TOSHIBA

32 Bit RISC Microcontroller
TX03 Series

TMPM365FYXBG

TOSHIBA CORPORATION
Semiconductor & Storage Products Company

Important Notices

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

1 Serial bus interface

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

1.1 Description

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

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

1.2 Condition

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

1.3 Workaround

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

1.4 How to Recover from These Phenomena

Perform recovery process by software.

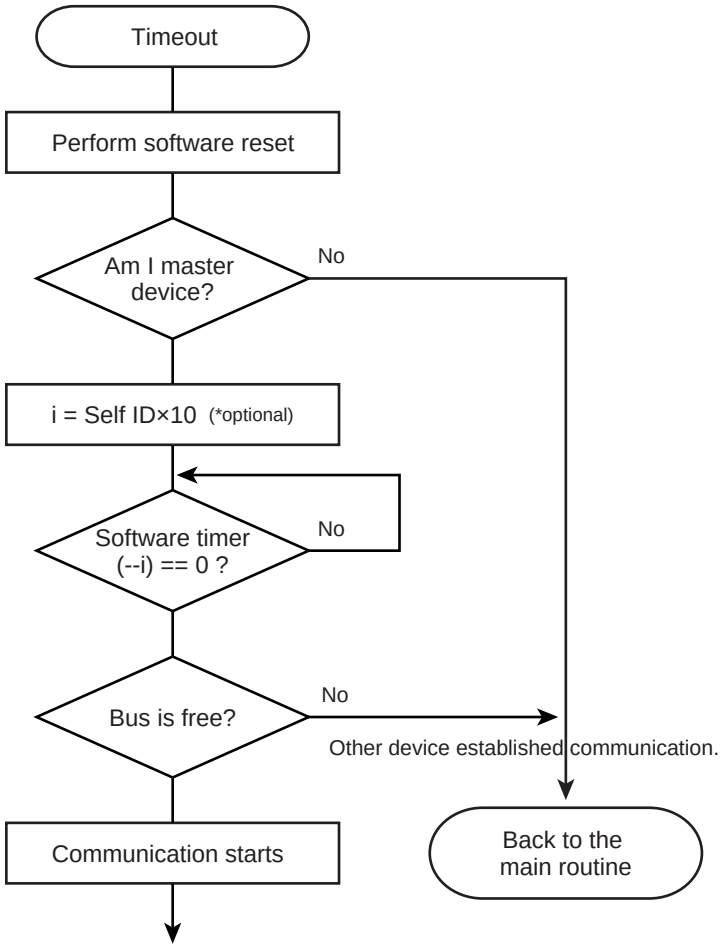
By using a timer, add timeout process to check whether communication is in a lock-up state.

An example of recovery process:

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

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

Example: Recovery process after a timeout is detected.



2 Transitions to Low-power Consumption Mode and Generating a Non-maskable Interrupt

This chapter describes the precautions at which non-maskable interrupt (NMI) occurs when the MCU enters low-power consumption mode.

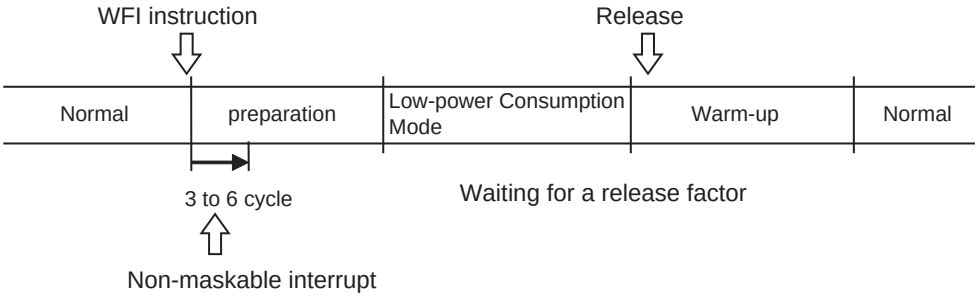
- STOP1

2.1 Description

When the WFI instruction is executed to enter the above-mentioned low-power consumption mode, if a non-maskable interrupt (NMI) occurs, the MCU may enter low-power consumption mode without the process for releasing low-power consumption mode.

Note 1: An NMI notice and flag setting to the CPU are normal; therefore the NMI process after low-power consumption mode is released can be performed.

Note 2: When the MCU has entered low-power consumption mode, factors other than an NMI are accepted; however an NMI may not be accepted.



2.2 Conditions

- The WFI instruction is executed to enter low-power consumption mode.
- A non-maskable interrupt occurs after WFI instruction execution and within 3 to 6 cycles.

2.3 Workaround

Do not use a non-maskable interrupt as a release factor for the above-mentioned low-power consumption modes.

To avoid generating a non-maskable interrupt, the following settings should be specified before the MCU enters the above-mentioned low-power consumption mode.

- NMI pin: This input pin must be fixed to "High".
- Watchdog timer: Stop the watchdog timer or set the reset function.
- Voltage detection circuit: Stop the voltage detection circuit or set the reset function.

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
2013/06/12	1	First Release
2022/09/30	2	Contents Revised
2023/07/21	3	Contents Revised

Table of Contents

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

TMPM365FYXBG

1.1 Features 1
1.2 Block Diagram 4
1.3 Pin Layout (Top view) 5
1.4 Pin names and Functions 6
1.4.1 Sorted by Pin 6
1.4.2 Sorted by Port 12
1.5 Pin Numbers and Power Supply Pins 18
1.6 Internal regulator pins 18

2. Processor Core

2.1 Information on the processor core 19
2.2 Configurable Options 19
2.3 Exceptions/ Interruptions 20
2.3.1 Number of Interrupt Inputs 20
2.3.2 Number of Priority Level Interrupt Bits 20
2.3.3 SysTick 20
2.3.4 SYSRESETREQ 20
2.3.5 LOCKUP 20
2.3.6 Auxiliary Fault Status register 20
2.4 Events 21
2.5 Power Management 21
2.6 Exclusive access 21

3. Debug Interface

3.1 Specification Overview 23
3.2 SWJ-DP 23
3.3 ETM 23
3.4 Pin Functions 24
3.5 Peripheral Functions in Halt Mode 25
3.6 Connection with a Debug Tool 26
3.6.1 About connection with debug tool 26
3.6.2 Important points of using debug interface pins used as general-purpose ports 26

4. JTAG Interface

4.1 Overview 27
4.2 Signal Summary and Connection Example 28
4.3 Outline 29
4.4 JTAG Controller and Registers 29
4.5 Instruction Register 30
4.6 Boundary Scan Register 31
4.7 Test Access Port (TAP) 32
4.8 TAP Controller 32
4.9 Resetting the TAP Controller 32
4.10 State Transitions of the TAP Controller 33
4.11 Boundary Scan Order 36
4.12 Instructions Supported by the JTAG Controller Cells 37

5. Memory Map

5.1 Memory map 41
5.1.1 Memory map of the TMPM365FYXBG 42
5.2 SFR area detail 43

6. Reset

6.1 Initial state 45
6.2 Cold reset 45
6.3 Warm reset 46
6.3.1 Reset period 46
6.4 After reset 46

7. Watchdog Timer(WDT)

7.1 Configuration 47
7.2 Register 48
7.2.1 WDMOD(Watchdog Timer Mode Register) 48
7.2.2 WDCR (Watchdog Timer Control Register) 50
7.3 Operations 51
7.3.1 Basic Operation 51
7.3.2 Operation Mode and Status 51
7.4 Operation when malfunction (runaway) is detected 52
7.4.1 INTWDT interrupt generation 52
7.4.2 Internal reset generation 53
7.5 Control register 54
7.5.1 Watchdog Timer Mode Register (WDMOD) 54
7.5.2 Watchdog Timer Control Register(WDCR) 54
7.5.3 Setting example 55
7.5.3.1 Disabling control
7.5.3.2 Enabling control
7.5.3.3 Watchdog timer clearing control
7.5.3.4 Detection time of watchdog timer

8. Clock/Mode control

8.1 Features 57
8.2 Registers 58
8.2.1 Register List 58
8.2.2 CGSYSCR (System control register) 59
8.2.3 CGOSCCR (Oscillation control register) 60
8.2.4 CGSTBYCR (Standby control register) 62
8.2.5 CGPLLSEL (PLL Selection Register) 63
8.2.6 CGUSBCTL (USB clock control register) 64
8.2.7 CGPROTECT (Protect register) 65
8.3 Clock control 66
8.3.1 Clock type 66
8.3.2 Initial Values after Reset 66
8.3.3 Clock system Diagram 67
8.3.4 Warm-up function 68
8.3.5 Clock Multiplication Circuit (PLL) 70
8.3.5.1 Start operation
8.3.6 System clock 72
8.3.6.1 System Clock setting
8.3.7 Prescaler Clock Control 74
8.3.8 System Clock Pin Output Function 74
8.4 Modes and Mode Transitions 75
8.4.1 Mode Transitions 75
8.5 Operation mode 76
8.5.1 NORMAL mode 76
8.6 Low Power Consumption Modes 76
8.6.1 IDLE mode 76
8.6.2 STOP1 mode 77
8.6.3 Low power Consumption Mode Setting 78
8.6.4 Operational Status in Each Mode 79
8.6.5 Releasing the Low Power Consumption Mode 80
8.6.6 Warm-up 81
8.6.7 Clock Operations in Mode Transition 82
8.6.7.1 Transition of operation modes: NORMAL → STOP1 → NORMAL

9. Exceptions

9.1 Overview 83
9.1.1 Exception Types 83
9.1.2 Handling Flowchart 84
9.1.2.1 Exception Request and Detection
9.1.2.2 Exception Handling and Branch to the Interrupt Service Routine (Pre-emption)
9.1.2.3 Executing an ISR
9.1.2.4 Exception exit
9.2 Reset Exceptions 89
9.3 Non-Maskable Interrupts (NMI) 90
9.4 SysTick 90
9.5 Interrupts 91
9.5.1 Interrupt Sources 91
9.5.1.1 Interrupt Route
9.5.1.2 Generation
9.5.1.3 Transmission
9.5.1.4 Precautions when using external interrupt pins
9.5.1.5 List of Interrupt Sources
9.5.1.6 Active level
9.5.2 Interrupt Handling 96
9.5.2.1 Flowchart
9.5.2.2 Preparation
9.5.2.3 Detection by Clock Generator
9.5.2.4 Detection by CPU
9.5.2.5 CPU processing
9.5.2.6 Interrupt Service Routine (ISR)
9.6 Exception/Interrupt-Related Registers 101
9.6.1 Register List 101

9.6.2 NVIC Registers....102
9.6.2.1 SysTick Control and Status Register
9.6.2.2 SysTick Reload Value Register
9.6.2.3 SysTick Current Value Register
9.6.2.4 SysTick Calibration Value Register
9.6.2.5 Interrupt Set-Enable Register 1
9.6.2.6 Interrupt Set-Enable Register 2
9.6.2.7 Interrupt Clear-Enable Register 1
9.6.2.8 Interrupt Clear-Enable Register 2
9.6.2.9 Interrupt Set-Pending Register 1
9.6.2.10 Interrupt Set-Pending Register 2
9.6.2.11 Interrupt Clear-Pending Register 1
9.6.2.12 Interrupt Clear-Pending Register 2
9.6.2.13 Interrupt Priority Register
9.6.2.14 Vector Table Offset Register
9.6.2.15 Application Interrupt and Reset Control Register
9.6.2.16 System Handler Priority Register
9.6.2.17 System Handler Control and State Register
9.6.3 Clock generator registers....124
9.6.3.1 CGIMCGA(CG Interrupt Mode Control Register A)
9.6.3.2 CGIMCGB(CG Interrupt Mode Control Register B)
9.6.3.3 CGIMCGC(CG Interrupt Mode Control Register C)
9.6.3.4 CGICRCG(CG Interrupt Request Clear Register)
9.6.3.5 CGNMIFLG(NMI Flag Register)
9.6.3.6 CGRSTFLG (Reset Flag Register)

