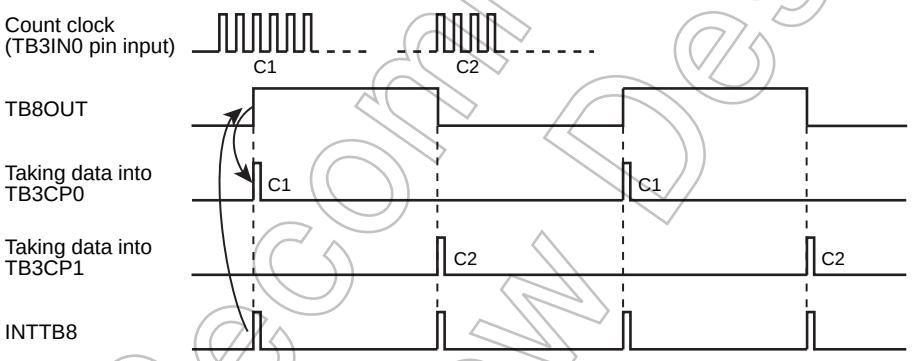


Figure 13-7 Frequency Measurement

13.7.3 Pulse width measurement

By using the capture function, the "High" level width of an external pulse can be measured. Specifically, by putting it in a free-running state using the prescaler output clock, an external pulse is input through the TBxIN0 pin and the up-counter (UC) is made to count up. A trigger is generated at each rising and falling edge of the external pulse by using the capture function and the value of the up-counter is taken into the capture registers (TBxCP0, TBxCP1). The CPU must be programmed so that INTCAPx1 is generated at the falling edge of an external pulse input through the TBxIN0 pin.

The "High" level pulse width can be calculated by multiplying the difference between TBxCP0 and TBxCP1 by the clock cycle of an internal clock.

For example, if the difference between TBxCP0 and TBxCP1 is 100 and the cycle of the prescaler output clock is 0.5 μs , the pulse width is 100 × 0.5 μs = 50 μs.

Caution must be exercised when measuring pulse widths exceeding the UC maximum count time which is dependant upon the source clock used. The measurement of such pulse widths must be made using software.

The "Low" level width of an external pulse can also be measured. In such cases, the difference between C2 generated the first time and C1 generated the second time is initially obtained by performing the second stage of INTCAPx0 interrupt processing as shown in "Figure 13-8 Pulse Width Measurement" and this difference is multiplied by the cycle of the prescaler output clock to obtain the "Low" level width.

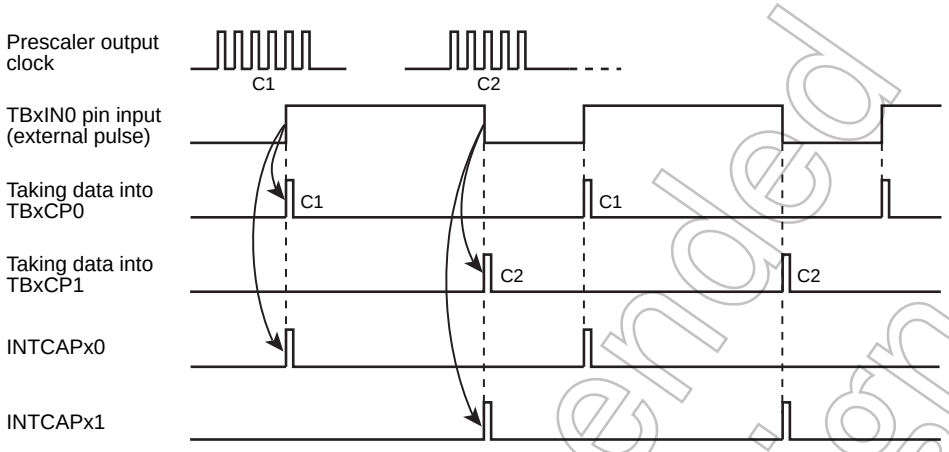


Figure 13-8 Pulse Width Measurement

13.7.4 Time Difference Measurement

The time difference of two events can be measured by the capture function. The up-counter (UC) is made to count up by putting it in a free-running state using the prescaler output clock.

The value of UC is taken into the capture register (TBxCP0) at the rising edge of the TBxIN0 pin input pulse. The CPU must be programmed to generate INTCAPx0 interrupt at this time.

The value of UC is taken into the capture register (TBxCP1) at the rising edge of the TBxIN1 pin input pulse. The CPU must be programmed to generate INTCAPx1 interrupt at this time.

The time difference can be calculated by multiplying the difference between TBxCP1 and TBxCP0 by the clock cycle of an internal clock.

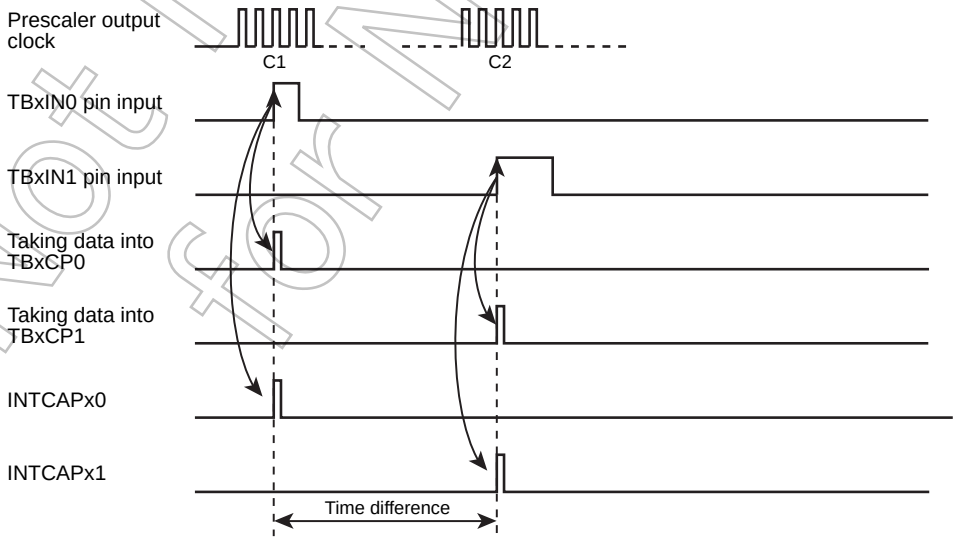


Figure 13-9 Time Difference Measurement

14. USB Device Controller (USBD)

This section describes about USB device controller.

An endpoint is described as EP in this section.

14.1 Outline

1. Conforming to Universal Serial Bus Specification Rev.2.0.
2. Supports Full-Speed (Low-Speed is not supported).
3. USB protocol processing
4. Detects SOF/USB_RESET/SUSPEND/RESUME.
5. Generates and checks packet IDs.
6. Checks CRC5. Generates and checks CRC16.
7. Supports 4 transfer modes (Control/Interrupt /Bulk/ Isochronous).
8. Supports 8 EPs.

Table 14-1 Endpoints

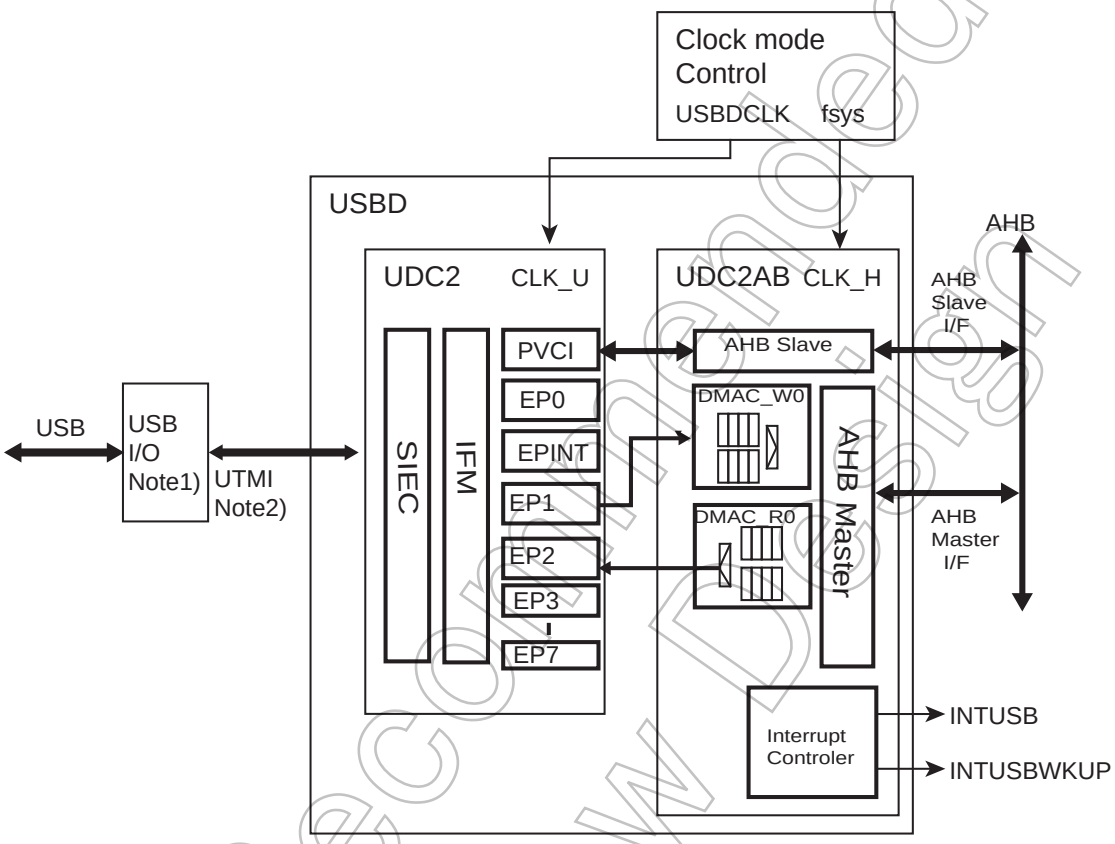
EP0:	Control	64byte × 1 FIFO
EP1:	Control / Interrupt / Bulk / Isochronous (IN)	64byte × 2 FIFO
EP2:	Control / Interrupt / Bulk / Isochronous (OUT)	64byte × 2 FIFO
EP3:	Control / Interrupt / Bulk / Isochronous (IN)	64byte × 2 FIFO
EP4:	Control / Interrupt / Bulk / Isochronous (OUT)	64byte × 2 FIFO
EP5:	Control / Interrupt / Bulk / Isochronous (IN)	64byte × 2 FIFO
EP6:	Control / Interrupt / Bulk / Isochronous (OUT)	64byte × 2 FIFO
EP7:	Control / Interrupt / Bulk / Isochronous (IN)	64byte × 2 FIFO

9. Supports Dual Packets Mode (except for EP 0)
10. Interrupt source signals to the interrupt controller: INTUSB, INTUSBWKUP

14.2 System Structure

The USB device controller consists of the USB-Spec2.0 device controller (hereinafter called UDC2) and the bus bridge (hereinafter called UDC2AB) which connects the UDC2 and the AHB bus.

In this section, "14.2.1 AHB Bus Bridge (UDC2AB)" describes the configuration of the UDC2AB, and "14.2.2 Toshiba USB-Spec2.0 Device Controller (UDC2)" describes that of the UDC2.



Note1) TMPM366FDXBG/FYXBG/FWXBG has USB I/O which can be used for Full Speed mode not Low speed mode. The word "PHY" in this section should be read as USB I/O.
Note2) USB2.0 Transceiver Macrocell Interface

Figure 14-1 Block diagram of the USB device controller

14.2.1 AHB Bus Bridge (UDC2AB)

UDC2AB is the bus bridge between the Toshiba USB-Spec2.0 device controller (hereinafter called UDC2) and the AHB.

UDC2AB has the DMA controller that supports the AHB master transfer and controls transfer between the specified address on the AHB and the Endpoints-FIFO (EP I/F) in the UDC2.

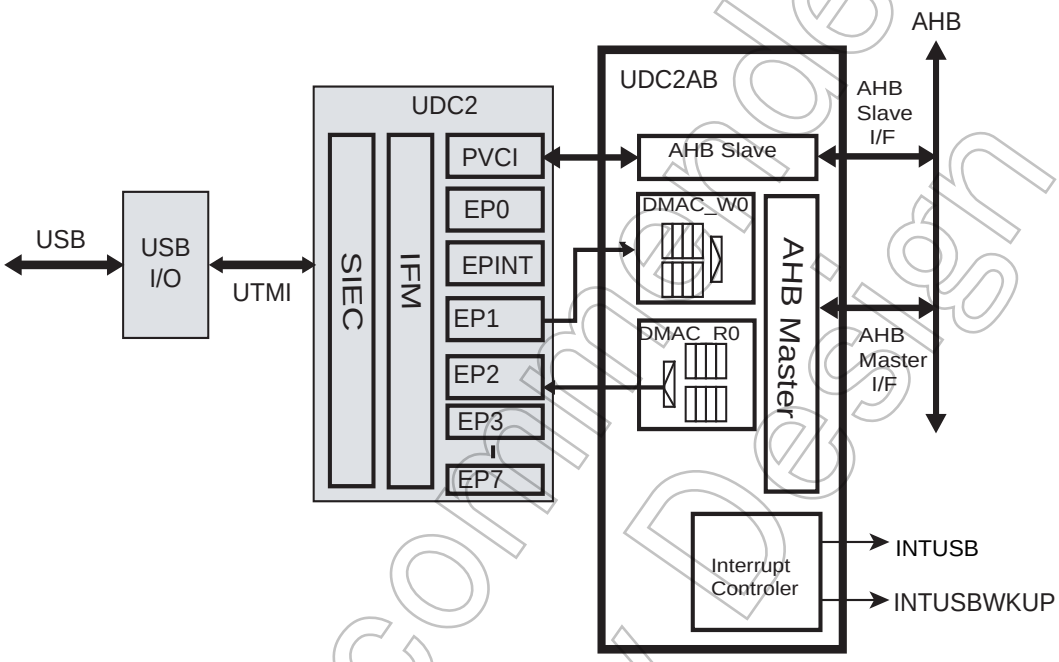


Figure 14-2 UDC2AB Block Diagram

14.2.1.1 Functions and Features

UDC2AB has the following functions and features:

1. Connections with UDC2

   There is no specific restriction on the EP configuration for the UDC2 to be connected. However, the DMA controller in UDC2AB (AHB master function) can be connected with one Rx-EP and one Tx-EP. Accesses to other EPs (including EP0) should be made through PVCI I/F of UDC2 using the AHB slave function. Please note the EPx_FIFO register of a UDC2 EP in master transfer with the DMA controller cannot be accessed through PVCI I/F.

   If the maximum packet size of the EP to be connected with the AHB master read function is an odd number, there are some restrictions on the usage. See "(3) Setting the maximum packet size in Master Read transfers" for more information.

2. AHB functions

   AHB master and AHB slave functions are provided.

   a. AHB master function

      There are two DMA channels available for the USB device controller. One channel is allocated to the Rx-Ep and the Tx-Ep each.

Table 14-2 AHB Master Functions

Single Burst (INCR/INCR8) transactions	Supported
Split transaction	Supported
Little Endian	Supported
Protection Control	Supported
Early Burst Termination	Supported
Address bus width	32-bit
Data bus width	32-bit
Byte Word Transaction	Supported

The image of Endian conversion is as shown below.

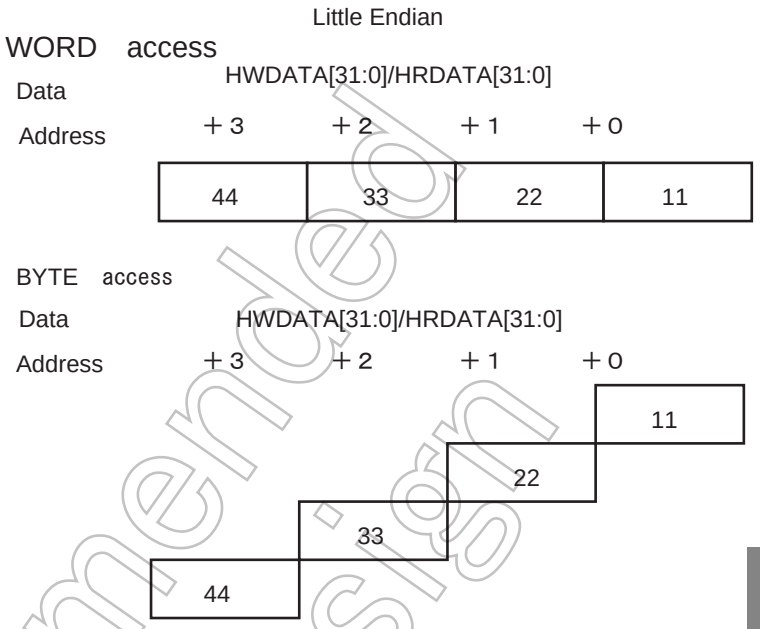


Figure 14-3 Image of Endian conversion in AHB Master function

b. AHB Slave Function

Specification of the AHB Slave function.

Little Endian	Supported
Single transaction	Supported
Address width	32-bit
Data width	32-bit
Transaction in bytes or words	Supported

The image of Endian conversion is as shown below.

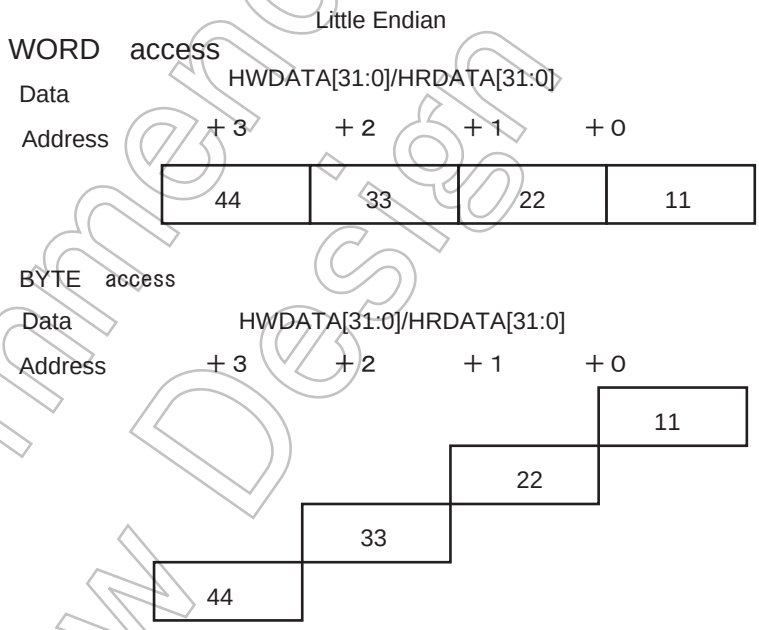


Figure 14-4 Image of Endian conversion in AHB Slave function

14.2.1.2 Configuration

UDC2AB mainly consists of the AHB Slave function that controls the access to the UDC2AB internal registers and UDC2 registers (UDC2 PVCI I/F) and the AHB Master function that controls the DMA access to the UDC2 EP I/F.

The AHB Master function has two built-in channels; Master Read Channel (AHB to UDC2) and Master Write Channel (UDC2 to AHB), which enable DMA transfer between the EP I/F of Rx-EP and Tx-EP of UDC2. Each channel has two built-in 8-word buffers (four in total).

14.2.1.3 Clock Domain

fsys provided by the clock/mode control circuit is connected to CLK_H of the UDC2AB. fsys stops or starts according to the low-power consumption mode of the TMPM366FDXBG/FYXBG/FWXBG.

Since CLK_H is not provided during the fsys being stopped in the low-power consumption mode, INTUSBwill not be generated.

Therefore, to detect the connection and disconnection of the VBUS, an interrupt to used needs to be selected from INTUSBPON generated by the USBPON pin and INTUSB at the moment when CLK_H starts or stops.

Refer to "14.5.5.2 Sequence of USB Bus Power (VBUS) Connection/Disconnection" for more information.

USBDCLK provided from clock / mode control circuit is connected to CLK_U of UDC2. USBDCLK stops or starts according to the register. To stop or start CLK_U by detecting status such as suspend and resume, configure clock / mode control circuit using software.

14.2.2 Toshiba USB-Spec2.0 Device Controller (UDC2)

UDC2 controls the connection of USB functions to the Universal Serial Bus. UDC2 automatically processes the USB protocol and its PHY-end interface can be accessed via UTMI.

1. SIEC (Serial Interface Engine Control) block

   This block manages the protocol in USB. Its major functions are:
   - Checks and generates PIDs
   - Checks and generates CRCs
   - Checks device addresses

2. IFM block

   This block controls SIEC and EPs. Its major functions are:
   - Writes the received data to the relevant EPs when received an OUT-Token.
   - Reads the transmit data from the relevant EPs when an IN-Token is received.
   - Controls and manages the status of UDC2.0.

3. PVCI-I/F block

   This block controls reading and writing between IFM and external register access bus (PVCI).

   PVCI bus accesses via UDC2AB.

4. EP0 block

   This block controls sending and receiving data in Control transfers. When sending or receiving data with DATA-Stage of Control transfers, you should access the FIFO in this block via PVCI-I/F.

5. EPx block

   This block controls sending and receiving data of EPx (x = 1 to 7). FIFO can be directly accessed via the EP-I/F. The EP-I/F can make burst transfers.

   Please note there are two EPs; one for sending (EPTX) and another for receiving (EPRX). Direction of EPs (send/receive) will be fixed on a hardware basis.

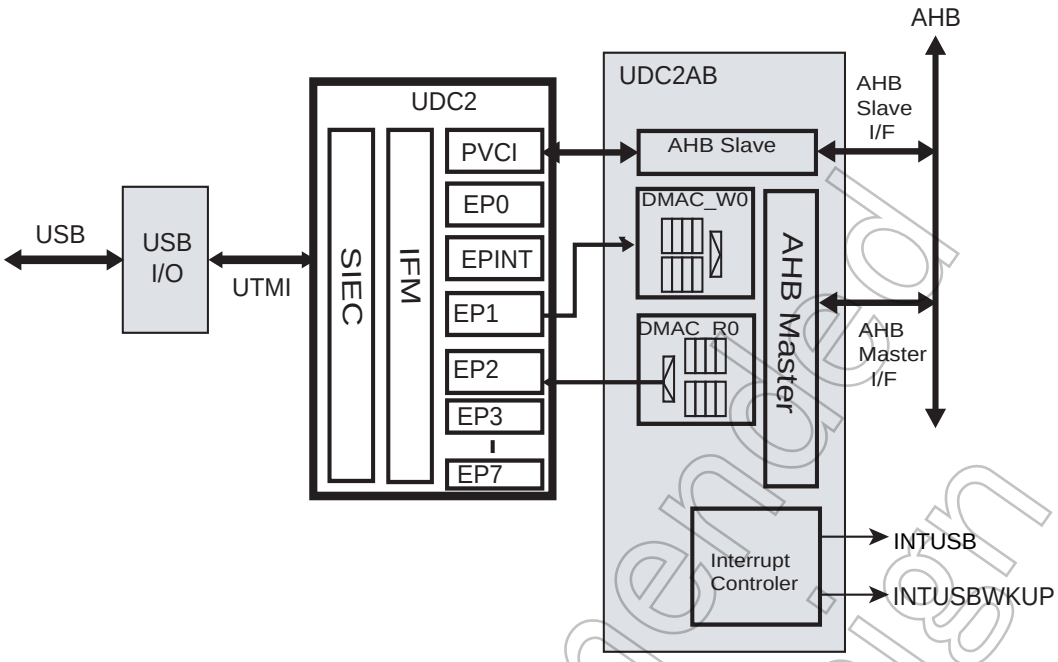


Figure 14-5 Block diagram of UDC2

14.2.2.1 Features and Functions

The main features and functions are as follows:

1. Complies with Universal Serial Bus Specification Rev. 2.0.
2. Complies with Full-Speed (FS) (not complies with Low-Speed).
3. USB protocol processing
4. Detects SOF/USB_RESET/SUSPEND/RESUME.
5. Generates and checks packet IDs.
6. Checks CRC5. Generates and checks CRC16.
7. Supports 4 transfer modes (Control/Interrupt/Bulk/Isochronous).
8. Supports up to 8 EPs.
9. Supports Dual Packet Mode (EP 0 excluded)
10. EP 1 to 7 can directly access FIFO (EP-I/F)
11. Complies with USB 2.0 Transceiver Macrocell Interface (UTMI) (8 bits @ 48 MHz)

14.2.2.2 Specifications of Flags

The UDC2 core outputs various events on USB as flags when they occur. This section discusses those flags.

1. USB_RESET

   Asserts "High" while receiving USB_RESET. Since UDC2 returns to the Default-State by receiving USB_RESET, the application also needs to return to the Default-State.

   In Full-Speed operation, UDC2 asserts this flag when SE0 on the USB bus was recognized for 2.5 s or longer. Then, after UDC2 has driven Chirp-K for about 1.5 ms the flag will be deasserted when either one of the following states was recognized:

   a. Chirp from the host (K-J-K-J-K-J) was recognized.
   b. 2 ms or longer has passed without recognizing Chirp from the host (K-J-K-J-K-J).

Note: While the time when the host begins Chirp and the driving time of Chirp-K and Chirp-J depend on the host, asserting period of the USB_RESET flag is around 1.74 ms to 3.5 ms.

2. INT_SETUP

   In Control transfers, asserts "High" after receiving the Setup-Token. When this interrupt is recognized, the software should read the Setup-Data storage register (8 bytes) to make judgment of request.This interrupt will be deasserted by writing 1 into the UDFS2INT<i_setup>. UDFS2INT should be cleared at the point the interrupt was recognized.

3. INT_STATUS_NAK

   In Control transfers, when the host proceeds to the STATUS-Stage and transmits packets while UDC2 is processing the DATA-Stage (before issuing the "Setup_Fin" command), UDC2 will return "NAK" and asserts this flag to "High". When this interrupt is recognized, the software should issue the "Setup_Fin" command from the Command register to make the STATUS-Stage of UDC2 end. This interrupt will be deasserted by writing 1 into the UDFS2INT<i_status_nak>. UDFS2INT should be cleared at the point the interrupt was recognized.

4. INT_STATUS

   In Control transfers, asserts "High" after finishing the STATUS-Stage normally. This interrupt will be deasserted by writing 1 into the UDFS2INT<i_status>. UDFS2INT should be cleared at the point the interrupt was recognized.

5. INT_EP0

   In the DATA-Stage of Control transfers, asserts "High" when "ACK" was sent or received (when the transaction finished normally). This interrupt will be deasserted by writing 1 to the UDFS2INT<i_ep0>. UDFS2INT should be cleared at the point the interrupt was recognized.

6. INT_EP

In EPs other than EP 0, asserts "High" when "ACK" was sent or received (when the transaction finished normally). In that case, which EP the transfer was made can be identified by checking UDFS2INTEP. This interrupt will be deasserted by writing 1 into the UDFS2INT<i_ep>, or by writing 1 into all bits set in UDFS2INTEP. UDFS2INT should be cleared at the point the interrupt was recognized.

7. INT_RX_ZERO

"High" is asserted when Zero-Length data is received. In Control transfers, however, "High" is asserted only when Zero-Length data is received in the DATA-Stage. It will not be asserted when Zero-Length data is received in the STATUS-Stage. Which EP has received the data can be identified by reading the UDFS2CMD<rx_nulpkt_ep> or checking UDFS2INTRX0. This interrupt will be deasserted by writing 1 into the UDFS2INT<i_rx_data0>, or by writing 1 into all bits set in UDF2INTRX0. UDFS2INTRX0 should be cleared at the point the interrupt was recognized.

8. INT_SOF

Asserts "High" when SOF was received. This interrupt will be deasserted by writing 1 into the UDFS2INT<i_osf>. UDFS2INT should be cleared at the point the interrupt was recognized.

SOF is a packet indicating the start of a frame. It is transmitted from the host to devices every 1ms in the Full-Speed transfers.

9. INT_NAK

In EPs other than EP 0, asserts "High" when NAK is transmitted. In that case, which EP has transmitted the NAK can be identified by checking UDFS2INTNAK. This interrupt will be deasserted by writing 1 into the UDFS2INT<i_nak>, or by writing 1 into all bits set in UDFS2INTNAK. By default, this flag will not be asserted when NAK was transmitted. Therefore, you should write 0 into the relevant EP of UDFS2INTNAKMASK to release the mask in order to use this flag.

14.2.2.3 Commands to EP

This section describes about the commands to be issued by UDFS2CMD<com> for the EP specified by UDFS2CMD<ep>.

1. 0x0 : Reserved

Not to be specified.

2. 0x1 : Setup_Fin

Should be issued only for EP0.

This is a command for setting the end of DATA-Stage in Control transfers. As UDC2 continues to send back "NAK" to the STATUS-Stage until this command is issued, the command should be issued when the DATA-Stage finishes or INT_STATUS_NAK was received.

Note: During Control-WR transfer, read all data received during the Data-Stage first, and then Issue the Setup-Fin command.

3. 0x2 : Set_DATA0

Can be issued to EPs except EP0. Should not be issued to EP0.

A command for clearing toggling of EPs. While toggling is automatically updated by UDC2 in normal transfers, this command should be issued if it needs to be cleared by software.

4. 0x03 : EP_Reset

   Can be issued to any EP.

   A command for clearing the data and status of EPs. Issue this command when you want to reset an EP in such cases as setting EPs of Set_Configuration and Set_Interface or resetting the EP by Clear_Feature. This command will reset the following 5 points:

   a. Clear the UDFS2EP0STS<toggle> / UDFS2EPxSTS<toggle> to DATA0.
   b. Clear the UDFS2EP0STS<status> / UDFS2EPxSTS<status> to Ready.
   c. Clear the UDFS2EP0MSZ<dset> / UDFS2EPxMSZ<dset> and the UDFS2EP0DSZ / UDFS2EPxDSZ.
   d. Clear UDFS2EP0MSZ<tx0_data> / UDFS2EPxMSZ<tx_0data>.
   e. Clear the UDFS2EPxSTS<disable>.

   UDC2 makes toggling control by hardware for every transfer. If this command is issued when a transfer of EPs is in progress, toggling of the relevant EP will also be cleared which may cause the synchronization with the host be lost. As in the case of receiving requests as mentioned above, the command should be issued when it is possible to make synchronization with the host.

5. 0x4 : EP_Stall

   Can be issued to EPs except EP0. Should not be issued to EP0.

   A command for setting the status of EPs to "Stall". Issue this command when you want to set the status of an EP to "Stall" in such cases as stalling an EP by Set_Feature. When this command is issued, "STALL" will be always sent back for the EP set. However, the Stall status of EP0 will be cleared when the Setup-Token is received.

   This command should not be issued for EPs where Isochronous transfers are used, since transfers are made without Handshake in Isochronous transfers. Even if this command is issued for EPs where Isochronous transfers are set (by UDFS2EOxSTS<t_type>), "STALL" will not be sent back.

6. 0x5 : EP_Invalid

   Can be issued to EPs except EP0. Should not be issued to EP0.

   A command for setting the status of EPs to "Invalid". Please issue this command when disabling EPs that will not be used when using Set_Config or Set_Interface to set EPs. When this command is issued, the EPs set will make no response. This command should not be issued while transfers of each EP are in progress.

7. 0x6 : Reserved

   Not to be specified.

8. 0x7 : EP_Disable

   Can be issued to EPs except EP0. Should not be issued to EP0.

   A command for making an EP disabled. When this command is issued, "NAK" will be always sent back from the EP set. This command should not be issued for EPs where Isochronous transfers are used, since transfers are made without Handshake in Isochronous transfers. Even if this command is issued for EPs where Isochronous transfers are set (by UDFS2EPxSTS<t_type>), "NAK" will not be sent back.

9. 0x8 : EP_Enable

Can be issued to EPs except EP0. Should not be issued to EP0.

A command for making an EP enabled. Issue this command to cancel the disabled status set by "EP_Disable" command.

10. 0x9 : All_EP_Invalid

    Setting for EP is invalid.

    A command for setting the status of all EPs other than EP0 to "Invalid". Issue this command when you want to apply the "EP_Invalid" command for all EPs. Issue this command when processing Set_Configuration and Set_Interface like the "EP_Invalid" command.

11. 0xA : USB_Ready

    Should be issued only for EP0.

    A command for making connection with the USB cable. Issue this command at the point when communication with the host has become effective after confirming the connection with the cable. Pull-Up of D+ will be made only after this command is issued, and the status of cable connection will be sent to the host.

    Please note that the device state of UDC2 (UDFS2ADR<configued> <addressed> <default>) will be set to "Default" when this command was issued.

12. 0xB : Setup_Received

    Should be issued for EP0.

    A command for informing UDC2 that the SETUP-Stage of a Control transfer was recognized. Issue this command after accepting the INT_SETUP interrupt and the request code was recognized. As UDC2 continues to send back "NAK" to the DATA-Stage/STATUS-Stage until this command is issued, the command should be issued at the end of the INT_SETUP interrupt processing routine.

13. 0xC : EP_EOP

    Can be issued to any EP.

    A command for informing UDC2 that the transmit data has been written. Issue this command when transmitting data with byte size smaller than the maximum transfer bytes (FIFO capacity of the EP or MaxPacketSize, whichever smaller). Issuing this command will set the Data set flag and the data will be sent back to IN-Token from the host. It should not be used when setting Zero-Length data or data of MaxPacketSize.

14. 0xD : EP_FIFO_Clear

    Can be issued to any EP.

    A command for clearing the data of an EP. The UDFS2EPxMSZ<dset> and the UDFS2EPxDSZ will be cleared at the same time. Issue this command when you want to clear the data currently stored in the FIFO before transmitting the data to the host and set the latest data, for instance in Interrupt transfers. If this command is issued while accessing the EP-I/F, the FIFO of the EP will not be successfully cleared. When issuing this command, epx_val of EP-I/F should be set to 0 before issuing.

15. 0xE : EP_TX_0DATA

    Can be issued to any EP.

    A command for setting Zero-Length data to an EP. Issue this command when you want to transmit Zero-Length data. In the case of transmitting Zero-Length data in Bulk-IN transfers and others to indicate the final transfer, read UDFS2EPxDSZ and confirm it is 0 (no data ex-

ists in the FIFO of EPx) before setting this command. In the case of writing data from EP-I/F, set this command after the data was written and epx_val became 0. When this command was set, UDFS2EPxMS<tx_0data> of the EP will be set.

Set the next data when the UDFS2EPxMS<tx_0data> is confirmed to be 0. During Isochronous-IN transfer, Zero -Length data is automatically transferred to the IN-Token if data is not set in the FIFO of EP. Do not issue this command.

16. 0xF : Reserved

    Not to be specified.

Settings for the following commands will be suspended when issued during a USB transfer, which will be executed after the USB transfer has finished. Suspension of the command will take place for each EP.

- 0x2: Set_DATA0
- 0x3: EP_Reset
- 0x4: EP_Stall
- 0x5: EP_Invalid
- 0x7: EP_Disable
- 0x8: EP_Enable
- 0x9: All_EP_Invalid
- 0xD: EP_FIFO_Clear
- 0xE: EP_TX_0DATA

Therefore, when commands were issued successively for the same EP while a USB transfer is in progress, commands will be overwritten and only the one last issued will be valid. If you need to issue commands to an EP successively, poll UDFS2EPxSTS / UDFS2EPxDSZ to confirm that the command has become valid before issuing next ones. Also, when making an access to the EP-I/F immediately after clearing the FIFO using the EP_Reset/EP_FIFO_Clear command, poll UDFS2EPxDSZ to confirm that the command has become valid before resuming the access to the EP-I/F.

For EP 0, the following commands will be invalid until the Setup_Received command is issued after receiving the Setup-Token:

- 0x1: Setup_Fin
- 0x2: Set_DATA0
- 0x3: EP_Reset
- 0x4: EP_Stall
- 0xC: EP_EOP
- 0xD: EP_FIFO_Clear
- 0xE: EP_TX_0DATA

When the "EP_Stall" command was set to EPx, "Stall" will be set to the UDFS2EPxSTS<status>. When EP_Disable was set, 1 will be set to the UDFS2EPxSTS<disable>. When these two commands (EP_Stall and EP_Disable) were set to the same EPx and the status becomes "Stall" with UDFS2EPxSTS<disable> = 1, "STALL" will be transmitted in the transfer.

When the "EP_Invalid" command was set to EPx, "Invalid" will be set to the UDFS2EPxSTS<status>. When the two commands (EP_Invalid and EP_Disable) were set to the same EPx and the status becomes "Invalid" with UDFS2EPxSTS<disable> = 1, no response will be made in the transfer.

When UDFS2EPxSTS<disable> is 1 and UDFS2EPxMSZ<tx_0data> is 1, Zero-Length data will be transmitted once in the transfer. After the Zero-Length data was successfully transferred, "NAK" will be transmitted.

14.3 How to connect with the USB bus

The circuit below shows how to connect TMPM366FDXBG/FYXBG/FWXBG with the USB bus.

Input the VBUS signal to USBPON pin to detect the connection of the USB power (VBUS).

The Pull-Up process using the pull-up resistor of the USB-DDP and the in-line damping resistor to the USB-DDM are required. Also, a ON/OFF control using a port needs to be added to the pull-up resistor. The pull-up resistor should be disconnected when voltage is not applied to the VBUS.

If USB-DDP and USB-DDM are unstable, we recommend to add a pull-down resistor to R_1.

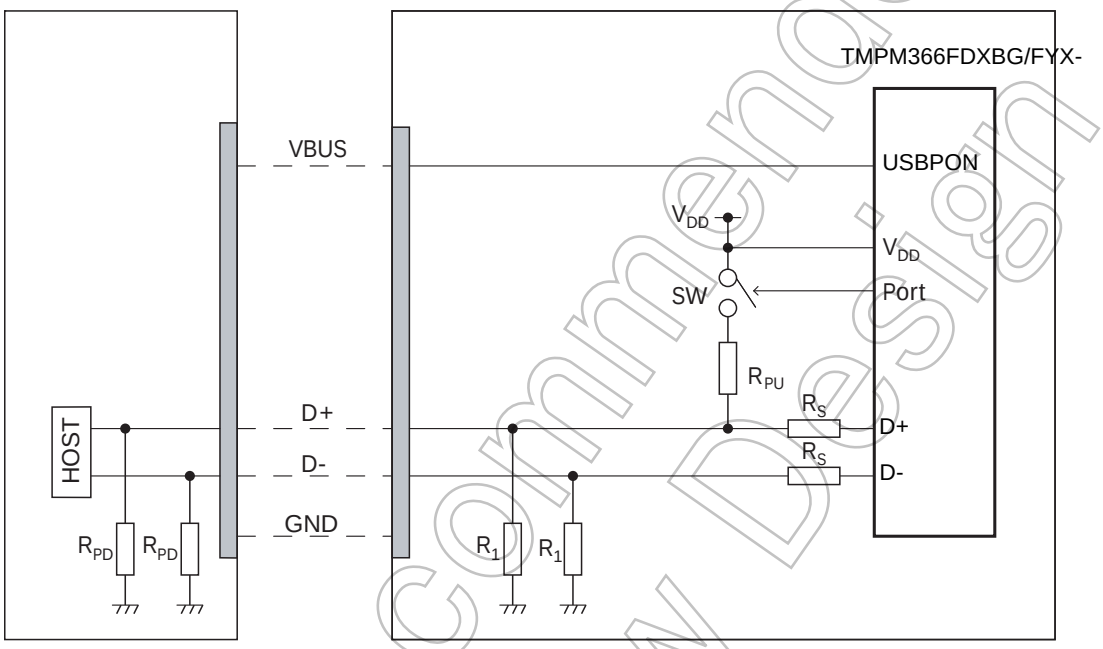


Figure 14-6 Connection example of the USB bus and TMPM366FDXBG/FYXBG/FWXBG.

Note: R_1=500kΩ or higher (recommended value), R_S=33Ω (recommended value), R_{PU}=1.5kΩ (recommended value)

14.4 Registers

The register map of the USBD consists of registers for setting the UDC2AB and that for the UDC2.

When the registers for setting UDC2 are accessed, UDC2AB automatically accesses UDC2 via PVCI I/F.

The register for setting the UDC2AB is 32-bit width. The register for setting the UDC2 is 16-bit width, which is allocated to [15:0]. [31:16] is allocated to the read-only, for indefinite values.

14.4.1 UDC2AB Register

14.4.1.1 UDC2AB Register list

BaseAddress=0x4000_8000

Register name		Address(Base+)
Interrupt Status Register	UDFSINTSTS	0x0000
Interrupt Enable Register	UDFSINTENB	0x0004
Master Write Timeout Register	UDFSMWTOUT	0x0008
UDC2 Setting Register	UDFSC2STSET	0x000C
DMAC Setting register	UDFSMSTSET	0x0010
DMAC Read Request Register	UDFSDMACRDREQ	0x0014
DMAC Read Value Register	UDFSDMACRDVL	0x0018
UDC2 Read Request Register	UDFSUDC2RDREQ	0x001C
UDC2 Read Value Register	UDFSUDC2RDVL	0x0020
-	Reserved	0x0024 to 0x0038 (note 2)
Arbiter Setting Register	UDFSARBTSET	0x003C
Master Write Start Address Register	UDFSMWSADR	0x0040
Master Write End Address Register	UDFSMWEADR	0x0044
Master Write Current Address Register	UDFSMWCADR	0x0048 (note 1)
Master Write AHB Address Register	UDFSMWAHBADR	0x004C
Master Read Start Address Register	UDFSMRSADR	0x0050
Master Read End Address Register	UDFSMREADR	0x0054
Master Read Current Address Register	UDFSMRCADR	0x0058 (note 1)
Master Read AHB Address Register	UDFSMRAHBADR	0x005C
-	Reserved	0x0060 to 0x007C (note 2)
Power Detect Control Register	UDFSPWCTL	0x0080
Master Status Register	UDFSMSTSTS	0x0084
Timeout Count Register	UDFSTOUTCNT	0x0088 (note 1)
-	Reserved	0x008C to 0x1FC

Note 1: Be sure to make Read accesses via UDFSDMACRDREQ.

Note 2: Those shown as "Reserved" above are prohibited to access. Read / Write is prohibited to those "Reserved" areas.

14.4.1.2 UDFSINTSTS (Interrupt Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	int_mw_rerror	int_ powerdetect	-	-	int_dmac_ reg_rd	int_udc2_ reg_rd
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	int_mr_ahberr	int_mr_ep_ dset	int_mr_end_ add	int_mw_ ahberr	int_mw_ timeout	int_mw_end_ add	int_mw_set_ add	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	int_usb_ reset_end	int_usb_reset	int_suspend_ resume
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	int_nak	int_ep	int_ep0	int_sof	int_rx_zero	int_status	int_status_ nak	int_setup
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-30	-	R	Read as undefined.
29	int_mw_rerror	R/W	Will be set to 1 when the access to the EP has started Master Write transfer during the setting of common bus access (UDFS2EPxSTS<bus_sel> is 0).(UDFS2EPxSTS<bus_sel> is 0) 0: Not detected 1: EP read error occurred during master write.
28	int_ powerdetect	R/W	When the status of VBUSPOWER input of UDC2AB changed, it is set to "1". 0:No changed 1:Status changed
27-26	-	R	Read as undefined.
25	int_dmac_ reg_rd	R/W	Will be set to 1 when the register access executed by the setting of UDFSDMACRDREQ is completed and the value read to UDFSDMACCRDVL is set. 0: Not detected. 1:Register read completed.
24	int_udc2_reg_ rd	R/W	Will be set to 1 when the UDC2 access executed by the setting of UDFSDMACRDREQ is completed and the value read to UDFSDMACRDLV is set. Also set to 1 when Write access to the internal register of UDC2 is completed. 0: Not detected. 1: Register read/write completed.
23	int_mr_ahberr	R/W	This status will be set to 1 when the AHB error has occurred during the operation of Master Read transfer. After this interrupt has occurred, the Master Read transfer block needs to be reset by the UDFSMST-SET<mr_reset >. 0: Not detected. 1: AHB error occurred.
22	int_mr_ep_dset	R/W	Will be set to 1 when the FIFO of EP for UDC2 Tx to be used for Master Read transfer becomes writable (not full). 0: FIFO is not writable 1: FIFO is writable
21	int_mr_end_ add	R/W	Will be set to 1 when the Master Read transfer has finished. 0: Not detected 1: Master Read transfer finished
20	int_mw_ahberr	R/W	This status will be set to 1 when the AHB error has occurred during the operation of Master Write transfer. After this interrupt has occurred, the Master Write transfer block needs to be reset by UDFSMSTSET<mw_re-set>. 0: Not detected. 1: AHB error occurred.

Bit	Bit Symbol	Type	Function
19	int_mw_ timeout	R/W	This status will be set to 1 when time-out has occurred during the operation of Master Write transfer. 0: Not detected. 1: Master Write transfer timed out.
18	int_mw_end_ add	R/W	Will be set to 1 when the Master Write transfer has finished. 0: Not detected. 1: Master Write transfer timed out.
17	int_mw_set_ add	R/W	Will be set to 1 when the data to be sent by Master Write transfer is set to the corresponding EP of Rx while the Master Write transfer is disabled. 0: Not detected. 1: Master Write Transfer address request.
16-11	-	R	Read as undefined.
10	int_usb_reset_ end	R/W	Indicates whether UDC2 has deasserted the usb_reset signal. The timing in which UDC2 sets the UDC2 register to the initial value after USB_RESET is after the usb_reset signal is deasserted. To detect this timing, use this bit. The status of the usb_reset signal can be checked using UDFSPWCTL<usb_reset>. 0: UDC2 has not deasserted the usb_reset signal after this bit was cleared. 1: Indicates UDC2 has deasserted the usb_reset signal.
9	int_usb_reset	R/W	Indicates whether UDC2 has asserted the usb_reset signal. The status of the usb_reset signal can be checked using UDFSPWCTL<usb_reset>. 0: UDC2 has not asserted the usb_reset signal after this bit was cleared. 1:Indicates UDC2 has asserted the usb_reset signal.
8	int_suspend_ resume	R/W	Asserts 1 each time the suspend_x signal of UDC2 changes. The status can be checked using the UDFSPWCTL<suspend_x>. 0:Status has not changed. 1: Status has changed.
7	int_nak	R	The int_nak signal of UDC2 can be directly read. To clear it, clear the corresponding bit of UDFS2INT or UDFS2INTNAK of UDC2.
6	int_ep	R	The int_ep signal of UDC2 can be directly read. To clear it, clear the corresponding bit of UDFS2INT or UDFS2INTEP.
5	int_ep0	R	The int_ep0 signal of UDC2 can be directly read. To clear it, clear the corresponding bit of UDFS2INT.
4	int_sof	R	The int_ep0 signal of UDC2 can be directly read. To clear it, clear the corresponding bit of UDFS2INT.
3	int_rx_zero	R	The int_rx_zero signal of UDC2 can be directly read. To clear it, clear the corresponding bit of UDFS2INT or UDFS2INTRX0.
2	int_status	R	The int_status signal of UDC2 can be directly read. To clear it, clear the corresponding bit of UDFS2INT.
1	int_status_nak	R	The int_status_nak signal of UDC2 can be directly read. To clear it, clear the corresponding bit of UDFS2INT.
0	int_setup	R	The int_setup signal of UDC2 can be directly read. To clear it, clear the corresponding bit of UDFS2INT.

The connection between the output signals of UDC2 and the bits [10:9] and [7:0]of this register is shown below.

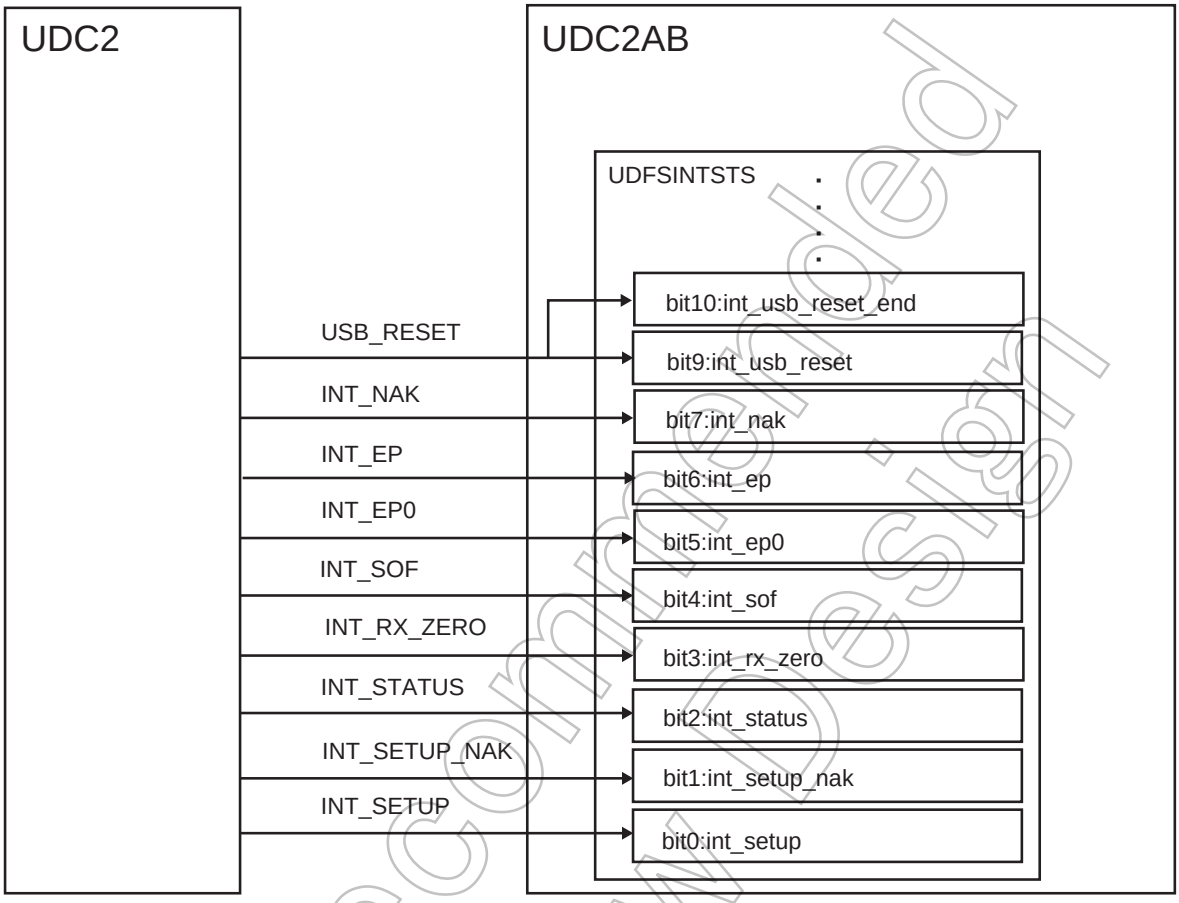


Figure 14-7 Connection between the flag output signals and interrupt bits.

14.4.1.3 UDFSINTENB(Interrupt Enable Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	mw_rerror_en	power_ detect_en	-	-	dmac_reg_ rd_en	udc2_reg_ rd_en
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	mr_ahberr_en	mr_ep_ dset_en	mr_end_ add_en	mw_ahberr_ en	mw_ timeout_en	mw_end_ add_en	mw_set_ add_en	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	usb_reset_ end_en	usb_reset_en	suspend_ resume_en
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
31-30	-	R	Read as undefined.
29	mw_rerror_en	R/W	Controls the mw_rerror interrupt. 0: Disable 1: Enable
28	power_detect_ en	R/W	Controls the power_detect interrupt. 0: Disable 1: Enable
27-26	-	R	Read as undefined.
25	dmac_reg_ rd_en	R/W	Controls the dmac_reg_rd interrupt. 0: Disable 1: Enable
24	udc2_reg_ rd_en	R/W	Controls the udc2_reg_rd interrupt. 0: Disable 1: Enable
23	mr_ahberr_en	R/W	Controls the mw_ahberr interrupt. 0: Disable 1: Enable
22	mr_ep_dset_en	R/W	Controls the mr_ep_dset interrupt. 0: Disable 1: Enable
21	mr_end_ add_en	R/W	Controls the mr_end_add interrupt. 0: Disable 1: Enable
20	mw_ahberr_en	R/W	Controls the mw_ahberr interrupt. 0: Disable 1: Enable
19	mw_ timeout_en	R/W	Controls the mw_timeout interrupt. 0: Disable 1: Enable
18	mw_end_ add_en	R/W	mw_end_add interrupt. 0: Disable 1: Enable
17	mw_set_ add_en	R/W	mw_set_add interrupt. 0:Disable 1: Enable
16-11	-	R	Read as undefined.

Bit	Bit Symbol	Type	Function
10	usb_reset_ end_en	R/W	usb_reset_end interrupt. 0: Disable 1: Enable
9	usb_reset_en	R/W	usb_reset interrupt. 0: Disable 1: Enable
8	suspend_ resume_en	R/W	suspend_resume interrupt. 0: Disable 1: Enable
7-0	-	R	Read as undefined.

14.4.1.4 UDFSMWTOUT(Master Write Timeout Register)

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

Bit	Bit Symbol	Type	Function
31-1	timeoutset	R/W	The setting should not be changed during the Master Write transfer. Timeout occurs when the number of times CLK_U was set is counted after the data of Master Write (Rx) EP is exhausted. The timeout counter comprises 32 bits of which upper 31 bits can be set by timeoutset [31:1] of this register, while the lowest bit of the counter is set to 1. As CLK_U is 48 MHz, approximately 20 [ns] to 89 [s] can be set as a timeout value. While CLK_U stopped (PHY is being suspended and so on), no timeout interrupt will occur as the counter does not work.
0	timeout_en	R/W	Used to enable Master Write timeout. It is set to Enable by default. The setting should not be changed during the Master Write transfer. 0: Disable 1: Enable

14.4.1.5 UDFSC2STSET(UDC2 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	-			eopb_enable	-	-	-	tx0
After reset	0	0	0	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Read as undefined.
4	eopb_enable	R/W	Used to enable Master Read EOP. It is set to Enable by default. The setting should not be changed during the Master Read transfer. If this bit is 0, the final data transfer to UDC2 will not take place when the last word is 1 byte. If the last word is 2 bytes, the final data transfer to UDC2 will take place when epx_w_eop = 0. If this bit is 1, the final data transfer to UDC2 will take place when epx_w_eop = 1 regardless of byte number of the last word. Note: See the section"14.5.4.1 Master Read transfer" for more information. 0: Master Read EOP Disabled. 1: Master Read EOP Enabled.
3-1	−	R	Read as undefined.
0	tx0	R/W	Used to transmit NULL packets at an EP connected to the Master Read operation side. Only valid when the UDFSMSTSTS<mrepempty> is 1, otherwise this bit is ignored. It will be automatically cleared to 0 after writing. Setting 1 to this bit will assert the epx_tx0data signal of the UDC2 EP-I/F and the value of 1 is retained during the transmission of NULL packets. After this bit is set, next data setting for Tx-EP should not be made until it is cleared. 0: No operation 1: Transmits NULL packets.

14.4.1.6 UDFSMSTSET(DMAC 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	-	-	-	-	-	-	-	m_burst_type
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	mr_reset	mr_abort	mr_enable	-	mw_reset	mw_abort	mw_enable
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-9	−	R	Read as undefined.
8	m_burst_type	R/W	Selects the type of HBURST[2:0] when making a burst transfer in Master Write/Read transfers. The type of burst transfer made by UDC2AB is INCR8 (burst of 8 beat increment type). Accordingly, 0 (initial value) should be set in normal situation. However, in case INCR can only be used as the type of burst transfer based on the AHB specification of the system, set 1 to this bit. In that case, UDC2AB will make INCR transfer of 8 beat. Please note the number of beat in burst transfers cannot be changed. Setting of this bit should be made in the initial setting of UDC2AB. The setting should not be changed after the Master Write/Read transfers started. Note: UDC2AB does not make burst transfers only in Master Write/Read transfers. It combines burst transfers and single transfers. This bit affects the execution of burst transfers only. 0: INCR8 1: INCR
7	-	R	Read as undefined.
6	mr_reset	R/W	Initializes the Master Read transfer block of UDC2AB. However, as the FIFOs of EPs are not initialized, you need to access the UDFS2CMD of UDC2 to initialize the corresponding EP separately from this reset. This reset should be used after stopping the Master operation. This bit will be automatically cleared to 0 after being set to 1. Subsequent Master Read transfers should not be made until it is cleared. 0: No operation 1: Reset
5	mr_abort	W	Controls Master Read transfers. Master Read operations can be stopped by setting 1 to this bit. When aborted during transfers, transfer of buffers for Master Read to UDC2 is interrupted and the <mr_enable> bit is cleared, stopping the Master Read transfer. Aborting completes when the <mr_enable> bit is disabled to 0 after setting this bit to 1. 0: No operation 1: Abort
4	mr_enable	R/W	Controls Master Read transfers. Enabling should be made when setting the transfer address is completed. It will be automatically disabled as the master transfer finishes. Since Master Read operations cannot be disabled with this register, use the <mr_abort> bit if the Master Read transfer should be stopped. 0: Disable 1: Enable
3	−	R	Read as undefined.
2	mw_reset	R/W	Initializes the Master Write transfer block. However, as the FIFOs of EPs are not initialized, you need to access the UDFS2CMD of UDC2 to initialize the corresponding EP separately from this reset. This reset should be used after stopping the Master operation. This bit will be automatically cleared to 0 after being set to 1. Subsequent Master Write transfers should not be made until it is cleared. 0: No operation 1: Reset
1	mw_abort	W	Controls Master Write transfers. Master Write operations can be stopped by setting 1 to this bit. When aborted during transfers, transfer of buffers for Master Write from UDC2 is interrupted and the <mw_enable> bit is cleared, stopping the Master Write transfer. Aborting completes when the <mw_enable> bit is disabled to 0 after setting this bit to 1. 0:No operation 1: Abort
0	mw_enable	R/W	Controls Master Write transfers. Enabling should be made when setting the transfer address is completed. It will be automatically disabled as the master transfer finishes. Since Master Write operations cannot be disabled with this register, use the <mw_abort> bit if the Master Write transfer should be stopped. 0: Disable 1: Enable

14.4.1.7 UDFSDMACRDREQ(DMAC Read Request Register)

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

Bit	Bit Symbol	Type	Function
31	dmardreq	R/W	The bit for requesting read access to the DMAC registers. Setting this bit to 1 will make a read access to the address specified by <dmardadr>. When the read access is complete and the read value is stored in the UDFSDMACRDVL, this bit will be automatically cleared and the UDFSINTSTS<dmac_reg_rd> will be set to 1. 0: No operation 1: Issue a read request
30	dmardclr	R/W	The bit for forcibly clearing the register read access request associated with DMAC. Setting this bit to 1 will forcibly stop the register read access request by <dmardreq> and the value of <dmardreq> will be cleared to 0. After the forced clearing completes, this bit will be automatically cleared. 0: No operation 1: Issue forced clearing
29-8	−	R	Read as undefined.
7-2	dmardadr[5:0]	R/W	Sets the address of the register (upper 6 bits) to be read. It should be set in combination with the <dmardreq> mentioned above. Any one of the following addresses should be set: 0x48: Read the UDFSMWCADR. 0x58: Read the UDFSMRCADR. 0x88: read the UDFSTOUTCNT.
1-0	−	R	Read as undefined.

14.4.1.8 UDFSDMACRDVL(DMAC Read Value Register)

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

Bit	Bit Symbol	Type	Function
31-0	dmardata[31:0]	R	This register stores the data requested by UDFSDMACRDREQ. This register should not be accessed when the UDFSDMACRDREQ<dmardreq> is set to 1.

14.4.1.9 UDFSUDC2RDREQ(UDC2 Read Request Register)

	31	30	29	28	27	26	25	24
bit symbol	udc2rdreq	udc2rdclr	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	udc2rdadr	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	udc2rdadr						-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	udc2rdreq	R/W	The bit for requesting read access to the UDC2 registers. Setting this bit to 1 will make a read access to the address set in the udc2rdadr bit. When the read access is complete and the read value is set to UDFSDMACRDVL, this bit will be automatically cleared and the UDINTSTS<int_udc2_reg_rd> bit of Interrupt Status register will be set to 1. During a write access to UDC2 registers, it works as a status bit which indicates the access being made to display the value of 1. Subsequent accesses to UDC2 registers should not be made while this bit is set to 1. 0: No operation 1: Issue a read request
30	udc2rdclr	R/W	The bit for forcibly clearing the read/write access request of UDC2 registers. Setting this bit to 1 will forcibly stop the register read request/UDC2 write access by udc2rdreq and the value of udc2rdreq will be 0. After the forced clearing completes, this bit will be automatically cleared to 0. When interrupted, the read and write values during the access will not be secured. 0 : No operation 1 : Issue forced clearing
29-10	−	R	Read as undefined.
9-2	udc2rdadr[7:0]	R/W	Set the address of the UDC2 register [9:2] to be read. Refer to "14.4.1.1 UDC2AB Register list" for information about the register address of the UDC2. The offset addresses in the above register table (0x0200 to 0x0334) are releant. Set it with the above udc2rdreq.
1-0	−	R	Read as undefined.

14.4.1.10 UDFSUDC2RDVL(UDC2 Read 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	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	udc2rdata							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	udc2rdata							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15-0	udc2rdata[15:0]	R	This register stores the data requested by UDFSDMACRDREQ. This register should not be accessed when the UDFSUDC2RDREQ <udc2rdreq> is set to 1.

14.4.1.11 UDFSARBTSET(Arbiter Setting Register)

	31	30	29	28	27	26	25	24
bit symbol	abt_en	-	-	abtmod	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	abtpri_w1		-	-	abtpri_w0	
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	abtpri_r1		-	-	abtpri_r0	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31	abt_en	R/W	Enables the arbiter operation when making an access between DMAC and AHB. 0 should be set to this bit when setting the <abtmod>, <abtpri_w1>, <abtpri_w0>, <abtpri_r1> and <abtpri_r0> of this register. Be sure to set this bit to 1 before starting a DMA access. 0: Disable (DMA access not allowed) 1: Enable
30-29	−	R	Read as undefined.
28	abdmod	R/W	Sets the mode of arbiter. Write access is only available when the <abt_en> bit is set to 0. If 0 is set to this bit, access rights to the AHB bus will be given in a round-robin fashion regardless of the values set to each <abtpri_w1>, <abtpri_w0>, <abtpri_r1> and <abtpri_r0> bit. If 1 is set to this bit, access rights to the AHB bus will be given in accordance with the access priority based on the values set to each <abtpri _ w1>, <abtpri _ w0>, <abtpri _ r1> and <abtpri _ r0> bit. 0: Round-robin 1: Fixed-priority
27-14	−	R	Read as undefined.
13-12	abtpri_w1	R/W	Set the priority of DMA accesses for Master Write 1 when the fixed priority mode is selected. Write access is only available when the <abt_en> bit is set to 0. Priority ranges from 00 (highest) to 11 (lowest).
11-10	−	R	Read as undefined.
9-8	abtpri_w0	R/W	Set the priority of DMA accesses for Master Write 0 when the fixed priority mode is selected. Write access is only available when the <abt_en> bit is set to 0. Priority ranges from 00 (highest) to 11 (lowest).
7-6	−	R	Read as undefined.
5-4	abtpri_r1	R	Set the priority of DMA accesses for Master Read 1 when the fixed priority mode is selected. Write access is only available when the <abt_en> bit is set to 0. Priority ranges from 00 (highest) to 11 (lowest).
3-2	−	R	Read as undefined.
1-0	abtpri_r0	R/W	Set the priority of DMA accesses for Master Read 0 when the fixed priority mode is selected. Write access is only available when the <abt_en> bit is set to 0. Priority ranges from 00 (highest) to 11 (lowest).

Note: Be sure to set different priority values for the <abtpri_w1>, <abtpri_w0>, <abtpri_r1>, and <abtpri_r0> bits. If the same priority values are set, you will not be able to set 1 to <abt_en>.

(1) Relationship of DMAC and the priority area of the Arbiter setting register

Current UDC2AB specification supports one DMAC for Master Write (DMAC_W0) and one DMAC for Master Read (DMAC_R0). The second DMAC for Master Write (DMAC_W1) and the second DMAC for Master Read (DMAC_R1) are not supported.

Accordingly, setting priority for DMAC_W1 and DMAC_R1 has virtually no meaning, but you should be sure to set different priority values for the abtpri_w1, abtpri_w0, abtpri_r1, and abtpri_r0 bits as mentioned above.

There will be no problem to set values for the corresponding register areas of an unpackaged DMAC. The priority areas of Arbiter Setting register correspond with DMAC as shown below.

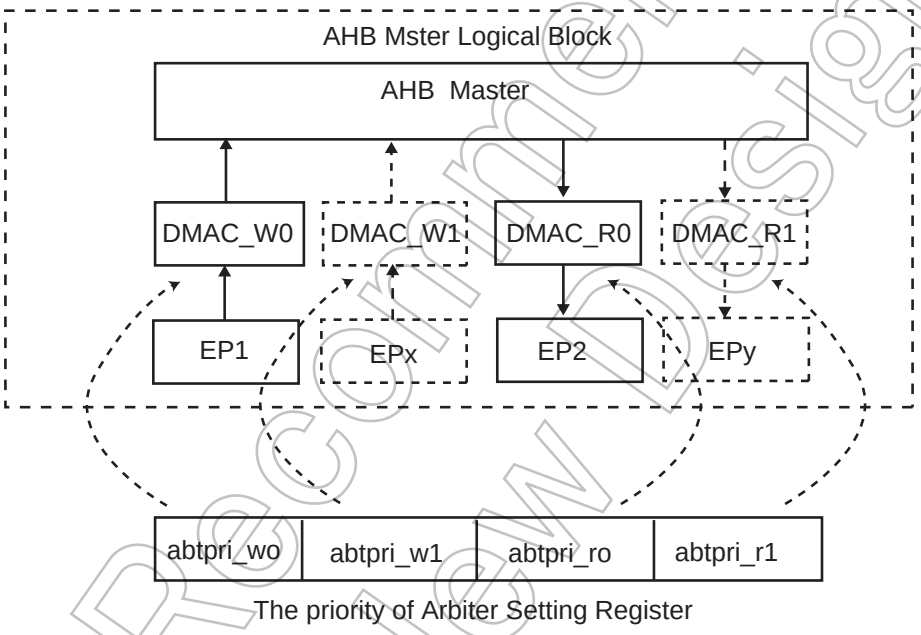


Figure 14-8 Relationship between DMAC and priority areas

14.4.1.12 UDFSMWSADR(Master Write Start Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	mwsadr[31:0]	R/W	Set the start address of Master Write transfer. However, as this master operation only supports address increments, values lower than the UDFSMWEADR should be set.

14.4.1.13 UDFSMWEADR(Master Write End Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	mweadr[31:0]	R/W	Set the end address of Master Write transfer. However, as this master only supports address increments, values above the UDFSMWSADR should be set.

14.4.1.14 UDFSMWCADR(Master Write Current Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	mwcadr[31:0]	R	Displays the addresses to which transfers from EPs to the Master Write buffers have been currently completed in Master Write transfers. This can be used in case a timeout interrupt has occurred or an error occurred during the transfer process. This address is incremented at the point when the data is set from the EP to the Master Write buffer, while the data will reside inside the target device or the Master Write buffer during the Master Write transfer process until the displayed address.

14.4.1.15 UDFSMWAHBADR(Master Write AHB Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	mrsadr[31:0]	R	Displays the address where the transfer to the target device has completed in Master Write transfer. This can be used in case a timeout interrupt has occurred or an error occurred during the transfer process. This address is incremented at the point when the data is set to the target device, while the data will reside inside the target device or during the Master Write transfer process until the displayed address.

14.4.1.16 UDFSMRSADR(Master Read Start Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	mrsadr[31:0]	R/W	Set the start address of Master Read transfer. However, as this master only supports address increments, values lower than the UDFSMWEADR should be set.

14.4.1.17 UDFSMREADR(Master Read End Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	mreadr[31:0]	R/W	Set the end address of Master Read transfer. However, as this master only supports address increments, values above the UDFSMRSADR should be set.

14.4.1.18 UDFSMRCADR(Master Read Current Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	mrcadr[31:0]	R	Displays the address to which transfers from the target device to the EP have been currently completed in Master Read transfers. This address is incremented at the point when the data is set from the Master Read buffer to the EP, while the data will reside inside the FIFO for the EP during the Master Read transfer process until the displayed address.

14.4.1.19 UDFSMRAHBADR(Master Read AHB Address Register)

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

Bit	Bit Symbol	Type	Function
31-0	mrahbadr[31:0]	R	Displays the address where the transfer from the target device to UDC2AB has completed in Master Read transfer. This address is incremented at the point when the data is set from the target device, while the data will reside inside the buffer or the FIFO for the EP during the Master Read transfer process until the displayed address.

14.4.1.20 UDFSPWCTL(Power Detect 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	wakeup_en	phy_re-mote_wkup	phy_resetb	suspend_x	phy_suspend	pw_detect	pw_resetb	usb_reset
After reset	0	0	1	1	0	0	1	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	wakeup_en	R/W	Set this bit to '1' if you want to shift the TMPM366FDXBG/FYXBG/FWXBG to low-power consumption mode to stop CLK_H when the USB is suspended. If this bit is set to '1', the $\overline{\text{WAKEUP}}$ signal will be asserted to 0 asynchronously when the suspended status (<suspend_x>=1) is cancelled. This allows TMPM366FDXBG/FYXBG/FWXBG to resuming from the low-power consumption mode using INTUSBWKUP. Refer to "14.5.7 Suspend / Resume" for more information about how to use this bit. 0: Do not assert the $\overline{\text{WAKEUP}}$ signal. 1:Assert the $\overline{\text{WAKEUP}}$ signal.
6	phy_remo-to_wkup	R/W	This bit is used to perform the remote wakeup function of USB. Setting this bit to 1 makes it possible to assert the udc2_wakeup output signal (wakeup input pin of UDC2) to 1. However, since setting this bit to 1 while no suspension is detected by UDC2 (when suspend_x = 1) will be ignored (not to be set to 1), be sure to set it only when suspension is detected. It will be automatically cleared to 0 when resuming the USB is completed (when suspend_x is deasserted). See also "1.3.24.8 Suspend / Resume" for more information on using this bit. Refer to "14.5.7 Suspend / Resume" for more information about how to use this bit. 0: No operation 1: Wakeup
5	phy_resetb	R/W	Setting this bit to 0 will make the PHYRESET output signal asserted to 1. The PHYRESET signal can be used to reset PHY. Since this bit will not be automatically released, be sure to clear it to 1 after the specified reset time of PHY. 0: Reset asserted 1: Reset deasserted
4	suspend_x	R	Detects the suspend signal (a value of the suspend_x signal from UDC2 synchronized). 0: Suspended (<suspend_x> = 0) 1: Unsuspended (<suspend_x> = 1)
3	phy_suspend	R/W	Setting this bit to 1 will make the $\overline{\text{PHYSUSPEND}}$ output signal asserted to 0 (CLK_H synchronization). It can be used as a pin for suspending PHY. Setting this bit to 1 makes the UDC2 register and UDFSDMACRDREQ not accessible. It will be automatically cleared to 0 when resumed (when suspend_x of UDC2 is deasserted). Refer to "14.5.7 Suspend / Resume" for more information about how to use this bit. 0: Not suspended. 1: Suspended
2	pw_detect	R	Indicates the status of the VBUSPOWER input of the UDC2AB. 0: USB bus disconnected (VBUSPOWER = 0) 1: USB bus connected (VBUSPOWER = 1)
1	pw_resetb	R/W	Software reset for UDC2AB(See "14.5.1 Reset" for details). Setting this bit to 0 will make the PW_RESETB output pin asserted to 0. Resetting should be made while the master operation is stopped. Since this bit will not be automatically released, be sure to clear it. 0: Reset asserted. 1: Rest deasserted.
0	usb_reset	R	The value of the usb_reset signal from the UDC2 synchronized. 0: usb_reset = 0 1: usb_reset = 1

14.4.1.21 UDFSMSTSTS(Master 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	-	-	-	mrepempty	mrbfemp	mwbfemp	mrepdset	mwepdset
After reset	0	0	0	1	1	1	0	0

Bit	Bit Symbol	Type	Function
31-5	−	R	Reset as undefined.
4	mrepempty	R	This is a register that indicates the EP for UDC2Rx is empty. Ensure that this bit is set to 1 when sending a NULL packet using the UDFSC2STSET<tx0>. (This bit is the eptx_empty input signal with CLK_H synchronization.) 0: Indicates the EP contains some data. 1: Indicates the EP is empty.
3	mrbfemp	R	Indicates whether or not the buffer for the Master Read DMA in UDC2AB is empty. 0: Indicates the buffer for the Master Read DMA contains some data. 1: Indicates the buffer for the Master Read DMA is empty.
2	mwbfemp	R	Indicates whether or not the buffer for the Master Write DMA in UDC2AB is empty. 0: Indicates the buffer for the Master Write DMA contains some data. 1: Indicates the buffer for the Master Write DMA is empty.
1	mrepdset	R	This bit will be set to 1 when the data to be transmitted is set to the Tx-EP of UDC2 by Master Read DMA transfer, making no room to write in the EP. It will turn to 0 when the data is transferred from UDC2 by the IN-Token from the host. While this bit is set to 0, DMA transfers to the EP can be made. (This bit is the eptx_dataset input signal with CLK_H synchronization.) 0: Data can be transferred into the EP. 1: There is no space to transfer data in the EP.
0	mwepdset	R	This bit will be set to 1 when the data received is set to the Rx-EP of UDC2. It will turn to 0 when the entire data was read by the DMA for Master Write. (This bit is the eprx_dataset input signal with CLK_H synchronization.) 0: No data exists in the EP. 1: There is some data to be read in the EP.

14.4.1.22 UDFSTOUTCNT(Timeout Count Register)

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

Bit	Bit Symbol	Type	Function
31-0	tmoutcnt[31:0]	R	This is used for debugging. Values of the timer can be read when the UDFSMWTOUT<timeout_en> is enabled. It will be decremented each time CLK_U is counted after the EP for Master Write (Rx-EP) becomes empty. This register cannot be read by directly specifying the address. In order to read it, set a value to the UDFSDMACRDREQ and then read the value from UDFSDMACRDVL.

14.4.2 UDC2 Register

14.4.2.1 UDC2 Registers

BaseAddress=0x4000_8000

Register name		Address(Base+)
UDC Address-State Register	UDFS2ADR	0x0200
UDC2 Frame Register	UDFS2FRM	0x0204
Reserved	-	0x0208
UDC2 Command Register	UDFS2CMD	0x020C
UDC2 bRequest-bmRequest Type Register	UDFS2BRQ	0x0210
UDC2 wValue register	UDFS2WVL	0x0214
UDC2 wIndex Register	UDFS2WIDX	0x0218
UDC2 wLength Register	UDFS2WLGTH	0x021C
UDC2 INT Register	UDFS2INT	0x0220
UDC2 INT EP Register	UDFS2INTEP	0x0224
UDC2 INT EP Mask Register	UDFS2INTEPMSK	0x0228
UDC2 INT RX DATA0 Register	UDFS2INTRX0	0x022C
UDC2 EP0 MaxPacketSize Register	UDFS2EP0MSZ	0x0230
UDC2 EP0 Status Register	UDFS2EP0STS	0x0234
UDC2 EP0 Datasize Register	UDFS2EP0DSZ	0x0238
UDC2 EP0 FIFO Register	UDFS2EP0FIFO	0x023C
UDC2 EP1 MaxPacketSize Register	UDFS2EP1MSZ	0x0240
UDC2 EP1 Status Register	UDFS2EP1STS	0x0244
UDC2 EP1 Datasize Register	UDFS2EP1DSZ	0x0248
UDC2 EP1 FIFO Register	UDFS2EP1FIFO	0x024C

BaseAddress=0x4000_8000

Register name		Address(Base+)
UDC2 EP2 MaxPacketSize Register	UDFS2EP2MSZ	0x0250
UDC2 EP2 Status Register	UDFS2EP2STS	0x0254
UDC2 EP2 Datasize Register	UDFS2EP2DSZ	0x0258
UDC2 EP2 FIFO Register	UDFS2EP2FIFO	0x025C
UDC2 EP3 MaxPacketSize Register	UDFS2EP3MSZ	0x0260
UDC2 EP3 Status Register	UDFS2EP3STS	0x0264
UDC2 EP3 Datasize Register	UDFS2EP3DSZ	0x0268
UDC2 EP3 FIFO Register	UDFS2EP3FIFO	0x026C
UDC2 EP4 MaxPacketSize Register	UDFS2EP4MSZ	0x0270
UDC2 EP4 Status Register	UDFS2EP4STS	0x0274
UDC2 EP4 Datasize Register	UDFS2EP4DSZ	0x0278
UDC2 EP4 FIFO Register	UDFS2EP4FIFO	0x027C
UDC2 EP5 MaxPacketSize Register	UDFS2EP5MSZ	0x0280
UDC2 EP5 Status Register	UDFS2EP5STS	0x0284
UDC2 EP5 Datasize Register	UDFS2EP5DSZ	0x0288
UDC2 EP5 FIFO Register	UDFS2EP5FIFO	0x028C
UDC2 EP6 MaxPacketSize Register	UDFS2EP6MSZ	0x0290
UDC2 EP6 Status Register	UDFS2EP6STS	0x0294
UDC2 EP6 Datasize Register	UDFS2EP6DSZ	0x0298
UDC2 EP6 FIFO Register	UDFS2EP6FIFO	0x029C
UDC2 EP7 MaxPacketSize Register	UDFS2EP7MSZ	0x02A0
UDC2 EP7 Status Register	UDFS2EP7STS	0x02A4
UDC2 EP7 Datasize Register	UDFS2EP7DSZ	0x02A8
UDC2 EP7 FIFO Register	UDFS2EP7FIFO	0x02AC
Reserved	-	0x02B0 to 0x32C
UDC2 INT NAK Register	UDFS2INTNAK	0x0330
UDC2 INT NAK MASK Register	UDFS2INTNAKMSK	0x0334
Reserved	-	0x0338 to 0x03FC

Note 1: Those shown as "Reserved" above and the area from 0x0400 to 0x0FFF are prohibited to access. Read / Write is prohibited to these areas.
Note 2: The registers are initialized by reset_x or USB_reset.

14.4.2.2 How to access the UDC2 register

The bits 15-0 of the AHB data bus of UDC2AB are connected to the UDC data bus.

The bit31-16 are read-only (read value: undefined).

Make a WORD (32-bit) access for both write and read. (However, a BYTE (8-bit) access may be made for Write accesses to the EPx_FIFO register. Details will be described later).

It will take some time to complete an access for both write and read.

Be sure to begin subsequent accesses after the previous UDC2 register access is completed, using the int_udc2_reg_rd interrupt. (You can also use the UDFSUDC2RDREQ<udc2rdreq> to confirm the access status when reading.)

- Write access

  When making a write access to the UDC2 register, write it directly in the relevant address.
- Read access

  When making a read access to the UDC2 register, use UDFSUDC2RDREQ and DFSUDC2RDVL.

  First, you set the address to access to the UDFSUDC2RDREQ and then read the data from the UDFSUCS2RDVL for reading. You cannot read the data directly from the address shown in the "14.4.2.1 UDC2 Registers".
- EPx_FIFO register

  When making a write access to the EPx_FIFO register, a lower 1-byte access may be required in UDC2 PVCI I/F. In such a case, make a BYTE access to the lower 1 byte for UDC2AB.

  If a lower 1-byte access is required when making a read access, make an access via UDFSUDC2RDREQ as usual and read the data from UDFSUDC2RDVL. In that case, the access to UDFSAUDC2RDVL can be either by WORD or BYTE.
- Reserved Register in UDC2

  Do not make any access to registers of EPs not supported by UDC2 to be connected and to "Reserved" registers. (In case those registers are accessed, the access from UDC2AB to UDC2 itself will take place. It will be a Dummy write to UDC2 in case of write accesses. In case of read accesses, the read data from UDC2 (udc2_rdata) will be an indefinite value and the indefinite value will be set to the UDFSUDC2RDVL.)
- Accesses when UDC2 is suspended

  When UDC2 is in the suspended status, register accesses to UDC2 become unavailable if the clock (= CLK_U) supply from clock/mode control circuit is stopped. Make no register accesses to UDC2 in such cases. If the UDC2 register is accessed when the UDFSPWCTL<phy_suspend> is set to 1, an AHB error will be returned.

Access flow diagram for UDC2 register is shown below.

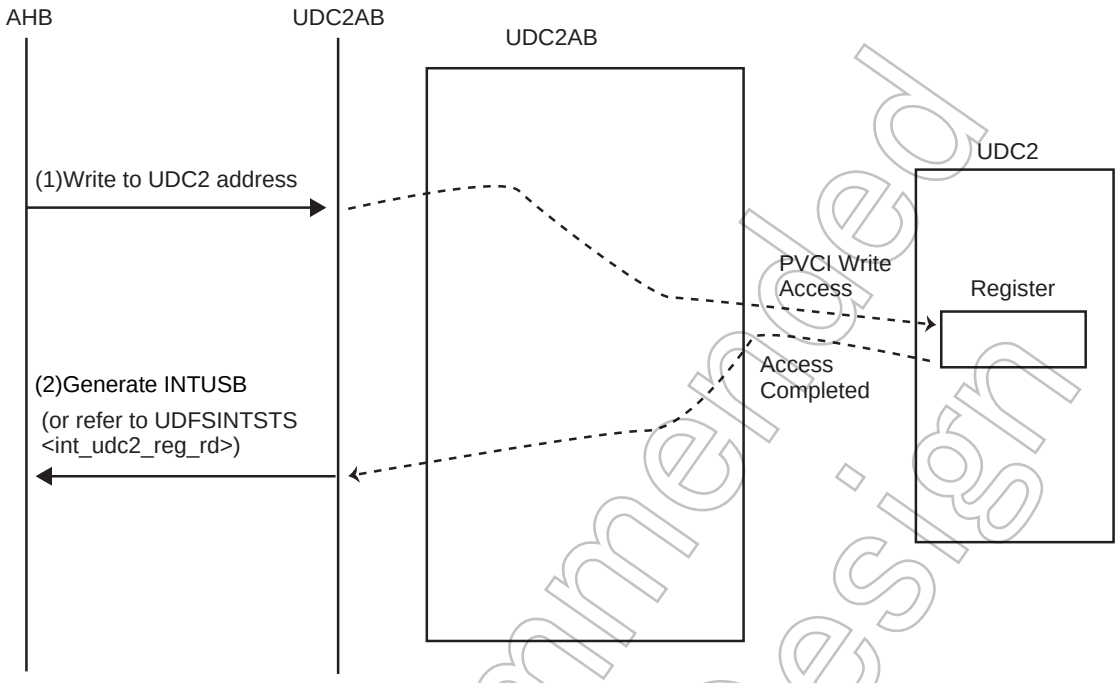


Figure 14-9 Write Access Flow Diagram of the UDC2 Register

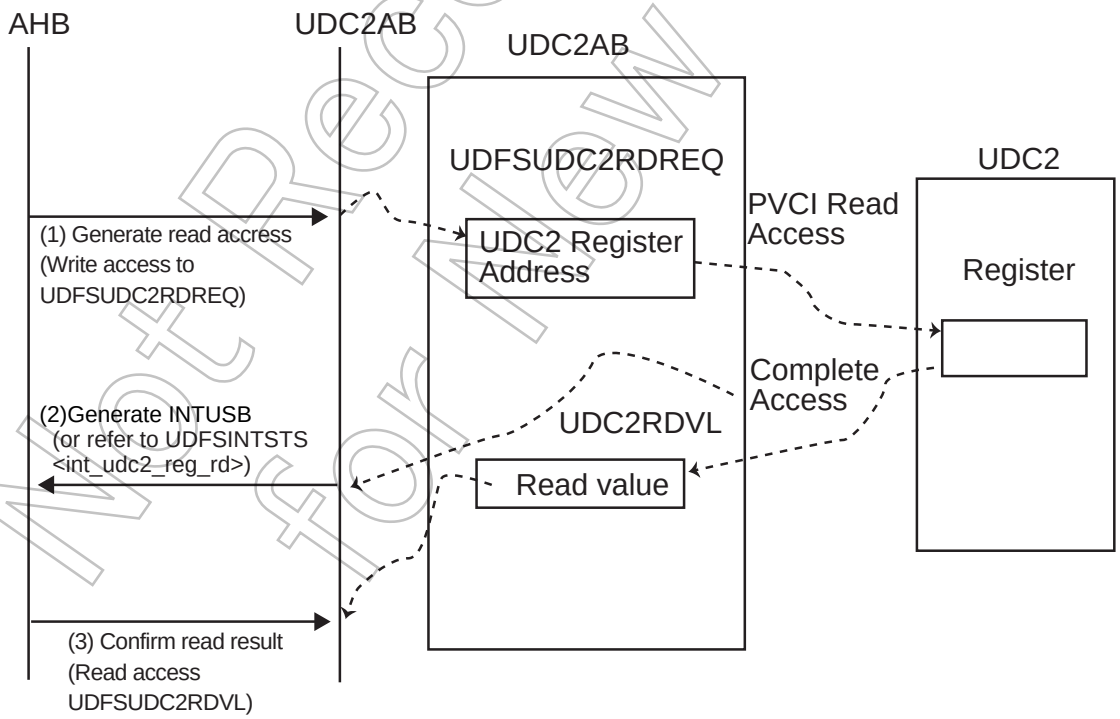


Figure 14-10 Read Access Flow Diagram of the UDC2 Register

14.4.2.3 UDFS2ADR(Address-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	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	stage_err	ep_bi_mode	cur_speed		suspend	configured	addressed	default
After reset	0	0	0	0	0	0	0	1
	7	6	5	4	3	2	1	0
bit symbol	-	dev_adr						
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	–	R	Read as undefined.
15	stage_err	R/W	Indicates whether Control transfers finished normally up to the STATUS-Stage. 1 will be set when the Setup-Token is received in DATA-Stage/STATUS-Stage or in the case of "STALL" transmission. When set, it will be cleared when the next Control transfer has been finished normally. 0: Other than above conditions. 1: Received the Setup-Token in DATA-Stage/STATUS-Stage or "STALL" transmission.
14	ep_bi_mode	R/W	Selects whether to use the EP bidirectionally as a driver. Setting this bit to 1 will enable an EP number to be used bidirectionally in USB communication. 0: Single direction 1: Dual direction
13-12	cur_speed[1:0]	R	Indicates the present transfer mode on the USB bus. 00: Reserved 01: Full-Speed 10: Reserved 11: Reserved
11	suspend	R	Indicates whether or not UDC2 is in suspended state. 0: Normal 1: Suspend
10	configured	R/W	Set the present device state of UDC2. This should be set in accordance with the request received from the host. Please note that you should not set 1 to more than one bit. 001: default (to be set when the DeviceAddress = 0 was specified by the Set_address request in Default / Address state (this will be set by the hardware when USB_RESET is received)) 010: addressed (to be set when ConfigurationValue = 0 was specified by the Set_configuration request after the Set_address request finished normally and in the Address/Configured state). 100: configured (to be set when the Set_configuration request is received).
9	addressed	R/W	
8	default	R/W	
7	–	R	Read as undefined.
6-0	dev_adr[6:0]	R/W	Sets the device address assigned by the host. The device address should be set after Set_address has finished normally (after STATUS-Stage finished normally).

14.4.2.4 UDFS2FRM(Frame 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	create_sof	-	f_status		-	frame		
After reset	0	0	1	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	frame							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15	create_sof	R/W	Sets whether to generate the SOF flag internally when the SOF from the host is unavailable due to a bus error. This should be set if you wish to synchronize frames by SOF in Isochronous transfers. If enabled, the internal frame time counter will operate and the SOF flag will be output even when the SOF-Token could not be received successfully. 0: Generates no flag 1: Generates a flag
14	−	R	read as undefined.
13-12	f_status[1:0]	R	Indicates the status of the frame number. 00: Before: Will be set if the Micro SOF/SOF was not received when 1frame-time (Full-Speed:1 ms) has passed after receiving the Micro SOF/SOF when <create_sof> is enabled. In the Frame register, the frame number received in the last Micro SOF/SOF has been set. 01: Valid: Will be set when the Micro SOF/SOF was received. Indicates a valid frame number is set in the Frame register. 10: Lost: Indicates that the frame number maintained by the host is not synchronized with the value of UDFS2FRM. Accordingly, this will be set in the following cases: 1. When the system was reset or suspended. 2. If the next Micro SOF/SOF was not received when 2 frame-time (Full-Speed: 1 x 2 ms) has passed after receiving the previous Micro SOF/SOF when <create_sof> is enabled. Also note that transition to the Lost status only happens after the system was reset or when it is suspended if <create_sof> is disabled.
11	−	R	Read as undefined.
10-0	frame[10:0]	R	Indicates the frame number when SOF is received. This will be valid when <f_status> is "Valid". Should not be used if <f_status> is "Before" or "Lost" as correct values are not set.