10. Input/Output Ports

10.1 Port Functions....133
10.1.1 Function Lists....133
10.1.2 Port Registers Outline....136
10.1.3 Port States in STOP1 Mode....137
10.2 Port functions....138
10.2.1 Port A (PA0 to PA7)....138
10.2.1.1 Port A register
10.2.1.2 PADATA (Port A data register)
10.2.1.3 PACR (Port A output control register)
10.2.1.4 PAOD (Port A open drain control register)
10.2.1.5 PAPUP (Port A pull-up control register)
10.2.1.6 PAIE (Port A input control register)
10.2.2 Port B (PB0 to PB7)....142
10.2.2.1 Port B Register
10.2.2.2 PBDATA (Port B data register)
10.2.2.3 PBCR (Port B output control register)
10.2.2.4 PBOD (Port B open drain control register)
10.2.2.5 PBPUP (Port B pull-up control register)
10.2.2.6 PBIE (Port B input control register)
10.2.3 Port C (PC0 to PC2)....146
10.2.3.1 Port C Register
10.2.3.2 PCDATA (Port C data register)
10.2.3.3 PCCR (Port C output control register)
10.2.3.4 PCFR1 (Port C function register 1)
10.2.3.5 PCFR3 (Port C function register 3)
10.2.3.6 PCFR4 (Port C function register 4)
10.2.3.7 PCOD (Port C open drain control register)
10.2.3.8 PCPUP (Port C pull-up control register)
10.2.3.9 PCIE (Port C input control register)
10.2.4 Port D (PD0 to PD7)....151
10.2.4.1 Port D Register
10.2.4.2 PDDATA (Port D data register)
10.2.4.3 PDCR (Port D output control register)
10.2.4.4 PDFR3 (Port D function register 3)
10.2.4.5 PDOD (Port D open drain control register)
10.2.4.6 PDPUP (Port D pull-up control register)
10.2.4.7 PDIE (Port D input control register)
10.2.5 Port E (PE0 to PE7)....156
10.2.5.1 Port E Register
10.2.5.2 PEDATA (Port E data register)
10.2.5.3 PECR (Port E output control register)
10.2.5.4 PEFR1 (Port E function register 1)
10.2.5.5 PEFR3 (Port E function register 3)
10.2.5.6 PEFR4 (Port E function register 4)

10.2.5.7 PEOD (Port E open drain control register)
10.2.5.8 PEPUP (Port E pull-up control register)
10.2.5.9 PEIE (Port E input control register)
10.2.6 Port F (PF0 to PF7)....162
10.2.6.1 Port F Register
10.2.6.2 PFDATA (Port F data register)
10.2.6.3 PFCR (Port F output control register)
10.2.6.4 PFFR2 (Port F function register 2)
10.2.6.5 PFFR3 (Port F function register 3)
10.2.6.6 PFOD (Port F open drain control register)
10.2.6.7 PFPUP (Port F pull-up control register)
10.2.6.8 PFIE (Port F input control register)
10.2.7 Port G (PG0 to PG5)....167
10.2.7.1 Port G Register
10.2.7.2 PGDATA (Port G data register)
10.2.7.3 PGCR (Port G output control register)
10.2.7.4 PGFR1 (Port G function register 1)
10.2.7.5 PGFR3 (Port G function register 3)
10.2.7.6 PGFR4 (Port G function register 4)
10.2.7.7 PGOD (Port G open drain control register)
10.2.7.8 PGPUP (Port G pull-up control register)
10.2.7.9 PGIE (Port G input control register)
10.2.8 Port H (PH0 to PH4)....173
10.2.8.1 Port H Register
10.2.8.2 PHDATA (Port H data register)
10.2.8.3 PHCR (Port H output control register)
10.2.8.4 PHFR1 (Port H function register 1)
10.2.8.5 PHFR3 (Port H function register 3)
10.2.8.6 PHOD (Port H open drain control register)
10.2.8.7 PHPUP (Port H pull-up control register)
10.2.8.8 PHIE (Port H input control register)
10.2.9 Port I (PI0 to PI7)....178
10.2.9.1 Port I Register
10.2.9.2 PIDATA (Port I data register)
10.2.9.3 PICR (Port I output control register)
10.2.9.4 PIFR1(Port I function register 1)
10.2.9.5 PIOD (Port I open drain control register)
10.2.9.6 PIPUP (Port I pull-up control register)
10.2.9.7 PIPDN (Port I pull-down control register)
10.2.9.8 PIIE (Port I input control register)
10.2.10 Port J (PJ0 to PJ7)....185
10.2.10.1 Port J Register
10.2.10.2 PJDATA (Port J data register)
10.2.10.3 PJCR (Port J output control register)
10.2.10.4 PJFR2 (Port J function register 2)
10.2.10.5 PJFR3 (Port J function register 3)
10.2.10.6 PJPUP (Port J pull-up control register)
10.2.10.7 PJIE (Port J input control register)
10.2.11 Port K (PK0 to PK3)....189
10.2.11.1 Port K Register
10.2.11.2 PKDATA (Port K data register)
10.2.11.3 PKCR (Port K output control register)
10.2.11.4 PKFR2 (Port K function register 2)
10.2.11.5 PKFR3 (Port K function register 3)
10.2.11.6 PKPUP (Port K pull-up control register)
10.2.11.7 PKIE (Port K input control register)
10.3 Block Diagrams of Ports....193
10.3.1 Port Types....193
10.3.2 Type FT1....194
10.3.3 Type FT2....195
10.3.4 Type FT3....196
10.3.5 Type FT4....197
10.3.6 Type FT5....198
10.3.7 Type FT6....199
10.4 Appendix (Port setting List)....200
10.4.1 Port A Setting....200
10.4.2 Port B Setting....201
10.4.3 Port C Setting....202
10.4.4 Port D Setting....203
10.4.5 Port E Setting....204
10.4.6 Port F Setting....205
10.4.7 Port G Setting....206
10.4.8 Port H Setting....207

10.4.9 Port I Setting....208
10.4.10 Port J Setting....209
10.4.11 Port K Setting....210

11. DMA Controller(DMAC)

11.1 Overview....211
11.2 DMA transfer type....212
11.3 Block diagram....213
11.4 Product information of TMPM365FYXBG....214
11.4.1 Peripheral function supported with Peripheral to Peripheral Transfer....214
11.4.2 DMA request....214
11.4.3 Interrupt request....215
11.4.4 Base address of registers....215
11.5 Description of Registers....216
11.5.1 DMAC register list....216
11.5.2 DMACxIntStatus (DMAC Interrupt Status Register)....217
11.5.3 DMACxIntTCStatus (DMAC Interrupt Terminal Count Status Register)....218
11.5.4 DMACxIntTCClear (DMAC Interrupt Terminal Count Clear Register)....219
11.5.5 DMACxIntErrorStatus (DMAC Interrupt Error Status Register)....220
11.5.6 DMACxIntErrClr (DMAC Interrupt Error Clear Register)....221
11.5.7 DMACxRawIntTCStatus (DMAC Raw Interrupt Terminal Count Status Register)....222
11.5.8 DMACxRawIntErrorStatus (DMAC Raw Error Interrupt Status Register)....223
11.5.9 DMACxEnbldChns (DMAC Enabled Channel Register)....224
11.5.10 DMACxSoftBReq (DMAC Software Burst Request Register)....225
11.5.11 DMACxSoftSReq (DMAC Software Single Request Register)....227
11.5.12 DMACxConfiguration (DMAC Configuration Register)....229
11.5.13 DMACxCnSrcAddr (DMAC Channelx Source Address Register)....230
11.5.14 DMACxCnDestAddr (DMAC Channelx Destination Address Register)....231
11.5.15 DMACxCnLLI (DMAC Channelx Linked List Item Register)....232
11.5.16 DMACxCnControl (DMAC Channeln Control Register)....233
11.5.17 DMACxCnConfiguration (DMAC Channel n Configuration Register)....235
11.6 Special Functions....237
11.6.1 Scatter/gather function....237
11.6.2 Linked list operation....238

12. 16-bit Timer/Event Counters(TMRB)

12.1 Outline....241
12.2 Differences in the Specifications....242
12.3 Configuration....243
12.4 Registers....244
12.4.1 Register list according to channel....244
12.4.2 TBxEN (Enable register)....245
12.4.3 TBxRUN(RUN register)....246
12.4.4 TBxCR (Control register)....247
12.4.5 TBxMOD (Mode register)....248
12.4.6 TBxFFCR (Flip-flop control register)....250
12.4.7 TBxST (Status register)....251
12.4.8 TBxIM (Interrupt mask register)....252
12.4.9 TBxUC (Up counter capture register)....253
12.4.10 TBxRG0 (Timer register 0)....254
12.4.11 TBxRG1 (Timer register 1)....254
12.4.12 TBxCP0 (Capture register 0)....255
12.4.13 TBxCP1 (Capture register 1)....255
12.4.14 TBxDMA(DMA request enable register)....256
12.5 Description of Operations for Each Circuit....257
12.5.1 Prescaler....257
12.5.2 Up-counter (UC)....262

12.5.3 Timer registers (TBxRG0, TBxRG1)....262
12.5.4 Capture....263
12.5.5 Capture registers (TBxCP0, TBxCP1)....263
12.5.6 Up-counter capture register (TBxUC)....263
12.5.7 Comparators (CP0, CP1)....263
12.5.8 Timer Flip-flop (TBxFF0)....263
12.5.9 Capture interrupt (INTCAPx0, INTCAPx1)....263
12.6 Description of Operations for Each Mode....264
12.6.1 16-bit Interval Timer Mode....264
12.6.2 16-bit Event Counter Mode....264
12.6.3 16-bit PPG (Programmable Pulse Generation) Output Mode....265
12.6.4 Timer synchronous mode....266
12.6.5 External trigger count start mode....267
12.7 Applications using the Capture Function....268
12.7.1 One-shot pulse output triggered by an external pulse....268
12.7.2 Frequency measurement....270
12.7.3 Pulse width measurement....270
12.7.4 Time Difference Measurement....271

13. USB Device Controller (USBD)

13.1 Outline....273
13.2 System Structure....274
13.2.1 AHB Bus Bridge (UDC2AB)....275
13.2.1.1 Functions and Features
13.2.1.2 Configuration
13.2.1.3 Clock Domain
13.2.2 Toshiba USB-Spec2.0 Device Controller (UDC2)....280
13.2.2.1 Features and Functions
13.2.2.2 Specifications of Flags
13.2.2.3 Commands to EP
13.3 How to connect with the USB bus....287
13.4 Registers....288
13.4.1 UDC2AB Register....288
13.4.1.1 UDC2AB Register list
13.4.1.2 UDFSINTSTS (Interrupt Status Register)
13.4.1.3 UDFSINTENB(Interrupt Enable Register)
13.4.1.4 UDFSMWTOUT(Master Write Timeout Register)
13.4.1.5 UDFSC2STSET(UDC2 Setting Register)
13.4.1.6 UDFSMSTSET(DMAC Setting Register)
13.4.1.7 UDFSDMACRDREQ(DMAC Read Request Register)
13.4.1.8 UDFSDMACRDVL(DMAC Read Value Register)
13.4.1.9 UDFSUDC2RDREQ(UDC2 Read Request Register)
13.4.1.10 UDFSUDC2RDVL(UDC2 Read Value Register)
13.4.1.11 UDFSARBTSET(Arbiter Setting Register)
13.4.1.12 UDFSMWSADR(Master Write Start Address Register)
13.4.1.13 UDFSMWEADR(Master Write End Address Register)
13.4.1.14 UDFSMWCADR(Master Write Current Address Register)
13.4.1.15 UDFSMWAHBADR(Master Write AHB Address Register)
13.4.1.16 UDFSMRSADR(Master Read Start Address Register)
13.4.1.17 UDFSMREADR(Master Read End Address Register)
13.4.1.18 UDFSMRCADR(Master Read Current Address Register)
13.4.1.19 UDFSMRAHBADR(Master Read AHB Address Register)
13.4.1.20 UDFSPWCTL(Power Detect Control Register)
13.4.1.21 UDFSMSTSTS(Master Status Register)
13.4.1.22 UDFSTOUTCNT(Timeout Count Register)
13.4.2 UDC2 Register....311
13.4.2.1 UDC2 Registers
13.4.2.2 How to access the UDC2 register
13.4.2.3 UDFS2ADR(Address-State register)
13.4.2.4 UDFS2FRM(Frame register)
13.4.2.5 UDFS2CMD(Command register)
13.4.2.6 UDFS2BRQ(bRequest-bmRequest Type register)
13.4.2.7 UDFS2WVL(wValue register)
13.4.2.8 UDFS2WIDX(wIndex register)
13.4.2.9 UDFS2WLGTH(wLength register)
13.4.2.10 UDFS2INT(INT register)
13.4.2.11 UDFS2INTEP(INT_EP register)

13.4.2.12 UDFS2INTEPMSK(INT_EP_MASK register)
13.4.2.13 UDFS2INTRX0(INT_RX_DATA0 register)
13.4.2.14 UDFS2INTNAK(INT_NAK register)
13.4.2.15 UDFS2INTNAKMSK(INT_NAK_MASK register)
13.4.2.16 UDFS2EP0MSZ(EP0_MaxPacketSize register)
13.4.2.17 UDFS2EP0STS(EP0_Status register)
13.4.2.18 UDFS2EP0DSZ(EP0_Datasize register)
13.4.2.19 UDFS2EP0FIFO(EP0_FIFO register)
13.4.2.20 UDFS2EPxMSZ(EPx_MaxPacketSizeRegister)
13.4.2.21 UDFS2EPxSTS(EPx_Status register)
13.4.2.22 UDFS2EPxDSZ(EPx_Datasize register)
13.4.2.23 UDFS2EPxFIFO(EPx_FIFO register)
13.5 Description of UDC2AB operation....341
13.5.1 Reset....341
13.5.2 Interrupt Signals....342
13.5.2.1 INTUSB Interrupt Signal
13.5.2.2 INTUSBWKUP Interrupt
13.5.3 Operation Sequence....344
13.5.4 Master Transfer Operation....346
13.5.4.1 Master Read transfer
13.5.4.2 Master Write transfer
13.5.5 USB Power Management Control....350
13.5.5.1 Connection Diagram of Power Management Control Signal
13.5.5.2 Sequence of USB Bus Power (VBUS) Connection/Disconnection
13.5.6 USB Reset....351
13.5.7 Suspend / Resume....352
13.5.7.1 Shift to the suspended state
13.5.7.2 Resuming from suspended state (resuming from the USB host)
13.5.7.3 Resuming from the suspend state (disconnect)
13.5.7.4 Remote wakeup from the suspended state
13.6 USB Device Response....359
13.7 Flow of Control in Transfer of EPs....361
13.7.1 EP0....361
13.7.1.1 Control-RD transfer
13.7.1.2 Control-WR transfer (without DATA-Stage)
13.7.1.3 Control-WR transfer (with DATA-Stage)
13.7.1.4 Example of using the INT_STATUS_NAK flag
13.7.1.5 Processing when standard request
13.7.2 EPs other than EP0....375
13.8 Suspend/Resume State....376
13.8.1 Shift to the suspended state....376
13.8.2 Resuming from suspended state....376
13.8.2.1 Resuming by an output from the host
13.8.2.2 Resuming by way remote wakeup from UDC2
13.9 USB-Spec2.0 Device Controller Appendix....377
13.9.1 Appendix A System Power Management....377
13.9.1.1 Connect / Disconnect Operations
13.9.1.2 Reset Operation
13.9.1.3 Suspend Operation
13.9.1.4 Resume Operation
13.9.2 Appendix B About Setting an Odd Number of Bytes as MaxPacketSize....383
13.9.2.1 Setting an odd number in the UDFS2EPxMSZ
13.9.3 Appendix C Isochronous Translator....386
13.9.3.1 Accessing an EP using Isochronous transfer
13.9.3.2 Restrictions on command usage to EP when using Isochronous transfer

14. Serial Channel (SIO/UART)

14.1 Overview....387
14.2 Difference in the Specifications of SIO Modules....387
14.3 Configuration....388
14.4 Registers Description....389
14.4.1 Registers List in Each Channel....389
14.4.2 SCxEN (Enable Register)....390
14.4.3 SCxBUF (Buffer Register)....391
14.4.4 SCxCR (Control Register)....392

14.4.5 SCxMOD0 (Mode Control Register 0)....393
14.4.6 SCxMOD1 (Mode Control Register 1)....394
14.4.7 SCxMOD2 (Mode Control Register 2)....395
14.4.8 SCxBRCR (Baud Rate Generator Control Register), SCxBRADD (Baud Rate Generator Control Register 2)....397
14.4.9 SCxFCNF (FIFO Configuration Register)....399
14.4.10 SCxRFC (RX FIFO Configuration Register)....400
14.4.11 SCxTFC (TX FIFO Configuration Register) (Note2)....401
14.4.12 SCxRST (RX FIFO Status Register)....402
14.4.13 SCxTST (TX FIFO Status Register)....403
14.4.14 SCxDMA (DMA request enable register)....404
14.5 Operation in Each Mode....405
14.6 Data Format....406
14.6.1 Data Format List....406
14.6.2 Parity Control....407
14.6.2.1 Transmission
14.6.2.2 Receiving Data
14.6.3 STOP Bit Length....407
14.7 Clock Control....408
14.7.1 Prescaler....408
14.7.2 Serial Clock Generation Circuit....412
14.7.2.1 Baud Rate Generator
14.7.2.2 Clock Selection Circuit
14.8 Transmit/Receive Buffer and FIFO....416
14.8.1 Configuration....416
14.8.2 Transmit/Receive Buffer....416
14.8.3 FIFO....417
14.9 Status Flag....417
14.10 Error Flag....417
14.10.1 OERR Flag....418
14.10.2 PERR Flag....418
14.10.3 FERR Flag....418
14.11 Receive....419
14.11.1 Receive Counter....419
14.11.2 Receive Control Unit....419
14.11.2.1 I/O interface mode
14.11.2.2 UART Mode
14.11.3 Receive Operation....419
14.11.3.1 Receive Buffer
14.11.3.2 Receive FIFO Operation
14.11.3.3 I/O interface mode with SCLK output
14.11.3.4 Read Received Data
14.11.3.5 Wake-up Function
14.11.3.6 Overrun Error
14.12 Transmission....423
14.12.1 Transmission Counter....423
14.12.2 Transmission Control....423
14.12.2.1 I/O Interface Mode
14.12.2.2 UART Mode
14.12.3 Transmit Operation....423
14.12.3.1 Operation of Transmission Buffer
14.12.3.2 Transmit FIFO Operation
14.12.3.3 I/O interface Mode/Transmission by SCLK Output
14.12.3.4 Under-run error
14.13 Handshake function....427
14.14 Interrupt/Error Generation Timing....428
14.14.1 RX Interrupts....428
14.14.1.1 Single Buffer / Double Buffer
14.14.1.2 FIFO
14.14.2 TX interrupts....429
14.14.2.1 Single Buffer / Double Buffer
14.14.2.2 FIFO
14.14.3 Error Generation....430
14.14.3.1 UART Mode
14.14.3.2 IO Interface Mode
14.15 Software Reset....430
14.16 DMA request....430

14.17 Operation in Each Mode.....431
14.17.1 Mode 0 (I/O interface mode).....431
14.17.1.1 Transmitting Data
14.17.1.2 Receive
14.17.1.3 Transmit and Receive (Full-duplex)
14.17.2 Mode 1 (7-bit UART mode).....442
14.17.3 Mode 2 (8-bit UART mode).....442
14.17.4 Mode 3 (9-bit UART mode).....443
14.17.4.1 Wake up function
14.17.4.2 Protocol

15. Serial Bus Interface (I2C/SIO)

15.1 Configuration.....446
15.2 Register.....447
15.2.1 Registers for each channel.....447
15.3 I2C Bus Mode Data Format.....448
15.4 Control Registers in the I2C Bus Mode.....449
15.4.1 SBIxCR0(Control register 0).....449
15.4.2 SBIxCR1(Control register 1).....450
15.4.3 SBIxCR2(Control register 2).....452
15.4.4 SBIxSR (Status Register).....453
15.4.5 SBIxBR0(Serial bus interface baud rate register 0).....454
15.4.6 SBIxDBR (Serial bus interface data buffer register).....454
15.4.7 SBIxI2CAR (I2Cbus address register).....455
15.5 Control in the I2C Bus Mode.....456
15.5.1 Serial Clock.....456
15.5.1.1 Clock source
15.5.1.2 Clock Synchronization
15.5.2 Setting the Acknowledgement Mode.....457
15.5.3 Setting the Number of Bits per Transfer.....457
15.5.4 Slave Addressing and Address Recognition Mode.....457
15.5.5 Operating mode.....457
15.5.6 Configuring the SBI as a Transmitter or a Receiver.....458
15.5.7 Configuring the SBI as a Master or a Slave.....458
15.5.8 Generating Start and Stop Conditions.....458
15.5.9 Interrupt Service Request and Release.....459
15.5.10 Arbitration Lost Detection Monitor.....460
15.5.11 Slave Address Match Detection Monitor.....461
15.5.12 General-call Detection Monitor.....461
15.5.13 Last Received Bit Monitor.....461
15.5.14 Data Buffer Register (SBIxDBR).....461
15.5.15 Baud Rate Register (SBIxBR0).....462
15.5.16 Software Reset.....462
15.6 Data Transfer Procedure in the I2C Bus ModeI2C.....463
15.6.1 Device Initialization.....463
15.6.2 Generating the Start Condition and a Slave Address.....463
15.6.2.1 Master mode
15.6.2.2 Slave mode
15.6.3 Transferring a Data Word.....465
15.6.3.1 Master mode (<MST> = "1")
15.6.3.2 Slave mode (<MST> = "0")
15.6.4 Generating the Stop Condition.....470
15.6.5 Restart Procedure.....470
15.6.6 Data Transfer using DMA.....472
15.6.6.1 How to transfer data in master mode
15.6.6.2 How to transfer data in slave mode
15.7 Control register of SIO mode.....481
15.7.1 SBIxCR0(control register 0).....481
15.7.2 SBIxCR1(Control register 1).....482
15.7.3 SBIxDBR (Data buffer register).....483
15.7.4 SBIxCR2(Control register 2).....484
15.7.5 SBIxSR (Status Register).....485
15.7.6 SBIxBR0 (Baud rate register 0).....486

15.8 Control in SIO mode 487
15.8.1 Serial Clock 487
15.8.1.1 Clock source
15.8.1.2 Shift Edge
15.8.2 Transfer Modes 489
15.8.2.1 8-bit transmit mode
15.8.2.2 8-bit receive mode
15.8.2.3 8-bit transmit/receive mode
15.8.2.4 Data retention time of the last bit at the end of transmission

16. Analog/Digital Converter (ADC)

16.1 Outline 495
16.2 Configuration 496
16.3 Registers 497
16.3.1 Register list 497
16.3.2 ADCLK (Conversion Clock Setting Register) 498
16.3.3 ADMOD0 (Mode Control Register 0) 500
16.3.4 ADMOD1 (Mode Control Register 1) 501
16.3.5 ADMOD2 (Mode Control Register 2) 503
16.3.6 ADMOD3 (Mode Control Register 3) 504
16.3.7 ADMOD4 (Mode Control Register 4) 505
16.3.8 ADMOD5 (Mode Control Register 5) 507
16.3.9 ADMOD6 (Mode Control Register 6) 508
16.3.10 ADMOD7 (Mode Control Register7) 509
16.3.11 ADCMPCR0 (Monitor Control Register 0) 510
16.3.12 ADCMPCR1 (AD Monitor Control Register 1) 512
16.3.13 ADCMP0 (AD Conversion Result Comparison Register 0) 513
16.3.14 ADCMP1 (AD Conversion Result Comparison Register 1) 514
16.3.15 ADREG00 to ADREG11 (Normal Conversion Result Register 00 to 11) 515
16.3.16 ADREGSP (Highest-priority Conversion Result Register) 516
16.4 Description of Operations 517
16.4.1 Analog Reference Voltage 517
16.4.2 AD Conversion Mode 517
16.4.2.1 Normal AD Conversion
16.4.2.2 Highest-priority AD conversion
16.4.3 AD monitor function 518
16.4.4 Selecting the Input Channel 520
16.4.5 AD Conversion Details 521
16.4.5.1 Starting AD Conversion
16.4.5.2 AD Conversion
16.4.5.3 Highest-priority AD conversion requests during normal AD conversion
16.4.5.4 Stopping Repeat Conversion Mode
16.4.5.5 Reactivating normal AD conversion
16.4.5.6 Conversion completion
16.4.5.7 Interrupt generation timings and AD conversion result storage register

17. Flash Memory Operation

17.1 Features 529
17.1.1 Memory Size and Configuration 529
17.1.2 Function 530
17.1.3 Operation Mode 530
17.1.3.1 Mode Description
17.1.3.2 Mode Determination
17.1.4 Memory Map 532
17.1.5 Protect/Security Function 533
17.1.5.1 Protect Function
17.1.5.2 Security Function
17.1.6 Register 534
17.1.6.1 Register List
17.1.6.2 FCFLCS (Flash control register)
17.1.6.3 FCSECBIT (Security bit register)