14.4.2.5 UDFS2CMD(Command 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	int_toggle	-	-	-	rx_nulpkt_ep			
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ep				com			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15	int_toggle	R/W	Makes the DATA-PID toggle when Handshake is not received in Interrupt-IN transfers. 0: Do not toggle when not received 1: Toggle when not received as well
14-12	−	R	Read as undefined.
11-8	rx_nulpkt_ep [3:0]	R	Indicates the receiving EP when Zero-Length data is received. When the INT_RX_ZERO flag is asserted, read this bit to check to which EP it was asserted. Once Zero-Length data is received and the EP number is retained, the value of this register will be retained until Zero-Length data is received next time or hardware reset is made. If there is more than one EP of OUT direction, this bit will be renewed each time Zero-Length data is received. In that case, UDFS2INTRX0 can be used to identify which EP has received the data.
7-4	ep[3:0]	R/W	Sets the EP where the command to be issued will be valid. (Do not specify an EP not existing.)
3-0	com[3:0]	R/W	Sets the command to be issued for the EP selected in ep[3:0]. Refer to "14.2.2.3 Commands to EP" for more information. 0x0: Reserved 0x1: Setup_Fin 0x2: Set_DATA0 0x3: EP_Reset 0x4: EP_Stall 0x5: EP_Invalid 0x6: Reserved 0x7: EP_Disable 0x8: EP_Enable 0x9: All_EP_Invalid 0xA: USB_Ready 0xB: Setup_Received 0xC: EP_EOP 0xD: EP_FIFO_Clear 0xE: EP_TX_0DATA 0xF: Reserved

14.4.2.6 UDFS2BRQ(bRequest-bmRequest Type 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	request							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	dir	req_type		recipient				
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined
15-8	request[7:0]	R	Indicates the data of the second byte received with the Setup-Token (bRequest field).
7	dir	R	Indicates the data of the first byte received with the Setup-Token (bmRequestType field). Direction of Control transfers. 0: Control-WR transfer 1: Control-RD transfer
6-5	req_type[1:0]	R	Type of requests 00: Standard request 01: Class request 10: Vendor request 11: Reserved
4-0	recipient[4:0]	R	Requests are received by 0_0000: Device 0_0001: Interface 0_0010: EP 0_0011: etc. 0_0100-1_1111: Reserved

14.4.2.7 UDFS2WVL(wValue 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	value							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	value							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15-8	value[15:8]	R	Indicates the data of the fourth byte received with the Setup-Token (wValue (High) field).
7-0	value[7:0]	R	Indicates the data of the third byte received with the Setup-Token (wValue (Low) field).

14.4.2.8 UDFS2WIDX(wIndex 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	index							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	index							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined
15-8	index[15:8]	R	Indicates the data of the sixth byte received with the Setup-Token (wIndex (High) field).
7-0	index[7:0]	R	Indicates the data of the fifth byte received with the Setup-Token (wIndex (Low) field).

14.4.2.9 UDFS2WLGTH(wLength 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	length							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	length							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined
15-8	length[15:8]	R	Indicates the data of the eighth byte received with the Setup-Token (wLength (High) field).
7-0	length[7:0]	R	Indicates the data of the seventh byte received with the Setup-Token (wLength (Low) field).

14.4.2.10 UDFS2INT(INT 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	m_nak	m_ep	m_ep0	m_sof	m_rx_data0	m_status	m_status_nak	m_setup
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	i_nak	i_ep	i_ep0	i_sof	i_rx_data0	i_status	i_status_nak	i_setup
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15	m_nak	R/W	Sets whether or not to output <i_nak> to the INT_NAK pin. 0: output 1: no output
14	m_ep	R/W	Sets whether or not to output <i_ep> to INT_EP pin. 0: output 1: no output
13	m_ep0	R/W	Sets whether or not to output <i_ep0> to INT_EP0 pin. 0: output 1: no output
12	m_sof	R/W	Sets whether or not to output <i_sof> to INT_SOF pin. 0: output 1: no output
11	m_rx_data0	R/W	Sets whether or not to output <i_rx_data0> to INT_RX_ZERO pin. 0: output 1: no output
10	m_status	R/W	Sets whether or not to output <i_status> to INT_STATUS pin. 0: output 1: no output
9	m_status_nak	R/W	Sets whether or not to output <i_status_nak> to INT_STATUS_NAK pin. 0: output 1: no output
8	m_setup	R/W	Sets whether or not to output <i_setup> to INT_SETUP pin. 0: output 1: no output
7	i_nak	R/W	This will be set to 1 when NAK is transmitted by EPs except EP0. (EPs to which you wish to output the INT_NAK flag can be selected using UDFS2INTNAKMASK). Writing 1 to this bit will make each bit of UDFS2INTNAK cleared to 0.
6	i_ep	R/W	This will be set to 1 when transfers to EPs other than EP0 have successfully finished (EPs to which you wish to output the flag can be selected using UDFS2INTNAKMASK). Writing 1 to this bit will make each bit of UDFS2INTEP cleared to 0.
5	i_ep0	R/W	This will be set to 1 when the transfer to EP0 has successfully finished.
4	i_sof	R/W	This will be set to 1 when the SOF-token is received or after 1 frame-time was counted in the create_sof mode.
3	i_rx_data0	R/W	This will be set to 1 when Zero-Length data is received. (EPs to which you wish to output the flag can be selected using UDFS2INTNAKMASK). Writing 1 to this bit will make each bit of UDFSINTTRX0 cleared to 0. This will not be set to 1 when Zero-Length data is received in the STATUS-Stage of Control-RD transfers.

Bit	Bit Symbol	Type	Function
2	i_status	R/W	This will be set to 1 when the STATUS-Stage has successfully finished in Control transfers at EP0. (This will be set to 1 when Zero-Length data is received in the STATUS-Stage and successfully finished in Control-RD transfers, and when Zero-Length data is transmitted in the STATUS-Stage and successfully finished in Control-WR transfers.)
1	i_status_nak	R/W	This will be set to 1 when the packet of STATUS-Stage is received in the Control-RD transfers at EP0. When this bit was set which means the DATA-Stage has finished, set the "Setup-Fin" command by the UDFS2CMD to make the stage of UDC2 proceed to the STATUS-Stage. When receiving the data having the size of an integral multiple of MaxPacketSize (64 bytes) in the DATA-Stage of Control-WR transfers, Zero-Length data may be received to indicate the end of the DATA-Stage. After that, as the end of the DATA-Stage can be recognized by this i_status_nak when receiving the In-token in the STATUS-Stage, make UDC2 proceed to the STATUS-Stage.
0	i_setup	R/W	This will be set to 1 when the Setup-Token was received in Control transfers at EP0.

14.4.2.11 UDFS2INTEP(INT_EP 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	i_ep7	i_ep6	i_ep5	i_ep4	i_ep3	i_ep2	i_ep1	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15-8	Reserved	R/W	Write as "0".
7-1	i_ep7 to i_ep1	R/W	Flags to indicate the transmitting/receiving status of EPs (except for EP0). The relevant bit will be set to 1 when the transfer to EPs other than EP0 has successfully finished. (EPs to which you wish to output the int_ep flag can be selected using UDFS2INTEPMSK.). 0: No data transmitted/received. 1: Some data transmitted/received
0	−	R/W	Read as undefined.

14.4.2.12 UDFS2INTEPMSK(INT_EP_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	m_ep7	m_ep6	m_ep5	m_ep4	m_ep3	m_ep2	m_ep1	m_ep0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15-8	Reserved	R/W	Write as "0".
7-0	m_ep7 to m_ep0	R/W	Mask control of flag output. 0 : output 1: no output Sets whether or not to output flags of UDFS2INTEP and UDFS2INTRX0 to the int_ep pin and the int_rx_zero pin respectively. When an EP is masked, each bit of UDFS2INTEP will be set when the transfer of the relevant EP has successfully finished, but the int_ep pin will not be asserted. Similarly, when an EP is masked, each bit of UDFS2INTRX0 will be set when Zero-Length data is received at the relevant EP, but the int_rx_zero pin will not be asserted. However, bit 0 is only valid for UDFS2INTRX0.

(1) How to use UDFS2INT / UDFS2INTEP / UDFS2INTEPMSK

An example of using UDFS2INT / UDFS2INTEP / UDFS2INTEPMSK is provided for EP1 to 3.

1. When using EP 1 and EP 2 with DMA (EP I/F) and using only EP3 via PVCI-I/F

UDFS2INT	<i_ep>	Used as the interrupt source of EP3. This bit is also used when clearing.
	<m_ep>	Used as the mask of the interrupt source of EP3.
UDFS2INTEP	<i_ep1>	Don't care
	<i_ep2>	Don't care
	<i_ep3>	Don't care
UDFS2IN-TEPMSK	<m_ep1>	Set 1 to mask the bit.
	<m_ep2>	Set 1 to mask the bit.
	<m_ep3>	Write 0.

2. When using EP2 and EP3 via PVCI-I/F and using EP1 with DMA

After initialization, set 1 to UDFS2INTEPMSK of the EP to be used with DMA to mask it. When making interrupt responses for more than one EPs, be sure to use UDFS2INTEP. Ignore UDFS2INT<i_ep> and always enable <m_ep> as 0.

Do not clear the source using UDFS2INT<i_ep>. After the interrupt has occurred, you need to check UDFS2INT and UDFS2INTEP to determine the source. When clearing the source, use each source bit of UDFS2INT interrupt to clear it.

UDFS2INT	<i_ep>	Write as 0.
	<m_ep>	Write as 0.
UDFS2INTEP	<i_ep1>	Don't care
	<i_ep2>	Used as the interrupt source of EP2. This bit is also used when clearing.
	<i_ep3>	Used as the interrupt source of EP3. This bit is also used when clearing.
UDFS2IN-TEPMSK	<m_ep1>	Set 1 to mask the bit.
	<m_ep2>	Used as the mask of the interrupt source of EP2. Write as "0".
	<m_ep3>	Used as the mask of the interrupt source of EP3. Write as "0".

14.4.2.13 UDFS2INTRX0(INT_RX_DATA0 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	rx_d0_ep7	rx_d0_ep6	rx_d0_ep5	rx_d0_ep4	rx_d0_ep3	rx_d0_ep2	rx_d0_ep1	rx_d0_ep0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15-8	Reserved	R/W	Write as "0".
7-0	rx_d0_ep7 to rx_d0_ep0	R/W	Flags for indicating Zero-Length data received at EP 0:No Zero-Length data received 1:Zero-Length data received The relevant bit will be set to 1 when EPs have received Zero-Length data. (EPs to which you wish to output the int_rx_zero flag can be selected using UDFS2INTEPMSK) For bit 0 (EP 0), it will be set to 1 only when Zero-Length data is received in the DATA-Stage while processing the request. Since it will not be set when Zero-Length data is received in the STATUS-Stage, use the int_status flag.

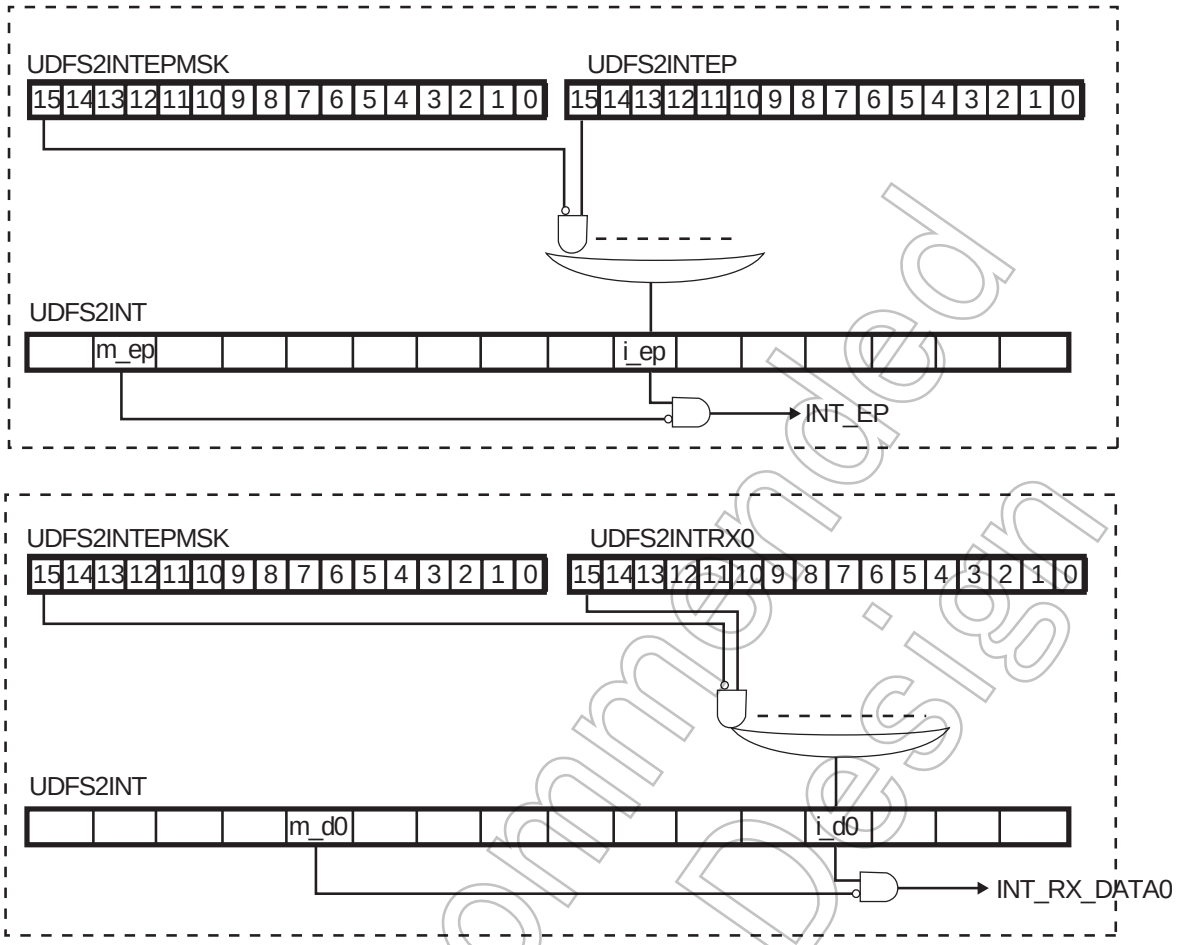


Figure 14-11 Interrupt Status and Mask Register

14.4.2.14 UDFS2INTNAK(INT_NAK 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	i_ep7	i_ep6	i_ep5	i_ep4	i_ep3	i_ep2	i_ep1	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15-8	Reserved	R/W	Write as "0",
7-1	i_ep7 to i_ep1	R/W	Flags to indicate the status of transmitting NAK at EPs (except for EP0) 0: No NAK transmitted 1: NAK transmitted The relevant bit will be set to 1 when NAK is transmitted by EPs other than EP0. (EPs to which you wish to output the INT_NAK flag can be selected using UDFS2INTEPMSK.)
0	−	R	Read as undefined.

14.4.2.15 UDFS2INTNAKMSK(INT_NAK_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	m_ep7	m_ep6	m_ep5	m_ep4	m_ep3	m_ep2	m_ep1	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15-8	Reserved	R/W	Write as "0".
7-1	m_ep7 to m_ep1	R/W	Mask control of flag output 0: output 1: no output Sets whether or not to output flags of UDFS2INTNAK to the int_nak pin respectively. When EPs are masked, each bit of UDFS2INTNAK will be set when NAK is transmitted in the transfer of the relevant EP, but the int_nak pin will not be asserted.
0	−	R	Read as undefined.

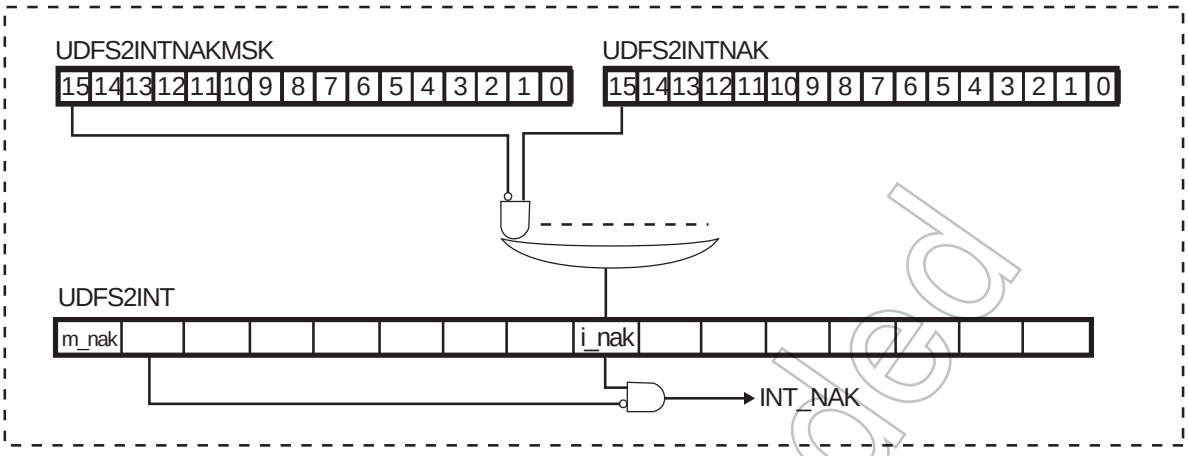


Figure 14-12 Interrupt and Status Register

14.4.2.16 UDFS2EP0MSZ(EP0_MaxPacketSize 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	tx_0data	-	-	dset	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	max_pkt						
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as defined.
15	tx_0data	R	When the "EP_TX_0DATA" command is issued to EP0 by UDFS2CMD, this bit will be set to 1 which will be cleared to 0 after the Zero-Length data has been transmitted.
14-13	−	R	Read as defined.
12	dset	R	Indicates the status of UDFS2EP0FIFO. It will be cleared to 0 when the Setup-Token is received. 0: No valid data exists 1: Valid data exists
11-7	−	R	Read as "0".
6-0	max_pkt[6:0]	R/W	Sets MaxPacketSize of EP0.

14.4.2.17 UDFS2EP0STS(EP0_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	ep0_mask	-	toggle		status			-
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
31-16	−	R	Read as undefined.
15	ep0_mask	R	Will be set to 1 after the Setup-Token is received. Will be cleared to 0 when the "Setup_Received" command is issued. No data will be written into the UDFS2EP0FIFO while this bit is 1. 0: Data can be written into UDFS2EP0FIFO. 1: Data can not be written into UDFS2EP0FIFO.
14	−	R	Read as undefined.
13-12	toggle[1:0]	R	Indicates the present toggle value of EP. 00: DATA0 01: DATA1 10: Reserved 11: Reserved
11-9	status[2:0]	R	Indicates the present status of EP0. It will be cleared to "Ready" when the Setup-Token is received. 000: Ready (Indicates the status is normal) 001: Busy (To be set when returned "NAK" in the STATUS-Stage) 010: Error (To be set in case of CRC error in the received data, as well as when timeout has occurred after transmission of the data) 011: Stall (Returns "STALL" when data longer than the Length was requested in Control-RD transfers) 100 to 111: Reserved
8-0	−	R	Read as undefined.

14.4.2.18 UDFS2EP0DSZ(EP0_Datasize 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	-	size						
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as undefined.
6-0	size[6:0]	R	Indicates the number of valid data bytes stored in UDFS2EP0FIFO. It will be cleared to when the Setup-Token is received.

14.4.2.19 UDFS2EP0FIFO(EP0_FIFO 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	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	−	R	Read as undefined.
15-0	data[15:0]	R/W	Used for accessing data from PVCI-I/F to EP0. For the method of accessing this register, see "14.7.1.1 Control-RD transfer" , "14.7.1.2 Control-WR transfer (without DATA-Stage)" and "14.7.1.3 Control-WR transfer (with DATA-Stage)". The data stored in this register will be cleared when the request is received (when the INT_SETUP interrupt is asserted).

14.4.2.20 UDFS2EPxMSZ(EPx_MaxPacketSizeRegister)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	tx_0data	-	-	dset (note1)	-	max_pkt		
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	max_pkt							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15	tx_0data	R	When the "EPx_TX_0DATA" command is issued to EPx by UDFS2CMD or Zero-Length data has been set at EP-I/F, this bit will be set to 1. It will be cleared to 0 after the Zero-Length data has been transmitted.
14-13	−	R	Read as undefined.
12	dset	R	Indicates the status of EPx_FIFO. 0: No valid data exists 1: Valid data exists
11	−	R	Read as undefined.
10-0	max_pkt[10:0]	R/W	Sets MaxPacketSize of EPx. Set this when configuring the EP when Set_Configuration and Set_Interface are received. Set an even number for a transmit EP. On USB, when MaxPacketSize of a transmit EP is an odd number, set an even number to max_pkt and make the odd number of accesses to the EP. (For instance, set 1024 to max_pkt when the MaxPacketSize should be 1023 bytes.) Note: For details, refer to "14.9.2 Appendix B About Setting an Odd Number of Bytes as MaxPacketSize".

Note 1: The initial value of <dset> after reset is 1 when the EPx is a transmit EP, while it is 0 when the EPx is a receive EP.

Note 2: The initial value of <dset > after USB_RESET is 1 when the EPx is a transmit EP, while it is "Retain" when the EPx is a receive EP.

Note 3: x=1 to 7

14.4.2.21 UDFS2EPxSTS(EPx_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	pkt_mode	bus_sel	toggle		status			disable
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	dir	-	-	-	t_type		num_mf	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as undefined.
15	pkt_mode	R/W	Selects the packet mode of EPx. Selecting the Dual mode makes it possible to retain two pieces of packet data for the EPx. 0: Single mode 1: Dual mode
14	bus_sel	R/W	Select the bus to access to the FIFO of EPx. 0: Common bus access 1: Direct access
13-12	toggle[1:0]	R	Indicates the present toggle value of EPx. 00: DATA0 01: DATA1 10: DATA2 11: MDATA
11-9	status[2:0]	R	Indicates the present status of EPx. By issuing EP_Reset from UDFSCMD, the status will be "Ready." 000: Ready (Indicates the status is normal) 001: Reserved 010: Error (To be set in case a receive error occurred in the data packet, or when timeout has occurred after transmission. However, it will not be set when "Stall" or "Invalid" has been set.) 011: Stall (To be set when "EP-Stall" was issued by UDFS2CMD.) 100 to 110: Reserved 111: Invalid (Indicates this EP is invalid)
8	disable	R	Indicates whether transfers are allowed for EPx. If "Not Allowed," "NAK" will be always returned for the Token sent to this EP. 0: Allowed 1: Not Allowed
7	dir	R/W	Sets the direction of transfers for this EP. 0: OUT (Host-to-device) 1: IN (Device-to-host)
6-4	−	R	Read as undefined.
3-2	t_type[1:0]	R/W	Sets the transfer mode for this EP. 00: Control 01: Isochronous 10: Bulk 11: Interrupt
1-0	num_mf[1:0]	R/W	When the Isochronous transfer is selected, set how many times the transfer should be made in the frames. 00: 1-transaction 01: 2-transaction 10: 3-transaction 11: Reserved

Note 1: Setting for this register should be made when configuring the EP when Set_Configuration and Set_Interface are received.

Note 2: x=1 to 7

Note 3: Each EP depend on the product specification. For EP1, EP3, EP5, EP7 which is fixed for IN transfers, <dir> can be set to "1" only. For EP2, EP4, EP6 which is fixed for OUT transfers, dir can be set to "0" only.

14.4.2.22 UDFS2EPxDSZ(EPx_Datasize 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	size							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-11	–	R	Read as undefined.
10-0	size[10:0]	R	Indicates the number of valid data bytes stored in EP1_FIFO. In the Dual Packet mode, the number of data bytes to be accessed first will be shown.

Note: x=1 to 7

14.4.2.23 UDFS2EPxFIFO(EPx_FIFO 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	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	−	R	0.
15-0	data[15:0]	R/W	Used for accessing data from PVCI-I/F to EPx.

Note: x=1 to 7

14.5 Description of UDC2AB operation

14.5.1 Reset

UDC2AB supports software reset by the UDFSPWCTL<pw_resetb>.

It also supports master channel reset (UDFSMSTSET<mr_reset>/<mw_reset>) for DMAC master transfers.

- Software reset (UDFSPWCTL<pw_resetb>)

  Some bits of each register are initialized by hardware reset but not initialized by software reset with the values retained. As details are provided in the descriptions of each register, refer to "14.4.1.1 UDC2AB Register list".

  When the USB bus power is detected, make software reset as initialization is needed.

- Master channel reset (UDFSMSTSET<mr_reset><mw_reset>

  While the <mw_reset> bit is provided for the Master Write transfer block and the <mr_reset> bit for the Master Read transfer block, only the relevant master blocks are initialized and the UDC2AB register will not be initialized. For more information on using each reset, see "14.4.1.6 UDFSMSTSET(DMAC Setting Register)".

14.5.2 Interrupt Signals

There are two interrupt output signals of UDC2AB, INTUSB and INTUSBWKUP.

14.5.2.1 INTUSB Interrupt Signal

Interrupt output signal of INTUSB consists of interrupts generated by UDC2 and that generated by other sources.

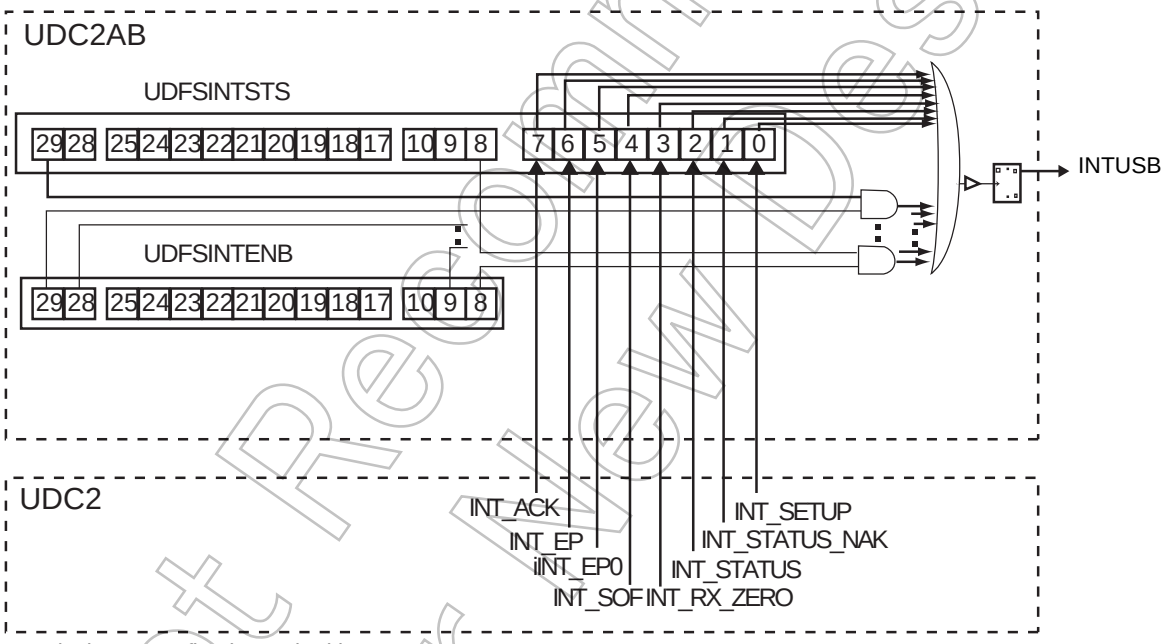
Once the interrupt condition is met, UDC2AB sets the corresponding bit of its UDFSINTSTS. When that bit is set, INTUSB will be asserted if the relevant bit of UDFSINTENB has been set to "Enable."

When the relevant bit of UDFSINTENB has been set to "Disable," 1 will be set to the corresponding bit of UDFSINTSTS while INTUSB will not be asserted.

When the relevant bit of UDFSINTENB is set to "Enable" with UDFSINTSTS set, INTUSB will be asserted immediately after the setting is made.

Initial values for UDFSINTENB are all 0 (Disable).

Interrupt output signal of INTUSB will not be generated while CLK_H is stopped.



Note) The UDC2 flag is masked by UDFS2INT.

Figure 14-13 Relationship of INTUSB and registers

14.5.2.2 INTUSBWKUP Interrupt

INTUSBWKUP interrupt occurs at the falling edge of the $\overline{\text{WAKEUP}}$ output signal.

WAKEUP will be asserted when the following conditions match:UDFSPWCTL<wakeup_en> is 1 and the suspended condition is cancelled (UDFSPWCTL<suspend_x>=1). WAKEUP will be asserted when VBUS is disconnected (VBUSPOWER=0) as well.

INTUSBWKUP interrupt occurs regardless of the status of CLK_H.

14.5.3 Operation Sequence

The operation sequence of UDC2AB is as follows:

1. Hardware reset
2. Set the interrupt signal

   Configure the INTUSB interrupt, the INTUSBWKUP interrupt and the USBPON interrupt.
3. VBUS detection (connect) and reset

   Refer to "14.5.5.2 Sequence of USB Bus Power (VBUS) Connection/Disconnection" and "14.5.1 Reset" for details.
4. USB enumeration response

   Refer to "14.6 USB Device Response" for details.
5. Master Read / Master Write transfer
   a. Master Read transfer

      Make a Master Read transfer corresponding to the receiving request from the USB host. Refer to "14.5.4.1 Master Read transfer" for details.
   b. Master Write transfer

      Make a Master Write transfer corresponding to the sending request from the USB host. Refer to "14.5.4.2 Master Write transfer" for details.
6. VBUS detection (disconnect)

   The USB bus power supply may be disconnected at any time.

   Refer to "14.5.5.2 Sequence of USB Bus Power (VBUS) Connection/Disconnection" for details.

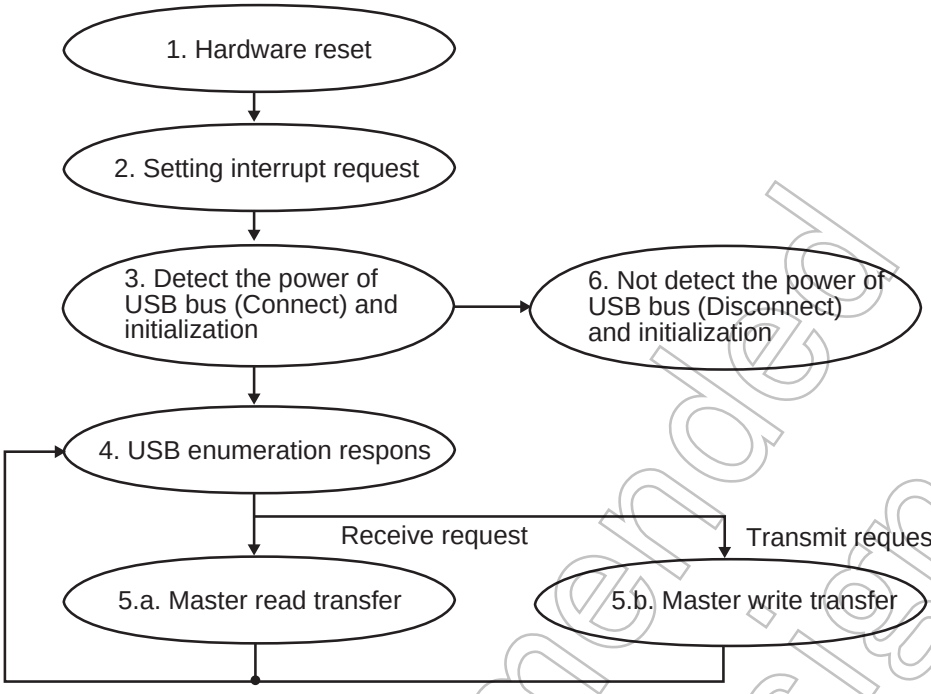


Figure 14-14 Operation Sequence

14.5.4 Master Transfer Operation

This section describes the master transfer operation of UDC2AB.

When you start a master transfer, be sure to set the transfer setting of the relevant EP of UDC2 (UDFS2EPxSTS<bus_sel>) to the direct access mode. It is prohibited to start DMAC when it is set to "Common bus access."

14.5.4.1 Master Read transfer

(1) Master Read mode

There are two modes of the master read mode: EOP enable mode and EOP disable mode.

(a) EOP enable mode

Master Read transfers when UDC2STSET<eopb_enable> is set to 1 (Master Read EOP enable) are described here. Master Read operations will be as follows:

1. Set UDFSMWSADR and UDFSMWEADR.
2. Set the bits associated to the master read operation of UDFSMSTSET and set 1 to <mr_enable>.
3. UDC2AB starts the data transfer to the EP of UDC2. UDC2 transfers the data to the IN token from the USB host.
4. When the Master Read transfer reaches the Master Read end address, UDC2AB asserts the mr_end_add interrupt.
5. After the handling by the software ended, return to 1.

- About Short packets

  If the transfer size (Master Read End Address - Master Read Start Address Å{1) is not the same size as the Max packet size, the last IN transfer will be the transfer of short packets.

Example: In case Master Read transfer size 139 bytes and the Max packet size 64 bytes.

Transfer will take place in:

1st time	→	2nd time	→	3rd time
64 bytes		64 bytes		11 bytes

- About mr_end_add interrupt

  The mr_end_add interrupt occurs when the data transfer to the UDC2 EP is finished. In order to confirm whether the entire data has been transferred from UDC2 to the USB host, check the UDFSMSTSTS<mrepempty>.

(b) EOP Disable mode

Master Read transfers when UDC2STSET<eopb_enable > is set to 0 (Master Read EOP disable) are described here. Master Read operations will be as follows:

1. Set the register associated to the UDFSMWSADR and UDFSMWEADR.
2. Set the bits associated to the Master Read operation of UDFSMSTSET and set 1 to the <mr_enable>.
3. UDC2AB starts the data transfer to the EP of UDC2. UDC2 transfers the data to the IN token from the USB host.
4. When the Master Read transfer reaches the Master Read end address, UDC2AB asserts the mr_end_add interrupt. If the FIFO of the EP has reached the MAX packet size during Master Read transfer, UDC2 transfers the data to the IN token from the USB host. However, if it has not reached yet, data will remain in the FIFO to the next transfer.
5. After the handling by the software ended, return to 1.

Note: When UDC2AB is used in the EOP Disable mode, short packets will not be sent out even if the data string to be sent has been transferred. EOP Disable mode should be used only in case the size of the data string is a multiple of the maximum packet size.

The mode can be used if the total size of data string is a multiple of the maximum packet size. For example, the following transfer may be allowed:

Example:
Size of the first Master Read transfer :100 bytes
Size of the second Master Read transfer :28 bytes (total of first and second transfer = 128bytes)
Max packet size :64 bytes

A transfer of 64 bytes will be made twice for the IN transfer.

(2) Aborting of Master Read transfer

You can abort Master Read transfers with the following operation.

1. Use UDC2 Command register to set the status of the relevant EP to Disabled (EP_Disable). (If aborted without making the EP disabled, unintended data may be sent to the USB host.)
2. In order to stop the Master Read transfer, set 1 (Abort) to UDFSMSTSET <mr_abort>.
3. In order to confirm that the transfer is aborted, check that the UDFSMSTSET<mr_enable> was disabled to 0. Subsequent operations should not be made while the mr_enable bit is 1.

   (Information on the address where the transfer ended when aborted can be confirmed with Master Read Current Address and Master Read AHB Address registers.)
4. In order to initialize the Master Read transfer block, set 1 (Reset) to UDFSMSTSET<mr_reset>.
5. Use the Command register (EP_FIFO_Clear) to initialize the FIFO for the relevant EP.
6. Use the Command register (EP_Enable) to enable the relevant EP.

(3) Setting the maximum packet size in Master Read transfers

If the maximum packet size of the EP to be connected with the Master Read function of UDC2AB will be an odd number, there will be following restrictions to which you should pay attention:

- Even if the maximum packet size of the EP should be handled as an odd number, the setting of the UDFS2EPxMSZ<max_pkt> should be an even number.

Note: Refer to the "14.9.2 Appendix B About Setting an Odd Number of Bytes as MaxPacketSize" for details.

- Set the UDC2STSET<eopb_enable> to 1 (Master Read EOP enable).
- Make the transfer size to be specified for one Master Read transfer (Master Read End Address - Master Read Start Address + 1) not exceed the maximum packet size of an odd number.

Example:
Set the maximum packet size of EP (value to pass to the USB host) to be 63bytes.
Make the setting of the UDFS2EPxMSZ<max_pkt> to be 64 bytes.
Keep the transfer size to be specified for one Master Read transfer to 63 bytes or less.

14.5.4.2 Master Write transfer

(1) Master Write Transfer Sequence

Master Write operations will be as follows:

1. Set UDFSMWSADR and UDFSMWEADR.
2. Set the bits associated to the UDFSMSTSET and set 1 to the <mw_enable>.
3. UDC2AB makes a Master Write transfer to the data in the EP received from the USB host.
4. Since the mw_end_add interrupt will be asserted when the writing ended to reach the Master Write End Address (with no timeout processed), you should make necessary arrangement with the software. UDC2 will return to 1 after receiving the correct packet.

Note: UDC2AB will assert the mw_set_add interrupt when the packet is received normally from the USB host with the UDFSMSTSET<mw_enable disabled.

(2) Timeout

Master Write transfers would not finish if the OUT transfer from the USB host should stagnate before reaching the Master Write End Address during the transfer. In order to cope with such circumstances, you can set the timeout function.

When this timeout function is used, all data stored in the buffer in UDC2AB at the point of timeout will be transferred to AHB.

Timeout can be processed with the following operation.

1. Make an access to the UDFSMWTOUT before starting a Master Write transfer and set timeoutset (timeout time) to make <timeout_en> enabled 1.
2. Start the Master Write transfer in accordance with the instruction in the preceding section.

3. When the timeout has occurred, the mw_timeout interrupt will be asserted. (The mw_end_add interrupt will not be asserted.) In that case, the Master Write transfer is not completed to reach the Master Write End Address. UDC2AB clears the UDFSMSTSET<mw_enable> to 0.

4. In UDFSMWCADR, the address to which the transfer has completed to the AHB end can be confirmed.

Please note that the timeout counter advances during the Master Write transfer with the timeout function enabled, but the counter will be reset to the preset value when the OUT transfer from the USB host to the relevant EP is received and begin recounting (see the Figure 14-15). It means that the time until timeout is "from the point when the last transfer from the USB host to the relevant EP has occurred during the Master Write transfer to the preset time," rather than "from the point when the Master Write transfer has begun to the preset time."

If you do not use the timeout function, be sure to set the UDFSMWTOUT<timeout_en> to "Disable 0" before starting the Master Write transfer. In that case, the transfer will not finish until reaching the preset Master Write End Address.

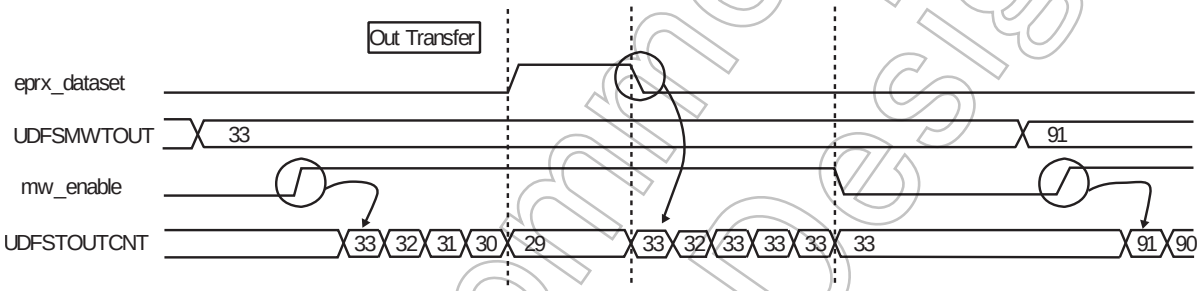


Figure 14-15 Example of MW timeout count

(3) Aborting of Master Write transfers

You can abort Master Write transfers with the following operation.

1. Use UDFS2CMD to set the status of the relevant EP to Disable (EP_Disable).

2. In order to stop the Master Write transfer, set 1 (Abort) to the UDFSMSTSET<mw_abort>.

3. In order to confirm the transfer is aborted, check the UDFSMSTSET<mw_enable> was disabled to 0. Subsequent operations should not be made while the <mw_enable> is 1. (Information on the address where the transfer ended when aborted can be confirmed with Master Write Current Address and Master Write AHB Address registers.)

4. In order to initialize the Master Write transfer block, set 1 (Reset) to the UDFSMSTSET<mw_reset>.

5. Use UDFS2CMD (EP_FIFO_Clear) to initialize the FIFO for the relevant EP.

6. Use UDFS2CMD to set the status of the relevant EP to Enable (EP_Enable).

14.5.5 USB Power Management Control

In USB, operations related to power management including detection of USB bus power supply, suspending and resuming are also prescribed in addition to normal packet transfers. This section discusses about how to control those operations.

Note: Be sure to see the USB 2.0 Specification for details of operations.

14.5.5.1 Connection Diagram of Power Management Control Signal

Below is a connection diagram of signals related to power management control.

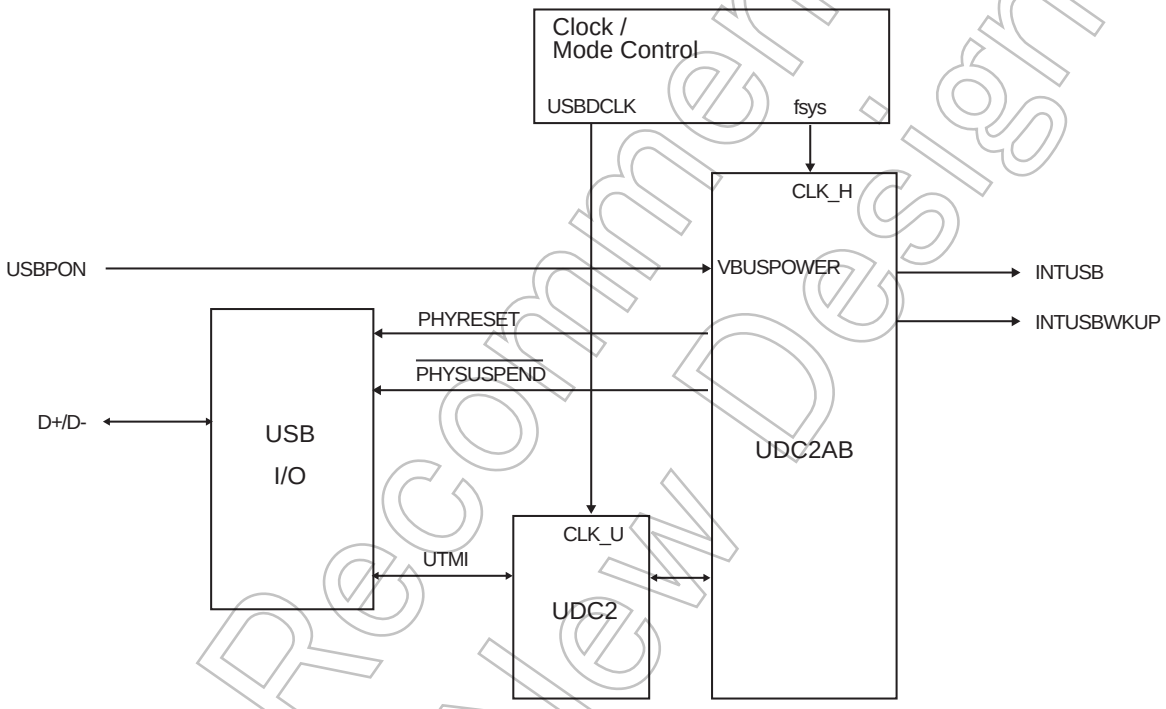


Figure 14-16 Connection Diagram of Power Management Control Signal

14.5.5.2 Sequence of USB Bus Power (VBUS) Connection/Disconnection

(1) Connect

If CLK_H is operating, the USB bus power (VBUS) connection is detected using the INTUSB(powerdetect) interrupt and UDFSPWCTL<pw_detect>. If UCLK_H is stopped, the USB power connection (VBUS) is detected using the INTUSBPON interrupt signal.

After detecting bus power (VBUS), initialize UDC2AB and UDC2 following sequence.

1. Use the UDFSPWCTL<pw_resetb> to make software reset. (The <pw_resetb> bit is not automatically released and should be cleared by software.).
2. Make an access to UDC2AB and UDC2 registers to make necessary initial settings.
3. Use UDFS2CMD to issue the USB Ready command. UDC2 notifies the USB host of the connection via PHY. This condition enables UDC2 to accept USB_RESET from the USB host.
4. Once USB_RESET from the USB host is detected, UDC2 initializes the registers inside UDC2 and enumeration with the USB host becomes available. When USB_RESET is detected, the usb_reset / usb_reset_end interrupt occurs.

(2) Disconnect

If CLK_H is operating, the USB bus power (VBUS) disconnection is detected using the INTUSB (powerdetect) interrupt and UDFSPWCTL<pw_detect>. If CLK_H is stopped, the USB power (VBUS) disconnection is detected using the INTUSBWKUP interrupt.ÅB

When the disconnection of the USB bus power (VBUS) is detected, each master transfer will not automatically stop. Then use the pw_resetb bit of Power Detect Control register to make software reset.

14.5.6 USB Reset

USB_RESET may be received not only when the USB host is connected but also at any timing.

UDC2AB asserts the usb_reset / usb_reset_end interrupt when UDC2 has received USB_RESET and returns to the default state. At this time, master transfers will not automatically stop. Use the abort function to end the transfers. Values are initialized by USB_RESET for some registers of UDC2, while they are retained for other registers (refer to the section of UDC2).

Resetting of UDC2 registers when USB_RESET is recognized should be made after the usb_reset_end interrupt has occurred. This is because UDC2 initializes UDC2 registers at the time it deasserts the usb_reset signal.

14.5.7 Suspend / Resume

14.5.7.1 Shift to the suspended state

UDC2AB makes notification of detecting the suspended state of UDC2 by the INTUSB (suspend_resume) interrupt and the UDFSPWCTL<suspend_x>.

Since master transfers will not automatically stop in this circumstance, you should use the aborting function of each master transfer to make forcible termination if needed.

In case PHY needs to be suspended (clock stop) after the necessary processes finished by software, you can set the UDFSPWCTL<phy_suspend> to make UDC2AB assert PHYSUSPEND which will put PHY in suspended state.

14.5.7.2 Resuming from suspended state (resuming from the USB host)

The procedures to resume from the suspended state is performed based o the condition of the CLK_H.

When resuming is recognized, make settings again for restarting master transfers.

1. Stopping the CLK_H

   The procedures to stop the CLH_H and the signal variation are as shown below.

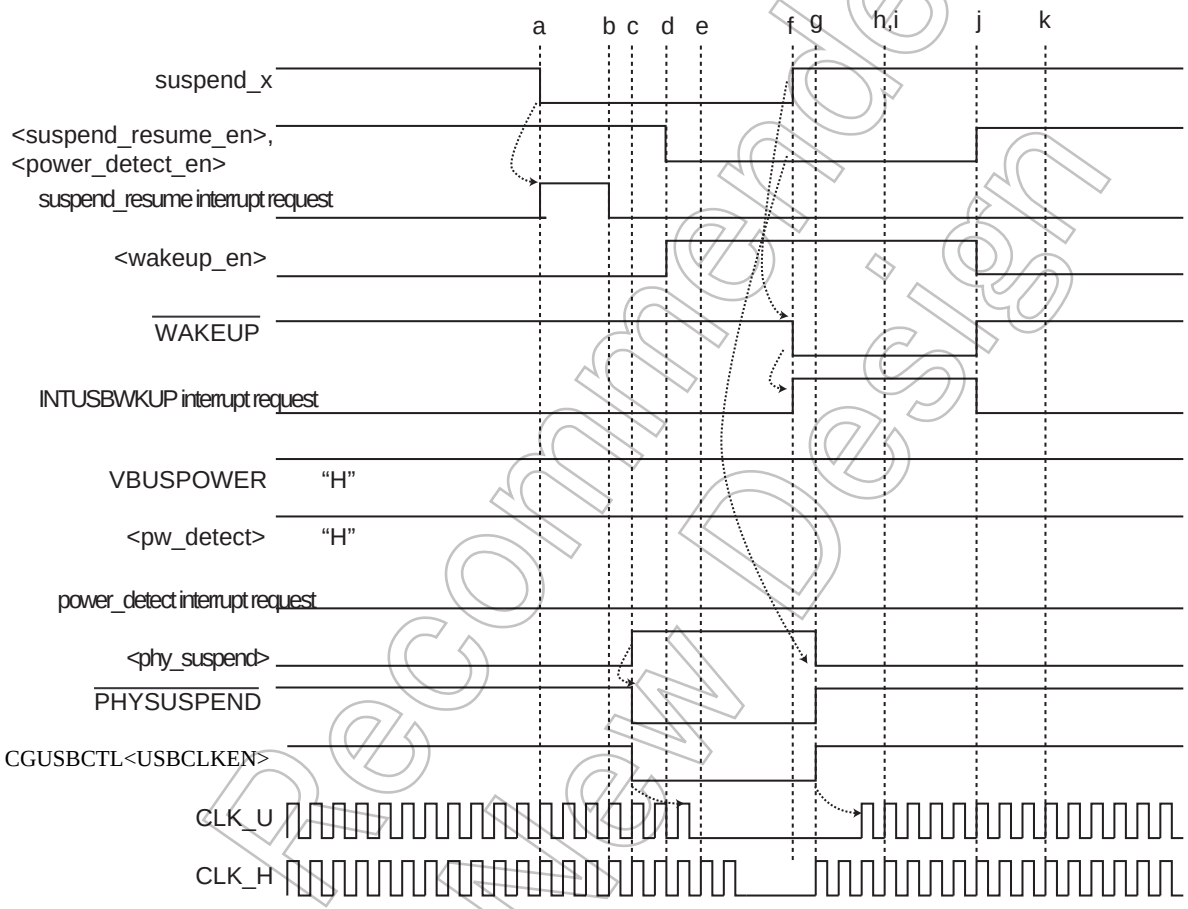


Figure 14-17 Signal operations when suspended and resumed (when CLK_H is stopped)

a. The suspend_x of the UDC2 is asserted to zero by detecting the suspend state on the USB bus, and the INTUSB(suspend_resume) interrupt occurs.
b. The service routine of the INTUSB(suspend_resume) interrupt clears the interrupt factor.
c. Set the UDFSPWCTL<phy_suspend> to "1". Setting the <phy_suspend> to "1" asserts the PHYSUSPEND output signal to "0".

   Zero clear the CGUSBCTL<USBCLKEN> of the clock/mode control circuit CGUSBCTL<USBCLKEN> to stop the CLK_U.
d. Set the UDFSPWCTL<wakeup_en> to "1". Zero clear the UDFSINTENB<power_detect_en><suspend_resume_en> not to generate the INTUSBD(power_detect, suspend_resumu) interrupt.
e. With the INTUSBWKUP interrupt, the operation mode moves into the low-power consumption mode and stops the CLK_H.

f. By detecting Resume on the USB bus, the $\overline{\text{WAKEUP}}$ output signal will be asserted to 0 asynchronously.By $\overline{\text{WAKEUP}}$ output signal, INTUSBWKUP occurs and the low-power consumption mode is cancelled. Then, supply of CLK_H starts.

g. With the supply of CLK_H, $\overline{\text{PHYSUSPEND}}$ output signal is automatically asserted to "1", and <phy_suspend> is zero-cleared.

Set CGUSBCTL<USBCLKEN> of the clock/mode control circuit to "1" to activate the CLK_U.

h. 2.5 s after the interrupt is asserted (time required for the signal to stabilize when VBUS is disconnected) and check UDFSPWCTL<pw_detect>. If the UDFSPWCTL<pw_detect> is "1", $\overline{\text{WAKEUP}}$ is asserted by Resume. If UDFSPWCTL<pw_detect> is "0", $\overline{\text{WAKEUP}}$ is asserted by disconnection of the VBUS.

i. To resume, perform the sequences below.To disconnect, perform the sequences of the "14.5.7.3 Resuming from the suspend state (disconnect)".

j. Clears the interrupt factor and <wakeup_en> to deassert the $\overline{\text{WAKEUP}}$ output signal. Set <suspend_resume_en> to "1".

k. Resumes from the suspended state.

2. For CLK_H to work

The procedures to get the CLK_H work and the signal changes are shown as below.

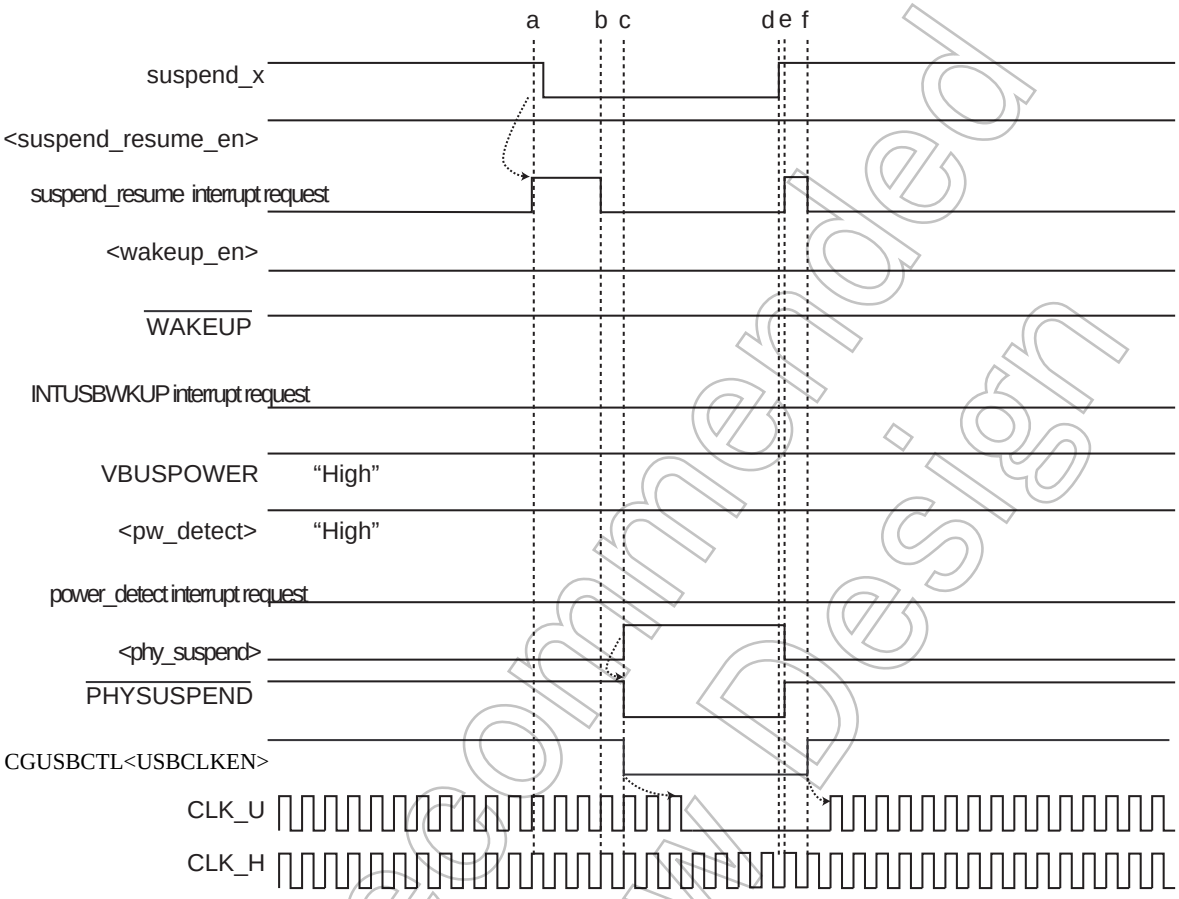


Figure 14-18 Operation of suspend/resume signals (to get the CLK_H work)

a. INTUSB(suspend_resume) interrupt occurs by detecting the suspended state on the USB bus.
b. Clears the interrupt source in the INTUSB(suspend_resume) interrupt service routine.
c. Set "1" to UDFSPWCTL<phy_suspend>. Setting <phy_suspend> to "1" assert the $\overline{\text{PHYSUSPEND}}$ output signal to "0".

   Set CGUSBCTL<USBCLKEN> of the clock/mode circuit to "0" to stop the CLK_U.
d. The suspend_x becomes "1" by detecting resume on the USB bus.

   Aslo, $\overline{\text{PHYSUSPEND}}$ output signal is deasserted to "1" by detecting the rising edge of the suspend_x.
e. INTUSB(suspend_resume) interrupt occurs.
f. Interrupt source is zero cleared in the service routine of the INTUSBD(suspend_resume).

   Set CGUSBCTL<USBCLKEN> of the clock/mode circuit to "1" to get CLK_U work.
g. Deasserting the $\overline{\text{PHYSUSPEND}}$ output signal will resume the supply of the CLK_U.
h. Resumes from the suspend state.

14.5.7.3 Resuming from the suspend state (disconnect)

The procedures to resume from the suspended state (disconnection) and the signal change are shown as below.

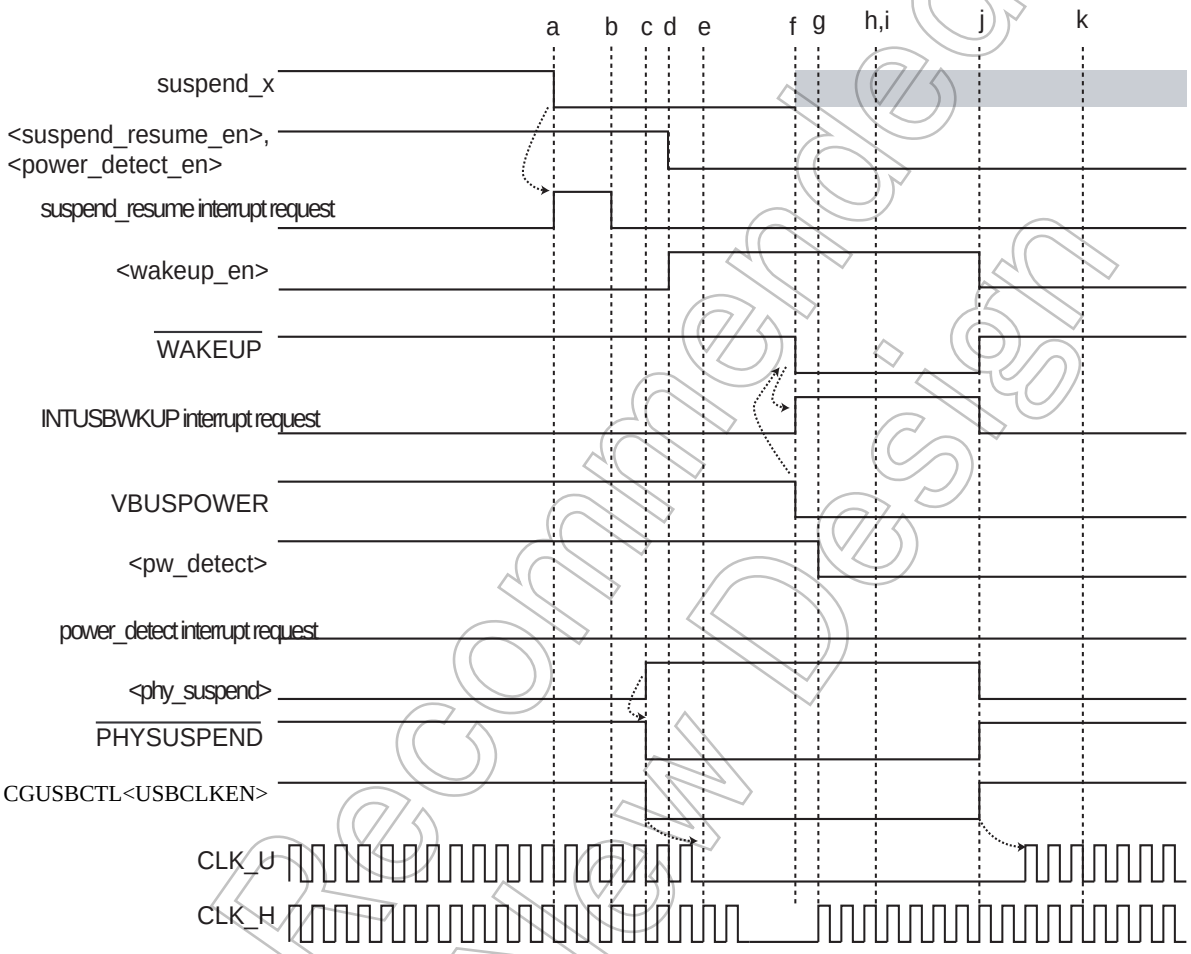


Figure 14-19 Operation of suspend/disconnect signals (to stop CLK_H)

a. 0 is asserted to the suspend_x of the UDC2 by detecting the suspend state on the USB bus and this generates the INTUSB(suspend_resume) interrupt.
b. Interrupt source is cleared by the service routine of the INTUSB(suspend_resume) interrupt.
c. Set UDFSPWCTL<phy_suspend> to "1". Setting the <phy_suspend> to "1" asserts the $\overline{\text{PHYSUSPEND}}$ output signal to "0".

   Set CGUSBCTL<USBCLKEN> of the clock/mode circuit to "0" to stop the CLK_U.
d. Set the UDFSPWCTL<wakeup_en> to "1". Zero clear the UDFSINTENB<power_detect_en><suspend_resume_en> not to generate INTUSBD(power_detect, suspend_resumu) interrupt.
e. With the INTUSBWKUP interrupt, the operating mode moves into the low-power consumption mode to stop the CLK_H.
f. If disconnection is detected on the USB bus, the VBUSPOWER pin becomes "0" and the $\overline{\text{WAKEUP}}$ output signal will be asynchronously asserted to "0".
g. $\overline{\text{INTUSBWKUP}}$ interrupt is generated by the $\overline{\text{WAKEUP}}$ output signal and low-power consumption mode is cancelled. The supply of CLK_H starts.

h. 2.5 s after the interrupt is asserted (time required for the signal to stabilize when VBUS is disconnected), check the UDFSPWCTL<pw_detect>. If UDFSPWCTL<pw_detect> is "1", $\overline{\text{WAKEUP}}$ is asserted by resume. If UDFSPWCTL<pw_detect> is "0", $\overline{\text{WAKEUP}}$ is asserted by disconnection of VBUS.

i. If the factor is the resume, perform the sequence written in the "14.5.7.2 Resuming from suspended state (resuming from the USB host)". If the factor is disconnection, perform the sequence below.

j. Zero clear the <phy_suspend> to deassert the $\overline{\text{PHYSYSPEND}}$ output signal.

Set the CGUSBCTL<USBCLKEN> of the clock/mode circuit to "1" to get the CLK_U work.Clear the interrupt factor and <wakeup_en> to deassert the $\overline{\text{WAKEUP}}$ output signal.

k. Set UDFSPWCTL<pw_resetb> using software, initialize the UDC2AB.

14.5.7.4 Remote wakeup from the suspended state

The procedure of remote wakeup from the suspended state and the signal change are shown below.

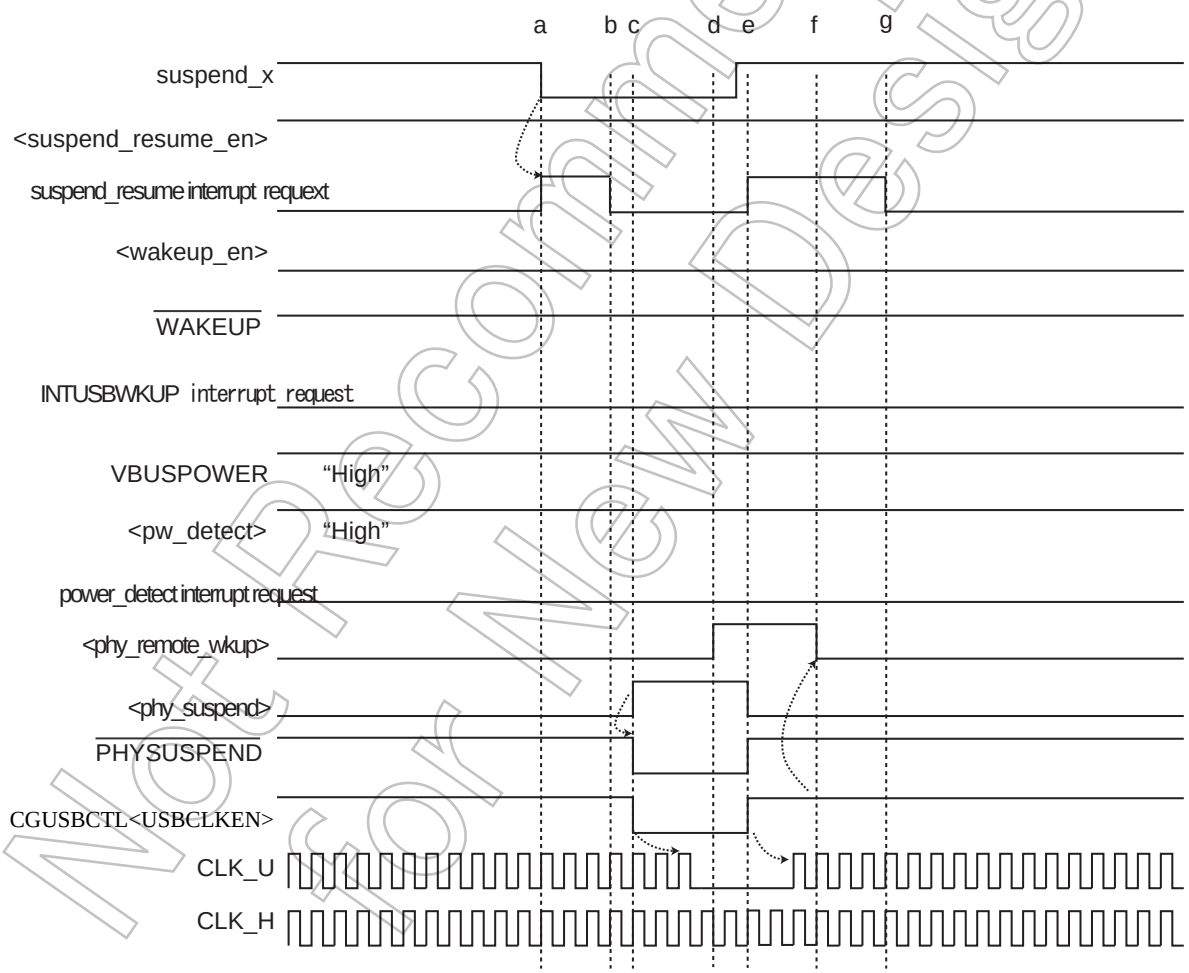


Figure 14-20 Operation of suspend/remote wakeup signals

a. suspend_x of the UDC2 is asserted to 0 by detecting the suspended state on the USB bus and the INTUSB(suspend_resume) interrupt occurs.

b. Clears the interrupt source in the service routine of the INTUSB(suspend_resume) interrupt.

c. Set the UDFSPWCTL<phy_suspend> to "1". $\overline{\text{PHYSUSPEND}}$ output signal is asserted to "0" by setting <phy_suspend> to "1"

Set the CGUSBCTL<USBCLKEN> of the clock/mode circuit to "0" to stop the CLK_U.

d. When requesting remote wakeup, set the UDFSPWCTL<phy_remote_wkup> to 1. Setting the <phy_remote_wkup> to "1" will cause the UDC2 to make a remote wakeup request on the USB bus. Also, <suspend_x> will be deasserted to 1 asynchronously.
e. Deasserting <suspend_x> will cause the INTUSB(suspend_resume) interrupt to occur and the PHYSUSPEND output signal to be deasserted to 1.
f. Set the CGUSBCTL<USBCLKEN> of the clock/mode circuit to "1" to get the CLK_U work.

When the CLK_U starts operating, <phy_remote_wkup> is automatically cleared to "0".

g. Clear the interrupt source.

14.6 USB Device Response

UDC2 initializes the inside of UDC2 and sets various registers when hardware reset is detected, USB_RESET is detected, and an enumeration response is made. This section discusses the operations of UDC2 in each status as well as how to control them externally.

1. When hardware reset is detected

   Be sure to reset hardware for UDC2 after the power-on operation. After the hardware reset, UDC2 initializes internal registers and all EPs are in the invalid status, which means the device itself is "Disconnected."

   In order to make the status of UDC2 to "Default," issue the "USB_Ready" command. Issuing this command will put UDC2 in the "Full-Speed" mode, enable the Pull-Up resistance of D+ and notify the host of "Connect".

   In this status, only the USB_RESET signal is accepted from the host.

2. When USB_RESET is detected

   UDC2 initializes internal registers when Bus Reset (USB_RESET) is detected on the USB signal, putting the device in the "Default" status. In this status only EP 0 gets "Ready" enabling enumeration with the host.

3. When "Set_address" request is received

   By setting 010 to the UDFS2ADR<configured> <addressed> <default> and the received address value to the <dev_adr> after receiving the "Set_address" request, UDC2 will be in the "Addressed" status. Setting for this register should be made after the Control transfer has successfully finished (after the STATUS-Stage has ended).

   Transfers to EPs other than EP 0 cannot be made in this status.

4. When "Set_configuration" and "Set_interface" requests are received

   By setting 100 to the UDFS2ADR<configured> <addressed> <default> after receiving the "Set_configuration" and "Set_interface" requests, UDC2 will be in the "Configured" status.

   In the "Configured" status, you can make transfers to the EP to which status settings have been made.

   In order to make the EP "Ready," the following settings should be made:

   - Set the maximum packet size to UDFS2EPxMSZ
   - Set the transfer mode to UDFS2EPxSTS
   - Issue the EP_Reset command to UDFS2CMD

   EPs will be available for transmitting and receiving data after these settings have been made.

   Figure 14-21 shows the "Device State Diagram".

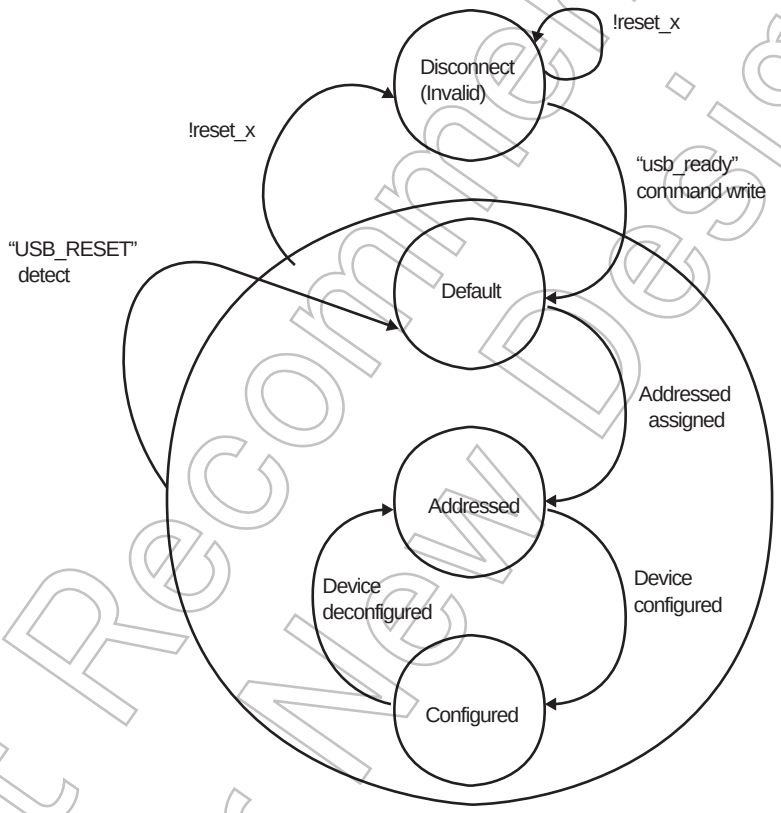


Figure 14-21 Device state diagram

14.7 Flow of Control in Transfer of EPs

14.7.1 EP0

EP0 supports Control transfer and is used as device control for enumeration. EP0 supports only Single packet mode.

Control transfers have SETUP-Stage, DATA-Stage and STATUS-Stage

The types of transfer are categorized into the following major types:

- Control-RD transfer
- Control-WR transfer (without DATA-Stag)
- Control-WR transfer (with DATA-Stage)

UDC2 makes control of those three stages by hardware. Flows in each type of transfer are described below.

14.7.1.1 Control-RD transfer

The flow of control in Control-RD transfers is shown below.

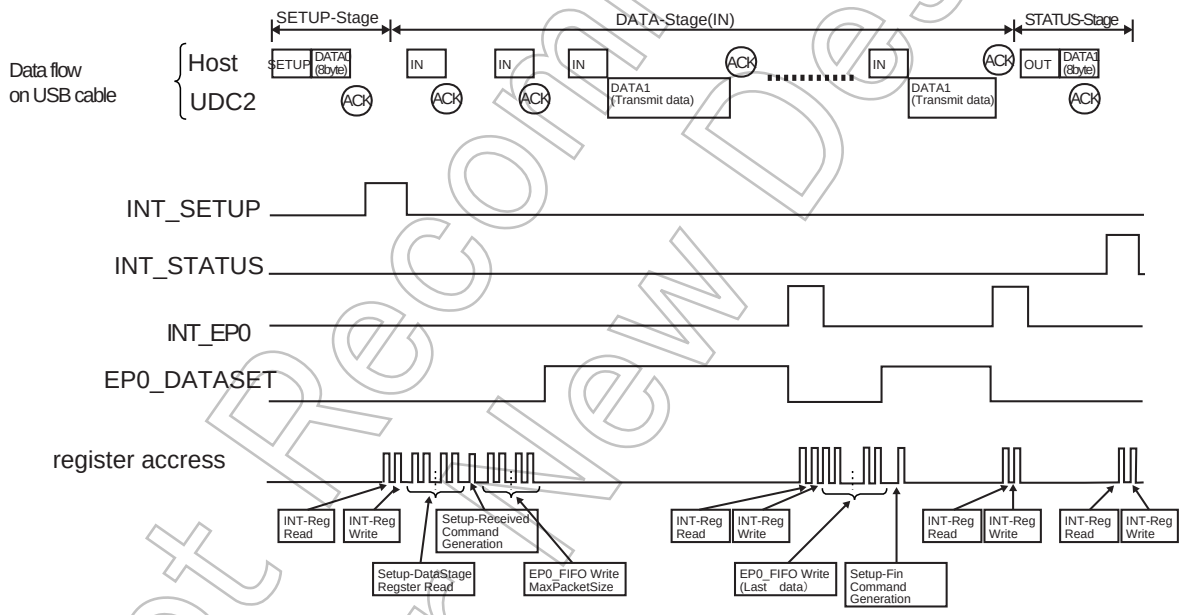


Figure 14-22 Flow of the control in Control-RD transfer

The following description is based on the assumption that the UDFS2EP0MSZ<dset> is set to "EP0_DATASET flag".

(1) SETUP-Stage

UDC2 asserts the INT_SETUP flag when it has received the Setup-Token. This flag can be cleared by writing 1 into the UDFS2INT<i_setup>. In case flags are combined externally, read the UDFS2INT to confirm which flag is asserted and write "1" into the relevant bit.

Then read Setup-Data storage registers (bRequest-bmRequestType, wValue, wIndex, and wLength registers) to determine the request.

Finally, issue the "Setup_Received" command to inform UDC2 that the SETUP-Stage has finished. Since UDC2 does not allow writing data into the EP0-FIFO before this command is issued, it will keep returning "NAK" to the IN-Token from the host until the command is issued.

(2) DATA-Stage

Write the data to be transmitted to the IN-Token into the EP0-FIFO. If the byte size of the data to send is larger than the MaxPacketSize, divide them into groups of MaxPacketSize before writing. When the number of data reached the MaxPacketSize, the EP0_DATASET flag is asserted.

When the data have been transmitted to the IN-Token from the host with no problem, UDC2 deasserts the EP0_DATASET flag and asserts INT_EP0. Any data remaining to be transmitted should be written into the EP0-FIFO.

If the size of the data to be written is smaller than the MaxPacketSize, issue the "EP_EOP" command to EP0 to inform UDC2 that it is a short packet. With this command, UDC2 recognizes the end of the packet and transmits the short packet data.

Finally, issue the "Setup_Fin" command to inform UDC2 that the DATA-Stage has finished.

(3) STATUS-Stage

When the "Setup_Fin" command is issued, UDC2 will automatically make Handshake for the STATUS-Stage. When the STATUS-Stage finished with no problem, the INT_STATUS flag is asserted. When received a packet of STATUS-Stage from the host before the "Setup_Fin" command is issued, UDC2 will return "NAK" and asserts the INT_STATUS_NAK flag. Therefore, if this flag is asserted, be sure to issue the "Setup_Fin" command.

14.7.1.2 Control-WR transfer (without DATA-Stage)

The flow of control in Control-WR transfer (without DATA-Stage) is shown below.

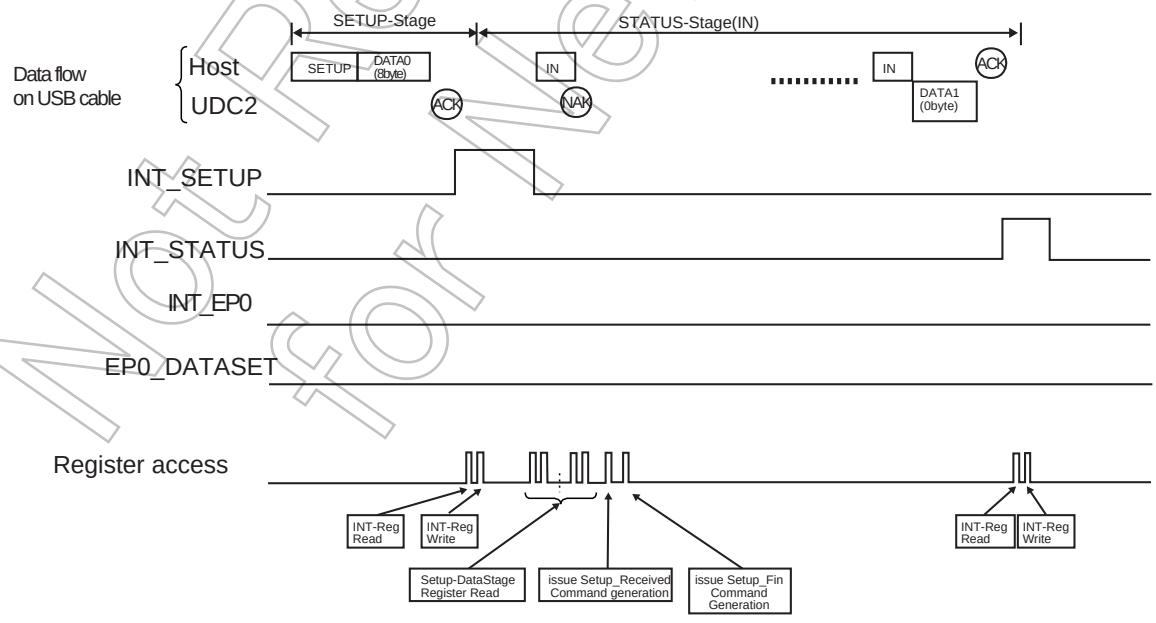


Figure 14-23 Flow of control in Control-WR transfer (without DATA-Stage)

(1) SETUP-Stage

Perform the same procedure described in "14.7.1.1 Control-RD transfer".

(2) STATUS-Stage

After issuing the "Setup_Received" command, make register accesses to UDC2 based on each request. Issue the "Setup_Fin" command when all the register accesses to UDC2 have finished. Subsequent processes are basically the same as the STATUS-Stage described in "14.7.1.1 Control-RD transfer". UDC2 will keep on returning "NAK" until the "Setup_Fin" command is issued.

Note: While register accesses required for each request are made to UDC2 between 'Issuing the "Setup_Received" command' and 'Issuing the "Setup_Fin" command', register accesses are needed after the end of STATUS-Stage in some cases such as Set Address request and Set Feature (TEST_MODE). Processes required for the standard requests are described in "14.7.1.5 Processing when standard request".

14.7.1.3 Control-WR transfer (with DATA-Stage)

The flow of control in Control-WR transfer (with DATA-Stage) is shown below.

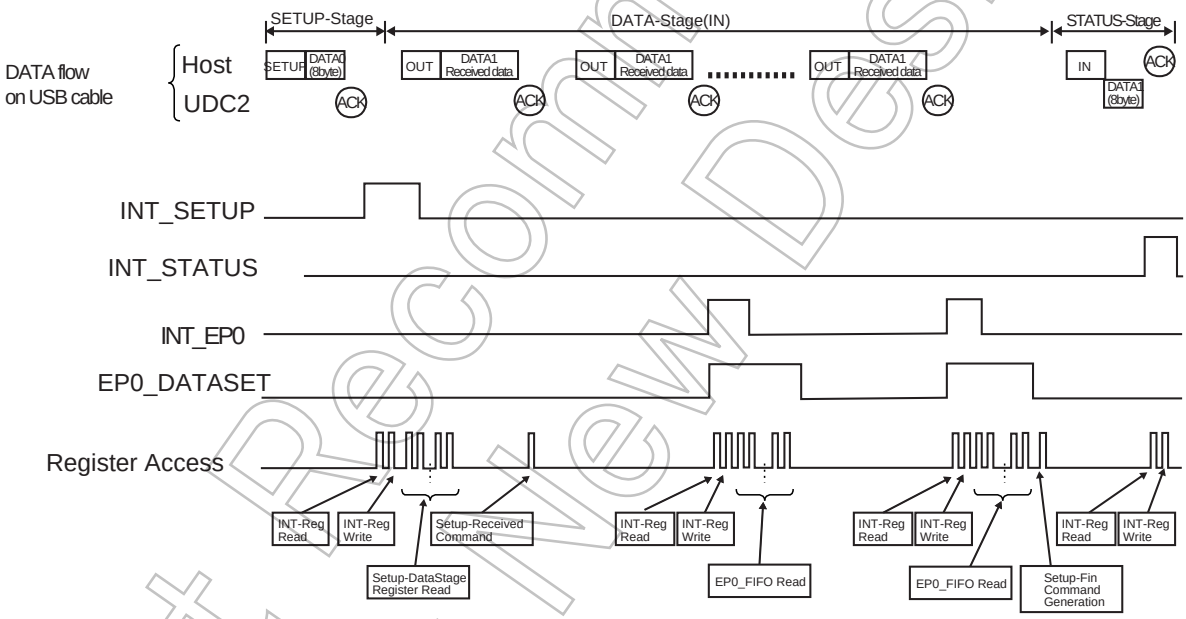


Figure 14-24 Flow of control in Control-WR transfers (with DATA-Stage)

(1) SETUP-Stage

To be processed in the same way of SETUP-Stage as described in"14.7.1.1 Control-RD transfer".

(2) DATA-Stage

When the data is received from the host with no problem, UDC2 asserts the EP0_DATASET flag and asserts the INT_EP0 flag. When this flag is asserted, read the data from EP0_FIFO after confirming the received data size in the UDFS2EP0FIFO, or read the data from EP0_FIFO polling the EP0_DATASET flag.

When the byte size of received data has been read, UDC2 deasserts the EP0_DATASET flag.

(3) STATUS-Stage

To be processed in the same way as in the STATUS-Stage described in "14.7.1.1 Control-RD transfer".

14.7.1.4 Example of using the INT_STATUS_NAK flag

When processing requests without DATA-Stage, the INT_STATUS_NAK flag may get asserted by receiving STATUS-Stage from the host before clearing the INT_SETUP flag after it has been asserted, especially in High-Speed transfers. In case such multiple interrupts should be avoided as much as possible, you can use a method to mask the INT_STATUS_NAK flag for request having no DATA-Stage. In such case, basically set 1 to UDFS2INT<m_status_nak>, while 0 should be set only when requests having DATA-Stage are received. (An example for Control-RD transfers is provided below.)

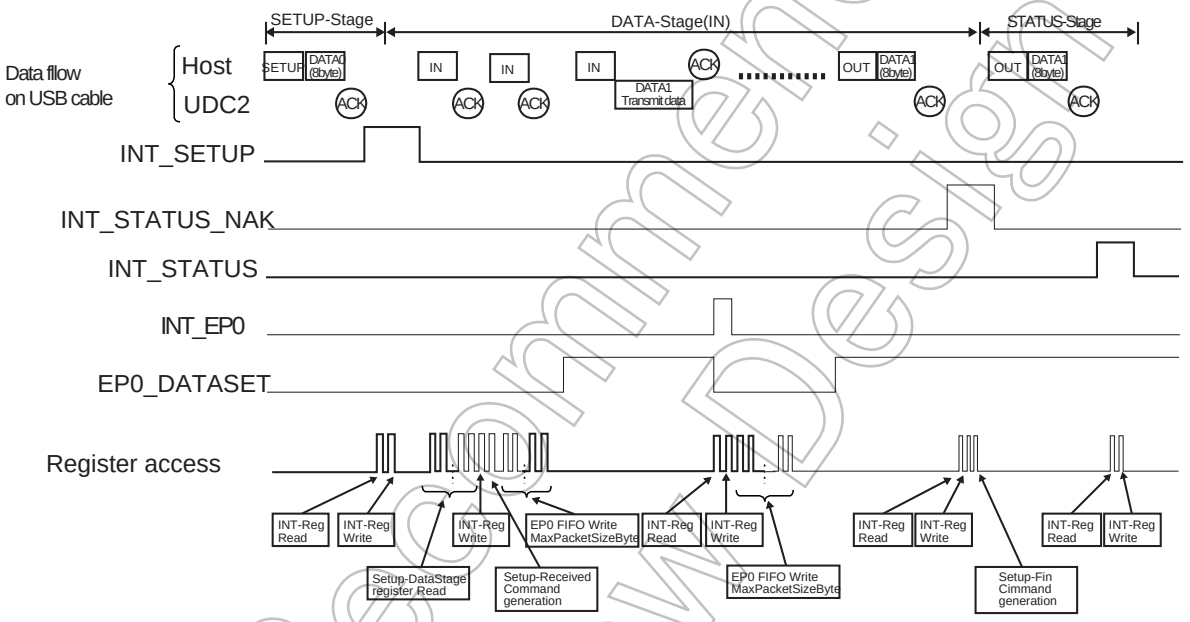


Figure 14-25 Example of using the INT_STATUS_NAK flag in Control-RD transfers

(1) SETUP-Stage

After the INT_SETUP flag was asserted, clear the UDFS2INT<i_setup> is set to 1, it should be also cleared.

Then, if the request was judged to have DATA-Stage by reading Setup-Data storage registers, set the UDFS2INT<m_status_nak> to 0. Then issue the "Setup_Received" command.

(2) DATA-Stage→STATUS-Stage

When the INT_STATUS_NAK flag was asserted, the device should also proceed to the STATUS-Stage. Clear the UDFS2INT<i_status_nak> and then issue the "Setup_Fin" command. Also, set 1 to the UDFS2INT<m_status_nak> in order to get ready for subsequent transfers.

14.7.1.5 Processing when standard request

Examples of making register accesses to UDC when standard requests are received are provided below. Descriptions of each request are basically provided for each state of the device (Default, Address, and Configured).

For the information on register accesses common to each request, see 14.7.1.1 , 14.7.1.2 and 14.7.1.3.

You should note, however, descriptions provided below do not include the entire details of standard requests in USB 2.0. Since methods to access registers may vary depending on each user's usage, be sure to refer to the USB 2.0 specifications. You should also refer to the USB 2.0 specifications for "Recipient," "Descriptor Types," "Standard Feature Selectors," "Test Mode Selectors" and other terms appear in the descriptions below.

- Standard requests for "14.7.1.1 Control-RD transfer".

  Get Status Get Description Get Configuration

  Get Interface Get Frame
- Standard requests for "14.7.1.2 Control-WR transfer (without DATA-Stage)".

  Clear Feature Set Feature Set Address

  Set Configuration Set Interface
- Standard requests for "14.7.1.3 Control-WR transfer (with DATA-Stage)"

  Set Description

Note 1: Descriptions with double underlines refer to register accessed to UDC2.
Note 2: Writing accesses to UDFS2CMD are described in the following manner for simplicity:

(Example 1) When writing 0x0 to UDFS2CMD<ep> and 0x4 to <com>

→Issue the EP-Stall command to EP0

(Example 2) When writing the relevant EP to UDFS2CMD<ep> and 0x5 to <com>

→Issue the EP-Invalid command to the relevant EP

(1) Get Status Request

To meet this request, the status of the specified receiving end (recipient) is returned.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
1000_0000 1000_0001 1000_0010	GET_STATUS	Zero	Zero Interface EP	Two	Device Interface, or EP Status

- Common to all states:

  If the EP/Interface specified by wIndex does not exist, issue the EP-Stall command to EP0.
- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.
- Address state:

<recipient> = Device : Write the information on the device (Table 14-3) to UDFS2EP0FIFO.

<recipient> = Interface: Issue the EP-Stall command to EP0

<recipient> = EP : If wIndex=0(EP0), write the information on EP0 (Table 14-5) to UDFS2EP0FIFO. If wIndex≠0(EPx), issue the EP-Stall command to EP0.

- Configured state:

<recipient> = Device : Write the information on the device (Table 14-3) to UDFS2EP0FIFO.

<recipient> = Interface: If the interface specified by lwIndex, write the information on the interface (Table 14-4) to UDFS2EP0FIFO.

<recipient> = EP : If the EP specified by wIndex, write the information on the relevant EP(Table 14-5) to UDFS2EP0FIFO.

Table 14-3 Information on the device to be returned by Get Status request

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0
D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	Remote Wakeup	Self Powered

RemoteWakeup (D1): 0 indicates the bus power while 1 indicates the selfpower.

SelfPowered (D0): 0 indicates the remote wakeup function is disabled while 1 indicates it is enabled.

Table 14-4 Information on the interface to be returned by Get Status

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0
D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

Please note that all bits are 0.

Table 14-5 Information on the EP to be returned by Get Status request

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0
D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	Halt

Halt (D1): If this bit is 1, it indicates that the relevant EP is in the "Halt" state.

(2) Clear Feature Request

To meet this request, the particular functions are cleared and disabled.

bmRequesType	bRequest	wValue	wIndex	wLength	Data
1000_0000 1000_0001 1000_0010	CLEAR_FEATURE	Feature Selector	Zero Interface EP	Zero	None

- Common to all states:

  If Feature Selector (wValue) which cannot be cleared (disabled) or does not exist is specified, issue the EP-Stall command to EP0.

  If the EP/Interface specified by wIndex does not exist, issue the EP-Stall command to EP0.

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state:

  <recipient> = Device : If wValue=1, disable the DEVICE_REMOTE_WAKEUP function at the user program. No register access to UDC2 is required.

  <recipient> = Interface: Issue the EP-Stall command to EP0.

  <recipient> = EP : If wIndex≠0(EPx), issue the EP-Stall command to EP0.

  If wValue=0 andwIndex=0(EP0), clear the Halt state of EP0 but no register access to UDC2 is required.

- Configured state:

  <recipient> = Device : If wValue=1, disable the DEVICE_REMOTE_WAKEUP function at the user program. No register access to UDC2 is required.

  <recipient> = Interface: Issue the EP-Stall command to EP0.(note)

  <recipient> = EP : If wValue=0 and wIndex=0(EPx), issue the EP-Reset command to the relevant command. If wValue=0 and wIndex=0(EP0), clear the Halt state of EP0 but no register access to UDC2 is required.

Note: EP 0 is to be stalled based on the interpretation of the USB 2.0 specifications that "No Feature Selector exists for Interface" here. For more information, see the USB Specification.

(3) Set Feature Request

To meet this request, the specific functions are set or enabled.

BmRequesetType	BRequesdt	wValue	wIndex		wLength	Data
1000_0000 1000_0001 1000_0010	SET_FEATURE	Feature Selector	Zero Interface EP	Test Selector	Zero	None

- Common to all state:

  If Feature Selector (wValue) which cannot be set (enabled) or does not exist is specified, Eissue the EP-Stall command to the EP0.

  If the EP/Interface specified by the lower byte of wIndex does not exist, issue the EP-Stall command to EP0.

Note: When using a vendor-specific nonstandard Test Selector, the appropriate operation should be made.

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications except for the above-mentioned TEST_MODE.

- Address state:

<recipient> = Device : If wValue=1, disable the DEVICE_REMOTE_WAKEUP function at the userÅfs end. No register access to UDC2 is required.
<recipient> = Interface: Issue the EP-Stall command to EP0.
<recipient> = EP : If the lower byte of wIndex ≠0 (EPx), issue the EP-Stall to EP0.
If wValue=0 and the lower byte of wIndex=0 (EP0), make EP0 to Halt state. (note 2)

- Configured state:

<recipient> = Device : If wValue=1, enalbe the DEVICE_REMOTE_WAKEUP function at the userÅfs end. No register access to UDC2 is required.
<recipient> = Interface: Issue the EP-Stall command to EP0.(note 1)
<recipient> = EP : If wValue=0 and the lower byte of wIndex≠0(EPx), issue the EP-Stall command to EP0.
If wValue=0 and the lower byte of wIndex=0(EP0), make EP0 to Halt state.(note 2)

Note 1: EP 0 is to be stalled based on the interpretation of the USB specifications that "No Feature Selector exists for Interface" here. For more information, see the USB specifications.

Note 2: USB 2.0 specifications include such description that "Performing the Halt function for EP 0 is neither necessary nor recommended." Accordingly, it can be interpreted that it is not necessary to set UDC2 to the Stall state in this case.

In order to actually make EP 0 be in the Halt state, users have to manage the "Halt state.

Then, when a request is received in the "Halt state", such processes as to issue the EP-Stall command to EP0 in DATA-Stage/STATUS-Stage will be required. (Even if EP0 is set to the Stall state, UDC2 will cancel the Stall state when the Setup-Token is received and will return "ACK.")

As such, the process when SetFeature/ClearFeature is received for EP 0 varies depending on user's usage.

(4) Set Address Request

To meet this request, device addresses are set.

BmRequesetType	BRequesdt	wValue	wIndex	wLength	Data
0000_0000	SET_ADDRESS	Device Address	Zero	Zero	None

For this request, make register accesses shown below within 2 ms after the STATUS-Stage has ended.

(The device address should not be changed before the Setup_Fin command is issued.)

- Default state:

  wValue=0: Keep the default state. No register access to UDC2 is required.

  wValue≠0: Set wValue to UDFS2ADR<dev_adr> and set 010 to <configured>, <addressed> and <default>. UDC2 will be put in the address state.

- Address state:

  wValue=0: Set 0x00 to UDFS2ADR<dev_adr> and 010 to <configured>, <addressed> and <default>. UDC2 will be put in the default state.

  wValue≠0: Set wValue to UDFS2ADR<dev_adr>. UDC2 will be set to a new device address.

- Configured state:

  Nothing is specified for the operation of devices by the USB 2.0 specification.

(5) Get Descriptor Request

To this request, the specified descriptor is returned.

BmRequesetType	BRequesdt	wValue	wIndex	wLength	Data
1000_0000	GET_DESCRIPTOR	Descriptor Type and Descriptor Index	Zero or Language ID	Descriptor Length	Descriptor

Common to all states:

Write the descriptor information specified by wValue to UDFS2EP0FIFO for the byte size specified by wLength. If the byte size to write is larger than the MaxPacketSize of EP 0, you need to divide the data to write it several times (refer to "14.7.1.1 Control-RD transfer" for details). (If the length of the descriptor is longer than wLength, write the information for wLength bytes from the beginning of the descriptor. If the length of the descriptor is shorter than wLength, write the full information for the descriptor.)

If the descriptor specified by wValue is not supported by the user, issue the EP-Stall command to EP0.

(6) Set Descriptor Request

BmRequesetType	BRequesdt	wValue	wIndex	wLength	Data
0000_0000	SET_DESCRIPTOR	Device Type and Descriptor Index	Language ID or Zero	Descriptor Length	Descriptor

- Common to all states:

  When this request is not supported, issue the EP-Stall command to EP0.
- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.
- Address state & Configured state:

  Read the information on the description received by UDC2 from UDFS2EP0FIFO.

(7) Get Configuration Request

BmRequesetType	BRequesdt	wValue	wIndex	wLength	Data
1000 0000	GET_CONFIGURATION	Zero	Zero	One	Configuration Value

- Default state:

  To this request, the Configuration value of the current device is returned.
- Address state:

  Write 0x00 to UDFS2EP0FIFO. As this is not configured, 0 should be returned.
- Configured state:

  Write the current configuration value to the UDFS2EP0FIFO.

  Since this has been configured, values other than 0 should be returned.

(8) Set Configuration Request

To meet this request, Device Configuration is set.

BmRequesetType	BRequesdt	wValue	wIndex	wLength	Data
0000 0000	SET_CONFIGURATION	Configuration Value	Zero	Zero	None

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state:

  When wValue = 0:

  - Keeps the address state. No register access to UDC2 is required.

  When wValue≠0 and the wValue is a Configuration value matching the descriptor :

  - Set 100 to UDFS2ADR<configured> <addressed> <default>.

  <For EPs to use>

  - Set MaxPacketSize to UDFS2EPxMSZ<max_pkt>.
  - Set respective values to UDFS2EPxSTS<pkt_mode>, <bus_sel>, <dir>, <t_type> and <num_mf>.
  - Issue the EP-Reset coomand to the relevant EPs.

  When wValue≠0 and the wValue is a Configuration value not matching the descriptor:

  - Issue the EP-Stall command to EP0.

- Configured state:

  When wValue = 0:

  - Set 010 to UDFS2ADR<configured> <addressed> <default>.
  - Issue the All-EP-Invalid command.

  When Value≠0 and it is a Configuration value matching the descriptor:

  <For EPs to use>

  - Set the MaxPacketSize to UDFS2EPxMSZ<max_pkt>.
  - Set respective values to UDFS2EPxSTS<pkt_mode>, <bus_sel>, <dir>, <t_type> and <num_mf>.
  - Issue the EP-Reset command to the relevant EPs.

  <For EPs to become unused>

  - Issue the EP-Invalid command to the relevant EPs.

  When wValue≠0 and the wValue is a Configuration value not matching the descriptor :

  - Issue the EP-Stall command to EP0.

(9) Get Interface Request

To meet this request, the AlternateSetting value set by the specified interface is returned.

BmRequesetType	BRequesdt	wValue	wIndex	wLength	Data
1000_0001	GET_INTERFACE	Zero	Interface	One	Alternate Setting

- Common to all states:

  If the interface specified by wIndex, issue the EP-Stall command to EP0.
- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.
- Address state:

  Issue the EP-Stall to EP0.
- Configured state:

  Write the current alternate setting value of the interface specified by the wIndex to UDFS2EP0FIFO.

(10) Set Interface Request

To meet this request, the Alternate Setting value of the specified interface is set.

BmRequesetType	BRequesdt	wValue	wIndex	wLength	Data
0000_0001	SET_INTERFACE	Alternate Setting	Interface	Zero	None

- Common to all states:

  If the interface specified by wIndex does not exist or if the Alternate Setting specified by wValue does not exist, issue the EP-Stall command to EP0.
- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.
- Address state:

  Issue the EP-Stall command to EP0.
- Configured state:

  <For the EPs to use in Alternate Setting of the specified interface>
  - Set MaxPacketSize to UDFS2EPxMSZ<max_pkt>.
  - Set respective values to UDFS2EPxSTS<pkt_mode>, <bus_sel>, <dir>, <t_type> and <num_mf>.
  - EP-ResetIssue the EP-Reset command to the relevant EPs.

  <For EPs to become unused>
  - Issue the EP-Invalid command to the relevant EPs.

(11) Synch Frame Request

To meet this request, the Synch Frame of the EP is returned.

BmRequesetType	BRequesdt	wValue	wIndex	wLength	Data
1000 0010	SYNCH_FRAME	Zero	EP	Two	Frame Number

- Common to all states:

  If this request is not supported by the EP specified by wIndex, issue the EP-Stall command to EP0.
- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.
- Address state:

  Issue the EP-Stall command to EP0.
- Configured state:

  Write the Frame Number of the EP specified by wIndex to UDFS2EP0FIFO.

14.7.2 EPs other than EP0

EPs other than EP 0 support Bulk (send/receive), Interrupt (send/receive), and Isochronous (send/receive) transfers and are used to transmit and receive data. They also support the Dual Packet mode which enables high-speed data communication.

14.8 Suspend/Resume State

UDC2 enters into a suspended state based on the signal condition from the host. It also returns from the suspended state by resuming operation by the host or UDC2.

Shifting between the states is described below.

14.8.1 Shift to the suspended state

Though the host issues SOF with given intervals (FS: 1 ms) in the normal state, it will stop issuing this SOF to the device when it tries to make the device suspended and the data on the USB signal line will be unchanged keeping the idle state. UDC2 is always monitoring the "line_state" from PHY and makes judgment of whether it is in the suspended state or USB_RESET when the idle state is detected for 3 ms or longer. If judged to be in the suspended state, it will assert "suspend_x" to "Low" and enter in the suspended state.

Please note accesses to registers will be unavailable while UDC2 is suspended, since supply of CLK from clock/mode control circuit.

14.8.2 Resuming from suspended state

Resuming from the suspended state can be made in two ways; by outputting a resuming state from the host and by way of remote wakeup from UDC2 (outputting a resuming state).

Resuming process in each case is described below.

14.8.2.1 Resuming by an output from the host

When a resuming state is output by the host, UDC2 deasserts suspend_x to "High" to declare resuming from the suspend state.

14.8.2.2 Resuming by way remote wakeup from UDC2

The remote wakeup function may not be supported by some applications, and it needs to be permitted by the USB host at the time of bus enumeration. You should not assert "wakeup" unless permitted by the system.

If permitted by the system, asserting the "wakeup" pin will make UDC2 output a resuming state to the host to start resuming. Please note that the clock supply from clock/mode control circuit is stopped when UDC2 is suspended, so you should keep asserting wakeup until it resumes. The remote wakeup should be operated after 2 ms or more has passed after suspend_x was asserted to "Low".

14.9 USB-Spec2.0 Device Controller Appendix

14.9.1 Appendix A System Power Management

In USB, operations related to the enumeration and power control signals (D+/D-) for reset and suspend from the host are also prescribed, in addition to normal transfer operations. This Appendix provides information about the specifications of USB 2.0 PHY to be connected and clock control on the system level required for processes related to the D+/D- signals. For details of each process, please be sure to check the USB Specification Revision 2.0, USB-I/O specification.

The words in Appendix A are described below.

1. Reset:

   The operation of the D+/D- signals for initializing the USB device (hereafter called "the device") from the USB host (hereafter called "the host"). After reset, enumeration is performed and then normal transfer operations such as Bulk transfers begin. Upon being connected, the device is always reset. The device also needs to support reset operation at any other arbitrary timing.

2. Suspend

   If no bus activity on the D+/D- lines including SOF is initiated by the host for 3 ms or longer, the device needs to be put in the suspend mode to reduce power consumption. In this case, the device is required to perform certain operations such as stopping the clock.

3. Resume

   The operation of the D+/D- signals for resuming normal operation from the suspend mode. Resume operation can be initiated either by the host or the device. Resume operation from the device is called "remote wakeup".

The each operation is described below. The time in ( ) is value in the USB 2.0 Specification.

14.9.1.1 Connect / Disconnect Operations

(1) Connect Operation

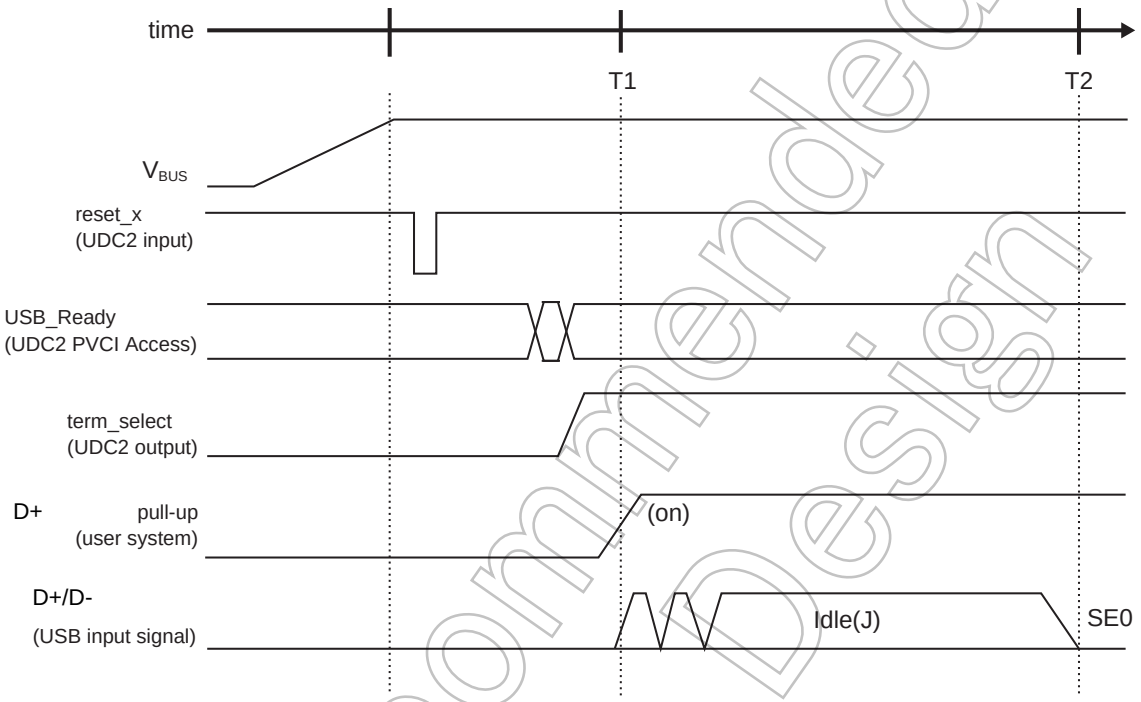


Figure 14-26 Connect operation timing

- T0: VBUS detection

  When Vbus is detected, a system reset (reset_x input) should be applied to UDC2.

  xcvr_select is "High" and term_select is "Low".
- T1: Device connect (no later than 100ms after T0)

  The device must enable D+ no later than 100 ms after Vbus detection (T0) to notify the host of the connected state. Therefore, when Vbus is detected and the device is ready to communicate with the host, the system should access the UDFS2CMD in UDC2 to set the USB_Ready command. After that, the user system sets the port using software to enable the D+ pull-up.
- T2:USB Reset Start (more than 100ms after 100ms)

(2) Disconnect Operation

When a disconnected state is detected, it is recommended to apply a system reset to UDC2.

14.9.1.2 Reset Operation

The "reset" here refers to the "Reset Signaling" defined in the USB 2.0 Specification, not the system reset (reset_x) to UDC2.

(1) When Operation in FS Mode after Reset

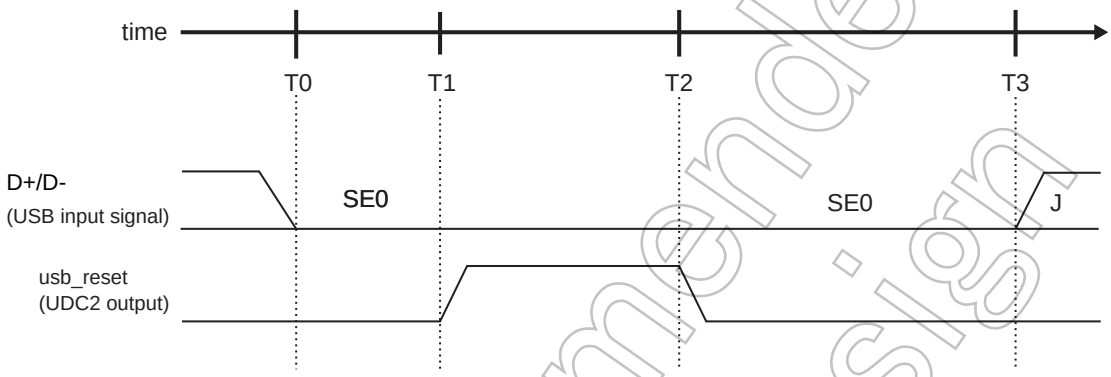


Figure 14-27 Reset Operation Timing

- T0: Reset start

  Upon recognizing SE0 from the host, UDC2 starts counting to recognize the reset.
- T1: Reset recognition (more than 2.5 μs after T0)

  When UDC2 detects SE0 for more than approximately 68 μs after T0, it recognizes the reset from the host and drives usb_reset "High".
- T2: deassert of USB reset

  At this point, usb_reset is driven "Low" more than 3.5ms from T1.
- T3: Reset end (more than 10ms after T0)

  When SE0 from the host finishes and the device enters an idle state, it indicates the end of reset operation. The reset period from the host lasts a minimum of 10 ms.

(2) Notes on Reset Operation

- Initialization of registers after reset

  When the reset from the host is completed (when usb_reset changes from "High" to "Low"), all the internal registers of UDC2 are initialized (For the initial value of each register, refer to "14.4 Registers").

  Note that registers that are set while usb_reset is "High" are also initialized. Therefore, the UDC2 registers should be set after the reset period is completed.
- DMA transfer (EP-I/F access) after reset

  When a reset from the host occurs during DMA transfer, the UDFS2EPxSTS is initialized and the bus access mode is set to "common bus access". Therefore, DMA transfer cannot be continued properly. When a reset occurs, the DMA controller must also be initialized.

  In the enumeration operation after reset, configure each EP and then initialize the EPs by setting the EP_Reset command in the UDFS2CMD.

14.9.1.3 Suspend Operation

(1) Suspend operation

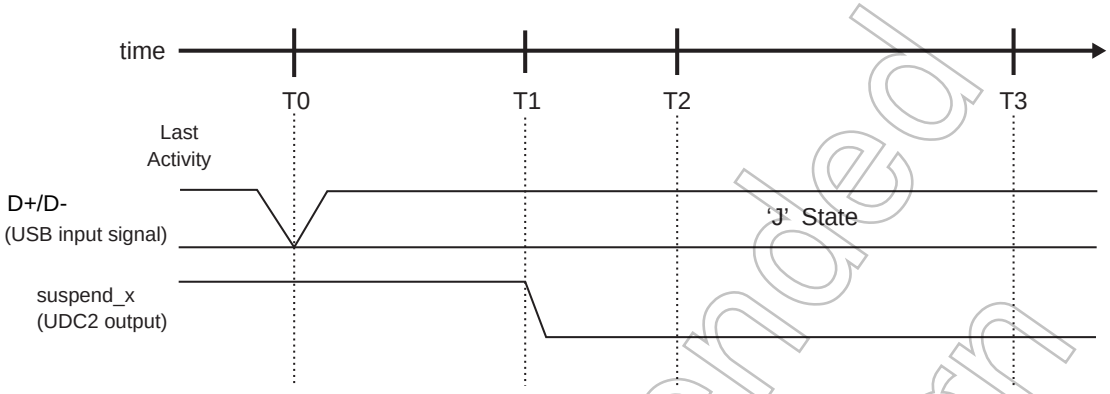


Figure 14-28 Suspend operation timing

- T0: End of bus activity

  When the end of bus activity from the host (the end of packet) is detected, UDC2 starts counting to recognize suspend.
- T1: Recognition of suspend (3 ms after T0)

  When the "FS-J" is detected for more than 3 ms after T0, UDC2 recognizes suspend and drives suspend_x "Low".
- T2: Remote wakeup start enable (5 ms after T0)

  Resume operation from the device (remote wakeup) is enabled 5 ms after T0.
- T3: Transition to suspend state (10 ms after T0)

  The device must enter the suspend state no later than 10 ms after T0. Processes required of the device system to enter the suspend state, such as stopping the CLK_U, must be performed during this period.

  It is necessary to control Clock /mode control circuit to stop the CLK_U to UDC2.

(2) Notes on Suspend Operation

- Internal registers during the suspend state

  During the suspend state, UDC2 retains the internal register values, the contents of FIFOs, and the state of each flag. These values and states are also retained after the suspend state is exited by resume operation.

  When the CLK_H to UDC2 is stopped, the internal registers in UDC2 cannot be accessed via PVC-I/F and EP-I/F.

14.9.1.4 Resume Operation

(1) Resume Operation by the Host

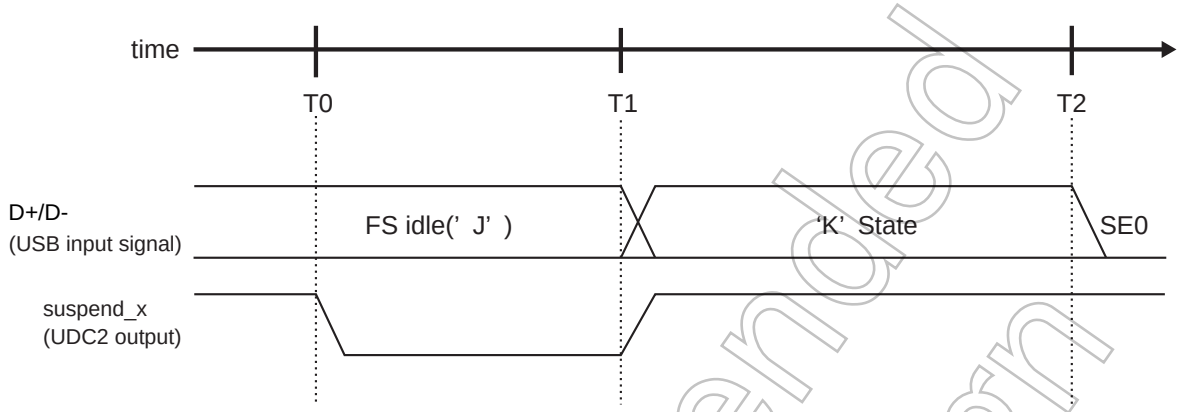


Figure 14-29 Resume operation timing by the host

- T0: suspend_x output of UDC2 is "Low".
- T1: Start of host resume (No timing specifications)

  The host starts resume operation ("FS-K") at arbitrary timing to wake up the device from the suspend state. At this point, UDC2 sets suspend_x to "High". (Even if the CLK_U to UDC2 is stopped, suspend_x becomes "High").

  During suspend, when CLK_H to UDC2 stops, resume the CLK_H by controlling clock/ mode control circuit .

  When CLK to UDC2 is stopped, it is necessary to control clk_em.
- T2: End of host resume (more than 20 ms after T1)

  The host resume operation ("FS-K") lasts for more than 20 ms, and completes after "SE0".

(2) Resume Operation by the Device (Remote Wakeup)

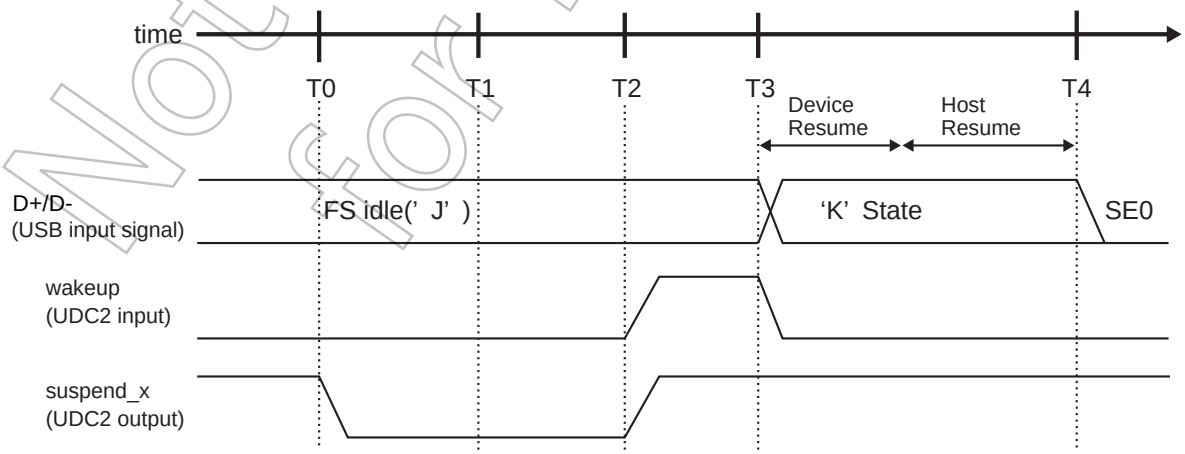


Figure 14-30 Remote wakeup operation timing

- T0: suspend_x output of UDC2 is "Low".
- T1: Remote wakeup start enable (more than 2 ms after T0)

  The device can be brought out of the suspend state by using the wakeup input of UDC2. Note that the USB specification prohibits remote wakeup for 5 ms after start of the suspend state. The wakeup signal should be set to "High" a minimum of 2 ms after T0 as 3 ms have already elapsed from the start of suspend operation to T0.
- T2: Wakeup input to UDC2 is "High" (after T1)

  Set the wakeup signal to "High". No timing requirements are specified for this operation. At this point, UDC2 sets suspend_x to "High". (Even if the CLK_H input to UDC2 is stopped, suspend_x becomes "High".) UDC2 requires the clock input to start resume operation ("FSK"). Then, keep wakeup at "High" until clock supply is resumed.
- T3: Start of device resume

  When the CLK_H input to UDC2 is resumed, UDC2 starts the device resume ("FS-K"). The device resume period is approximately 2 ms. After confirming the device resume, the host starts the host resume operation.
- T4: End of host resume (more than 20 ms after T3)

  The host resume operation ("FS-K") lasts for more than 20 ms, and completes after "SE0".

(3) Notes on Resume Operation

The restriction on use of remote wakeup are shown as follows.

To support remote wakeup as the device system, the device must notify the host in the Configuration descriptor that the remote wakeup function is enabled. Even if remote wakeup is supported, it is disabled by default. Remote wakeup can only be used after it is enabled by a request from the host. Use of remote wakeup using the wakeup input is allowed only when these conditions are satisfied.

When using this function, be sure to refer to 14.8 of the USB 2.0 Specification which offers detailed description.

14.9.2 Appendix B About Setting an Odd Number of Bytes as MaxPacketSize

14.9.2.1 Setting an odd number in the UDFS2EPxMSZ

The USB specification allows MaxPacketSize (hereafter referred to as MPS) of each EP to be set as either an odd or even number of bytes for Isochronous and Interrupt transfers. (For Control and Bulk transfers, only an even number can be set.)

In UDC2, MPS is set through UDFS2EPxMSZ<max_pkt>. The EP FIFOs of UDC2 only support even numbers of bytes. It is therefore recommended that MSP be set as an even number of bytes as a general rule.

When using MPS by odd bytes, it is possible to make <max_pkt> into odd number. However, there are restrictions shown in Table 14-6 by the access method of a bus. In the case of EP direct access, an odd number cannot be set in <max_pkt> for a transmit EP. In this case, an even number should be set in <max_pkt> and write accesses to the EP FIFO should be controlled to implement an odd number of maximum write bytes. (For example, when MPS is 1023 bytes, <max_pkt> should be set to 1024 bytes.)

Table 14-6 Restrictions on the setting of max_pkt

	Receive EP	Transmit EP
Common bus access (PVCI-IF)	An odd or even number can be set	An odd or even number can be set
EP direct access (EP-I/F)	An odd or even number can be set	Only an even number can be set.

Based on the above, the following pages describe how to set an odd number of bytes as MPS for each bus access method.

(1) Receive EP and common bus access

Either an odd or even number of bytes can be set in <max_pkt>. The access method is the same for both cases.

(2) Transmit EP and common bus access

Either an odd or even number of bytes can be set in <max_pkt>.

However, the following points must be observed in making common bus accesses for writing the maximum number of bytes with max_pkt = odd number.

The following shows an example in which <max_pkt> = 5 and the maximum number of bytes (5 bytes) are to be written.

- In the last access (5th byte), make sure that udc_be = 01.
- Because it is access of MPS, Do not issue the EP_EOP command in the UDFS2CMD.

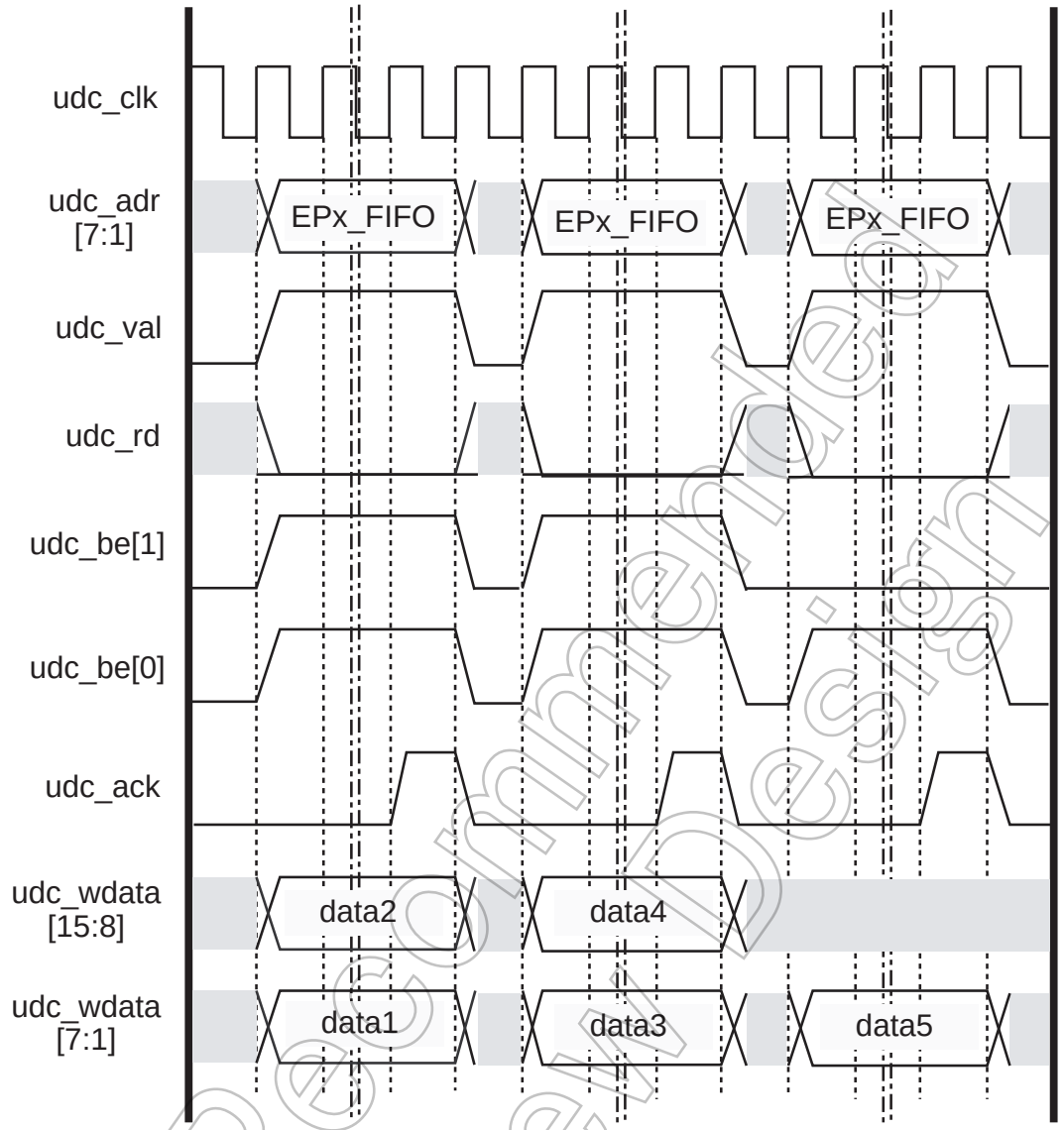


Figure 14-31 MPS write access with max_pkt = odd number (common bus access)

(3) Receive EP and EP direct access

Either an odd or even number can be set in <max_pkt>. The access method is the same for both cases.

(4) Transmit EP and EP direst access

Only an even number of bytes can be set in <max_pkt>. To use an odd number of bytes as MPS for a transmit EP, the following settings are required.

- When MPS is 1023
  - Set <max_pkt> is 1024.
  - The maximum number of bytes that can be written to the EP is 1023 bytes. (It is not allowed to write the 1024th byte.)
  - "wMaxPacketSize" of the EP descriptor to be managed by firmware should be set to 1023. (This is the value to be sent to the USB host by the Get Descriptor request.)

The following shows an example in which max_pkt = 1024 and the maximum number of bytes (1023 bytes) are to be written.

- In the last access (1023rd byte), make sure that epx_w_be = 01.

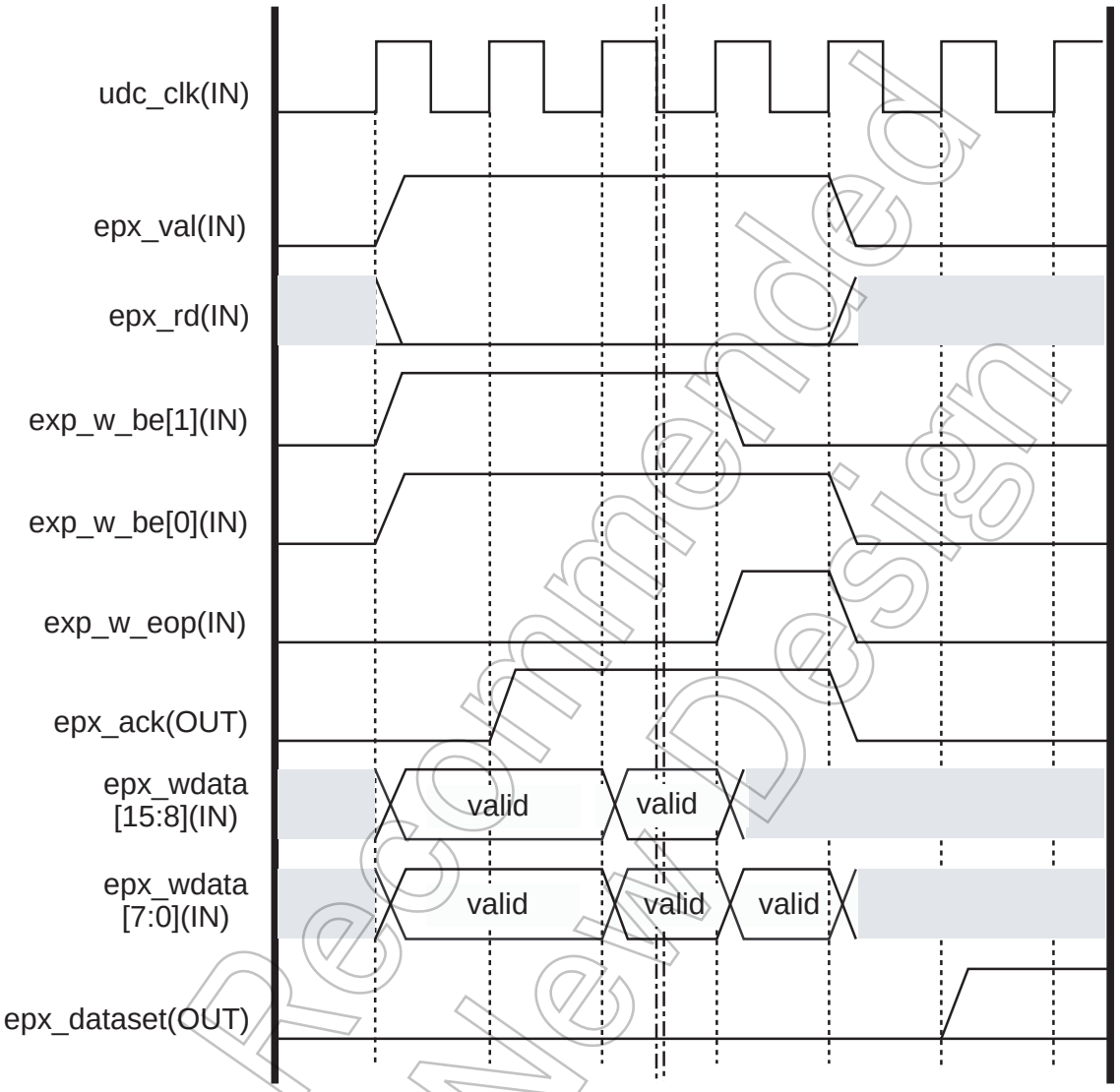


Figure 14-32 MPS (odd number) write access with max_pkt = even number (EP direct access)

14.9.3 Appendix C Isochronous Translator

In Isochronous transfers, the isochronism of data is critical and transfers occur per frame. Therefore, accesses to an EP (FIFO) using Isochronous transfers require a certain level of performance (speed). In UDC2, the access method to each EP can be selected from PVCI-I/F and EP-I/F. The FIFO configuration can be selected from Single mode and Dual mode. However, for an EP using Isochronous transfers, it is recommended to use EP-IF and Dual mode.

14.9.3.1 Accessing an EP using Isochronous transfer

The maximum data payload size is 1023 bytes in FS mode. To transfer 1023 bytes using Dual mode, 2048 bytes of RAM are required. Transfers are performed per frame (1 ms) in FS mode. In FS mode, One transactions can be made in one frame.

(Information such as the payload size and the number of transactions must be set in the relevant UDC2 register. This information must also be managed by software as the EP descriptor information to be sent to the host.)

14.9.3.2 Restrictions on command usage to EP when using Isochronous transfer

Compared to other transfers, Isochronous transfers have certain restrictions on handshake, toggle, the number of transactions in a frame, etc., limiting the types of commands that can be used. As a general rule, commands must not be issued to EPs during Isochronous transfers. While a request is being processed, the EP_Reset or EP_Invalid command may be used as necessary.

(When using PVCI-I/F as the EP access method, use the EP_EOP command.)

(About the Appendix)

For descriptions concerning the USB Specification, be sure to check the USB Specification (revision 2.0).

15. Serial Channel (SIO/UART)

15.1 Overview

This device has two mode for the serial channel, one is the synchronous communication mode (I/O interface mode), and the other is the 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 1/2, 1/8, 1/32, 1/128.
  - Make it possible to divide from the prescaler output clock frequency into 1-16.
  - Make it possible to divide from the prescaler output clock frequency into 1, N+m/16 (N=2-15, m=1-15), 16. (only UART mode)
  - The usable system clock (only UART mode).
- Double Buffer /FIFO

  The usable double buffer function, and the usable FIFO buffers of transmit and receive in all for maximum 4-byte.
- I/O Interface Mode
  - Transfer Mode: the half duplex (transmit/receive), the full duplex
  - Clock: Output (fixed rising edge) /Input (selectable rising/falling edge)
  - Make it possible to specify the interval time of continuous transmission.
- 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 $\overline{\text{CTS}}$ pin

In the following explanation, "x" represents channel number.

15.2 Difference in the Specifications of SIO Modules

TMPM366FDXBG/FYXBG/FWXBG has two SIO channels.

Each channel functions independently. The used pins, interrupt, DMA request and UART source clock in each channel are collected in the following.

Table 15-1 Difference in the Specifications of SIO Modules

	Pin name			Interrupt		DMA request	UART source clock
	TXD	RXD	$\overline{\text{CTSx}}$/SCLKx	Receive Interrupt	Transmit Interrupt		
Channel 0	PE0	PE1	PE2	INTRX0	INTTX0	Support	TB8OUT
Channel 1	PC0	PC1	PC2	INTRX1	INTTX1	Support	TB8OUT

15.3 Configuration

Figure 15-1 shows SIO block diagram.

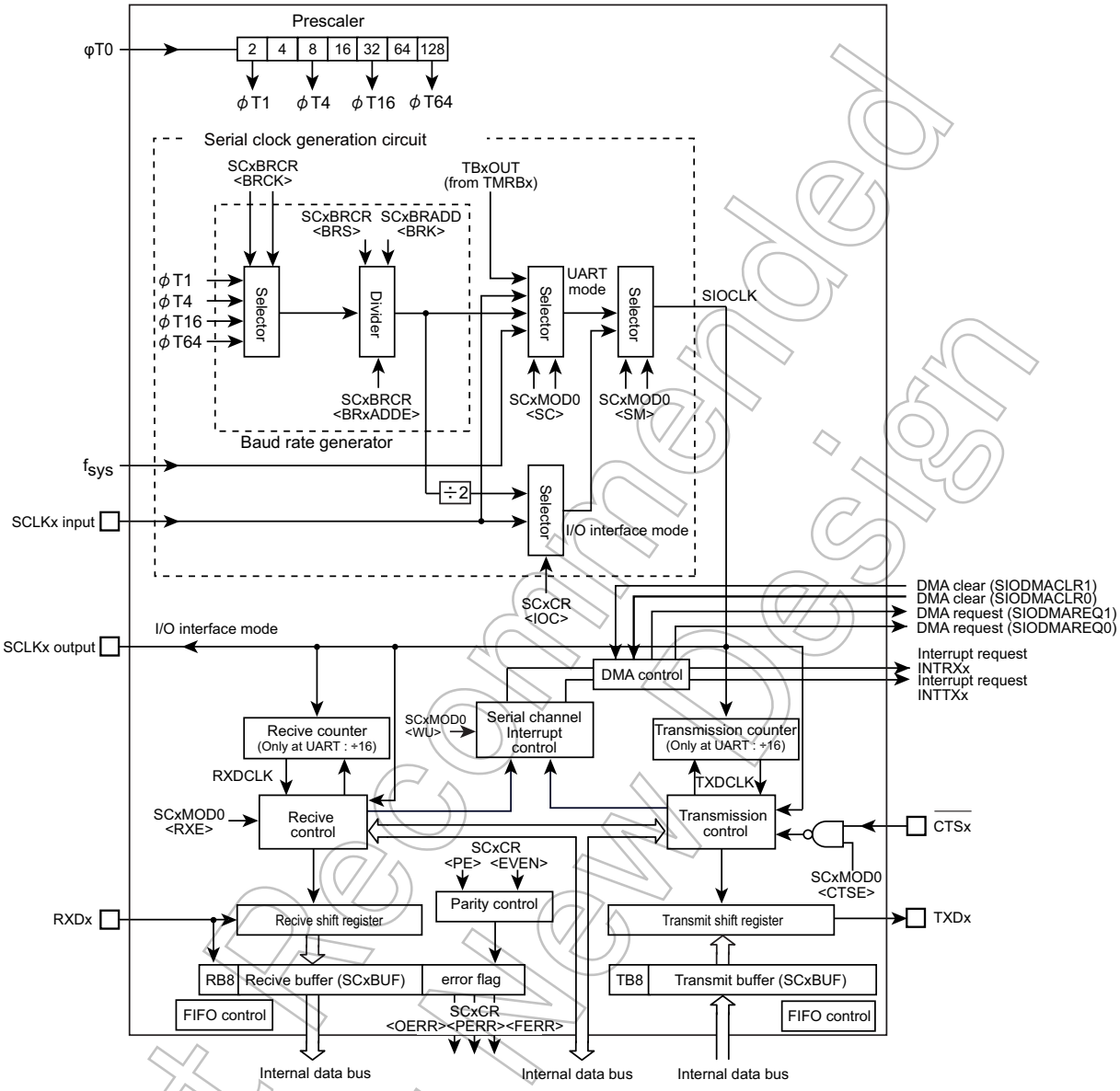


Figure 15-1 SIO Block Diagram

15.4 Registers Description

15.4.1 Registers List in Each Channel

The each channel registers and addresses are shown below.

Channel x	Base Address
Channel0	0x400E_1000
Channel1	0x400E_1100

Register name (x=0 to 1)		Address (Base+)
Enable register	SCxEN	0x0000
Buffer register	SCxBUF	0x0004
Control register	SCxCR	0x0008
Mode control register 0	SCxMOD0	0x000C
Baud rate generator control register	SCxBRCR	0x0010
Baud rate generator control register 2	SCxBRADD	0x0014
Mode control register 1	SCxMOD1	0x0018
Mode control register 2	SCxMOD2	0x001C
RX FIFO configuration register	SCxRFC	0x0020
TX FIFO configuration register	SCxTFC	0x0024
RX FIFO status register	SCxRST	0x0028
TX FIFO status register	SCxTST	0x002C
FIFO configuration register	SCxFCNF	0x0030
DMA request enable register	SCxDMA	0x0034

Note 1: Do not modify any control register when data is being transmitted or received.

15.4.2 SCxEN (Enable Register)

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

Bit	Bit Symbol	Type	Function
31-1	−	R	Read as 0.
0	SIOE	R/W	SIO operation 0: Disabled 1: Enabled Specified the SIO operation. To use the SIO, set <SIOE> = "1". When the operation is disabled, no clock is supplied to the other registers in the SIO module. This can reduce the power consumption. If the SIO operation is executed and then disabled, the settings will be maintained in each register except for SCxTFC<TIL>.

Note: **In case that SCxEN<SIOE>="0" (Stop SIO operation) or the operation mode is changed to IDLE mode with SCxMOD1<I2SC>="0" (Stop SIO operation in IDLE mode), SCxTFC is initialzed again.**

15.4.3 SCxBUF (Buffer Register)

SCxBUF works as a transmit buffer or FIFO for write operation and as a receive buffer or FIFO for read operation.

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7-0	TB[7:0] / RB [7:0]	R/W	[write] TB : Transmit buffer / FIFO [read] RB : Receive buffer / FIFO

15.4.4 SCxCR (Control Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	RB8	R	Receive data bit 8 (For UART) 9th bit of the received data in the 9 bits UART mode.
6	EVEN	R/W	Parity (For UART) 0: Odd 1: Even Selects even or odd parity. "0" : odd parity, "1" : even parity. The parity bit may be used only in the 7- or 8-bit UART mode.
5	PE	R/W	Add parity (For UART) 0: Disabled 1: Enabled Controls enabling/ disabling parity. The parity bit may be used only in the 7- or 8-bit UART mode.
4	OERR	R	Overrun 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
1	SCLKS	R/W	Selecting input clock edge (For I/O Interface) Set to "0" in the clock output mode. 0: Data in the transmit buffer is sent to TXDx pin one bit at a time on the falling edge of SCLKx. Data from RXDx pin isrecieved in the recieve buffer one bit at a time on the rising edge of SCLKx. In this case, the SCLKx starts from high level. 1: Data in the transmit buffer is sent to TXDx pin one bit at a time on the rising edge of SCLKx. Data from RXDx pin isrecieved in the recieve buffer one bit at a time on the falling edge of SCLKx. In this case, the SCLKx starts from low level.
0	IOC	R/W	Selecting clock (For I/O Interface) 0: Baud rate generator 1: SCLK pin input

Note: **Any error flag (OERR, PERR, FERR) is cleared to "0" when read.**

15.4.5 SCxMOD0 (Mode Control Register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TB8	CTSE	RXE	WU	SM		SC	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	TB8	R/W	Transmit data bit 8 (For UART) Writes the 9th bit of transmit data in the 9 bits UART mode.
6	CTSE	R/W	Handshake function control (For UART) 0: CTS disabled 1: CTS enabled Controls handshake function. Setting "1" enables handshake function using $\overline{\text{CTS}}$ pin.
5	RXE	R/W	Receive control (Note) 0: Disabled 1: Enabled
4	WU	R/W	Wake-up function (For UART) 0: Disabled 1: Enabled This function is available only at 9-bit UART mode. In other mode, this function has no meaning. In it is Enabled, Interrupt only when RB9 = "1" at 9-bit UART mode.
3-2	SM[1:0]	R/W	Specifies transfer mode. 00: I/O interface mode 01: 7-bit length UART mode 10: 8-bit length UART mode 11: 9-bit length UART mode
1-0	SC[1:0]	R/W	Serial transfer clock (For UART) 00: Timer TB8OUT 01: Baud rate generator 10: Internal clock fsys 11: External clock (SCLK input) (As for the I/O interface mode, the serial transfer clock can be set in the control register (SCxCR).

Note: With <RXE> set to "0", set each mode register (SCxMOD0, SCxMOD1 and SCxMOD2). Then set <RXE> to "1".

Note: Do not stop the receive operation (by setting SCxMOD0<RXE> = "0") when data is being received.

15.4.6 SCxMOD1 (Mode Control Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	I2SC	FDPX		TXE	SINT			-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	I2SC	R/W	IDLE 0: Stop 1: Operate Specifies the IDLE mode operation.
6-5	FDPX[1:0]	R/W	Transfer mode setting 00: Transfer prohibited 01: Half duplex (Receive) 10: Half duplex (Transmit) 11: Full duplex Configures the transfer mode in the I/O interface mode. Also configures the FIFO if it is enabled. In the UART mode, it is used only to specify the FIFO configuration.
4	TXE	R/W	Transmit control (Note2) 0 :Disabled 1: Enabled This bit enables transmission and is valid for all the transfer modes.
3-1	SINT[2:0]	R/W	Interval time of continuous transmission (For I/O interface) 000: None 001: 1SCLK 010: 2SCLK 011: 4SCLK 100: 8SCLK 101: 16SCLK 110: 32SCLK 111: 64SCLK This parameter is valid only for the I/O interface mode when SCLK pin output is selected. In other modes, this function has no meaning. Specifies the interval time of continuous transmission when double buffering or FIFO is enabled in the I/O interface mode.
0	−	R/W	Write a "0".

Note 1: **Specify the all mode first and then enable the <TXE> bit.**

Note 2: **Do not stop the transmit operation (by setting <TXE> = "0") when data is being transmitted.**

Note 3: **In case that SCxEN<SIOE>="0" (Stop SIO operation) or the operation mode is changed to IDLE mode with SCxMOD1<I2SC>="0" (Stop SIO operation in IDLE mode), SCxTFC is initiailzed again.**

15.4.7 SCxMOD2 (Mode Control Register 2)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	TBEMP	R	Transmit buffer empty flag. 0: Full 1: Empty If double buffering is disabled, this flag is insignificant. This flag shows that the transmit double buffers are empty. When data in the transmit double buffers is moved to the transmit shift register and the double buffers are empty, this bit is set to "1". Writing data again to the double buffers sets this bit to "0".
6	RBFLL	R	Receive buffer full flag. 0: Empty 1: Full This is a flag to show that the receive double buffers are full. When a receive operation is completed and received data is moved from the receive shift register to the receive double buffers, this bit changes to "1" while reading this bit changes it to "0". If double buffering is disabled, this flag is insignificant.
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.
4	SBLEN	R/W	STOP bit (for UART) 0 : 1-bit 1 : 2-bit This specifies the length of transmission stop bit in the UART mode. On the receive side, the decision is made using only a single bit regardless of the <SBLEN> setting.
3	DRCHG	R/W	Setting transfer direction 0: LSB first 1: MSB first Specifies the direction of data transfer in the I/O interface mode. In the UART mode, set this bit to LSB first.

<TXRUN>	<TBEMP>	Status
1	−	Transmission in progress
0	1	Transmission completed
	0	Wait state with data in Transmit buffer

Bit	Bit Symbol	Type	Function
2	WBUF	R/W	Double-buffer 0: Disabled 1 : Enabled This parameter enables or disables the transmit/receive double buffers to transmit (in both SCLK output/input modes) and receive (in SCLK output mode) data in the I/O interface mode and to transmit data in the UART mode. When receiving data in the I/O interface mode (SCLK input) and UART mode, double buffering is enabled in both cases that 0 or 1 is set to <WBUF> bit.
1-0	SWRST[1:0]	R/W	Software reset Overwriting "01" in place of "10" generates a software reset. When this software reset is executed, the following bits are initialized and the transmit/receive circuit, the transmit circuit and the FIFO become initial state (see Note1 and Note2).

Register	Bit
SCxMOD0	<RXE>
SCxMOD1	<TXE>
SCxMOD2	<TBEMP>, <RBFLL>, <TXRUN>
SCxCR	<OERR>, <PERR>, <FERR>
SCxDMA (note2)	<DMAEN1>, <DMAEN0>

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.

15.4.8 SCxBRCR (Baud Rate Generator Control Register), SCxBRADD (Baud Rate Generator Control Register 2)

The division ratio of the baud rate generator can be specified in the registers shown below.

SCxBRCR

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	−	R/W	Write "0".
6	BRADDE	R/W	N + (16 − K)/16 divider function (For UART) 0: disabled 1: enabled This division function can only be used in the UART mode.
5-4	BRCK[1:0]	R/W	Select input clock to the baud rate generator. 00: φT1 01: φT4 10: φT16 11: φT64
3-0	BRS[3:0]	R/W	Division ratio "N" 0000: 16 0001: 1 0010: 2 ... 1111: 15

SCxBRADD

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

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as 0.
3-0	BRK[3:0]	R/W	Specify K for the "N + (16 − K)/16" division (For UART) 0000: Prohibited 0001: K = 1 0010: K = 2 ... 1111: K = 15

Table 15-2 lists the settings of baud rate generator division ratio.

Table 15-2 Setting division ratio

	<BRADDE> = "0"	<BRADDE> = "1" (Note1) (Only UART mode)
<BRS>	Specify "N" (Note2) (Note3)	
<BRK>	No setting required	Specify "K" (Note4)
Division ratio	Divide by N	$N + \frac{(16 - K)}{16}$ 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: **As a division ratio, 1 ("0001") or 16 ("0000") can not be applied to N when using the "N + (16 - K)/16" division function in the UART mode.**

Note 3: **The division ratio "1" of the baud rate generator can be specified only when the double buffering is used in the I/O interface mode.**

Note 4: **Specifying "K = 0" is prohibited.**

15.4.9 SCxFCNF (FIFO Configuration Register)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0
7-5	-	R/W	Be sure to write "000"
4	RFST	R/W	Bytes used in RX FIFO 0:Maximum 1:Same as FILL level of RX FIFO When RX FIFO is enabled, the number of RX FIFO bytes to be used is selected (Note1) 0: The maximun number of bytes of the FIFO configured (see also <CNFG>). 1: Same as the fill level for receive interrupt generation specified by SCxRFC <RIL[1:0]>
3	TFIE	R/W	TX interrupt for TX FIFO 0: Disabled 1:Enabled When TX FIFO is enabled, transmit interrupts are enabled or disabled by this parameter.
2	RFIE	R/W	RX interrupt for RX FIFO 0: Disabled 1:Enabled When RX FIFO is enabled, receive interrupts are enabled or disabled by this parameter.
1	RXTXCNT	R/W	Automatic disable of RXE/TXE 0: None 1: Auto disabled Controls automatic disabling of transmission and reception. Setting "1" enables to operate as follows Half duplex RX: When receive shift register, the receive buffer and the RX FIFO are filled, SCxMOD0<RXE> is automatically set to "0" to inhibit further reception. Half duplex TX: When the TX FIFO, the transmit buffer and the transmit shift register is empty, SCxMOD1<TXE> is automatically set to "0" to inhibit further transmission. Full duplex: When either of the above two conditions is satisfied, TXE/RXE are automatically set to "0" to inhibit further transmission and reception.
0	CNFG	R/W	Enables FIFO. 0: Disabled 1: Enabled If enabled, the SCxMOD1 <FDPX[1:0]> setting automatically configures FIFO as follows: (The type of TX/RX can be specified in the mode control register 1 SCxMOD1<FDPX[1:0]>). Half duplex RX: RX FIFO 4byte Half duplex TX: TX FIFO 4byte Full duplex: RX FIFO 2byte + TX FIFO 2byte

Note 1: **Regarding TX FIFO, the maximum number of bytes being configured is always available. The available number of bytes is the bytes already written to the TX FIFO.**

Note 2: **The FIFO can not use in 9bit UART mode.**

15.4.10 SCxRFC (RX FIFO Configuration Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	RFCS	W	RX FIFO clear (Note) 1: Clear When SCxRFC<RFCS> is set to "1", the receive FIFO is cleared and SCxRST<RLVL> is "000". And also the read pointer is initialized.
6	RFIS	R/W	Select interrupt generation condition 0: when the data reaches to the specified fill level. 1: when the data reaches to the specified fill level or the data exceeds the specified fill level at the time data is read.
5-2	−	R	Read as 0.
1-0	RIL[1:0]	R/W	FIFO fill level to generate RX interrupts

	Half duplex	Full duplex
00	4byte	2byte
01	1byte	1byte
10	2byte	2byte
11	3byte	1byte

Note: To use TX/RX FIFO buffer, TX/RX FIFO must be cleared after setting the SIO transfer mode (half duplex/full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").

15.4.11 SCxTFC (TX FIFO Configuration Register) (Note2)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	TFCS	W	TX FIFO clear (Note 1) 1: Clears TX FIFO. When SCxTST<TFCS> is set to "1", the transmit FIFO is cleared and SCxTST<TLVL> is "000". And also the write pointer is initialized.
6	TFIS	R/W	Selects interrupt generation condition. 0: An interrupt is generated when the data reaches to the specified fill level. 1: An interrupt is generated when the data reaches to the specified fill level or the data can not reach the specified fill level at the time new data is read.
5-2	−	R	Read as 0.
1-0	TIL[1:0]	R/W	Selects FIFO fill level.

	Other than full duplex	Full duplex
00	Empty	Empty
01	1 byte	1 byte
10	2 byte	Empty
11	3 byte	1 byte

Note 1: **To use TX/RX FIFO buffer, TX/RX FIFO must be cleared after setting the SIO transfer mode (half duplex/ full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").**

Note 2: **After you perform the following operations, configure the SCxTFC register again.**

SCxEN<SIOE> = "0" (SIO operation stop)

Conditions are as follows:SCxMOD1<I2SC> = "0" (operation is prohibited in IDLE mode) and releasing the low power consumption mode which started by the WFI (Wait For Interrupt) instruction.

15.4.12 SCxRST (RX FIFO Status Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	ROR	R	RX FIFO Overrun (Note) 0: Not generated 1: Generated
6-3	−	R	Read as 0.
2-0	RLVL[2:0]	R	Status of RX FIFO fill level. 000: Empty 001: 1 byte 010: 2 byte 011: 3 byte 100: 4 byte

Note: **The <ROR> bit is cleared to "0" when receive data is read from the SCxBUF register.**

15.4.13 SCxTST (TX FIFO Status Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	TUR	R	TX FIFO under run (Note) 0: Not generated 1: Generated.
6-3	−	R	Read as 0.
2-0	TLVL[2:0]	R	Status of TX FIFO fill level. 000: Empty 001: 1 byte 010: 2 byte 011: 3 byte 100: 4 byte

Note: **The <TUR> bit is cleared to "0" when transmit data is written to the SCxBUF register.**

15.4.14 SCxDMA (DMA request 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	-	-	-	-	-	-	DMAEN1	DMAEN0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as 0.
1	DMAEN1	R/W	Enable DMA request. DMA request is generated by receive interrupt INTRX. 0: disable 1: enable
0	DMAEN0	R/W	Enable DMA request. DMA request is generated by receive interrupt INTTX. 0: disable 1: enable

Note 1: **When DMA request is generated during DMA transfer is being, it is not kept and nesting.**

15.5 Operation in Each Mode

Table 15-3 shows the modes and data formats.

Table 15-3 Mode and Data format

Mode	Mode type	Data length	Transfer direction	Specifies whether to use parity bits.	STOP bit length (transmit)
Mode 0	Synchronous communication mode (IO interface mode)	8 bit	LSB first/MSB first	-	-
Mode 1	Asynchronous communication mode (UART mode)	7 bit	LSB first	o	1 bit or 2 bit
Mode 2		8 bit		o	
Mode 3		9 bit		×	

Mode 0 is a synchronous communication and can be used to extend I/O. This mode transmits and receives data in synchronization with SCLK. SCLK can be used for both input and output.

The direction of data transfer can be selected from LSB first and MSB first. This mode is not allowed either to use parity bits or STOP bits.

The mode 1, mode 2 and mode 3 are asynchronous modes and the transfer direction is fixed to the LSB first.

Parity bits can be added in the mode 1 and mode 2. The mode 3 has a 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.

15.6 Data Format

15.6.1 Data Format List

Figure 15-2 shows data format.

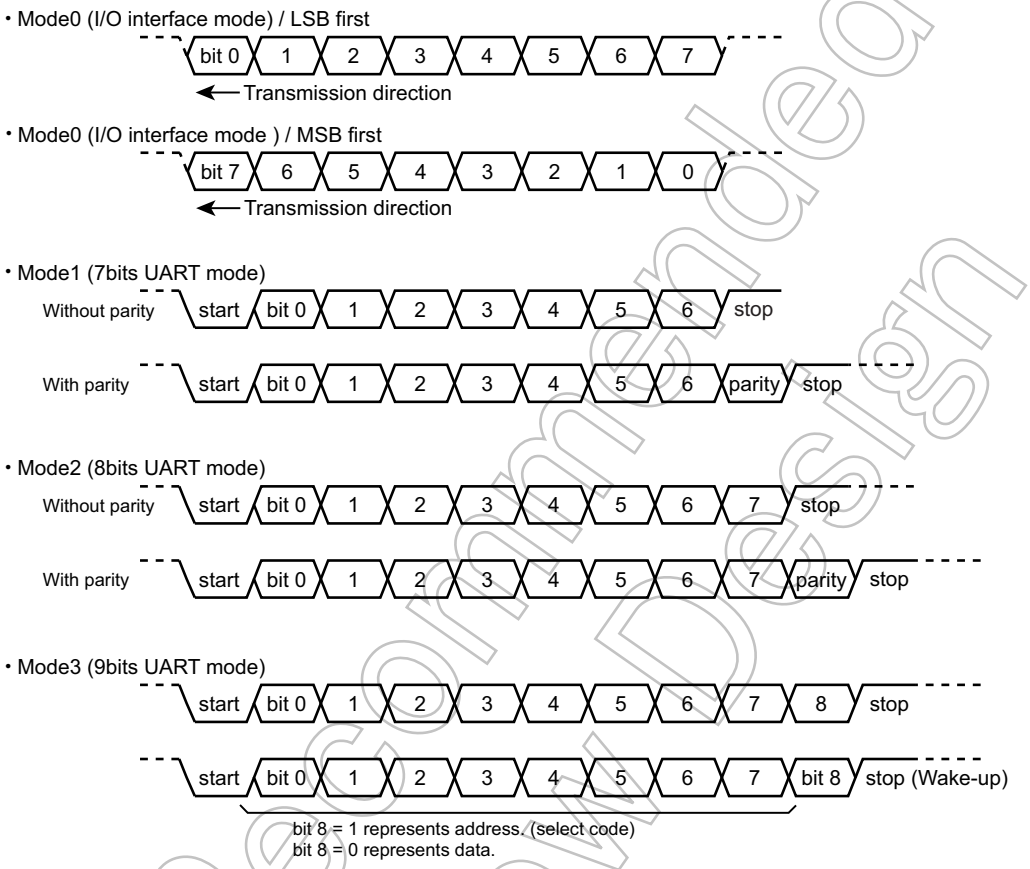


Figure 15-2 Data Format

15.6.2 Parity Control

The parity bit can be added only in the 7- or 8-bit UART mode.

Setting "1" to SCxCR<PE> enables the parity.

The <EVEN> bit of SCxCR selects either even or odd parity.

15.6.2.1 Transmission

Upon data transmission, the parity control circuit automatically generates the parity with the data in the transmit buffer.

After data transmission is complete, the parity bit will be stored in SCxBUF<TB7> in the 7-bit UART mode 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.

15.6.2.2 Receiving Data

If the received data is moved from the receive shift register to the receive buffer, a parity is generated.

In the 7-bit UART mode, the generated parity is compared with the parity stored in SCxBUF<RB7>, while in the 8-bit UART mode, it is compared with the one in SCxCR<RB8>.

If there is any difference, a parity error occurs and the <PERR> of the SCxCR register is set to "1".

In use of the FIFO, <RERR> indicates that a parity error was generated in one of the received data.

15.6.3 STOP Bit Length

The length of the STOP bit in the UART transmission mode can be selected from one bit or two bits by setting the SCxMOD2<SBLEN>. The length of the STOP bit data is determined as one-bit when it is received regardless of the setting of this bit.

15.7 Clock Control

15.7.1 Prescaler

There is a 7-bit prescaler to divide a prescaler input clock ΦT0 by 2, 8, 32 and 128.

Use the CGSYSCR register in the clock/mode control block to select the input clock ΦT0 of the prescaler.

The prescaler becomes active only when the baud rate generator is selected as a transfer clock by SCxMOD0<SC[1:0]> = "01".

The below tables show the resolution of the input clock to the baud rate generator.

Table 15-4 Clock Resolution to the Baud Rate Generator fc = 40 MHz

peripheral clock selection CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Prescaler clock selection CGSYSCR <PRCK[2:0]>	Prescaler output clock resolution			
			φT1	φT4	φT16	φT64
0 (fgear)	000 (fc)	000 (fperiph/1)	fc/2^1 (0.05 μs)	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)
		001 (fperiph/2)	fc/2^2 (0.1 μs)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)
		010 (fperiph/4)	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)
		011 (fperiph/8)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)
		100 (fperiph/16)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		101 (fperiph/32)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)
	100 (fc/2)	000 (fperiph/1)	fc/2^2 (0.1 μs)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)
		001 (fperiph/2)	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)
		010 (fperiph/4)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)
		011 (fperiph/8)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		100 (fperiph/16)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)
		101 (fperiph/32)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)	fc/2^{13} (204.8 μs)
	101 (fc/4)	000 (fperiph/1)	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)
		001 (fperiph/2)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)
		010 (fperiph/4)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		011 (fperiph/8)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)
		100 (fperiph/16)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)	fc/2^{13} (204.8 μs)
		101 (fperiph/32)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)	fc/2^{14} (409.6 μs)
	110 (fc/8)	000 (fperiph/1)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)
		001 (fperiph/2)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		010 (fperiph/4)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)
		011 (fperiph/8)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)	fc/2^{13} (204.8 μs)
		100 (fperiph/16)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)	fc/2^{14} (409.6 μs)
		101 (fperiph/32)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)	fc/2^{13} (204.8 μs)	fc/2^{15} (819.2 μs)
	111 (fc/16)	000 (fperiph/1)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		001 (fperiph/2)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)
		010 (fperiph/4)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)	fc/2^{13} (204.8 μs)
		011 (fperiph/8)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)	fc/2^{14} (409.6 μs)
		100 (fperiph/16)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)	fc/2^{13} (204.8 μs)	fc/2^{15} (819.2 μs)
		101 (fperiph/32)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)	fc/2^{14} (409.6 μs)	fc/2^{16} (1638 μs)

Table 15-4 Clock Resolution to the Baud Rate Generator fc = 40 MHz

peripheral clock selection CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Prescaler clock selection CGSYSCR <PRCK[2:0]>	Prescaler output clock resolution			
			φT1	φT4	φT16	φT64
1 (fc)	000 (fc)	000 (fperiph/1)	fc/2^1 (0.05 μs)	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)
		001 (fperiph/2)	fc/2^2 (0.1 μs)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)
		010 (fperiph/4)	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)
		011 (fperiph/8)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)
		100 (fperiph/16)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		101 (fperiph/32)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)
	100 (fc/2)	000 (fperiph/1)	–	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)
		001 (fperiph/2)	fc/2^2 (0.1 μs)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)
		010 (fperiph/4)	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)
		011 (fperiph/8)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)
		100 (fperiph/16)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		101 (fperiph/32)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)
	101 (fc/4)	000 (fperiph/1)	–	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)
		001 (fperiph/2)	–	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)
		010 (fperiph/4)	fc/2^3 (0.2 μs)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)
		011 (fperiph/8)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)
		100 (fperiph/16)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		101 (fperiph/32)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)
	110 (fc/8)	000 (fperiph/1)	–	–	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)
		001 (fperiph/2)	–	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)
		010 (fperiph/4)	–	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)
		011 (fperiph/8)	fc/2^4 (0.4 μs)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)
		100 (fperiph/16)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		101 (fperiph/32)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)
	111 (fc/16)	000 (fperiph/1)	–	–	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)
		001 (fperiph/2)	–	–	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)
		010 (fperiph/4)	–	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)
		011 (fperiph/8)	–	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)
		100 (fperiph/16)	fc/2^5 (0.8 μs)	fc/2^7 (3.2 μs)	fc/2^9 (12.8 μs)	fc/2^{11} (51.2 μs)
		101 (fperiph/32)	fc/2^6 (1.6 μs)	fc/2^8 (6.4 μs)	fc/2^{10} (25.6 μs)	fc/2^{12} (102.4 μs)

Note 1: **The prescaler output clock φTn must be selected so that the relationship "φTn ≤ fsys / 2" is satisfied (so that φTn is slower than fsys / 2).**

Note 2: **Do not change the clock gear while SIO is operating.**

Note 3: **The dashes in the above table indicate that the setting is prohibited.**

Table 15-5 Clock Resolution to the Baud Rate Generator fc = 48 MHz

peripheral clock selection CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Prescaler clock selection CGSYSCR <PRCK[2:0]>	Prescaler output clock resolution			
			φT1	φT4	φT16	φT64
0 (fgear)	000 (fc)	000 (fperiph/1)	fc/2^1 (0.0417 μs)	fc/2^3 (0.167 μs)	fc/2^5 (0.667 μs)	fc/2^7 (2.67 μs)
		001 (fperiph/2)	fc/2^2 (0.0833 μs)	fc/2^4 (0.333 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		010 (fperiph/4)	fc/2^3 (0.167 μs)	fc/2^5 (0.667 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)
		011 (fperiph/8)	fc/2^4 (0.333 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)
		100 (fperiph/16)	fc/2^5 (0.667 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)
		101 (fperiph/32)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)	fc/2^{12} (85.3 μs)
	100 (fc/2)	000 (fperiph/1)	fc/2^2 (0.0833 μs)	fc/2^4 (0.333 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)
		001 (fperiph/2)	fc/2^3 (0.167 μs)	fc/2^5 (0.667 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)
		010 (fperiph/4)	fc/2^4 (0.333 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)
		011 (fperiph/8)	fc/2^5 (0.667 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)
		100 (fperiph/16)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)	fc/2^{12} (85.3 μs)
		101 (fperiph/32)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)	fc/2^{13} (171 μs)
	101 (fc/4)	000 (fperiph/1)	fc/2^3 (0.167 μs)	fc/2^5 (0.667 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)
		001 (fperiph/2)	fc/2^4 (0.333 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)
		010 (fperiph/4)	fc/2^5 (0.667 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)
		011 (fperiph/8)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)	fc/2^{12} (85.3 μs)
		100 (fperiph/16)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)	fc/2^{13} (171 μs)
		101 (fperiph/32)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)	fc/2^{12} (85.3μs)	fc/2^{14} (341 μs)
	110 (fc/8)	000 (fperiph/1)	fc/2^4 (0.333 μs)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)
		001 (fperiph/2)	fc/2^5 (0.667 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)
		010 (fperiph/4)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)	fc/2^{12} (85.3 μs)
		011 (fperiph/8)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)	fc/2^{13} (171 μs)
		100 (fperiph/16)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)	fc/2^{12} (85.3μs)	fc/2^{14} (341 μs)
		101 (fperiph/32)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)	fc/2^{13} (171 μs)	fc/2^{15} (683 μs)
	111 (fc/16)	000 (fperiph/1)	fc/2^5 (0.667 μs)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)
		001 (fperiph/2)	fc/2^6 (1.33 μs)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)	fc/2^{12} (85.3 μs)
		010 (fperiph/4)	fc/2^7 (2.67 μs)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)	fc/2^{13} (171 μs)
		011 (fperiph/8)	fc/2^8 (5.33 μs)	fc/2^{10} (21.3 μs)	fc/2^{12} (85.3μs)	fc/2^{14} (341 μs)
		100 (fperiph/16)	fc/2^9 (10.7 μs)	fc/2^{11} (42.7 μs)	fc/2^{13} (171 μs)	fc/2^{15} (683 μs)
		101 (fperiph/32)	fc/2^{10} (21.4 μs)	fc/2^{12} (85.4 μs)	fc/2^{14} (341 μs)	fc/2^{16} (1366 μs)

Table 15-5 Clock Resolution to the Baud Rate Generator fc = 48 MHz

peripheral clock selection CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Prescaler clock selection CGSYSCR <PRCK[2:0]>	Prescaler output clock resolution			
			φT1	φT4	φT16	φT64
1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0417 μs)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
		001 (fperiph/2)	$fc/2^2$ (0.0833 μs)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		010 (fperiph/4)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
		011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
		100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
		101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
	100 (fc/2)	000 (fperiph/1)	–	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
		001 (fperiph/2)	$fc/2^2$ (0.0833 μs)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		010 (fperiph/4)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
		011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
		100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
		101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
	101 (fc/4)	000 (fperiph/1)	–	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
		001 (fperiph/2)	–	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		010 (fperiph/4)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
		011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
		100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
		101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
	110 (fc/8)	000 (fperiph/1)	–	–	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
		001 (fperiph/2)	–	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		010 (fperiph/4)	–	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
		011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
		100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
		101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
	111 (fc/16)	000 (fperiph/1)	–	–	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
		001 (fperiph/2)	–	–	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		010 (fperiph/4)	–	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
		011 (fperiph/8)	–	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
		100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
		101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)

Note 1: **The prescaler output clock φTn must be selected so that the relationship "φTn ≤ fsys / 2" is satisfied (so that φTn is slower than fsys / 2).**

Note 2: **Do not change the clock gear while SIO is operating.**

Note 3: **The dashes in the above table indicate that the setting is prohibited.**

15.7.2 Serial Clock Generation Circuit

The serial clock circuit is a block to generate transmit and receive clocks (SIOCLK) and consists of the circuits in which clocks can be selected by the settings of the baud rates generator and modes.

15.7.2.1 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 2, 8, 32 and 128.

This input clock is selected by setting the SCxBRCR<BRCK>.

(2) Baud Rate Generator output clock

The frequency division ratio of the output clock in the baud rate generator is set by SCxBRCR and SCxBRADD.

The following frequency divide ratios can be used; 1/N frequency division in the I/O interface mode ,either 1/N or N + (16-K)/16 in the UART mode.

The table below shows the frequency division ratio which can be selected.

Mode	Divide Function Setting SCxBRCR<BRADDE>	Divide by N SCxBRCR<BRS>	Divide by K SCxBRADD<BRK>
I/O interface	Divide by N	1 to 16 (Note)	-
UART	Divide by N	1 to 16	-
	N + (16-K)/16 division	2 to 15	1 to 15

Note: 1/N (N=1)frequency division ratio can be used only when a double buffer is enabled.

15.7.2.2 Clock Selection Circuit

A clock can be selected by setting the modes and the register.

Modes can be specified by setting the SCxMOD0<SM>.

The input clock in I/O interface mode is selected by setting SCxCR. The clock in UART mode is selected by setting SCxMOD0<SC>.

(1) Transfer Clock in I/O interface mode

Table 15-6 shows clock selection in I/O interface mode.

Table 15-6 Clock Selection in I/O Interface Mode

Mode SCxMOD0<SM>	Input/Output selection SCxCR<IOC>	Clock edge selection SCxCR<SCLKS>	Clock of use
I/O interface mode	SCLK output	Set to "0". (Fixed to the rising edge)	Divided by 2 of the baud rate generator output.
	SCLK input	Rising edge	SCLK input rising edge
		Falling edge	SCLK input falling edge

To get the highest baud rate, the baud rate generator must be set as below.

Note: **When deciding clock settings, make sure that AC electrical character is satisfied.**

- Clock/mode control block settings
  - fc = 40MHz
  - fgear = 40MHz (CGSYSCR<GEAR[2:0]> = "000" : fc selected)
  - φT0 = 40MHz (CGSYSCR<PRCK[2:0]> = "000" : 1 division ratio)
- SIO settings (if double buffer is used)
  - Clock (SCxBRCR<BRCK[1:0]> = "00" : φT1 selected) = 20MHz
  - Divided clock frequency (SCxBRCR<BRS[3:0]> = "0001" : 1 division ratio) = 20MHz

  1 division ratio can be selected if double buffer is used. In this case, baud rate is 10Mbps because 20MHz is divided by 2.
- SIO settings (if double buffer is not used)
  - Clock (SCxBRCR<BRCK[1:0]> = "00" : φT1 selected) = 20MHz
  - Divided clock frequency (SCxBRCR<BRS[3:0]> = "0010" : 2 division ratio) = 10MHz

  2 division ratio is the highest if double buffer is not used. In this case, baud rate is 5Mbps because 10MHz is divided by 2.

To use SCLK input, the following conditions must be satisfied.

- If double buffer is used

  - SCLK cycle > 6/fsys

  The highest baud rate is less than 48 ÷ 6 = 8 Mbps.
- If double buffer is not used

  - SCLK cycle > 8/fsys

The highest baud rate is less than 48 ÷ 8 = 6 Mbps.

(2) Transfer clock in the UART mode

Table 15-7 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 15-7 Clock Selection in UART Mode

Mode SCxMOD0<SM>	Clock selection SCxMOD0<SC>
UART Mode	Timer output
	Baud rate generator
	fsys
	SCLK input

The examples of baud rate in each clock settings.

- If the baud rate generator is used
  - fc = 40MHz
  - fgear = 40MHz (CGSYSCR<GEAR[2:0]> = "000" : fc selected)
  - φT0 = 40MHz (CGSYSCR<PRCK[2:0]> = "000" : 1 division ratio)
  - Clock = ΦT1 = 20MHz (SCxBRCR<BRCK[1:0]> = "00" : φT1 selected)

  The highest baud rate is 1.25Mbps because 20MHz is divided by 16.

  Table 15-8 shows examples of baud rate when the baud rate generator is used with the following clock settings.

  - fc = 9.8304MHz
  - fgear = 9.8304MHz (CGSYSCR<GEAR[2:0]> = "000" : fc selected)
  - φT0 = 4.9152MHz (CGSYSCR<PRCK[2:0]> = "001" : 2 division ratio)

Table 15-8 Example of UART Mode Baud Rate (Using the Baud Rate Generator)

fc [MHz]	Division ratio N (SCxBRCR<BRS>)	φT1 (fc/4)	φT4 (fc/16)	φT16 (fc/64)	φT64 (fc/256)
9.830400	2	76.800	19.200	4.800	1.200
	4	38.400	9.600	2.400	0.600
	8	19.200	4.800	1.200	0.300
	16	9.600	2.400	0.600	0.150

Unit : kbps

- If the SCLK input is used

  To use SCLK input, the following conditions must be satisfied.

  - SCLK cycle > 2/fsys

  The highest baud rate must be less than 48 ÷ 2 ÷ 16 = 1.5 Mbps.

- If fsys is used

  Since the highest value of fsys is 48MHz, the highest baud rate is 48 ÷ 16 = 3Mbps.

- If timer output is used

To enable the timer output, the following condition must be set: a timer flip-flop output inverts when the value of the counter and that of TBxRG1 match. The SIOCLK clock frequency is "Setting value of TBxRG1 × 2".

Baud rates can be obtained by using the following formula.

Baud rate calculation

$$\text{Transfer rate} = \frac{\text{Clock frequency selected by CGSYSCR<PRCK[1:0]>}}{(\text{TBxRG1} \times 2) \times 2 \times 16}$$

In the case the timer prescaler clock ΦT1 (2divition ratio) is selected.

One clock cycle is a period that the timer flip-flop is inverted twice.

Table 15-9 shows the examples of baud rates when the timer output is used with the following clock settings.

- fc = 32MHz / 9.8304MHz / 8MHz
- fgear = 32MHz / 9.8304MHz / 8MHz (CGSYSCR<GEAR[2:0]> = "000" : fc selected)
- φT0 = 16MHz / 4.9152MHz / 4MHz (CGSYSCR<PRCK[2:0]> = "001" : 2 division ratio)
- Timer count clock = 4MHz / 1.2287MHz / 1MHz (TBxMOD<TBCLK[1:0]> = "01" : φT1 selected)

Table 15-9 Example of UART Mode Baud Rate (Using the Timer Output)

TBxRG0 setting	fc		
	32MHz	9.8304MHz	8MHz
0x0001	250	76.8	62.5
0x0002	125	38.4	31.25
0x0003	-	25.6	-
0x0004	62.5	19.2	15.625
0x0005	50	15.36	12.5
0x0006	-	12.8	-
0x0008	31.25	9.6	-
0x000A	25	7.68	6.25
0x0010	15.625	4.8	-
0x0014	12.5	3.84	3.125

Unit : kbps

15.8 Transmit/Receive Buffer and FIFO

15.8.1 Configuration

Figure 15-3 shows the configuration of transmit buffer, receive buffer and FIFO.

Appropriate settings are required for using buffer and FIFO. The configuration may be predefined depending on the mode.

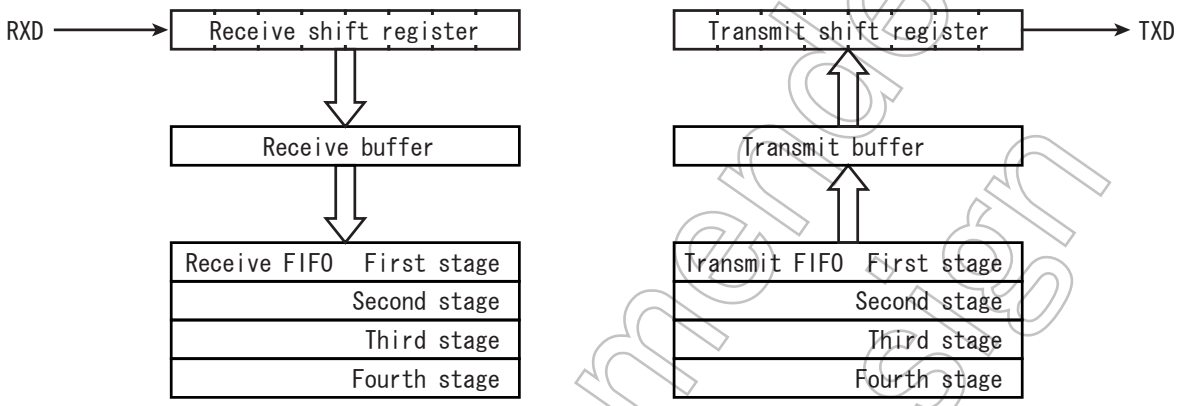


Figure 15-3 The Configuration of Buffer and FIFO

15.8.2 Transmit/Receive Buffer

Transmit buffer and receive buffer are double-buffered. The buffer configuration is specified by SCxMOD2<WBUF>.

In the case of using a receive buffer, if SCLK input is set to generate clock output in the I/O interface mode or the UART mode is selected, it's double buffered despite the <WBUF> settings. In other modes, it's according to the <WBUF> settings.

Table 15-10 shows correlation between modes and buffers.

Table 15-10 Mode and buffer Composition

Mode		SCxMOD2<WBUF>	
		"0"	"1"
UART	Transmit	Single	Double
	Receive	Double	Double
I/O interface (SCLK input)	Transmit	Single	Double
	Receive	Double	Double
I/O interface (SCLK output)	Transmit	Single	Double
	Receive	Single	Double

15.8.3 FIFO

In addition to the double buffer function above described, 4-byte FIFO can be used.

To enable FIFO, enable the double buffer by setting SCxMOD2<WBUF> to "1" and SCxFCNF<CNFG> to "1". The FIFO buffer configuration is specified by SCxMOD1<FDPX[1:0]>.

Note: To use TX/RX FIFO buffer, TX/RX FIFO must be cleared after setting the SIO transfer mode (half duplex/full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").

Table 15-11 shows correlation between modes and FIFO.

Table 15-11 Mode and FIFO Composition

	SCxMOD1<FDPX[1:0]>	RX FIFO	TX FIFO
Half duplex RX	"01"	4byte	-
Half duplex TX	"10"	-	4byte
Full duplex	"11"	2byte	2byte

15.9 Status Flag

The SCxMOD2 register has two types of flag. This bit is significant only when the double buffer is enabled.

<RBFLL> is a flag to show that the receive buffer is full. When one frame of data is received and the data is moved from the receive shift register to the receive buffers, this bit changes to "1" while reading this bit changes it to "0".

<TBEMP> shows that the transmit buffers are empty. When data in the transmit buffers is moved to the transmit shift register, this bit is set to "1" When data is set to the transmit buffers, the bit is cleared to "0".

15.10 Error Flag

Three error flags are provided in the SCxCR register. The meaning of the flags is changed depending on the modes. The table below shows the meanings in each mode.

These flags are cleared to "0" after reading the SCxCR register.

Mode	Flag		
	<OERR>	<PERR>	<FERR>
UART	Overrun error	Parity error	Framing error
I/O Interface (SCLK input)	Overrun error	Underrun error (When using double buffer or FIFO)	Fixed to 0
		Fixed to 0 (When a double buffer and FIFO unused)	
I/O Interface (SCLK output)	Undefined	Undefined	Fixed to 0

15.10.1 OERR Flag

In both UART and I/O interface modes, this bit is set to "1" when an error is generated by completing the reception of the next frame of receive data before the receive buffer has been read. If the receive FIFO is enabled, the received data is automatically moved to the receive FIFO and no overrun error will be generated until the receive FIFO is full (or until the usable bytes are fully occupied).

In the I/O interface with SCLK output mode, the SCLK output stops upon setting the flag.

Note: To switch the I/O interface SCLK output mode to other modes, read the SCxCR register and clear the overrun flag.

15.10.2 PERR Flag

This flag indicates a parity error in the UART mode and an under-run error in the I/O interface mode.

In the UART mode, <PERR> is set to "1" when the parity generated from the received data is different from the parity received.

In the I/O interface mode, <PERR> is set to "1" under the following conditions when a double buffer is enabled.

In the SCLK input mode, <PERR> is set to "1" when the SCLK is input after completing data output of the transmit shift register with no data in the transmit buffer.

In the SCLK output mode, <PERR> is set to "1" after completing output of all data and the SCLK output stops.

Note: To switch the I/O interface SCLK output mode to other modes, read the SCxCR register and clear the under-run flag.

15.10.3 FERR Flag

A framing error is generated if the corresponding stop bit is determined to be "0" by sampling the bit at around the center. Regardless of the stop bit length settings in the SCxMOD2<SBLEN>register, the stop bit status is determined by only 1.

This bit is fixed to "0" in the I/O interface mode.

15.11 Receive

15.11.1 Receive Counter

The receive counter is a 4-bit binary counter and is up-counted by SIOCLK. In the UART mode, sixteen SIOCLK clock pulses are used in receiving a single data bit and the data symbol is sampled at the seventh, eighth, and ninth pulses. From these three samples, majority logic is applied to decide the received data.