17.2 Detail of Flash Memory 536
17.2.1 Function 536
17.2.2 Operation Mode of Flash Memory 536
17.2.3 Hardware Reset 536
17.2.4 How to Execute Command 537
17.2.5 Command Description 537
17.2.5.1 Automatic Page Program
17.2.5.2 Automatic Chip Erase
17.2.5.3 Automatic Block Erase
17.2.5.4 Automatic Protect Bit Program
17.2.5.5 Auto Protect Bit Erase
17.2.5.6 ID-Read
17.2.5.7 Read Command and Read/reset Command (Software Reset)
17.2.6 Command Sequence 540
17.2.6.1 Command Sequence List
17.2.6.2 Address Bit Configuration in the Bus Cycle
17.2.6.3 Block Address(BA)
17.2.6.4 How to Specify Protect Bit (PBA)
17.2.6.5 ID-Read Code (IA, ID)
17.2.6.6 Example of Command Sequence
17.2.7 Flowchart 546
17.2.7.1 Automatic Program
17.2.7.2 Automatic Erase
17.3 How to Reprogram Flash using Single Boot Mode 548
17.3.1 Mode Setting 548
17.3.2 Interface Specification 548
17.3.3 Restrictions on Internal Memories 549
17.3.4 Operation Command 549
17.3.4.1 RAM Transfer
17.3.4.2 Flash Memory Chip Erase and Protect Bit Erase
17.3.5 Common Operation regardless of Command 550
17.3.5.1 Serial Operation Mode Determination
17.3.5.2 Acknowledge Response Data
17.3.5.3 Password Determination
17.3.5.4 CHECK SUM Calculation
17.3.6 Transfer Format at RAM Transfer 556
17.3.7 Transfer Format of Flash memory Chip Erase and Protect Bit Erase 557
17.3.8 Boot Program Whole Flowchart 560
17.3.9 Reprogramming Procedure of Flash using reprogramming algorithm in the on-chip BOOT ROM 561
17.3.9.1 Step-1
17.3.9.2 Step-2
17.3.9.3 Step-3
17.3.9.4 Step-4
17.3.9.5 Step-5
17.3.9.6 Step-6
17.4 Programming in the User Boot Mode 564
17.4.1 (1-A) Procedure that a Programming Routine Stored in Flash memory 564
17.4.1.1 Step-1
17.4.1.2 Step-2
17.4.1.3 Step-3
17.4.1.4 Step-4
17.4.1.5 Step-5
17.4.1.6 Step-6
17.4.2 (1-B) Procedure that a Programming Routine is transferred from External Host 568
17.4.2.1 Step-1
17.4.2.2 Step-2
17.4.2.3 Step-3
17.4.2.4 Step-4
17.4.2.5 Step-5
17.4.2.6 Step-6

18. Port Section Equivalent Circuit Schematic

18.1 PA0 to 7,PB0 to 7 573
18.2 PC0 to 2, PD0 to 7, PE0 to 7, PF1 to 7, PG0 to 4, PH0 to 4, PI0 to 7 573
18.3 PG5 574
18.4 PJ0 to 7,PK0 to 3 574

18.5 PF0.....575
18.6 X1,X2.....575
18.7 RESET,NMI.....575
18.8 MODE.....576
18.9 FTEST3.....576
18.10 AVREFH,AVREFL.....576

19. Electrical Characteristics

19.1 Absolute Maximum Ratings.....577
19.2 DC Electrical Characteristics (1/3).....578
19.3 DC Electrical Characteristics (2/3).....579
19.4 DC Electrical Characteristics (3/3).....580
19.5 12-bit ADC Electrical Characteristics.....581
19.6 AC Electrical Characteristics.....582
19.6.1 AC measurement condition.....582
19.6.2 Serial Channel (SIO/UART).....582
19.6.2.1 I/O Interface mode
19.6.3 Serial Bus Interface (I2C/SIO).....585
19.6.3.1 I2C Mode
19.6.3.2 Clock-Synchronous 8-Bit SIO mode
19.6.4 Event Counter.....588
19.6.5 Capture.....588
19.6.6 External Interrupt.....588
19.6.7 NMI.....589
19.6.8 SCOUT Pin AC Characteristic.....589
19.6.9 ADTRG Tringger Input Pin AC Characteristic.....589
19.6.10 USB Timing.....589
19.6.11 Debug Communication.....590
19.6.11.1 SWD Interface
19.6.11.2 JTAG Interface
19.6.12 ETM Trace.....591
19.6.13 On chip oscillator.....591
19.6.14 External clock input.....591
19.6.15 Flash Memory Characteristics.....592
19.7 Recommended Oscillation Circuit.....593
19.7.1 Ceramic oscillator.....593
19.7.2 Crystal oscillator.....593
19.7.2.1 Precautions for designing printed circuit board
19.8 Handling Precaution.....594
19.8.1 Voltage level of power supply at power-on.....594

20. Package Dimensions

TMPM365FYXBG

The TMPM365FYXBG is a 32-bit RISC microprocessor series with an ARM Cortex™-M3 microprocessor core.

Product Name	ROM (FLASH)	RAM	Package
TMPM365FYXBG	256 Kbyte	24 Kbyte	LFBGA105

Features of the TMPM365FYXBG 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 Flash ROM : 256 Kbyte
   - On chip SRAM : 24 Kbyte

3. DMA controller(DMAC): 2 channels

   Memory / Peripheral function / External memory

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

5. Watchdog timer (WDT): 1 channel

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

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

   Either UART mode or synchronous communication mode can be selected (4byte FIFO)

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

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

8. USB Device controller (USBD) : 1channel
   - Compliance Universal Serial Bus Specification Rev.2.0.
   - 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.

9. 12-bit AD converter (ADC): 12channels
   - Start up with 16-bit timer / Start up with an external trigger input
   - Fixed-channel / Scan channel mode
   - AD monitoring 2ch
   - Conversion time 1μs. (@fsys = 40MHz),1.67μs. (@fsys = 48MHz)

10. Interrupt source
    - Internal : 47 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.

11. Non-maskable interrupt (NMI)

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

12. Input/ output ports (PORT): 74 pins (One 5V tolerant input include)

13. Low power consumption mode

    Low power consumption mode: IDLE, STOP1

14. Clock generator (CG)
    - On-chip PLL (x8, the system clock is devided by 2)

- Clock gear function: The high-speed clock can be divided into 1/1, 1/2, 1/4, 1/8 or 1/16.

15. Endian

Little endian

16. Debug interface

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

17. JTAG interface

Boundary scan

18. Maximum operating frequency: 48 MHz

19. Operating voltage range

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

3.0 V to 3.45 V (when USB is used)

20. Temperature range

- -40 to 85 degrees (except during Flash writing/ erasing)
- 0 to 70 degrees (during Flash writing/ erasing)

21. Package

LFBGA105 (9mm × 9mm, 0.5mm pitch)

1.2 Block Diagram

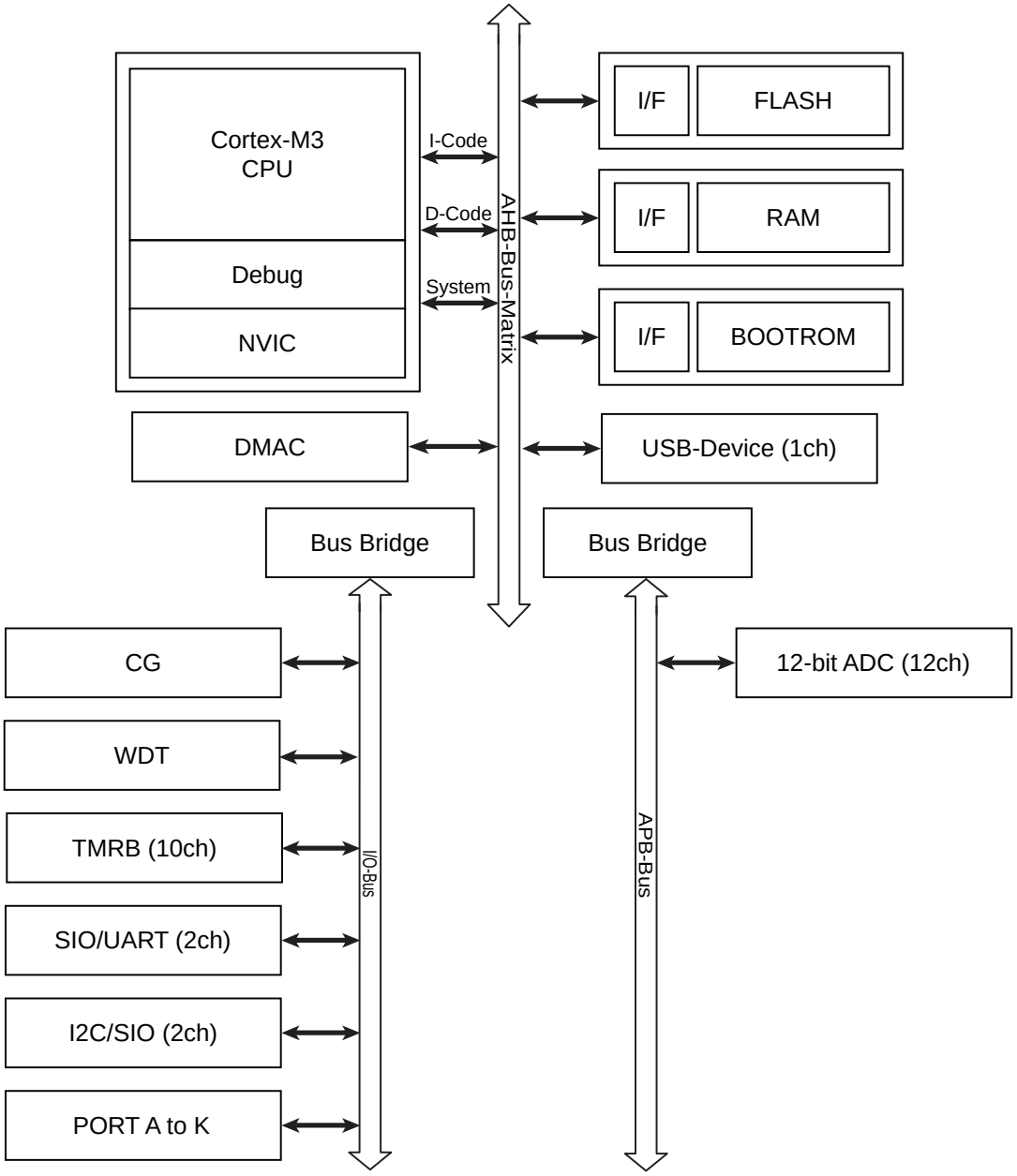


Figure 1-1 TMPM365FYXBGBlock Diagram

1.3 Pin Layout (Top view)

Figure 1-2 shows the pin layout of TMPM365FYXBG.

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15
C1	C2	—	—	—	—	—	—	—	—	—	—	—	C14	C15
D1	D2	—	—	—	—	—	—	—	—	—	—	—	D14	D15
E1	E2	—	—	—	—	—	—	—	—	—	—	—	E14	E15
F1	F2	—	—	—	—	—	—	—	—	—	—	—	F14	F15
G1	G2	—	—	—	—	—	—	—	—	—	—	—	G14	G15
H1	H2	—	—	—	—	—	—	—	—	—	—	—	H14	H15
J1	J2	—	—	—	—	—	—	—	—	—	—	—	J14	J15
K1	K2	—	—	—	—	—	—	—	—	—	—	—	K14	K15
L1	L2	—	—	—	—	—	—	—	—	—	—	—	L14	L15
M1	M2	—	—	—	—	—	—	—	—	—	—	—	M14	M15
N1	N2	N3	—	—	—	—	—	—	—	—	—	—	N14	N15
P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15

Figure 1-2 Pin Layout (BGA105)

1.4 Pin names and Functions

Table 1-1 and Table 1-2 sort the input and output pins of the TMPM365FYXBG by pin or port. Each table includes alternate pin names and functions for multi-function pins.

1.4.1 Sorted by Pin

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

Type	Pin No.	Pin Name	Input/ Output	Function
Function	A1	PK0 AIN08 INT2 TB1IN0	Input Input Input Input	I/O port Analog input External interrupt pin Inputting the TMRB capture trigger
Function	A2	PJ7 AIN07 INT9 TB0IN1	I/O Input Input Input	I/O port Analog input External interrupt pin Inputting the TMRB capture trigger
Function	A3	PJ5 AIN05	I/O Input	I/O port Analog input
Function	A4	PJ3 AIN03	I/O Input	I/O port Analog input
Function	A5	PJ1 AIN01	I/O Input	I/O port Analog input
PS	A6	RVDD3	−	Power supply pin for internal regulator
PS	A7	RVSS	−	GND pin for internal regulator
Function	A8	REGOUT	Output	Output pin for regulator
Function	A9	REGIN	Input	Input pin for regulator
Function	A10	PE2 SCLK0 TB2OUT $\overline{CTS0}$	I/O I/O Output Input	I/O portr Serial clock input/ output TMRB output Handshake input pin
PS	A11	DVDD3A	−	Power supply pin
Clock	A12	X1	Input	Connected to a high-speed oscillator / External clock input pin
PS	A13	DVSSC	−	GND pin for oscillator
Clock	A14	X2	Output	Connected to a high-speed oscillator
Function	A15	PE0 TXD0	I/O Output	I/O port Sending serial data
Function	B1	PK1 AIN09 INT3 TB1IN1	I/O Input Input Input	I/O port Analog input External interrupt pin Inputting the TMRB capture trigger
Function	B2	PJ6 AIN06 TB0IN0	I/O Input Input	I/O port Analog input Inputting the TMRB capture trigger