15.11.2 Receive Control Unit

15.11.2.1 I/O interface mode

In the SCLK output mode with SCxCR <IOC> set to "0", the RXD pin is sampled on the rising edge of the shift clock outputted to the SCLK pin.

In the SCLK input mode with SCxCR <IOC> set to "1", the serial receive data RXD pin is sampled on the rising or falling edge of SCLK input signal depending on the SCxCR <SCLKS> setting.

15.11.2.2 UART Mode

The receive control unit has a start bit detection circuit, which is used to initiate receive operation when a normal start bit is detected.

15.11.3 Receive Operation

15.11.3.1 Receive Buffer

The received data is stored by 1 bit in the receive shift register. When a complete set of bits has been stored, the interrupt 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 "0" cleared by reading the receive buffer.

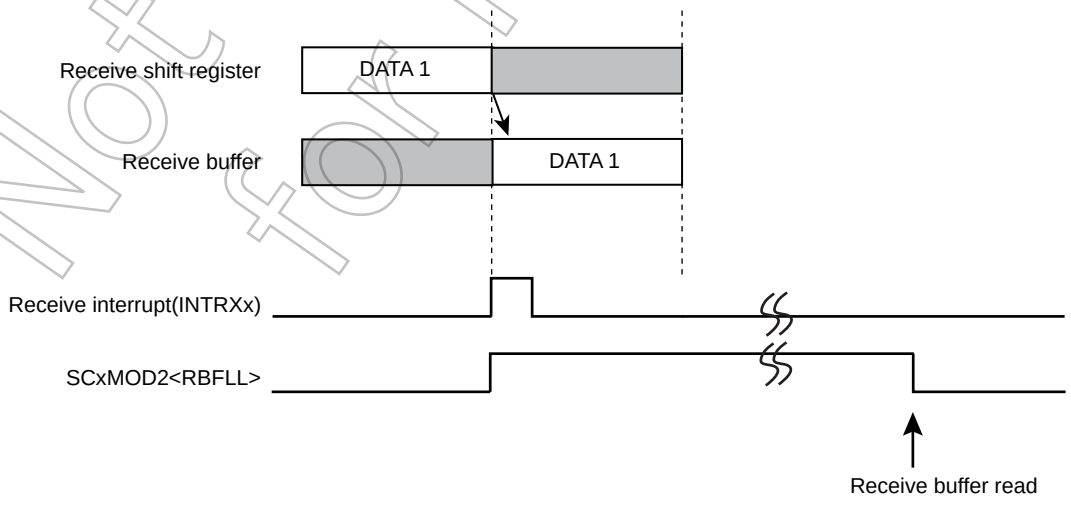


Figure 15-4 Receive Buffer Operation

15.11.3.2 Receive FIFO Operation

When FIFO is enabled, the received data is moved from receive buffer to receive FIFO and the receive buffer full flag is cleared immediately. An interrupt will be generated according to the SCxRFC<RIL> setting.

Note: When the data with parity bit are received in UART mode by using the FIFO, the parity error flag is shown the occurring the parity error in the received data.

The following describes configurations and operations in the half duplex RX mode.

SCxMOD1[6:5] =01	: Transfer mode is set to half duplex mode
SCxFCNF[4:0] = 10111	: Automatically inhibits continuous reception after reaching the fill level. : The number of bytes to be used in the receive FIFO is the same as the interrupt generation fill level.
SCxRFC[1:0] = 00	: The fill level of FIFO in which generated receive interrupt is set to 4-byte.
SCxRFC[7:6] = 11	: Clears receive FIFO and sets the condition of interrupt generation.

After setting of the above FIFO configuration, the data reception is started by writing "1" to the SCxMOD0 <RXE>. When the data is stored all in the receive shift register, receive buffer and receive FIFO, SCxMOD0<RXE> is automatically cleared and the receive operation is finished.

In this above condition, if the continuous reception after reaching the fill level is enabled, and it is possible to receive a data continuously with and reading the data in the FIFO.

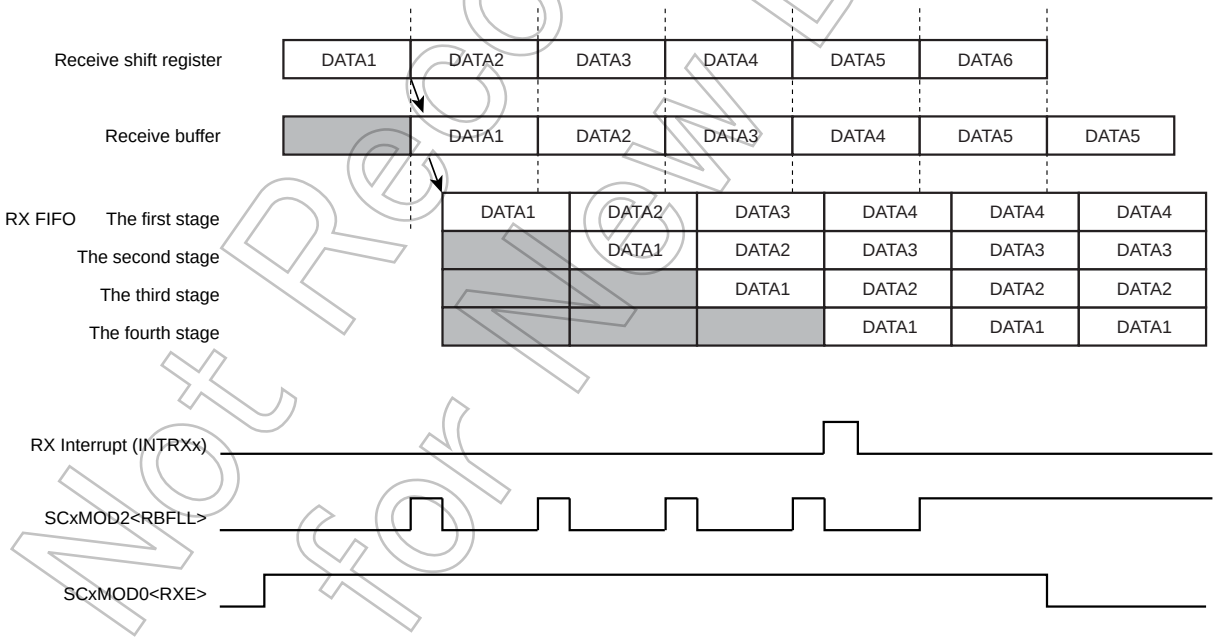


Figure 15-5 Receive FIFO Operation

15.11.3.3 I/O interface mode with SCLK output

In the I/O interface mode and SCLK output setting, SCLK output stops when all received data is stored in the receive buffer and FIFO. So, in this mode, the overrun error flag has no meaning.

The timing of SCLK output stop and re-output depends on receive buffer and FIFO.

(1) Case of single buffer

Stop SCLK output after receiving a data. In this mode, I/O interface can transfer each data with the transfer device by hand-shake.

When the data in a buffer is read, SCLK output is restarted.

(2) Case of double buffer

Stop SCLK output after receiving the data into a receive shift register and a receive buffer.

When the data is read, SCLK output is restarted.

(3) Case of FIFO

Stop SCLK output after receiving the data into a shift register, received buffer and FIFO.

When one byte data is read, the data in the received buffer is transferred into FIFO and the data in the receive shift register is transferred into received buffer and SCLK output is restarted.

And if SCxFCNF<RXTXCNT> is set to "1", SCLK stops and receive operation stops with clearing SCxMOD0<RXE> bit too.

15.11.3.4 Read Received Data

In spite of enabling or disabling FIFO, read the received data from the receive buffer (SCxBUF).

When receive FIFO is disabled, the buffer full flag SCxMOD2<RBFLL> is cleared to "0" by this reading. In the case of the next data can be received in the receive shift register before reading a data from the receive buffer. The parity bit to be added in the 8-bit UART mode as well as the most significant bit in the 9-bit UART mode will be stored in SCxCR<RB8>.

When the receive FIFO is available, the 9-bit UART mode is prohibited because up to 8-bit data can be stored in FIFO. In the 8-bit UART mode, the parity bit is lost but parity error is determined and the result is stored in SCxCR<PERR>.

15.11.3.5 Wake-up Function

In the 9-bit UART mode, the slave controller can be operated in the wake-up mode by setting the wake-up function SCxMOD0 <WU> to "1." In this case, the interrupt INTRXx will be generated only when SCxCR <RB8> is set to "1."

15.11.3.6 Overrun Error

When FIFO is disabled, the overrun error is occurred and set overrun flag without completing data read before receiving the next data. When overrun error is occurred, a content of receive buffer and SCxCR<RB8> is not lost, but a content of receive shift register is lost.

When FIFO is enabled, overrun error is occurred and set overrun flag by no reading the data before moving the next data into received buffer when FIFO is full. In this case, the content of FIFO are not lost.

In the I/O interface mode with SCLK output setting, the clock output automatically stops, so this flag has no meaning.

Note: When the mode is changed from I/O interface SCLK output mode to the other mode, read SCxCR and clear overrun flag.

15.12 Transmission

15.12.1 Transmission Counter

The transmit counter is a 4-bit binary counter and is counted by SIOCLK as in the case of the receive counter. In UART mode, it generates a transmit clock (TXDCLK) on every 16th clock pulse.

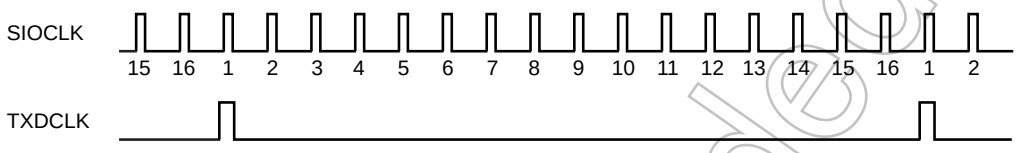


Figure 15-6 Generation of Transmission Clock

15.12.2 Transmission Control

15.12.2.1 I/O Interface Mode

In the SCLK output mode with SCxCR<IOC> set to "0", each bit of data in the transmit buffer is outputted to the TXD pin on the falling edge of the shift clock outputted from the SCLK pin.

In the SCLK input mode with SCxCR<IOC> set to "1", each bit of data in the transmit buffer is outputted to the TXD pin on the rising or falling edge of the SCLK input signal according to the SCxCR<SCLKS> setting.

15.12.2.2 UART Mode

When the transmit data is written in the transmit buffer, data transmission is initiated on the rising edge of the next TXDCLK and the transmit shift clock signal is also generated.

15.12.3 Transmit Operation

15.12.3.1 Operation of Transmission Buffer

If double buffering is disabled, the CPU writes data only to Transmit shift Buffer and the transmit interrupt INTTXx is generated upon completion of data transmission.

If double buffering is enabled (including the case the transmit FIFO is enabled), data written to the transmit buffer is moved to the transmit shift register. The INTTXx interrupt is generated at the same time and the transmit buffer empty flag (SCxMOD2<TBEMP>) is set to "1". This flag indicates that the next transmit data can be written. When the next data is written to the transmit buffer, the <TBEMP> flag is cleared to "0".

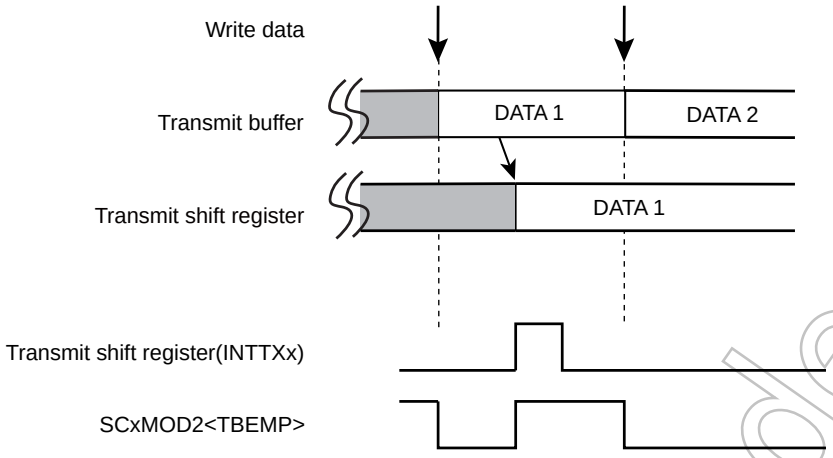


Figure 15-7 Operation of Transmission Buffer (Double-buffer is enabled)

15.12.3.2 Transmit FIFO Operation

When FIFO is enabled, the maximum 5-byte data can be stored using the transmit buffer and FIFO. Once transmission is enabled, data is transferred to the transmit shift register from the transmit buffer and start transmission. If data exists in the FIFO, the data is moved to the transmit buffer immediately, and the <TBEMP> flag is cleared to "0".

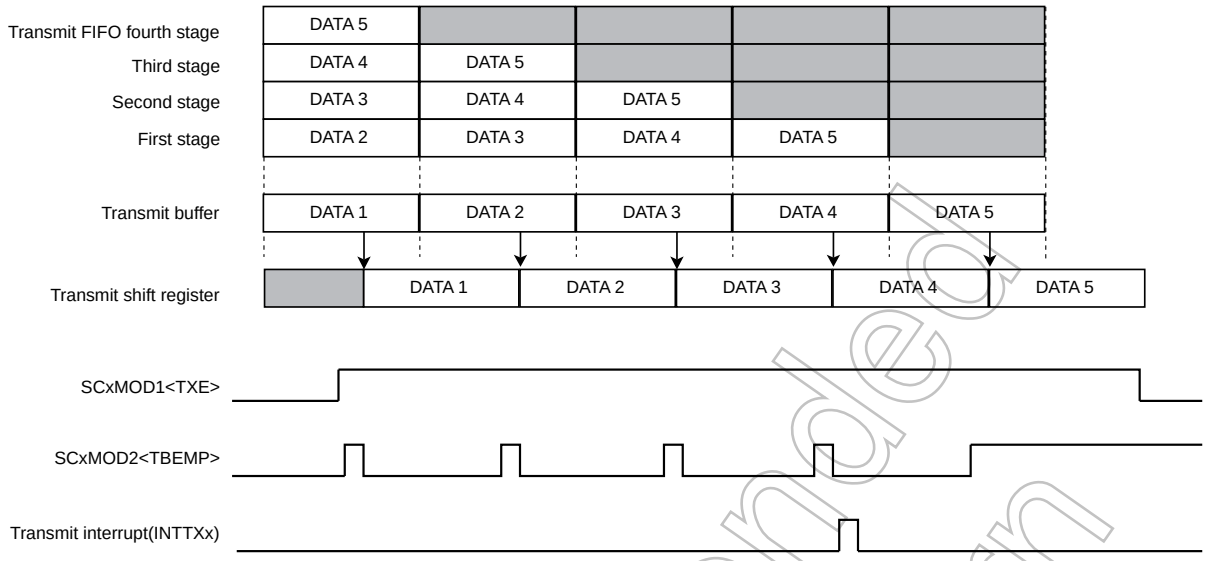
Note: **To use TX FIFO buffer, TX FIFO must be cleared after setting the SIO transfer mode (half duplex/full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").**

Settings and operations to transmit 4-byte data stream by setting the transfer mode to half duplex are shown as below.

SCxMOD1[6:5] =10	: Transfer mode is set to half duplex.
SCxFCNF[4:0] = 11011	: Transmission is automatically disabled if FIFO becomes empty. The number of bytes to be used in the receive FIFO is the same as the interrupt generation fill level.
SCxTFC[1:0] = 00	: Sets the interrupt generation fill level to "0".
SCxTFC[7:6] = 11	: Clears receive FIFO and sets the condition of interrupt generation.
SCxFCNF[0] = 1	: Enable FIFO.

After above settings are configured, data transmission can be initiated by writing 5 bytes of data to the transmit buffer or FIFO, and setting the SCxMOD1<TXE> bit to "1". When the last transmit data is moved to the transmit buffer, the transmit FIFO interrupt is generated. When transmission of the last data is completed, the clock is stopped and the transmission sequence is terminated.

Once above settings are configured, if the transmission is not set as auto disabled, the transmission should lasts by writing transmit data.



15.12.3.3 I/O interface Mode/Transmission by SCLK Output

If SCLK is set to generate clock the I/O interface mode, the SCLK output automatically stops when all data transmission is completed and underrun error will not occur.

The timing of suspension and resume of SCLK output is different depending on the buffer and FIFO usage.

(1) Single Buffer

The SCLK output stops each time one frame of data is transferred. Handshaking for each data with the other side of communication can be enabled. The SCLK output resumes when the next data is written in the buffer.

(2) Double Buffer

The SCLK output stops upon completion of data transmission of the transmit shift register and the transmit buffer. The SCLK output resumes when the next data is written in the buffer.

(3) FIFO

The transmission of all data stored in the transmit shift register, transmit buffer and FIFO is completed, the SCLK output stops. The next data is written, SCLK output resumes.

If SCxFCNF<RXTXCNT> is configured, SCxMOD0<TXE> bit is cleared at the same time as SCLK stop and the transmission stops.

15.12.3.4 Under-run error

If the transmit FIFO is disabled in the I/O interface SCLK input mode and if no data is set in transmit buffer before the next frame clock input, which occurs upon completion of data transmission from transmit shift register, an under-run error occurs and SCxCR<PERR> is set to "1".

In the I/O interface mode with SCLK output setting, the clock output automatically stops, so this flag has no meaning.

Note: **Before switching the I/O interface SCLK output mode to other modes, read the SCxCR register and clear the underrun flag.**

15.13 Handshake function

The function of the handshake is to enable frame-by-frame data transmission by using the CTS (Clear to send) pin and to prevent overrun errors. This function can be enabled or disabled by SCxMOD0<CTSE>.

When the $\overline{\text{CTS}}$ pin is set to "High" level, the current data transmission can be completed but the next data transmission is suspended until the $\overline{\text{CTS}}$ pin returns to the "Low" level. However in this case, the INTTXx interrupt is generated in the normal timing, the next transmit data is written in the transmit buffer, and it waits until it is ready to transmit data.

Note: **(1) If the $\overline{\text{CTS}}$ signal is set to "H" during transmission, the next data transmission is suspended after the current transmission is completed.**

(2) Data transmission starts on the first falling edge of the TXDCLK clock after $\overline{\text{CTS}}$ is set to "L".

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.

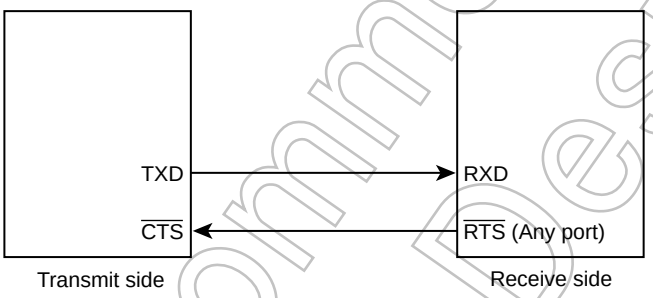


Figure 15-8 Handshake Function

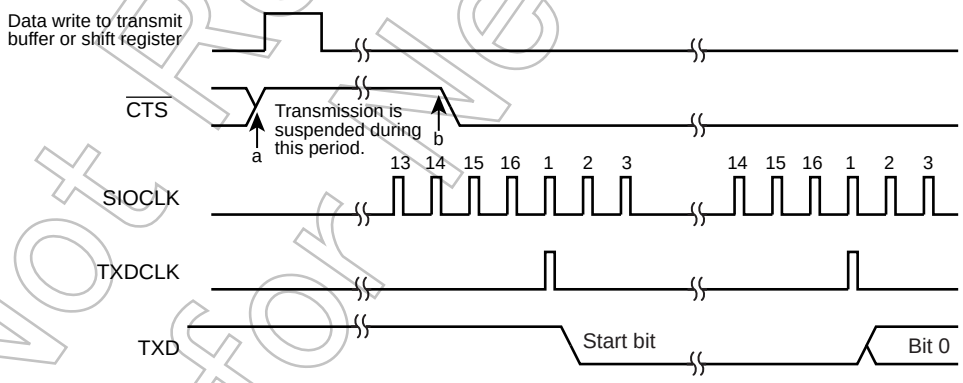


Figure 15-9 $\overline{\text{CTS}}$ Signal timing

15.14 Interrupt/Error Generation Timing

15.14.1 RX Interrupts

Figure 15-10 shows the data flow of receive operation and the route of read.

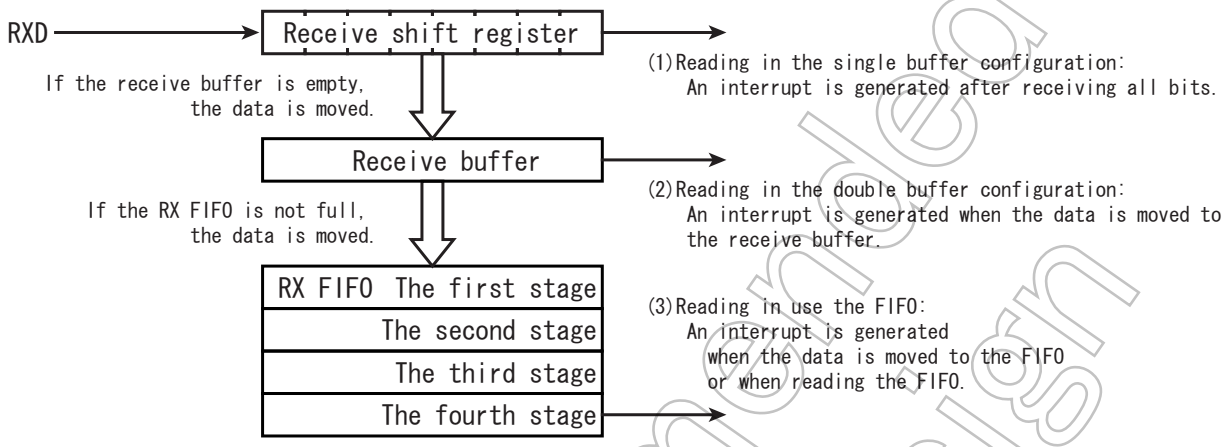


Figure 15-10 Receive Buffer/FIFO Configuration Diagram

15.14.1.1 Single Buffer / Double Buffer

RX interrupts are generated at the time depends on the transfer mode and the buffer configurations, which are given as follows.

Buffer Configurations	UART modes	IO interface modes
Single Buffer	–	・Immediately after the raising / falling edge of the last SCLK (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Double Buffer	・Around the center of the first stop bit	・Immediately after the raising / falling edge of the last SCLK (Rising or falling is determined according to SCxCR<SCLKS> setting.) ・On data transfer from the shift register to the buffer by reading buffer.

Note: Interrupts are not generated when an overrun error is occurred.

15.14.1.2 FIFO

When the FIFO is used, a receive interrupt occurs depending on the timing described in Table 15-12 and the condition specified with SCxRFC<RFIS>.

Table 15-12 Receive Interrupt conditions in use of FIFO

SCxRFC <RFIS>	Interrupt conditions	Interrupt generation timing
"0"	When FIFO fill level (SCxRST<RLVL[2:0]>) = Receive FIFO fill level to generate receive interrupt <RIL[1:0] >	・When received data is transferred from receive buffer to receive FIFO
"1"	When FIFO fill level (SCxRST<RLVL[2:0]>) ≥ Receive FIFO fill level to generate receive interrupt <RIL[1:0]>	・When received data is transferred from receive buffer to receive FIFO ・When received data is read from receive FIFO

15.14.2 TX interrupts

Figure 15-11 shows the data flow of transmit operation and the route of read.

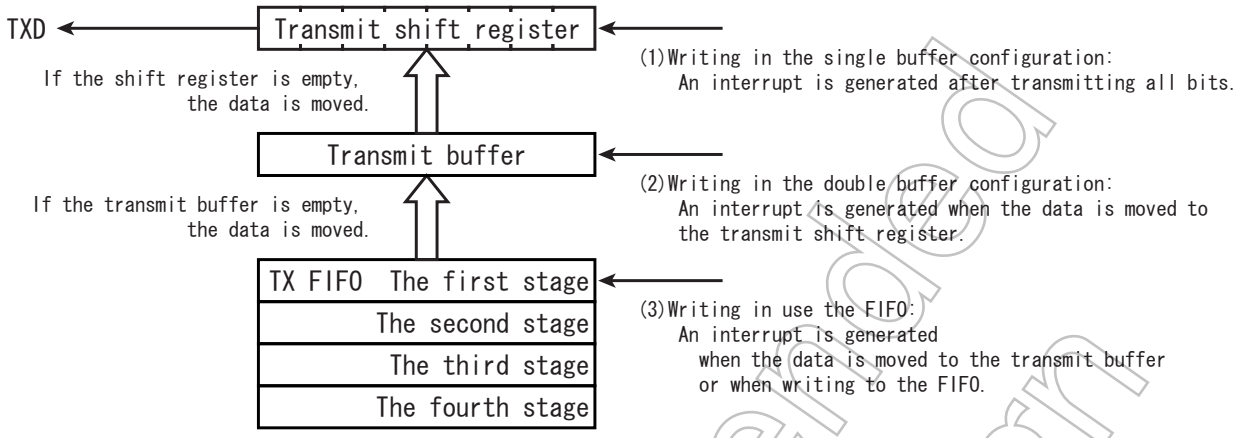


Figure 15-11 Transmit Buffer/FIFO Configuration Diagram

15.14.2.1 Single Buffer / Double Buffer

TX interrupts are generated at the time depends on the transfer mode and the buffer configurations, which are given as follows.

Buffer Configurations	UART modes	IO interface modes
Single Buffer	Just before the stop bit is sent	Immediately after the raising / falling edge of the last SCLK (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Double Buffer	When a data is moved from the transmit buffet to the transmit shift register.	

Note: If double buffer is enabled, a interrupt is also generated when the data is moved from the buffer to the shift register by writing to the buffer.

15.14.2.2 FIFO

When the FIFO is used, a transmit interrupt occurs depending on the timing described in Table 15-13 and the condition specified with SCxTFC<TFIS>.

Table 15-13 Transmit Interrupt conditions in use of FIFO

SCxTFC <TFIS>	Interrupt condition	Interrupt generation timing
"0"	When FIFO fill level (SCxTST<TLVL[2:0]>) = Transmit FIFO fill level to generate transmit interrupt <TIL[1:0] >	・When transmitted data is transferred from transmit FIFO to transmit buffer
"1"	When FIFO fill level (SCxTST<TLVL[2:0]>) ≤ Transmit FIFO fill level to generate transmit interrupt <TIL[1:0] >	・When transmitted data is transferred from transmit FIFO to transmit buffer ・When transmit data is write into transmit FIFO

15.14.3 Error Generation

15.14.3.1 UART Mode

modes	9 bits	7 bits 8 bits 7 bits+ Parity 8 bits + Parity
Framing Error Overrun Error	Around the center of stop bit	
Parity Error	-	Around the center of parity bit

15.14.3.2 IO Interface Mode

Overrun Error	Immediately after the raising / falling edge of the last SCLK (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Underrun Error	Immediately after the rising or falling edge of the next SCLK. (Rising or falling is determined according to SCxCR<SCLKS> setting.)

Note: Over-run error and Under-run error have no meaning in SCLK output mode.

15.15 Software Reset

Software reset is generated by writing SCxMOD2<SWRST[1:0]> as "10" followed by "01".

As a result, SCxMOD0<RXE>, SCxMOD1<TXE>, SCxMOD2<TBEMP><RBFLL><TXRUN>, SCxCR

<OERR><PERR><FERR> are initialized. And the receive circuit, the transmit circuit and the FIFO become initial state. Other states are maintained.

15.16 DMA request

DMA request to DMAC is generated at the timing of SIO/UART interrupt request (INTRXx, INTTXx). When DMA transfer is used, set SCxDMA (x=0, 1,).

15.17 Operation in Each Mode

15.17.1 Mode 0 (I/O interface mode)

Mode 0 consists of two modes, the SCLK output mode to output synchronous clock and the SCLK input mode to accept synchronous clock from an external source.

The following operational descriptions are for the case use of FIFO is disabled. For details of FIFO operation, refer to the previous sections describing receive/transmit FIFO functions.

15.17.1.1 Transmitting Data

(1) SCLK Output Mode

- If the transmit double buffer is disabled (SCxMOD2<WBUF> = "0")

  Data is output from the TXD pin and the clock is output from the SCLK pin each time the CPU writes data to the transmit buffer. When all data is output, an interrupt (INTTXx) is generated.

- If the transmit double buffer is enabled (SCxMOD2<WBUF> = "1")

  Data is moved from the transmit buffer to the transmit shift register when the CPU writes data to the transmit buffer while data transmission is halted or when data transmission from the transmit buffer (shift register) is completed. Simultaneously, the transmit buffer empty flag SCxMOD2<TBEMP> is set to "1", and the INTTXx interrupt is generated.

  When data is moved from the transmit buffer to the transmit shift register, if the transmit buffer has no data to be moved to the transmit shift register, INTTXx interrupt is not generated and the SCLK output stops.

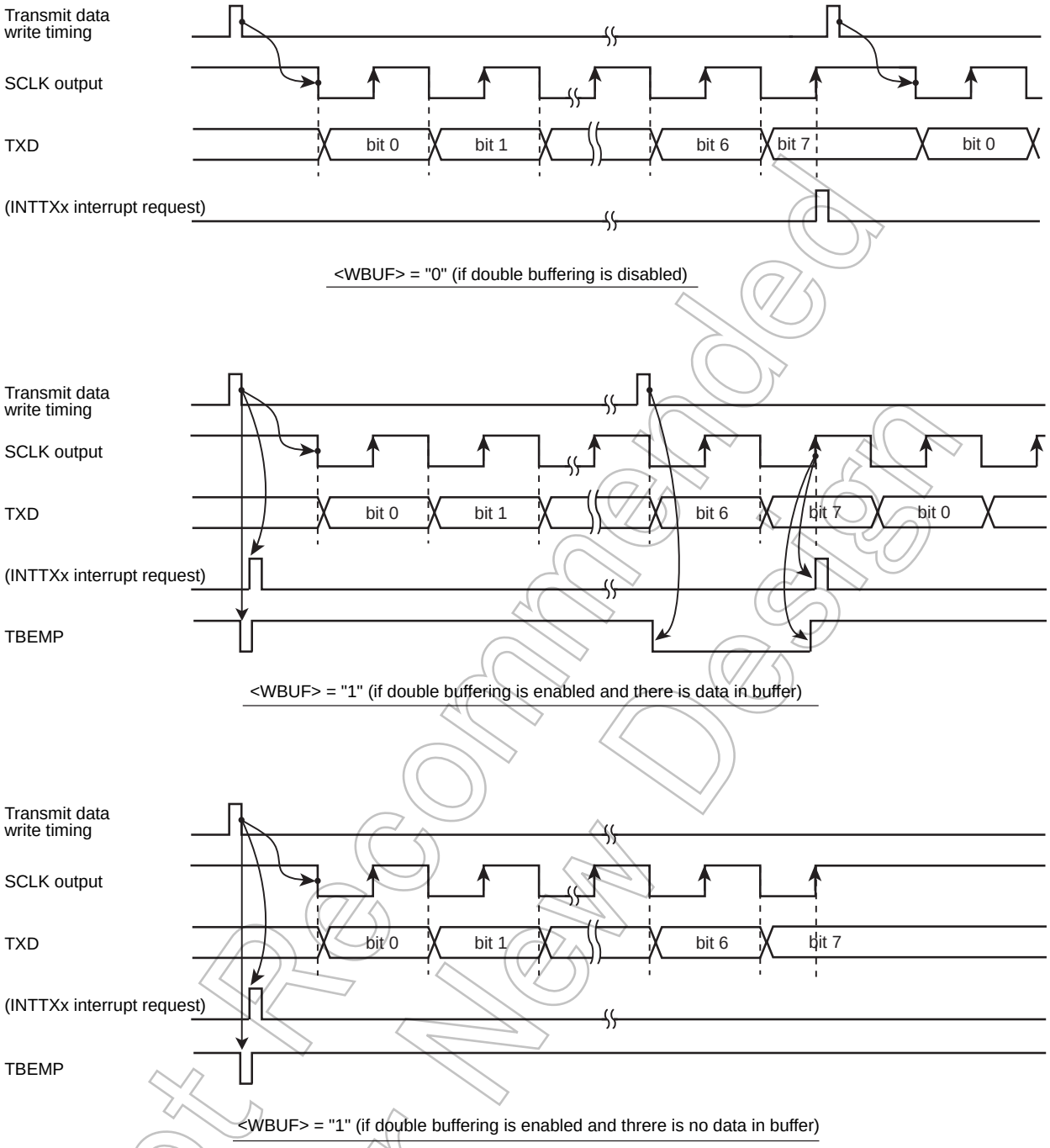


Figure 15-12 Transmit Operation in the I/O Interface Mode (SCLK Output Mode)

(2) SCLK Input Mode

- If double buffering is disabled (SCxMOD2<WBUF> = "0")

  If the SCLK is input in the condition where data is written in the transmit buffer, 8-bit data is outputted from the TXD pin. When all data is output, an interrupt INTTXx is generated. The next transmit data must be written before the timing point "A" as shown in Figure 15-13.

- If double buffer is enabled (SCxMOD2<WBUF> = "1")

  Data is moved from the transmit buffer to the transmit shift register when the CPU writes data to the transmit buffer before the SCLK input becomes active or when data transmission from the transmit shift register is completed. Simultaneously, the transmit buffer empty flag SCxMOD2<TBEMP> is set to "1", and the INTTXx interrupt is generated.

  If the SCLK input becomes active while no data is in the transmit buffer, although the internal bit counter is started, an under-run error occurs and 8-bit dummy data (0xFF) is sent.

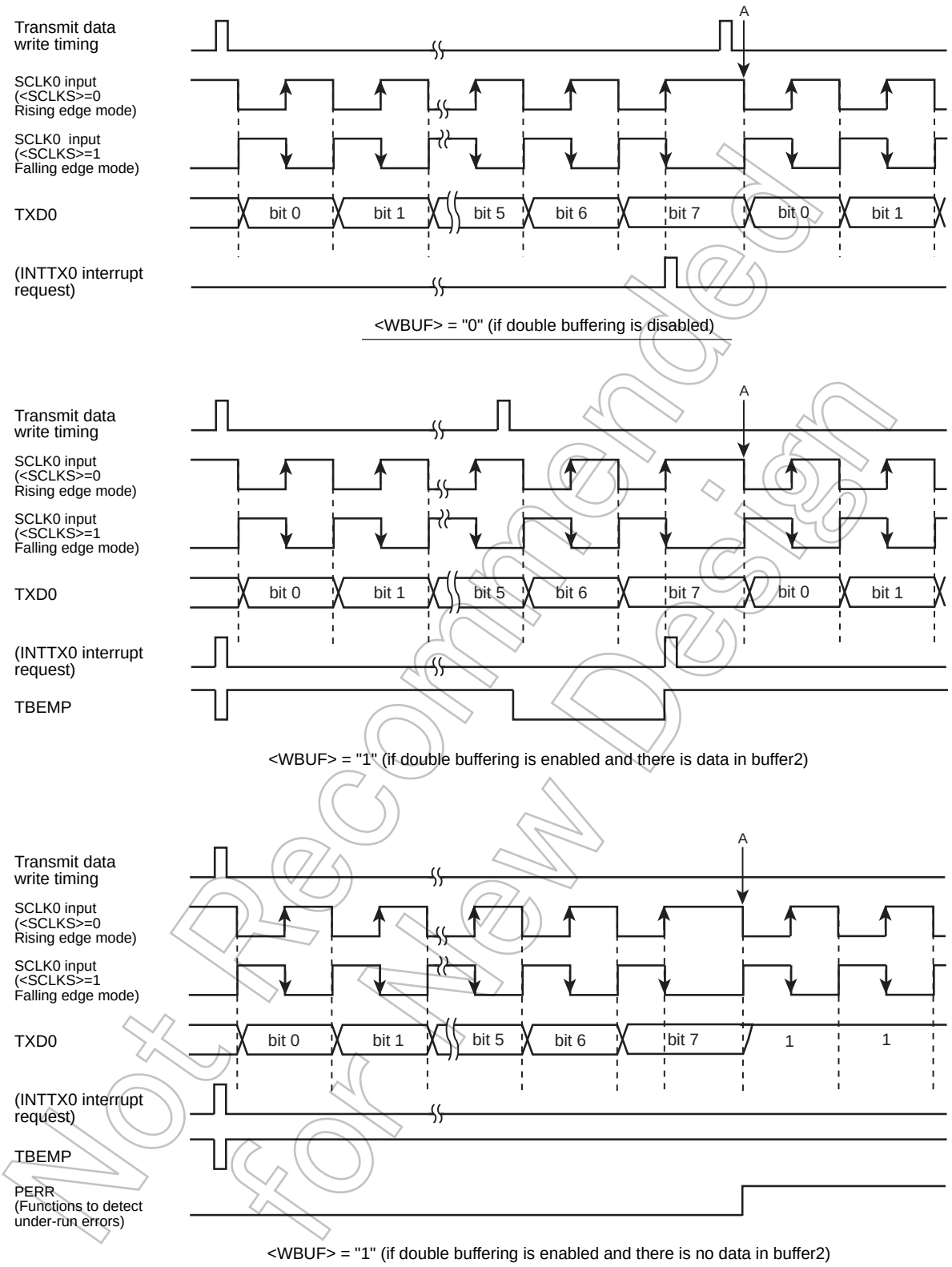


Figure 15-13 Transmit Operation in the I/O Interface Mode (SCLK Input Mode)

15.17.1.2 Receive

(1) SCLK Output Mode

The SCLK output can be started by setting the receive enable bit SCxMOD0<RXE> to "1".

- If double buffer is disabled (SCxMOD2<WBUF> = "0")

  A clock pulse is outputted from the SCLK pin and the next data is stored into the shift register each time the CPU reads received data. When all the 8 bits are received, the INTRXx interrupt is generated.

- If double buffer is enabled (SCxMOD2<WBUF> = "1")

  Data stored in the shift register is moved to the receive buffer and the receive buffer can receive the next frame. A data is moved from the shift register to the receive buffer, the receive buffer full flag SCxMOD2<RBFLL> is set to "1" and the INTRXx is generated.

  While data is in the receive buffer, if the data cannot be read from the receive buffer before completing reception of the next 8 bits, the INTRXx interrupt is not generated and the SCLK output stops. In this state, reading data from the receive buffer allows data in the shift register to move to the receive buffer and thus the INTRXx interrupt is generated and data reception resumes.

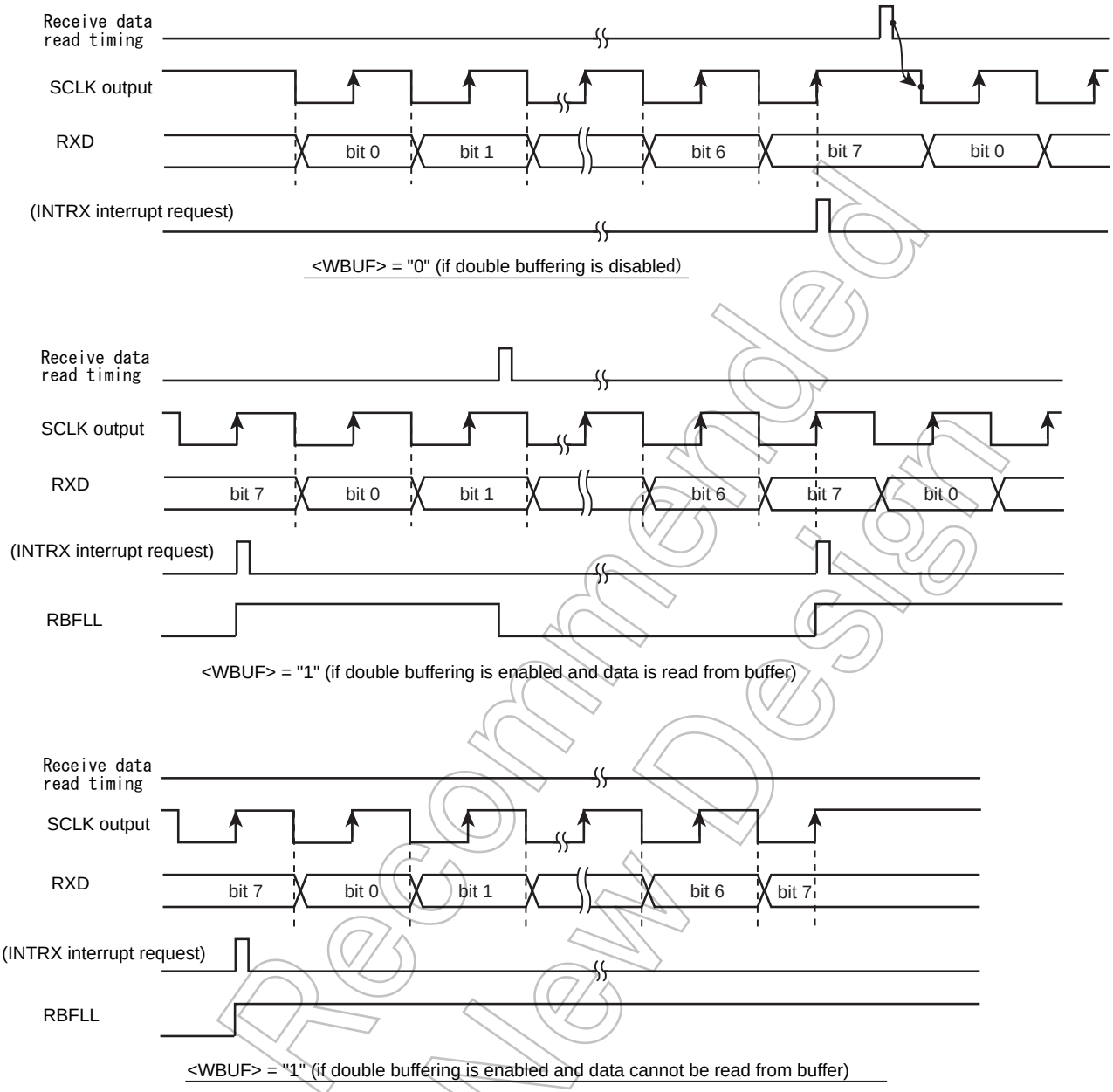


Figure 15-14 Receive Operation in the I/O Interface Mode (SCLK Output Mode)

(2) SCLK Input Mode

In the SCLK input mode, receiving double buffering is always enabled, the received frame can be moved to the receive buffer from the shift register, and the receive buffer can receive the next frame successively.

The INTRx receive interrupt is generated each time received data is moved to the receive buffer.

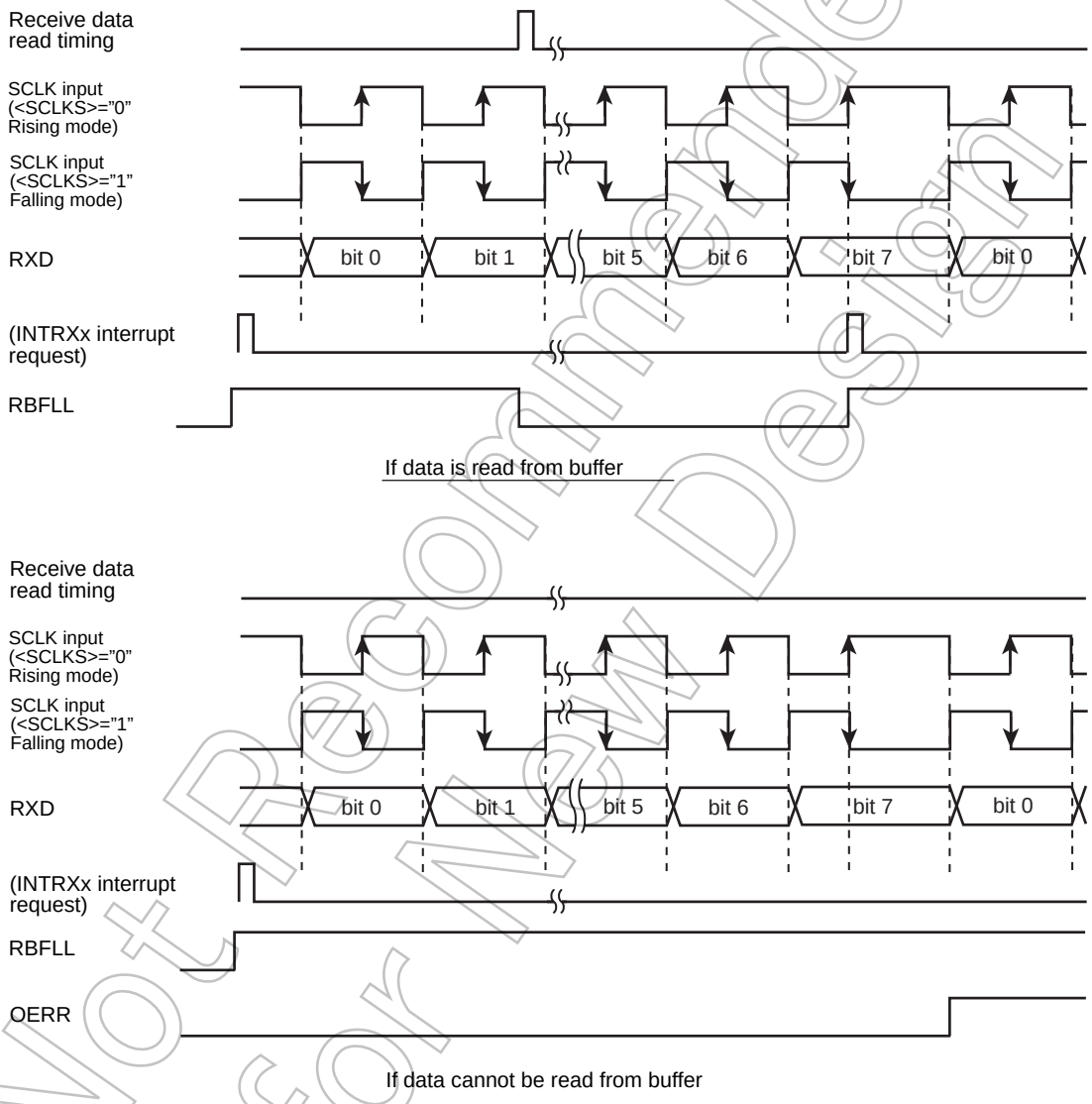


Figure 15-15 Receive Operation in the I/O Interface Mode (SCLK Input Mode)

15.17.1.3 Transmit and Receive (Full-duplex)

(1) SCLK Output Mode

- If SCxMOD2<WBUF> is set to "0" and the double buffers are disabled

  SCLK is outputted when the CPU writes data to the transmit buffer.

  Subsequently, 8 bits of data are shifted into receive buffer and the INTRXx receive interrupt is generated. Concurrently, 8 bits of data written to the transmit buffer are outputted from the TXD pin, the INTTXx transmit interrupt is generated when transmission of all data bits has been completed. Then, the SCLK output stops.

  The next round of data transmission and reception starts when the data is read from the receive buffer and the next transmit data is written to the transmit buffer by the CPU. The order of reading the receive buffer and writing to the transmit buffer can be freely determined. Data transmission is resumed only when both conditions are satisfied.

- If SCxMOD2<WBUF> is set to "1" and the double buffers are enabled

  SCLK is outputted when the CPU writes data to the transmit buffer.

  8 bits of data are shifted into the receive shift register, moved to the receive buffer, and the INTRXx interrupt is generated. While 8 bits of data is received, 8 bits of transmit data is outputted from the TXD pin. When all data bits are sent out, the INTTXx interrupt is generated and the next data is moved from the transmit buffer to the transmit shift register.

  If the transmit buffer has no data to be moved to the transmit buffer (SCxMOD2<TBEMP> = 1) or when the receive buffer is full (SCxMOD2<RBFULL> = 1), the SCLK output is stopped. When both conditions, receive data is read and transmit data is written, are satisfied, the SCLK output is resumed and the next round of data transmission and reception is started.

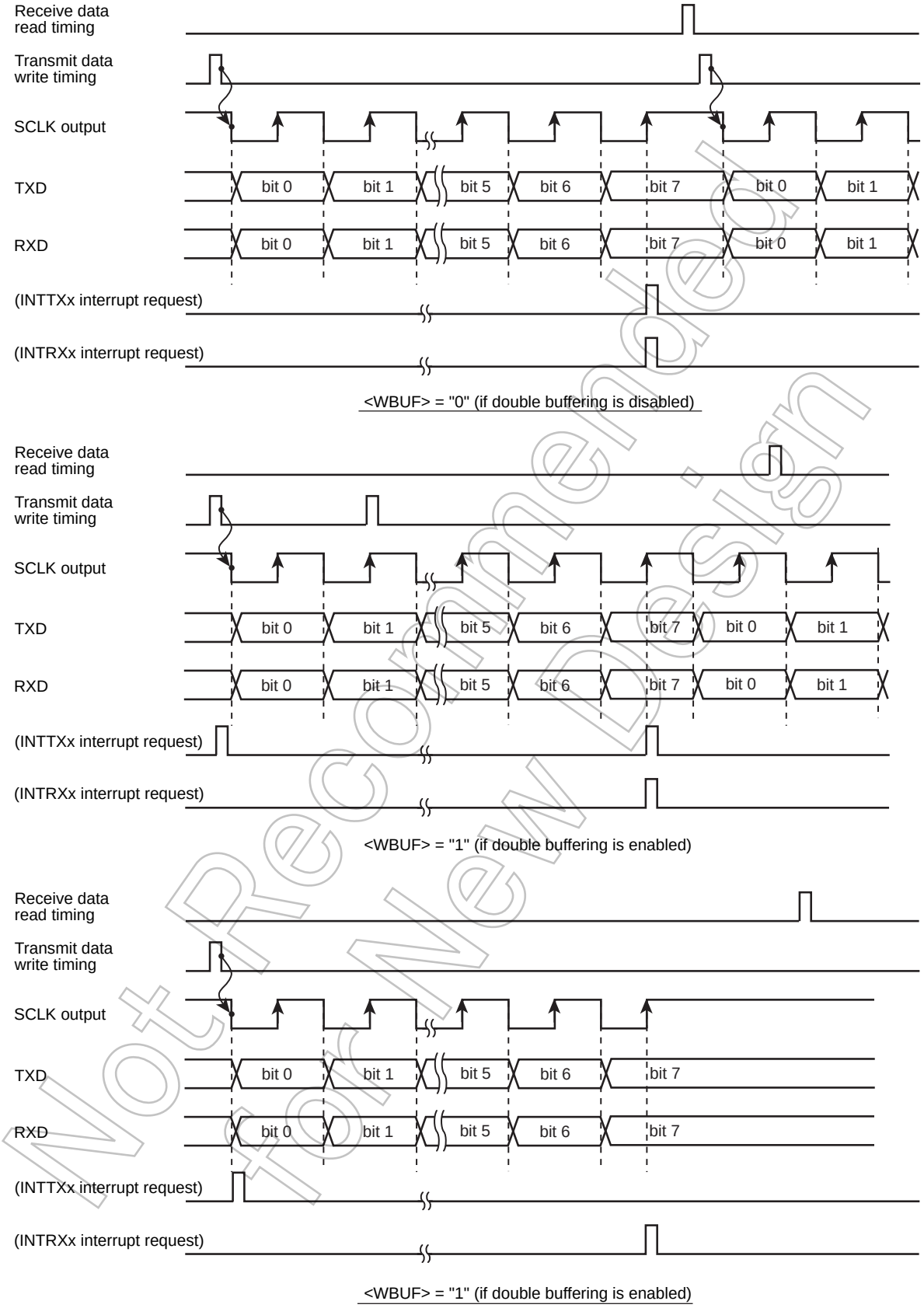


Figure 15-16 Transmit/Receive Operation in the I/O Interface Mode (SCLK Output Mode)

(2) SCLK Input Mode

- If SCxMOD2<WBUF> is set to "0" and the transmit double buffer is disabled

  When receiving data, double buffer is always enabled regardless of the SCxMOD2 <WBUF> settings.

  8-bit data written in the transmit buffer is outputted from the TXD pin and 8 bit of data is shifted into the receive buffer when the SCLK input becomes active.The INTTXx interrupt is generated upon completion of data transmission. The INTTRXx interrupt is generated when the data is moved from shift register to receive buffer after completion of data reception.

  Note that transmit data must be written into the transmit buffer before the SCLK input for the next frame (data must be written before the point A in Figure 15-17). Data must be read before completing reception of the next frame data.

- If SCxMOD2<WBUF> is set to "1" and the double buffer is enabled.

  The interrupt INTRXx is generated at the timing the transmit buffer data is moved to the transmit shift register after completing data transmission from the transmit shift register. At the same time, data received is shifted to the shift register, it is moved to the receive buffer, and the INTRXx interrupt is generated.

  Note that transmit data must be written into the transmit buffer before the SCLK input for the next frame (data must be written before the point A in Figure 15-17). Data must be read before completing reception of the next frame data.

  Upon the SCLK input for the next frame, transmission from transmit shift register (in which data has been moved from transmit buffer) is started while receive data is shifted into receive shift register simultaneously.

  If data in receive buffer has not been read when the last bit of the frame is received, an over-run error occurs. Similarly, if there is no data written to transmit buffer when SCLK for the next frame is input, an under-run error occurs.

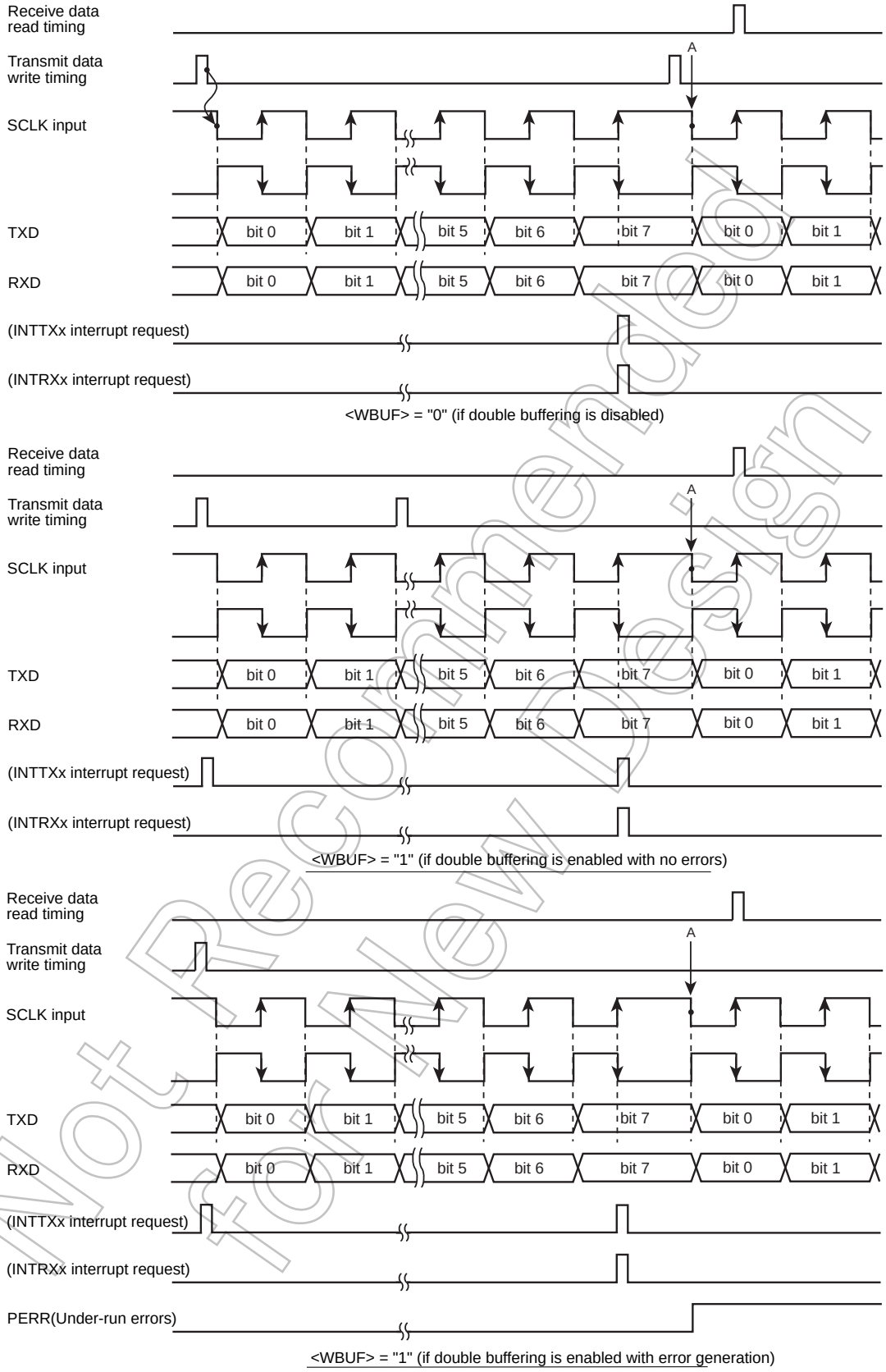


Figure 15-17 Transmit/Receive Operation in the I/O Interface Mode (SCLK Input Mode)

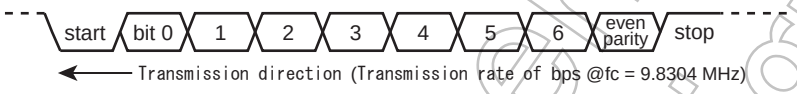
15.17.2 Mode 1 (7-bit UART mode)

The 7-bit UART mode can be selected by setting the serial mode control register (SCxMOD<SM[1:0]>) to "01".

In this mode, parity bits can be added to the transmit data stream; the serial mode control register (SCxCR<PE>) controls the parity enable/disable setting.

When <PE> is set to "1" (enable), either even or odd parity may be selected using the SCxCR<EVEN> bit. The length of the stop bit can be specified using SCxMOD2<SBLEN>.

The following table shows the control register settings for transmitting in the following data format.



Clocking conditions

system clock :	High-speed (fc)
High-speed clock gear:	X1 (fc)
Prescaler clock:	fperiph/2 (fperiph = fsys)

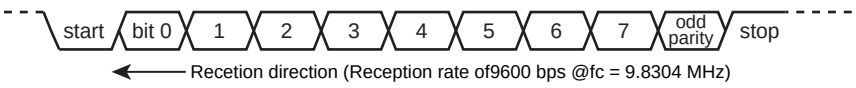
		7	6	5	4	3	2	1	0	
SCxMOD0	←	x	0	-	0	0	1	0	1	Set 7-bit UART mode
SCxCR	←	x	1	1	x	x	x	0	0	Even parity enabled
SCxBRCR	←	0	0	1	0	0	1	0	0	Set 2400bps
SCxBUF	←	*	*	*	*	*	*	*	*	Set transmit data

x : don't care - : no change

15.17.3 Mode 2 (8-bit UART mode)

The 8-bit UART mode can be selected by setting SCxMOD0<SM[1:0]> to "10." In this mode, parity bits can be added and parity enable/disable is controlled using SCxCR<PE>. If <PE> = "1" (enabled), either even or odd parity can be selected using SCxCR<EVEN>.

The control register settings for receiving data in the following format are as follows:



Clocking conditions

System clock:	High-speed (fc)
High-speed clock gear:	X1
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

15.17.4 Mode 3 (9-bit UART mode)

The 9-bit UART mode can be selected by setting SCxMOD0<SM[1:0]> to "11." In this mode, parity bits must be disabled (SCxCR<PE> = "0").

The most significant bit (9th bit) is written to bit 7 <TB8> of the serial mode control register 0 (SCxMOD0) for transmitting data. The data is stored in bit 7 <RB8> of the serial control register SCxCR.

When writing or reading data to/from the buffers, the most significant bit must be written or read first before writing or reading to/from SCxBUF.

The stop bit length can be specified using SCxMOD2<SBLEN>.

15.17.4.1 Wake up function

In the 9-bit UART mode, slave controllers can be operated in the wake-up mode by setting the wake-up function control bit SCxMOD0<WU> to "1."

In this case, the interrupt INTRXx will be generated only when SCxCR<RB8> is set to "1".

Note: The TXD pin of the slave controller must be set to the open drain output mode using the ODE register.

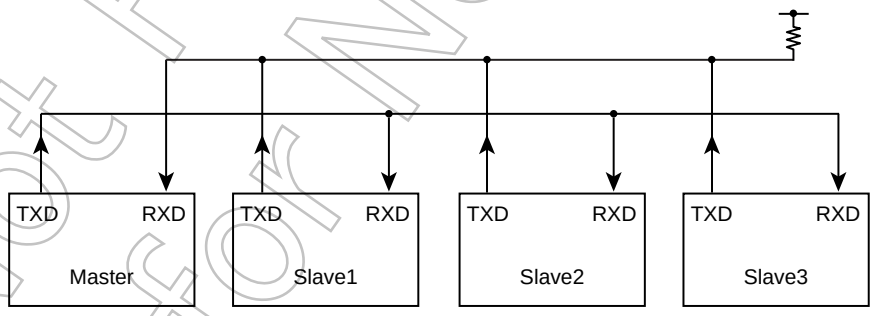


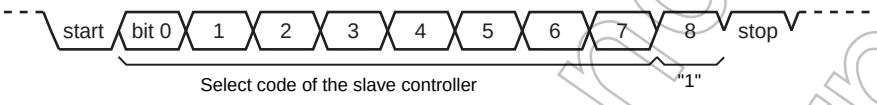
Figure 15-18 Serial Links to Use Wake-up Function

15.17.4.2 Protocol

1. Select the 9-bit UART mode for the master and slave controllers.

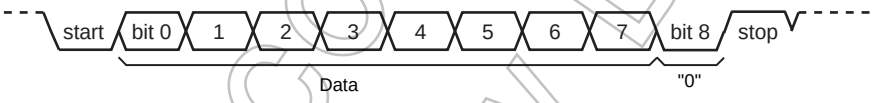
2. Set SCxMOD<WU> to "1" for the slave controllers to make them ready to receive data.

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 bit to "0".

5. The master controller transmits data to the designated slave controller (the controller of which SCxMOD<WU> bit is cleared to "0"). In this, the most significant bit (bit 8) <TB8> must be set to "0".



6. The slave controllers with the <WU> bit set to "1" ignore the receive data because the most significant bit (bit 8) <RB8> is set to "0" and thus no interrupt (INTRXx) is generated.Also, the slave controller with the <WU> bit set to "0" can transmit data to the master controller to inform that the data has been successfully received.

16. Asynchronous Serial Channel (UART)

16.1 Overview

This device has the Asynchronous serial channel (UART) with Modem control.

Their features are given in the following.

- Transmit FIFO

  8-bit width/ 32 location deep.
- Receive FIFO

  12-bit width/ 32 location deep.
- Transmit /Receive data format
  - DATA bits: 5,6,7,8 bits can be selected.
  - PARITY : use / no use
  - STOP bit : 1bit / 2 bits
- FIFO ON/OFF
  - ON (FIFO mode)/
  - OFF (characters mode)
- Interrupt
  - Combined interrupt factors are output to interrupt controller.
  - The permission of each interrupt factor is programmable.
- Baud rate generator

  Generates a common transmit and receive internal clock form UART internal reference clock input.

  Supports baud rates of up to 2.95Mbps at fsys = 48MHz.
- DMA
- IrDA 1.0 Function
  - Max data rate : 115.2 kbps (half-duplex).
  - support low power mode
- Control pins

  TXD (IROUT)

  RXD (IRIN)

  $\overline{\text{CTS}}$

  RIN

  $\overline{\text{RTS}}$

  DCD

  DSR

  DTR
- Hardware flow control

  RTS support

  CTS support

(1) UART transmit / receive data format.

Transmit / receive data format			
START	DATA (LSB → MSB)	PARITY	STOP

(2)Receive FIFO data format

	Receive data (LSB → MSB)								Framing error flag	Parity error flag	Break error flag	Overrun error flag
Bit Number	0	1	2	3	4	5	6	7				
Receive 8-bit data	1	1	1	1	1	1	1	1				
Receive 7-bi data	1	1	1	1	1	1	1	0				
Receive 6-bit data	1	1	1	1	1	1	0	0				
Receive 5-bit data	1	1	1	1	1	0	0	0				

16.2 Configuration

Figure 16-1 shows UART block diagram.

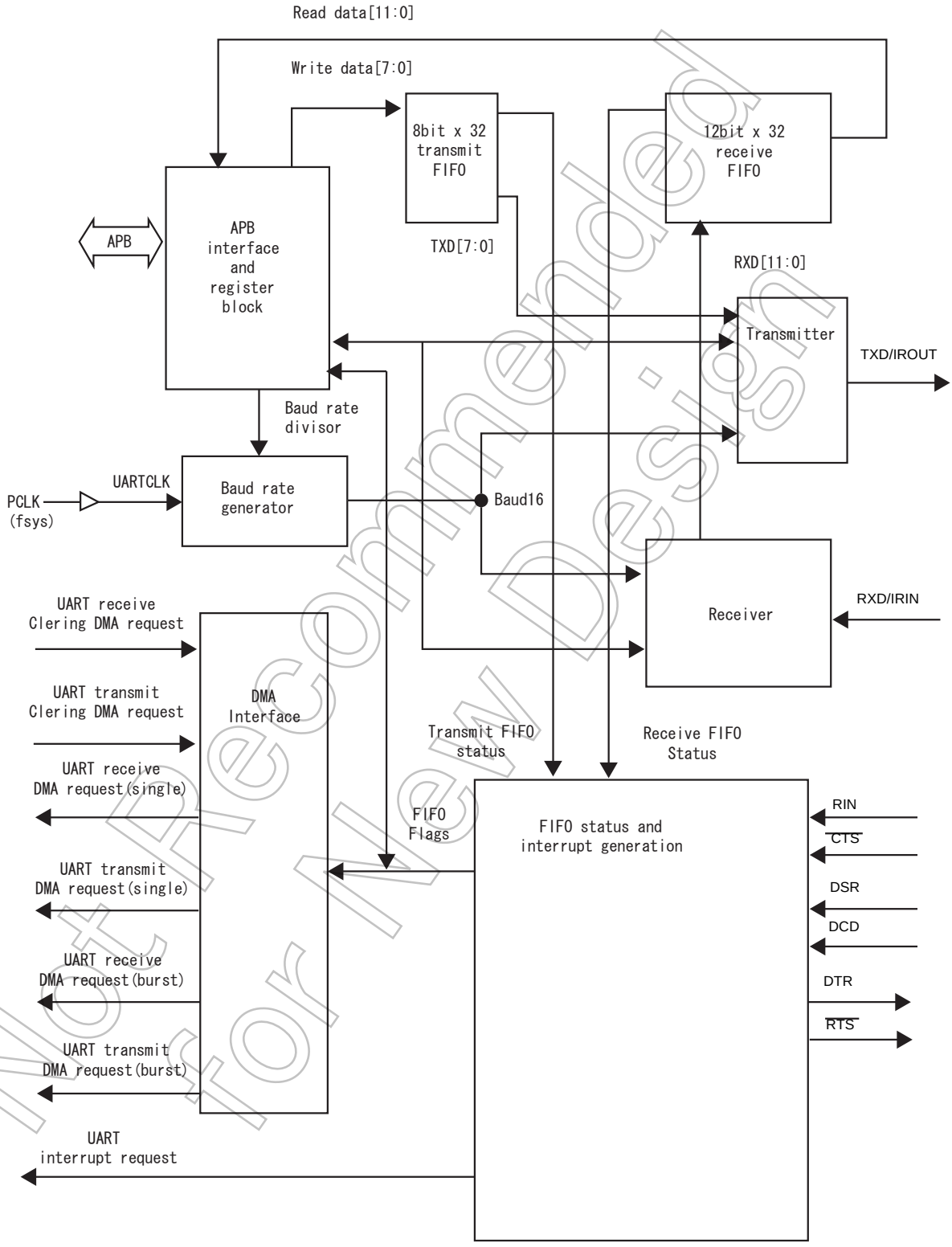


Figure 16-1 UART Channel Block Diagram

16.3 Registers Description

16.3.1 Registers List

The channel registers and addresses are shown below.

Base Address = 0x4004_8000

Register name		Address (Base+)
Data register	UARTDR	0x0000
Receive status register	UARTRSR	0x0004
Error clear register	UARTECR	0x0004
Reserved	-	0x0008 to 0x0017
Flag register	UARTFR	0x0018
Reserved	-	0x001C
IrDA low-power counter	UARTILPR	0x0020
Integer baud rate register	UARTIBDR	0x0024
Fractional baud rate register	UARTFBDR	0x0028
Line control register	UARTLCR_H	0x002C
Control register	UARTCR	0x0030
interrupt FIFO level select register	UARTIFLS	0x0034
Interrupt mask set/clear register	UARTIMSC	0x0038
Raw interrupt status register	UARTRIS	0x003C
Masked interrupt status register	UARTMIS	0x0040
Interrupt clear register	UARTICR	0x0044
DMA control register	UARTDMACR	0x0048
Reserved	-	0x004C to 0x0FFF

Note: **You must disable the UART before any of the control registers are reprogrammed. When the UART is disabled in the middle of transmit or receive operation, it stops after the transmission of the current character is completed.**

16.3.2 UARTDR (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 This bit is set to 1 if data is received and the receive FIFO is already full. In this case, the received data is not stored in the FIFO and is discarded. The bit is cleared to 0 once an empty space is made in the FIFO and a new data can be written to it.
10	BE	R	Break error This bit is set to 1 if a break condition was detected, indicating that the receive data input (defined as start, data parity, and stop bits) was held Low for a period longer than a full-word transmission time. In FIFO mode, this error is occurred by the highest character in FIFO. Break is occurred only when one zero character is loaded in FIFO. The next character is enabled when received data makes 1 (marking state) and the next effective start bit is received.
9	PE	R	Parity error When this bit is set to 1, it indicates that the parity of the received data does not match the parity defined by bits 2 and 7 of the UARTLCR_H register. In FIFO mode, this error is occurred by the highest character in FIFO.
8	FE	R	Framing error When this bit is set to 1, it indicates that the received data did not have a valid stop bit (a valid stop bit is 1). In FIFO mode, this error is occurred by the highest character in FIFO.
7-0	DATA[7:0]	R/W	Read : Receive data Write : Transmit data

16.3.3 UARTRSR (Receive status Register)

UARTRSR and UARTECR are mapped to same address.

These functions differ in read and write operations.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	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 This bit is set to 1 if data is received and the FIFO is already full. This bit is cleared 0 by writing a data to UARTECR. When FIFO is already full, the received data is not stored in the FIFO. Therefore the contents of FIFO are valid and only the contents of shift register is over written. In this case, CPU must be read out a data from FIFO to make empty FIFO.
2	BE	R	Break error This bit is set to 1 if a break condition was detected, indicating that the receive data input (defined as start, data bit, data parity, and stop bits) was held Low for longer than a full-word transmission time. This bit is cleared 0 by writing a data to UARTECR. In FIFO mode, this error is occurred by the highest character in FIFO. Break is occurred only when one zero character is loaded in FIFO. The next character is enabled when received data makes 1 (marking state) and the next effective start bit is received.
1	PE	R	Parity error When this bit is set to 1, it indicates that the parity of the received data does not match the parity defined by bits 2 and 7 of the UARTxLCR_H register. This bit is cleared 0 by writing a data to UARTECR. In FIFO mode, this error is occurred by the highest character in FIFO.
0	FE	R	Framing error When this bit is set to 1, it indicates that the received data did not have a valid stop bit (a valid stop bit is 1). This bit is cleared 0 by writing a data to UARTECR. In FIFO mode, this error is occurred by the highest character in FIFO.

16.3.4 UARTECR (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	A write to this register clears framing, parity, break, and overrun errors. The data value has no significance. The address of this register is the same as that of the UARTRSR register.
2	BE	W	
1	PE	W	
0	FE	W	

Note 1: The UARTRSR/UARTECR register is the receive status register/error clear register. Receive status can also be read from UARTRSR. If the status is read from this register, the status information for break, framing and parity corresponds to the data read from UARTDR prior to reading UARTRSR. The status information for overrun is set immediately when an overrun condition occurs. A write to UARTECR clears the framing, parity, break and overrun errors. All the bits are cleared to 0 on reset.

Note 2: The receive data must be read first from UARTDR before the error status associated with that data is read from UARTRSR. This read sequence cannot be reversed because the status register UARTRSR is updated only when the data is read from the data register UARTDR. The status information can also be read directly from the UARTDR register.

16.3.5 UARTFR (UART Flag register)

The <TXFE>, <RXFF>, <TXFF>, and <RXFE> bits differ depending on the state of the <FEN> of the UARTLCR_H 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 undefined.
8	RI	R	RIng indicator flag. 1: Modem status input = 0
7	TXFE	R	<FEN>=1 (Transmit FIFO empty flag) 0: Not empty 1: Empty
			<FEN>=0 (Transmit hold register empty flag) 0: Not empty 1: Empty
6	RXFF	R	<FEN>=1 (Receive FIFO full flag) 0: Not full 1: Full
			<FEN>=0 (Receive hold register full flag) 0: Not full 1: Full
5	TXFF	R	<FEN>=1 (Transmit FIFO full flag) 0: Not full 1: Full
			<FEN>=0 (Transmit hold register full flag) 0: Not full 1: Full
4	RXFE	R	<FEN>=1 (Receive FIFO empty flag) 0: Not empty 1: Empty
			<FEN>=0 (Receive hold register empty flag) 0: Not empty 1: Empty
3	BUSY	R	BUSY flag 0: The UART has stopped transmitting data 1: The UART is transmitting data. (BUSY)
2	DCD	R	Data carrier detect (DCD) flag 1: Modem status input = 0
1	DSR	R	Data set ready (DSR) flag 1: Modem status input = 0

Bit	Bit Symbol	Type	Function
0	CTS	R	Clear To Send (CTS) flag 1: Modem status input = 0

1. Transmit FIFO

   The transmit FIFO is an 8-bit wide, 32-location deep FIFO memory buffer. CPU data written across the APB interface is stored in the FIFO until it is read out by the transmit logic. The transmit FIFO can be disabled to act like a one-byte holding register.

2. Receive FIFO

   The receive FIFO is a 12-bit wide, 32-location deep, FIFO memory buffer. Received data and corresponding error bits are stored in the receive FIFO by the receive logic until they are read out by the CPU across the APB interface. The receive FIFO can be disabled to act like a one-byte holding register.

16.3.6 UARTILPR(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> = ($f_{UARTCLK}$ / $f_{IrLPBaud16}$). The UARTILPR 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 UARTCR<SIRLP> is set to 1.**

Note 2: **0x00 setting is prohibited.**

16.3.7 UARTIBDR (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 undefined.
15-0	BAUD DIVINT [15:0]	R/W	Integer part of buad rate divisor. (0x0002 to 0xFFFF) This register, when put together with the fractional baud rate divisor described next, provides the baud rate divisor BAUDDIV.

Note 1: **To update the contents of UARTxIBRD internally, the write to UARTxLCR_H must always be executed last. For details, refer to the description of UARTxLCR_H.**

Note 2: **Set this register before the UARTxCR<UARTEN> is set to 1.**

Note 3: **0x0000, 0x0001 cannot be set.**

Note 4: **The value of the worst case baud rate divisor of the set value due to the baud rate shift (total error) Ibetween 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

16.3.8 UARTFBDR(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	BAUDDIV FRAC [5:0]	R/W	Fractional part of baud rate divisor. 0x01 to 0x3F. The baud rate divisor is calculated as follows: Baud rate divisor BAUDDIV = ($f_{UARTCLK}$)/ (16 × baud rate) $f_{UARTCLK}$ is the frequency of UARTCLK. The BAUDDIV is comprised of the integer value (BAUD DIVINT) and the fractional value (BAUD DIVFRAC).

Note 1: To update the contents of UARTFBRD internally, the write to UARTLCR_H must always be executed last. For details, refer to the description of UARTLCR_H.

Note 2: Set this register before the UARTCR<UARTEN> is set to 1.

Note 3: The minimum baud rate divisor is 1 and the maximum baud rate divisor is 65535. Therefore, The integral part of baud rate divisor can not be set 0. And the fractional part of baud rate divisor must be set to 0 when the integral part of baud rate divisor is 65535.

Example: Calculating the divisor value

When the required baud rate is 230400 and $f_{UARTCLK}$ = 4 MHz:

Baud rate divisor = $(4 \times 10^6)/ (16 \times 230400) = 1.085$

Therefore, BRDI = 1 and BRDF = 0.085

Fractional part is $((0.085 \times 64) + 0.5) = 5.94$.

The integer part of this, m=0x5, should be set as the fractional baud rate divisor value.

Generated baud rate divisor = 1 + 5/64 = 1.078

Generated baud rate = $(4 \times 10^6)/ (16 \times 1.078) = 231911$

Error = (231911 - 230400)/ 230400 × 100 = 0.656%

The maximum error using a 6-bit UARTFBRD register = 1/64 × 100 = 1.56%

This error occurs when m = 1, and it is cumulative over 64 clock ticks.

16.3.9 UARTLCR_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	R/W	Stick parity select : refer to Table 16-1 for the truth table. When <SPS>, <EPS> and <PEN> of the UARTLCR_H register are set, the parity bit is transmitted and checked as a 0. When <SPS> and <PEN> are set and <EPS> is 0, the parity bit is transmitted and checked as a 1. When this bit is cleared, the stick parity is disabled. Refer toTable 16-1 for the truth table of <SPS>, <EPS>, and <PEN> bits.
6-5	WLEN[1:0]	R/W	Word length : 00: 5bits 01: 6bits 10: 7bits 11: 8bits This bit indicates the number of data bits transmitted or received in a frame.
4	FEN	R/W	FIFO control : 0: character mode 1: FIFO mode When this bit is set to 1, transmit and receive FIFO buffers are enabled (FIFO mode). When this bit is cleared to 0, the FIFOs are disabled (character mode) and they become 1-byte deep holding registers.
3	STP2	R/W	Stop bit select : 0: 1bit 1: 2 bits When this bit is set to 1, two stop bits are transmitted at the end of a frame. The receive logic does not check for the second stop bit being received.
2	EPS	R/W	Even parity select: 0: Odd 1: Even When this bit is set to 1, even parity generation and checking are performed during transmission and reception. This function checks whether the number of 1s contained in the data bits and parity bit is even. When this bit is cleared to 0, odd parity check is performed to check whether the number of 1s is odd. This bit has no effect when parity is disabled by Parity Enable bit <PEN> being cleared to 0. Refer to Table 16-1 for the truth table.
1	PEN	R/W	Parity control: 0:Disable 1: Enable When this bit is set to 1, parity check and generation are enabled. Otherwise, parity is disabled and no parity bit is added to data frames. Refer to Table 16-1 for the truth table of <SPS>, <EPS>, and <PEN> bits.
0	BRK	R/W	Send break : 0: No effect 1: Send break When this bit is set to 1, the TXD output remains LOW after the current character is transmitted. For generation of the transmit break condition, this bit must be asserted while at least one frame is or longer being transmitted. Even when the break condition is generated, the contents of the transmit FIFO are not affected.

Note: **When you set UARTLCR_H, UARTIBRD and UARTFBRD, UARTLCR_H must be set at the end. When you update only UARTIBRD or UARTFBRD, UARTLCR_H register must be set again.**

Table 16-1 is the truth table of the <SPS>, <EPS> and <PEN> bits of the UARTLCR_H register.

Table 16-1 Truth table of UARTLCR_H <SPS>, <EPS> and <PEN>

Parity enable <PEN>	Even parity select <EPS>	Stick parity select <SPS>	Parity bit(transmitted or checked)
0	×	×	No transmitted or checked
1	1	0	Even parity
1	0	0	Odd parity
1	0	1	1
1	1	1	0

16.3.10 UARTCR (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 undefined.
15	CTSEBN	R/W	CTS hardware flow control enable : 0:Disable 1: Enable When this bit is set to 1, CTS hardware flow control is enabled. Data is transmitted only after the $\overline{CTS}$ signal has been asserted.
14	RTSEN	R/W	RTS hardware flow control enable : 0: Disable 1: Enable When this bit is set to 1, RTS hardware flow control is enabled. Data is transmitted only when there is an empty space in the receive FIFO.
13-12	-	R	Read as undefined.
11	RTS	R/W	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	R/W	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	R/W	UART receive enable : 0: Disable 1: Enable When this bit is set to 1, the receive circuit of the UART is enabled. Data reception occurs for either UART function or SIR function according to the setting of <SIREN>. When the UART is disabled in the middle of receive operation, it completes current reception and the subsequent receptions are disabled.
8	TXE	R/W	UART transmit enable : 0: Disable 1: Enable When this bit is set to 1, the transmit circuit of the UART is enabled. Data transmission occurs for either UART function or SIR function according to the setting of <SIREN>. When the UART is disabled in the middle of transmit operation, it completes the current transmission before stopping.
7	Reserved	R/W	Write as zero.
6-3	Reserved	-	Read as undefined.

Bit	Bit Symbol	Type	Function
2	SIRLP	R/W	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 (IROUT) 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	R/W	SIR enable : 0 : Disable 1 : Enable 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 IROUT and IRIN pins are enabled. The TXD pin remains in the marking state (set to 1). Signal transitions on the RXD pin or modem status input have no effect. When IrDA circuit is disabled, IROUT remains cleared to 0 (no light pulse is generated) and the IRIN pin has no effect.
0	UARTEN	R/W	UART enable : 0 : Disable 1 : Enable When this bit is set to 1, the UART is enabled. Data transmission and reception occur for either UART function or SIR function according to the setting of <SIREN>. When the UART is disabled in the middle of transmit or receive operation, it completes current transmission or reception before stopping.

16.3.11 UARTIFLS (UART interrupt FIFO level select register)

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

Bit	Bit Symbol	Type	Function
31-6	−	R	Read as undefined .
5-3	RXIFLSEL[2:0]	R/W	Receive interrupt FIFO level select (1 word = 12 bits): The trigger points for the receive interrupt are as follows 000: Receive FIFO becomes ≥ 1/8 full 001: Receive FIFO becomes ≥ 1/4 full 010: Receive FIFO becomes ≥ 1/2 full 011: Receive FIFO becomes ≥ 3/4 full 100: Receive FIFO becomes ≥ 7/8 full 101to 111: Reserved
2-0	TXIFSEL[2:0]	R/W	Transmit FIFO level select (1 word = 8 bits) : The trigger points for the transmit interrupt are as follows: 000: Transmit FIFO becomes ≤ 1/8 full 001: Transmit FIFO becomes ≤ 1/4 full 010: Transmit FIFO becomes ≤ 1/2 full 011: Transmit FIFO becomes ≤ 3/4 full 100: Transmit FIFO becomes ≤ 7/8 full 101 to 111: Reserved

The UARTIFLS register is the interrupt FIFO level select register. This register is used to define the FIFO level at which UARTTXINTR and UARTRXINTR are generated.

The interrupts are generated based on a transition through a level rather than based on the level. For example, an interrupt is generated at a point when the third word has been stored in the receive FIFO which contained two words.

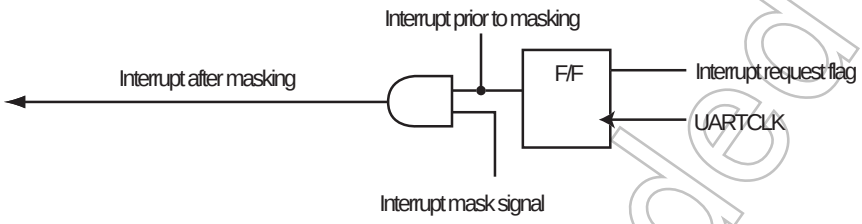
16.3.12 UARTIMSC (UART Interrupt mask set/clear Register)

	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	-	-	-	-	-	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 undefined.
10	OEIM	R/W	Overrun error interrupt mask : 0: Clear the mask 1: Set the mask
9	BEIM	R/W	Break error interrupt mask : 0: Clear the mask 1: Set the mask
8	PEIM	R/W	Parity error interrupt mask : 0: Clear the mask 1: Set the mask
7	FEIM	R/W	Framing error interrupt mask : 0: Clear the mask 1: Set the mask
6	RTIM	R/W	Receive time out interrupt mask : 0: Clear the mask 1: Set the mask
5	TXIM	R/W	Transmit FIFO interrupt mask : 0: Clear the mask 1: Set the mask
4	RXIM	R/W	Receive FIFO interrupt mask : 0: Clear the mask 1: Set the mask
3	DSRMIM	R/W	DSR modem interrupt mask : 0: Clear the mask 1: Set the mask
2	DCDMIM	R/W	DCD modem interrupt mask : 0: Clear the mask 1: Set the mask
1	CTSMIM	R/W	CTS modem interrupt mask : 0: Clear the mask 1: Set the mask
0	RIMIM	R/W	RIN modem interrupt mask : 0: Clear the mask 1: Set the mask.

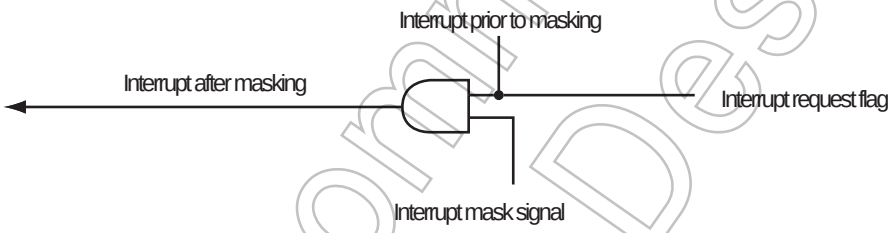
・UART interrupt generation block diagrams.

1. Block diagram of the break error <BE>, parity error <PE> and framing error <PE> flags



・The interrupt request flag state changes in real time and is retained in the F/F. Each flag can be cleared by a write to the corresponding bit in the interrupt clear register.

2. Block diagram of the overrun error <OE> flag.



・The interrupt request flag state by the overrun error <OE> flag changes in real time and its state is not retained . And the <OE> flag is cleared by a read of receive FIFO.

16.3.13 UARTRIS (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 undefined.
10	OERIS	R	Overrun error raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
9	BERIS	R	Break error raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
8	PERIS	R	Parity error raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
7	FERIS	R	Framing error raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
6	RTRIS	R	Receive time out raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
5	TXRIS	R	Transmit raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
4	RXRIS	R	Receive raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
3	DSRRMIS	R	DSR modem raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
2	DCDRMIS	R	DCD modem raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
1	CTSRMIS	R	CTS modem raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.
0	RIRMIS	R	RIN modem raw interrupt status : 0: Interrupt not requested 1: Interrupt requested.

Note: **All the bits, except the modem raw status interrupt bits (bits 3 to 0), are cleared to 0 when reset. The modem status bit are undefined after reset.**

16.3.14 UARTMIS (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 undefined.
10	OEMIS	R	Overrun error masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
9	BEMIS	R	Break error masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
8	PEMIS	R	Parity error masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
7	FEMIS	R	Framing error masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
6	RTMIS	R	Receive time out masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
5	TXMIS	R	Transmit masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
4	RXMIS	R	Receive masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
3	DSRMMIS	R	DSR modem masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
2	DCDMMIS	R	DCD modem masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
1	CTSMMIS	R	CTS modem masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.
0	RIMMIS	R	RIN modem masked interrupt status: 0: Interrupt not requested. 1: Interrupt requested.

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.**

16.3.15 UARTICR (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	−	R	Write 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	Received time out 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: The UARTICR register is a write-only interrupt clear register. When a bit of this register is set to 1, the associated interrupt is cleared. A write of 0 to any bit of this register is invalid.

16.3.16 UARTDMACR (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 undefined.
2	DMAONERR	R/W	DMA on error : 0: Not available 1: Available When this bit is set to 1, the DMA receive request output, UARTRXDMASREQ or UARTRXDMABREQ, is disabled on assertion of a UART error interrupt.
1	TXDMAE	R/W	Transmit FIFO DMA enable : 0: Disable 1: Enable
0	RXDMAE	R/W	Receive FIFO DMA enable : 0: Disable 1: Enable

Note 1: For example, if 19 characters have to be received and the watermark level is programmed to be four, then the DMA controller transfers four bursts of four characters and three single transfers to complete the stream.

Note 2: The bus width must be set to 8-bits, if you transfer the data of transmit/ receive FIFO by using DMAC.

16.4 Operation Description

16.4.1 Baud rate generator

The baud rate generator contains the internal Baud16 clock circuit which controls the timing of UART transmit and receive, and the internal IrLPBaud16 circuit which generates the pulse width of the IrDA encoded transmit bit stream when in low-power mode.

16.4.2 Transmit FIFO

The transmit FIFO is an 8-bit wide, 32-location deep, FIFO memory buffer. CPU data written across the APB interface is stored in the FIFO until it is read out by the transmit logic. You can disable the transmit FIFO to act like a one-byte holding register.

16.4.3 Receive FIFO

The receive FIFO is a 12-bit wide, 32 locations deep, FIFO memory buffer. Received data and corresponding error bits are stored in the receive FIFO by the receive logic until they are read out by the CPU across the APB interface. The receive FIFO can be disabled to act like a one-byte holding register.

16.4.4 Transmit logic

The transmit logic performs parallel-to-serial conversion on the data read from the transmit FIFO. Control logic outputs the serial bit stream beginning with a start bit, data bits with the Least Significant Bit (LSB) first, followed by the parity bit, and then the stop bits according to the programmed configuration in control registers.

16.4.5 Receive logic

The receive logic performs serial-to-parallel conversion on the received bit stream after a start bit has been detected. Error check for overrun, parity and frame and line break detection are also performed. Their error bit data is written to the receive FIFO.

16.4.6 Interrupt generation logic

UART outputs a maskable combined interrupt for every interrupt sources.

16.4.7 Interrupt timing

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

Note: STOP bit in above table means the last STOP bit. (The number of STOP bit can be selected among 1 or 2.)

16.4.8 UART interrupt block

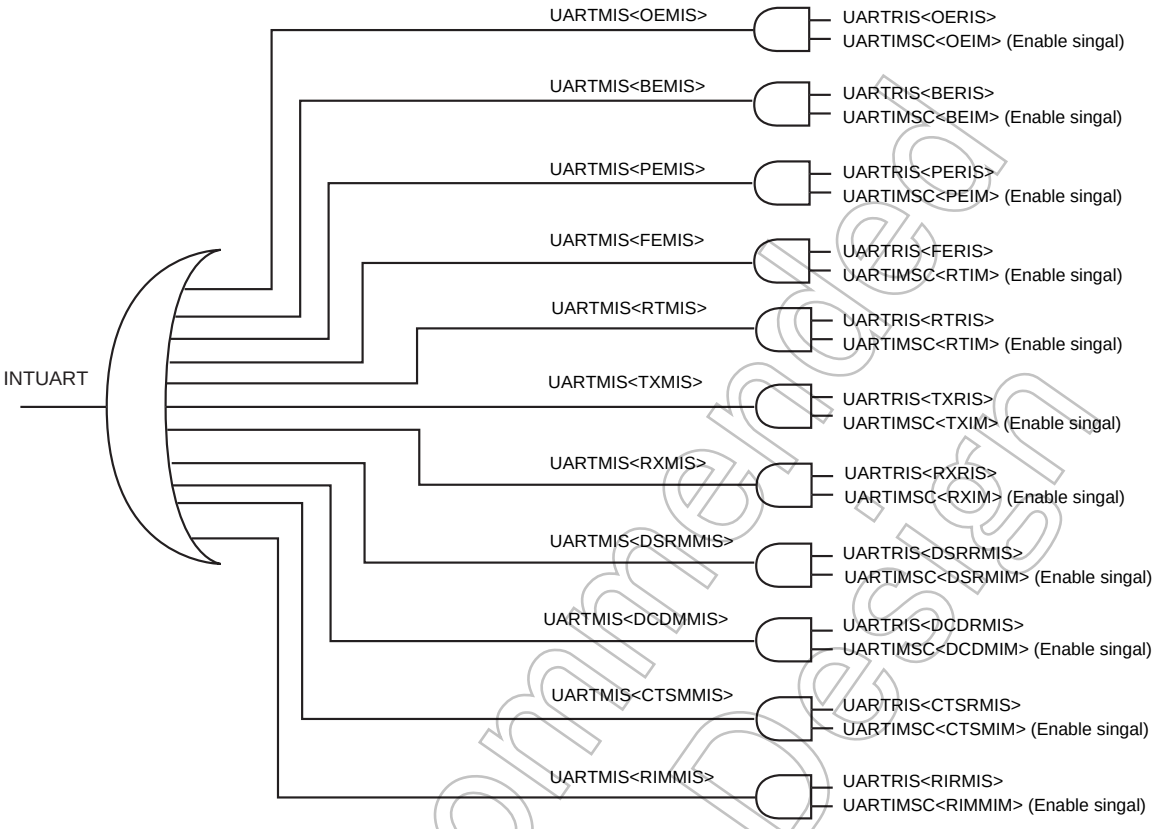


Figure 16-2 UART interrupt block

16.4.9 DMA interface

The UART support DMA controller.

16.4.10 IrDA circuit description

The IrDA is comprised of:

- 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 16-4**

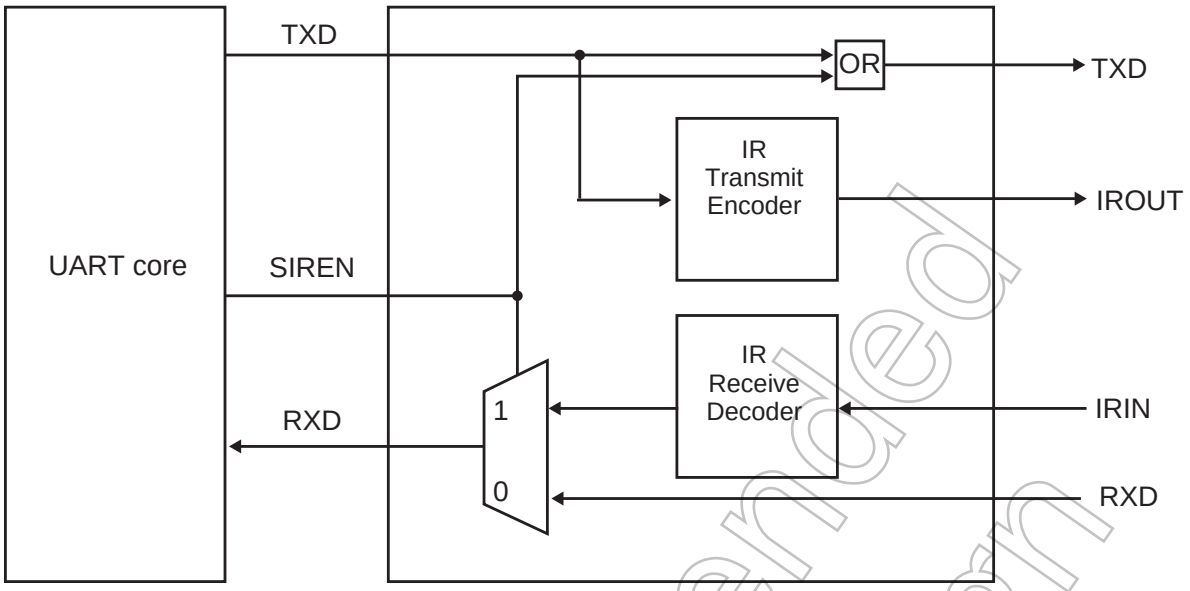


Figure 16-3 IrDA Circuit Block Diagram

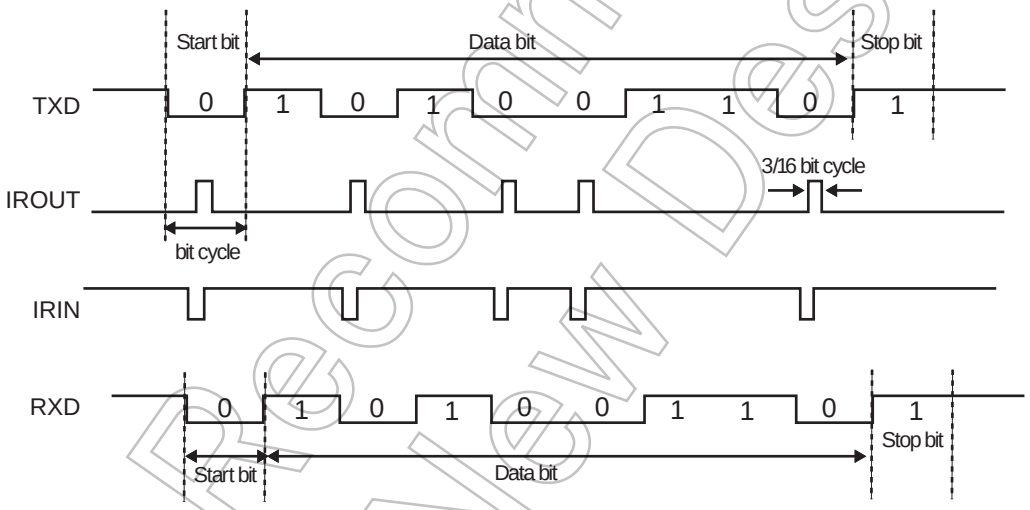


Figure 16-4 IrDA Data Modulation

16.4.11 Hardware flow control

The hardware flow control feature is fully selectable, and enables you to control the serial data flow by using the $\overline{\text{RTSx}}$ output and $\overline{\text{CTSx}}$ input signals.

Figure 16-5 shows how the two devices can communicate with each other using hardware flow control.

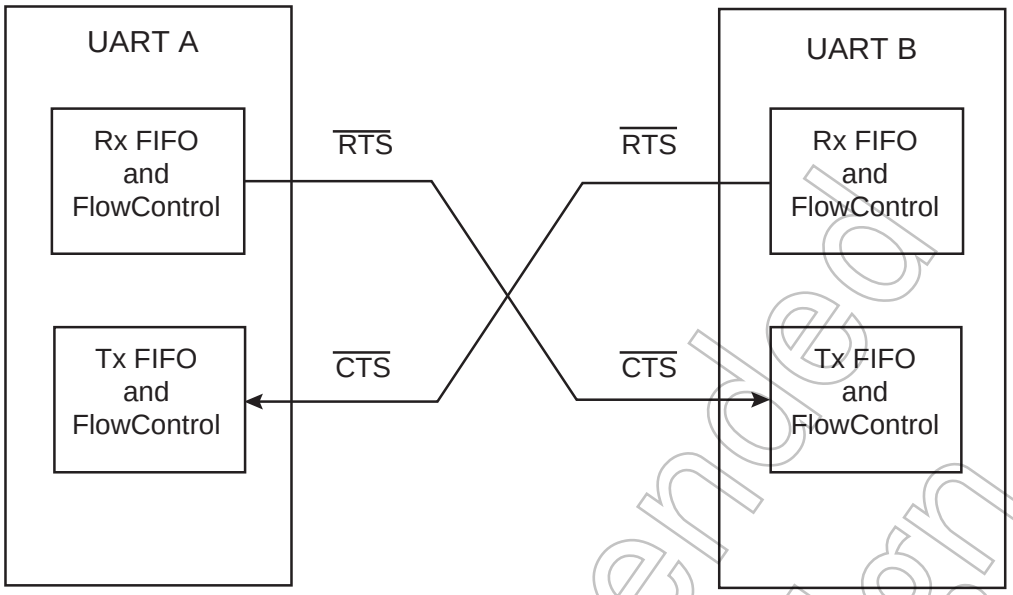


Figure 16-5 Hardware Flow Control

1. RTS flow control

   The RTS flow control logic is linked to the programmable receive FIFO watermark levels. When RTS flow control is enabled, the $\overline{RTS}$ is asserted until the receive FIFO is filled up to the watermark level.

   When the amount of data stored in the receive FIFO exceeds watermark level, the $\overline{RTS}$ signal is deasserted, indicating that there is no more room to receive.

   The $\overline{RTS}$ signal is reasserted when data has been read out of the receive FIFO and it is filled to less than the watermark level.

   Even if RTS flow control is disabled, communication can be enabled.

2. CTS flow control

   If CTS flow control is enabled, then the transmitter checks the $\overline{CTS}$ signal before transmitting. If the $\overline{CTS}$ signal is asserted, it transmits the byte, otherwise transmission does not occur.

   The data transmission continues while $\overline{CTS}$ is asserted and the transmit FIFO is not empty. If the transmit FIFO is empty, no data is transmitted even when the $\overline{CTS}$ signal is asserted.

   If the $\overline{CTS}$ signal is deasserted while CTS flow control is enabled, the current data transmission is completed before stopping.

   Even if CTS flow control is disabled, communication can be enabled.

UARTCR		RTSx	Description
<CTSEN>	<RTSEN>		
1	1	0 (note)	Both RTS and CTS flow control enabled.
1	0	1	Only CTS flow control enabled.
0	1	0 (note)	Only RTS flow control enabled.
0	0	1	Both RTS and CTS flow control disabled.

Note: During in the <RTSEN>=1(Enable), the $\overline{RTS}$ is set to 0(Enable) until the receive FIFO is filled up to the watermark level.

17. Synchronous Serial Port (SSP)

17.1 Overview

This LSI contains the SSP (Synchronous Serial Port) with 3 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	In master mode	fsys/2 to fsys/65024
	In slave mode (Note)	fsys/12 to fsys/65024
DMA		Supported
Internal test function		The internal loopback test mode is available.
Control pin (x = 0 to 2)		SPxCLK,SPxFSS,SPxDO,SPxDI

17.2 Block Diagram

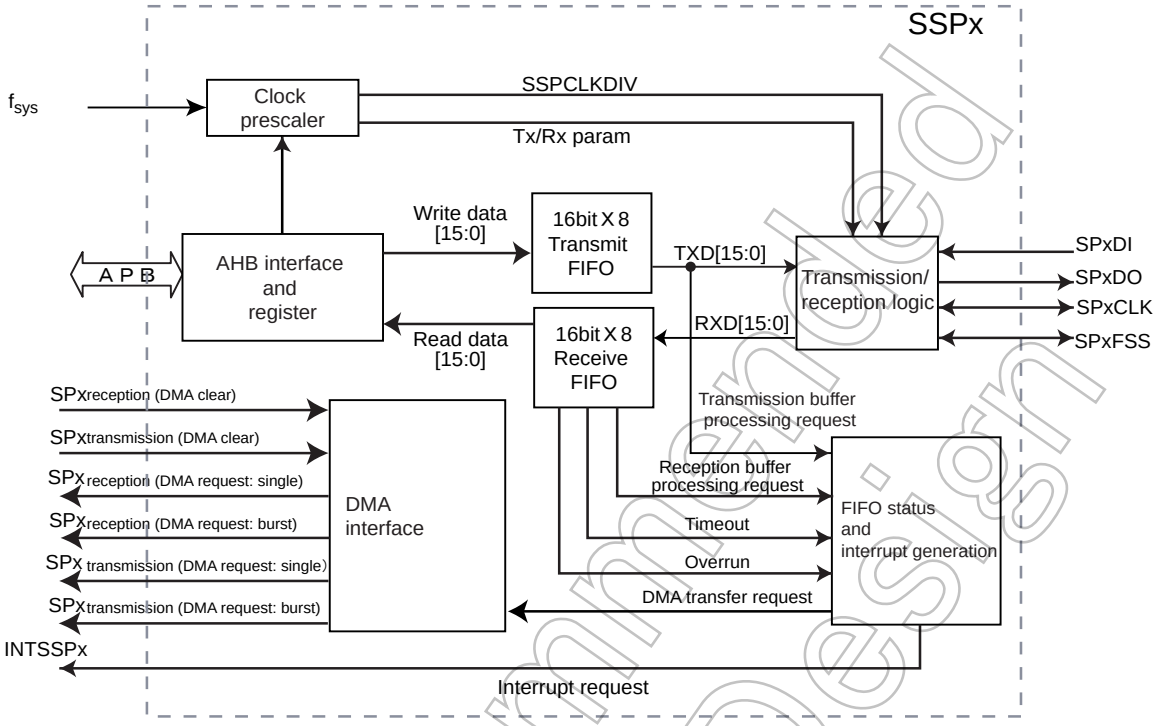


Figure 17-1 SSP block diagram

17.3 Register

17.3.1 Register List

Channel x	Base Address
Channel0	0x4004_0000
Channel1	0x4004_1000
Channel2	0x4004_2000

Register Name(x=0~2)		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

Note 1: These registers in the above table allows to access only word (32 bits) basis.
Note 2: Access to the "Reserved" area is prohibited.

17.3.2 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	Description
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.

17.3.3 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	Description
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

17.3.4 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	Description
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.

17.3.5 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	Description
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.

17.3.6 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	Description
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.

17.3.7 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	Description
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 conditioal interrupt to indicate that data was written when the receive FIFO was in the full condition.

17.3.8 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	Description
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

17.3.9 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	Description
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

17.3.10 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	Description
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

17.3.11 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	Description
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

17.4 Overview of SSP

This LSI contains the SSP with 3channels.

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.

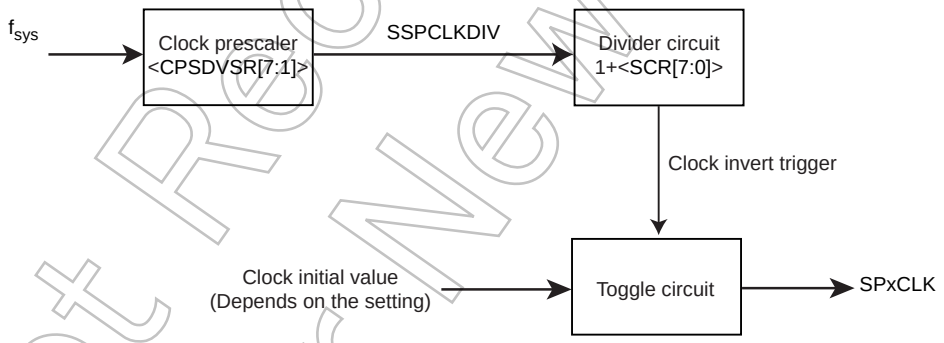
17.4.1 Clock prescaler

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>))



17.4.2 Transmit FIFO

This is a 16-bit wide, 8-layered transmit FIFO buffer, which is shared in master and slave modes.

17.4.3 Receive FIFO

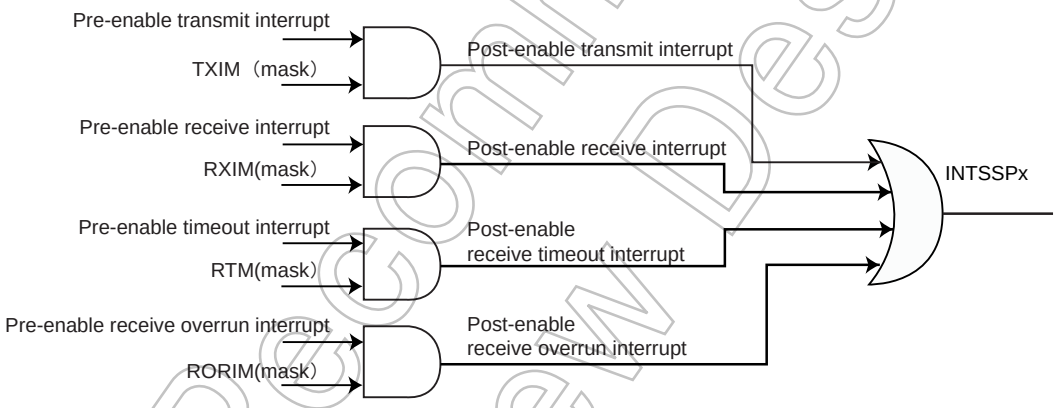
This is a 16-bit wide 8-layered receive FIFO buffer, which is shared in master and slave modes.

17.4.4 Interrupt generation logic

The High active 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").

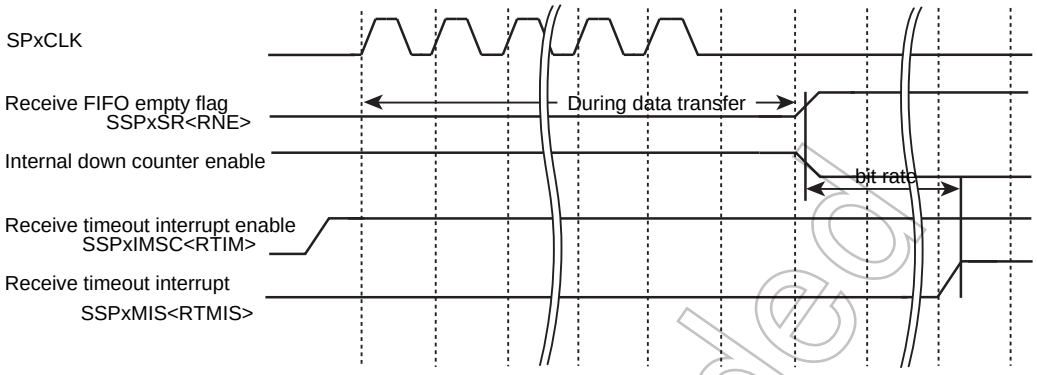
The first transmitted data can be written in the FIFO by using this interrupt.

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.

17.4.5 DMA interface

The DMA operation of the SSP is controlled through SSPxDMACR register.

When there are more data than the watermark level (half of the FIFO) in the receive FIFO, the receive DMA request is asserted.

When the amount of data left in the transmit FIFO is less than the watermark level (half of the FIFO), the transmit DMA request is asserted.

To clear the transmit/receive DMA request, an input pin for the transmit/receive DMA request clear signals, which are asserted by the DMA controller, is provided.

Set the DMA burst length to four words.

Note: For the remaining three words, the SSP does not assert the burst request.

Each request signal remains asserted until the relevant DMA clear signal is asserted. After the request clear signal is deasserted, a request signal can become active again, depending on the conditions described above. All request signals are deasserted if the SSP is disabled or the DMA enable signal is cleared.

The following table shows the trigger points for DMABREQ, for both the transmit and receive FIFOs.

	Burst length	
Watermark level	Transmit (number of empty locations)	Receive (number of filled locations)
1/2	4	4

17.5 SSP operation

17.5.1 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

17.5.2 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.

17.5.3 Clock ratios

When setting a frequency for fsys , the following conditions must be met.

- In master mode

  f_{SPxCLK} (maximum) → $f_{sys}/2$

  f_{SPxCLK} (minimum) → $f_{sys}/(254\times256)$
- In slave mode

  f_{SPxCLK} (maximum) → $f_{sys}/12$

  f_{SPxCLK} (minimum) → $f_{sys}/(254\times256)$

17.6 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 "17.6.1" to "17.6.3" for details of each frame format.

17.6.1 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.

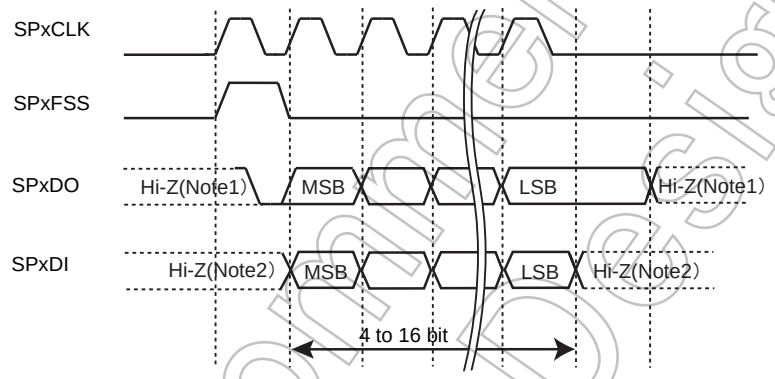


Figure 17-2 SSI frame format (transmission/reception during single transfer)

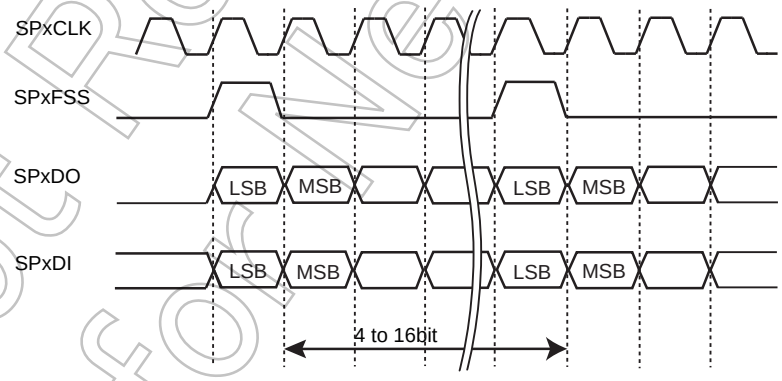


Figure 17-3 SSI frame format (transmission/reception during continuous transfer)

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.

17.6.2 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.

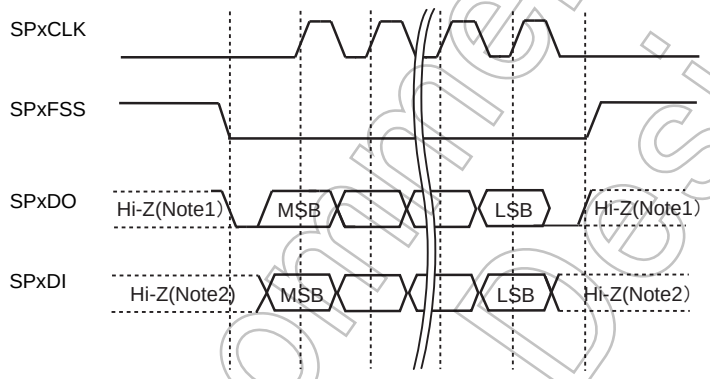


Figure 17-4 SPI frame format (single transfer, <SPO>="0" & <SPH>="0")

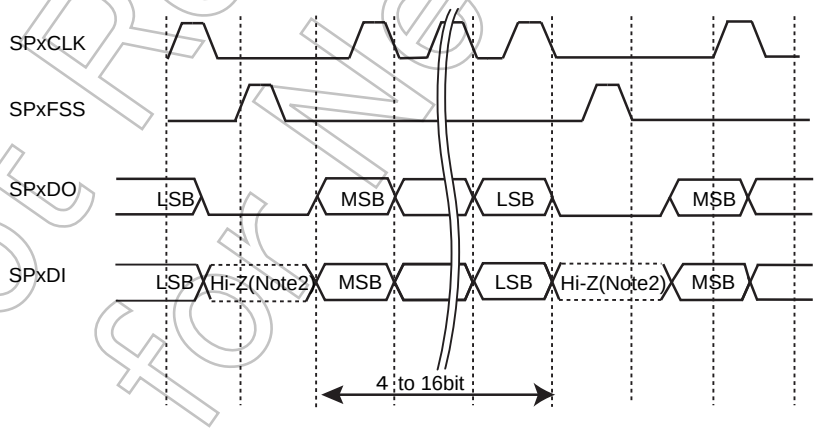


Figure 17-5 SPI frame format (continuous transfer,<SPO>="0" & <SPH>="0")

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.

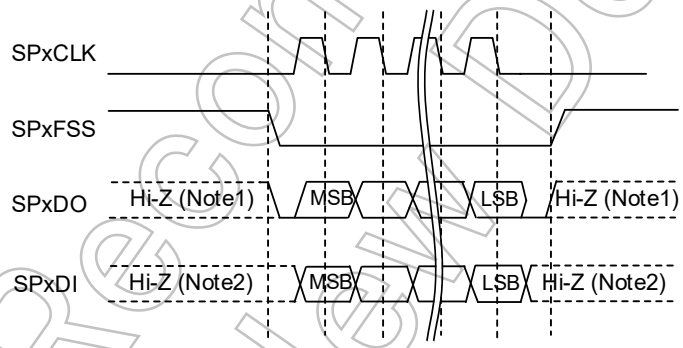


Figure 17-6 SPI frame format (Single & continuous transfer,<SPO>="0" & <SPH>="1")

Figure 17-6 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.

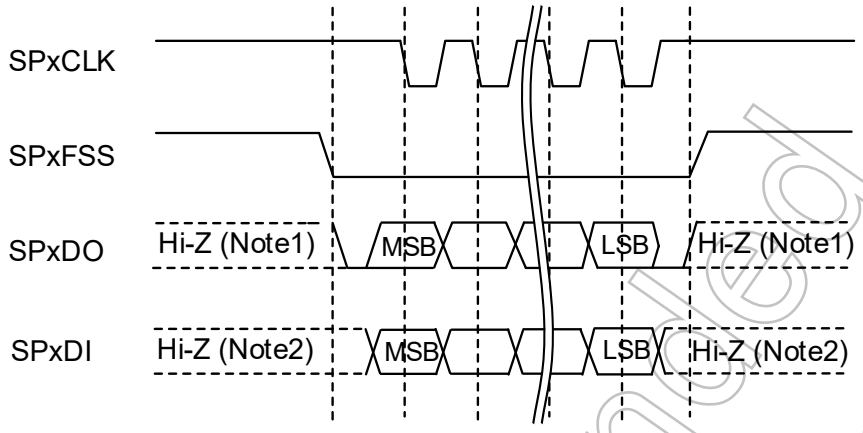


Figure 17-7 SPI frame format (single transfer,<SPO>="1" & <SPH>="0")

Figure 17-7 shows the SPI frame format with <SPO>=1 &<SPH>=0, which is for single transmission signal sequence.

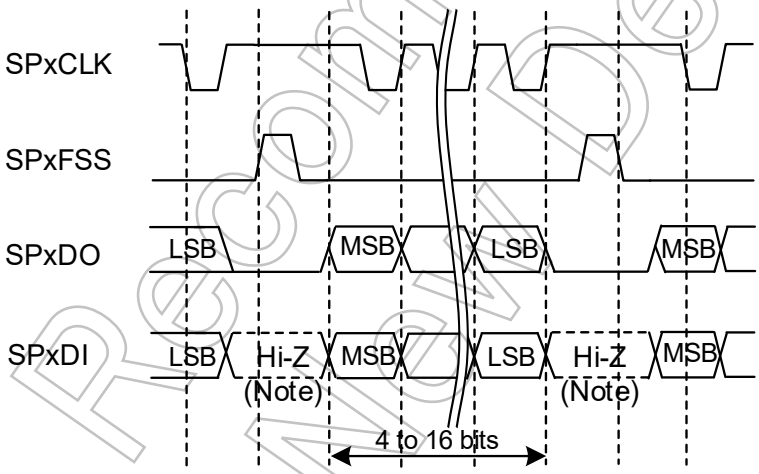


Figure 17-8 SPI frame format (continuous transfer,<SPO>="1" & <SPH>="0")

Figure 17-8 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.

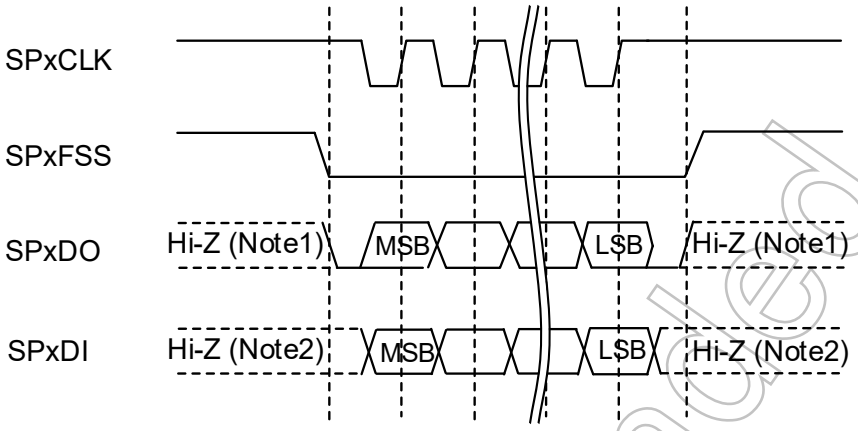


Figure 17-9 SPI frame format (Single & continuous transfer,<SPO>="1" & <SPH>="1")

Figure 17-9 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.

17.6.3 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.

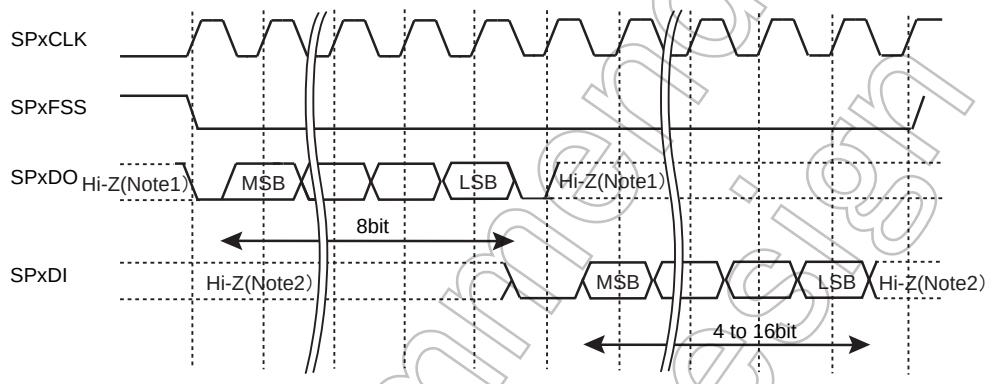


Figure 17-10 Microwire frame format (single transfer)

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 tristated 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".

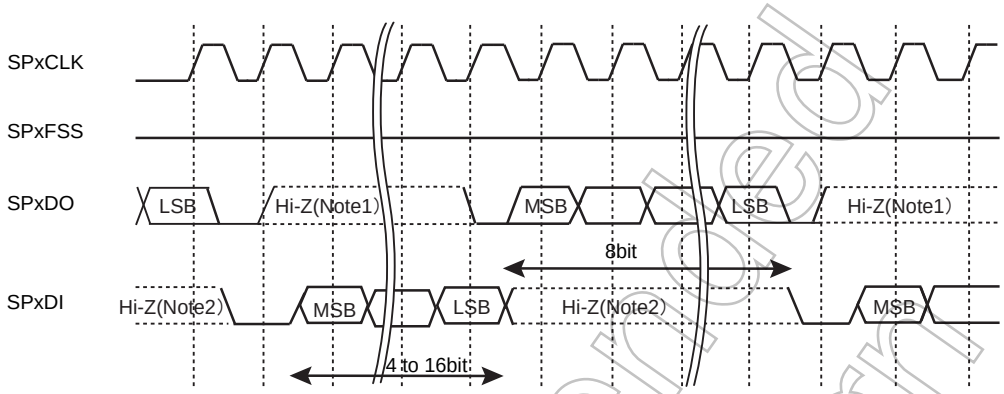


Figure 17-11 Microwire frame format (continuous transfer)

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.

18. Serial Bus Interface (I2C/SIO)

The TMPM366FDXBG/FYXBG/FWXBG contains 2 Serial Bus Interface (I2C/SIO) channels, in which the following two operating modes are included:

- I2C bus mode (with multi-master capability)
- Clock-synchronous 8-bit SIO mode

In the I2C bus mode, the I2C/SIO is connected to external devices via SCL and SDA.

In the clock-synchronous 8-bit SIO mode, the I2C/SIO is connected to external devices via SCK, SI and SO.

The following table shows the programming required to put the I2C/SIO in each operating mode.

Table 18-1 Port settings for using serial bus interface

channel	Operating mode	pin	Port Function Register	Port Output Control Register	Port Input Control Register	Port Open Drain Output Control Register
SBI0	I2C bus mode	SCL0 :PG1 SDA0 :PG0	PGFR1[1:0] = 11	PGCR[1:0] = 11	PGIE[1:0] = 11	PGOD[1:0] = 11
	SIO mode	SCK0 :PG2 SI0 :PG1 SO0 :PG0	PGFR1[2:0] = 111	PGCR[2:0] = 101(SCK0 output) PGCR[2:0] = 001(SCK0 input)	PGIE[2:0] = 010(SCK0 output) PGIE[2:0] = 110(SCK0 input)	PGOD[2:0] = xxx
SBI1	I2C bus mode	SCL1 :PE5 SDA1 :PE4	PEFR1[5:4] = 11	PECR[5:4] = 11	PEIE[5:4] = 11	PEOD[5:4] = 11
	SIO mode	SCK1 :PE6 SI1 :PE5 SO1 :PE4	PEFR1[6:4] = 111	PECR[6:4] = 101(SCK1 output) PECR[6:4] = 001(SCK1 input)	PEIE[6:4] = 010(SCK1 output) PEIE[6:4] = 110(SCK1 input)	PEOD[6:4] = xxx

Note:x: Don't care

18.1 Configuration

The configuration is shown in Figure 18-1.

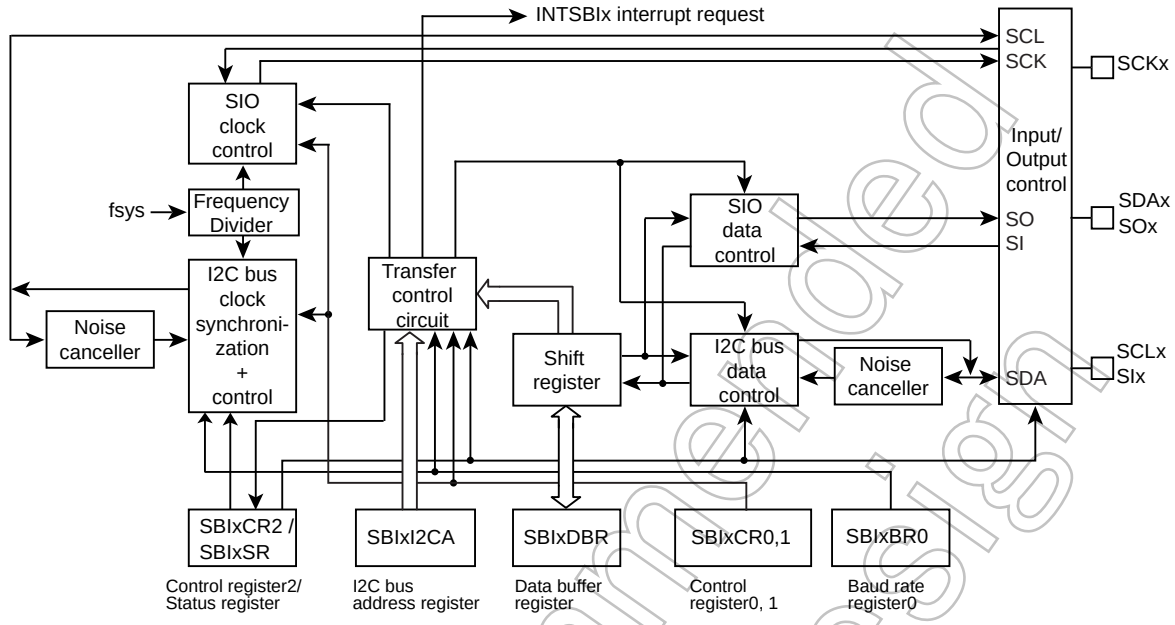


Figure 18-1 (I2C/SIO) Block Interface

18.2 Register

The following registers control the serial bus interface and provide its status information for monitoring.

The register below performs different functions depending on the mode. For details, refer to "18.4 Control Registers in the I2C Bus Mode" and "18.7 Control register of SIO mode".

18.2.1 Registers for each channel

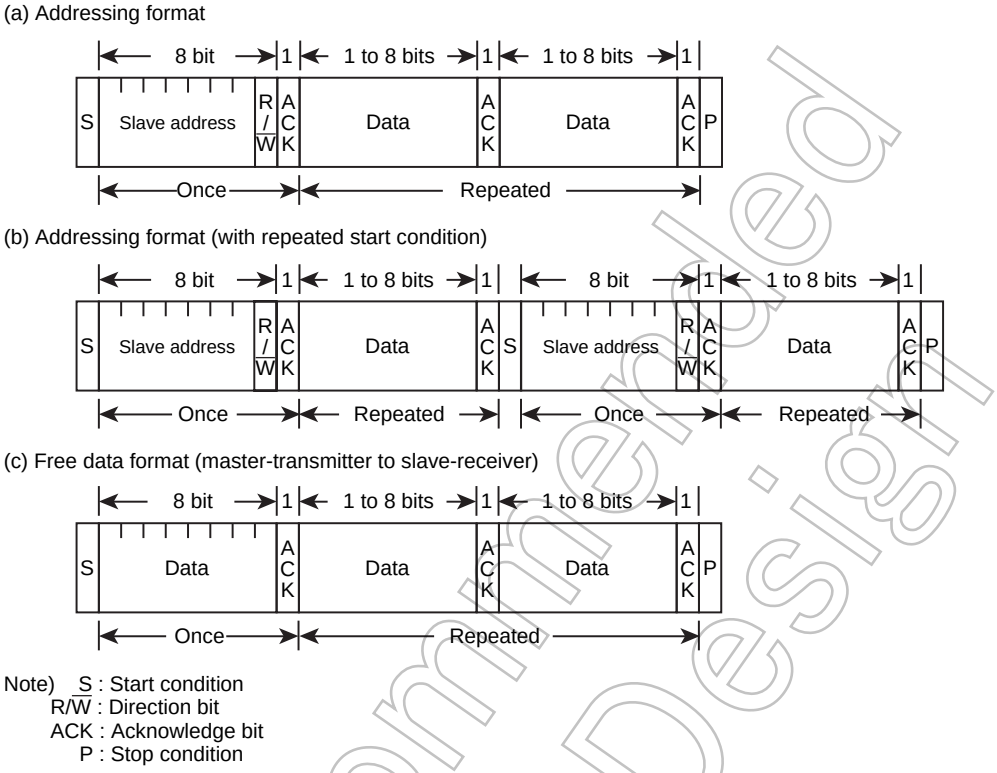
The tables below show the registers and register addresses for each channel.

Channel x	Base Address
Channel0	0x400E_0000
Channel1	0x400E_0100

Register name(x=0,1,)		Address(Base+)
Control register 0	SBIxCR0	0x0000
Control register 1	SBIxCR1	0x0004
Data buffer register	SBIxDBR	0x0008
I2C bus address register	SBIxI2CAR	0x000C
Control register 2	SBIxCR2 (writing)	0x0010
Status register	SBIxSR (reading)	
Baud rate register 0	SBIxBR0	0x0014

18.3 I2C Bus Mode Data Format

Figure 18-2 shows the data formats used in the I2C bus mode.



Note) S : Start condition
R/W : Direction bit
ACK : Acknowledge bit
P : Stop condition

Figure 18-2 I2C Bus Mode Data Formats

18.4 Control Registers in the I2C Bus Mode

The following registers control the serial bus interface in the I2C bus mode and provide its status information for monitoring.

18.4.1 SBIxCR0(Control register 0)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	SBIEN	R/W	Serial bus interface operation 0:Disable 1:Enable To use the serial bus interface, enable this bit first. For the first time in case of setting to enable, the relevant SBI registers can be read or written. Since all clocks except SBIxCR0 stop if this bit is disabled, power consumption can be reduced by disabling this bit. If this bit is disabled after it's been enabled once, the settings of each register are retained.
6-0	-	R	Read as 0.

Note: To use the serial bus interface, enable this bit first.

18.4.2 SBIxCR1(Control register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	BC			ACK	-	SCK2	SCK1	SCK0 / SWRMON
After reset	0	0	0	0	1	0	0	1(Note3)

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-5	BC[2:0]	R/W	Select the number of bits per transfer (Note 1)
4	ACK	R/W	Master mode 0: Acknowledgement clock pulse is not generated. 1: Acknowledgement clock pulse is generated. Slave mode 0: Acknowledgement clock pulse is not counted. 1: Acknowledgement clock pulse is counted.
3	-	R	Read as 1.
2-1	SCK[2:1]	R/W	Select internal SCL output clock frequency (Note 2).
0	SCK[0]	W	
	SWRMON	R	On reading <SWRMON>: Software reset status monitor 0:Software reset operation is in progress. 1:Software reset operation is not in progress.

<BC>	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

000	n = 5	462 kHz	System Clock: fsys (= 48MHz) Clock gear : fc/1 Frequency = $\frac{fsys}{2^n + 72}$ [Hz]
001	n = 6	353 kHz	
010	n = 7	240 kHz	
011	n = 8	146 kHz	
100	n = 9	82 kHz	
101	n = 10	44 kHz	
110	n = 11	23 kHz	
111		reserved	

Note 1: **Clear <BC[2:0]> to "000" before switching the operation mode to the SIO mode.**

Note 2: **For details on the SCL line clock frequency, refer to "18.5.1 Serial Clock".**

Note 3: **After a reset, the <SCK[0]/SWRMON> bit is read as "1". However, if the SIO mode is selected at the SBIxCR2 register, the initial value of the <SCK[0]> bit is "0".**

Note 4: **The initial value for selecting a frequency is <SCK[2:0]>=000 and is independent of the read initial value.**

Note 5: **When <BC[2:0]>="001" and <ACK>="0" in master mode, SCL line may be fixed to "L" by falling edge of SCL line after generation of STOP condition and the other master devices can not use the bus. In the case of bus which is connected with several master devices, the bumber of bits per transfer should be set equal or more than 2 before generation of STOP condition.**

18.4.3 SBIxCR2(Control register 2)

This register serves as SBIxSR register by reading it.

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	MST	W	Select master/slave 0: Slave mode 1: Master mode
6	TRX	W	Select transmit/ receive 0: Receive 1: Transmit
5	BB	W	Start/stop condition generation 0: Stop condition generated 1: Start condition generated
4	PIN	W	Clear INTSBIx interrupt request 0: − 1: Clear interrupt request
3-2	SBIM[1:0]	W	Select serial bus interface operating mode (Note) 00: Port mode (Disables a serial bus interface output) 01: SIO mode 10: I2C bus mode 11: Reserved
1-0	SWRST[1:0]	W	Software reset generation Write "10" followed by "01" to generate a reset. When writing, set <SBIM[1:0]> to "10"; I2Cbus mode.

Note: **Make sure that modes are not changed during a communication session.Ensure that the bus is free before switching the operating mode to the port mode. Ensure that the port is at the "High" level before switching the operating mode from the port mode to the I2C bus or clock-synchronous 8-bit SIO mode.**

18.4.4 SBIxSR (Status Register)

This register serves as SBIxCR2 by writing to it.

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	MST	R	Master/slave selection monitor 0: Slave mode 1: Master mode
6	TRX	R	Transmit/receive selection monitor 0: Receive 1: Transmit
5	BB	R	I2C bus state monitor 0: Free 1: Busy
4	PIN	R	INTSBIx interrupt request monitor 0:Interrupt request generated 1: Interrupt request cleared
3	AL	R	Arbitration lost detection 0: − 1:Detected
2	AAS	R	Slave address match detection 0: − 1: Detected (This bit is set when the general call is detected as well.)
1	ADO	R	General call detection 0: − 1:Detected
0	LRB	R	Last received bit monitor 0:Last received bit "0" 1:Last received bit "1"

18.4.5 SBIxBR0(Serial bus interface baud rate register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	I2SBI	-	-	-	-	-	-
After reset	1	0	1	1	1	1	1	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	-	R	Read as 1.
6	I2SBI	R/W	Operation at the IDLE mode 0: Stop 1: Operate
5-1	-	R	Read as 1.
0	-	R/W	Be sure to write "0".

18.4.6 SBIxDBR (Serial bus interface data buffer register)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-0	DB[7:0]	R (Receive)/ W (Transmit)	Receive data / Transmit data

Note 1: **The transmission data must be written in to the register from the MSB (bit 7). The received data is stored in the LSB.**

Note 2: **Since SBIxBDR has independent buffers for writing and reading, a written data cannot be read. Thus, read-modify-write instructions, such as bit manipulation, cannot be used.**

18.4.7 SBIxI2CAR (I2Cbus address register)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-1	SA[6:0]	R/W	Set the slave address when the SBI acts as a slave device.
0	ALS	R/W	Specify address recognition mode. 0: Recognize its slave address. 1: Do not recognize its slave address (free-data format).

Note 1: **Please set the bit 0 <ALS> of I2C bus address register SBIxI2CAR to "0", except when you use a free data format. It operates as a free data format when setting it to "1". Selecting the master fixes to transmission. Selecting the slave fixes to reception.**

Note 2: **Do not set SBIxI2CAR to "0x00" in slave mode. (If SBIxI2CAR is set to "0x00", it's recognized that the slave address matches the START byte ("0x01") of the I2C standard received in slave mode.)**

18.5 Control in the I2C Bus Mode

18.5.1 Serial Clock

18.5.1.1 Clock source

SBIxCR1<SCK[2:0]> specifies the maximum frequency of the serial clock to be output from the SCL pin in the master mode.

$t_{LOW} = 2^{n-1}/fsys + 58/fsys$
$t_{HIGH} = 2^{n-1}/fsys + 14/fsys$
$fscl = 1/(t_{LOW} + t_{HIGH})$
$= \frac{fsys}{2^n + 72}$

SBIxCR1<SCK[2:0]>	n
000	5
001	6
010	7
011	8
100	9
101	10
110	11

Figure 18-3 Clock source

Note: **The maximum speeds in the standard and high-speed modes are specified to 100kHz and 400kHz respectively following the communications standards. Notice that the internal SCL clock frequency is determined by the fsys used and the calculation formula shown above.**

18.5.1.2 Clock Synchronization

The I2C bus is driven by using the wired-AND connection due to its pin structure. The first master that pulls its clock line to the "Low" level overrides other masters producing the "High" level on their clock lines. This must be detected and responded by the masters producing the "High" level.

Clock synchronization assures correct data transfer on a bus that has two or more master.

For example, the clock synchronization procedure for a bus with two masters is shown below.

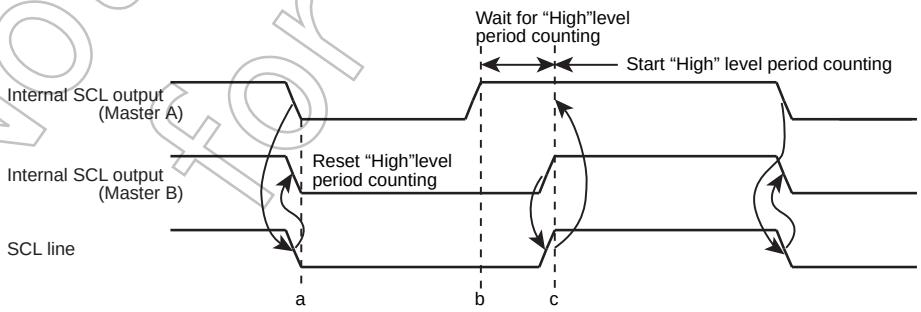


Figure 18-4 Example of Clock Synchronization

At the point a, Master A pulls its internal SCL output to the "Low" level, bringing the SCL bus line to the "Low" level. Master B detects this transition, resets its "High" level period counter, and pulls its internal SCL output level to the "Low" level.

Master A completes counting of its "Low" level period at the point b, and brings its internal SCL output to the "High" level. However, Master B still keeps the SCL bus line at the "Low" level, and Master A stops counting of its "High" level period counting.After Master A detects that Master B brings its internal SCL output to the "High" level and brings the SCL bus line to the "High" level at the point c, it starts counting of its "High" level period.

After that Master finishes counting the "High" level period, the Master pulls the SCL pin to "Low" and the SCL bus line becomes "Low".

This way, the clock on the bus is determined by the master with the shortest "High" level period and the master with the longest "Low" level period among those connected to the bus.

18.5.2 Setting the Acknowledgement Mode

Setting SBIxCR1<ACK> to "1" selects the acknowledge mode.When operating as a master, the SBI adds one clock for acknowledgment signal. In slave mode, the clock for acknowledgement signals is counted. In transmitter mode, the SBI releases the SDAx pin during clock cycle to receive acknowledgement signals from the receiver. In receiver mode, the SBI pulls the SDAx pin to the "Low" level during the clock cycle and generates acknowledgement signals. Also in slave mode, if a general-call address is received, the SBI pulls the SDAx pin to the "Low" level during the clock cycle and generates acknowledgement signals.

By setting <ACK> to "0", the non-acknowledgment mode is activated. When operating as a master, the SBI does not generate clock for acknowledgement signals. In slave mode, the clock for acknowledgement signals is counted.

18.5.3 Setting the Number of Bits per Transfer

SBIxCR1<BC[2:0]> specifies the number of bits of the next data to be transmitted or received.

Under the start condition, <BC[2:0]> is set to "000", causing a slave address and the direction bit to be transferred in a packet of eight bits. At other times, <BC[2:0]> keeps a previously programmed value.

18.5.4 Slave Addressing and Address Recognition Mode

Setting "0" to SBIxI2CAR<ALS> and a slave address in SBIxI2CAR<SA[6:0]> sets addressing format, and then the SBI recognizes a slave address transmitted by the master device and receives data in the addressing format.

If <ALS> is set to "1", the SBI does not recognize a slave address and receives data in the free data format. In the case of free data format, a slave address and a direction bit are not recognized; they are recognized as data immediately after generation of the start condition.

18.5.5 Operating mode

The setting of SBIxCR2<SBIM[1:0]> controls the operating mode. To operate in I2C mode, ensure that the serial bus interface pins are at "High" level before setting <SBIM[1:0]> to "10". Also, ensure that the bus is free before switching the operating mode to the port mode.

18.5.6 Configuring the SBI as a Transmitter or a Receiver

Setting SBIxCR2<TRX> to "1" configures the SBI as a transmitter. Setting <TRX> to "0" configures the SBI as a receiver.

At the slave mode:

- when data is transmitted in the addressing format.
- when the received slave address matches the value specified at SBIxI2CAR.
- when a general-call address is received; i.e., the eight bits following the start condition are all zeros.

If the value of the direction bit (R/$\overline{W}$) is "1", <TRX> is set to "1" by the hardware. If the bit is "0", <TRX> is set to "0".

As a master device, the SBI receives acknowledgement from a slave device. If the direction bit of "1" is transmitted, <TRX> is set to "0" by the hardware. If the direction bit is "0", <TRX> changes to "1". If the SBI does not receive acknowledgement, <TRX> retains the previous value.

<TRX> is cleared to "0" by the hardware when it detects the stop condition on the bus or the arbitration lost.

If SBI is used in free data format, <TRX> is not changed by the hardware.

18.5.7 Configuring the SBI as a Master or a Slave

Setting SBIxCR2<MST> to "1" configures the SBI to operate as a master device.

Setting <MST> to "0" configures the SBI as a slave device. <MST> is cleared to "0" by the hardware when it detects the stop condition on the bus or the arbitration lost.

18.5.8 Generating Start and Stop Conditions

When SBIxSR<BB> is "0", writing "1" to SBIxCR2<MST, TRX, BB, PIN> causes the SBI to start a sequence for generating the start condition and to output the slave address and the direction bit prospectively written in the data buffer register. <ACK> must be set to "1" in advance.

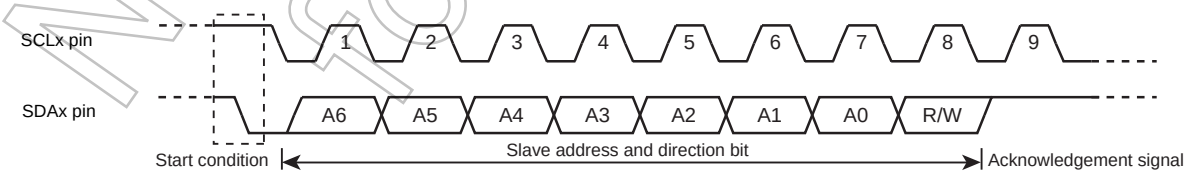


Figure 18-5 Generating the Start Condition and a Slave Address

When <BB> is "1", writing "1" to <MST, TRX, PIN> and "0" to <BB> causes the SBI to start a sequence for generating the stop condition on the bus. The contents of <MST, TRX, BB, PIN> should not be altered until the stop condition appears on the bus.

If SCL bus line is pulled "Low" by other devices when the stop condition is generated, the stop condition is generated after the SCL line is released.

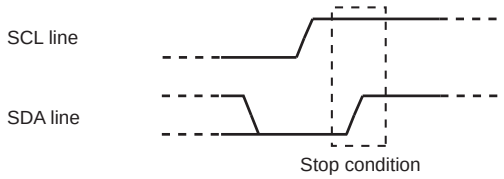


Figure 18-6 Generating the Stop Condition

SBIxSR<BB> can be read to check the bus state. <BB> is set to "1" when the start condition is detected on the bus (the bus is busy), and cleared to "0" when the stop condition is detected (the bus is free).

18.5.9 Interrupt Service Request and Release

In master mode, a serial bus interface request (INTSBIx) is generated when the transfer of the number of clock cycles set by <BC> and <ACK> is completed.

In slave mode, INTSBIx is generated under the following conditions.

- After output of the acknowledge signal which is generated when the received slave address matches the slave address set to SBIxI2CAR<SA[6:0]>.
- After the acknowledge signal is generated when a general-call address is received.
- When the slave address matches or a data transfer is completed after receiving a general-call address.

In the address recognition mode (<ALS> = "0"), INTSBIx is generated when the received slave address matches the values specified at SBIxI2CAR or when a general-call (eight bits data following the start condition is all "0") is received.

When an interrupt request (INTSBIx) is generated, SBIxCR2<PIN> is cleared to "0". While <PIN> is cleared to "0", the SBI pulls the SCL line to the "Low" level.

<PIN> is set to "1" when data is written to or read from SBIxDBR. It takes a period of t_{LOW} for the SCL line to be released after <PIN> is set to "1". When the program writes "1" to <PIN>, it is set to "1". However, writing "0" does not clear this bit to "0".

Note: When arbitration is lost in master mode, <PIN> is not cleared to "0" if the slave address does not match (INTSBIx is generated).

18.5.10 Arbitration Lost Detection Monitor

The I2C bus has the multi-master capability (there are two or more masters on a bus), and requires the bus arbitration procedure to ensure correct data transfer.

A master that attempts to generate the start condition while the bus is busy loses bus arbitration, with no start condition occurring on the SDA and SCL lines.The I2C-bus arbitration takes place on the SDA line.

The arbitration procedure for two masters on a bus is shown below.

Up until the point a, Master A and Master B output the same data. At the point a, Master A outputs the "Low" level and Master B outputs the "High" level.

Then Master A pulls the SDA bus line to the "Low" level because the line has the wired-AND connection. When the SCL line goes high at the point b, the slave device reads the SDA line data, i.e., data transmitted by Master A. At this time, data transmitted by Master B becomes invalid.

This condition of Master B is called "Arbitration Lost". Master B releases its SDA pin, so that it does not affect the data transfer initiated by another master. If two or more masters have transmitted exactly the same first data word, the arbitration procedure continues with the second data word.

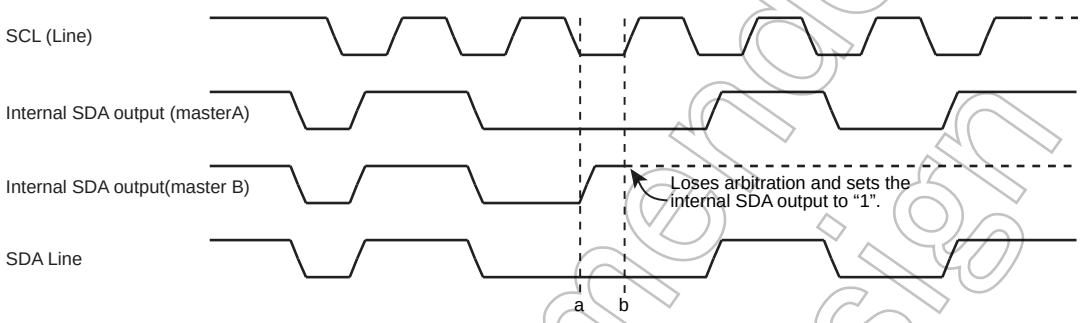


Figure 18-7 Lost Arbitration

A master compares the SDA bus line level and the internal SDA output level at the rising of the SCL line. If there is a difference between these two values, Arbitration Lost occurs and SBIxSR<AL> is set to "1".

When <AL> is set to "1", SBIxSR<MST, TRX> are cleared to "0", causing the SBI to operate as a slave receiver.Therefore, the serial bus interface circuit stops the clock output during data transfer after <AL> is set to "1".

<AL> is cleared to "0" when data is written to or read from SBIxDBR or data is written to SBIxCR2.

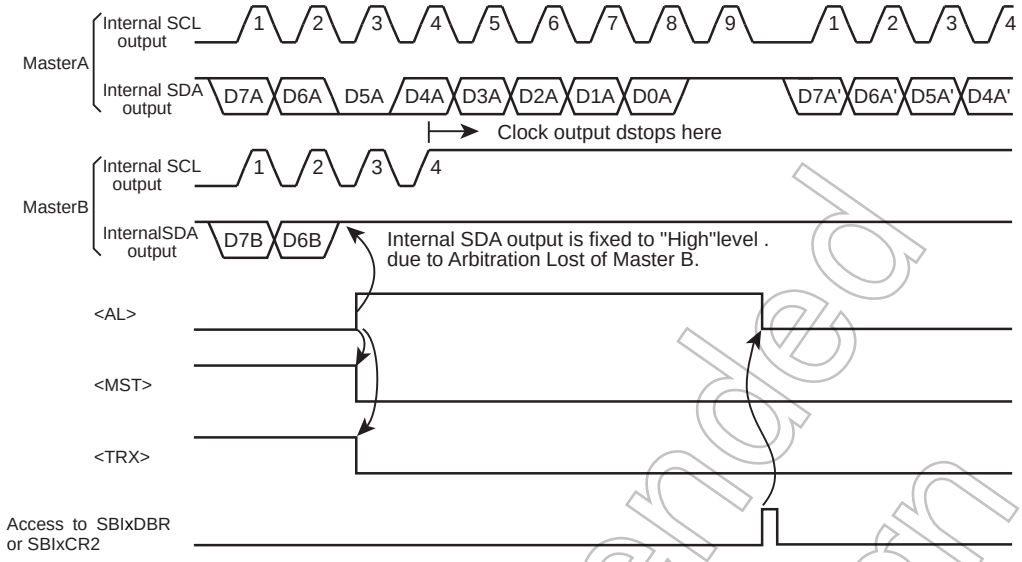


Figure 18-8 Example of Master B Lost Arbitration (D7A = D7B, D6A = D6B)

18.5.11 Slave Address Match Detection Monitor

When the SBI operates as a slave device in the address recognition mode (SBIxI2CAR<ALS>="0"), SBIxSR<AAS> is set to "1" on receiving the general-call address or the slave address that matches the value specified at SBIxI2CAR.

When <ALS> is "1", <AAS> is set to "1" when the first data word has been received. <AAS> is cleared to "0" when data is written to or read from SBIxDBR.

18.5.12 General-call Detection Monitor

When the SBI operates as a slave device, SBIxSR<ADO> is set to "1" when it receives the general-call address; i.e., the eight bits following the start condition are all zeros.

<ADO> is cleared to "0" when the start or stop condition is detected on the bus.

18.5.13 Last Received Bit Monitor

SBIxSR<LRB> is set to the SDA line value that was read at the rising of the SCL line.

In the acknowledgment mode, reading SBIxSR<LRB> immediately after generation of the INTSBIx interrupt request causes ACK signal to be read.

18.5.14 Data Buffer Register (SBIxDBR)

Reading or writing SBIxDBR initiates reading received data or writing transmitted data.

When the SBI is acting as a master, setting a slave address and a direction bit to this register generates the start condition.

18.5.15 Baud Rate Register (SBIxBR0)

The SBIxBR0<I2SBI> register determines if the SBI operates or not when it enters the IDLE mode.

This register must be programmed before executing an instruction to switch to the standby mode.

18.5.16 Software Reset

If the serial bus interface circuit locks up due to external noise, it can be initialized by using a software reset.

Writing "10" followed by "01" to SBIxCR2<SWRST[1:0]> generates a reset signal that initializes the serial bus interface circuit. When writing, set <SBIM[1:0]> to "10"; I2Cbus mode. After a reset, all control registers and status flags are initialized to their reset values. When the serial bus interface is initialized, <SWRST> is automatically cleared to "0".

Note: A software reset causes the SBI operating mode to switch from the I2C mode to the port mode.

18.6 Data Transfer Procedure in the I2C Bus ModeI2C

18.6.1 Device Initialization

First, program SBIxCR1<ACK, SCK[2:0]>. Writing "000" to SBIxCR1<BC[2:0]> at the time.

Next, program SBIxI2CAR by specifying a slave address at <SA[6:0]> and an address recognition mode at <ALS>. (<ALS> must be cleared to "0" when using the addressing format).

To configure the Serial Bus Interface as a slave receiver, ensure that the serial bus interface pin is at "High" first. Then write "0" to SBIxCR2<MST, TRX, BB>, "1" to <PIN>, "10" to <SBIM[1:0]> and "0" to the bit 1 and 0.

Note: Initialization of the serial bus interface circuit must be completed within a period that any device does not generate start condition after all devices connected to the bus were initialized. If this rule is not followed, data may not be received correctly because other devices may start transfer before the initialization of the serial bus interface circuit is completed.

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	0	0	X	0	X	X	X	Specifies ACK and SCL clock.
SBIxI2CAR	←	X	X	X	X	X	X	X	X	Specifies a slave address and an address recognition mode.
SBIxCR2	←	0	0	0	1	1	0	0	0	Configures the SBI as a slave receiver.

Note: X; Don't care

18.6.2 Generating the Start Condition and a Slave Address

18.6.2.1 Master mode

In the master mode, the following steps are required to generate the start condition and a slave address.

First, ensure that the bus is free (<BB> = "0"). Then, write "1" to SBIxCR1<ACK> to select the acknowledgment mode. Write to SBIxDBR a slave address and a direction bit to be transmitted.

When <BB> = "0", writing "1111" to SBIxCR2<MST, TRX, BB, PIN> generates the start condition on the bus. Following the start condition, the SBI generates nine clocks from the SCL pin. The SBI outputs the slave address and the direction bit specified at SBIxDBR with the first eight clocks, and releases the SDA line in the ninth clock to receive an acknowledgment signal from the slave device.

The INTSBIx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to "0". In the master mode, the SBI holds the SCL line at the "Low" level while <PIN> is = "0".<TRX> changes its value according to the transmitted direction bit at generation of the INTSBIx interrupt request, provided that an acknowledgment signal has been returned from the slave device.

Note: To output salve address, check with software that the bus is free before writing to SBIxDBR. If this rule is not followed, data being output on the bus may get ruined.

Settings in main routine

```
                        7  6  5  4  3  2  1  0
Reg.          ←  SBIxSR
Reg.          ←  Reg. e 0x20
if Reg.       ≠  0x00                          Ensures that the bus is free.
Then
SBIxCR1       ←  X  X  X  1  0  X  X  X        Selects the acknowledgement mode.
SBIxDBR       ←  X  X  X  X  X  X  X  X        Specifies the desired slave address and direction.
SBIxCR2       ←  1  1  1  1  1  0  0  0        Generates the start condition.
Example of INTSBI0 interrupt routine
    Clears the interrupt request.
    Processing
    End of interrupt
```

18.6.2.2 Slave mode

In the slave mode, the SBI receives the start condition and a slave address.

After receiving the start condition from the master device, the SBI receives a slave address and a direction bit from the master device during the first eight clocks on the SCL line.

If the received address matches its slave address specified at SBIxI2CAR or is equal to the general-call address, the SBI pulls the SDA line to the "Low" level during the ninth clock and outputs an acknowledgment signal.

The INTSBIx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to "0". In the slave mode, the SBI holds the SCL line at the "Low" level while <PIN> is "0".

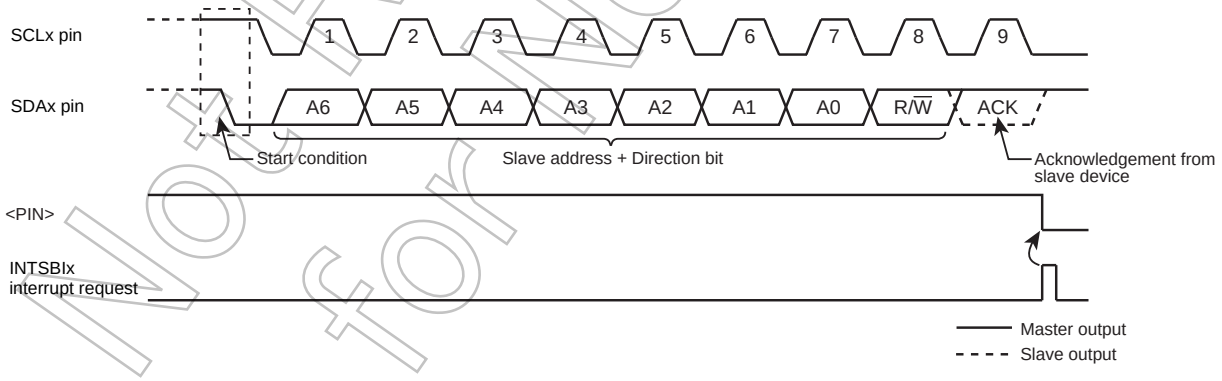


Figure 18-9 Generation of the Start Condition and a Slave Address

18.6.3 Transferring a Data Word

At the end of a data word transfer, the INTSBIx interrupt is generated to test <MST> to determine whether the SBI is in the master or slave mode.

18.6.3.1 Master mode (<MST> = "1")

Test <TRX> to determine whether the SBI is configured as a transmitter or a receiver.

(1) Transmitter mode (<TRX> = "1")

Test <LRB>. If <LRB> is "1", that means the receiver requires no further data.

The master then generates the stop condition as described later to stop transmission.

If <LRB> is "0", that means the receiver requires further data.If the next data to be transmitted has eight bits, the data is written into SBIxDBR. If the data has different length, <BC[2:0]> and <ACK> are programmed and the transmit data is written into SBIxDBR.Writing the data makes <PIN> to "1", causing the SCL pin to generate a serial clock for transferring a next data word, and the SDA pin to transfer the data word.

After the transfer is completed, the INTSBIx interrupt request is generated, <PIN> is cleared to "0", and the SCL pin is pulled to the "Low" level.

To transmit more data words, test <LRB> again and repeat the above procedure.

INTSBIx interrupt

```
if MST = 0
Then go to the slave-mode processing.
if TRX = 0
Then go to the receiver-mode processing.
if LRB = 0
Then go to processing for generating the stop condition.
SBIxCR1   ←  X  X  X  X  0  X  X  X   Specifies the number of bits to be transmitted and
                                      specify whether ACK is required.
SBIxDBR   ←  X  X  X  X  X  X  X  X   Writes the transmit data.
End of interrupt processing.
```

Note: **X; Don't care**

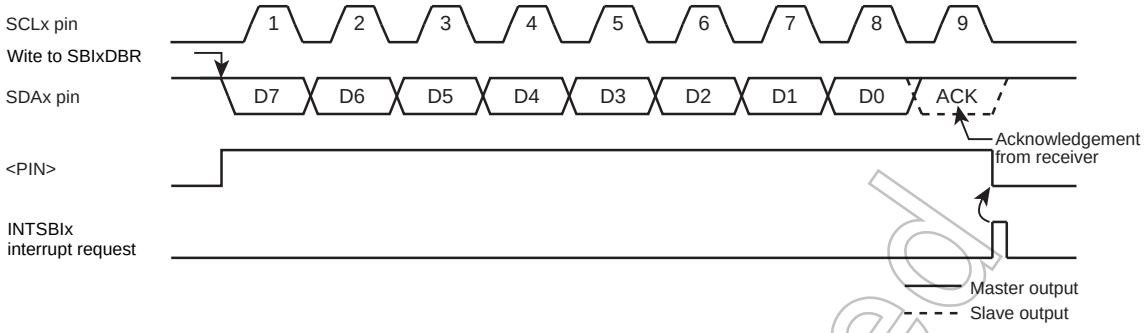


Figure 18-10 <BC[2:0]>= "000",<ACK>= "1" (Transmitter Mode)

(2) Receiver mode (<TRX> = "0")

If the next data to be transmitted has eight bits, the transmit data is written into SBIxDBR.

If the data has different length, <BC[2:0]> and <ACK> are programmed and the received data is read from SBIxDBR to release the SCL line. (The data read immediately after transmission of a slave address is undefined.)On reading the data, <PIN> is set to "1", and the serial clock is output to the SCL pin to transfer the next data word.In the last bit, when the acknowledgment signal becomes the "Low" level, "0" is output to the SDA pin.

After that, the INTSBIx interrupt request is generated, and <PIN> is cleared to "0", pulling the SCL pin to the "Low" level.Each time the received data is read from SBIxDBR, one-word transfer clock and an acknowledgement signal are output.

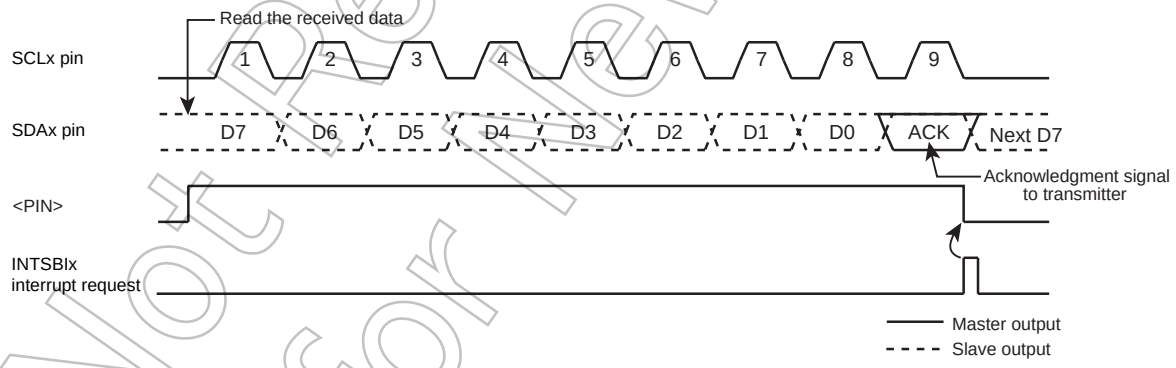


Figure 18-11 <BC[2:0]>= "000",<ACK>= "1" (Receiver Mode)

To terminate the data transmission from the transmitter, <ACK> must be cleared to "0" immediately before reading the data word second to last.

This disables generation of an acknowledgment clock for the last data word.

When the transfer is completed, an interrupt request is generated. After the interrupt processing, <BC[2:0]> must be set to "001" and the data must be read so that a clock is generated for 1-bit transfer.

At this time, the master receiver holds the SDA bus line at the "High" level, which signals the end of transfer to the transmitter as an acknowledgment signal.

In the interrupt processing for terminating the reception of 1-bit data, the stop condition is generated to terminate the data transfer.

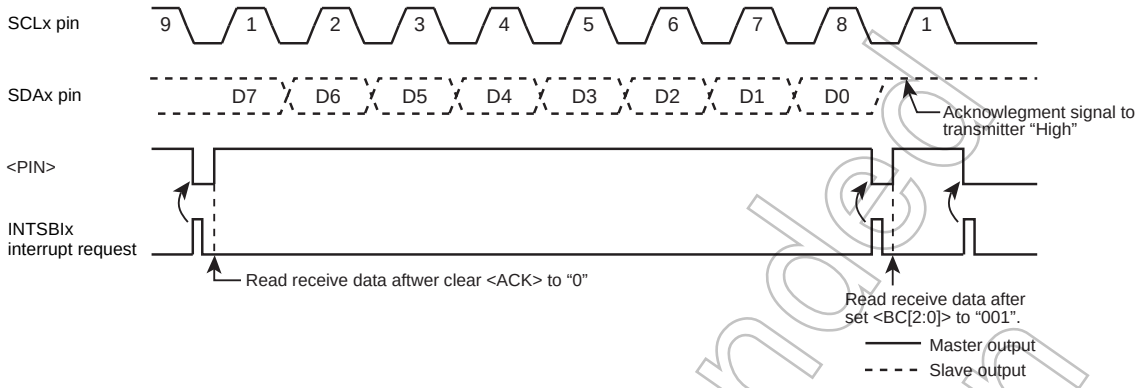


Figure 18-12 Terminating Data Transmission in the Master Receiver Mode

Example: When receiving N data word

INTSBIx interrupt (after data transmission)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	X	X	X	X	0	X	X	X	Sets the number of bits of data to be received and specify whether ACK is required.
Reg.	←	SBIxDBR								Reads dummy data.

End of interrupt

INTSBIx interrupt (first to (N-2)th data reception)

		7	6	5	4	3	2	1	0	
Reg.	←	SBIxDBR								Reads the first to (N-2)th data words.

End of interrupt

INTSBIx interrupt ((N-1)th data reception)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	X	X	X	0	0	X	X	X	Disables generation of acknowledgement clock.
Reg.	←	SBIxDBR								Reads the (N-1)th data word.

End of interrupt

INTSBIx interrupt (Nth data reception)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	0	1	0	0	X	X	X	Disables generation of acknowledgement clock.
Reg.	←	SBIxDBR								Reads the Nth data word.

End of interrupt

INTSBIx interrupt (after completing data reception)

Processing to generate the stop condition.	Terminates the data transmission.

End of interrupt

Note: X; Don't care

18.6.3.2 Slave mode (<MST> = "0")

In the slave mode, the SBI generates the INTSBIx interrupt request on four occasions:

1) when the SBI has received any slave address from the master.

2) when the SBI has received a general-call address.

3) when the received slave address matches its address.

4) when a data transfer has been completed in response to a general-call.

Also, if the SBI detects Arbitration Lost in the master mode, it switches to the slave mode.

Upon the completion of data word transfer in which Arbitration Lost is detected, the INTSBIx interrupt request is generated, <PIN> is cleared to "0", and the SCL pin is pulled to the "Low" level.

When data is written to or read from SBIxDBR or when <PIN> is set to "1", the SCLx pin is released after a period of t_{LOW}.

In the slave mode, the normal slave mode processing or the processing as a result of Arbitration Lost is carried out.

SBIxSR<AL>, <TRX>, <AAS> and <ADO> are tested to determine the processing required.

"Table 18-2 Processing in Slave Mode"shows the slave mode states and required processing.

Example: When the received slave address matches the SBI's own address and the direction bit is "1" in the slave receiver mode.

INTSBIx interrupt

if TRX = 0

Then go to other processing.

if AL = 0

Then go to other processing.

if AAS = 0

Then go to other processing.

SBIxCR1	←	X	X	X	1	0	X	X	X	Sets the number of bits to be transmitted.
SBIxDBR	←	X	X	X	X	X	X	X	X	Sets the transmit data.

Note:X; Don't care

Table 18-2 Processing in Slave Mode

<TRX>	<AL>	<AAS>	<ADO>	State	Processing
1	1	1	0	Arbitration Lost is detected while the slave address was being transmitted and the SBI received a slave address with the direction bit "1" transmitted by another master.	Set the number of bits in a data word to <BC[2:0]> and write the transmit data into SBIxDBR.
	0	1	0	In the slave receiver mode, the SBI received a slave address with the direction bit "1" transmitted by the master.	
		0	0	In the slave transmitter mode, the SBI has completed a transmission of one data word.	Test LRB. If it has been set to "1", that means the receiver does not require further data. Set <PIN> to 1 and reset <TRX> to 0 to release the bus. If <LRB> has been reset to "0", that means the receiver requires further data. Set the number of bits in the data word to <BC[2:0]> and write the transmit data to the SBIxDBR.
0	1	1	1/0	Arbitration Lost is detected while a slave address is being transmitted, and the SBI receives either a slave address with the direction bit "0" or a general-call address transmitted by another master.	Read the SBIxDBR (a dummy read) to set <PIN> to 1, or write "1" to <PIN>.
		0	0	Arbitration Lost is detected while a slave address or a data word is being transmitted, and the transfer is terminated.	
	0	1	1/0	In the slave receiver mode, the SBI received either a slave address with the direction bit "0" or a general-call address transmitted by the master.	
		0	1/0	In the slave receiver mode, the SBI has completed a reception of a data word.	Set the number of bits in the data word to <BC[2:0].> and read the received data from SBIxDBR.

18.6.4 Generating the Stop Condition

When SBIxSR<BB> is "1", writing "1" to SBIxCR2<MST, TRX, PIN> and "0" to <BB> causes the SBI to start a sequence for generating the stop condition on the bus.

Do not alter the contents of <MST, TRX, BB, PIN> until the stop condition appears on the bus.

If another device is holding down the SCL bus line, the SBI waits until the SCL line is released.

After that, the SDA pin goes "High", causing the stop condition to be generated.

		7	6	5	4	3	2	1	0	
SBIxCR2	←	1	1	0	1	1	0	0	0	Generates the stop condition.

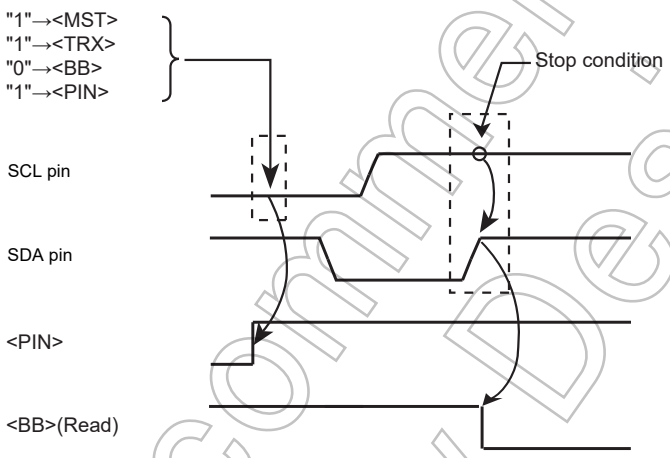


Figure 18-13 Generating the Stop Condition

18.6.5 Restart Procedure

Restart is used when a master device changes the data transfer direction without terminating the transfer to a slave device.The procedure of generating a restart in the master mode is described below.

First, write SBIxCR2<MST, TRX, BB> to "0" and write "1" to <PIN> to release the bus. At this time, the SDAx pin is held at the "High" level and the SCLx pin is released. Because no stop condition is generated on the bus, other devices recognize that the bus is busy.

Then, test SBIxSR<BB> and wait until it becomes "0" to ensure that the SCLx pin is released.

Next, test <LRB> and wait until it becomes "1" to ensure that no other device is pulling the SCLx bus line to the "Low" level.

Once the bus is determined to be free by following the above procedures, follow the procedures described in "18.6.2 Generating the Start Condition and a Slave Address"to generate the start condition.

To satisfy the setup time of restart, at least 4.7μs wait period (in the standard mode) must be created by the software after the bus is determined to be free.

Note 1: **Do not write <MST> to "0" when it is "0". (Restart cannot be initiated.)**

Note 2: **When the master device is acting as a receiver, data transmission from the slave device which serves as a transmitter must be completed before generating a restart. To complete data transfer, slave device must receive a "High" level acknowledge signal. For this reason, <LBR> before generating a restart becomes "1", the rising edge of the SCL line is not detected even <LBR>=**

"1" is confirmed by following the restart procedure. To check the status of the SCL line, read the port.

		7	6	5	4	3	2	1	0	
SBIxCR2	←	0	0	0	1	1	0	0	0	Releases the bus.
if SBIxSR<BB> ≠ 0										Checks that the SCL pin is released.
Then										
if SBIxSR<LRB> ≠ 1										Checks that no other device is pulling the SCL pin to the "Low".
Then										
4.7 μs Wait										
SBIxCR1	←	X	X	X	1	0	X	X	X	Selects the acknowledgment mode.
SBIxDBR	←	X	X	X	X	X	X	X	X	Sets the desired slave address and direction.
SBIxCR2	←	1	1	1	1	1	0	0	0	Generates the start condition.

Note: X; Don't care

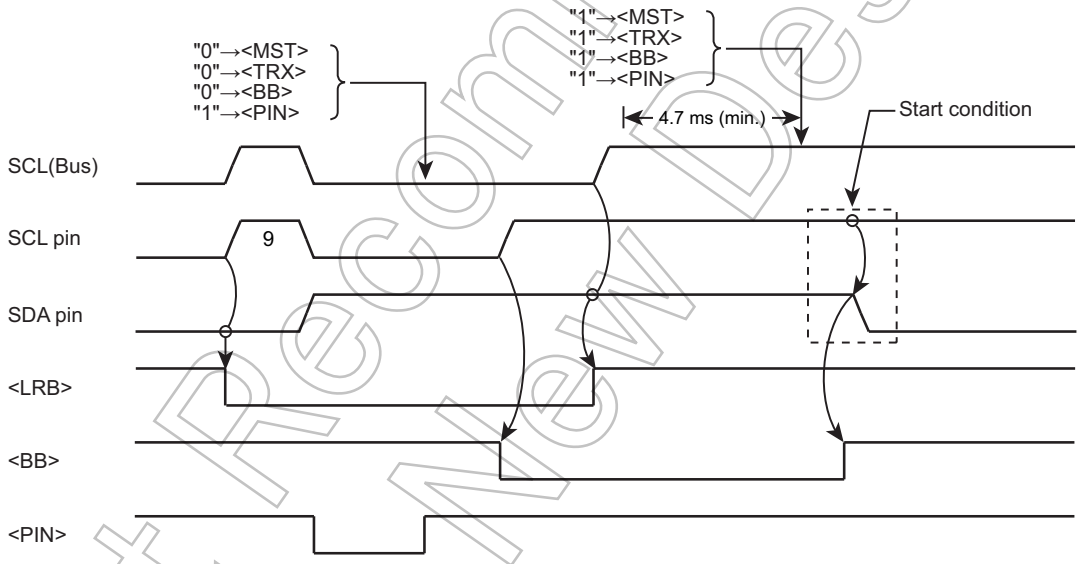


Figure 18-14 Timing Chart of Generating a Restart

18.7 Control register of SIO mode

The following registers control the serial bus interface in the clock-synchronous 8-bit SIO mode and provide its status information for monitoring.

18.7.1 SBIxCR0(control register 0)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	SBIEN	R/W	Serial bus interface operation. 0:Disable 1: Enable Enable this bit before using the serial bus interface. If this bit is disabled, power consumption can be reduced because all clocks except SBIxCR0 stop. If the serial bus interface operation is enabled and then disabled, the settings will be maintained in each register.
6-0	-	R	Read as 0.

18.7.2 SBIxCR1(Control register 1)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	SIOS	R/W	Transfer Start/Stop 0: Stop 1: Start
6	SIOINH	R/W	Transfer 0: Continue 1: Forced termination
5-4	SIOM[1:0]	R/W	Select transfer mode 00: Transmit mode 01: Reserved 10:Transmit/receive mode 11:Receive mode
3	-	R	Read as 1.
2-0	SCK[2:0]	R/W	On writing <SCK[2:0]>: Select serial clock frequency. (Note 1)

000	n = 3	3 MHz
001	n = 4	1.5 MHz
010	n = 5	750 kHz
011	n = 6	375 kHz
100	n = 7	187.5 kHz
101	n = 8	93.8 kHz
110	n = 9	46.9 kHz
111	–	External clock

System clock: fsys (=48MHz)
Clock gear: fc/1
Frequency = $\frac{fsys/2}{2^n}$ [Hz]

Note 1: **After a reset, the <SCK[0]> bit is read as "1". However, if the SIO mode is selected at the SBIxCR2 register, the initial value is read as "0". In this document, the value written in the column "after reset" is the value after setting the SIO mode in the initial state. The descriptions of the SBIxCR2 register and the SBIxSR register are the same.**

Note 2: **Set <SIOS> to "0" and <SIOINH> to "1" before programming the transfer mode and the serial clock.**

18.7.3 SBIxDBR (Data buffer register)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-0	DB[7:0]	R	Receive data
		W	Transmit data

Note 1: **The transmission data must be written in to the register from the MSB (bit 7). The received data is stored in the LSB.**

Note 2: **Since SBIxDBR has independent buffers for writing and reading, a written data cannot be read. Thus, read-modify-write instructions, such as bit manipulation, cannot be used.**

18.7.4 SBIxCR2(Control register 2)

This register serves as SBIxSR register by writing to it.

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7-4	-	R	Read as 1. (Note 1)
3-2	SBIM[1:0]	W	Select serial bus interface operating mode (Note 2) 00: Port mode 01: SIO mode 10: I2Cbus mode 11: Reserved
1-0	-	R	Read as 1. (Note 1)

Note 1: **In this document, the value written in the column "after reset" is the value after setting the SIO mode in the initial state.**

Note 2: **Make sure that modes are not changed during a communication session.**

18.7.5 SBIxSR (Status Register)

This register serves as SBIxCR2 by writing to it.

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-4	-	R	Read as 1.(Note 1)
3	SIOF	R	Serial transfer status monitor. 0: Completed 1: In progress
2	SEF	R	Shift operation status monitor 0: Completed. 1: In progress
1-0	-	R	Read as 1. (Note 1)

Note: **In this document, the value written in the column "after reset" is the value after setting the SIO mode in the initial state.**

18.7.6 SBIxBR0 (Baud rate register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	I2SBI	-	-	-	-	-	-
After reset	1	0	1	1	1	1	1	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	-	R	Read as 1.
6	I2SBI	R/W	Operation in IDLE mode. 0: Stop 1: Operate
5-1	-	R	Read as 1.
0	-	R/W	Make sure to write "0".

18.8 Control in SIO mode

18.8.1 Serial Clock

18.8.1.1 Clock source

Internal or external clocks can be selected by programming SBIxCR1<SCK[2:0]>.

(1) Internal clocks

In the internal clock mode, one of the seven frequencies can be selected as a serial clock, which is output to the outside through the SCKx pin.

At the beginning of a transfer, the SCKx pin output becomes the "High" level.

If the program cannot keep up with this serial clock rate in writing the transmit data or reading the received data, the SBI automatically enters a wait period. During this period, the serial clock is stopped automatically and the next shift operation is suspended until the processing is completed.