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

Type	Pin No.	Pin Name	Input/ Output	Function
Function	B3	PJ4 AIN04	I/O Input	I/O port Analog input
Function	B4	PJ2 AIN02	I/O Input	I/O port Analog input
Function	B5	PJ0 AIN00	I/O Input	I/O port Analog input
PS	B6	RVDD3	−	Power supply pin for internal regulator
Function	B7	PE7 INT4	I/O Input	I/O port External interrupt pin
Function	B8	PE6 SCK1	I/O I/O	I/O port Inputting and outputting a clock if the serial bus interface operates in SIO mode
Function	B9	PE5 SCL1/SI1	I/O I/O	I/O port Clock pin in I2C mode / Data pin in SIO mode
Function	B10	PE4 SDA1/SO1	I/O I/O	I/O port Data pin in I2C mode / Data pin in SIO mode
Function	B11	PE3 INT5 TB3OUT	I/O Input Output	I/O port External interrupt pin TMRB output
Function	B12	$\overline{NMI}$	Input	Non-maskable interrupt (note) With a noise filter (about 30ns (typical value))
PS	B13	DVSSA	−	GND pin
Control	B14	MODE	Input	Mode pin: (note) MODE pin must be connected to GND.
Function	B15	$\overline{RESET}$	Input	Reset input pin (note) With a pull-up and a noise filter (about 30ns (typical value))
Function	C1	PK2 AIN10 TB6IN0	I/O Input Input	I/O port Analog input Inputting the TMRB capture trigger
Function	C2	PK3 AIN11 TB6IN1	I/O Input Input	I/O port Analog input Inputting the TMRB capture trigger
Function	C14	PE1 RXD0	I/O Input	I/O port Receiving serial data
Function	C15	PD0 TB7OUT	I/O Output	I/O port TMRB output
PS	D1	AVREFH	Input	Supplying the AD converters with a reference power supply. (note) AVREFH must be connected to power supply even if AD converters are not used.
PS	D2	AVDD3	Input	Supplying the AD converters with a power supply. (note) AVDD3 must be connected to power supply even if AD converters are not used.
Function	D14	PD2 TB9OUT	I/O Output	I/O port TMRB output

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

Type	Pin No.	Pin Name	Input/ Output	Function
Function	D15	PD1 TB8OUT	I/O Output	I/O port TMRB output
PS	E1	AVREFL	Input	Supplying the AD converters with a reference GND. (note) AVREFL must be connected to GND even if AD converters are not used.
PS	E2	AVSS	Input	AD converters: GND pin. (note) AVSS must be connected to GND even if the AD converters are not used.
Function	E14	PD4	I/O	I/O port
Function	E15	PD3 $\overline{\text{ADTRG}}$	I/O Input	I/O port AD trigger input
PS	F1	DVSSA	−	GND pin
Control	F2	BSC	Input	Boundary scan control pin
PS	F14	DVDD3C	−	Power supply pin for USB
PS	F15	DVDD3C	−	Power supply pin for USB
PS	G1	DVDD3A	−	Power supply pin
Function/ Debug	G2	PI7 $\overline{\text{TRST}}$	I/O Input	I/O port Debug pin
PS	G14	DVSS3C	−	GND pin for USB
Function	G15	D+	I/O	USB pin (D+)
Function/ Debug	H1	PI2 TRACECLK	I/O Output	I/O port Debug pin
Function/ Debug	H2	PI6 TDI	I/O Input	I/O port Debug pin
PS	H14	DVSS3C	−	GND pin for USB
Function	H15	D−	I/O	USB pin (D-)
Function/ Debug	J1	PI1 TRACEDATA0	I/O Output	I/O port Debug pin
Function/ Debug	J2	PI5 TDO/SWV	I/O Output	I/O port Debug pin
PS	J14	DVSSA	−	GND pin
PS	J15	DVDD3A	−	Power supply pin
Function/ Debug	K1	PI0 TRACEDATA1	I/O Output	I/O port Debug pin
Function/ Debug	K2	PI4 TMS/SWDIO	I/O I/O	I/O port Debug pin
Function	K14	PD6	I/O	I/O port

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

Type	Pin No.	Pin Name	Input/ Out-put	Function
Function	K15	PD5	I/O	I/O port
Function/ Debug	L1	PH0 TRACEDATA2	I/O Out-put	I/O port Debug pin
Function/ Debug	L2	PI3 TCK/SWCLK	I/O Input	I/O port Debug pin
Function	L14	PD7 SCOUT	I/O Out-put	I/O port System clock Output
Function	L15	PB7	I/O	I/O port
Function/ Debug	M1	PH1 TRACEDATA3	I/O Out-put	I/O port Debug pin
Function	M2	PH2 TB4OUT	I/O Out-put	I/O port TMRB output
Function	M14	PB5	I/O	I/O port
Function	M15	PB6	I/O	I/O port
Function	N1	PH3 TB5OUT	I/O Out-put	I/O port TMRB output
Function	N2	PH4 INT8	I/O Input	I/O port External interrupt pin
Control	N3	FTEST3	–	TEST pin: (note) TEST pin must be left OPEN.
Function	N14	PB3	I/O	I/O port
Function	N15	PB4	I/O	I/O port
Function	P1	PG5 INT1 USBPON	I/O Input Input	I/O port (5V tolerant input) (Note) External interrupt pin USB conection detection pin (VBUS Detect)
Function	P2	PG4 TB4IN1	I/O Input	I/O port Inputting the TMRB capture trigger
Function	P3	PG2 SCK0 TB3IN1	I/O I/O Input	I/O port Inputting and outputting a clock if the serial bus interface operates in SIO mode Inputting the TMRB capture trigger
Function	P4	PG1 SCL0/SI0 TB3IN0	I/O I/O Input	I/O port Clock pin in I2C mode / Data pin in SIO mode Inputting the TMRB capture trigger
Function	P5	PG0 SDA0/SO0	I/O I/O	I/O port Data pin in I2C mode / Data pin in SIO mode
Function	P6	PF2	I/O	I/O port

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

Type	Pin No.	Pin Name	Input/ Output	Function
Function	P7	PF4 INT6 TB5IN0	I/O Input Input	I/O port External interrupt pin Inputting the TMRB capture trigger
Function	P8	PF5 INT7 TB5IN1	I/O Input Input	I/O port External interrupt pin Inputting the TMRB capture trigger
Function	P9	PF6	I/O	I/O port
Function	P10	PF7	I/O	I/O port
Function	P11	PA2	I/O	I/O port
Function	P12	PA4	I/O	I/O port
Function	P13	PA6	I/O	I/O port
Function	P14	PB1	I/O	I/O port
Function	P15	PB2	I/O	I/O port
Function	R1	PG3 INT0 TB4IN0	I/O Input Input	I/O port External interrupt pin Inputting the TMRB capture trigger
Function	R2	PC0 TXD1 TB2IN0	I/O Output Input	I/O port Sending serial data Inputting the TMRB capture trigger
Function	R3	PC1 RXD1 TB2IN1	I/O Input Input	I/O port Receiving serial data Inputting the TMRB capture trigger
Function	R4	PC2 SCLK1 TB0OUT $\overline{CTS1}$	I/O I/O Output Input	I/O port Serial clock input/ output TMRB output Handshake input pin
Function/ Control	R5	PF0 $\overline{BOOT}$ TB6OUT	Output Input Output	Output port Setting a single boot mode: This pin goes into single boot mode by sampling "Low" at the rise of a $\overline{RESET}$ signal. TMRB output
Function	R6	PF1	I/O	I/O port
Function	R7	PF3	I/O	I/O port
PS	R8	DVSSA	−	GND pin
PS	R9	DVDD3A	−	Power supply pin
Function	R10	PA0	I/O	I/O port
Function	R11	PA1	I/O	I/O port

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

Type	Pin No.	Pin Name	Input/ Out- put	Function
Function	R12	PA3	I/O	I/O port
Function	R13	PA5	I/O	I/O port
Function	R14	PA7	I/O	I/O port
Function	R15	PB0	I/O	I/O port

Note: Only when input is enabled, these pins tolerate 5V inputs. Note that these pins cannot 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 (1/6)

PORT	Type	Pin No.	Pin Name	Input/ Out-put	Function
PORT A	Function	R10	PA0	I/O	I/O port
PORT A	Function	R11	PA1	I/O	I/O port
PORT A	Function	P11	PA2	I/O	I/O port
PORT A	Function	R12	PA3	I/O	I/O port
PORT A	Function	P12	PA4	I/O	I/O port
PORT A	Function	R13	PA5	I/O	I/O port
PORT A	Function	P13	PA6	I/O	I/O port
PORT A	Function	R14	PA7	I/O	I/O port
PORT B	Function	R15	PB0	I/O	I/O port
PORT B	Function	P14	PB1	I/O	I/O port
PORT B	Function	P15	PB2	I/O	I/O port
PORT B	Function	N14	PB3	I/O	I/O port
PORT B	Function	N15	PB4	I/O	I/O port
PORT B	Function	M14	PB5	I/O	I/O port
PORT B	Function	M15	PB6	I/O	I/O port
PORT B	Function	L15	PB7	I/O	I/O port
PORT C	Function	R2	PC0 TXD1 TB2IN0	I/O Out-put Input	I/O port Sending serial data Inputting the TMRB capture trigger
PORT C	Function	R3	PC1 RXD1 TB2IN1	I/O Input Input	I/O port Receiving serial data Inputting the TMRB capture trigger
PORT C	Function	R4	PC2 SCLK1 TB0OUT $\overline{CTS1}$	I/O I/O Out-put Input	I/O port Serial clock input/ output TMRB output Handshake input pin
PORT D	Function	C15	PD0 TB7OUT	I/O Out-put	I/O port TMRB output
PORT D	Function	D15	PD1 TB8OUT	I/O Out-put	I/O port TMRB output
PORT D	Function	D14	PD2 TB9OUT	I/O Out-put	I/O port TMRB output

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

PORT	Type	Pin No.	Pin Name	Input/ Output	Function
PORT D	Function	E15	PD3 $\overline{\text{ADTRG}}$	I/O Input	I/O port AD trigger input
PORT D	Function	E14	PD4	I/O	I/O port
PORT D	Function	K15	PD5	I/O	I/O port
PORT D	Function	K14	PD6	I/O	I/O port
PORT D	Function	L14	PD7 SCOUT	I/O Output	I/O port System clock Output
PORT E	Function	A15	PE0 TXD0	I/O Output	I/O port Sending serial data
PORT E	Function	C14	PE1 RXD0	I/O Input	GND pin Receiving serial data
PORT E	Function	A10	PE2 SCLK0 TB2OUT $\overline{\text{CTS0}}$	I/O I/O Output Input	I/O port Serial clock input/ output TMRB output Handshake input pin
PORT E	Function	B11	PE3 INT5 TB3OUT	I/O Input Output	I/O port External interrupt pin TMRB output
PORT E	Function	B10	PE4 SDA1/SO1	I/O I/O	I/O port Data pin in I2C mode / Data pin in SIO mode
PORT E	Function	B9	PE5 SCL1/SI1	I/O I/O	I/O port Clock pin in I2C mode / Data pin in SIO mode
PORT E	Function	B8	PE6 SCK1	I/O I/O	I/O port Inputting and outputting a clock if the serial bus interface operates in SIO mode
PORT E	Function	B7	PE7 INT4	I/O Input	I/O port External interrupt pin
PORT F	Function/ Control	R5	PF0 $\overline{\text{BOOT}}$ TB6OUT	Output Input Output	Output port Setting a single boot mode: This pin goes into single boot mode by sampling "Low" at the rise of a $\overline{\text{RESET}}$ signal. TMRB output
PORT F	Function	R6	PF1	Input/ O	I/O port
PORT F	Function	P6	PF2	I/O	I/O port
PORT F	Function	R7	PF3	I/O	I/O port
PORT F	Function	P7	PF4 INT6 TB5IN0	I/O Input Input	I/O port External interrupt pin Inputting the TMRB capture trigger
PORT F	Function	P8	PF5 INT7 TB5IN1	I/O Input Input	I/O port External interrupt pin Inputting the TMRB capture trigger

Table 1-2 Pin Names and Functions Sorted by Port (3/6)

PORT	Type	Pin No.	Pin Name	Input/ Output	Function
PORT F	Function	P9	PF6	I/O	I/O port
PORT F	Function	P10	PF7	I/O	I/O port
PORT G	Function	P5	PG0 SDA0/SO0	I/O I/O	I/O port Data pin in I2C mode / Data pin in SIO mode
PORT G	Function	P4	PG1 SCL0/SI0 TB3IN0	I/O I/O Input	I/O port Clock pin in I2C mode / Data pin in SIO mode Inputting the TMRB capture trigger
PORT G	Function	P3	PG2 SCK0 TB3IN1	I/O I/O Input	I/O port Inputting and outputting a clock if the serial bus interface operates in SIO mode Inputting the TMRB capture trigger
PORT G	Function	R1	PG3 INT0 TB4IN0	I/O Input Input	I/O port External interrupt pin Inputting the TMRB capture trigger
PORT G	Function	P2	PG4 TB4IN1	I/O Input	I/O port Inputting the TMRB capture trigger
PORT G	Function	P1	PG5 INT1 USBPON	I/O Input Input	I/O port (1 5V tolerant input) (Note) External interrupt pin USB conection detection pin (VBUS Detect)
PORT H	Function/ Debug	L1	PH0 TRACEDA-TA2	I/O Out-put	I/O port Debug pin
PORT H	Function/ Debug	M1	PH1 TRACEDA-TA3	I/O Out-put	I/O port Debug pin
PORT H	Function	M2	PH2 TB4OUT	I/O Out-put	I/O port TMRB output
PORT H	Function	N1	PH3 TB5OUT	I/O Out-put	I/O port TMRB output
PORT H	Function	N2	PH4 INT8	I/O Input	I/O port External interrupt pin
PORT I	Function/ Debug	K1	PI0 TRACEDA-TA1	I/O Out-put	I/O port Debug pin
PORT I	Function/ Debug	J1	PI1 TRACEDA-TA0	I/O Out-put	I/O port Debug pin
PORT I	Function/ Debug	H1	PI2 TRACECLK	I/O Out-put	I/O port Debug pin
PORT I	Function/ Debug	L2	PI3 TCK/SWCLK	I/O Input	I/O port Debug pin
PORT I	Function/ Debug	K2	PI4 TMS/SWDIO	I/O I/O	I/O port Debug pin
PORT I	Function/ Debug	J2	PI5 TDO/SWV	I/O Out-put	I/O port Debug pin