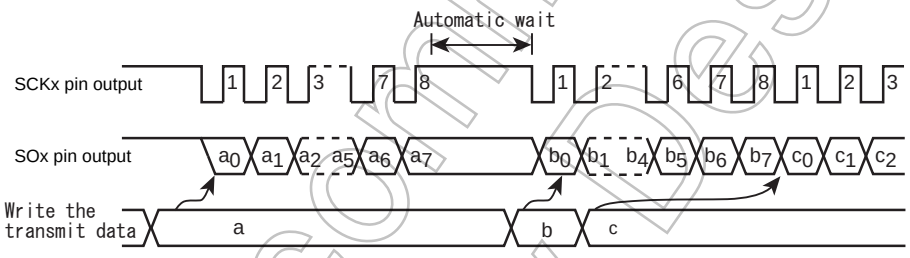


Figure 18-15 Automatic Wait

(2) External clock (<SCK[2:0]> = "111")

The SBI uses an external clock supplied from the outside to the SCKx pin as a serial clock.

For proper shift operations, the serial clock at the "High" and "Low" levels must have the pulse widths as shown below.

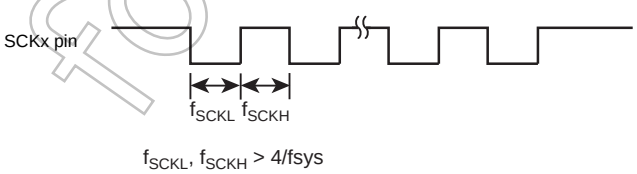


Figure 18-16 Maximum Transfer Frequency of External Clock Input

18.8.1.2 Shift Edge

Leading-edge shift is used in transmission. Trailing-edge shift is used in reception.

- Leading-edge shift

  Data is shifted at the leading edge of the serial clock (or the falling edge of the SCKx pin input/output).

- Trailing-edge shift

  Data is shifted at the trailing edge of the serial clock (or the rising edge of the SCKx pin input/output).

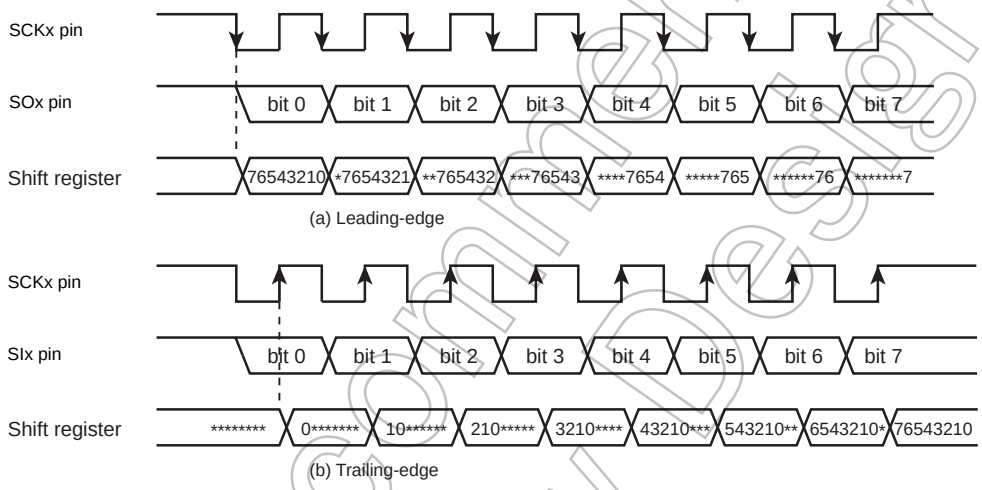


Figure 18-17 Shift Edge

18.8.2 Transfer Modes

The transmit mode, the receive mode or the transmit/receive mode can be selected by programming SBIxCR1<SIOM[1:0]>.

18.8.2.1 8-bit transmit mode

Set the control register to the transmit mode and write the transmit data to SBIxDBR.

After writing the transmit data, writing "1" to SBIxCR1<SIOS> starts the transmission. The transmit data is moved from SBIxDBR to a shift register and output to the SO pin, with the least-significant bit (LSB) first, in synchronization with the serial clock. Once the transmit data is transferred to the shift register, SBIxDBR becomes empty, and the INTSBIx (buffer-empty) interrupt is generated, requesting the next transmit data.

In the internal clock mode, the serial clock will be stopped and automatically enter the wait state, if next data is not loaded after the 8-bit data has been fully transmitted. The wait state will be cleared when SBIxDBR is loaded with the next transmit data.

In the external clock mode, SBIxDBR must be loaded with data before the next data shift operation is started. Therefore, the data transfer rate varies depending on the maximum latency between when the interrupt request is generated and when SBIxDBR is loaded with data in the interrupt service program.

At the beginning of transmission, the same value as in the last bit of the previously transmitted data is output in a period from setting SBIxSR<SIOF> to "1" to the falling edge of SCK.

Transmission can be terminated by clearing <SIOS> to "0" or setting <SIOINH> to "1" in the INTSBIx interrupt service program. If <SIOS> is cleared, remaining data is output before transmission ends. The program checks SBIxSR<SIOF> to determine whether transmission has come to an end. <SIOF> is cleared to "0" at the end of transmission. If <SIOINH> is set to "1", the transmission is aborted immediately and <SIOF> is cleared to "0".

When in the external clock mode, <SIOS> must be cleared to "0" before next data shifting. If <SIOS> does not be cleared to "0" before next data shifting, SBI output dummy data and stopped.

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	0	0	0	X	X	X	Selects the transmit mode.
SBIxDBR	←	X	X	X	X	X	X	X	X	Writes the transmit data.
SBIxCR1	←	1	0	0	0	0	X	X	X	Starts transmission.

INTSBIx interrupt

SBIxDBR	←	X	X	X	X	X	X	X	X	Writes the transmit data.

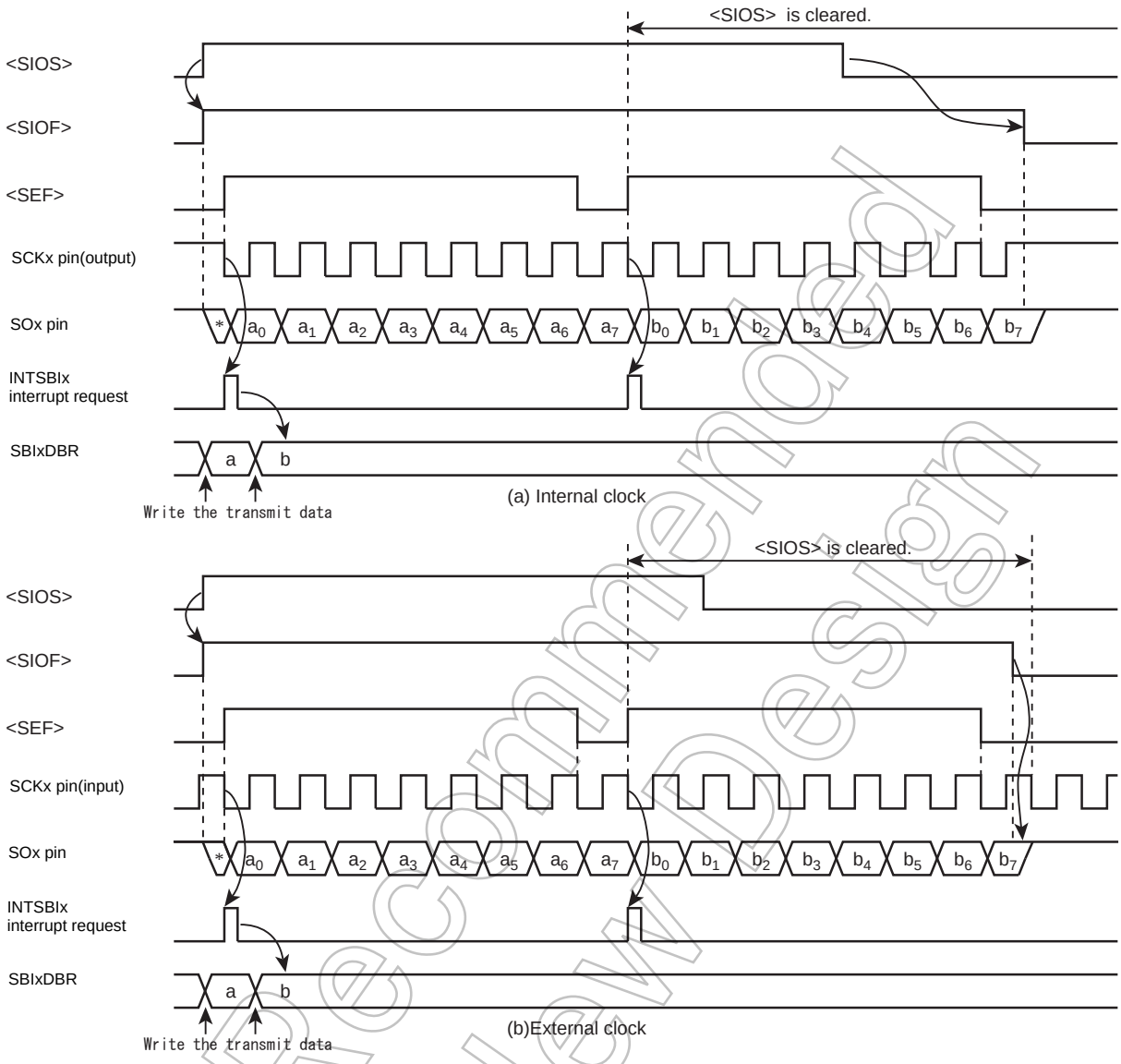


Figure 18-18 Transmit Mode

Example: Example of programming (external clock) to terminate transmission by <SIO>

```
                    7  6  5  4  3  2  1  0
if SBIxSR<SIOF> ≠ 0                          Recognizes the completion of the transmission.
Then
if SCK ≠ 1                                   Recognizes "1" is set to the SCK pin by monitoring the
                                             port.
Then
SBIxCR1        ←    0  0  0  0  0  1  1  1   Completes the transmission by setting <SIOS> = 0.
```

18.8.2.2 8-bit receive mode

Set the control register to the receive mode. Then writing "1" to SBIxCR1<SIOS> enables reception.Data is taken into the shift register from the SI pin, with the least-significant bit (LSB) first, in synchronization with the serial clock. Once the shift register is loaded with the 8-bit data, it transfers the received data to SBIxDBR and the INTSBIx (buffer-full) interrupt request is generated to request reading the received data. The interrupt service program then reads the received data from SBIxDBR.

In the internal clock mode, the serial clock will be stopped and automatically be in the wait state until the received data is read from SBIxDBR.

In the external clock mode, shift operations are executed in synchronization with the external clock. The maximum data transfer rate varies, depending on the maximum latency between generating the interrupt request and reading the received data

Reception can be terminated by clearing <SIOS> to "0" or setting <SIOINH> to "1" in the INTSBIx interrupt service program. If <SIOS> is cleared, reception continues until all the bits of received data are written to SBIxDBR. The program checks SBIxSR<SIOF> to determine whether reception has come to an end.<SIOF> is cleared to "0" at the end of reception. After confirming the completion of the reception, last received data is read. If <SIOINH> is set to "1", the reception is aborted immediately and <SIOF> is cleared to "0". (The received data becomes invalid, and there is no need to read it out.)

Note: **The contents of SBIxDBR will not be retained after the transfer mode is changed. The ongoing reception must be completed by clearing <SIOS> to "0" and the last received data must be read before the transfer mode is changed.**

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	1	1	0	X	X	X	Selects the receive mode.
SBIxCR1	←	1	0	1	1	0	X	X	X	Starts reception.

INTSBIx interrupt

Reg.	←	SBIxDBR	Reads the received data.

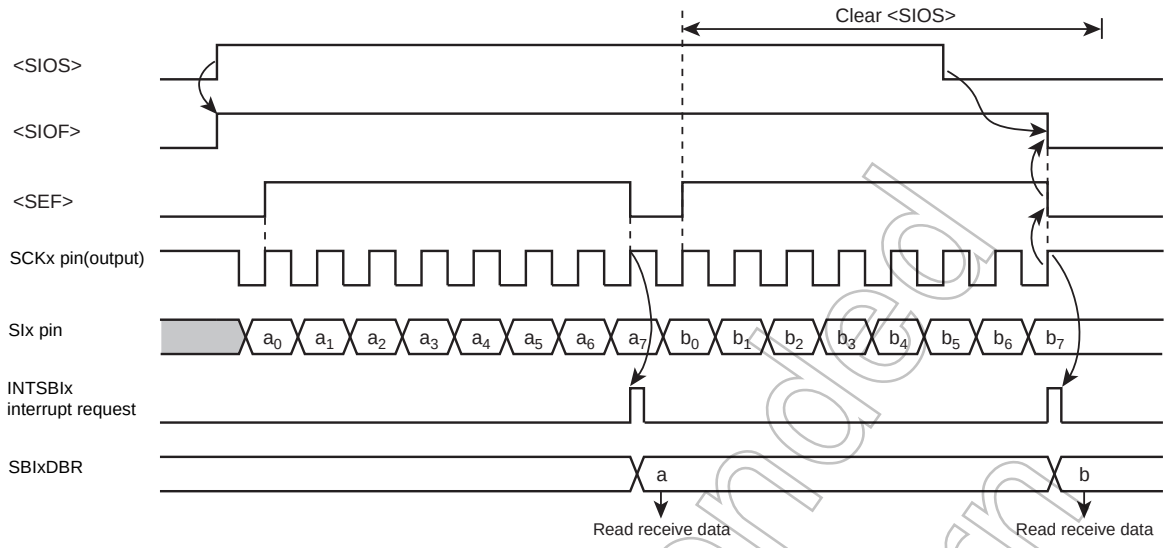


Figure 18-19 Receive Mode (Example: Internal Clock)

18.8.2.3 8-bit transmit/receive mode

Set the control register to the transfer/receive mode. Then writing the transmit data to SBIxDBR and setting SBIxCR1<SIOS> to "1" enables transmission and reception.The transmit data is output through the SOx pin at the falling of the serial clock, and the received data is taken in through the SI pin at the rising of the serial clock, with the least-significant bit (LSB) first. Once the shift register is loaded with the 8-bit data, it transfers the received data to SBIxDBR and the INTSBIx interrupt request is generated.The interrupt service program reads the received data from the data buffer register and writes the next transmit data. Because SBIxDBR is shared between transmit and receive operations, the received data must be read before the next transmit data is written.

In the internal clock operation, the serial clock will be automatically in the wait state until the received data is read and the next transmit data is written.

In the external clock mode, shift operations are executed in synchronization with the external serial clock. Therefore, the received data must be read and the next transmit data must be written before the next shift operation is started.The maximum data transfer rate for the external clock operation varies depending on the maximum latency between when the interrupt request is generated and when the transmit data is written.

At the beginning of transmission, the same value as in the last bit of the previously transmitted data is output in a period from setting <SIOF> to "1" to the falling edge of SCK.

Transmission and reception can be terminated by clearing <SIOS> to "0" or setting SBIxCR1<SIOINH> to "1" in the INTSBIx interrupt service program. If <SIOS> is cleared, transmission and reception continue until the received data is fully transferred to SBIxDBR. The program checks SBIxSR<SIOF> to determine whether transmission and reception have come to an end. <SIOF> is cleared to "0" at the end of transmission and reception.If <SIOINH> is set to "1", the transmission and reception is aborted immediately and <SIOF> is cleared to "0".

Note: **The contents of SBIxDBR will not be retained after the transfer mode is changed. The ongoing transmission and reception must be completed by clearing <SIOS> to "0" and the last received data must be read before the transfer mode is changed.**

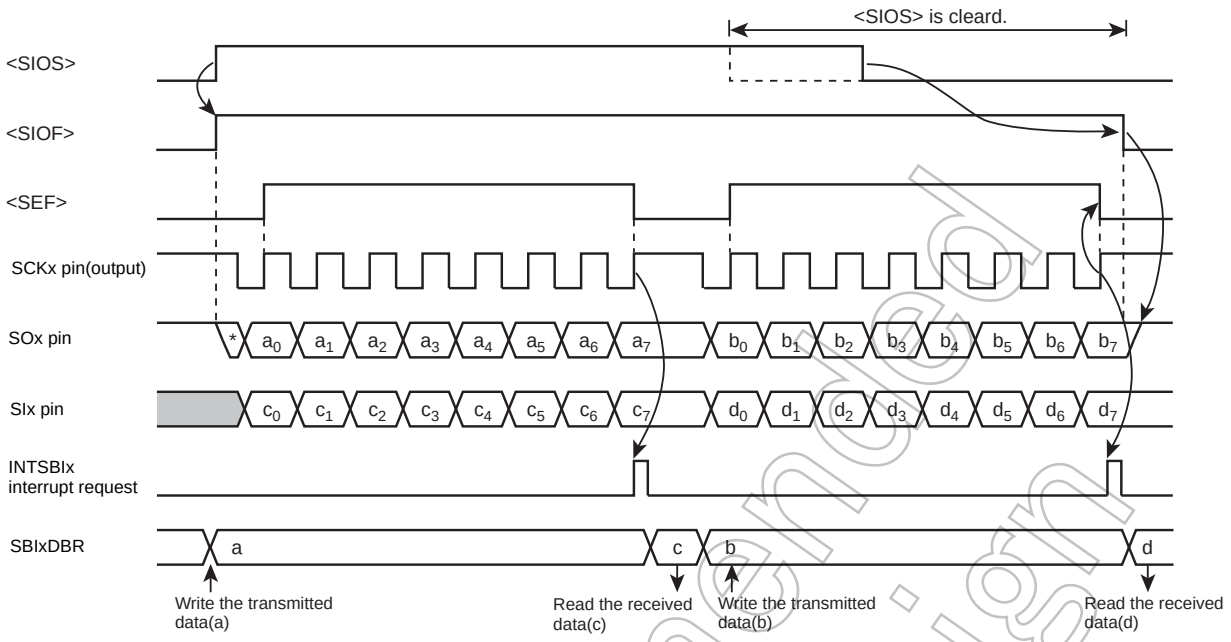


Figure 18-20 Transmit/Receive Mode (Example: Internal Clock)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	1	0	0	X	X	X	Selects the transmit mode.
SBIxDBR	←	X	X	X	X	X	X	X	X	Writes the transmit data.
SBIxCR1	←	1	0	1	0	0	X	X	X	Starts reception/transmission.

INTSBIx interrupt

Reg.	←	SBIxDBR								Reads the received data.
SBIxDBR	←	X	X	X	X	X	X	X	X	Writes the transmit data.

18.8.2.4 Data retention time of the last bit at the end of transmission

Under the condition SBIxCR1<SIOS>= "0", the last bit of the transmitted data retains the data of SCK rising edge as shown below. Transmit mode and transmit/receive mode are the same.

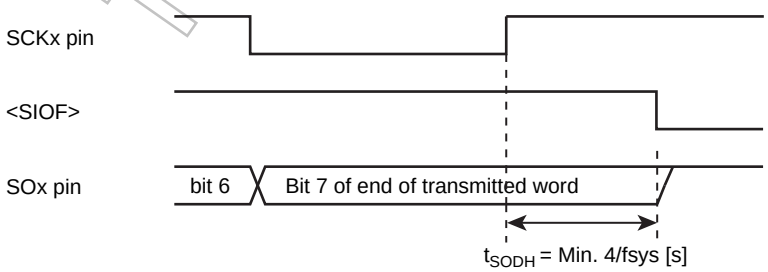


Figure 18-21 Data retention time of the last bit at the end of transmission

19. Analog/Digital Converter (ADC)

19.1 Outline

TMPM366FDXBG/FYXBG/FWXBG contains a 12-bit, sequential-conversion analog/digital converter (ADC) with 12 analog input channnels.

These 12 analog input channels (pins AIN00 through AIN11) are also used as input/output ports.

The 12-bit AD converter has the following features:

- Starting normal AD conversion and highest-priority AD conversion
  - Software activation
  - Activation with the 16-bit timer (TMRB)
  - Hardware activation with an external trigger input ($\overline{\text{ADTRG}}$ pin)
- AD conversion
  - Fixed-channel singlel conversion mode
  - Channel scan single conversion mode
  - Fixed-channel repeat conversion mode
  - Channnel scan repeat conversion mode
- Highest-priority AD conversion
- Normal AD conversion completion interrupt and highest-priority AD conversion completion interrupt
- Normal AD conversion and highest-priority AD conversion have the following status flags.
  - A flag indicating the AD conversion result data is valid, <ADRxRF>, and a flag indicating theAD conversion result data is overwritten, <OVRx>
  - Normal AD conversion completion flag and highest-priority AD conversion completion flag
  - Normal AD conversion busy flag and highest-priority AD conversion busy flag
- AD Monitor Function
  - When the AD monitor function is enabled, an interrupt is generated if any comparison result is matched.
- AD conversion clock can be contorolled from 1/fc to 1/16fc.
- When AD conversion is completed, two types of DMA requests are supported.
- Standby mode is supported.
- Output switching monitor function
  - This is the function that monitors output switching operation of general input-output ports, which is also used as analog input channels (pins AIN00 througe AIN11), during AD conversion. This monitor function is used to suggest the possibility that output switching operation during AD conversion affects conversion accuracy.

19.2 Configuration

Figure 19-1 shows the block diagram of the AD converter.

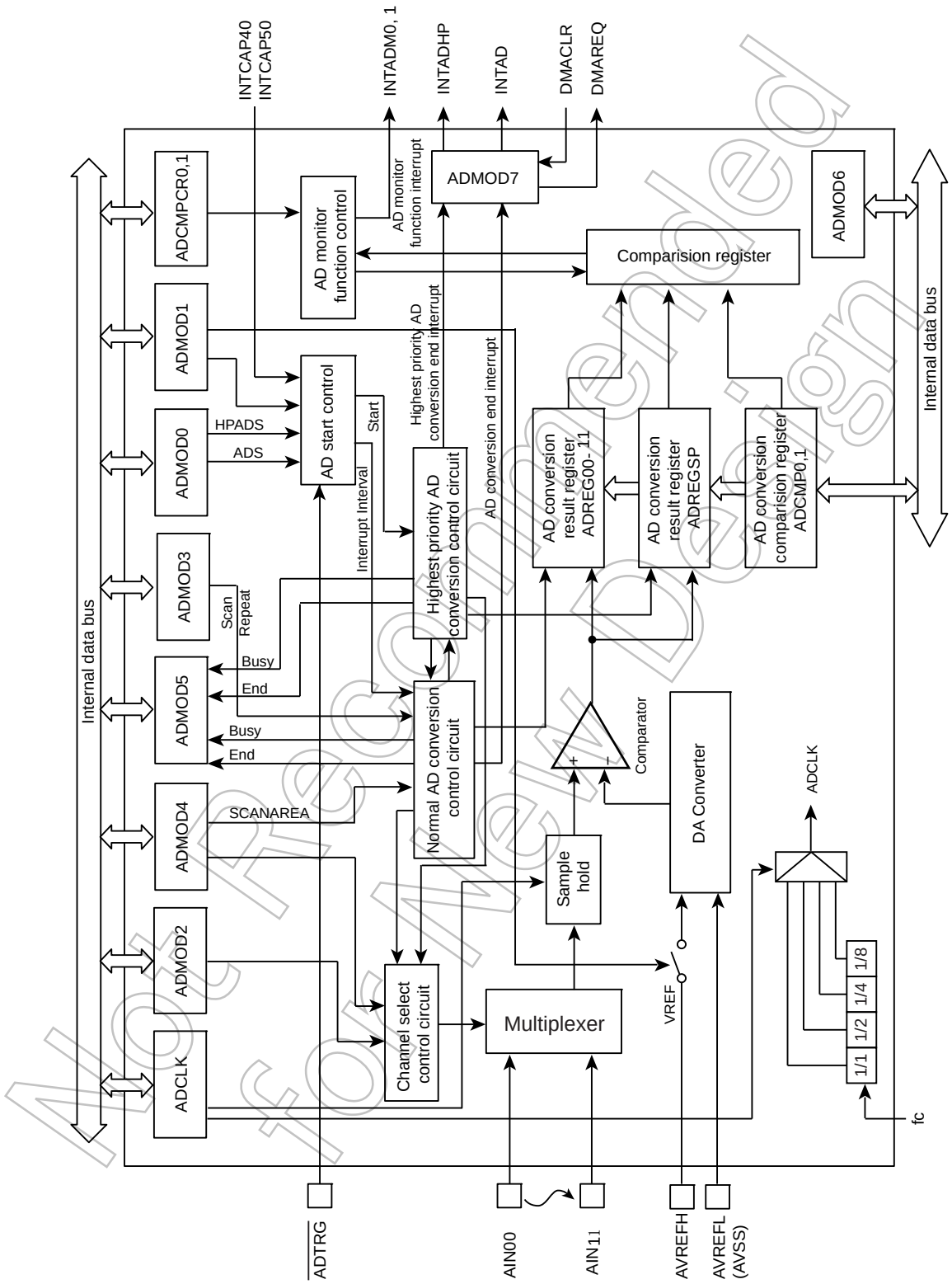


Figure 19-1 AD Converter Block Diagram

19.3 Registers

19.3.1 Register list

The control registers and addresses of the AD converter are as follows.

The AD converter is controlled by the AD mode control registers (ADMOD0 through ADMOD7). The result of AD conversion is stored in 12 AD conversion result registers, ADREG00 through ADREG11. The highest-priority conversion result is stored in the register ADREGSP.

Base Address = 0x4005_0000

Register name		Address(Base+)
Conversion Clock Setting Register	ADCLK	0x0000
Mode Control Register 0	ADMOD0	0x0004
Mode Control Register 1	ADMOD1	0x0008
Mode Control Register 2	ADMOD2	0x000C
Mode Control Register 3	ADMOD3	0x0010
Mode Control Register 4	ADMOD4	0x0014
Mode Control Register 5	ADMOD5	0x0018
Mode Control Register 6	ADMOD6	0x001C
Mode Control Register 7	ADMOD7	0x0020
Monitor Function Control Register 0	ADCMPCR0	0x0024
Monitor Function Control Register 1	ADCMPCR1	0x0028
Conversion Result Comparison Register 0	ADCMP0	0x002C
Conversion Result Comparison Register 1	ADCMP1	0x0030
Conversion Result Register 0	ADREG00	0x0034
Conversion Result Register 1	ADREG01	0x0038
Conversion Result Register 2	ADREG02	0x003C
Conversion Result Register 3	ADREG03	0x0040
Conversion Result Register 4	ADREG04	0x0044
Conversion Result Register 5	ADREG05	0x0048
Conversion Result Register 6	ADREG06	0x004C
Conversion Result Register 7	ADREG07	0x0050
Conversion Result Register 8	ADREG08	0x0054
Conversion Result Register 9	ADREG09	0x0058
Conversion Result Register 10	ADREG10	0x005C
Conversion Result Register 11	ADREG11	0x0060
Reserved	-	0x0064
Reserved	-	0x0068
Reserved	-	0x006C
Reserved	-	0x0070
Conversion Result Register SP	ADREGSP	0x0074
Reserved	-	0x0F00
Reserved	-	0x0F04
Reserved	-	0x0F08

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

19.3.2 ADCLK (Conversion Clock Setting Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7-4	ADSH[3:0]	R/W	Select the AD sample hold time. 0000: 10 × <ADCLK> 0001: 20 × <ADCLK> 0010: 30 × <ADCLK> 0011: 40 × <ADCLK> 0100: 80 × <ADCLK> 0101 to 1111: Reserved
3	−	R	Read as 0.
2-0	ADCLK[2:0]	R/W	Select the AD prescaler clock. 000: fc 001: fc/2 010: fc/4 011: fc/8 100 to 111: Reserved

Note 1: Specify ADCLK in range 4MHz ≤ ADCLK ≤ 40MHz. For example, when fosc = 12MHz and PLL = 8 multiplying, fc comes to 48MHz. In such case, set ADCLK<ADCLK[2:0]> to a value other than "000".

Note 2: Do not change the setting of <ADCLK > except when AD convertsion is suspended and ADMOD1<VREFON>="0".

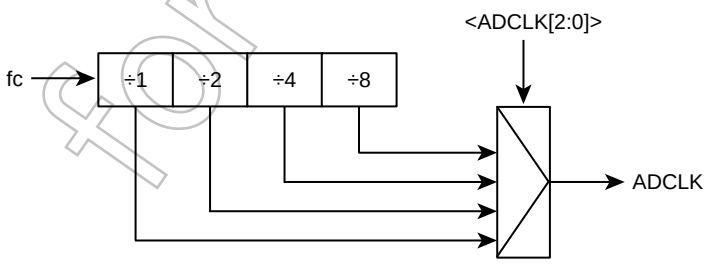


Figure 19-2 AD conversion clock (ADCLK)

A clock count required for conversion is 40 clocks at the minimum.

Examples of sample hold time and conversion time as shown as below.

<ADCLK[2:0]> Setting	<ADSH[3:0]>	Conversion time		
		fc=32MHz	fc=40MHz	fc=48MHz
000 (fc)	conversion clock × 10	1.25 μs	1.00 μs	−
	conversion clock × 20	1.56 μs	1.25 μs	−
	conversion clock × 30	1.88 μs	1.50 μs	−
	conversion clock × 40	2.19 μs	1.75 μs	−
	conversion clock × 80	3.44 μs	2.75 μs	−
001 (fc/2)	conversion clock × 10	2.50 μs	2.00 μs	1.67 μs
	conversion clock × 20	3.13 μs	2.50 μs	2.08 μs
	conversion clock × 30	3.75 μs	3.00 μs	2.50 μs
	conversion clock × 40	4.38 μs	3.50 μs	2.92 μs
	conversion clock × 80	6.88 μs	5.50 μs	4.58 μs
010 (fc/4)	conversion clock × 10	5.00 μs	4.00 μs	3.33 μs
	conversion clock × 20	6.25 μs	5.00 μs	4.17 μs
	conversion clock × 30	7.50 μs	6.00 μs	5.00 μs
	conversion clock × 40	8.75 μs	7.00 μs	5.83 μs
	conversion clock × 80	−	−	9.17 μs
011 (fc/8)	conversion clock × 10	10.0 μs	8.00 μs	6.67 μs
	conversion clock × 20	−	10.0 μs	8.33 μs
	conversion clock × 30	−	−	10.00 μs
	conversion clock × 40	−	−	−
	conversion clock × 80	−	−	−

Note 1: Do not change the setting of the AD conversion clock during AD conversion.

Note 2: Setting the element indicated by "−" in the above table is prohibited. Specify ADCLK setting in the 1μs to 10μs range.

19.3.3 ADMOD0 (Mode Control Register 0)

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

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as 0.
1	HPADS	W	Activate highest-priority AD conversion 0: Don't care 1: Start conversion "0" is always read.
0	ADS	W	Activate normal (software) AD conversion (Note 3) 0: Don't care 1: Start conversion "0" is always read.

Note 1: In use ADC, write "1" to ADMOD1<VREFON> first, and then start AD conversion or external trigger by setting ADMOD0<ADS> or <HPADS>.

Note 2: When both highest-priority AD conversion <HPADS> and normal AD conversion (software) are enabled and they are selected as ADTRG (external trigger input), highest-priority AD conversion is activated as a priority and normal AD conversion is not activated.

19.3.4 ADMOD1 (Mode Control Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	VREFON	I2AD	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 0.
7	VREFON	R/W	VREF application control (Note1 and Note2) 0: OFF 1: ON
6	I2AD	R/W	Specify operation mode in IDLE mode 0: Stop 1: Operation
5	RCUT	R/W	Control AVREFH-AVREFL current 0: Apply the current only in conversion. 1: Apply the current at any time except in RESET
4	−	R	Read as 0.
3	HPADHWS	R/W	Select hardware activation source of highest-priority AD conversion 0: External trigger 1: Interrupt of INTCAP40
2	HPADHWE	R/W	Activate highest-priority AD conversion triggered by hardware factors 0: Disable 1: Enable
1	ADHWS	R/W	Select hardware activation source of normal AD conversion (Note 3) 0: External trigger 1: Interrupt of INTCAP50
0	ADHWE	R/W	Activate normal AD conversion triggered by hardware factors 0: Disable 1: Enable

Note 1: In use AD conversion, write "1" to the ADMOD<VREFON> bit, wait for 3μs during which time the internal reference voltage should stabilize, and then start AD conversion or external trigger by setting ADMOD0<ADS> or <HPADS> to "1".

Note 2: Set <VREFON> to "0" to go into standby mode upon completion of AD conversion.

Note 3: The external trigger cannot be used for H/W activation of normal AD conversion when it is used for H/W activation of highest-priority AD conversion.

Note 4: If it is necessary to reduce a power current with IDLE or STOP mode and if either case shown below is applicable, you must first suspend the AD converter and then execute the instruction to put into standby mode.

1. In the case of putting into IDLE mode with ADMOD1 <I2AD> = "0".
2. In the case of putting into STOP mode.

19.3.5 ADMOD2 (Mode Control Register 2)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7-4	HPADCH[3:0]	R/W	Select analog input channels in highest-priority AD conversion. (See Table 19-1.) (prohibit "1100 to 1111")
3-0	ADCH[3:0]	R/W	Select analog input channels in normal AD conversion. (See Table 19-1.) (prohibit "1100 to 1111")

Table 19-1 Selection of input channels in normal AD conversion or highest-priority AD conversion

<HPADCH[3:0]>	Analog input channels in highest-priority AD conversion	<ADCH[3:0]>	Analog input channels in normal AD conversion
0000	AIN00	0000	AIN00
0001	AIN01	0001	AIN01
0010	AIN02	0010	AIN02
0011	AIN03	0011	AIN03
0100	AIN04	0100	AIN04
0101	AIN05	0101	AIN05
0110	AIN06	0110	AIN06
0111	AIN07	0111	AIN07
1000	AIN08	1000	AIN08
1001	AIN09	1001	AIN09
1010	AIN10	1010	AIN10
1011	AIN11	1011	AIN11

19.3.6 ADMOD3 (Mode Control Register 3)

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

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as 0.
6-4	ITM[2:0]	R/W	Specify interrupt in fixed channel repeat conversion mode. See Table 19-2.
3-2	−	R	Read as 0.
1	REPEAT	R/W	Specify repeat mode 0 : Single conversion mode 1 : Repeat conversion mode
0	SCAN	R/W	Specify scan mode 0 : Fixed channel mode 1 : Channel scan mode

Table 19-2 AD conversion interrupt specification in fixed channel repeat conversion mode

<ITM[2:0]>	Fixed channel repeat conversion mode <SCAN> = "0", <REPEAT> = "1"
000	Generate in interrupt once every single conversion.
001	Generate interrupt once every 2 conversions.
010	Generate interrupt once every 3 conversions.
011	Generate interrupt once every 4 conversions.
100	Generate interrupt once every 5 conversions.
101	Generate interrupt once every 6 conversions.
110	Generate interrupt once every 7 conversions.
111	Generate interrupt once every 8 conversions.

Note 1: <ITM[2:0]> is valid only when it's specified in the fixed channel repeat mode, <REPEAT> ="1" and <SCAN> = "0".

Note 2: When repeat conversion is aborted during repeat conversion (in <REPEAT>=1, fixed channel mode or channel scan mode), <REPEAT> is "0" cleared. In such case, do not change the setting except <REPEAT> bit.

19.3.7 ADMOD4 (Mode Control Register 4)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7-4	SCANAREA [3:0]	R/W	Range of channel scan (prohibit "1100 to 1111")
3-0	SCANSTA[3:0]	R/W	Select the start channel to be scanned. (prohibit "1100" to "1111")

To specify channel scan single mode, set ADMOD3<SCAN> to "1" and <REPEAT> to "0". And, to specify channel scan repeat mode, set ADMOD3<SCAN> to "1" and <REPEAT> to "1".

At first, select the start channel to be scanned. Then select the number of channels to be scanned, starting on the specified start channel.

For example, when ADMOD4<SCANSTA> is set to "0001"(AIN01) and <SCANAREA> is set to "0010" (3ch scan), three channels from AIN01 to AIN03 are scanned.

The following shows the range of assignable value to <SCANAREA> in relation to setting of <SCANSTA>.

Table 19-3 The range of assignable channel scan value

<SCANSTA[3:0]>	The start channel to be scanned	<SCANAREA[3:0]>	The range of assignable channel scan value
0000	AIN00	0000 to 1011	(1ch to 12ch)
0001	AIN01	0000 to 1010	(1ch to 11ch)
0010	AIN02	0000 to 1001	(1ch to 10ch)
0011	AIN03	0000 to 1000	(1ch to 9ch)
0100	AIN04	0000 to 0111	(1ch to 8ch)
0101	AIN05	0000 to 0110	(1ch to 7ch)
0110	AIN06	0000 to 0101	(1ch to 6ch)
0111	AIN07	0000 to 0100	(1ch to 5ch)
1000	AIN08	0000 to 0011	(1ch to 4ch)
1001	AIN09	0000 to 0010	(1ch to 3ch)
1010	AIN10	0000 to 0001	(1ch to 2ch)
1011	AIN11	0000	(1ch)

Note: In case of a setting other than listed above, AD conversion is not activated even if ADMOD0 register is set to activate AD conversion.

19.3.8 ADMOD5 (Mode Control Register 5)

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

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as 0.
3	HPEOCF	R	Highest-priority AD conversion completion flag (Note1) 0: Before or during conversion 1: Completion
2	HPADBF	R	Highest-priority AD conversion BUSY flag 0: During conversion halts 1: During conversion
1	EOCF	R	Normal AD conversion completion flag (Note1) 0: Before or during conversion 1: Completion
0	ADBF	R	Normal AD conversion BUSY flag 0: During conversion halts 1: During conversion

Note 1: This flag is "0" cleared by reading the ADMOD5 register.

Note 2: If it is necessary to reduce a power current with IDLE or STOP mode and if either case shown below is applicable, you must first stop the AD converter and then execute the instruction to put into standby mode.

1. In the case of putting into IDLE mode with ADMOD1<I2AD> = "0".
2. In the case of putting into STOP1/STOP2 mode.

19.3.9 ADMOD6 (Mode Control 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 0.
1-0	ADRST[1:0]	W	Overwriting 10 with 01 allows ADC to be software reset.

Note 1: When DMA transmission is executed by using AD conversion completion interrupt, software reset ADMOD6 <ADRST> first, and then operate DMAC(DMA request standby state) and configure (activate) the ADC.

Note 2: Initialization takes 3μs in case of the software reset.

19.3.10 ADMOD7 (Mode Control Register7)

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

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as 0.
3-2	−	R/W	Always write "0".
1	INTADHPDMA	R/W	Specify Highest-priority AD conversion DMA activation factor. 0 : Disable 1 : Enable
0	INTADDMA	RW	Specify normal AD conversion DMA activation factor. 0 : Disable 1 : Enable

19.3.11 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	-	-	ADBIG0	AINS0			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-12	−	R	Read as "0".
11-8	CMPCNT0[3:0]	R/W	Number of comparison until the judgment is confirmed. An interrupt is generated when the number of the counts is achieved. 0000 : 1 time count 0110 : 7 times count 1100 : 13 times count 0001 : 2 times count 0111 : 8 times count 1101 : 14 times count 0010 : 3 times count 1000 : 9 times count 1110 : 15 times count 0011 : 4 times count 1001 : 10 times count 1111 : 16 times count 0100 : 5 times count 1010 : 11 times count 0101 : 6 times count 1011 : 12 times count
7	CMP0EN	R/W	AD monitor function 0 0: Disable 1: Enable Setting the condition <CMP0EN>="0" (disabled) clears the number of counts.
6-5	−	R	Read as "0".
4	ADBIG0	R/W	Set the determination for small and large. 0: Larger than comparison register 1: Smaller than comparison register Every time when the AD conversion set to <AINS0[3:0]> is completed, compare the size of conversion results. If the result matches the settings of <ADBIG0>, the counter is incremented.
3-0	AINS0[3:0]	R/W	Set analog inputs as a target for comparison. 0000 : AIN00 0101 : AIN05 1010 : AIN10 0001 : AIN01 0110 : AIN06 1011 : AIN11 0010 : AIN02 0111 : AIN07 0011 : AIN03 1000 : AIN08 0100 : AIN04 1001 : AIN09 Prohibit "1100" to "1111".

Note: AD monitor function is used the fixed repeat mode and the scan repeat mode.

19.3.12 ADCMPCR1 (AD 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	-	-	ADBIG1	AINS1			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-12	−	R	Read as 0.
11-8	CMPCNT1[3:0]	R/W	Number of comparison until the judgment is confirmed. An interrupt is generated when the number of the counts is achieved. 0000 : 1 time count 0110 : 7 times count 1100 : 13 times count 0001 : 2 times count 0111 : 8 times count 1101 : 14 times count 0010 : 3 times count 1000 : 9 times count 1110 : 15 times count 0011 : 4 times count 1001 : 10 times count 1111 : 16 times count 0100 : 5 times count 1010 : 11 times count 0101 : 6 times count 1011 : 12 times count
7	CMP1EN	R/W	AD monitor function 1 0: Disable 1: Enable Setting the condition <CMP1EN>="0" (disabled) clears the number of counts.
6-5	−	R	Read as 0.
4	ADBIG1	R/W	Set the determination for small and large. 0: Larger than comparison register 1: Smaller than comparison register Every time when the AD conversion set to <AINS1[3:0]> is completed, compare the size of conversion results. If the result matches the settings of <ADBIG1>, the counter is incremented.
3-0	AINS1[3:0]	R/W	Set analog inputs as a target for comparison. 0000 : AIN00 0101 : AIN05 1010 : AIN10 0001 : AIN01 0110 : AIN06 1011 : AIN11 0010 : AIN02 0111 : AIN07 0011 : AIN03 1000 : AIN08 0100 : AIN04 1001 : AIN09 Prohibit "1100" to "1111".

Note: AD monitor function is used the fixed repeat mode and the scan repeat mode.

19.3.13 ADCMP0 (AD Conversion Result Comparison Register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	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 0.
11-0	AD0CMP[11:0]	W	Sets a value to be compared with the value of the conversion result register

Note: To write values into this register, the AD monitor function 0 must be disabled (ADCMPCR0<CMP0EN> ="0").

19.3.14 ADCMP1 (AD Conversion Result Comparison Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	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 0.
11-0	AD1CMP[11:0]	W	Sets a value to be compared with the value of the conversion result register

Note: To write values into this register, the AD monitor function 1 must be disabled (ADCMPCR1<CMP1EN> ="0").

19.3.15 ADREG00 to ADREG11 (Normal Conversion Result Register 00 to 11)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	ADPOSWF	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
31-15	−	R	Read as 0.
14	ADPOSWF	R	The output switching flag of AIN port. 0: Without switching 1: With switching When PxDATA register of general input-output port which is also used as AIN changes during AD conversion, the port output switching flag, <ADPOSWF>, is set to "1". In this case, when PxCR register corresponding to the changed bit is "1", there is a possibility that the output switching during AD conversion affects conversion accuracy. This bit is "0" cleared when registers, ADREG00 through ADREG11, are read.
13	ADOVRF	R	Overrun flag 0: Not generated. 1: Generated. If the conversion result is overwritten before reading <ADR>, this bit is set to "1". This bit is "0" cleared when registers, ADREG00 through ADREG11, are read.
12	ADRF	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 bit is "0" cleared when the conversion result of register, ADREG00 through ADREG11, are read.
11-0	ADR[11:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to Table 19-5, chapter 19.4.5.7.

Note: Do not do the output switching during AD conversion, when other analog / input-output ports are used as output port.

19.3.16 ADREGSP (Highest-priority Conversion Result 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	-	ADPO SWFSP	ADOVRFSP	ADRFSP	ADRSP			
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ADRSP							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-15	−	R	Read as 0.
14	ADPOSWFSP	R	The output switching flag of AIN port. 0: Without switching 1: With switching When PxDATA register of general input-output port which is also used as AIN changes during AD conversion, the port output switching flag, <ADPOSWFSP>, is set to "1". In this case, when PxCR register corresponding to the changed bit is "1", there is a possibility that the output switching during AD conversion affects conversion accuracy. This bit is "0" cleared when registers, ADREGx is read.
13	ADOVRFSP	R	Overrun flag 0: Not generated 1: Generated If the highest-priority AD conversion result is overwritten before reading <ADRSP>, "1" is set. This bit is "0" cleared when ADREGSP register is read.
12	ADRFSP	R	Highest-priority AD conversion result storage flag 0: Conversion result is not stored. 1: Conversion result is stored. If the highest-priority conversion result is stored, this bit is set to "1". This bit is "0" cleared when ADREGSP conversion result is read.
11-0	ADRSP[11:0]	R	Highest-priority AD conversion result Highest-priority conversion result is stored.

Note: Do not do the output switching during AD conversion, when other analog / input-output ports are used as output port.

19.4 Description of Operations

19.4.1 Analog Reference Voltage

The "High" level of the analog reference voltage shall be applied to the AVRFEH pin, and the "Low" shall be applied to the AVREFL pin.

To start AD conversion, make sure that you first write "1" to the <VREFON> bit, wait for 3 μs during which time the internal reference voltage should stabilize, and then write "1" to the ADMOD0<ADS> bit.

19.4.2 AD Conversion Mode

Two types of AD conversion are supported: normal AD conversion and highest-priority AD conversion.

19.4.2.1 Normal AD Conversion

For normal AD conversion, the following four operation modes are supported and the operation mode is selected with the 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 selected by ADMOD2 <ADCH>. After AD conversion is completed, ADMOD5<EOCF> is set to "1", ADMOD5<ADBF> is cleared to "0", and the AD conversion completion interrupt request (INTAD) is generated. <EOCF> is cleared to "0" upon read.

(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 is completed, ADMOD5<EOCF> is set to "1", ADMOD5<ADBF> is cleared to "0", and the conversion completion interrupt request (INTAD) is generated. <EOCF> is cleared to "0" upon read.

(3) Fixed channel repeat conversion mode

If ADMOD3<REPEAT, SCAN> is set to "10", AD conversion is performed in fixed channel repeat conversation mode.

In this mode, AD conversion is performed repeatedly for one channel selected by ADMOD2 <ADCH>. After AD conversion is completed, ADMOD5<EOCF> is set to "1". ADMOD5<ADBF> is not cleared to "0". It remains at "1". The timing with which the conversion completion interrupt re-

quest (INTAD) is generated can be selected by setting ADMOD3<ITM> to an appropriate setting. <EOCF> is set with the same timing as this interrupt INTAD is generated. <EOCF> is cleared to "0" upon read.

(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 selected by ADMOD4 <SCANAREA> from the start channel selected by ADMOD4 <SCANSTA>. Each time one AD scan conversion is completed, ADMOD5 <EOCF> is set to "1", and the conversion completion interrupt request (INTAD) is generated. ADMOD5 <ADBF> is not cleared to "0" and remains at "1". <EOCF> is cleared to "0" upon read.

19.4.2.2 Highest-priority AD conversion

By interrupting ongoing normal AD conversion, highest-priority AD conversion can be performed.

The fixed-channel single conversion is automatically selected, irrespective of the ADMOD3 <REPEAT, SCAN> setting. 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 highest-priority AD conversion completion interrupt (INTADHP) is generated, and ADMOD5 <HPEOCF> showing the completion of AD conversion is set to "1". <HPADBF> returns to "0". <HOEOCF> flag is cleared to "0" upon read.

Highest-priority AD conversion activated while highest-priority AD conversion is under way is ignored.

19.4.3 AD Monitor Function

This is a function for setting the channel fixed repeat mode and the scan repeat mode.

Setting "1" to both ADCMPCR0<CMP0EN> and ADCMPCR1<CMP1EN> enables the AD monitor function. The monitor function can also be enabled for both registers at the same time.

Here is an example, taking the ADCMPCR0.

Configure the following settings: analog input as a target for comparison to the <AINS0[3:0]> of the ADCMPCR0 register, large/small determination to the <ADBIG0> and the number of counts in determination to the <CMPCNT0[3:0]>.

Once AD conversion starts, every time when one single conversion completes, large/small determination is performed. If the result of the conversion matches the settings stored in the <ADBIG0>, increment the judgment counter.

AD monitor function interrupts (INTADM0) are generated when conditions set to the <ADBIG0> pile up and reach the counting number set to the <CMPCNT0[3:0]>. The counter values are held even the condition is different from that is set to the <ADBIG0>. If values of the conversion result storage register set to the ADCMPCR0 are the same as the values of a register as a comparison target, the count is not incremented. AD monitor function interrupt (INTADM0) is not generated.

This comparison is performed every time when a result is stored to the conversion result storage register, and an interrupt (INTADM0) occurs when a condition matches. Since the storage register used as AD monitor function is not usually read using software, registers correspond to overrun flags, from ADREG00 to 14<ADOVRF>, and conversion result storage flags, from ADREG00 to 14<ADRF>, are always set to "1". Thus, do not use those flags of the conversion result storage register when you use the AD monitor function.

Example : Set AIN00 input as fixed channel repeat conversion. Compare values of the AD conversion result comparison register (0x0888).

- ADMOD3=0x0002： fixed channel repeat conversion Note: AD conversion completion interrupt (INTAD) is disabled.
- ADCMPCR0 =0x02A0： target channel for comparison: AIN00, large/small determination: larger than the comparison register, AD monitor function: enabled, counting number of large/small determination: three counts.
- ADCMP0=0x0888: AD conversion result comparison register (comparison value 0x0888)

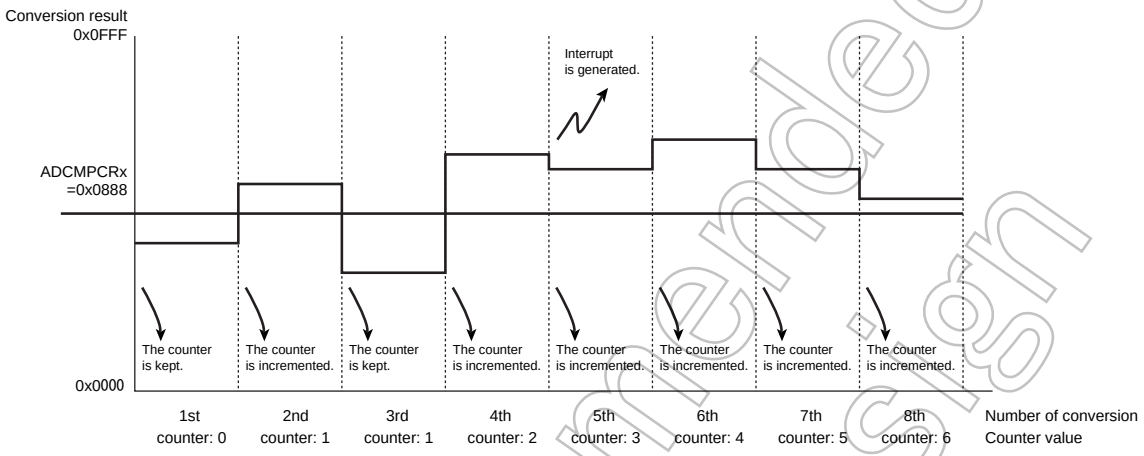


Figure 19-3 AD monitor function (Fixed channel repeat)

19.4.4 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 AIN00 through AIN11 by setting ADMOD2 <ADCH> to an appropriate setting

   - If the analog input channel is used in a scan state (ADMOD3<SCAN> = "1")

     The channel to be started can be specified by setting ADMOD4 <SCANSTA>. And, the number of channels to be scanned can be specified by setting ADMOD4 <SCANAREA>.

2. Highest-priority AD conversion mode

   One channel is selected from analog input pins from AIN00 through AIN11 by setting ADMOD2<HPADCH> to an appropriate setting. If highest-priority AD conversion has been activated during normal AD conversion, ongoing normal AD conversion is suspended, and restarts normal AD conversion after highest-priority AD conversion is completed.

19.4.5 AD Conversion Details

19.4.5.1 Starting AD Conversion

Two types of A/D conversion are supported: normal AD conversion and top-priority AD conversion. Normal AD conversion is activated by setting ADMOD0<ADS> to "1". Highest-priority AD conversion is activated by setting ADMOD0<HPADS> to "1".

Four operation modes are made available to normal AD conversion. In performing normal AD conversion, one of these operation modes must be selected by setting ADMOD3 <REPEAT, SCAN> to an appropriate setting. For highest-priority AD conversion, only one operation mode can be used: fixed channel single conversion mode.

Normal AD conversion can be activated using the H/W activation source selected by ADMOD1<ADHWS>, and highest-priority AD conversion can be activated using the HW activation source selected by ADMOD1<HPADHWS>. If bits of <ADHWS> and <HPADHWS> are "0", normal and highest-priority AD conversions are activated in response to the input of a falling edge through the ADTRG pin. If these bits are "1", normal AD conversion is activated in response to INTCAP50 generated by the 16-bit timer channel 5, and highest-priority AD conversion is activated in response to INTCAP40 generated by the 16-bit timer channel 4.

To permit H/W activation, set ADMOD1 <ADHWE> to "1" for normal AD conversion and set ADMOD1<HPADHWE> to "1" for highest-priority AD conversion.

Software activation is still valid even after H/W activation has been permitted.

Note: When an external trigger is used for the HW activation source of a highest-priority AD conversion, an external trigger cannot be set for activating normal AD conversion H/W start.

19.4.5.2 AD Conversion

When normal AD conversion starts, the AD conversion Busy flag (ADMOD5 <ADBF>) showing that AD conversion is under way is set to "1".

When highest-priority AD conversion starts, the highest-priority AD conversion Busy flag (ADMOD5 <HPADBF>) showing that AD conversion is under way is set to "1".

At that time, the value of the Busy flag ADMOD5<ADBF> for normal AD conversion before the start of highest-priority AD conversion is retained.

The value of the conversion completion flag ADMOD5 <EOCF> for normal AD conversion before the start of highest-priority AD conversion is retained.

Note: Normal AD conversion must not be activated when highest-priority AD conversion is under way. If activated when highest-priority AD conversion is under way, the highest-priority AD conversion completion flag cannot be set, and the flag for previous normal A/D conversion cannot be cleared.

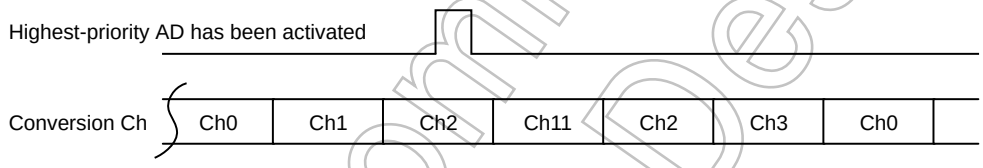
19.4.5.3 Highest-priority AD conversion requests during normal AD conversion

If highest-priority AD conversion has been activated during normal AD conversion, ongoing normal AD conversion is suspended, and restarts normal AD conversion after highest-priority AD conversion is completed.

If ADMOD0<HPADS> is set to "1" during normal AD conversion, ongoing normal AD conversion is suspended, and the highest-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 highest-priority AD conversion is stored in the storage register ADREGSP, normal AD conversion is resumed.

If H/W activation of highest-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 highest-priority AD conversion (fixed-channel single conversion) starts for a channel designated by ADMOD2<HPADCH>. After the result of this highest-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 AIN00 through AIN03 and if <HPADS> is set to "1" during AIN02 conversion, AIN02 conversion is suspended, and conversion is performed for a channel designated by <HPADCH> (AIN11 in the case shown below). After the result of conversion is stored in ADREGSP, channel repeat conversion is resumed, starting from AIN02.



19.4.5.4 Stopping Repeat Conversion Mode

To stop the AD conversion operation in the repeat conversion mode (fixed-channel repeat conversion mode or channel scan 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".

19.4.5.5 Reactivating normal AD conversion

If ADMOD0 <ADS> is set to "1" during normal AD conversion, normal AD conversion is reactivated. Ongoing normal AD conversion is suspended at the time that it is reactivated. At that time, the normal AD conversion Busy flag ADMOD5 <ADBF>, the normal AD conversion completion flag ADMOD5 <EOCF> and the storage result flag ADREGm <ADOVRF>, <ADRF> are cleared to "0". (m=00-11)

If H/W activation of normal 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. Ongoing normal AD conversion is suspended at the time that it is reactivated. At that time, the normal AD conversion Busy flag ADMOD5 <ADBF>, the normal AD conversion completion flag ADMOD5 <EOCF> and the storage result flag ADREGm <ADOVRF>, <ADRF> are cleared to "0". (m=00-11)

19.4.5.6 Conversion completion

(1) Normal AD conversion completion

When normal AD conversion is completed, the AD conversion completion interrupt (INTAD) is generated. The result of AD conversion is stored in the storage register, and two registers change: the register ADMOD5<EOCF> which indicates the completion of AD conversion and the register ADMOD5<ADBF>. The timing that interrupt request is generated and the timing that conversion result register <EOCF> <ADBF> changes vary according to a mode selected.

In mode other than fixed-channel repeat conversion mode, conversion results are stored in AD conversion result registers (ADREG00 through ADREG11) corresponding to a channel.

In fixed-channel repeat conversion mode, the conversion results are sequentially stored in storage registers ADREG00 through ADREG11. However, if interrupt setting on <ITM> is set to be generated each time one AD conversion is completed, the conversion result is stored only in ADREG00. If interrupt setting on <ITM> is set to be generated each time 8 AD conversions are completed, the conversion results are sequentially stored in ADREG00 through ADREG07.

Interrupt requests, flag changes and conversion result registers in each mode are as shown below.

- Fixed-channel single conversion mode

  After AD conversion is completed, ADMOD5 <EOCF> is set to "1", ADMOD5 <ADBF> is cleared to "0", and the interrupt request INTAD is generated.

  Conversion results are stored a conversion result register correspond to a channel.

- Channel scan single conversion mode

  After the channel scan conversion is completed, ADMOD5 <EOCF> is set to "1", ADMOD5 <ADBF> is cleared to "0", and the interrupt request INTAD is generated.

  Conversion results are stored a conversion result register correspond to a channel.

- Fixed-channel repeat conversion mode

  ADMOD5 <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 ADMOD3<ITM> to an appropriate setting. ADMOD5 <EOCF> is set with the same timing as this interrupt INTAD is generated.

  a. One conversion

     With ADMOD2 <ADCH[3:0]> set to "0000" (AIN00) and ADMOD3 <ITM[2:0]> set to "000", an interrupt request is generated each time one AD conversion is completed. In this case, the conversion results are always stored in the storage register ADREG00. After the conversion result is stored, <EOCF> is set to "1".

  b. 8 conversions

With ADMOD2 <ADCH[3:0]> set to "1011" (AIN11) and ADMOD3 <ITM[2:0]> set to "111", an interrupt request is generated each time 8 AD conversions are completed. In this case, the conversion results are sequentially stored in the storage register ADREG00 through ADREG07. After the conversion result is stored in ADREG07, <EOCF> is set to "1", and the storage of subsequent conversion results starts from ADREG00.

- Channel scan repeat conversion mode

  Each time one AD conversion is completed, ADMOD5<EOCF> is set to "1" and an interrupt request INTAD is generated. ADMOD5<ADBF> is not cleared to "0". It remains at "1".

  If ADMOD4 <SCANSTA[3:0]> is set to "0001" (AIN01) and ADMOD4 <SCANAREA [7:4]> is set to "1110" (11Ch scan), each time one AD conversion is completed, ADMOD5 <EOCF> is set to "1" and an interrupt request INTAD is generated. ADMOD5 <ADBF> is not cleared to "0" and remains at "1".

  AD conversion results are stored in a AD conversion result register corresponding to a channel.

(2) Highest-priority AD conversion completion

After the highest-priority AD conversion is completed, the highest-priority AD conversion completion interrupt (INTADHP) is generated, and ADMOD5<HPEOCF> which indicates the completion of highest-priority AD conversion is set to "1".

AD conversion results are stored in the AD 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> is set to "1". To confirm the completion of AD conversion and to obtain the results, poll this bit.

AD conversion result storage register must be read by word access. If <ADOVRF> = “0”, <ADRF> = “1” and <ADPOSWF> = “0”, a correct conversion result has been obtained.

(4) DMA request

After the normal AD conversion completion interrupt (INTAD) or the highest-priority AD conversion completion interrupt (INTADHP) is generated , DMA request is issued. DMA request after any interrupt is generated can be set to "disable" or "enable" by setting ADMOD7 register to an appropriate setting. A DMA request is issued in 2 system clocks (fsys) after AD conversion completion interrupt (INTAD or INTADHP) is generated.

19.4.5.7 Interrupt generation timings and AD conversion result storage register

Table 19-4 shows a relation in the following three items: AD conversion modes, interrupt generation timings and flag operations. Table 19-5 shows a relation between analog channel inputs and AD conversion result registers.

Table 19-4 Relations in conversion modes, interrupt generation timings and flag operations

Conversion mode		ADMOD3			Interrupt generation timing	ADMOD5		
		<REPEAT>	<SCAN>	<ITM[2:0]>		<EOCF>/ <HPEOCF> set timing (See note)	<ADBFN> (After the interrupt is generated)	<ADBFHP> (After the interrupt is generated)
Normal conversion	Fixed-channel single conversion	0	0	–	After generation is completed.	After generation is completed.	0	–
	Fixed-channel repeat conversion	1	0	000	Each time one conversion is completed.	After one conversion is completed.	1	–
				001	Each time 2 conversion is completed.	After 2 conversions are completed.	1	–
				010	Each time 3 conversion is completed.	After 3 conversions are completed.	1	–
				011	Each time 4 conversion is completed.	After 4 conversions are completed.	1	–
				100	Each time 5 conversion is completed.	After 5 conversions are completed.	1	–
				101	Each time 6 conversion is completed.	After 6 conversions are completed.	1	–
				110	Each time 7 conversion is completed.	After 7 conversions are completed.	1	–
				111	Each time 8 conversion is completed.	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	–	After one scan conversion is completed.	After one scan conversion is completed.	1	–
Highest-priority conversion		–	–	–	After generation is completed.	Conversion completion	–	0

Note 1: ADMOD5 <EOCF> and <HPEOCF> are cleared upon read.

Note 2: In repeat mode, ADMOD5 <ADBF> is not cleared to "0" even if any interrupt is generated. To suspend the repeat operation, ADMOD5 <ADBF> is cleared to "0" after ADMOD3 <REPEAT> is written "0" and AD conversion is completed.

Table 19-5 Relations between analog channel inputs and AD conversion result registers

Fixed-channel single mode	
Channel	Storage register
AIN00	ADREG00
AIN01	ADREG01
AIN02	ADREG02
AIN03	ADREG03
AIN04	ADREG04
AIN05	ADREG05
AIN06	ADREG06
AIN07	ADREG07
AIN08	ADREG08
AIN09	ADREG09
AIN10	ADREG10
AIN11	ADREG11

Fixed-channel repeat mode		
ADMOD3<ITM[2:0]>		Storage register
000	Interrupt by each time AD/C	ADREG00
001	Interrupt by each time 2 AD/C	ADREG00 to ADREG01
010	Interrupt by each time 3 AD/C	ADREG00 to ADREG02
011	Interrupt by each time 4 AD/C	ADREG00 to ADREG03
100	Interrupt by each time 5 AD/C	ADREG00 to ADREG04
101	Interrupt by each time 6 AD/C	ADREG00 to ADREG05
110	Interrupt by each time 7 AD/C	ADREG00 to ADREG06
111	Interrupt by each time 8 AD/C	ADREG00 to ADREG07

Channel scan single mode / repeat mode		
ADMOD4<SCANSTA> (Starts channel)	ADMOD4<SCANAREA> (Scan channel range)	Storage register
AIN00	12 channels	ADREG00 to ADRE11
AIN01	11 channels	ADREG01 to ADRE11
AIN02	10 channels	ADREG02 to ADRE11
AIN03	9 channels	ADREG03 to ADRE11
AIN04	8 channels	ADREG04 to ADRE11
AIN05	7 channels	ADREG05 to ADRE11
AIN06	6 channels	ADREG06 to ADRE11
AIN07	5 channels	ADREG07 to ADRE11
AIN08	4 channels	ADREG08 to ADRE11
AIN09	3channels	ADREG09 to ADRE11
AIN10	2 channels	ADREG10 to ADRE11
AIN11	1 channels	ADREG11 to ADRE11

Note: When the range of channel scan is set to out of the assignable value in channel scan mode, the AD conversion can not be activated even if ADMOD0 is set to activate AD conversion.

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	-	80	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

< 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 circuitd. 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.

Cautions for use 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.

19.4.5.8 The way of stopping ADC in the low power consumption mode

When ADC is stopped in the low power consumption mode, it is stopped by the following order.

1. ADC is stopped in the state of ADMOD1<VREFON>="1".
2. ADCLK<ADCLK[2:0]> is set to "010" (ADCLK=fc/4) or "100" (ADCLK=fc/8).
3. Make ADC initialize by a software reset (ADMOD6<ADRST[1:0]>="10" → "01").
4. The CPU enters in the low power consumption mode.

20. Flash Memory Operation

This section describes the hardware configuration and operation of the flash memory.

20.1 Flash Memory

20.1.1 Features

1. Memory capacity

The TMPM366FDXBG/FYXBG/FWXBG devices contain flash memory. The memory sizes and configurations of each device are shown in the table below.

Independent write access to each block is available. When the CPU is to access the internal flash memory, 32-bit data bus width is used.

2. Write/erase time

Writing is executed per page. The TMPM366FDXBG/FYXBG/FWXBG contain 128 words.

Page writing requires 1.25ms (typical) regardless of number of words.

A block erase requires 0.1 sec. (typical).

The following table shows write and erase time per chip.

Product Name	Memory Size	Block Configuration				Number of Words	Write Time	Erase Time
		128 KB	64 KB	32 KB	16 KB			
TMPM366FDFG	512 KB	3	1	2	−	128	1.28 sec	0.4 sec

Note: **The above values are theoretical values not including data transfer time.**
The write time per chip depends on the write method to be used by the user.

3. Programming method

There are two types of the onboard programming mode for the user to program (rewrite) the device while it is mounted on the user's board:

- The onboard programming mode

a. User boot mode

The user's original rewriting method can be supported.

b. Single boot mode

The rewriting method to use serial data transfer (Toshiba's unique method) can be supported.

4. Rewriting method

The flash memory included in this device is generally compliant with the applicable JEDEC standards except for some specific functions. Therefore, if the user is currently using an external flash mem-

ory device, it is easy to implement the functions into this device. Furthermore, the user is not required to build his/her own programs to realize complicated write and erase functions because such functions are automatically performed using the circuits already built-in the flash memory chip.

JEDEC compliant functions	Modified, added, or deleted functions
・ Automatic programming ・ Automatic chip erase ・ Automatic block erase ・ Data polling/toggle bit	<Modified> Block protect (only software protection is supported) <Deleted> Erase resume - suspend function

5. Protect/ Security Function

This device is also implemented with a read-protect function to inhibit reading flash memory data from any external writer device. On the other hand, rewrite protection is available only through command-based software programming; any hardware setting method to apply +12VDC is not supported. See the chapter "ROM protection" for details of ROM protection and security function.

Note: **If a password is set to 0xFF (erased data), it is difficult to protect data securely due to an easy-to-guess password. Even if Single Boot mode is not used, it is recommended to set a unique value as a password.**

20.1.2 Block Diagram of the Flash Memory Section

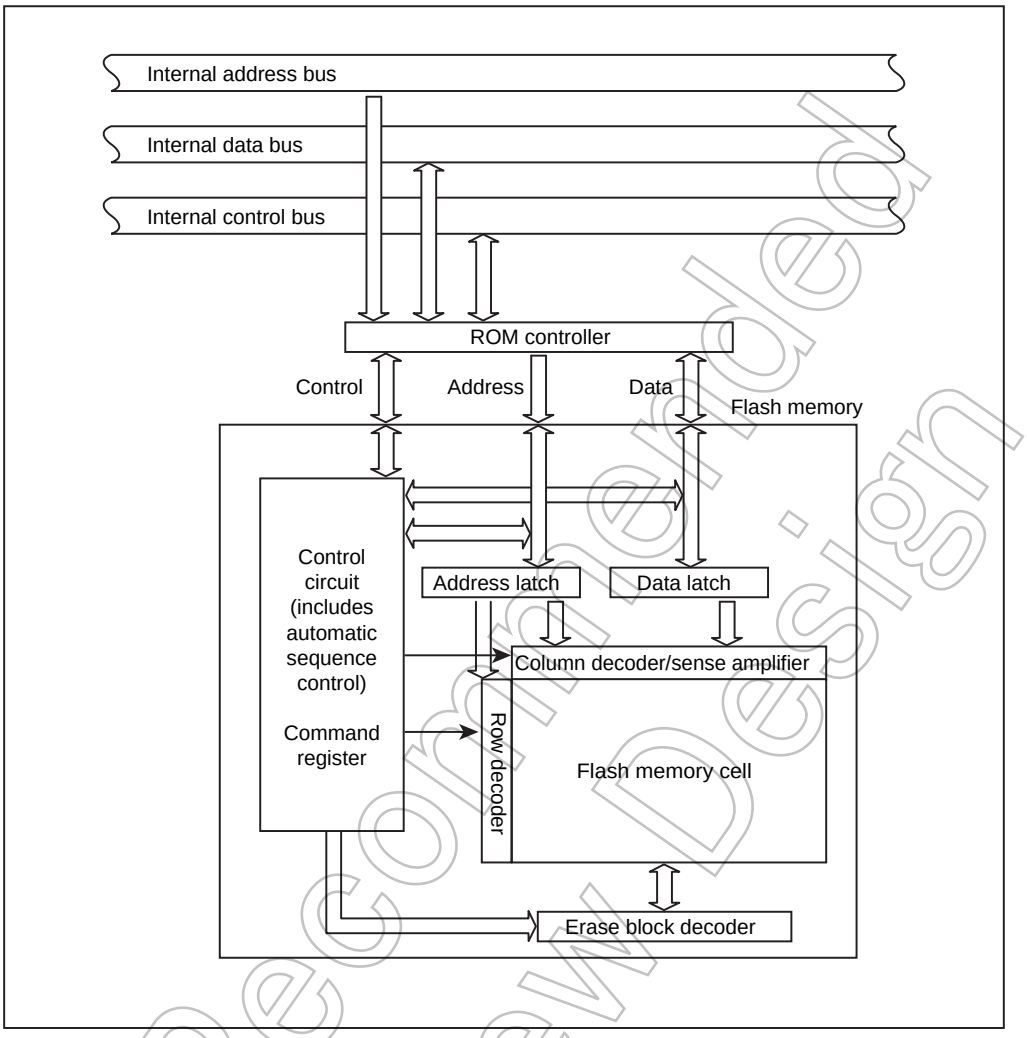


Figure 20-1 Block Diagram of the Flash Memory Section

20.2 Operation Mode

This device has three operation modes including the mode not to use the internal flash memory.

Table 20-1 Operation Modes

Operation mode	Operation details
Single chip mode	After reset is cleared, it starts up from the internal flash memory.
Normal mode	In this operation mode, two different modes, i.e., the mode to execute user application programs and the mode to rewrite the flash memory onboard the user's card, are defined. The former is referred to as "normal mode" and the latter "user boot mode.
User boot mode	The user can uniquely configure the system to switch between these two modes. For example, the user can freely design the system such that the normal mode is selected when the port "A0" is set to "1" and the user boot mode is selected when it is set to "0." The user should prepare a routine as part of the application program to make the decision on the selection of the modes.
Single boot mode	After reset is cleared, it starts up from the internal Boot ROM (Mask ROM). In the Boot ROM, an algorithm to enable flash memory rewriting on the user's set through the serial port of this device is programmed. By connecting to an external host computer through the serial port, the internal flash memory can be programmed by transferring data in accordance with predefined protocols.

Among the flash memory operation modes listed in the above table, the User Boot mode and the Single Boot mode are the programmable modes. These two modes, the User Boot mode and the Single Boot mode, are referred to as "Onboard Programming" modes where onboard rewriting of internal flash memory can be made on the user's card.

Either the Single Chip or Single Boot operation mode can be selected by externally setting the level of the $\overline{\text{BOOT}}$ (PF0) pin while the device is in reset status.

Table 20-2 Operation Mode Setting

Operation mode	Pin	
	$\overline{\text{RESET}}$	$\overline{\text{BOOT}}$ (PF0)
Single chip mode	0 → 1	1
Single boot mode	0 → 1	0

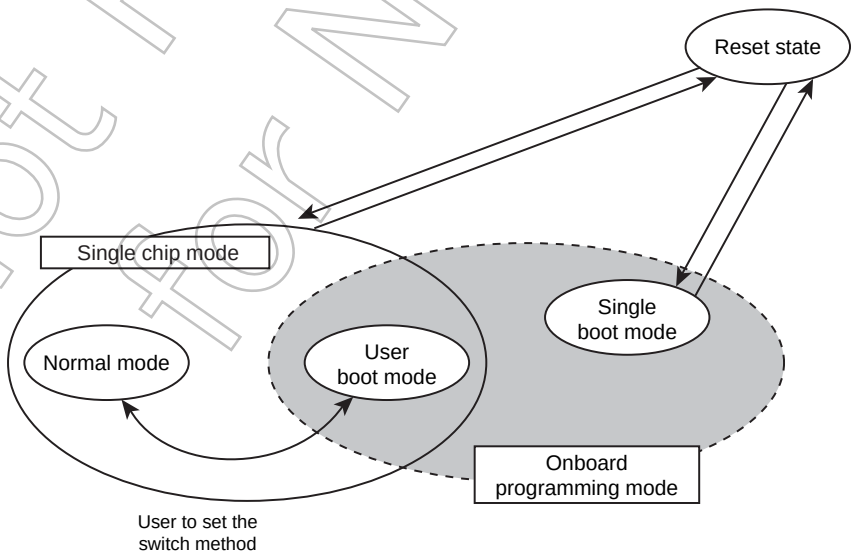


Figure 20-2 Mode Transition Diagram

20.2.1 Reset Operation

To reset the device, ensure that the power supply voltage is within the operating voltage range, that the internal oscillator has been stabilized, and that the $\overline{\text{RESET}}$ input is held at "0" for a minimum duration of 12 system clocks (0.3 μs with 40MHz operation; the "1/1" clock gear mode is applied after reset).

Note 1: **Regarding cold-reset of devices with internal flash memory; for devices with internal flash memory, it is necessary to apply "0" to the $\overline{\text{RESET}}$ inputs upon power on for a minimum duration of 1.4 milli-seconds regardless of the operating frequency.**

Note 2: **While flash automatic programming or deletion is in progress, at least 0.5 microseconds of reset period is required regardless of the system clock frequency. In this condition, it takes approx. 2 ms to enable reading after reset.**

20.2.2 User Boot Mode (Single chip mode)

User Boot mode is to use flash memory programming routine defined by users. It is used when the data transfer buses for flash memory program code on the old application and for serial I/O are different. It operates at the single chip mode; therefore, a switch from normal mode in which user application is activated at the single chip mode to User Boot Mode for programming flash is required. Specifically, add a mode judgment routine to a reset program in the old application.

The condition to switch the modes needs to be set by using the I/O of TMPM366FDXBG/FYXBG/FWXBG in conformity with the user's system setup condition. Also, flash memory programming routine that the user uniquely makes up needs to be set in the new application. This routine is used for programming after being switched to User Boot Mode. The execution of the programming routine must take place while it is stored in the area other than the flash memory since the data in the internal flash memory cannot be read out during delete/ writing mode. Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental corruption during subsequent Single-Chip (Normal mode) operations. Be sure not to cause any exceptions including a non-maskable while User Boot Mode.

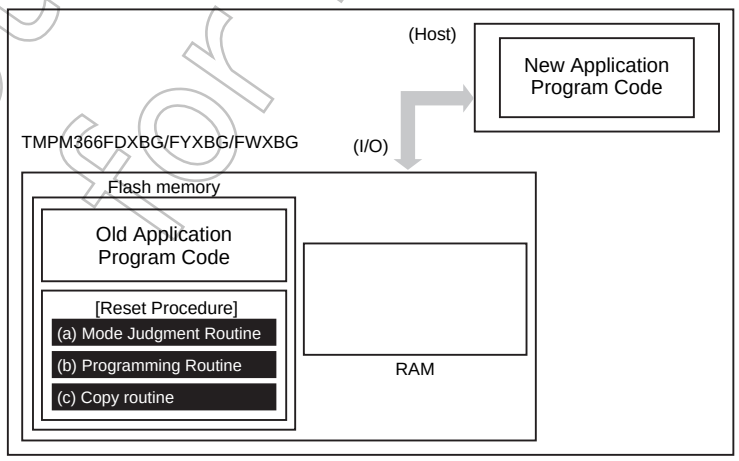
(1-A) and (1-B) are the examples of programming with routines in the internal flash memory and in the external memory. For a detailed description of the erase and program sequence, refer to "20.3 On-board Programming of Flash Memory (Rewrite/Erase)".

20.2.2.1 (1-A) Method 1: Storing a Programming Routine in the Flash Memory

(1) Step-1

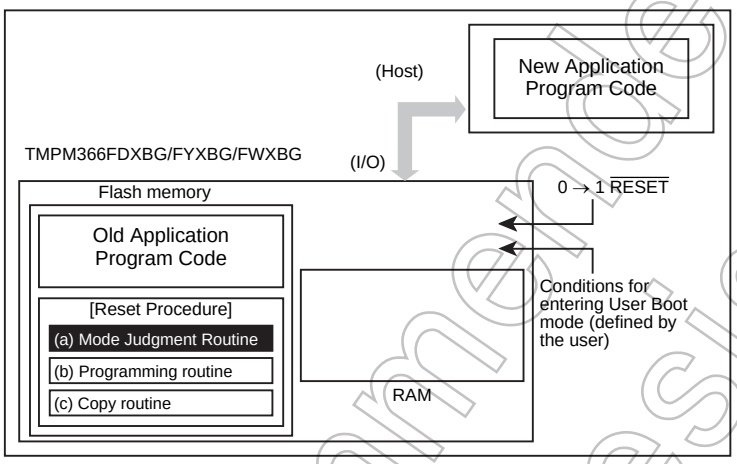
Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMPM366FDXBG/FYXBG/FWXBG on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

(a) Mode judgment routine:	Code to determine whether or not to switch to User Boot mode
(b) Programming routine:	Code to download new program code from a host controller and re-program the flash memory
(c) Copy routine:	Code to copy the data described in (b) from the TMPM366FDXBG/FYXBG/FWXBG flash memory to either the TMPM366FDXBG/FYXBG/FWXBG on-chip RAM or external memory device.



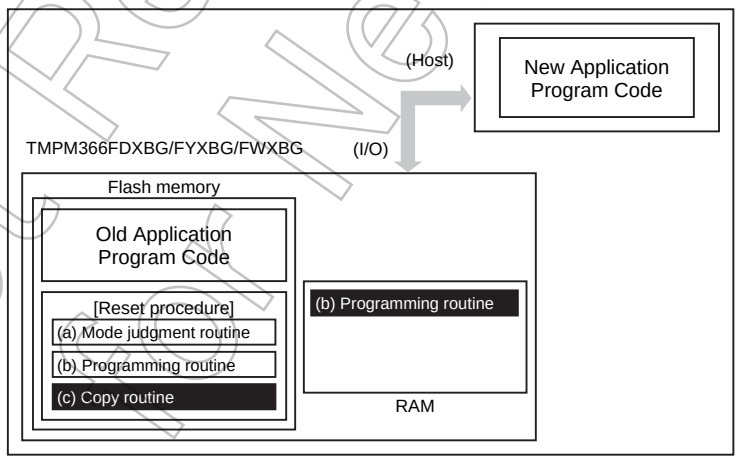
(2) Step-2

After $\overline{\text{RESET}}$ is released, the reset procedure determines whether to put the TMPM366FDXBG/FYXBG/FWXBG flash memory in User Boot mode. If mode switching conditions are met, the flash memory enters User Boot mode. (All interrupts including NMI must be disabled while in User Boot mode.)



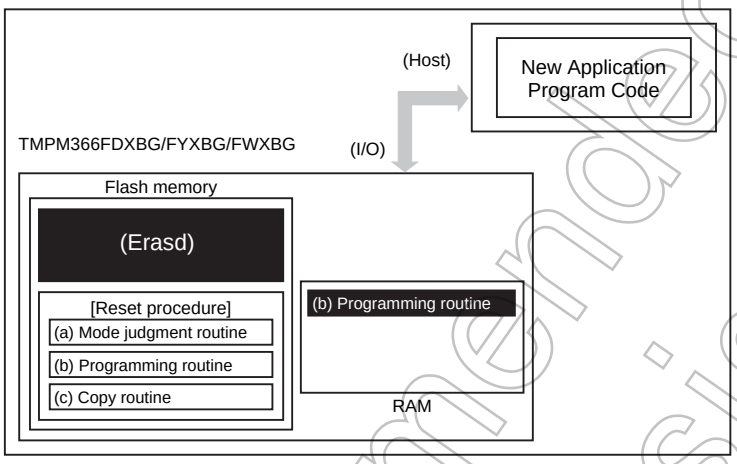
(3) Step-3

Once transition to User Boot mode is occurred, execute the copy routine (c) to copy the flash programming routine (b) to the TMPM366FDXBG/FYXBG/FWXBG on-chip RAM.



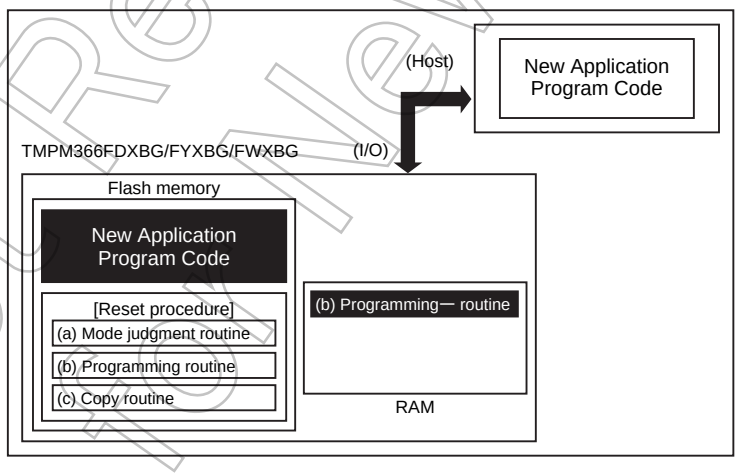
(4) Step-4

Jump program execution to the flash programming routine in the on-chip RAM to erase a flash block containing the old application program code.



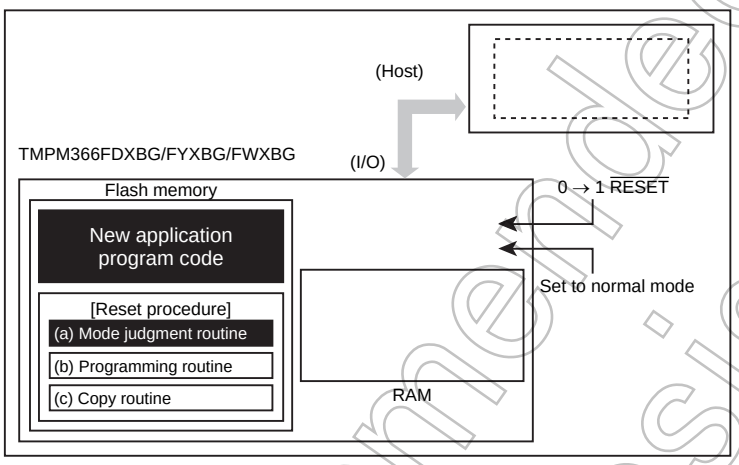
(5) Step-5

Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. When the programming is completed, the writing or erase protection of that flash block in the user's program area must be set.



(6) Step-6

Set $\overline{\text{RESET}}$ to "0" to reset the TMPM366FDXBG/FYXBG/FWXBG. Upon reset, the on-chip flash memory is put in Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.



20.2.2.2 (1-B) Method 2: Transferring a Programming Routine from an External Host

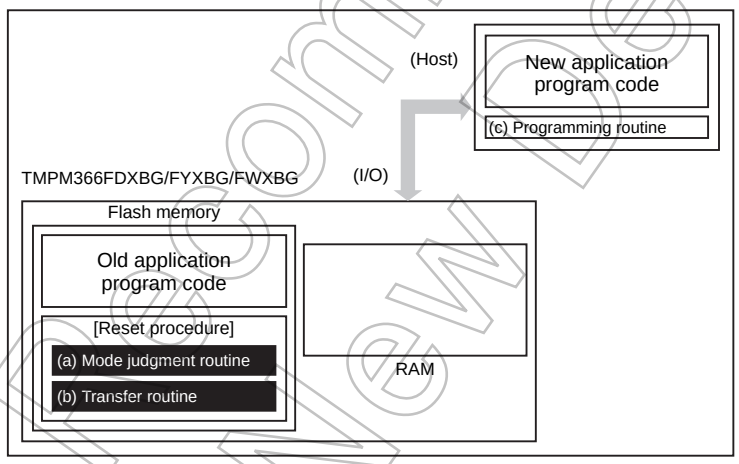
(1) Step-1

Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMPM366FDXBG/FYXBG/FWXBG on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

(a) Mode judgment routine: Code to determine whether or not to switch to User Boot mode
(b) Transfer routine: Code to download new program code from a host controller

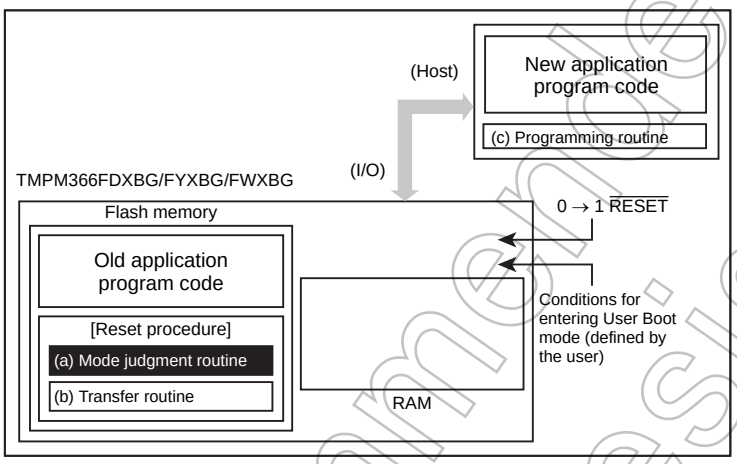
Also, prepare a programming routine shown below on the host controller:

(c) Programming routine: Code to download new program code from an external host controller and re-program the flash memory



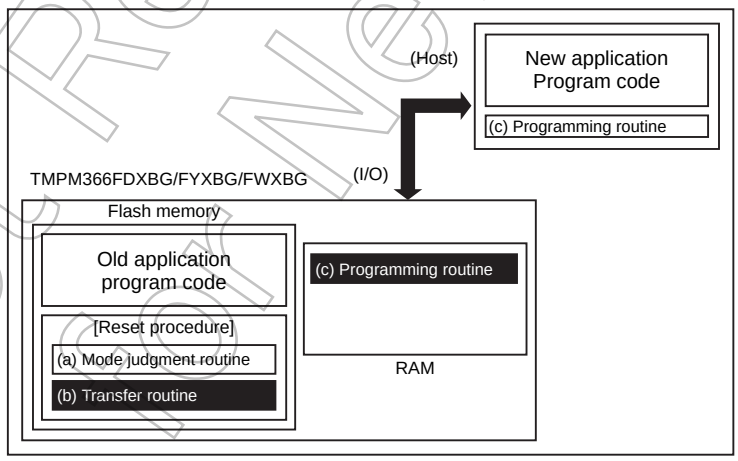
(2) Step-2

After $\overline{\text{RESET}}$ is released, the reset procedure determines whether to put the TMPM366FDXBG/FYXBG/FWXBG flash memory in User Boot mode. If mode switching conditions are met, the flash memory enters User Boot mode. (All interrupts including NMI must be disabled while in User Boot mode).



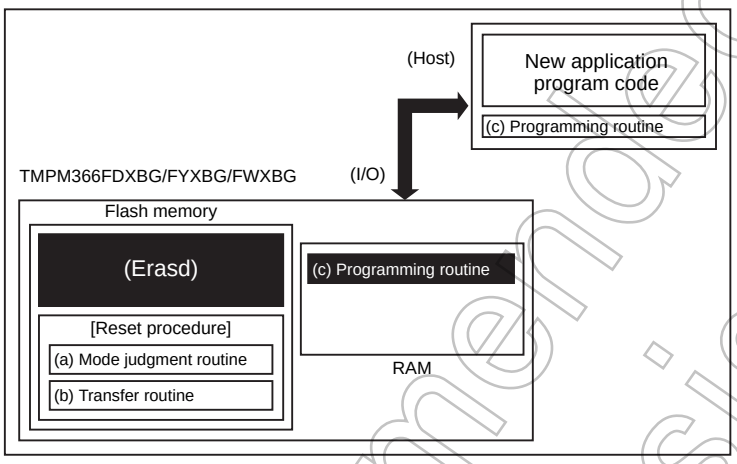
(3) Step-3

Once User Boot mode is entered, execute the transfer routine (b) to download the flash programming routine (c) from the host controller to the TMPM366FDXBG/FYXBG/FWXBG on-chip RAM.