Table 1-2 Pin Names and Functions Sorted by Port (4/6)

PORT	Type	Pin No.	Pin Name	Input/ Out- put	Function
PORT I	Function/ Debug	H2	PI6 TDI	I/O Input	I/O port Debug pin
PORT I	Function/ Debug	G2	PI7 $\overline{\text{TRST}}$	I/O Input	I/O port Debug pin
PORT J	Function	B5	PJ0 AIN00	I/O Input	I/O port Analog input
PORT J	Function	A5	PJ1 AIN01	I/O I	I/O port Analog input
PORT J	Function	B4	PJ2 AIN02	I/O Input	I/O port Analog input
PORT J	Function	A4	PJ3 AIN03	I/O Input	I/O port Analog input
PORT J	Function	B3	PJ4 AIN04	I/O Input	I/O port Analog input
PORT J	Function	A3	PJ5 AIN05	I/O Input	I/O port Analog input
PORT J	Function	B2	PJ6 AIN06 TB0IN0	I/O Input Input	I/O port Analog input Inputting the TMRB capture trigger
PORT J	Function	A2	PJ7 AIN07 INT9 TB0IN1	I/O Input Input Input	I/O port Analog input External interrupt pin Inputting the TMRB capture trigger
PORT K	Function	A1	PK0 AIN08 INT2 TB1IN0	I/O Input Input Input	Input/O port Analog input External interrupt pin Inputting the TMRB capture trigger
PORT K	Function	B1	PK1 AIN09 INT3 TB1IN1	I/O Input Input Input	I/O port Analog input External interrupt pin Inputting the TMRB capture trigger
PORT K	Function	C1	PK2 AIN10 TB6IN0	I/O Input Input	I/O port Analog input Inputting the TMRB capture trigger
PORT K	Function	C2	PK3 AIN11 TB6IN1	I/O Input Input	I/O port Analog input Inputting the TMRB capture trigger
-	Function	G15	D+	I/O	USB pin (D+)
-	Function	H15	D−	I/O	USB pin (D-)
-	Function	B15	$\overline{\text{RESET}}$	Input	Reset input pin (note) With a pull-up and a noise filter (about 30ns (typical value))

Table 1-2 Pin Names and Functions Sorted by Port (5/6)

PORT	Type	Pin No.	Pin Name	Input/ Out-put	Function
-	Function	B12	$\overline{NMI}$	Input	Non-maskable interrupt (note) With a noise filter (about 30ns (typical value))
-	Control	B14	MODE	Input	Mode pin: (note) MODE pin must be connected to GND.
-	Control	N3	FTEST3	−	TEST pin: (note) TEST pin must be left OPEN.
-	Control	F2	BSC	Input	Boundary scan control pin
-	Clock	A12	X1	Input	Connected to a high-speed oscillator / External clock input pin
-	Clock	A14	X2	Out-put	Connected to a high-speed oscillator
-	PS	G1	DVDD3A	−	Power supply pin
-	PS	R9	DVDD3A	−	Power supply pin
-	PS	J15	DVDD3A	−	Power supply pin
-	PS	A11	DVDD3A	−	Power supply pin
-	PS	F1	DVSSA	−	GND pin
-	PS	R8	DVSSA	−	GND pin
-	PS	J14	DVSSA	−	GND pin
-	PS	B13	DVSSA	−	GND pin
-	PS	A6	RVDD3	−	Power supply pin for internal regulator
-	PS	B6	RVDD3	−	Power supply pin for internal regulator
-	PS	A7	RVSS	−	GND pin for internal ragulator
-	PS	A13	DVSSC	−	GND pin for oscillator
-	PS	F14	DVDD3C	−	Power supply pin for USB
-	PS	F15	DVDD3C	−	Power supply pin for USB
-	PS	G14	DVSS3C	−	GND pin for USB
-	PS	H14	DVSS3C	−	GND pin for USB
-	PS	D1	AVREFH	Input	Supplying the AD converters with a reference power supply. (note) AVREFH must be connected to power supply even if AD converters are not used.
-	PS	E1	AVREFL	Input	Supplying the AD converters with a reference GND. (note) AVREFL must be connected to GND even if AD converters are not used.

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

PORT	Type	Pin No.	Pin Name	Input/ Out-put	Function
-	PS	D2	AVDD3	Input	Supplying the AD converters with a power supply. (note) AVDD3 must be connected to power supply even if AD converters are not used.
-	PS	E2	AVSS	Input	AD converters: GND pin. (note) AVSS must be connected to GND even if the AD converters are not used.
-	PS	A8	REGOUT	Out-put	Regulator output pin
-	PS	A9	REGIN	Input	Regulator input pin

Note: Only when input is enabled, these pins tolerate 5V inputs. Note that these pins cannot 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 ro 3.45 V)	G1,R9,J15,A11	PA,PB,PC,PD,PE,PF,PG, PH, PI, X1, X2, FTEST3, $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC
AVDD3		D2	PJ, PK
RVDD3		A6, B6	−
DVDD3C		F14,F15,G14,H14	D+,D−

1.6 Internal regulator pins

REGOUT and REGIN pin are connect to capacitor which is for stability of internal regulator.

REGOUT and REGIN must be connected to RVSS through capacitor for supply power to internal regulator.

Please connect the capacitor by the shortest distance.

Is not power supply to external circuits from these terminals.

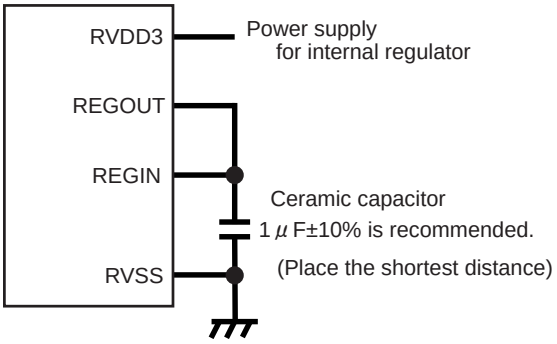


Figure 1-3 Internal regulator pins

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 TMPM365FYXBG.

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
TMPM365FYXBG	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 TMPM365FYXBG.

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.

TMPM365FYXBG has 57 interrupt inputs. The number of interrupt inputs is reflected in <INTLINESNUM [4:0]> bit of NVIC register. In this product, if read <INTLINESNUM[4:0]> bit, 0x01 is read out.

2.3.2 Number of Priority Level Interrupt Bits

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

TMPM365FYXBG 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.

TMPM365FYXBG 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.

TMPM365FYXBG 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, TMPM365FYXBG 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.

TMPM365FYXBG 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.

TMPM365FYXBG 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 TMPM365FYXBG does not use this function.

3. Debug Interface

3.1 Specification Overview

TMPM365FYXBG 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, 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: Ensure that to measure the power-consumption with debug tool connected in STOP1 mode is prohibitted.

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 TMPM365FYXBG 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 TMPM365FYXBG while JTAG is operating.

3. IDCODE

   Not available

4. Pins excluded from boundary scan register (BSR)
   a. Oscillator circuit pins (X1, X2)
   b. JTAG control pins (BSC)
   c. Power supply/GND pins (including reference supply pin for ADC)
   d. TEST pins (FTEST3)
   e. Function pins ($\overline{\text{RESET}}$)
   f. 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 TMPM365FYXBG 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.

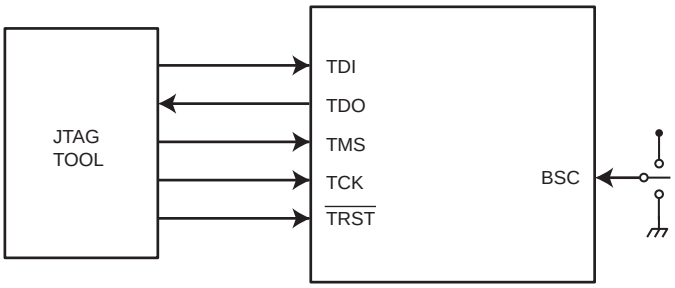


Figure 4-1 Example of connection with a JTAG development tool

Mode Setting Pin (BSC)	Operation mode
0	Set this pin to 0 except for Boundary Scan Mode. The TMPM365FYXBG operates as regular Debug Mode. Note: Debugging is not available if the internal BOOT is running.
1	The TMPM365FYXBG 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.

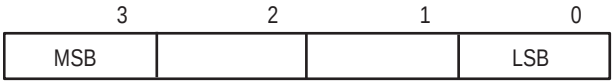


Figure 4-2 Instruction register

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

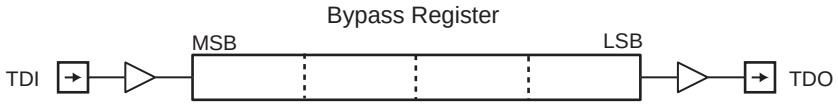


Figure 4-3 Instruction Register Shift Direction

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.

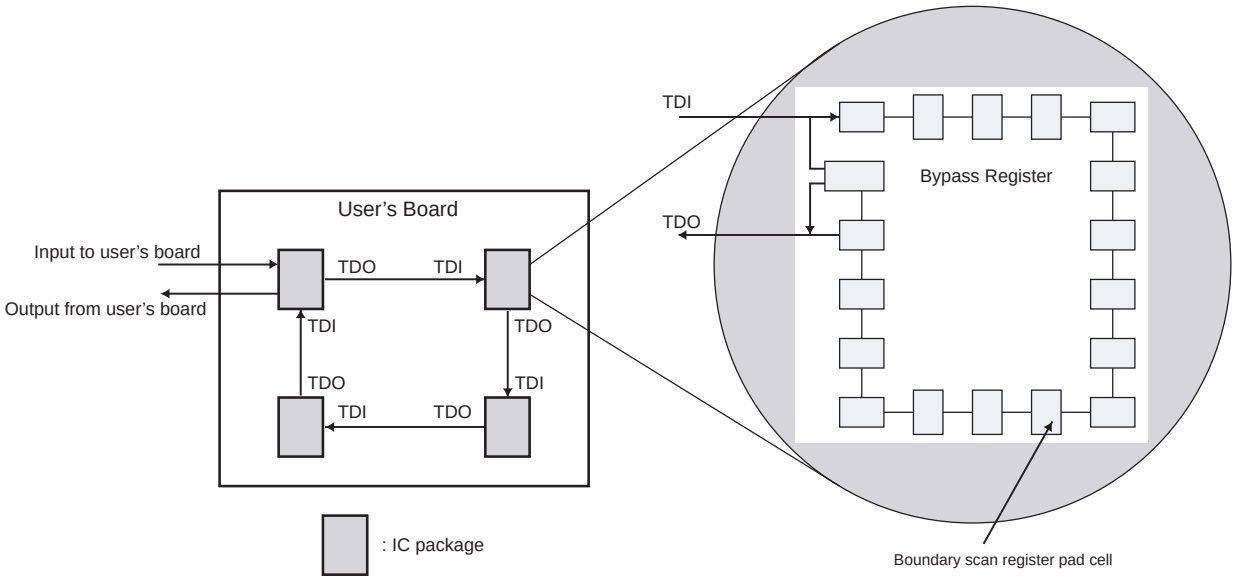


Figure 4-4 Bypass Register Operation

4.6 Boundary Scan Register

The boundary scan register provides all the inputs and outputs of the TMPM365FYXBG processor except some analog outputs and control signals. The pins of the TMPM365FYXBG 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 TMPM365FYXBG.

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.

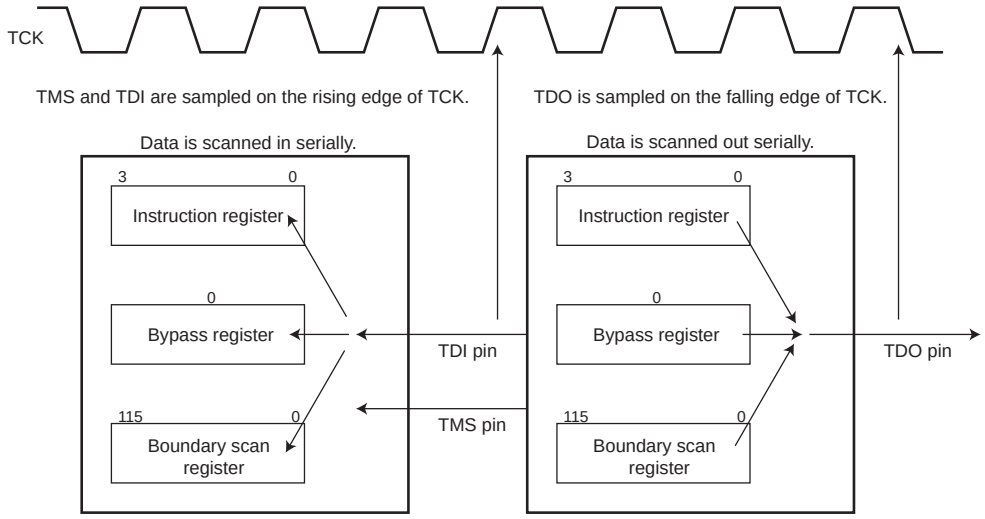


Figure 4-5 JTAG Test Access Port

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.

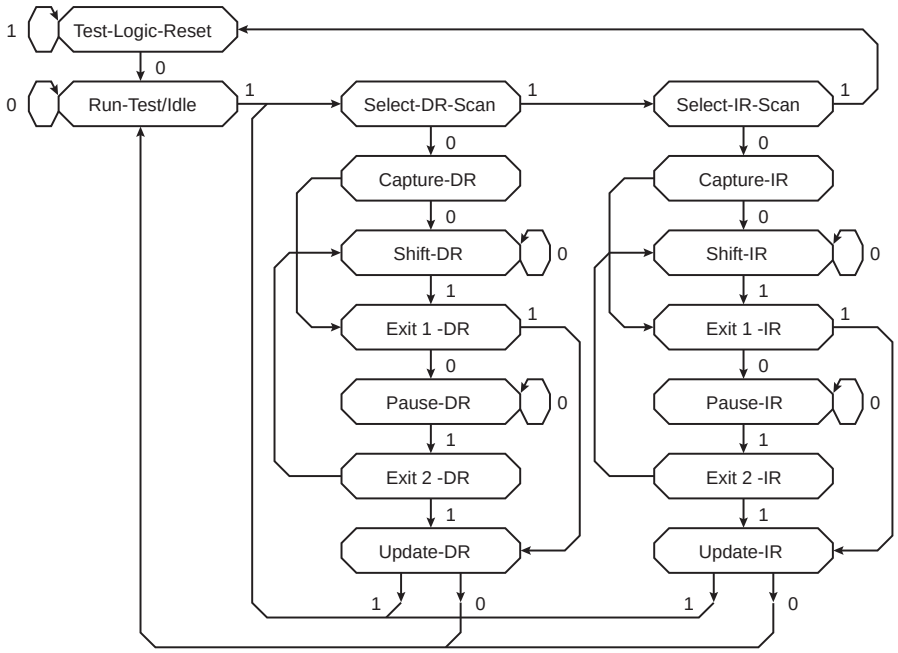


Figure 4-6 TAP Controller State Transition Diagram

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 TMPM365FYXBG 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 TMPM365FYXBG.

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.

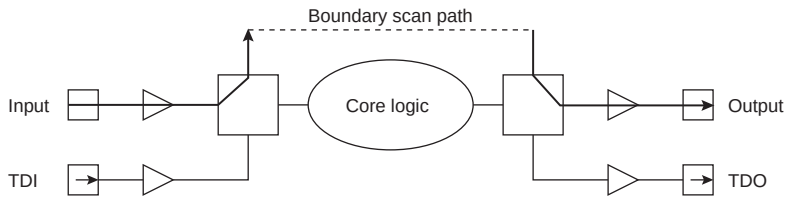


Figure 4-7 Test Data Flow when the EXTEST Instruction is Selected

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.

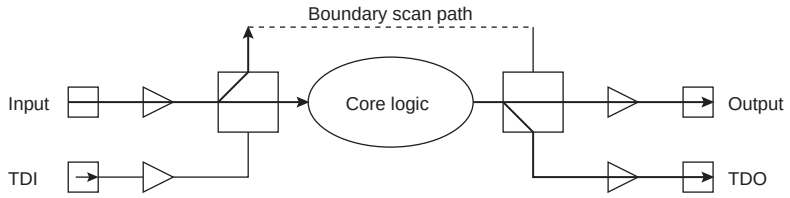


Figure 4-8 Test Data Flow while the SAMPLE is Selected

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.

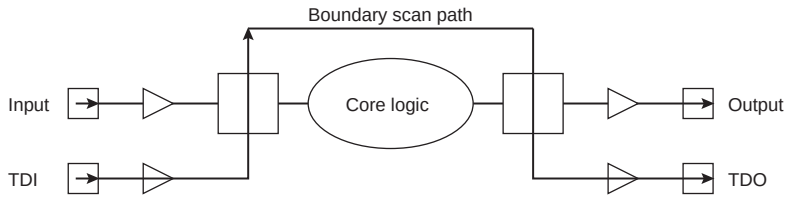


Figure 4-9 Test Data Flow while PRELOAD is Selected

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.

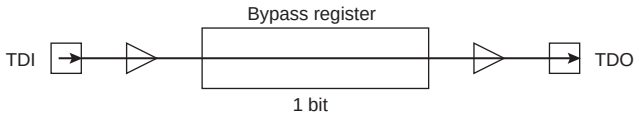


Figure 4-10 Test Data Flow when the BYPASS Instruction is Selected

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 the TMPM365FYXBG 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 TMPM365FYXBG

Figure 5-1shows the memory map of the TMPM365FYXBG.

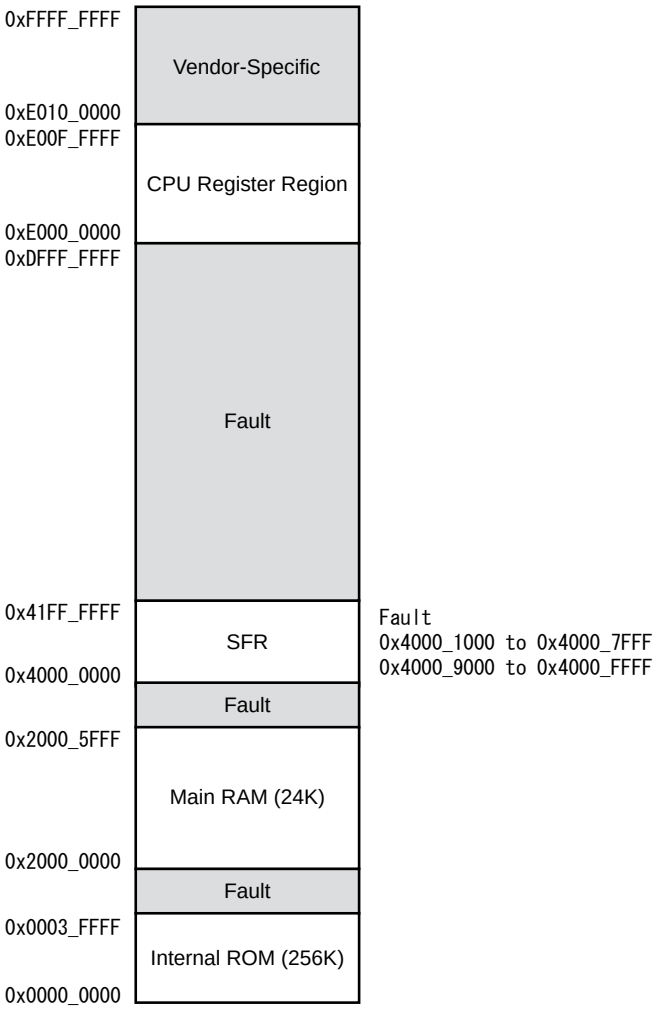


Figure 5-1 Memory Map

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(2ch)	0x4000_0028 - 0x4000_002C 0x4000_0034
0x4000_4000	0x4000_7FFF	Reserved	
0x4000_8000	0x4000_9FFF	USB(1ch)	
0x4000_A000	0x4003_FFFF	Reserved	
0x4004_0000	0x4004_7FFF	Reserved	
0x4004_8000	0x4004_BFFF	Reserved	
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	Reserved	
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	0x400E_0800 - 0x400E_0FFF
0x400E_1000	0x400E_1FFF	UART/SIO	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 TMPM365FYXBG 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

The internal circuits of the TMPM365FYXBG are undefined right after the power-on.

The register settings and pin status are undefined until the external reset pin ($\overline{\text{RESET}}$) receives low level after all the power supply voltage (DVDD3A, DVDD3C, RVDD3 and AVDD3) is applied.

6.2 Cold reset

The power-on sequence must include the time for the internal regulator, internal flash memory and internal oscillator to be stable.

In the TX03, the internal circuit automatically insert the time for the internal regulator, internal flash memory and internal oscillator. To keep this time, the reset pin must be hold "Low" level during 1ms or more after the voltage level of power supply is within the operation voltage level.

The internal reset signal is released after the level of reset pin ($\overline{\text{RESET}}$) is "High" and it takes aproximate 0.8ms.

Figure 6-1 shows the cold reset sequence after power-on.

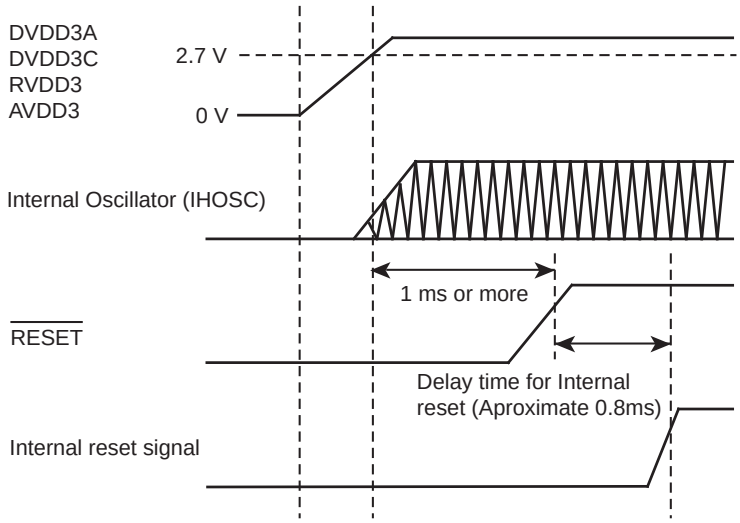


Figure 6-1 Cold Reset Sequence

Note: 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 TMPM365FYXBG, keeps the external reset pin ($\overline{\text{RESET}}$) for a minimum duration of 12 systemclocks.

The internal reset signal is released after the level of reset pin ($\overline{\text{RESET}}$) is "High" and it takes aproximate 0.8ms.

6.4 After reset

A 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 cold reset.

After reset, the PLL multiplication circuit is inactive and must be enabled 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.

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.