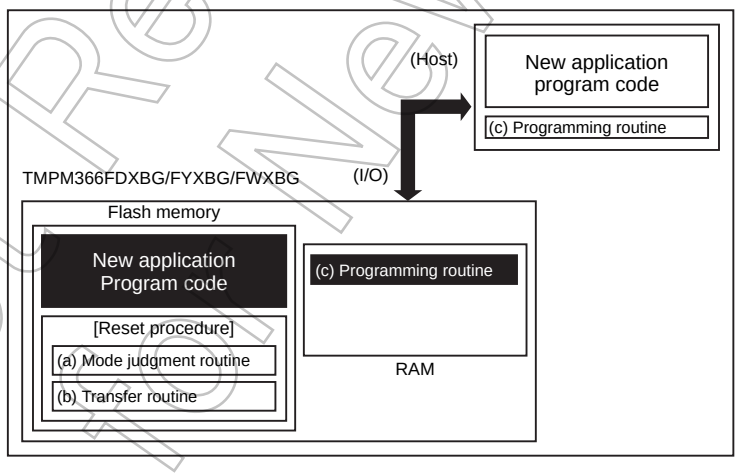
(4) Step-4

Jump program execution to the flash programming routine in the on-chip RAM to erase a flash block containing the old application program code.



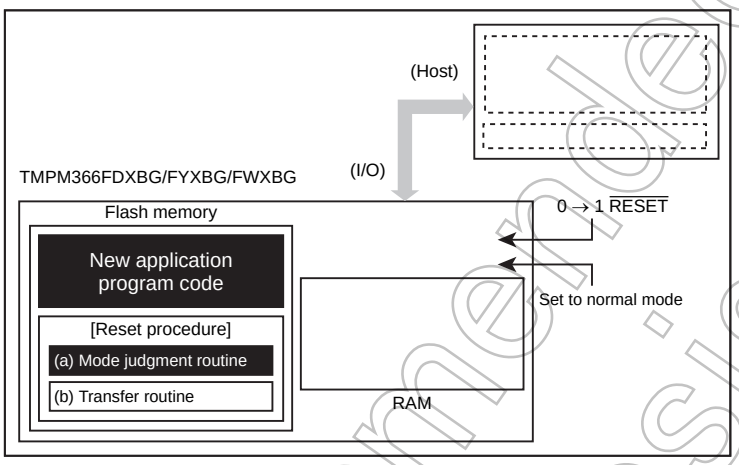
(5) Step-5

Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. When the programming is completed, the writing or erase protection of that flash block in the user program area must be set.



(6) Step-6

Set $\overline{\text{RESET}}$ to "0" low to reset the TMPM366FDXBG/FYXBG/FWXBG. Upon reset, the on-chip flash memory is put in Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.



20.2.3 Single Boot Mode

In Single Boot mode, the flash memory can be re-programmed by using a program contained in the TMPM366FDXBG/FYXBG/FWXBG on-chip boot ROM. This boot ROM is a masked ROM. When Single Boot mode is selected upon reset, the boot ROM is mapped to the address region including the interrupt vector table while the flash memory is mapped to an address region different from it.

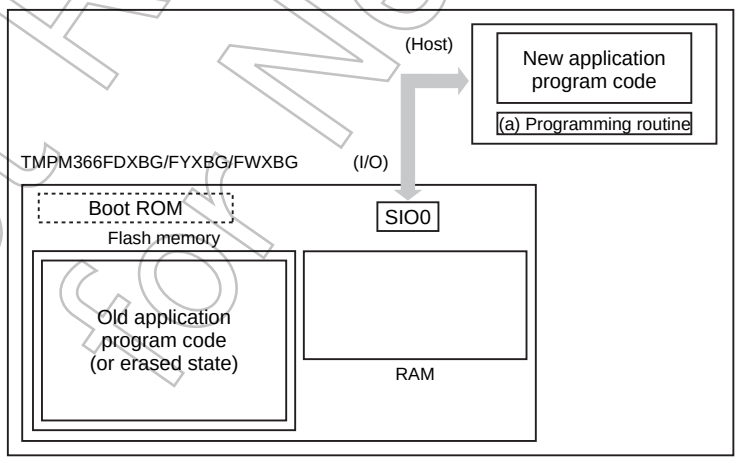
Single Boot mode allows for serial programming of the flash memory. Channel 0 of the SIO (SIO0) of the TMPM366FDXBG/FYXBG/FWXBG is connected to an external host controller. Via this serial link, a programming routine is downloaded from the host controller to the TMPM366FDXBG/FYXBG/FWXBG on-chip RAM. Then, the flash memory is re-programmed by executing the programming routine. The host sends out both commands and programming data to re-program the flash memory. Communications between the SIO0 and the host must follow the protocol described later. To secure the contents of the flash memory, the validity of the application's password is verified before a programming routine is downloaded into the on-chip RAM. If password matching fails, the transfer of a programming routine itself is aborted. As in the case of User Boot mode, all interrupts including the non-maskable interrupt (NMI) must be disabled in Single Boot mode while the flash memory is being erased or programmed. In Single Boot mode, the boot-ROM programs are executed in Normal mode.

Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental corruption during subsequent Single-Chip (Normal mode) operations.

20.2.3.1 (2-A) Using the Program in the On-Chip Boot ROM

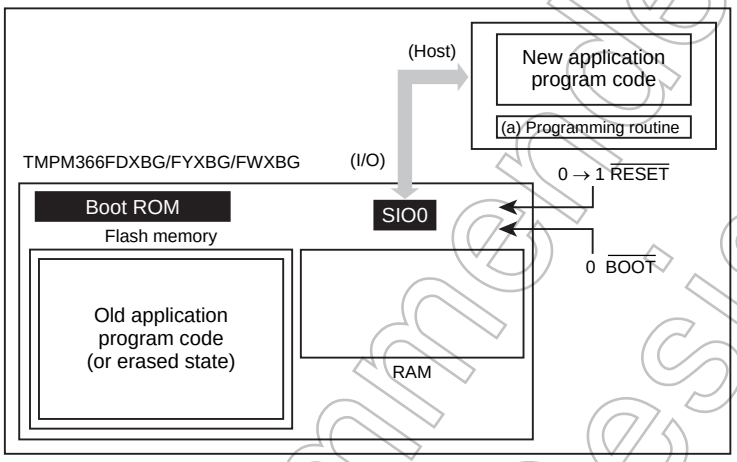
(1) Step-1

The flash block containing the older version of the program code need not be erased before executing the programming routine. Since a programming routine and programming data are transferred via the SIO (SIO0), the SIO0 must be connected to a host controller. Prepare a programming routine (a) on the host controller.



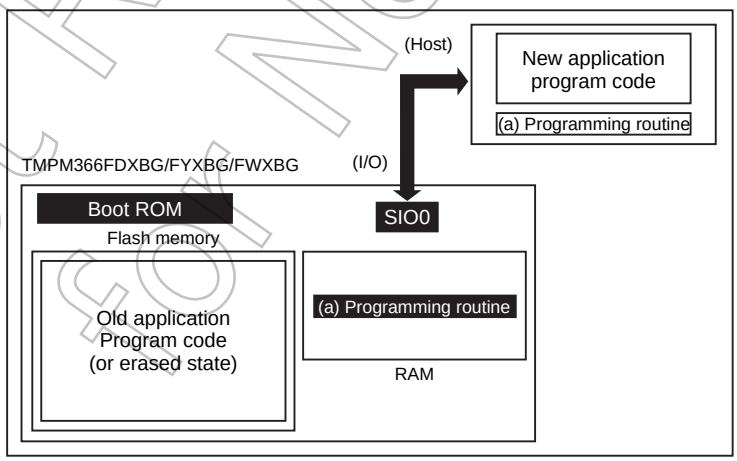
(2) Step-2

Set the $\overline{\text{RESET}}$ pin to "1" to cancel the reset of the TMPM366FDXBG/FYXBG/FWXBG when the $\overline{\text{BOOT}}$ pin has already been set to "0". After reset, CPU reboots from the on-chip boot ROM. The 12-byte password transferred from the host controller via SIO0 is first compared to the contents of the special flash memory locations. (If the flash block has already been erased, the password is 0xFF).



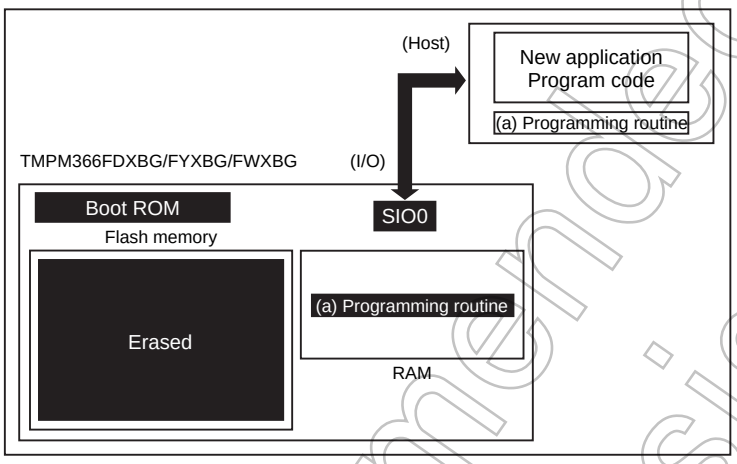
(3) Step-3

If the password was correct, the boot program downloads, via the SIO0, the programming routine (a) from the host controller into the on-chip RAM of the TMPM366FDXBG/FYXBG/FWXBG. The programming routine must be stored in the range from 0x2000_0800 to the end address of RAM.



(4) Step-4

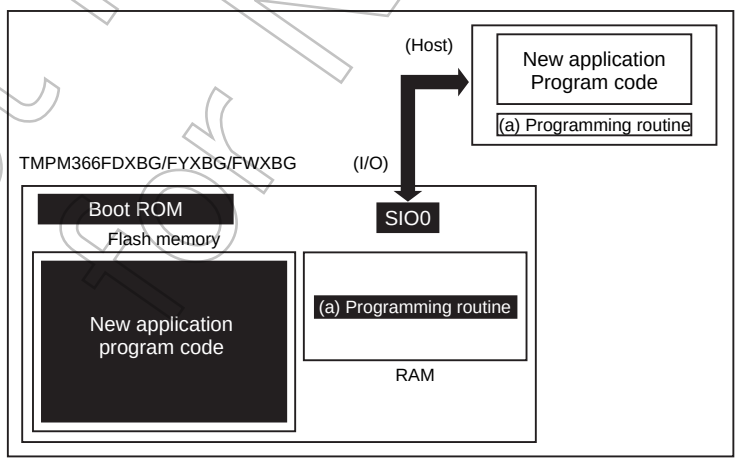
The CPU jumps to the programming routine (a) in the on-chip RAM to erase the flash block containing the old application program code. The Block Erase or Chip Erase command may be used.



(5) Step-5

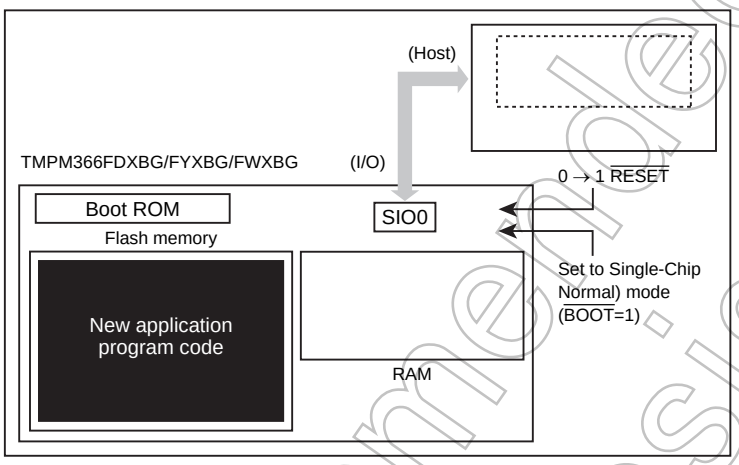
Next, the programming routine (a) downloads new application program code from the host controller and programs it into the erased flash block. When the programming is completed, the writing or erase protection of that flash block in the user's program area must be set.

In the example below, new program code comes from the same host controller via the same SIO0 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 board hardware and a programming routine to suit your particular needs.



(6) Step-6

When programming of the flash memory is complete, power off the board and disconnect the cable leading from the host to the target board. Turn on the power again so that the TMPM366FDXBG/FYXBG/FWXBG re-boots in Single-Chip (Normal) mode to execute the new program.



20.2.4 Configuration for Single Boot Mode

To execute the on-board programming, boot the TMPM366FDXBG/FYXBG/FWXBG with Single Boot mode following the configuration shown below.

BOOT(PF0) = 0
RESET = 0 → 1

Set the RESET input to "0", and set the each BOOT (PF0) pins to values shown above, and then release RESET (high).

20.2.5 Memory Map

Figure 20-3 shows a comparison of the memory maps in Normal and Single Boot modes. In Single Boot mode, the internal flash memory is mapped to 0x3F80_0000 and later addresses, and the Internal boot ROM (Mask ROM) is mapped to 0x0000_0000 through 0x0000_0FFF.

The internal flash memory and RAM addresses of each device are shown below.

Product Name	Flash Size	RAM Size	Flash Address (Single Chip/ Single Boot Mode)	RAM Address
TMPM366FDXBG/ FYXBG/FWXBG	512 KB	32 KB	0x0000_0000 to 0x0007_FFFF 0x3F80_0000 to 0x3F87_FFFF	0x2000_0000 to 0x2000_7FFF

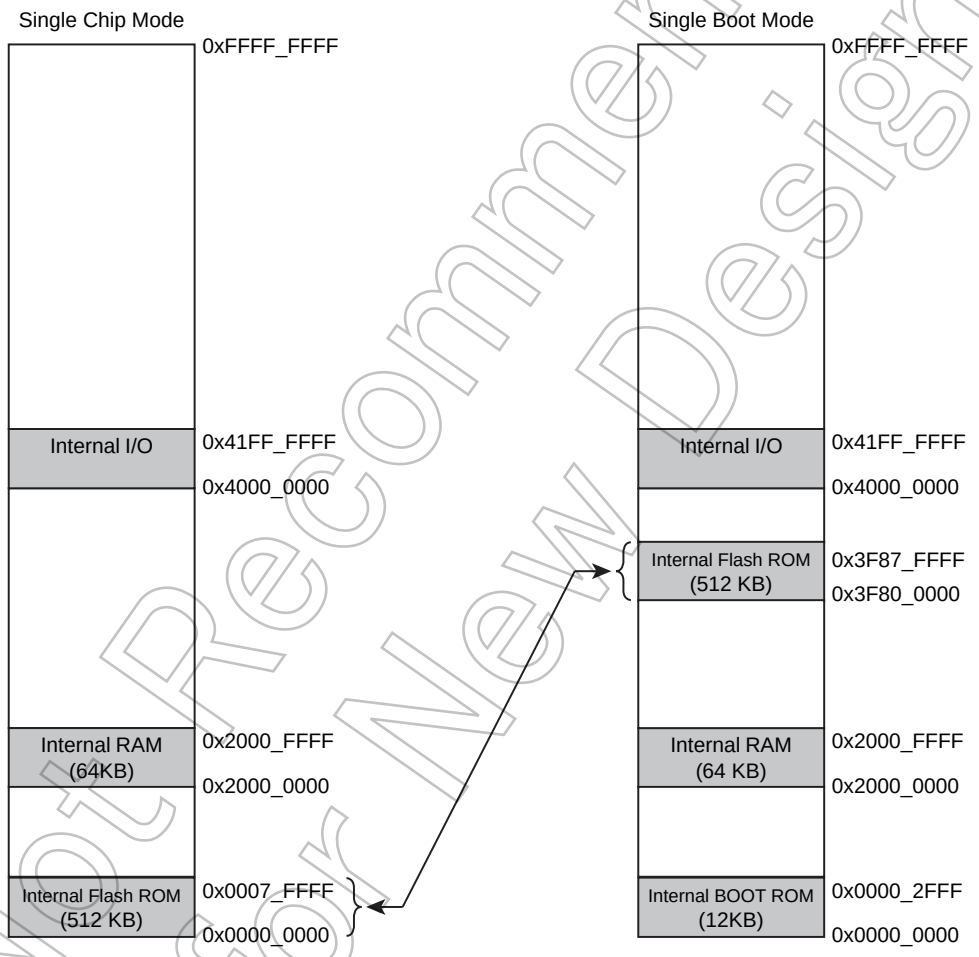


Figure 20-3 Memory Maps for TMPM366FDXBG/FYXBG/FWXBG

20.2.6 Interface specification

In Single Boot mode, an SIO channel is used for communications with a programming controller. The same configuration is applied to a communication format on a programming controller to execute the on-board programming. Both UART (asynchronous) and I/O Interface (synchronous) modes are supported.

In USB Boot mode, a USB port is used for communication with a programing controll .The same configuration is applied to a communication format on a programming controller to execute the on-board programming.

The communication formats are shown below.

- UART communication

  Communication channel : SIO channel 0

  Serial transfer mode : UART (asynchronous), half -duplex, LSB first

  Data length : 8 bit

  Parity bits : None

  STOP bits : 1 bit

  Baud rate : Arbitrary baud rate
- I/O interface mode

  Communication channel : SIO channel 0

  Serial transfer mode : I/O interface mode, full -duplex, LSB first

  Synchronization clock (SCLK0) : Input mode

  Handshaking signal : PE4 configured as an output mode

  Baud rate : Arbitrary baud rate
- USB Boot mode

  Communication port : D+/D-

  Full-Speed communication only

  Transfer mode : Control /Bulk

  USB clock : 48MHz(12/16MHz Crystal with PLL / External Input)

Table 20-3 Required Pin Connections

Pins		Interface		
		UART	I/O Interface Mode	USB
Mode-setting pin	MODE	Connect with Pull-down resistance		
	FTEST3	fixed to open		
	PE3	×	×	o (="L":Internal clock ,="H":External Clock)
	PE5	o (="L" input)	o (="L" input)	o (="H" input)
	PG5	×	×	o (for Vbus detection)
	$\overline{\text{BOOT}}$ (PF0)	o	o	o
	X1	-	-	o(12/16/48MHz)
Reset pin	$\overline{\text{RESET}}$	o	o	o
Communication pins	TXD0 (PE0)	o	o	×
	RXD0 (PE1)	o	o	×
	SCLK0 (PE2)	×	o (Input mode)	×
	PE4	×	o (Output mode)	o (Output mode)
	D+	×	×	o
	D-	×	×	o

o : used ×: unused

20.2.7 Data Transfer Format

Table 20-4 and Table 20-6 to Table 20-7 illustrate the operation commands and data transfer formats at each operation mode. In conjunction with this section, refer to "20.2.10 Operation of Boot Program".

Table 20-4 Single Boot Mode Commands

Code	Command
0x10	RAM transfer
0x40	Chip and protection bit erase

20.2.8 Restrictions on internal memories

Single Boot Mode places restrictions on the internal RAM and ROM as shown in Table 20-5.

Table 20-5 Restrictions in Single Boot Mode

Memory	Details
Internal RAM	A program contained in the BOOT ROM uses the area, through 0x2000_0000 to 0x2000_07FF as a work area. Store the RAM transfer program from 0x2000_0800 through the end address of RAM.
Internal ROM	The following addresses are assigned for storing software ID information and passwords. Storing program in these addresses is not recommendable. TMPM366FDXBG/FYXBG/FWXBG: 0x3F87_FFF0 to 0x3F87_FFFF

20.2.9 Transfer Format for Single Boot Mode commands

The following tables shows the transfer format for each Single Boot Mode command. Use this section in conjunction with Chapter "20.2.10 Operation of Boot Program".

20.2.9.1 RAM Transfer

Table 20-6 Transfer Format for the RAM Transfer Command

	Byte	Data Transferred from the Controller to the TMPM366FDXBG/FYXBG/FWXBG	Baud rate	Data Transferred from the TMPM366FDXBG/FYXBG/FWXBG to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode : 0x86 For I/O Interface mode : 0x30	Desired baud rate (Note 1)	−
	2 byte	−		ACK for the serial operation mode byte ・For UART mode -Normal acknowledge : 0x86 (The boot program aborts if the baud rate can not be set correctly.) ・For I/O Interface mode -Normal acknowledge :0x30
	3 byte	Command code (0x10)		−
	4 byte	−		ACK for the command code byte (Note 2) -Normal acknowledge : 0x10 -Negative acknowledge : 0xX1 -Communication error : 0xX8
	5 byte to 16 byte	Password sequence (12 bytes) 0x3F87_FFF4 to 0x3F87_FFFF		−
	17 byte	Check SUM value for bytes 5 - 16		−
	18 byte	−		ACK for the checksum byte (Note 2) -Normal acknowledge : 0x10 -Negative acknowledge : 0xX1 -Communication error : 0xX8
	19 byte	RAM storage start address 31 to 24		−
	20 byte	RAM storage start address 23 to 16		−
	21 byte	RAM storage start address 15 to 8		−
	22 byte	RAM storage start address 7 to 0		−
	23 byte	RAM storage byte count 15 to 8		−
	24 byte	RAM storage byte count 7 to 0		−
	25 byte	Check SUM value for bytes 19 to 24		−
	26 byte	−		ACK for the checksum byte (Note 2) -Normal acknowledge : 0x10 -Negative acknowledge : 0xX1 -Communication error : 0xX8
	27 byte to m byte	RAM storage data		−
	m + 1 byte	Checksum value for bytes 27 to m		−
	m + 2 byte	−		ACK for the checksum byte (Note 2) -Normal acknowledge : 0x10 -Negative acknowledge : 0xX1 -Communication error : 0xX8
RAM	m + 3 byte	−		Jump to RAM storage start address

Note 1: In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.

Note 2: In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.

Note 3: The 19th to 25th bytes must be within the RAM address range from 0x2000_0800 through the end address of RAM.

20.2.9.2 Chip Erase and Protect Bit Erase

Table 20-7 Transfer Format for the Chip and Protection Bit Erase Command

	Byte	Data Transferred from the Controller to the TMPM366FDXBG/FYXBG/FWXBG	Baud rate	Data Transferred from the TMPM366FDXBG/ FYXBG/FWXBG to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode : 0x86 For I/O Interface mode : 0x30	Desired baud rate (Note 1)	−
	2 byte	−		ACK for the serial operation mode byte ・For UART mode -Normal acknowledge : 0x86 ・For I/O Interface mode -Normal acknowledge : 0x30 (The boot program aborts if the baud rate can not be set correctly.)
	3 byte	Command code (0x40)		−
	4 byte	−		ACK for the command code byte (Note 2) -Normal acknowledge : 0x40 -Negative acknowledge : 0xX1 -Communication error : 0xX8
	5 byte to 16 byte	Password sequence (12 bytes) 0x3F87_FFF4 to 0x3F87_FFFF		−
	17 byte	Check SUM value for bytes 5 - 16		−
	18 byte	−		ACK for the checksum byte (Note 2) -Normal acknowledge : 0x40 -Negative acknowledge : 0xX1 -Communication error : 0xX8
	19 byte	Erase command code (0x54)		−
	20 byte	−		ACK for the command code byte (Note 2) -Normal acknowledge : 0x54 -Negative acknowledge : 0xX1 -Communication error : 0xX8
	21 byte	−		ACK for the chip erase command code byte -Normal acknowledge : 0x4F -Negative acknowledge : 0x4C
	22 byte	(Wait for the next command code.)		−

Note 1: **In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.**

Note 2: **In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.**

20.2.10 Operation of Boot Program

When Single Boot mode is selected, the boot program is automatically executed on startup. The boot program offers these four commands, of which the details are provided on the following subsections.

1. RAM Transfer command

   The RAM Transfer command stores program code transferred from a host controller to the on-chip RAM and executes the program once the transfer is successfully completed. The user program RAM space can be assigned to the range from 0x2000_0800 to the end address of RAM, whereas the boot program area (0x2000_0000 ~ 0x2000_07FF) is unavailable. The user program starts at the assigned RAM address.

   The RAM Transfer command can be used to download a flash programming routine of your own; this provides the ability to control on-board programming of the flash memory in a unique manner. The programming routine must utilize the flash memory command sequences described in Section 20.3. Before initiating a transfer, the RAM Transfer command verifies a password sequence coming from the controller against that stored in the flash memory.

Note: **If a password is set to 0xFF (erased data), it is difficult to protect data securely due to an easy-to-guess password. Even if Single Boot mode is not used, it is recommended to set a unique value as a password.**

2. Chip and Protection Bit Erase command

   This command erases the entire area of the flash memory automatically without verifying a password. All the blocks in the memory cell and their protection conditions are erased even when any of the blocks are prohibited from writing and erasing. When the command is completed, the FCSECBIT <SECBIT> bit is set to "1". This command serves to recover boot programming operation when a user forgets the password. Therefore password verification is not executed.

20.2.10.1 RAM Transfer Command

See Table 20-6 for the transfer format of this command.

1. The 1st byte specifies which one of the two serial operation modes is used. For a detailed description of how the serial operation mode is determined, see "20.2.10.4 Determination of a Serial Operation Mode" described later. If it is determined as UART mode, the boot program then checks if the SIO0 is programmable to the baud rate at which the 1st byte was transferred. During the first-byte interval, the RXE bit in the SC0MOD register is cleared.

   - To communicate in UART mode

     Send, from the controller to the target board, 0x86 in UART data format at the desired baud rate. If the serial operation mode is determined as UART, then the boot program checks if the SIO0 can be programmed to the baud rate at which the first byte was transferred. If that baud rate is not possible, the boot program aborts, disabling any subsequent communications.
   - To communicate in I/O Interface mode

     Send, from the controller to the target board, 0x30 in I/O Interface data format at 1/16 of the desired baud rate. Also send the 2nd byte at the same baud rate. Then send all subsequent bytes at a rate equal to the desired baud rate.

     In I/O Interface mode, the CPU sees the serial receive pin as if it were a general input port in monitoring its logic transitions. If the baud rate of the incoming data is high or the chip's operating frequency is high, the CPU may not be able to keep up with the speed of logic transitions. To prevent such situations, the 1st and 2nd bytes must be transferred at 1/16 of the desired baud rate; then the boot program calculates 16 times that as the desired baud rate. When the serial operation mode is determined as I/O Interface mode, the SIO0 is configured for SCLK Input mode. Beginning with the third byte, the controller must ensure that its AC timing restrictions are satisfied at the selected baud rate. In the case of I/O Interface mode, the boot program does not check the receive error flag; thus there is no such thing as error acknowledge (bit 3, 0xX8).

2. The 2nd byte, transmitted from the target board to the controller, is an acknowledge response to the 1st byte. The boot program echoes back the first byte: 0x86 for UART mode and 0x30 for I/O Interface mode.

   - UART mode

     If the SIO0 can be programmed to the baud rate at which the 1st byte was transferred, the boot program programs the SC0BRCR and sends back 0x86 to the controller as an acknowledge. If the SIO0 is not programmable at that baud rate, the boot program simply aborts with no error indication. Following the 1st byte, the controller should allow for a time-out period of five seconds. If it does not receive 0x86 within the allowed time-out period, the controller should give up the communication. The boot program sets the RXE bit in the SC0MOD0 register to enable reception ("1") before loading the SIO transmit buffer with 0x86.
   - I/O Interface mode

     The boot program programs the SC0MOD0 and SC0CR registers to configure the SIO0 in I/O Interface mode (clocked by the rising edge of SCLK0), writes 0x30 to the SC0BUF. Then, the SIO0 waits for the SCLK0 signal to come from the controller. Following the transmission of the 1st byte, the controller should send the SCLK clock to the target board after a certain idle time (several microseconds). This must be done at 1/16 the desire baud rate. If the 2nd byte, which is from the target board to the controller, is 0x30, then the controller should take it as a go-ahead. The controller must then deliver the 3rd byte to the target board at a rate equal to the desired baud rate. The boot program sets the RXE bit in the SC0MOD register to enable reception before loading the SIO transmit buffer with 0x30.

3. The 3rd byte transmitted from the controller to the target board is a command. The code for the RAM Transfer command is 0x10.

4. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits 0xX8 (bit 3) and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

   If the 3rd byte is equal to any of the command codes listed in Table 20-4, the boot program echoes it back to the controller. When the RAM Transfer command was received, the boot program echoes back a value of 0x10 and then branches to the RAM Transfer routine. Once this branch is taken, password verification is done. Password verification is detailed in a later section "Password". If the 3rd byte is not a valid command, the boot program sends back 0xX1 (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

5. The 5th to 16th bytes transmitted from the controller to the target board, are a 12-byte password. Each byte is compared to the contents of following addresses in the flash memory. The verification is started with the 5th byte and the smallest address in the designated area. If the password verification fails, the RAM Transfer routine sets the password error flag.

Product name	Area
TMPM366FDXBG/ FYXBG/FWXBG	0x3F87_FFF4 to 0x3F87_FFFF

6. The 17th byte is a checksum value for the password sequence (5th to 16th bytes). To calculate the checksum value for the 12-byte password, add the 12 bytes together, drop the carries and take the two’s complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in a later section "Checksum Calculation".

7. The 18th byte, transmitted from the target board to the controller, is an acknowledge response to the 5th to 17th bytes. First, the RAM Transfer routine checks for a receive error in the 5th to 17th bytes. If there was a receive error, the boot program sends back 0x18 (bit 3) and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., 1). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

   Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 5th to 16th bytes must result in 0x00 (with the carry dropped). If it is not 0x00, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 0x11 to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

   Finally, the RAM Transfer routine examines the result of the password verification. The following two cases are treated as a password error. In these cases, the RAM Transfer routine sends back 0x11 (bit 0) to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

   - Irrespective of the result of the password comparison, all the 12 bytes of a password in the flash memory are the same value other than 0xFF.
   - Not the entire password bytes transmitted from the controller matched those contained in the flash memory.

When all the above verification has been successful, the RAM Transfer routine returns a normal acknowledge response (0x10) to the controller.

8. The 19th to 22nd bytes, transmitted from the controller the target board, indicate the start address of the RAM region where subsequent data (e.g., a flash programming routine) should be stored. The 19th byte corresponds to bits 31.24 of the address and the 22nd byte corresponds to bits 7.0 of the address. The start address of the stored RAM must be even address.

9. The 23rd and 24th bytes, transmitted from the controller to the target board, indicate the number of bytes that will be transferred from the controller to be stored in the RAM. The 23rd byte corresponds to bits 15.8 of the number of bytes to be transferred, and the 24th byte corresponds to bits 7.0 of the number of bytes.

10. The 25th byte is a checksum value for the 19th to 24th bytes. To calculate the checksum value, add all these bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in a later section "Checksum Calculation".

11. The 26th byte, transmitted from the target board to the controller, is an acknowledge response to the 19th to 25th bytes of data. First, the RAM Transfer routine checks for a receive error in the 19th to 25th bytes. If there was a receive error, the RAM Transfer routine sends back 0x18 and returns to the command wait state (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., 1). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

    Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 19th to 25th bytes must result in 0x00 (with the carry dropped). If it is not 0x00, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 0x11 to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

    - The RAM storage start address must be within the range of 0x2000_0800 to the end address of RAM.

    When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (0x10) to the controller.

12. The 27th to mth bytes from the controller are stored in the on-chip RAM of the TMPM366FDXBG/FYXBG/FWXBG. Storage begins at the address specified by the 19th. 22nd bytes and continues for the number of bytes specified by the 23rd.24th bytes.

13. The (m+1) th byte is a checksum value. To calculate the checksum value, add the 27th to mth bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in a later section "Checksum Calculation".

14. The (m+2) th byte is a acknowledge response to the 27th to (m+1) th bytes. First, the RAM Transfer routine checks for a receive error in the 27th to (m+1) th bytes. If there was a receive error, the RAM Transfer routine sends back 0x18 (bit 3) and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., 1). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

    Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 27th to (m+1) th bytes must result in 0x00 (with the carry dropped). If it is not 0x00, one or more bytes of data has been corrupted. In case of a checksum error,

the RAM Transfer routine sends back 0x11 (bit 0) to the controller and returns to the command wait state (i.e., the 3rd byte) again. When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (0x10) to the controller.

15. If the (m+2) th byte was a normal acknowledge response, a branch is made to the address specified by the 19th to 22nd bytes after a normal acknowledge response (0x10) is transfered.

20.2.10.2 Chip and Protection Bit Erase Command

See Table 20-7 for the transfer format of this command.

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. From the Controller to the TMPM366FDXBG/FYXBG/FWXBG

   The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Product Information command is 0x40.

3. From the TMPM366FDXBG/FYXBG/FWXBG to the Controller

   The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte.

   Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits 0xX8 (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

   If the 3rd byte is equal to any of the command codes listed in Table 20-4, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 0x40. If the 3rd byte is not a valid command, the boot program sends back 0xX1 (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

4. From the Controller to the TMPM366FDXBG/FYXBG/FWXBG

   The 5th byte, transmitted from the target board to the controller, is the Chip Erase Enable command code (0x54).

5. From the TMPM366FDXBG/FYXBG/FWXBG to the Controller

   The 6th byte, transmitted from the target board to the controller, is an acknowledge response to the 5th byte.

   Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits 0xX8 (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

   If the 5th byte is equal to any of the command codes to enable erasing, the boot program echoes it back to the controller. When the Chip and Protection Erase command was received, the boot program echoes back a value of 0x54 and then branches to the Chip Erase routine. If the 5th byte is not a valid command, the boot program sends back 0xX1 (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

6. From the TMPM366FDXBG/FYXBG/FWXBG to the Controller

   The 7th byte indicates whether the Chip Erase command is normally completed or not.

   At normal completion, completion code (0x4F) is sent.

   When an error was detected, error code (0x4C) is sent.

7. The 9th byte is the next command code.

20.2.10.3 Acknowledge Responses

The boot program represents processing states with specific codes. Table 20-8 to Table 20-11 show the values of possible acknowledge responses to the received data. The upper four bits of the acknowledge response are equal to those of the command being executed. Bit 3 of the code indicates a receive error. Bit 0 indicates an invalid command error, a checksum error or a password error. Bit 1 and bit 2 are always "0". Receive error checking is not done in I/O Interface mode.

Table 20-8 ACK Response to the Serial Operation Mode Byte

Return Value	Meaning
0x86	The SIO can be configured to operate in UART mode. (See Note)
0x30	The SIO can be configured to operate in I/O Interface mode.

Note: **If the serial operation mode is determined as UART, the boot program checks if the SIO can be programmed to the baud rate at which the operation mode byte was transferred. If that baud rate is not possible, the boot program aborts, without sending back any response.**

Table 20-9 ACK Response to the Command Byte

Return Value	Meaning
0xX8 (See Note)	A receive error occurred while getting a command code.
0xX1 (See Note)	An undefined command code was received. (Reception was completed normally.)
0x10	The RAM Transfer command was received.
0x40	The Chip Erase command was received.

Note: **The upper four bits of the ACK response are the same as those of the previous command code.**

Table 20-10 ACK Response to the Checksum Byte

Return Value	Meaning
0xX8 (See Note)	A receive error occurred.
0xX1 (See Note)	A checksum or password error occurred.
0xX0 (See Note)	The checksum was correct.

Note: The upper four bits of the ACK response are the same as those of the operation command code. It is 1 (X ; RAM transfer command data [7:4]) when password error occurs.

Table 20-11 ACK Response to Chip and Protection Bit Erase Byte

Return Value	Meaning
0x54	The Erase enabling command was received.
0x4F	The Erase command was completed.
0x4C	The Erase command was abnormally completed.

20.2.10.4 Determination of a Serial Operation Mode

The first byte from the controller determines the serial operation mode. To use UART mode for communications between the controller and the target board, the controller must first send a value of 0x86 at a desired baud rate to the target board. To use I/O Interface mode, the controller must send a value of 0x30 at 1/16 the desired baud rate. Figure 20-4 shows the waveforms for the first byte.

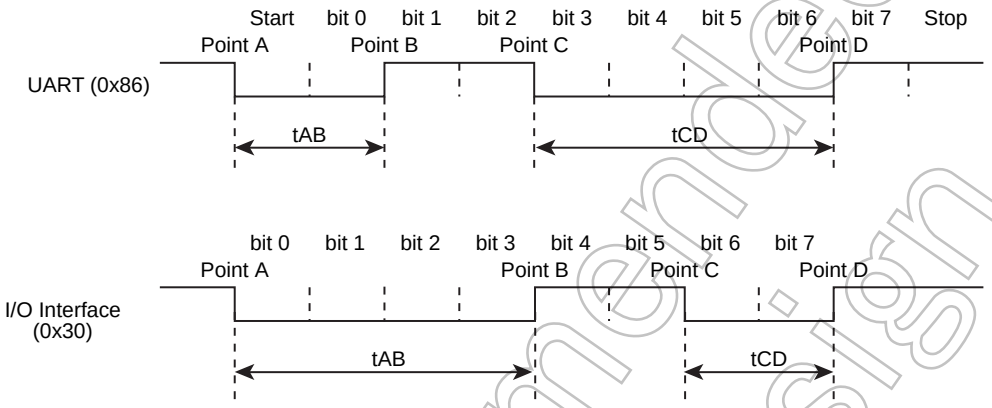


Figure 20-4 Serial Operation Mode Byte

After $\overline{\text{RESET}}$ is released, the boot program monitors the first serial byte from the controller, with the SIO reception disabled, and calculates the intervals of tAB, tAC and tAD. Figure 20-5 shows a flowchart describing the steps to determine the intervals of tAB, tAC and tAD. As shown in the flowchart, the boot program captures timer counts each time a logic transition occurs in the first serial byte. Consequently,the calculated tAB, tAC and tAD intervals are bound to have slight errors. If the transfer goes at a high baud rate, the CPU might not be able to keep up with the speed of logic transitions at the serial receive pin. In particular, I/O Interface mode is more prone to this problem since its baud rate is generally much higher than that for UART mode. To avoid such a situation, the controller should send the first serial byte at 1/16 the desired baud rate.

The flowchart in Figure 20-5 shows how the boot program distinguishes between UART and I/O Interface modes. If the length of tAB is equal to or less than the length of tCD, the serial operation mode is determined as UART mode. If the length of tAB is greater than the length of tCD, the serial operation mode is determined as I/O Interface mode. Bear in mind that if the baud rate is too high or the timer operating frequency is too low, the timer resolution will be coarse, relative to the intervals between logic transitions. This becomes a problem due to inherent errors caused by the way in which timer counts are captured by software; consequently the boot program might not be able to determine the serial operation mode correctly. To prevent this problem, reset UART mode within the programming routine.

For example, the serial operation mode may be determined to be I/O Interface mode when the intended mode is UART mode. To avoid such a situation, when UART mode is utilized, the controller should allow for a time-out period within which it expects to receive an echo-back (0x86) from the target board. The controller should give up the communication if it fails to get that echo-back within the allowed time. When I/O Interface mode is utilized, once the first serial byte has been transmitted, the controller should send the SCLK clock after a certain idle time to get an acknowledge response. If the received acknowledge response is not 0x30, the controller should give up further communications.

When the intended mode is I/O interface mode, the first byte does not have to be 0x30 as long as tAB is greater than tCD as shown above. 0x91, 0xA1 or 0xB1 can be sent as the first byte code to determine the falling edges of Point A and Point C and the rising edges of Point B and Point D. If tAB is greater than tCD and SIO is selected by the resolution of the operation mode determination, the second byte code is 0x30 even though the transmitted code on the first byte is not 0x30 (The first byte code to determine I/O interface mode is described as 0x30).

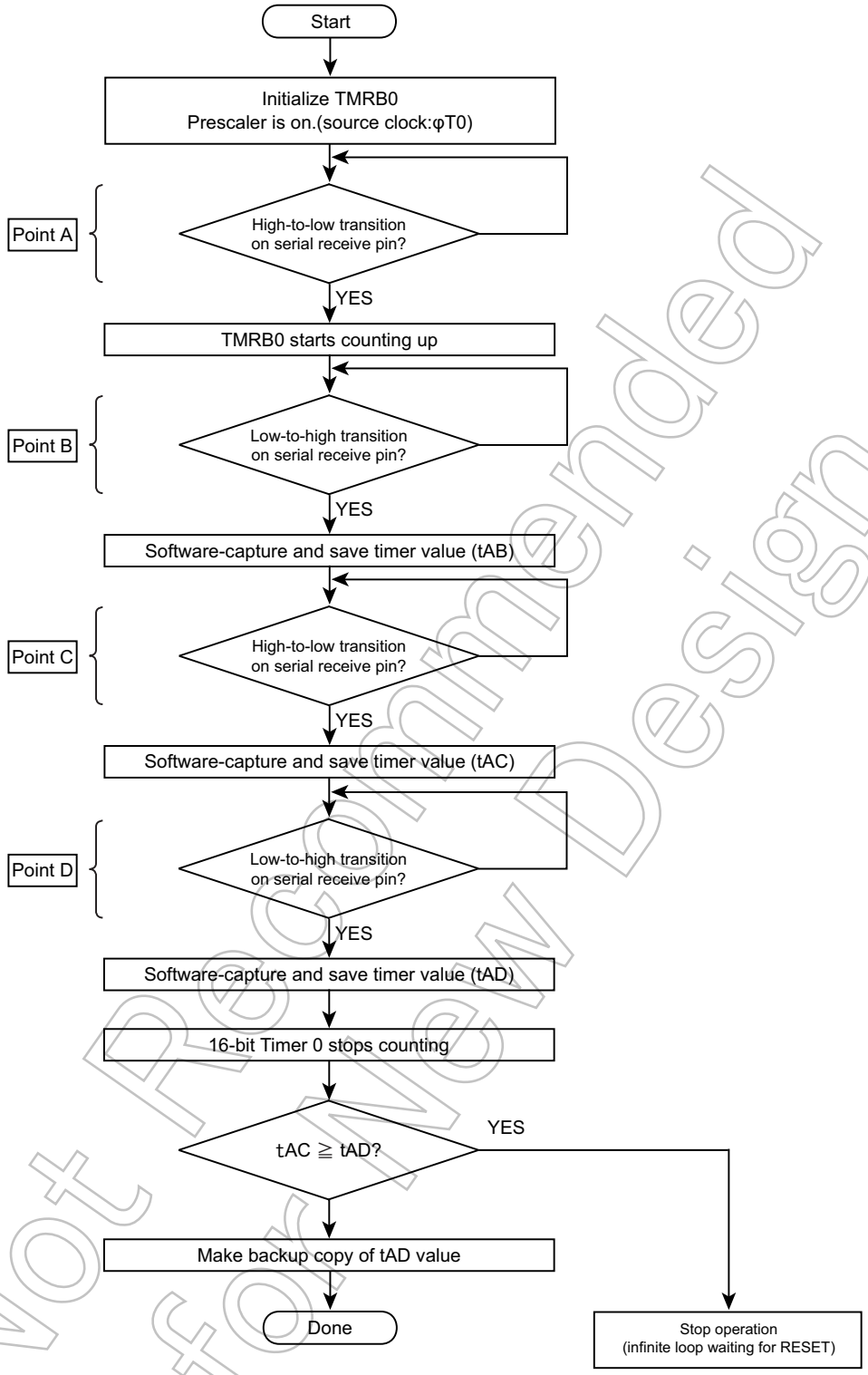


Figure 20-5 Serial Operation Mode Byte Reception Flowchart

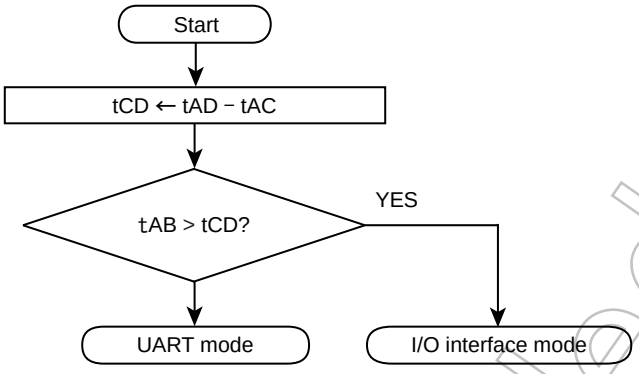


Figure 20-6 Serial Operation Mode Determination Flowchart

20.2.10.5 Password

The RAM Transfer command (0x10) causes the boot program to perform password verification. Following an echo-back of the command code, the boot program verifies the contents of the 12-byte password area within the flash memory. The following table shows the password area of each product.

Product name	Area
TMPM366FDXBG/ FYXBG/FWXBG	0x3F87_FFF4 to 0x3F87_FFFF

Note: **If a password is set to 0xFF (erased data area), it is difficult to protect data securely due to an easy-to-guess password. Even if Single Boot mode is not used, it is recommended to set a unique value as a password.**

If all these address locations contain the same bytes of data other than 0xFF, a password area error occurs as shown in Figure 20-7. In this case, the boot program returns an error acknowledge (0x11) in response to the checksum byte (the 17th byte), regardless of whether the password sequence sent from the controller is all 0xFFs.

The password sequence received from the controller (5th to 16th bytes) is compared to the password stored in the flash memory. All of the 12 bytes must match to pass the password verification. Otherwise, a password error occurs, which causes the boot program to reply an error acknowledge in response to the checksum byte (the 17th byte).

The password verification is performed even if the security function is enabled.

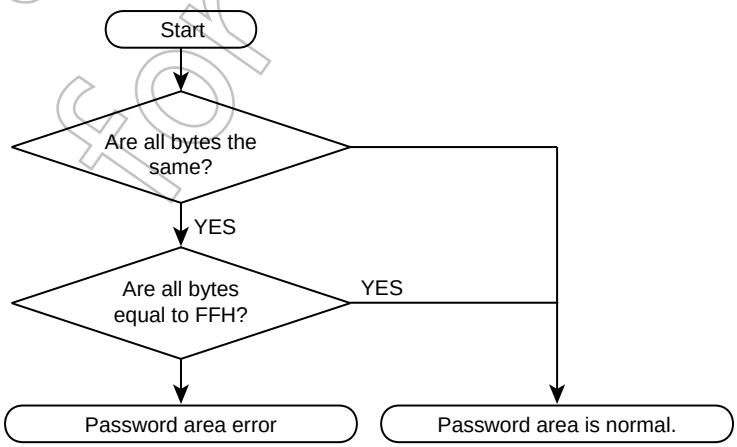


Figure 20-7 Password Area Verification Flowchart

20.2.10.6 Checksum Calculation

The checksum byte for a series of bytes of data is calculated by adding the bytes together, dropping the carries, and taking the two's complement of the total sum. The Show Flash Memory Sum command and the Show Product Information command perform the checksum calculation. The controller must perform the same checksum operation in transmitting checksum bytes.

Example) Assume the Show Flash Memory Sum command provides the upper and lower bytes of the sum as 0xE5 and 0xF6. To calculate the checksum for a series of 0xE5 and 0xF6:

Add the bytes together

0xE5 + 0xF6 = 0x1DB

Take the two's complement of the sum, and that is the checksum byte.

0 − 0xDB = 0x25

20.2.11 General Boot Program Flowchart

Figure 20-8 shows an overall flowchart of the boot program.

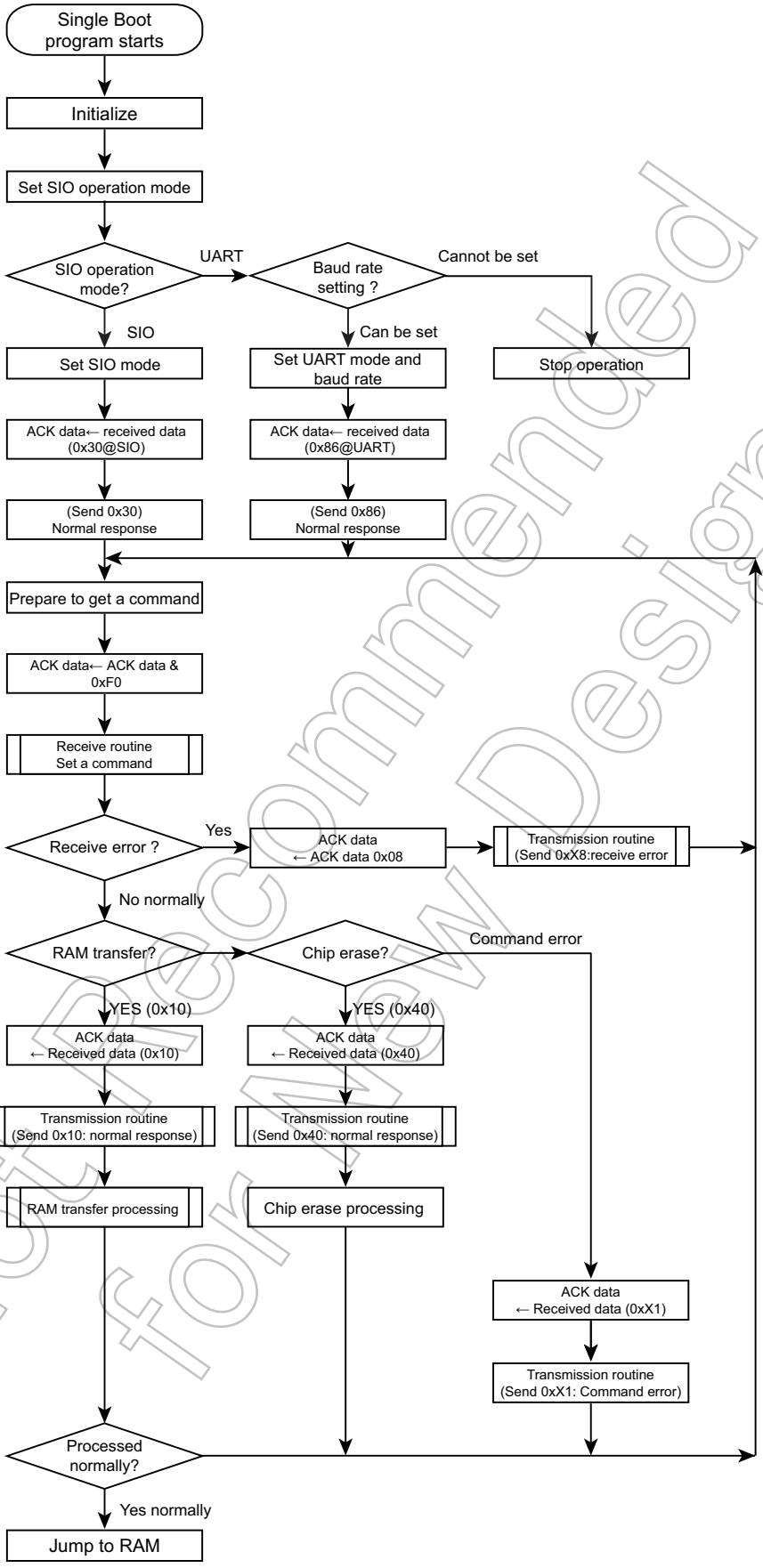


Figure 20-8 Overall Boot Program Flowchart

20.2.12 USB Boot

20.2.12.1 Boot Sequence

USB BOOT Sequence is shown as following table.

Table 20-12 USB Boot Sequence

USB BOOT Protocol		
PC		TMPM366FDXBG/FYXBG/ FWXBG
Start USB BOOT Protocol	→ ←	 Response
Send Password	→	
Confirm Password	→ ←	 Response
Send Boot Information	→	
Confirm Boot Information	→ ←	 Response
Plane Data	→	
Send Sum Data	→	
Confirm Sum Data	→ ←	 Response
Flash Erase Protocol		
Start Flash Erase Protocol	→ ←	 Response
Send Password	→	
Confirm Password	→ ←	 Response
Run Flash Erase	→ ←	 Response
Confirm Flash Erase	→ ←	 Response

20.2.12.2 USB Boot Command

USB Boot Command is shown as following table.

Table 20-13 Boot Command List

			Start USB Boot Protocol	Send Password	Confirm Password	Send Boot Information	Confirm Boot Information
bmRequestType	1byte	Vendor Class	in	out	in	out	in
bRequest	1byte	Command	0x18	0x20	0x28	0x30	0x38
wValue	2byte	0x0000	-	-	-	-	-
wIndex	2byte	Sequence ID	any data	Same as Start Protocol	Same as Start Protocol	Same as Start Protocol	Same as Start Protocol
wLength	2byte	Data Length	1	12	1	6	1
Data Stage	0 to 64 byte		0x18: OK 0x19: NG	Password[0]	0x28:OK 0x29:NG	RAM Address<31:24>	0x38:OK 0x39:NG
				Passwoed[1]		RAM Address<23:16>	
				Password[2]		RAM Address<15:8>	
				Password[3]		RAM Address<7:0>	
				Password[4]		Transfer Size<15:8>	
				Password[5]		Transfer Size<7:0>	
				Password[6]			
				Password[7]			
				Password[8]			
				Password[9]			
				Password[10]			
				Password[11]			

			Send Sum Data	Confirm Sum Data	Start Flash Erase Protocol	Run Flash Erase	Confirm Flash Erase
bmRequestType	1byte	Vendor Class	out	in	in	in	in
bRequest	1byte	Command	0x40	0x48	0x58	0x68	0x78
wValue	2byte	0x0000	-	-	-	-	-
wIndex	2byte	Sequence ID	Same as Start Protocol	Same as Start Protocol	any data	Same as Start Protocol	Same as Start Protocol
wLength	2byte	Data Length	1	1	1	1	1
Data Stage	0 - 64 byte		Sum Data	0x48: OK 0x49: NG	0x58:OK 0x59:NG	0x68: OK 0x69: NG	0x78:OK 0x79:NG

20.2.13 Descriptor

Descriptor in USB Boot Mode is shown as following table.

Table 20-14 Device Descriptor

Offset	Filed	value	Description
0	bLentgth	0x12	18 bytes
1	bDescripotType	0x01	Device descriptor
2	bcdUSB	0x00	USB version 2.0
3		0x02	
4	bDeviceClass	0x00	Device class (Not used)
5	bDeviceSubClass	0x00	Sub command (Not used)
6	bDeviceProtocol	0x00	Protocol (Not used)
7	bMaxPacketSize0	0x40	EP0 Maximum packet size 64 bytes
8	idVendor	0x30	Vender ID
9		0x09	
10	idProduct	0x69	Product ID
11		0x65	
12	bcdDevice	0x00	Device version
13		0x01	
14	iManufacture	0x00	String descriptor index shown as manufacturer
15	iProduct	0x00	String descriptor index shown as manufacturer
16	iSerialNumber	0x00	String descriptor index shown as manufacturer
17	bNumConfigurations	0x01	The number of configuration 1

Table 20-15 Configuration Descriptor

Offset	Filed	value	Description
0	bLentgth	0x09	9 bytes
1	bDescripotType	0x02	Configuration descripor
2	bTotal Lenght	0x20	The length which is added with configuration and end point (32 bytes)
3		0x00	
4	bNumInterfaces	0x01	The number of interface 1
5	bConfigurationValue	0x01	The number of configuration 1
6	iConfiguration	0x00	String descriptor index shown this configuration name (Not used)
7	bmAttributes	0x80	Bus power
8	MaxPower	0x31	Maximum power consumption (49mA)

Table 20-16 Interface Descriptor

Offset	Filed	value	Description
0	bLentgth	0x09	9 bytes
1	bDescripotType	0x04	Interface descriptor
2	bInterdfaceNumber	0x00	The number of interface 1
3	bAlternateSetting	0x00	The number of alternate setting 0
4	bNumEndpoints	0x02	Two end point
5	bInterfaceClass	0xFF	
6	bInterfaceSubClass	0x00	
7	bInterfaceProtocol	0x50	Bulk only protocol
8	iInterface	0x00	String descriptor index shown this interface name (Not used)

Table 20-17 Bulk-In Endpoint Descriptor

Offset	Filed	value	Description
0	bLentgth	0x07	7 bytes
1	bDescripotType	0x05	End point descriptor
2	bEndpointAddress	0x81	End point 1 is used as input
3	bmAttributes	0x02	Bulk transfer
4	wMaxPacketSize	0x40	
5		0x00	
6	bInterval	0x00	(Ignore because of bulk transfer)

Table 20-18 Bulk-Out Endpoint Descriptor

Offset	Filed	value	Description
0	bLentgth	0x07	7 bytes
1	bDescripotType	0x05	End point descriptor
2	bEndpointAddress	0x02	End point 2 is used as output
3	bmAttributes	0x02	Bulk transfer
4	wMaxPacketSize	0x40	Payload 64 bytes
5		0x00	
6	bInterval	0x00	(Ignore because of bulk transfer)

20.3 On-board Programming of Flash Memory (Rewrite/Erase)

In on-board programming, the CPU is to execute software commands for rewriting or erasing the flash memory. The rewrite/erase control program should be prepared by the user beforehand. Because the flash memory content cannot be read while it is being written or erased, it is necessary to run the rewrite/erase program from the internal RAM after shifting to the user boot mode.

20.3.1 Flash Memory

Except for some functions, writing and erasing flash memory data are in accordance with the standard JEDEC commands. In writing or erasing, use 32-bit data transfer command of the CPU to enter commands to the flash memory. Once the command is entered, the actual write or erase operation is automatically performed internally.

Table 20-19 Flash Memory Functions

Major functions	Description
Automatic page program	Writes data automatically per page.
Automatic chip erase	Erases the entire area of the flash memory automatically.
Automatic block erase	Erases a selected block automatically.
Protect function	The write or erase operation can be individually inhibited for each block.

20.3.1.1 Block Configuration

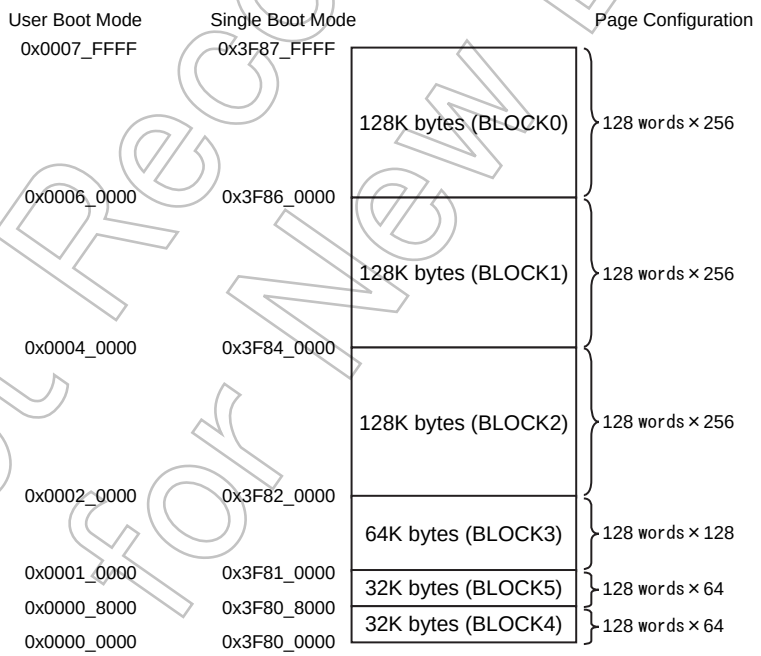


Figure 20-9 Block Configuration of Flash Memory

20.3.1.2 Basic operation

This flash memory device has the following two operation modes:

- The mode to read memory data (Read mode)
- The mode to automatically erase or rewrite memory data (Automatic operation)

Transition to the automatic mode is made by executing a command sequence while it is in the memory read mode. In the automatic operation mode, flash memory data cannot be read and any commands stored in the flash memory cannot be executed. In the automatic operation mode, any interrupt or exception generation cannot set the device to the read mode except when a hardware reset is generated. During automatic operation, be sure not to cause any exceptions other than reset and debug exceptions while a debug port is connected. Any exception generation cannot set the device to the read mode except when a hardware reset is generated.

(1) Read

When data is to be read, the flash memory must be set to the read mode. The flash memory will be set to the read mode immediately after power is applied, when CPU reset is removed, or when an automatic operation is normally terminated. In order to return to the read mode from other modes or after an automatic operation has been abnormally terminated, either the Read/reset command (a software command to be described later) or a hardware reset is used. The device must also be in the read mode when any command written on the flash memory is to be executed.

- Read/reset command and Read command (software reset)

  When a command that has not been completely written has to be canceled, the Read/reset command must be used. The Read command is used to return to the read mode after executing 32-bit data transfer command to write the data "0x0000_00F0" to an arbitrary address of the flash memory.
- With the Read/reset command, the device is returned to the read mode after completing the third bus write cycle.

(2) Command write

This flash memory uses the command control method. Commands are executed by executing a command sequence to the flash memory. The flash memory executes automatic operation commands according to the address and data combinations applied (refer to Command Sequence).

If it is desired to cancel a command write operation already in progress or when any incorrect command sequence has been entered, the Read/reset command is to be executed. Then, the flash memory will terminate the command execution and return to the read.

While commands are generally comprised of several bus cycles, the operation to apply 32-bit data transmit command to the flash memory is called "bus write cycle." The bus write cycles are to be in a specific sequential order and the flash memory will perform an automatic operation when the sequence of the bus write cycle data and address of a command write operation is in accordance with a predefined specific sequence. If any bus write cycle does not follow a predefined command write sequence, the flash memory will terminate the command execution and return to the read mode.

Note 1: **Command sequences are executed from outside the flash memory area.**

Note 2: **Each bus write cycle must be sequentially executed by 32-bit data transmit command. While a command sequence is being executed, access to the flash memory is prohibited. Also, don't generate any interrupt (except debug exceptions when a debug port is connected).If such an operation is made, it can result in an unexpected read access to the flash memory and the command sequencer may not be able to correctly recognize the command. While it could cause an abnormal termination of the command sequence, it is also possible that the written command is incorrectly recognized.**

Note 3: **For the command sequencer to recognize a command, the device must be in the read mode prior to executing the command. Be sure to check before the first bus write cycle**

that FCFLCS <RDY/BSY> is set to "1." It is recommended to subsequently execute a Read command.

Note 4: **Upon issuing a command, if any address or data is incorrectly written, be sure to perform a software reset to return to the read mode again.**

20.3.1.3 Reset (Hardware reset)

A hardware reset is used to cancel the operational mode set by the command write operation when forcibly termination during automatic programming/ erasing or abnormal termination during automatic operation.

The flash memory has a reset input as the memory block and it is connected to the CPU reset signal. Therefore, when the $\overline{\text{RESET}}$ input pin of this device is set to VIL or when the CPU is reset due to any overflow of the watch dog timer, the flash memory will return to the read mode terminating any automatic operation that may be in progress. It should also be noted that applying a hardware reset during an automatic operation can result in incorrect rewriting of data. In such a case, be sure to perform the rewriting again.

Refer to Section "20.2.1 Reset Operation" for CPU reset operations. After a given reset input, the CPU will read the reset vector data from the flash memory and starts operation after the reset is removed.

20.3.1.4 Commands

(1) Automatic Page Programming

Writing to a flash memory device is to make "1" data cells to "0" data cells. Any "0" data cell cannot be changed to a "1" data cell. For making "0" data cells to "1" data cells, it is necessary to perform an erase operation.

The automatic page programming function of this device writes data of each page. The TMPM366FDXBG/FYXBG/FWXBG contain 128 words in a page. A 128 word block is defined by a same [31:9] address and it starts from the address [8:0] = 0x00 and ends at the address [8:0] = 0x1FF. This programming unit is hereafter referred to as a "page".

Writing to data cells is automatically performed by an internal sequencer and no external control by the CPU is required. The state of automatic page programming (whether it is in writing operation or not) can be checked by FCFLCS [0] <RDY/BSY> .

Also, any new command sequence is not accepted while it is in the automatic page programming mode. If it is desired to interrupt the automatic page programming, use the hardware reset function. If the operation is stopped by a hardware reset operation, it is necessary to once erase the page and then perform the automatic page programming again because writing to the page has not been normally terminated.

The automatic page programming operation is allowed only once for a page already erased. No programming can be performed twice or more times irrespective of the data cell value whether it is "1" or "0." Note that rewriting to a page that has been once written requires execution of the automatic block erase or automatic chip erase command before executing the automatic page programming command again. Note that an attempt to rewrite a page two or more times without erasing the content can cause damages to the device.

No automatic verify operation is performed internally to the device. So, be sure to read the data programmed to confirm that it has been correctly written.

The automatic page programming operation starts when the third bus write cycle of the command cycle is completed. On and after the fifth bus write cycle, data will be written sequentially starting from the next address of the address specified in the fourth bus write cycle (in the fourth bus write cycle, the page top address will be command written) (32 bits of data is input at a time). Be sure to use the 32-bit data transfer command in writing commands on and after the fourth bus cycle. In this, any 32-bit data transfer commands shall not be placed across word boundary. On and after the

fifth bus write cycle, data is command written to the same page area. Even if it is desired to write the page only partially, it is required to perform the automatic page programming for the entire page. In this case, the address input for the fourth bus write cycle shall be set to the top address of the page. Be sure to perform command write operation with the input data set to "1" for the data cells not to be set to "0." For example, if the top address of a page is not to be written, set the input data of the fourth bus write cycle to 0xFFFFFFFF to command write the data.

Once the third bus cycle is executed, the automatic page programming is in operation. This condition can be checked by monitoring FCFLCS<RDY/BSY>. Any new command sequence is not accepted while it is in automatic page programming mode. If it is desired to stop operation, use the hardware reset function. Be careful in doing so because data cannot be written normally if the operation is interrupted. When a single page has been command written normally terminating the automatic page writing process, FCFLCS<RDY/BSY> is set to "1" and it returns to the read mode.

When multiple pages are to be written, it is necessary to execute the page programming command for each page because the number of pages to be written by a single execution of the automatic page program command is limited to only one page. It is not allowed for automatic page programming to process input data across pages.

Data cannot be written to a protected block. When automatic programming is finished, it automatically returns to the read mode. This condition can be checked by monitoring FCFLCS<RDY/BSY> . If automatic programming has failed, the flash memory is locked in the mode and will not return to the read mode. For returning to the read mode, it is necessary to execute hardware reset to reset the flash memory or the device. In this case, while writing to the address has failed, it is recommended not to use the device or not to use the block that includes the failed address.

Note: **Software reset becomes ineffective in bus write cycles on and after the fourth bus write cycle of the automatic page programming command.**

(2) Automatic chip erase

The automatic chip erase operation starts when the sixth bus write cycle of the command cycle is completed.

This condition can be checked by monitoring FCFLCS<RDY/BSY> . While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that data has been correctly erased. Any new command sequence is not accepted while it is in an automatic chip erase operation. If it is desired to stop operation, use the hardware reset function. If the operation is forced to stop, it is necessary to perform the automatic chip erase operation again because the data erasing operation has not been normally terminated.

Also, any protected blocks cannot be erased. If all the blocks are protected, the automatic chip erase operation will not be performed and it returns to the read mode after completing the sixth bus read cycle of the command sequence. When an automatic chip erase operation is normally terminated, it automatically returns to the read mode. If an automatic chip erase operation has failed, the flash memory is locked in the mode and will not return to the read mode.

For returning to the read mode, it is necessary to execute hardware reset to reset the device. In this case, the failed block cannot be detected. It is recommended not to use the device anymore or to identify the failed block by using the block erase function for not to use the identified block anymore.

(3) Automatic block erase (for each block)

The automatic block erase operation starts when the sixth bus write cycle of the command cycle is completed.

This status of the automatic block erase operation can be checked by monitoring FCFLCS <RDY/BSY> . While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that data has been correctly erased. Any new command sequence is not accepted while it is in an automatic block erase operation. If it is desired to stop operation, use the hardware reset function. In this case, it is necessary to perform the automatic block erase operation again because the data erasing operation has not been normally terminated.

Also, any protected blocks cannot be erased. If an automatic block erase operation has failed, the flash memory is locked in the mode and will not return to the read mode. In this case, execute hardware reset to reset the device.

(4) Automatic programming of protection bits (for each block)

This device is implemented with protection bits. This protection can be set for each block. See Table 20-24 for table of protection bit addresses. This device assigns 1 bit to 1 block as a protection bit. The applicable protection bit is specified by PBA in the seventh bus write cycle. By automatically programming the protection bits, write and/or erase functions can be inhibited (for protection) individually for each block. The protection status of each block can be checked by FCFLCS <BLPRO> to be described later. This status of the automatic programming operation to set protection bits can be checked by monitoring FCFLCS <RDY/BSY> . Any new command sequence is not accepted while automatic programming is in progress to program the protection bits. If it is desired to stop the programming operation, use the hardware reset function. In this case, it is necessary to perform the programming operation again because the protection bits may not have been correctly programmed. If all the protection bits have been programmed, all FCFLCS <BLPRO> are set to "1" indicating that it is in the protected state. This disables subsequent writing and erasing of all blocks.

Note: **Software reset is ineffective in the seventh bus write cycle of the automatic protection bit programming command. FCFLCS <RDY/BSY> turns to "0" after entering the seventh bus write cycle.**

(5) Automatic erasing of protection bits

Different results will be obtained when the automatic protection bit erase command is executed depending on the status of the protection bits and the security bits. It depends on the status of FCFLCS <BLPRO> whether all <BLPRO> are set to "1" or not if FCSECBIT<FCSECBIT> is 0x1. Be sure to check the value of FCFLCS <BLPRO> before executing the automatic protection bit erase command. See the chapter "ROM protection" for details.

- When all the FCFLCS <BLPRO> are set to "1" (all the protection bits are programmed):

  When the automatic protection bit erase command is command written, the flash memory is automatically initialized within the device. When the seventh bus write cycle is completed,the entire area of the flash memory data cells is erased and then the protection bits are erased. This operation can be checked by monitoring FCFLCS <RDY/BSY>. If the automatic operation to erase protection bits is normally terminated, FCFLCS will be set to "0x00000001".While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that it has been correctly erased. For returning to the read mode while the automatic operation after the seventh bus cycle is in progress, it is necessary to use the hardware reset to reset the device. If this is done, it is necessary to check the status of protection bits by FCFLCS <BLPRO> after retuning to the read mode and perform either the automatic protection bit erase, automatic chip erase, or automatic block erase operation, as appropriate.
- When FCFLCS <BLPRO> include "0" (not all the protection bits are programmed):

If the automatic protection bit is cleared to "0", the protection condition is canceled. With this device, protection bits can be programmed to an individual block and performed bit-erase operation in the four bits unit as shown in Table 20-25. The target bits are specified in the seventh bus write cycle.The protection status of each block can be checked by FCFLCS <BLPRO> to be described later. This status of the programming operation for automatic protection bits can be checked by monitoring FCFLCS <RDY/BSY>. When the automatic operation to erase protection bits is normally terminated, the protection bits of FCFLCS <BLPRO> selected for erasure are set to "0".

In any case, any new command sequence is not accepted while it is in an automatic operation to erase protection bits. If it is desired to stop the operation, use the hardware reset function. When the automatic operation to erase protection bits is normally terminated, it returns to the read mode.

Note: **The FCFLCS <RDY/BSY> bit is "0" while in automatic operation and it turns to "1" when the automatic operation is terminated.**

20.3.1.5 Flash control/ status register

Base Address = 0x41FF_F000

Register name		Address(Base+)
Reserved	-	0x0000 to 0x000F
Security bit register	FCSECBIT	0x0010
Reserved	-	0x0014 to 0x001F
Flash control register	FCFLCS	0x0020
Reserved	-	0x0024 to 0x0FFF

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

(1) FCSECBIT (Security bit register)

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

Bit	Bit Symbol	Type	Function
31-1	−	R	Read as 0.
0	SECBIT	R/W	Security bits 0:disabled 1:enabled

Note: **This register is initialized by cold reset or releasing STOP2 mode of standby mode.**

(2) FCFLCS (Flash 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	-	-	BLPRO5	BLPRO4	BLPRO3	BLPRO2	BLPRO1	BLPRO0
After reset	0	0	(Note2)	(Note2)	(Note2)	(Note2)	(Note2)	(Note2)
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	RDY/BSY
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-22	−	R	Read as 0.
21-16	BLPRO5 to BLPRO0	R	Protection for Block5 to 0 0: disabled 1: enabled Protection status bits Each of the protection bits represents the protection status of the corresponding block. When a bit is set to "1," it indicates that the block corresponding to the bit is protected. When the block is protected, data cannot be written to it.
15-1	−	R	Read as 0.
0	RDY/BSY	R	Ready/Busy (Note 1) 0: automatic operating 1:automatic operation terminated Ready/Busy flag bit The RDY/BSY output is provided as a means to monitor the status of automatic operation. This bit is a function bit for the CPU to monitor the function. When the flash memory is in automatic operation, it outputs "0" to indicate that it is busy. When the automatic operation is terminated, it returns to the ready state and outputs "1" to accept the next command. If the automatic operation has failed, this bit maintains the "0" output. By applying a hardware reset, it returns to "1."

Note 1: This command must be issued in the ready state. Issuing the command in the busy state may disable both correct command transmission and further command input. To exit from the condition, execute system reset. System reset requires at least 0.5 μs regardless of the system clock frequency. In this condition, it takes approx. 2 ms to enable reading after reset.

Note 2: The value varies depending on protection applied.

20.3.1.6 List of Command Sequences

Table 20-20 shows the addresses and the data of each command of flash memory.

Bus cycles are "bus write cycles" except for the second bus cycle of the Read command and the fourth bus cycle of the Read/reset command. Bus write cycles are executed by 32-bit (word) data transfer commands. (In the following table, only lower 8 bits data are shown.)

See Table 20-21 for the detail of the address bit configuration. Use a value of "Addr." in the Table 20-20 for the address [15:8] of the normal command in the Table 20-21.

Note: **Always set "0" to the address bits [1:0] in the entire bus cycle.**

Table 20-20 Flash Memory Access from the Internal CPU

Command sequence	First bus cycle	Second bus cycle	Third bus cycle	Fourth bus cycle	Fifth bus cycle	Sixth bus cycle	Seventh bus cycle
	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.
	Data	Data	Data	Data	Data	Data	Data
Read	0xXX	–	–	–	–	–	–
	0xF0	–	–	–	–	–	–
Read/Reset	0x54XX	0xAAXX	0x54XX	RA	–	–	–
	0xAA	0x55	0xF0	RD	–	–	–
Automatic page programming	0x54XX	0xAAXX	0x54XX	PA	PA	PA	PA
	0xAA	0x55	0xA0	PD0	PD1	PD2	PD3
Automatic chip erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	–
	0xAA	0x55	0x80	0xAA	0x55	0x10	–
Automatic block erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	BA	–
	0xAA	0x55	0x80	0xAA	0x55	0x30	–
Automatic protection bit programming	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x9A	0xAA	0x55	0x9A	0x9A
Automatic protection bit erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x6A	0xAA	0x55	0x6A	0x6A

Supplementary explanation

- RA: Read address
- RD: Read data
- PA: Program page address

  PD: Program data (32 bit data)

  After the fourth bus cycle, enter data in the order of the address for a page.
- BA: Block address
- PBA: Protection bit address

20.3.1.7 Address bit configuration for bus write cycles

Table 20-21 is used in conjunction with Table 20-20 "Flash Memory Access from the Internal CPU." Address setting can be performed according to the normal bus write cycle address configuration from the first bus cycle. "0" is recommended" in the Table 20-21 Address Bit Configuration for Bus Write Cycles can be changed as necessary.

Table 20-21 Address Bit Configuration for Bus Write Cycles

Address	Addr [31:19]	Addr [18]	Addr [17]	Addr [16]	Addr [15]	Addr [14]	Addr [13:11]	Addr [10]	Addr [9]	Addr [8]	Addr [7:0]

Table 20-22

Normal commands	Normal bus write cycle address configuration			
	Flash area	"0" is recommended.	Command	Addr[1:0]="0" (fixed) Others:0 (recommended)

Block erase	BA: Block address (Set the sixth bus write cycle address for block erase operation)				
	Block selection (Table 20-23)		Addr[1:0]="0" (fixed) , Others:0 (recommended)		
Automatic page programming	PA: Program page address (Set the fourth bus write cycle address for page programming operation)				
	Page address				Addr[1:0]="0" (fixed) Others:0 (recommended)
Protection bit programming	PBA: Protection bit address (Set the seventh bus write cycle address for protection bit programming				
	Flash area	Protection bit selection (Table 20-24)	Fixed to "0".	Protection bit selection (Table 20-24)	Addr[1:0]="0" (fixed) Others:0 (recommended)
Protection bit erase	PBA: Protection bit address (Set the seventh bus erase cycle address for protection bit erasure)				
	Flash area	Protection bit selection (Table 20-25)	Fixed to "0".	Protection bit selection (Table 20-25)	Addr[1:0]="0" (fixed) Others:0 (recommended)

As block address, specify any address in the block to be erased.

Table 20-23 Block Address Table

Block	Address (User boot mode)	Address (Single boot mode)	Size (Kbyte)
4	0x0000_0000 ~ 0x0000_7FFF	0x3F80_0000 ~ 0x3F80_7FFF	32
5	0x0000_8000 ~ 0x0000_FFFF	0x3F80_8000 ~ 0x3F80_FFFF	32
3	0x0001_0000 ~ 0x0001_FFFF	0x3F81_0000 ~ 0x3F81_FFFF	64
2	0x0002_0000 ~ 0x0003_FFFF	0x3F82_0000 ~ 0x3F83_FFFF	128
1	0x0004_0000 ~ 0x0005_FFFF	0x3F84_0000 ~ 0x3F85_FFFF	128
0	0x0006_0000 ~ 0x0007_FFFF	0x3F86_8000 ~ 0x3F87_FFFF	128

Note: **As for the addresses from the first to the fifth bus cycles, specify the upper addresses of the blocks to be erased.**

Table 20-24 Protection Bit Programming Address Table

Block	Protection bit	The seventh bus write cycle address					
		Address [18]	Address [17]	Address [16:11]	Address [10]	Address [9]	Address [8]
Block0	<BLPRO[0]>	0	0	Fixed to "0".	0	0	"0" is recommended.
Block1	<BLPRO[1]>	0	0		0	1	
Block2	<BLPRO[2]>	0	0		1	0	
Block3	<BLPRO[3]>	0	0		1	1	
Block4	<BLPRO[4]>	0	1		0	0	
Block5	<BLPRO[5]>	0	1		0	1	

Table 20-25 Protection Bit Erase Address Table

Block	Protection bit	The seventh bus write cycle address [18:17]	
		Address[18]	Address[17]
Block0 to 3	<BLPRO[0:3]>	0	0
Block4 to 5	<BLPRO[4:5]>	0	1

Note: **The protection bit erase command cannot erase by individual block.**

20.3.1.8 Flowchart

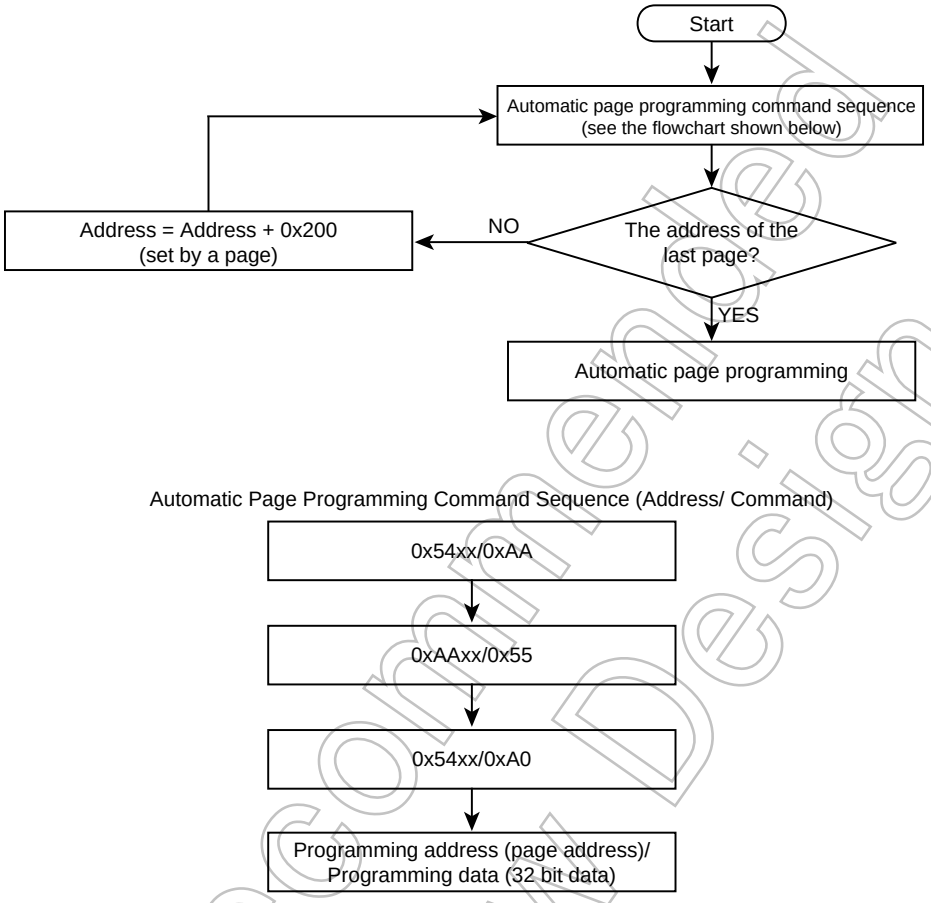


Figure 20-10 Automatic Programming

Note: Command sequence is executed by 0x54xx or 0x55xx.

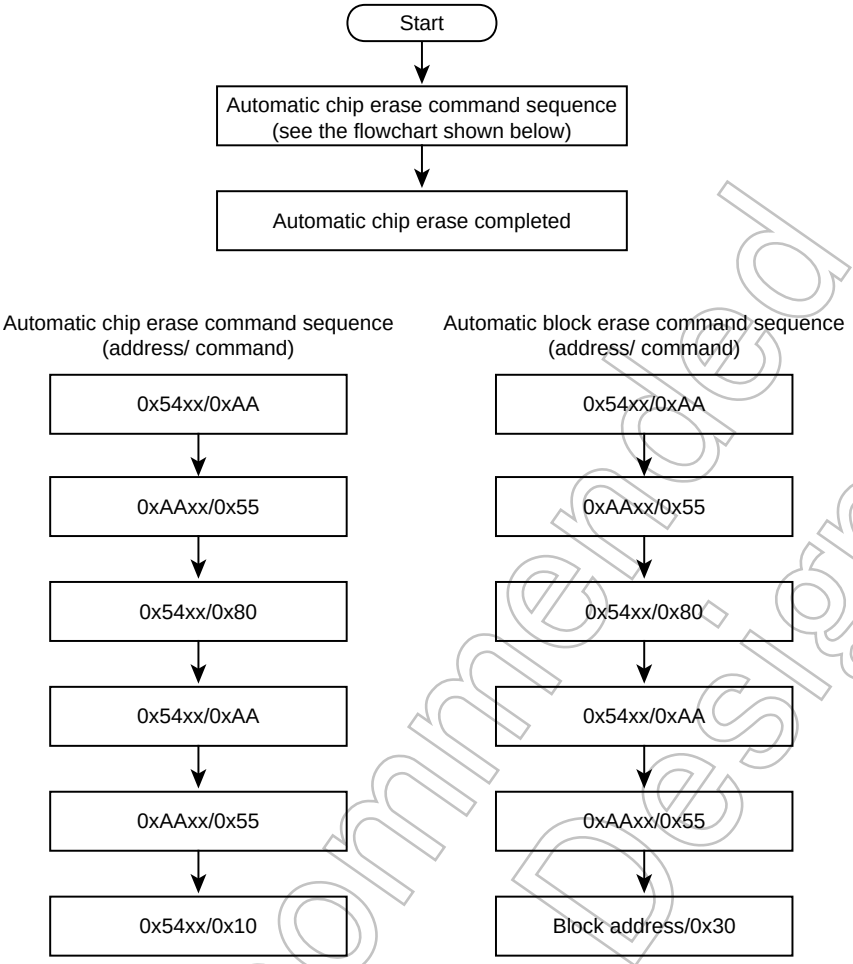


Figure 20-11 Automatic Erase

Note: **Command sequence is executed by 0x54xx or 0x55xx.**

21. ROM protection

21.1 Outline

The TMPM366FDXBG/FYXBG/FWXBG offers two kinds of ROM protection/ security functions.

One is a write/ erase-protection function for the internal flash ROM data.

The other is a security function that restricts internal flash ROM data readout and debugging.

21.2 Features

21.2.1 Write/ erase-protection function

The write/ erase-protection function enables the internal flash to prohibit the writing and erasing operation for each block.

To activate the function, write "1" to the corresponding bits to a block to protect. Writing "0" to the bits cancels the protection.

The protection settings of the bits can be monitored by the FCFLCS <BLPRO[5:0]> bit. See the chapter "Flash" for programming details.

21.2.2 Security function

The security function restricts flash ROM data readout and debugging.

This function is available under the conditions shown below.

1. The FCSECBIT <SECBIT> bit is set to"1".
2. All the protection bits (the FCFLCS<BLPRO> bits) used for the write/erase-protection function are set to "1".

Note: **The FCSECBIT <SECBIT> bit is set to "1" at a power-on reset right after power-on.**

Table 21-1 shows details of the restrictions by the security function.

Table 21-1 Restrictions by the security function

Item	Details
1) ROM data readout	Data can be read from CPU.
2) Debug port	Communication of JTAG/SW and trace are prohibited
3) Command for flash memory	Writing a command to the flash memory is prohibited. An attempt to erase the contents in the bits used for the write/ erase-protection erases all the protection bits.

21.3 Register

Base Address = 0x41FF_F000

Register name		Address(Base+)
Reserved	-	0x0000,0x0004
Security bit register	FCSECBIT	0x0010
Flash control register	FCFLCS	0x0020
Reserved	-	0x0024,0x0028

Note: **Access to the "Reserved" area is prohibited.**

21.3.1 FCFLCS (Flash 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	-	-	BLPRO5	BLPRO4	BLPRO3	BLPRO2	BLPRO1	BLPRO0
After reset	0	0	(Note2)	(Note2)	(Note2)	(Note2)	(Note2)	(Note2)
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	RDY/BSY
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-22	−	R	Read as 0.
21-16	BLPRO5 to BLPRO0	R	Protection for Block5 to 0 0: disabled 1: enabled Protection status bits Each of the protection bits represents the protection status of the corresponding block. When a bit is set to "1," it indicates that the block corresponding to the bit is protected. When the block is protected, data cannot be written to it.
15-1	−	R	Read as 0.
0	RDY/BSY	R	Ready/Busy (Note 1) 0: Auto operating 1:Auto operation terminated Ready/Busy flag bit The RDY/BSY output is provided as a means to monitor the status of automatic operation. This bit is a function bit for the CPU to monitor the function. When the flash memory is in automatic operation, it outputs "0" to indicate that it is busy. When the automatic operation is terminated, it returns to the ready state and outputs "1" to accept the next command. If the automatic operation has failed, this bit maintains the "0" output. By applying a hardware reset, it returns to "1."

Note 1: **This command must be issued in the ready state. Issuing the command in the busy state may disable both correct command transmission and further command input. To exit from the condition, execute system reset. System reset requires at least 0.5 ms regardless of the system clock frequency. In this condition, it takes approx. 2 ms to enable reading after reset.**

Note 2: **The value varies depending on protection applied.**

21.3.2 FCSECBIT(Security bit register)

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

Bit	Bit Symbol	Type	Function
31-1	−	R	Read as 0.
0	SECBIT	R/W	Security bit 0: Disabled 1: Enabled

Note: This register is initialized by cold reset and releasing STOP2 mode of the standby mode.

21.4 Writing and erasing

Writing and erasing protection bits are available with a single chip mode, single boot mode and writer mode.

21.4.1 Protection bits

Writing to the protection bits is done on block-by-block basis.

When the settings for all the blocks are "1", erasing must be done after setting the FCSECBIT <SECBIT> bit to "0". Setting "1" at that situation erases all the protection bits. To write and erase the protection bits, command sequence is used.

See the capter "Flash" for details

21.4.2 Security bit

The FCSECBIT <SECBIT> bit that activates security function is set to "1" at a power-on reset right after power-on.

The bit is rewritten by the following procedure.

1. Write the code 0xa74a9d23 to FCSECBIT register.
2. Write data within 16 clocks from the above.1.

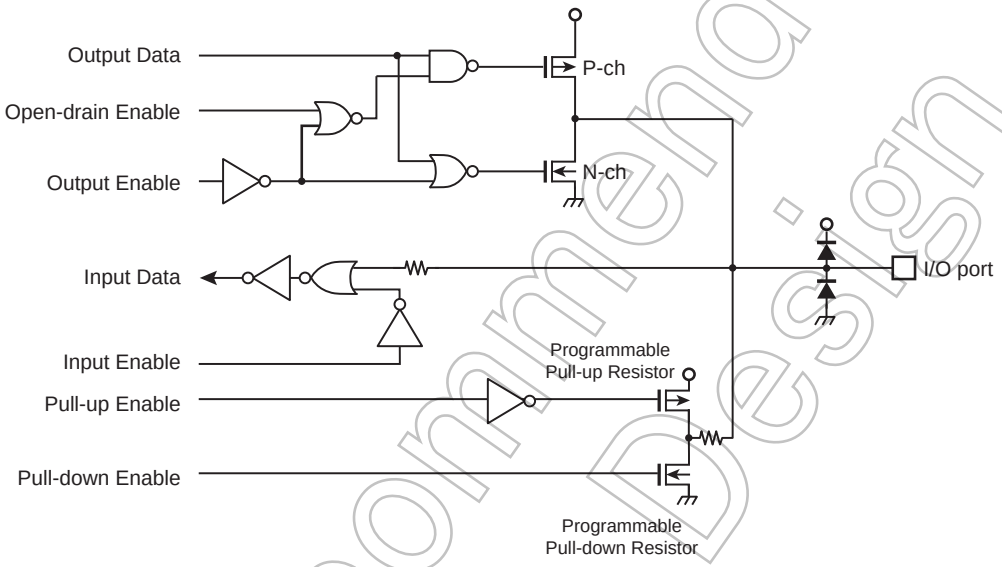
Note: The above procedure is enabled only when using 32-bit data transfer command.

22. Port Section Equivalent Circuit Schematic

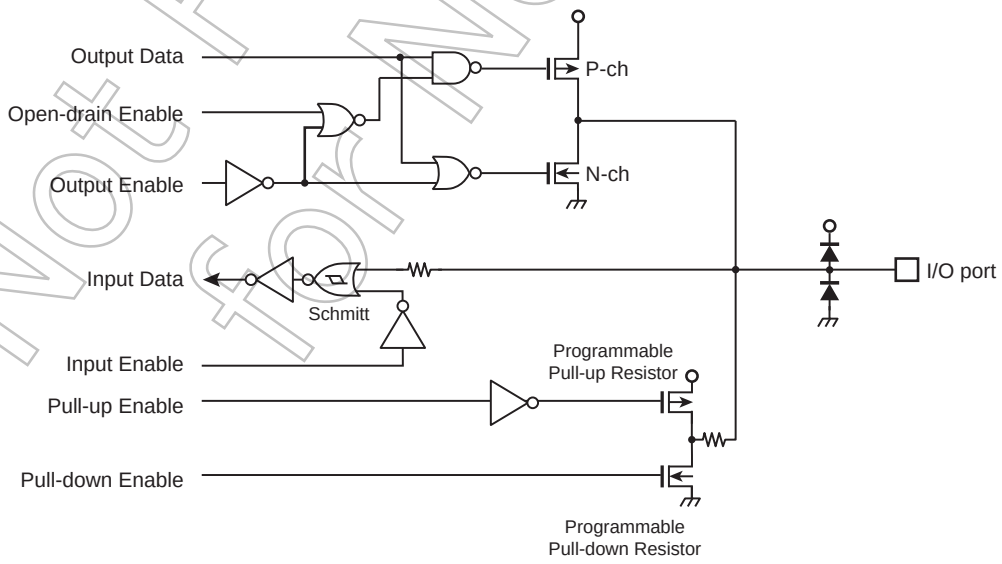
Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

The input protection resistance ranges from several tens of Ω to several hundreds of Ω. Feedback resistor and Damping resistor are shown with a typical value.

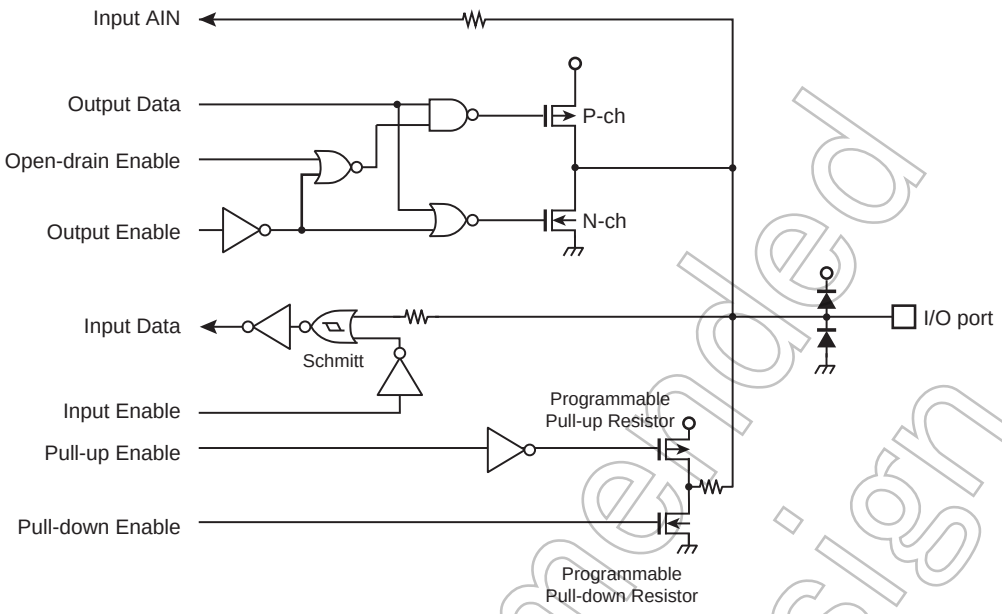
22.1 PA0 to 7,PB0 to 7



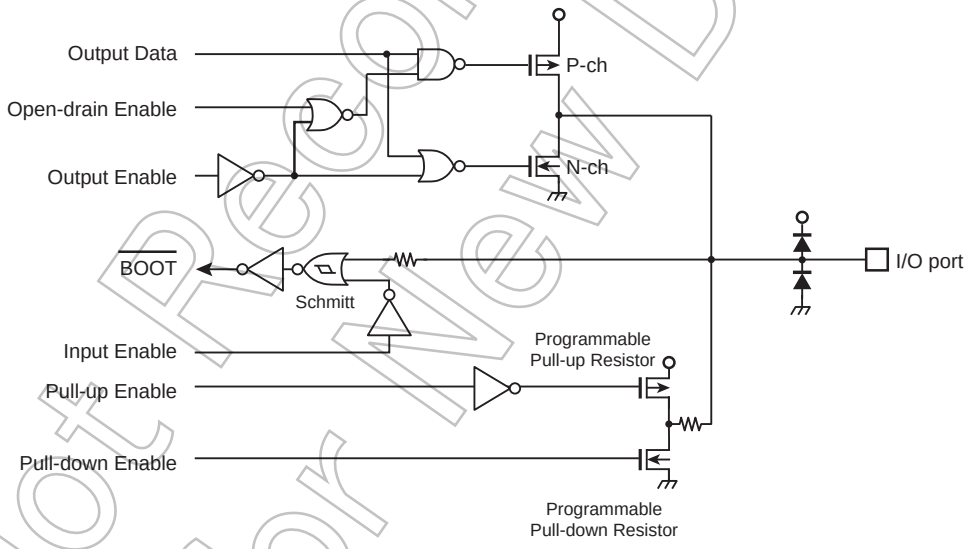
22.2 PC0 to 2, PD0 to 7, PE0 to 7, PF1 to 7, PG0 to 4, PH0 to 4, PI0 to 7



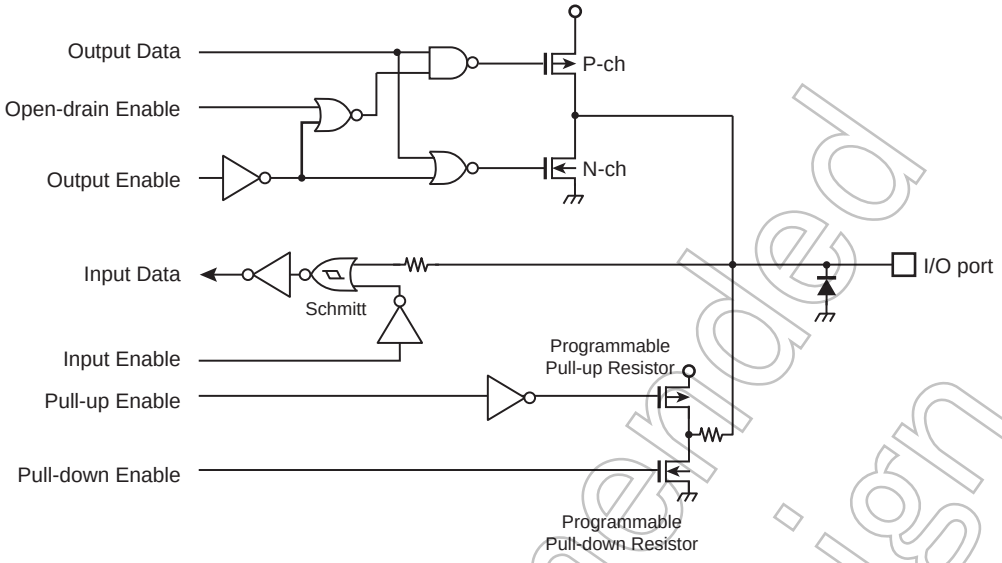
22.3 PJ0 to 7, PK0 to 3



22.4 PF0

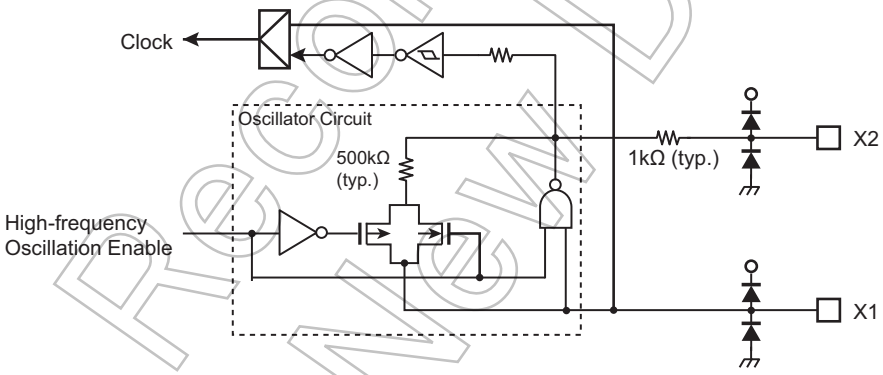


22.5 PG5



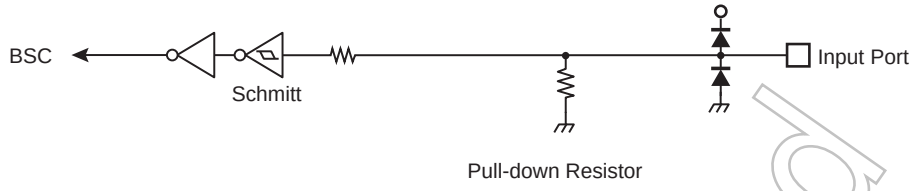
Note: Only when input is enabled, this pin tolerates 5V input.

22.6 X1,X2



22.7 $\overline{\text{RESET}}$,$\overline{\text{NMI}}$

22.8 BSC



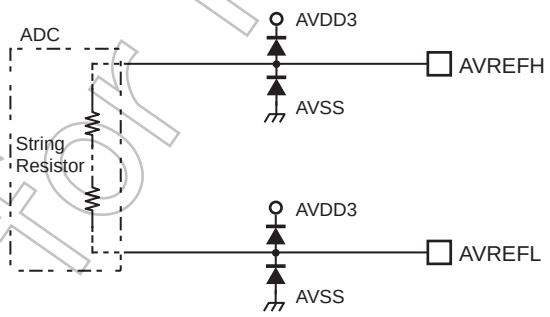
22.9 MODE

(Note)MODE pin is fixed to GND.

22.10 FTEST3

(Note)FTEST3 pin is fixed to Open.

22.11 AVREFH,AVREFL



23. Electrical Characteristics

23.1 Absolute Maximum Ratings

Parameter		Symbol	Rating	Unit
Supply voltage		DVDD3A	−0.3 to 3.9	V
		DVDD3C	−0.3 to 3.9	
		AVDD3	−0.3 to 3.9	
		RVDD3	−0.3 to 3.9	
Input voltage	Except below pins	V_{IN1}	−0.3 to VDD + 0.3	V
	PG5	V_{IN2}	−0.3 to 5.5	
Low-level output current	Per pin	I_{OL}	5	mA
	Total	ΣI_{OL}	50	
High-level output current	Per pin	I_{OH}	−5	
	Total	ΣI_{OH}	-50	
Power consumption (Ta = 85 °C)		PD	600	mW
Soldering temperature(10 s)		T_{SOLDER}	260	°C
Storage temperature		T_{STG}	−40 to 125	°C
Operating Temperature	Except during Flash W/E	T_{OPR}	−40 to 85	°C
	During Flash W/E		0 to 70	

Note: **Absolute maximum ratings are limiting values of operating and environmental conditions which should not be exceeded under the worst possible conditions. The equipment manufacturer should design so that no Absolute maximum rating value is exceeded with respect to current, voltage, power consumption, temperature, etc. Exposure to conditions beyond those listed above may cause permanent damage to the device or affect device reliability, which could increase potential risks of personal injury due to IC blowup and/or burning.**

23.2 DC Electrical Characteristics (1/3)

Ta = −40 to 85 °C

Parameter		Symbol	Condition		Min.	Typ. (Note 1)	Max.	Unit
Supply voltage	DVDD3A DVDD3C AVDD3 RVDD3 DVSSA = DVSSB = AVSS = RVSS = DVSSC = 0V	DVDD3A DVDD3C AVDD3 RVDD3 (Note 2)	f_{OSC} = 8 to 16 MHz fsys = 1 to 48 MHz	without USB	2.7	−	3.6	V
				with USB	3.0	−	3.45	V
Low-level input voltage	PA, PB, PC, PD, PE, PF PG, PH, PI, PJ, PK, $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC	V_{IL1}	2.7 V ≤ DVDD3A ≤ 3.6 V (Include 5V tolerant input pin)		−0.3	−	0.2 DVDD3A	V
	X1	V_{IL4}	2.7 V ≤ RVDD3 ≤ 3.6 V				0.2 RVDD3	
High-level input voltage	PA, PB, PC, PD, PE, PF PG4 to 0, PH, PI, PJ, PK, $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC	V_{IL1}	2.7 V ≤ DVDD3A ≤ 3.6 V		0.8 DVDD3A	−	DVDD3A+0.3	V
	PG5	V_{IL3}					5.5	
	X1	V_{IL2}	2.7 V ≤ RVDD3 ≤ 3.6 V		0.8 RVDD3		RVDD3 + 0.3	
Low-level output voltage		V_{OL}	I_{OL} = 2 mA, 2.7 V ≤ DVDD3A ≤ 3.6 V		−	−	0.4	V
High-level output voltage	PA, PB, PC, PD, PE, PF PG4 to 0, PH, PI, PJ, PK	V_{OH}	I_{OH} = −2 mA, 2.7 V ≤ DVDD3A ≤ 3.6 V		2.4	−	DVDD3A	V
	PG5				2.4	−	DVDD3A	
Input leakage current		I_{LI1}	0.0 ≤ V_{IN} ≤ DVDD3A 0.0 ≤ V_{IN} ≤ AVDD3		−	0.02	±5	μA
Output leakage current		I_{LO}	0.2 ≤ V_{IN} ≤ DVDD3A − 0.2 0.2 ≤ V_{IN} ≤ AVDD3 − 0.2		−	0.05	±10	
Pull-up resister at Reset		RRST	2.7 V ≤ DVDD3A ≤ 3.6 V		−	50	150	kΩ
Hysteresis voltage		VTH1	2.7 V ≤ DVDD3A ≤ 3.6 V		0.3	0.6	−	V
Programmable pull-up/pull-down resistor		PKH	2.7 V ≤ DVDD3A ≤ 3.6 V		−	50	150	kΩ
Pin capacitance (Except power supply pins)		C_{IO}	fc = 1 MHz		−	−	10	pF

Note 1: **Ta = 25 ×C, DVDD3A = DVDD3C = RVDD3 = AVDD3 = 3.3 V, unless otherwise noted.**

Note 2: **The same voltage must be supplied to DVDD3A, AVDD3, and RVDD3.**

Note 3: **Ensure that all power suply source, including DVDD3C, is power-off and then power-on again when DVDD3A, RVDD3 and AVDD3 falls below 2.7V which is minimum operation voltage.**

23.3 DC Electrical Characteristics (2/3)

DVDD3A = DVDD3C = RVDD3 = AVDD3 = 2.7 V to 3.6 V
DVSSA = DVSS3C = RVSS = AVSS,
Ta = −40 to 85 °C

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Low-level output current	I_{OL1}	Per pin 2.7 V ≤ DVDD3A ≤ 3.6 V Except PG5	−	−	2	mA
	I_{OL2}	Per pin 2.7 V ≤ DVDD3A ≤ 3.6 V PG5	−	−	2	
	ΣI_{OL3}	Per group, Port A	−	−	10	
	ΣI_{OL4}	Per group, Port B	−	−	10	
	ΣI_{OL5}	Per group, Port C	−	−	10	
	ΣI_{OL6}	Per group, Port D	−	−	10	
	ΣI_{OL7}	Per group, Port E	−	−	10	
	ΣI_{OL8}	Per group, Port F	−	−	10	
	ΣI_{OL9}	Per group, PortG	−	−	20	
	ΣI_{OL10}	Per group, Port H	−	−	10	
	ΣI_{OL11}	Per group, Port I	−	−	10	
	ΣI_{OL12}	Per group, Port J	−	−	10	
	ΣI_{OL13}	Per group, Port K	−	−	10	
	ΣI_{OL}	Total, all Port	−	−	35	mA
High-level output current	I_{OH1}	Per pin 2.7 V ≤ DVDD3A ≤ 3.6 V Except PG5	−	−	-2	mA
	I_{OH2}	Per pin 2.7 V ≤ DVDD3A ≤ 3.6 V PG5	−	−	-2	
	ΣI_{OH3}	Per group, Port A	−	−	-10	
	ΣI_{OH4}	Per group, Port B	−	−	-10	
	ΣI_{OH5}	Per group, Port C	−	−	-10	
	ΣI_{OH6}	Per group, Port D	−	−	-10	
	ΣI_{OH7}	Per group, Port E	−	−	-10	
	ΣI_{OH8}	Per group, Port F	−	−	-10	
	ΣI_{OH9}	Per group, Port G	−	−	-10	
	ΣI_{OH10}	Per group, Port H	−	−	-10	
	ΣI_{OH11}	Per group, Port I	−	−	-10	
	ΣI_{OH12}	Per group, Port J	−	−	-10	
	ΣI_{OH13}	Per group, Port K	−	−	-10	
	ΣI_{OH}	Total, all Port	−	−	-35	mA

Note: **The same voltage must be supplied to DVDD3A, AVDD3, and RVDD3.**

23.4 DC Electrical Characteristics (3/3)

DVDD3A = DVDD3C = AVDD3 = RVDD3 = 2.7 V to 3.6 V,
Ta = −40 to 85 °C

Parameter	Symbol	Condition	Min.	Typ. (Note1)	Max.	Unit
NORMAL (Note2) Gear 1/1	I_{DD}	fsys = 48 MHz USB enabled	−	36	50	mA
NORMALGear 1/1		fsys = 48 MHz USB disabled	−	34	46	
IDLE (Note3)			−	21	25	
STOP1		−	−	80	1200	µA
STOP2		−	−	3.5	50	

Note 1: **Ta = 25 °C, DVDD3A = DVDD3C = AVDD3 = RVDD3 = 3.3 V, unless otherwise noted.**

Note 2: **I_{DD} NORMAL: Measurement condition**
Operation program : The dhrystone ver. 2.1 operated in FLASH
Operation peripheral : 16-bit timer / event counter, ADC, serial channel, serial bus interface, asynchronous serial channel, USB Device controller (EP1, 3, 5 and 7 operated)

Note 3: **I_{DD} IDLE: Measurement condition**
Operation peripheral : All peripheral stopped
The currents flow through DVDD3A, DVDD3C, AVDD3 and RVDD3 are included.

23.5 12-bit ADC Electrical Characteristics

DVDD3A = DVDD3C = AVDD3 = RVDD3 = AVREFH = 2.7 V to 3.6 V
AVSS = DVSSA = DVSSC = RVSS = AVREFL = 0
Ta = −40 to 85 °C

Parameter		Symbol	Condition	Min.	Typ.	Max	Unit
Analog reference voltage(+)		AVREFH	−	2.7	3.3	3.6	V
Analog input voltage		VAIN	−	AVSS	−	VREFH	V
Power supply current of analog reference voltage	AD conversion	IREF	DVSSA = DVSSC = AVSS	−	1.5	2.0	mA
	Non-AD conversion			−	0.02	0.1	μA
INL error		−	AIN resistance ≤ 600 Ω AIN load capacitance ≥ 0.1 μF Conversion time ≥ 1.0 μs	−	4	6	LSB
DNL error				−	2	6	
Offset error				−	3	6	
Full-scale error				−	3	6	
INL error		−	AIN resistance ≤ 600 Ω AIN load capacitance ≥ 33 pF Conversion time ≥ 1.66 μs	−	3	6	
DNL error				−	2	6	
Offset error				−	4	6	
Full-scale error				−	2	6	
Conversion time		Tconv	−	1.0	−	10	μs

Note 1: **1LSB = (AVREFH − AVSS)/4096 [V]**
Note 2: **Peripheral functions are disable.**

23.6 AC Electrical Characteristics

23.6.1 AC measurement condition

The AC characteristics data of this chapter is measured under the following conditions unless otherwise noted

- Output levels: High = 0.8 × DVDD3A, 0.8 × DVDD3C
- Output levels: Low = 0.2 × DVDD3A, 0.2 × DVDD3C
- Input levels: Refer to low-level input voltage and high-level input voltage in "DC Electrical Characteristics".
- Load capacity: CL = 30pF

23.6.2 Serial Channel (SIO/UART)

23.6.2.1 I/O Interface mode

In the table below, the letter x represents the SIO operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

(1) SCLK input mode

[Data input]

Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max	Min.	Max	Min.	Max	
SCLK Clock High width (input)	t_{SCH}	4x	–	100	–	83.3	–	ns
SCLK Clock Low width (input)	t_{SCL}	4x	–	100	–	83.3	–	
SCLK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	–	200	–	166.6	–	
Valid Data input → SCLK rise or fall(Note1)	t_{SRD}	30	–	30.0	–	30.0	–	
SCLK rise or fall(Note1) → Input Data hold	t_{HSR}	x + 30	–	55.0	–	50.8	–	

[Data output]

Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
SCLK Clock High width (input)	t_{SCH}	4x	–	120 (Note3)	–	107.5 (Note3)	–	ns
SCLK Clock Low width (input)	t_{SCL}	4x	–	120 (Note3)	–	107.5 (Note3)	–	
SCLK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	–	240	–	215	–	
Output Data → SCLK rise or fall (Note 1)	t_{OSS}	$t_{SCY}/2 - 3x - 45$	–	0.00 (Note2)	–	0.00 (Note2)	–	
SCLK rise or fall (Note1) → Output Data hold	t_{OHS}	$t_{SCY}/2$	–	120	–	107.5	–	

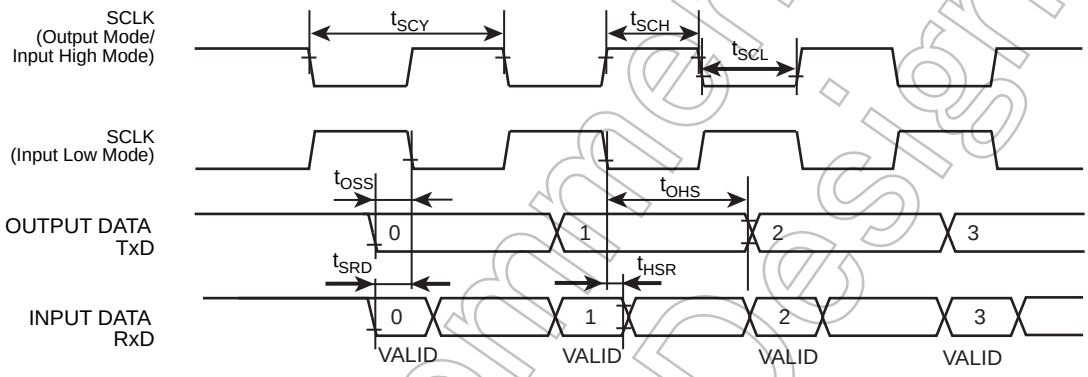
Note 1: **SCLK rise/fall : SCLK rise mode uses the rise timing of SCLK. SCLK fall mode uses the fall timing of SCLK.**

Note 2: **Use the frequency of SCLK in a range where the calculation value keeps positive.**

Note 3: The value indicates a minimum value that enables t_{OSS} to be zero or more.

(2) SCLK output mode

Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
SCLK cycle (programmable)	t_{SCY}	4x	−	100	−	83.3	−	ns
Output Data → SCLK rise	t_{OSS}	$t_{SCY}/2 - 30$	−	20	−	11.7	−	
SCLK rise → Output hold Data hold	t_{OHS}	$t_{SCY}/2 - 30$	−	20	−	11.7	−	
Valid Data Input → SCLK rise	t_{SRD}	45	−	45	−	45	−	
SCLK rise → Input Data hold	t_{HSR}	0	−	0	−	0	−	



23.6.3 Serial Bus Interface (I2C/SIO)

23.6.3.1 I2C Mode

In the table below, the letter x represents the I2C operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

n denotes the value of n programmed into the SCK (SCL output frequency select) field in the SBIxCR.

Parameter	Symbol	Equation		Standard Mode		Fast Mode		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
SCL clock frequency	t_{SCL}	0	−	0	100	0	400	kHz
Hold time for 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	$t_{SU;\ STA}$	(Note5)	−	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}	(Note5)	−	4.7	−	1.3	−	μs

Note 1: **SCL clock Low width (output) = $(2^{n-1} + 58)/x$**

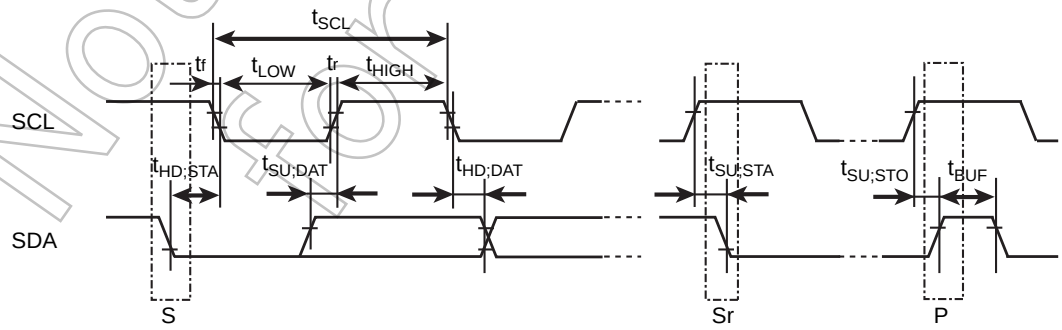
Note 2: **SCL clock High width (output) = $(2^{n-1} + 14)/x$ On I2C-bus specification, Maximum Speed of Standard Mode is 100kHz, Fast mode is 400khz. Internal SCL Frequency setting should comply with Note1 & Note2 shown above.**

Note 3: **The output data hold time is equal to 4x of internal SCL.**

Note 4: **The Philips I2C-bus specification states that a device must internally provide a hold time of at least 300 ns for the SDA signal to bridge the undefined region of the falling edge of SCL. However, this SBI does not satisfy this requirement. Also, the output buffer for SCL does not incorporate slope control of the falling edges; therefore, the equipment manufacturer should design so that the input data hold time shown in the table is satisfied, including tr/tf of the SCL and SDA lines.**

Note 5: **Software -dependent**

Note 6: **The Philips I2C-bus specification instructs that if the power supply to a Fast-mode device is switched off, the SDA and SCL I/O pins must be floating so that they don't obstruct the bus lines. However, this SBI does not satisfy this requirement.**



S: Start condition
Sr: Repeated start condition
P: Stop condition

23.6.3.2 Clock-Synchronous 8-Bit SIO mode

In the table below, the letter x represents the I2C operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

(1) SCK Input Mode (The electrical specifications below are for an SCK signal with a 50% duty cycle.)

[Data input]

Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
SCK Clock High width (input)	t_{SCH}	4x	−	100	−	83.3	−	ns
SCK Clock Low width (input)	t_{SCL}	4x	−	100	−	83.3	−	
SCK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	−	200	−	166	−	
Valid Data input → SCK rise	t_{SRD}	30 − x	−	5	−	9	−	
SCK rise → Input Data hold	t_{HSR}	2x + 30	−	80	−	71.7	−	

[Data output]

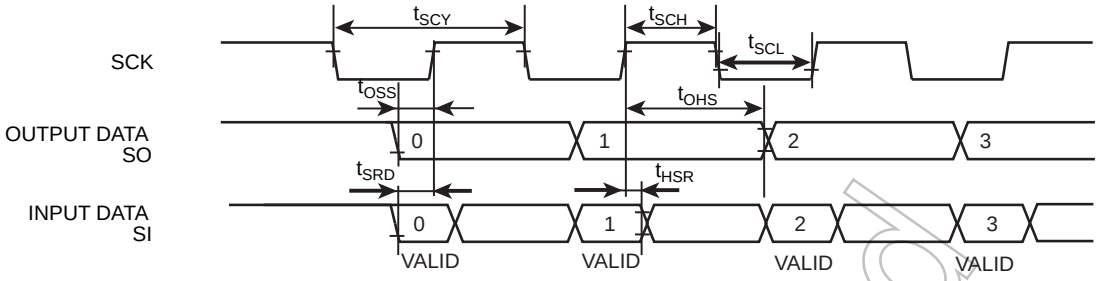
Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
SCK Clock High width (input)	t_{SCH}	4x	−	120 (Note2)	−	108 (Note2)	−	ns
SCK Clock Low width (input)	t_{SCL}	4x	−	120 (Note2)	−	108 (Note2)	−	
SCK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	−	240	−	215	−	
Output Data → SCK rise	t_{OSS}	$t_{SCY}/2 - 3x - 45$	−	0 (Note1)	−	0 (Note1)	−	
SCK rise → Output Data hold	t_{OHS}	$t_{SCY}/2 + x$	−	145	−	128	−	

Note 1: **Use the frequency of SCK in a range where the calculation value keeps positive.**
Note 2: The value indicates a minimum value that enables t_{OSS} to be zero or more.

(2) SCK Output Mode (The electrical specifications below are for an SCK signal with a 50% duty cycle.)

Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
SCK cycle (programmable)	t_{SCY}	16x	−	400	−	333	−	ns
Output Data → SCK rise	t_{OSS}	$t_{SCY}/2 - 20$	−	180	−	147	−	
SCK rise → Output Data hold	t_{OHS}	$t_{SCY}/2 - 20$	−	180	−	147	−	
Valid Data input → SCK rise	t_{SRD}	x + 45	−	70	−	65.8	−	
SCK rise → Input Data hold	t_{HSR}	0	−	0	−	0	−	

Note 1: SCK cycle after automatic wait becomes 14x.
Note 2: t_{OSS} after automatic wait may be $t_{SCY}/2$-x-20.



23.6.4 Synchronous serial Interface (SSP)

23.6.4.1 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 × DVDD3A, Low = 0.3 × DVDD3A
- Input levels: High = 0.9 × DVDD3A, Low = 0.1 × DVDD3A

Note: The "Equation" column in the table shows the specifications under the conditions DVDD3A = 2.7V to 3.6V.

Parameter	Symbol	Equation		fsys=40MHz (m=4, n=12)	fsys=48MHz (m=4, n=12)	Unit
		Min.	Max.			
SPxCLK Period (Master)	T_m	(m)T However more than 50ns	–	100 (10MHz)	83.3 (12MHz)	ns
SPxCLK Period (Slave)	T_s	(n)T	–	300 (3.3MHz)	250 (4MHz)	
SPxCLK rise up time	t_r	–	15	15	15	
SPxCLK fall down time	t_f	–	15	15	15	
Master mode: SPxCLK low level pulse width	t_{WLM}	(m)T/2 - 15	–	35	26.7	
Master mode: SPxCLK high level pulse width	t_{WHM}	(m)T/2 - 15	–	35	26.7	
Slave mode: SPxCLK low level pulse width	t_{WLS}	(n)T/2 - 15	–	135	110	
Slave mode: SPxCLK high level pulse width	t_{WHS}	(n)T/2 - 15	–	135	110	
Master mode: SPxCLK rise/fall → output data valid	t_{ODSM}	–	15	15	15	
Master mode: SPxCLK rise/fall → output data hold	t_{ODHM}	(m)T/2 - 13	–	35	28.7	
Master mode: SPxCLK rise/fall → input data valid delay time	t_{IDSM}	30	–	30	30	
Master mode: SPxCLK rise/fall → input data hold	t_{IDHM}	5	–	5	5	
Master mode: SPxFSS valid → SPxCLK rise/fall	t_{OFSM}	(m)T - 15	(m)T + 15	85 to 115	68.3 to 98.3	
Slave mode: SPxCLK rise/fall → output data valid delay time	t_{ODSS}	–	(3T) + 35	110	97.5	
Slave mode: SPxCLK rise/fall → output data hold	t_{ODHS} (Note1)	(n)T/2 + (2T)	–	200	166.7	
Slave mode: SPxCLK rise/fall → input data valid delay time	t_{IDSS}	10	–	10	10	
Slave mode: SPxCLK rise/fall → input data hold	t_{IDHS}	(3T) + 10	–	85	72.5	
Slave mode: SPxFSS valid → SPxCLK rise/fall	t_{OFSS}	(n)T + 5	–	305	255	

Note: Baud rate clock is set under below condition

Master Mode

$m=(<CPSDVSR>\times(1+<SCR>))= fsys /f_{SPxCLK}$ <CPSDVR> is set only even number. 65024≥ m ≥2

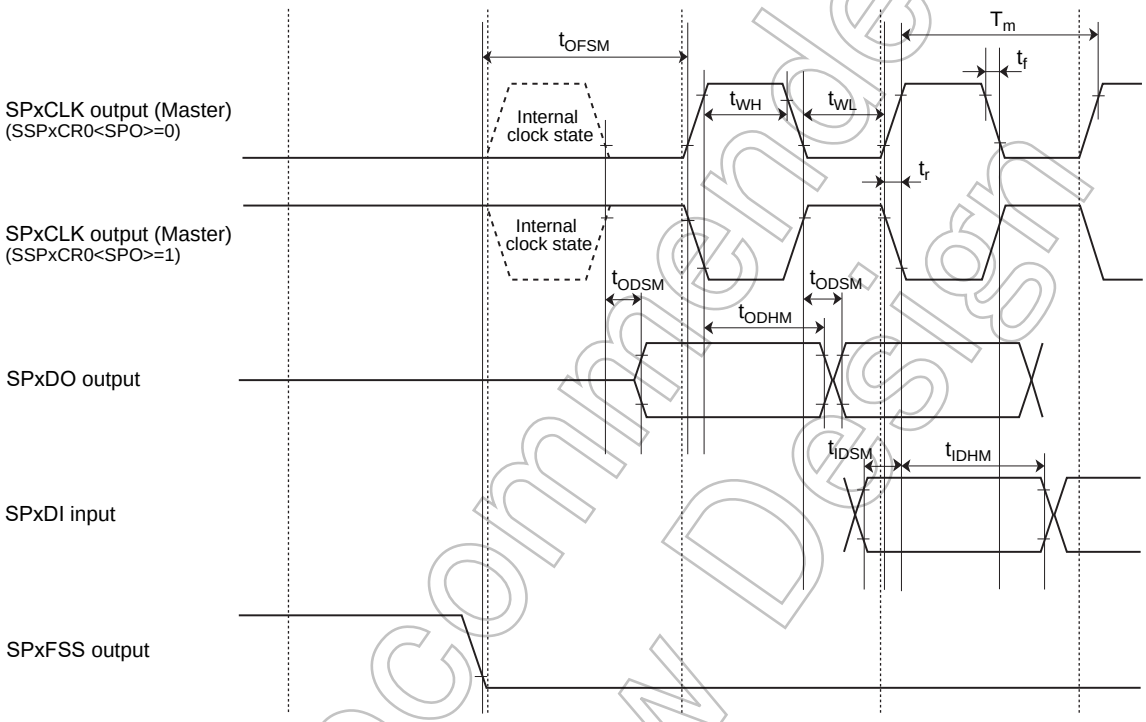
Slave mode

$n=(<CPSDVSR>\times(1+<SCR>))= fsys /f_{SPxCLK}$ 65024≥ n≥12

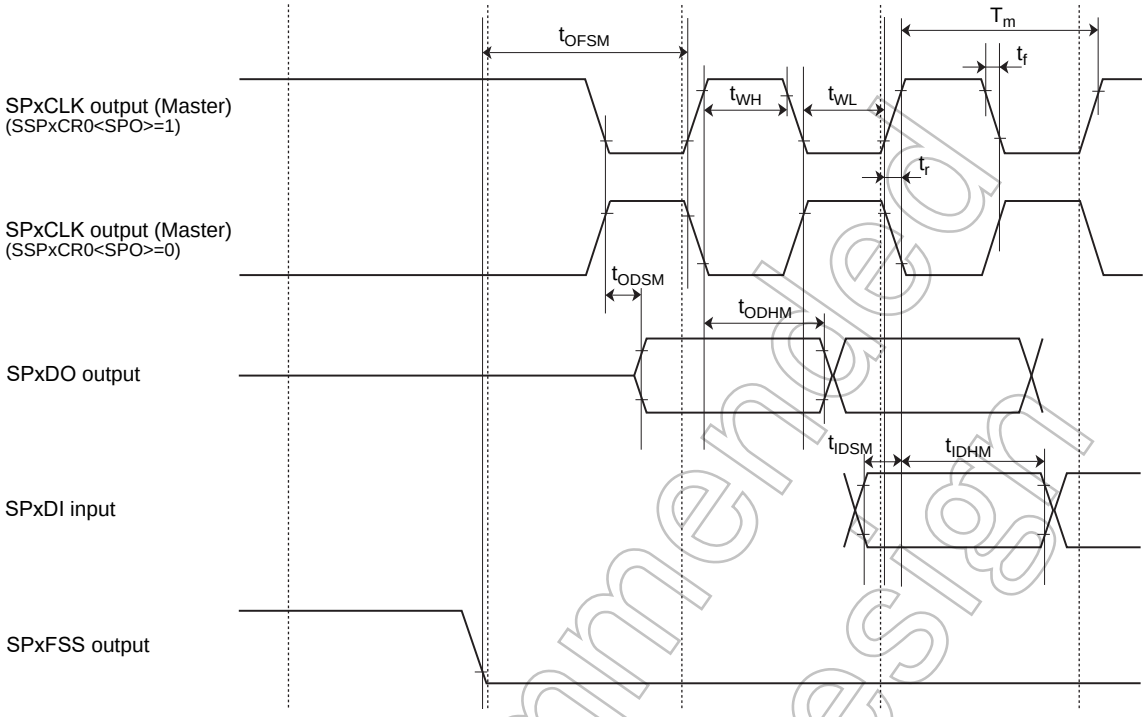
23.6.4.2 SSP SPI mode (Master)

- $fsys / 2 \geq f_{SPxCLK} \geq fsys / 65024$

(1) Master SSPxCR0<SPH>=0 (Data is latched on the first edge.)



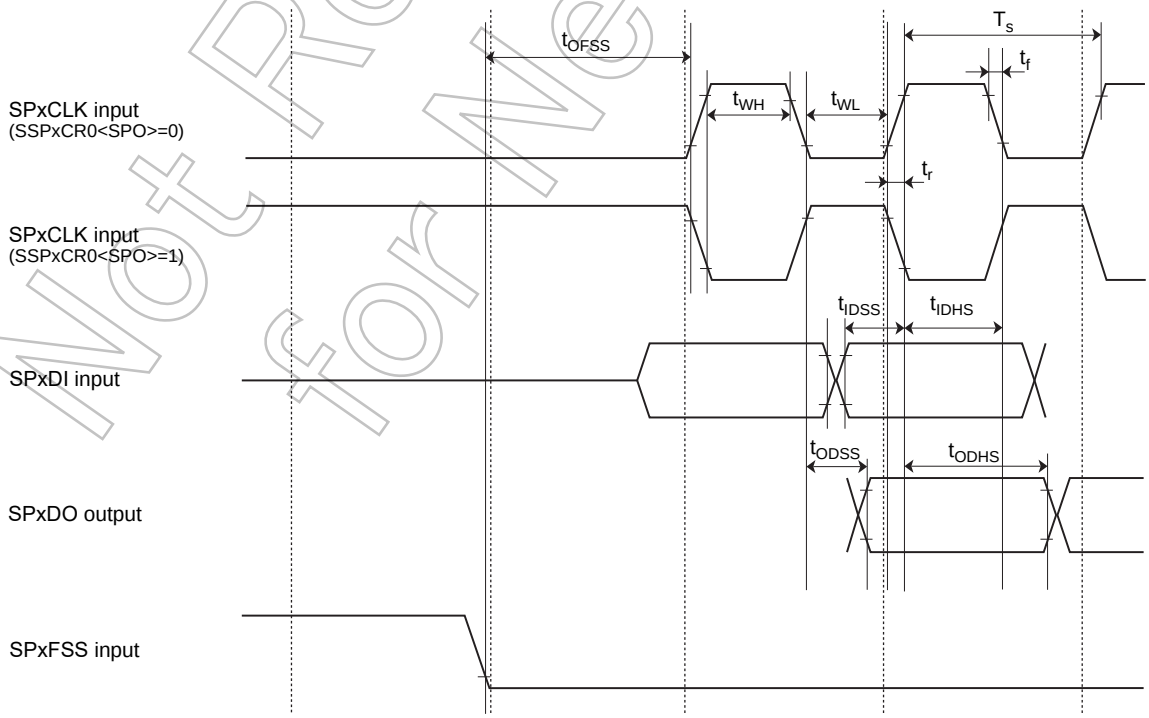
(2) Master SSPxCR0<SPH>=1 (Data is latched on the second edge.)



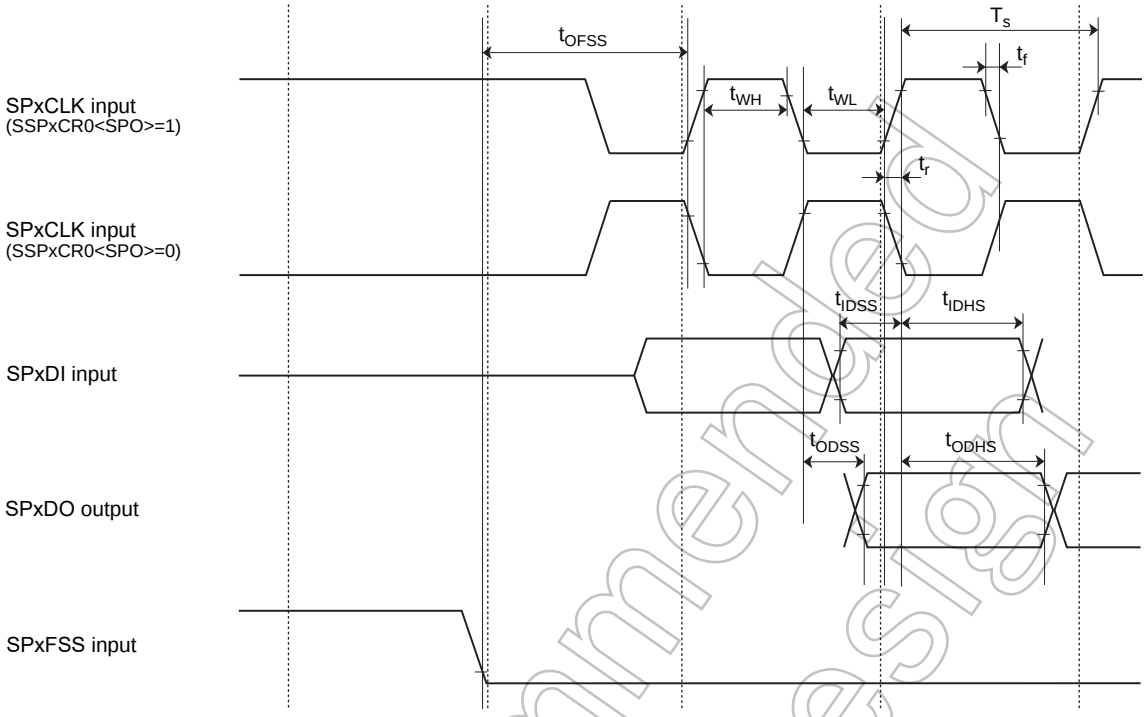
23.6.4.3 SSP SPI mode (Slave)

- $fsys / 12 \geq f_{SPxCLK} \geq fsys / 65024$

(1) Slave SSPxCR0<SPH>=0 (Data is latched on the first edge.)



(2) Slave SSPCR0<SPH>=1 (Data is latched on the second edge.)



23.6.5 16-bit timer / event counter

23.6.5.1 Event Counter

In the table below, the letter x represents the 16-bit timer / event counter operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
Clock Low pulse width	t_{VCKL}	2x + 100	−	150	−	142	−	ns
Clock High pulse width	t_{VCKH}	2x + 100	−	150	−	142	−	ns

23.6.5.2 Capture

In the table below, the letter x represents the 16-bit timer / event counter operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
Low pulse width	t_{CPL}	2x + 100	−	150	−	142	−	ns
High pulse width	t_{CPH}	2x + 100	−	150	−	142	−	ns

23.6.6 External Interrupt

In the table below, the letter x represents the fsys cycle time.

1. Except STOP1 and STOP2 release interrupts

Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
INT0 to 9 low level pulse width	t_{INTAL}	x + 100	−	125	−	121	−	ns
INT0 to 9 high level pulse width	t_{INTAH}	x + 100	−	125	−	121	−	ns

2. STOP1 release interrupts

Parameter	Symbol	Min.	Max.	Unit
INT0 to 9 low level pulse width	t_{INTBL}	100	−	ns
INT0 to 9 high level pulse width	t_{INTBH}	100	−	ns

3. STOP2 release interrupts

Parameter	Symbol	Min.	Max.	Unit
INT0 to 9 high level pulse width	t_{INTCH}	500	−	μs

23.6.7 $\overline{\text{NMI}}$

1. Except STOP1 and STOP2 release interrupts

Parameter	Symbol	Min.	Max.	Unit
$\overline{\text{NIM}}$ low level pulse width	t_{INTCL}	100	−	ns

2. STOP1 release interrupts

Parameter	Symbol	Min.	Max.	Unit
$\overline{\text{NMI}}$ low level pulse width	t_{INTBL}	100	−	ns

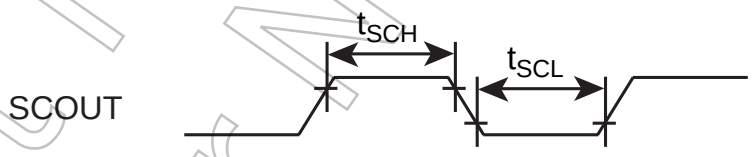
3. STOP2 release interrupts

Parameter	Symbol	Min.	Max.	Unit
$\overline{\text{NMI}}$ low level pulse width	t_{INTCL}	500	−	μs

23.6.8 SCOUT Pin AC Characteristic

Parameter	Symbol	Equation		40 MHz		48 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
High pulse width	t_{SCH}	0.5T − 5	−	7.5	−	5.4	−	ns
Low pulse width	t_{SCL}	0.5T − 5	−	7.5	−	5.4	−	ns

Note: **In the above table, the letter T represents the cycle time of the SCOUT output clock.**



23.6.9 $\overline{\text{ADTRG}}$ Trigger Input Pin AC Characteristic

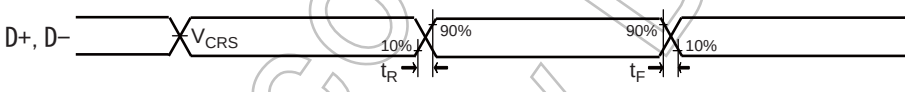
In the table below, the letter x represents fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		40MHz		48MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
Low level pulse width	T_{adl}	2 x + 20	−	70	−	62	−	ns
High level pulse interval	T_{adh}	2 x + 20	−	70	−	62	−	

23.6.10 USB Timing

DVDD3A = DVCC3C = RVDD = 3.0 to 3.45V
fsys = 48MHz

Parameter	Symbol	Min.	Max.	Unit
D+,D− rise time	t_R	4	20	ns
D+,D− withdraw time	t_F	4	20	
Data Line crossover voltage	V_{CRS}	1.3	2.0	V



23.6.11 External bus interface AC Characteristic

23.6.11.1 Separate Bus mode

Conditional variable : RWS = 1, TW = 2, RWH = 1 and CSH =1

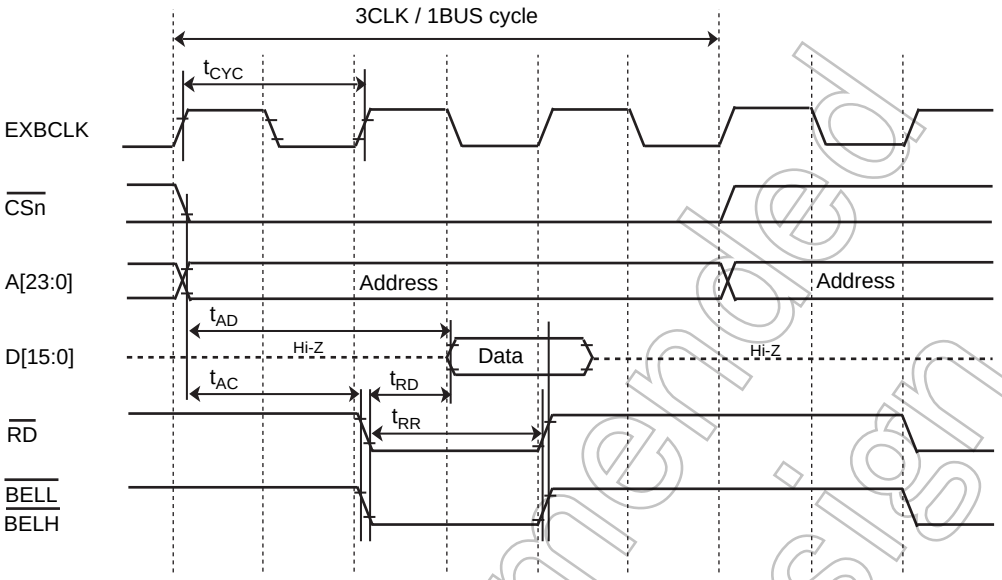
- RWS : Number of setup cycle insertion before $\overline{RD}$, $\overline{WR}$ asserted (TW = 0, 1, 2 or 4)
- TW : Number of internal wait insertion (TW = 0 to 15)
- RWH : Number of $\overline{RD}$, $\overline{WR}$ recovery cycle insertion (RWH = 0 to 6 or 8)
- CSH : Number of $\overline{CSn}$ recovery cycle insertion (CSH = 0, 1, 2 or 4)

DVDD3A = DVDD3C = AVDD3 = RVDD3 = 2.7 V to 3.6 V

Parameter	Symbol	Equation		40MHz		48MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
System clock period (x)	t_{SYS}	x	-	25	–	20.8	–	ns
External bus clock (EXBCLK)	t_{CYC}	x	-	25	–	20.8	–	
A[23:0] valid → $\overline{RD}$, $\overline{WR}$ asserted	t_{AC}	x (1+RWS)−10	-	40	–	31.7	–	
$\overline{RD}$, $\overline{WR}$ negated → A[23:0] hold	t_{CAR}	x (1+RWH+CSH)−10	-	65	–	52.5	–	
A[23:0] valid → D[15:0] data input	t_{AD}	-	x (2+RWS+TW)−40	–	85	–	64.2	
$\overline{RD}$ asserted → D[15:0] data input	t_{RD}	-	x (1+TW)−30	–	45	–	32.5	
$\overline{RD}$ Low-level pulse width	t_{RR}	x (1+TW)−12	-	63	–	50.5	–	
$\overline{RD}$ negated → D[15:0] hold	t_{HR}	0	-	0	–	0	–	
$\overline{RD}$ negated → A[23:0] output	t_{RAE}	x (1+RWH+CSH)−15	-	60	–	47.5	–	
$\overline{WR}$ Low-level pulse width	t_{WW}	x (1+TW)−15	-	60	–	47.5	–	
D[15:0] valid → $\overline{WR}$ negated	t_{DW}	x (1+TW)−15	-	60	–	47.5	–	
$\overline{WR}$ negated → D[15:0] hold	t_{WD}	x (1+RWH)−7	-	43	–	34.7	–	

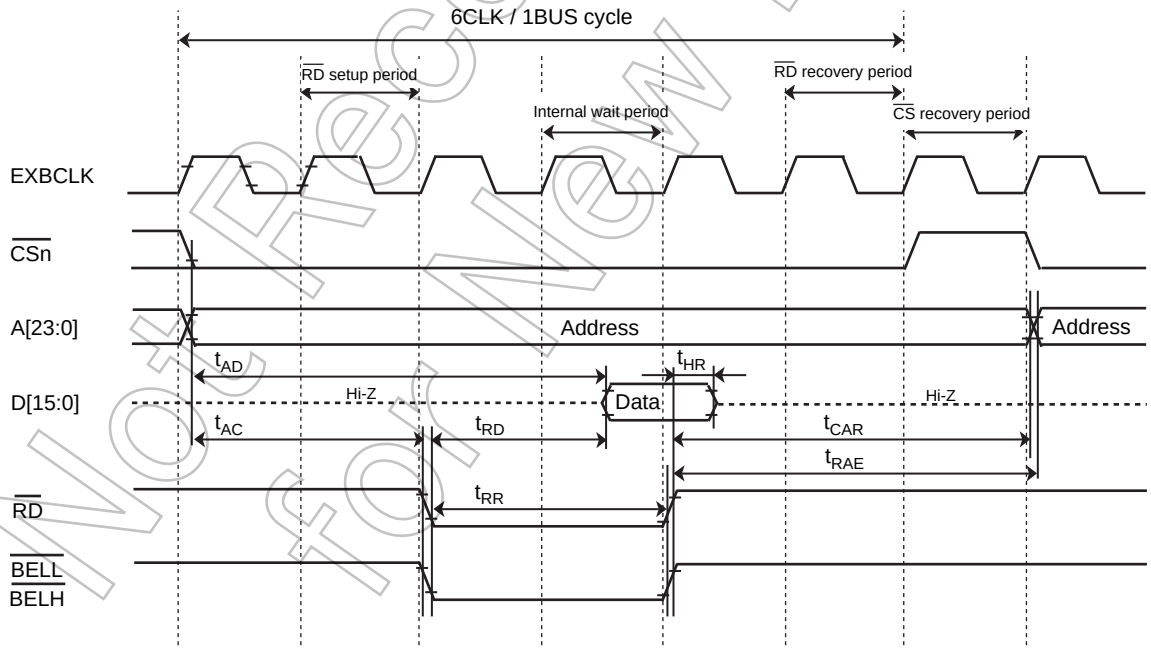
(1) Read cycle timing (minimum bus cycle)

(Neither Cycle expander, RD setup, Internal wait, CS recovery nor RD recovery function are used.)



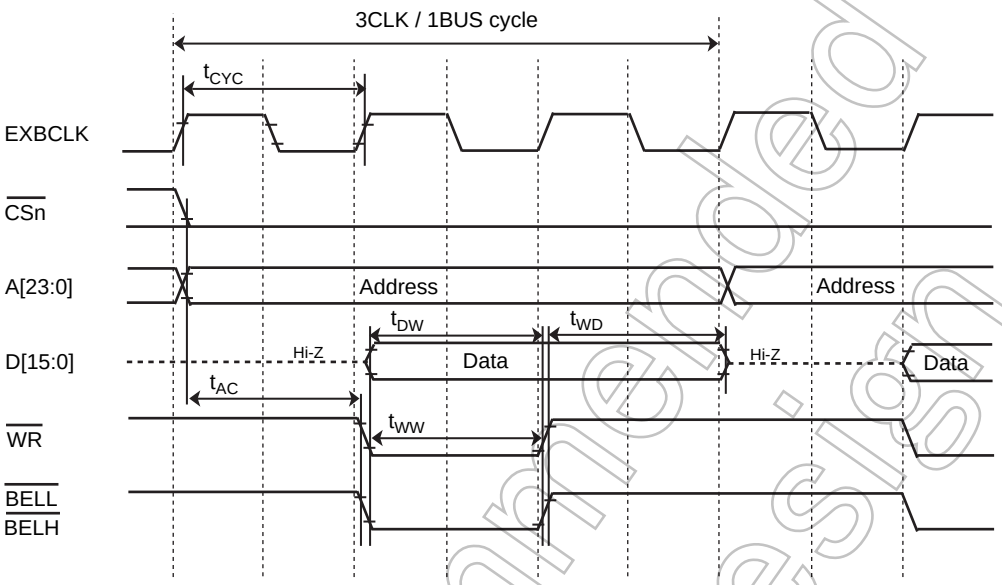
(2) Read cycle timing (1 bus cycle per 6 clock)

(RD setup, Internal wait, CS recovery and RD recovery function are set to 1 cycle though Cycle expander function is not used.)



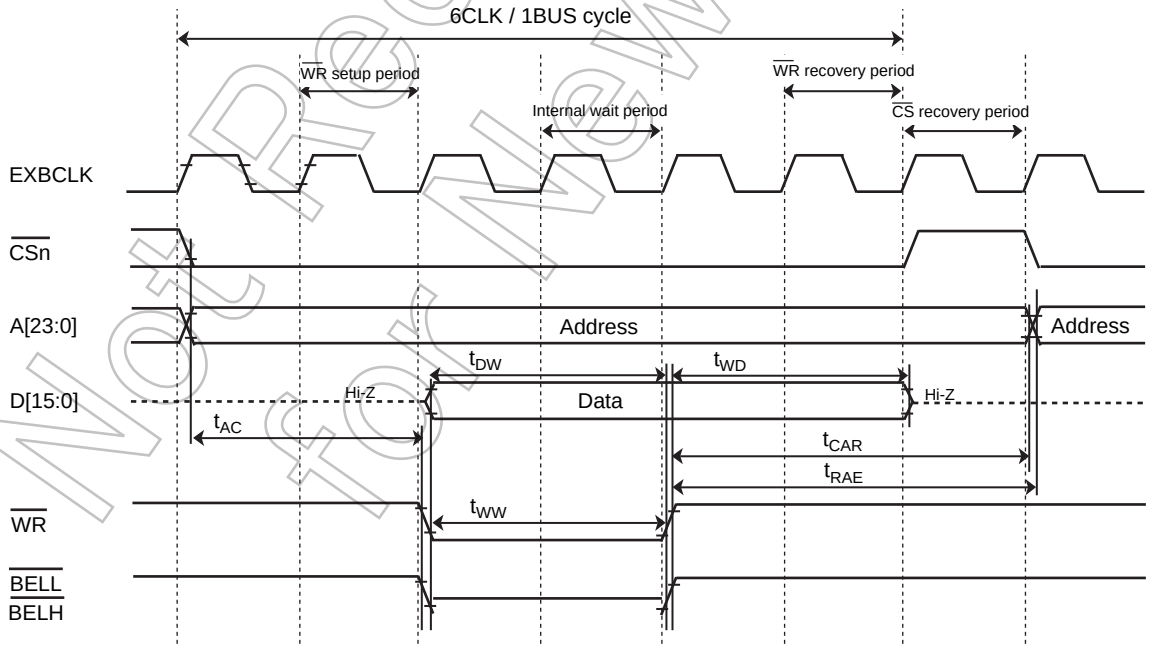
(3) Write cycle timing (minimum bus cycle)

(Neither Cycle expander, WR setup, Internal wait, CS recovery nor WR recovery function are used.)



(4) Write cycle timing (1 bus cycle per 6 clock)

(WR setup, Internal wait, CS recovery and WR recovery function are set to 1 cycle though Cycle expander function is not used.)



23.6.11.2 Multiplex Bus mode

Conditional variable : ALE = 1, RWS = 1, TW = 2, RWH = 1 and CSH =1

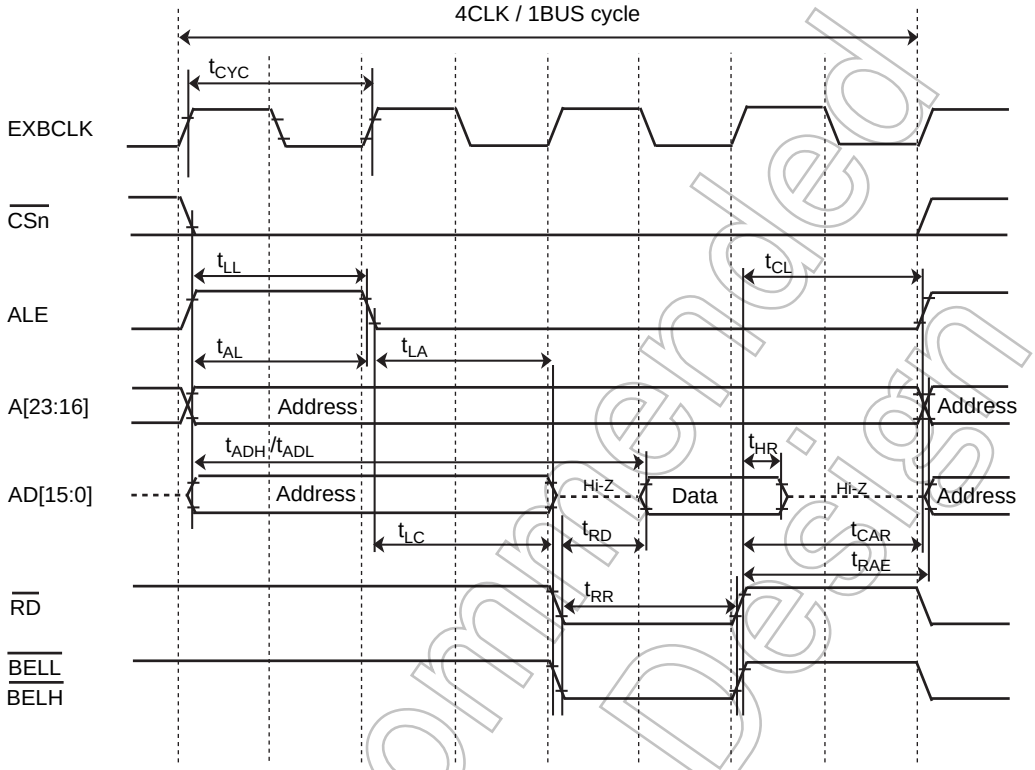
- ALE : Number of ALE cycle insertion (ALE = 0, 1, 2 or 4)
- RWS : Number of setup cycle insertion before $\overline{RD}$, $\overline{WR}$ asserted (TW = 0, 1, 2 or 4)
- TW : Number of internal wait insertion (TW = 0 to 15)
- RWH : Number of $\overline{RD}$, $\overline{WR}$ recovery cycle insertion (RWH = 0 to 6 or 8)
- CSH : Number of $\overline{CSn}$ recovery cycle insertion (CSH = 0, 1, 2 or 4)

DVDD3A = DVDD3C = AVDD3 = RVDD3 = 2.7 V to 3.6 V

Parameter	Symbol	Equation		40MHz		48MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
System clock period (x)	t_{SYS}	x	–	25	–	20.8	–	ns
External bus clock (EXBCLK)	t_{CYC}	x	–	25	–	20.8	–	
A[23:0] valid → ALE negated	t_{AL}	x (1+ALE)-26	–	24	–	15.7	–	
ALE negated → A[23:0] hold	t_{LA}	x (1+RWS)-7	–	43	–	34.7	–	
ALE high pulse width	t_{LL}	x (1+ALE)-15	–	35	–	26.7	–	
ALE negated → $\overline{RD}$ or $\overline{WR}$ asserted	t_{LC}	x (1+RWS)-7	–	43	–	34.7	–	
$\overline{RD}$ or $\overline{WR}$ negated → ALE asserted	t_{CL}	x (1+RWH+CSH)-20	–	55	–	42.5	–	
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+RWS)-19	–	81	–	64.3	–	
$\overline{RD}$ or $\overline{WR}$ negated → A[31:16] hold	t_{CAR}	x (1+RWH+CSH)-15	–	60	–	60	–	
A[15:0] valid → D[15:0] input A[23:16] valid → D[15:0] input	t_{ADL} t_{ADH}	–	x (3+ALE+RWS+TW)-35	–	140	–	111	
$\overline{RD}$ asserted → D[15:0] input	t_{RD}	–	x (1+TW)-30	–	45	–	32.5	
$\overline{RD}$ low pulse width	t_{RR}	x (1+TW)-12	–	63	–	50.5	–	
$\overline{RD}$ negated → D[15:0] hold	t_{HR}	0	–	0	–	0	–	
$\overline{RD}$ negated → A[23:0] output	t_{RAE}	x (1+RWH+CSH)-15	–	60	–	47.5	–	
$\overline{WR}$ low pulse width	t_{WW}	x (1+TW)-15	–	60	–	47.5	–	
D[15:0] valid → $\overline{WR}$ negated	t_{DW}	x (1+TW)-20	–	55	–	42.5	–	
$\overline{WR}$ negated → D[15:0] hold	t_{WD}	x (1+RWH)-7	–	43	–	34.7	–	

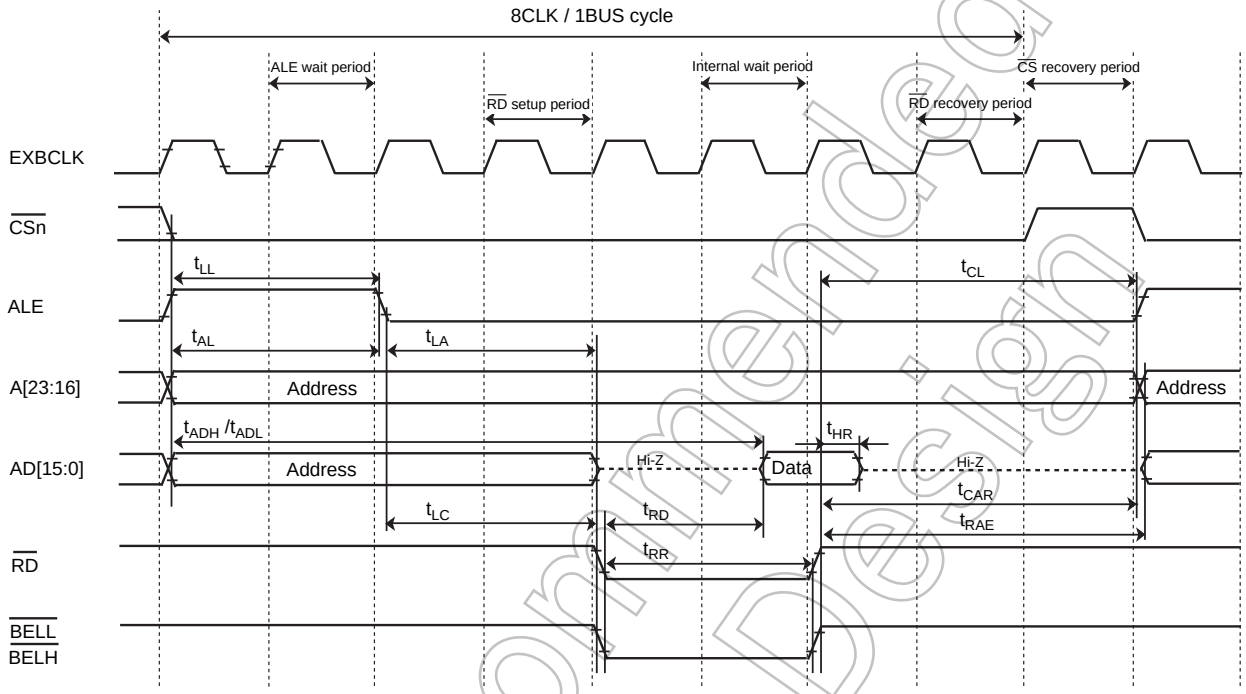
(1) Read cycle timing (minimum bus cycle)

(Neither Cycle expander, ALE wait, RD setup, Internal wait, CS recovery nor RD recovery function are used.)



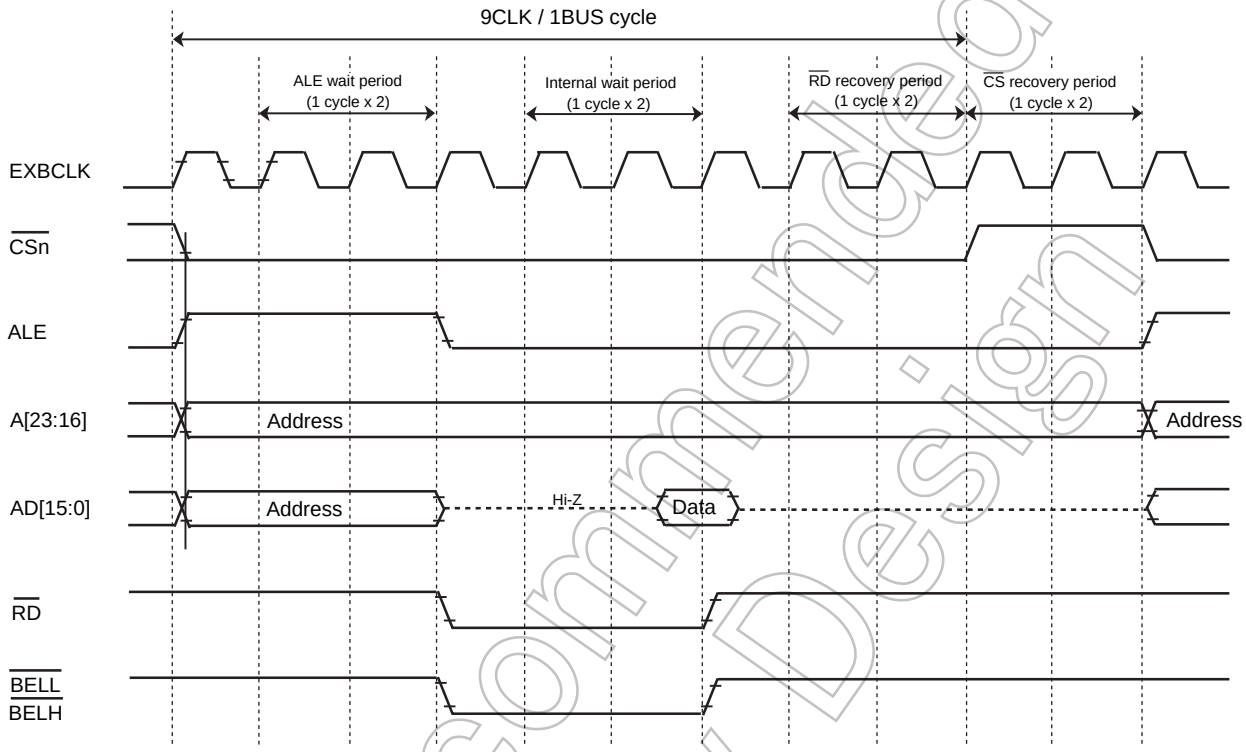
(2) Read cycle timing (1 bus cycle per 8 clock)

(ALE wait, RD setup, Internal wait, CS recovery and RD recovery function are set to 1 cycle though Cycle expander function is not used.)



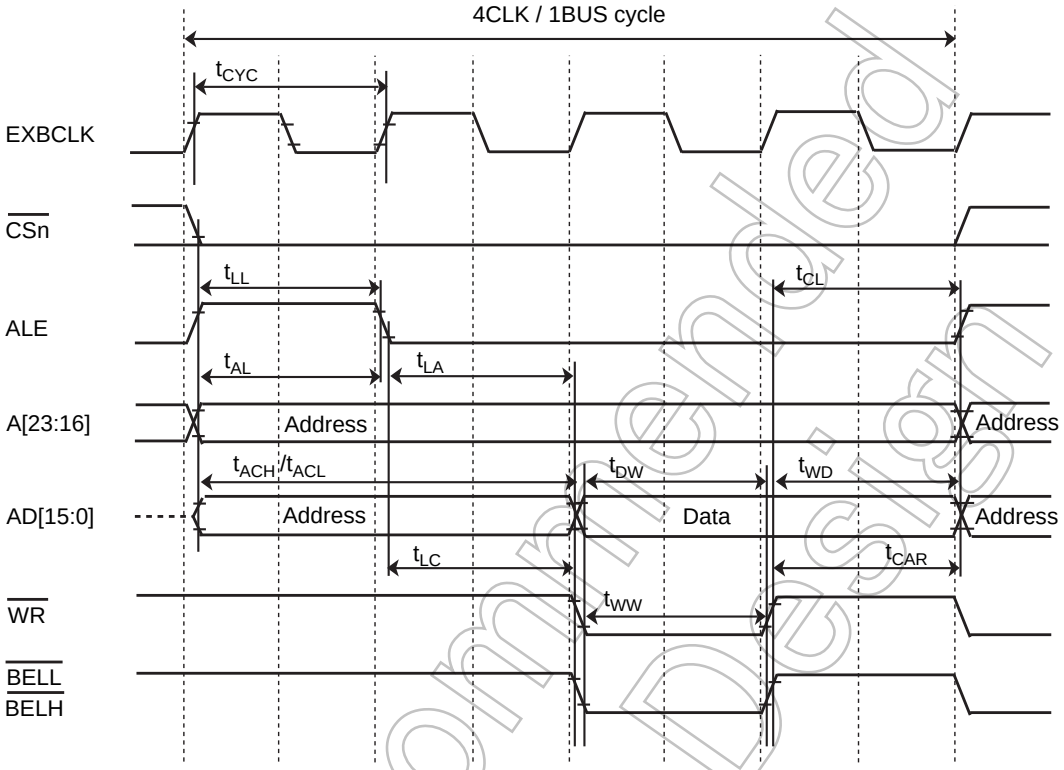
(3) Read cycle timing (1 bus cycle per 9 clock)

(ALE wait, RD setup, Internal wait, CS recovery and RD recovery function are set to 1 cycle though Cycle expander function is set double.)



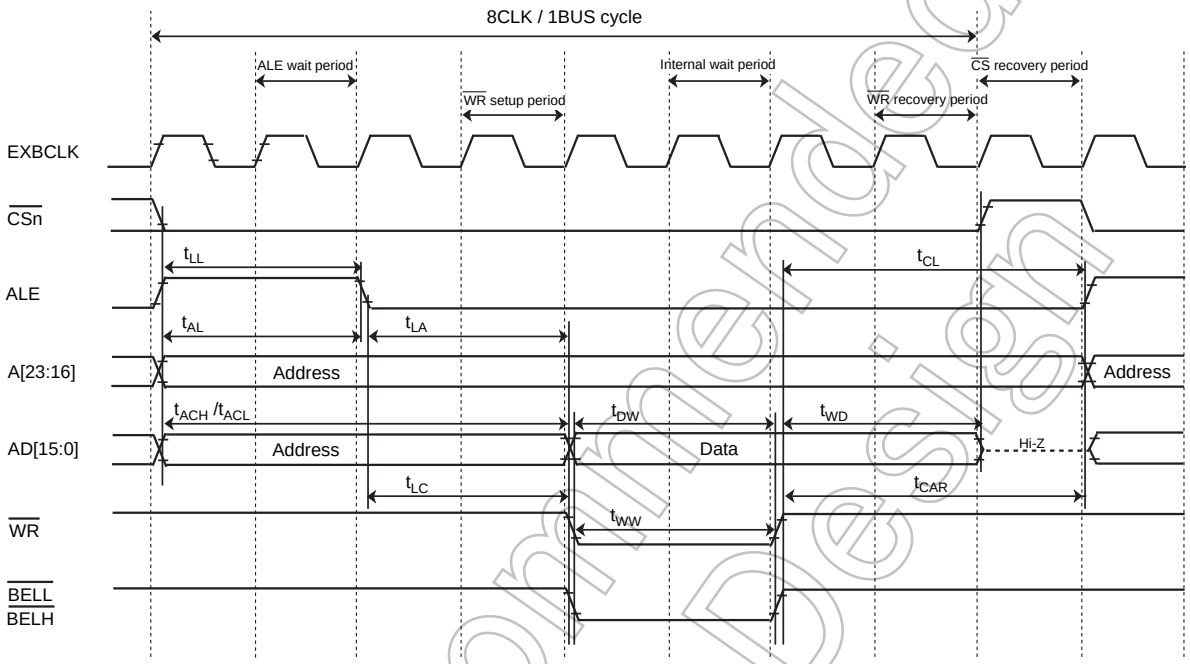
(4) Write cycle timing (minimum bus cycle)

(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.)



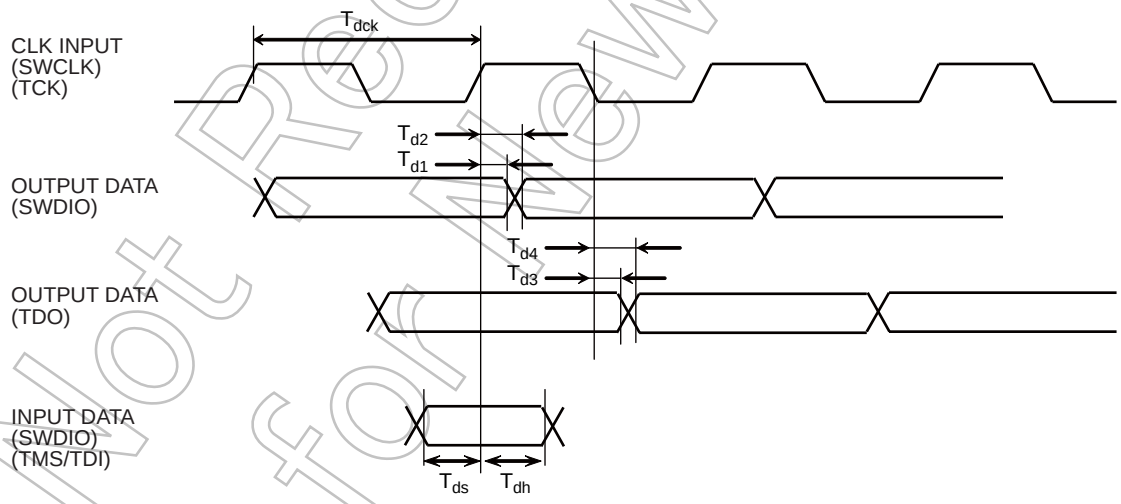
23.6.12 Debug Communication

23.6.12.1 SWD Interface

Parameter	Symbol	Min.	Max.	Unit
CLK cycle	T_{dck}	100	–	ns
CLK rise →Output data hold	T_{d1}	4	–	
CLK fall → Output data hold	T_{d2}	–	30	
Input data valid ←CLK rise	T_{ds}	20	–	
CLK rise → Input data hold	T_{dh}	15	–	

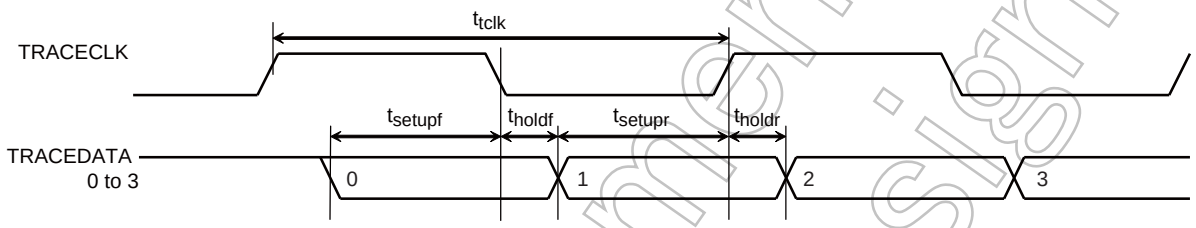
23.6.12.2 JTAG Interface

Parameter	Symbol	Min.	Max.	Unit
CLK cycle	T_{dck}	100	–	ns
CLK rise →Output data hold	T_{d3}	4	–	
CLK fall → Output data hold	T_{d4}	–	50	
Input data valid ←CLK rise	T_{ds}	20	–	
CLK rise → Input data hold	T_{dh}	15	–	



23.6.13 ETM Trace

Parameter	Symbol	Min.	Max.	Unit
TRACECLK cycle	t_{tclk}	50	−	ns
TRACEDATA valid ← TRACECLK rise	t_{setupr}	2	−	ns
TRACECLK rise → TRACEDATA hold	t_{holdr}	1	−	ns
TRACEDATA valid ← TRACECLK fall	t_{setupf}	2	−	ns
TRACECLK fall → TRACEDATA hold	t_{holdf}	1	−	ns



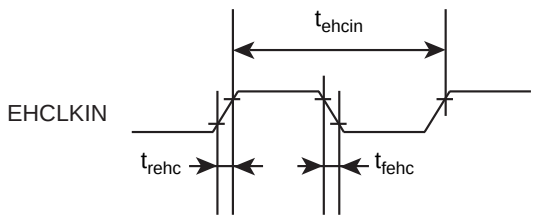
23.6.14 On chip oscillator

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillating frequency	IHOSC	Ta = -40 to 85°C	8	10	12	MHz
		Ta =0 to 65°C	8.5	10	11.5	

Note: The on-chip-oscillator can not be used as system clock (fsys) which is required oscillation accuracy.

23.6.15 External clock input

Parameter	Symbol	Min.	Typ.	Max.	Unit
External clock frequency	t_{ehcin}	8	−	48	MHz
External clock duty	−	45	−	55	%
External clock input rise time	t_{rehc}	−	−	10	ns
External clock input fall time	t_{fehc}	−	−	10	ns



23.6.16 Flash Memory Characteristics

Parameter	condition	Min.	Typ.	Max.	unit
Guarantee on Flash-Mmory Re-writing	DVDD3A = DVDD3C = AVDD3 = RVDD3 = 2.7 V to 3.6 V, Ta = 0 to 70 °c	-	-	100	times

23.7 Recommended Oscillation Circuit

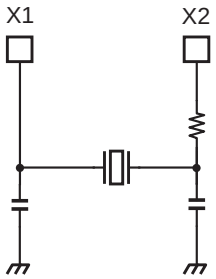


Figure 23-1 High-frequency oscillation connection

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 substratepatterns, please evaluate oscillation stability using the substrate you use.**

The TMPM366FDXBG, TMPM366FYXBG and TMPM366FWXBG have been evaluated by the oscillator vender below. Please refer this information when selecting exter-nal parts

23.7.1 Ceramic oscillator

The TMPM366FDXBG, TMPM366FYXBG and TMPM366FWXBG recommend the high-frequency oscillator by Murata Manufacturing Co., Ltd.

Please refer to the Murata Website for details.

23.7.2 Crystal oscillator

The TMPM366FDXBG, TMPM366FYXBG and TMPM366FWXBG recommend the high-frequency oscillator by KYOCERA Corporation.

Please refer to the KYOCERA Website for details.

23.7.2.1 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 lenghts of such patterns become shortest possible to prevent deterioration of characteristics due to stray capacitances and wiring inductance. For multi-layer circuit boards, it is importtant not to wire the ground and other signal patterns right beneath the oscillation circuit.

For more information ,please refer to the URL of the oscillator vendor.

23.8 Handling Precaution

23.8.1 Power-on sequence

The power-on sequence must include the time for the internal regulator, internal flash memory and internal oscillator to be stable and the reset. In the TX03, the internal circuit automatically insert the time for the internal regulator which requires the time at least 1 ms and after this, internal reset operation requires 4096 cycles on internal oscillation, therefore, A little bit of time differences occur until CPU start operate. And there are multiple independent Power supply, however please operate the power-on procedure simultaneously.

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

Figure 23-2 shows the power-on sequence.

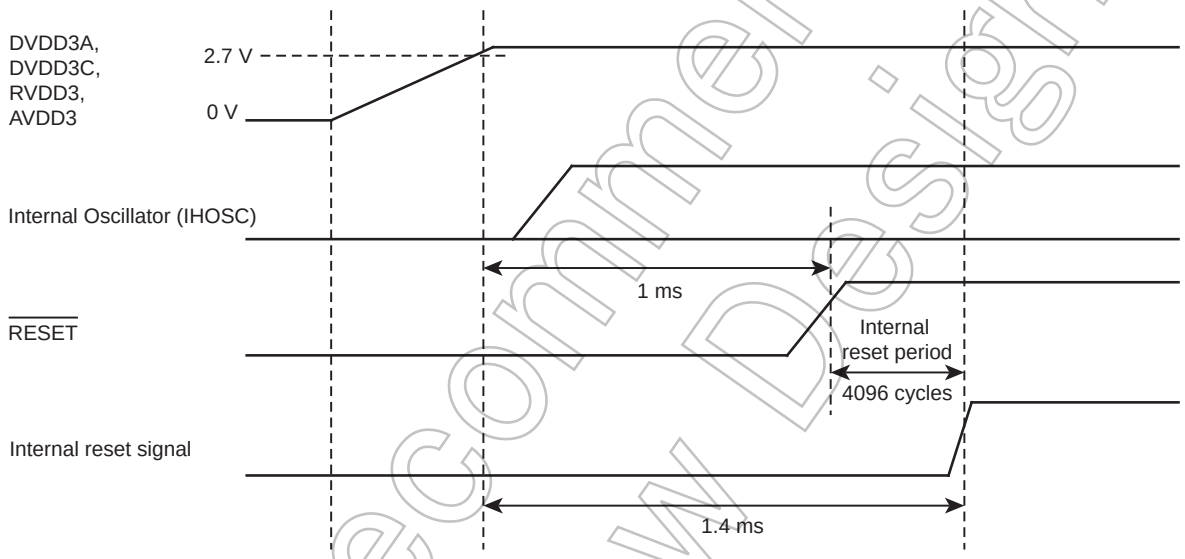
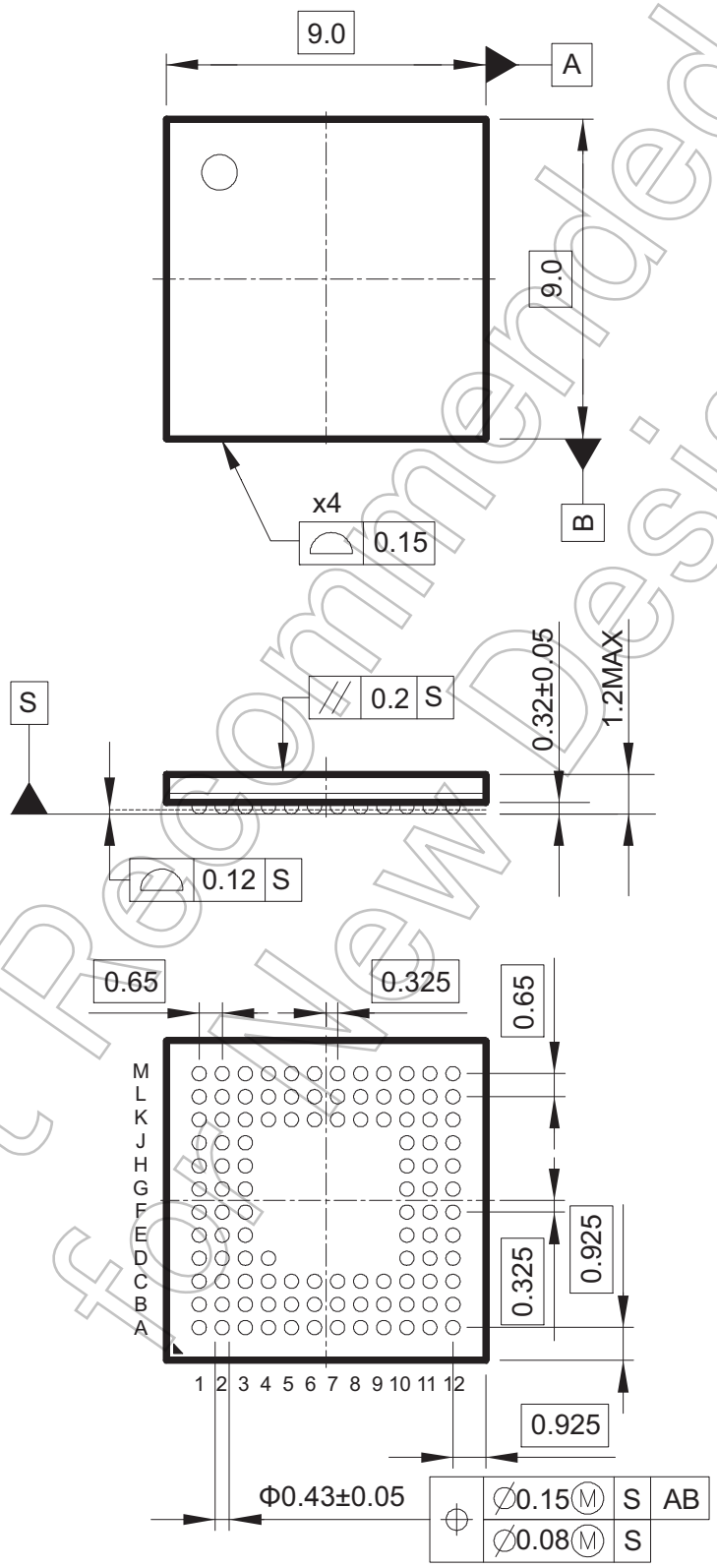


Figure 23-2 Power-on sequence

24. Package Dimensions

Type: P-TFBGA109-0909-0.65-002 Unit:mm



RESTRICTIONS ON PRODUCT USE

Toshiba Corporation and its subsidiaries and affiliates are collectively referred to as “TOSHIBA”.
Hardware, software and systems described in this document are collectively referred to as “Product”.

- TOSHIBA reserves the right to make changes to the information in this document and related Product without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating 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, lifesaving and/or life supporting medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, and devices related to power plant. IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT. For details, please contact your TOSHIBA sales representative or contact us via our website.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. **TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.**