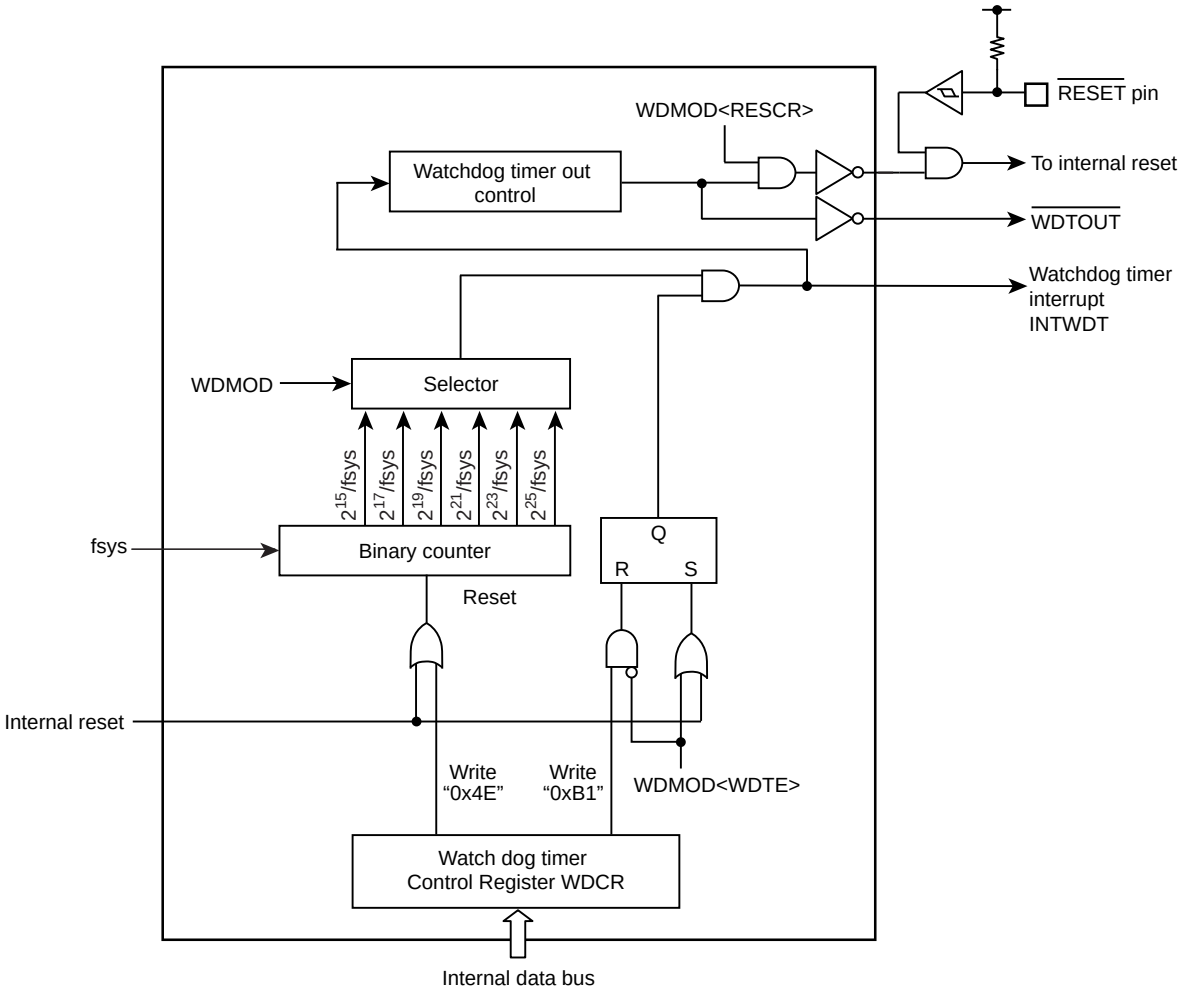


Figure 7-1 Block Diagram of the Watchdog Timer

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	43.70 ms	174.8 ms	699.1 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

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($\overline{\text{WDTOUT}}$).

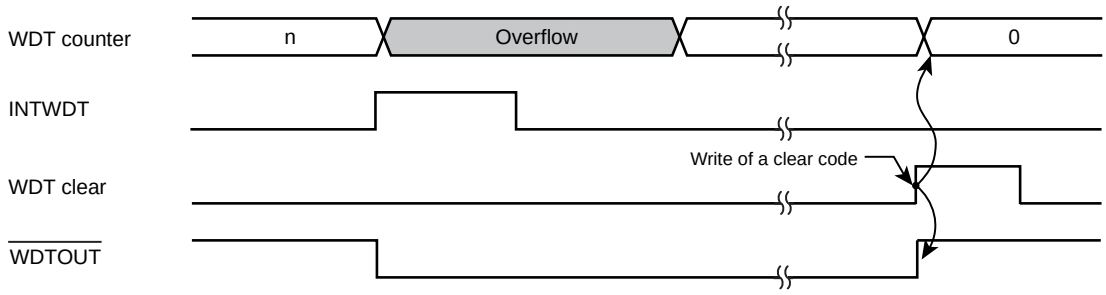


Figure 7-2 INTWDT interrupt generation

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.

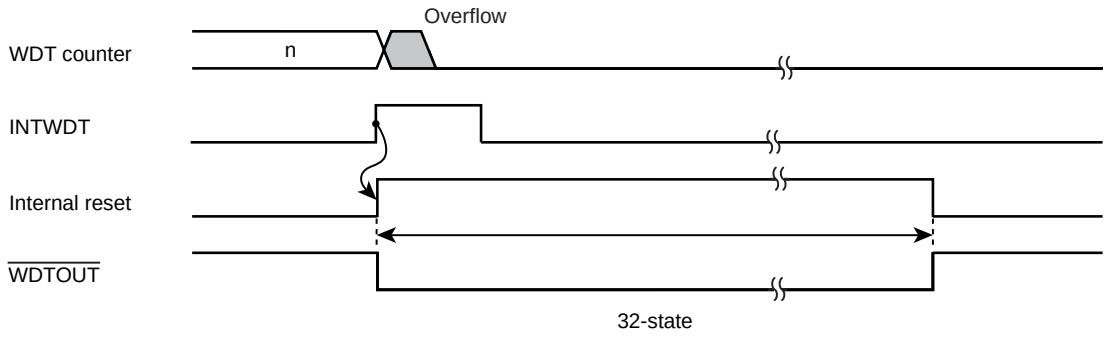


Figure 7-3 Internal reset generation

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 TMPM365FYXBG can operate in 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	Read as 0. 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

Note: Don't set the value reserved.

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 (f_{IHOSC}) 1: external (f_{eosc}) Selects warm-up counter by high-speed oscillator.
18	EHOSCSEL	R/W	External oscillator. 0: input external clock 1: external crystal oscillator
17	OSCSEL	R/W	High-speed oscillator 0: internal high-speed oscillator (EHOSC) 1: external high-speed oscillator (EHCLKIN) Selects warm-up counter by high-speed oscillator.
16	XEN2	R/W	Internal high-speed oscillator operation 0: Stop 1: Oscillation
15-12	−	R/W	Write "0" after reset.
11-10	−	R	Read as 0.
9	−	R/W	Write "0".
8	XEN1	R/W	External high-speed oscillator mode 0: Stop 1: Oscillation
7-3	−	R/W	Write "00110"
2	PLLON	R/W	PLL (multiplying circuit) operation (Note3) 0: Stop 1: Oscillation
1	WUEF	R	Staatus of warm-up timer (WUP) 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 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	-	-	-	-	-	-	-	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.
19-17	−	R/W	Write "0" after reset.
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: Reserved 110: Reserved 111: Reserved Don’t set the value reserved.

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
0	PLLSEL	R/W	Use of PLL 0: f_{osc} 1: f_{pLL} (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 when CGOSCCR<PLLON>=0 (PLL stop).

Note 2: Select PLL multiplying value which is shown in Table 8-2.

Note 3: Returning from the STOP1 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	0	0	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 0 .

Note 1: When <USBCLKSEL> is modified, <USBCLKEN> must be clear to "0".
Note 2: <USBCLKSEL> and <USBCLKEN> are modified 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, 0x400F_3000 to 0x400F_303B: write configuration to each register 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 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.

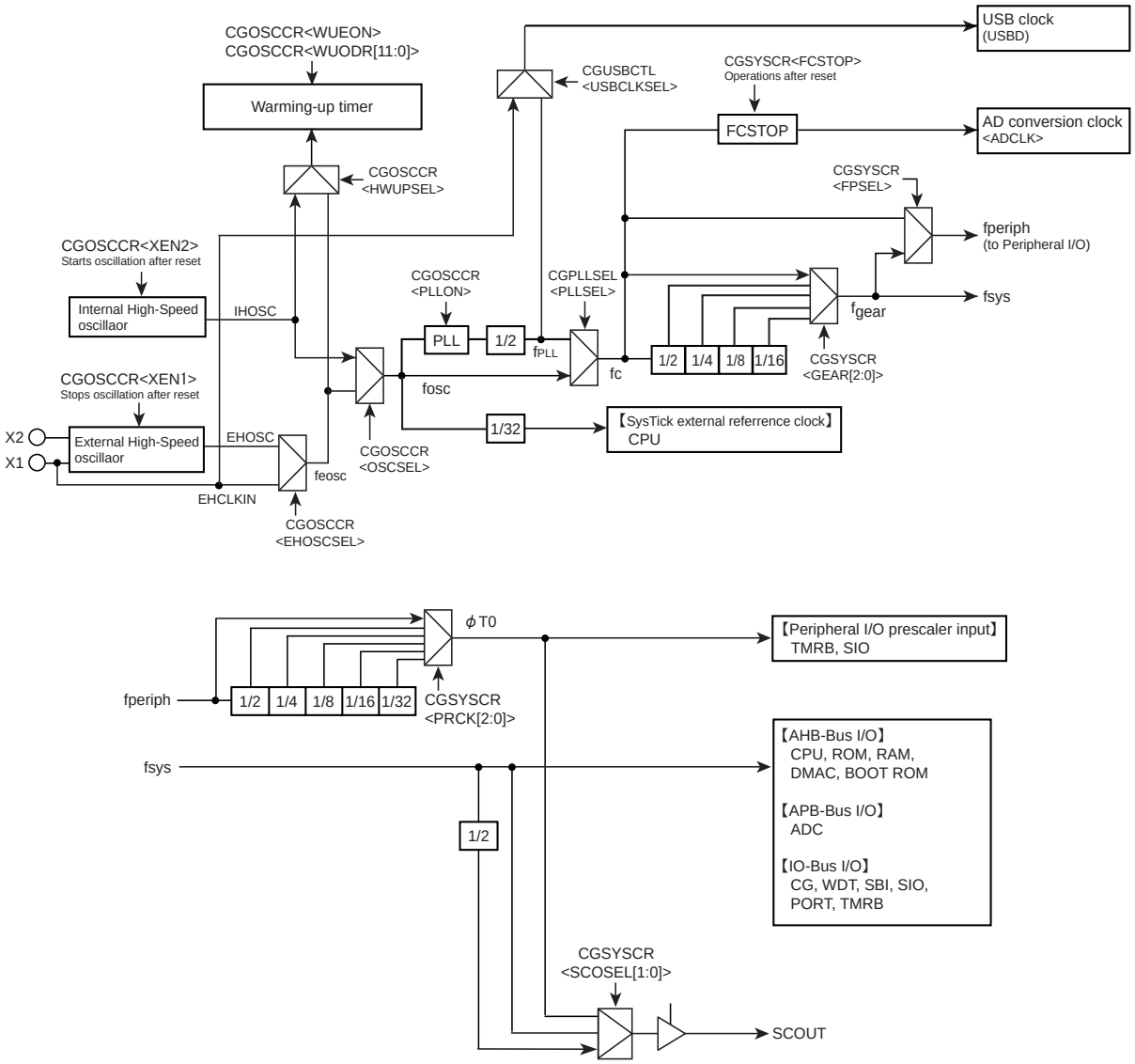


Figure 8-1 Clock Block Diagram

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.

The detail of warm-up function is described in "8.6.6 Warm-up".

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

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
CGOSCCR<XEN2> = "1"	: Internal high-speed oscillator (IHOSC) 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: Setting warm-up count value to CGOSCCR<WUODR[11:0]>, wait until this value is reflected, then transit to stand-by mode by executing a command "WFI"

Note 4: When returning from STOP1 mode, related bits CGPLLSEL<PLLSEL>, CGOSCCR<HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1> and <PLLON> are initialized because of internal high-speed oscillator starting, and CGOSCCR<WUODR[11:0]> is not initialized.

8.3.5 Clock Multiplication Circuit (PLL)

This circuit outputs the f_{PLL} clock that is multiplied by 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 Start operation

The PLL is disabled after reset.

In order to use PLL, set a multiplication value to CGPLLSEL<PLLSET> while CGOSCCR<PLLON> is "0". Then wait until approximately 100μs has elapsed as a PLL initial stabilization time, and set "1" to <PLLON> to start PLL operation. After that, in order to use f_{PLL} clock, wait until approximately 100μs has elapsed as a lock up time, then after set "1" to CGPLLSEL<PLLSEL>. This allows fosc to be multiplied by 8-fold. Note that a warm-up time is required until PLL operation becomes stable using warm-up function.

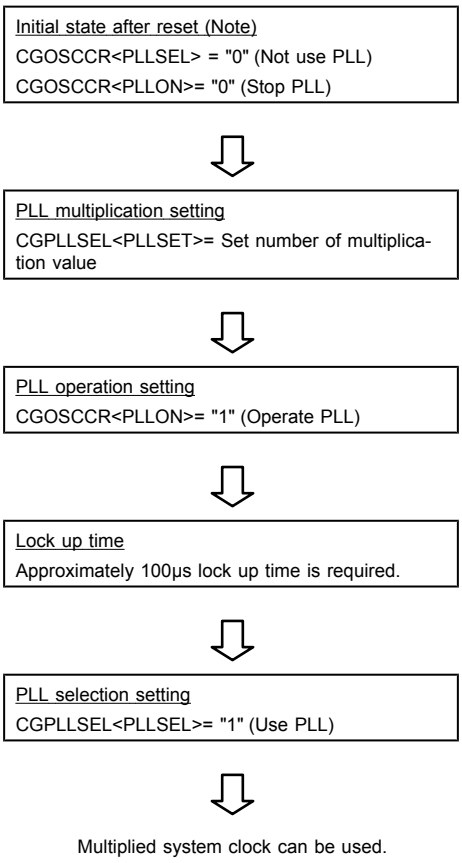
Note: Do not use PLL while internal oscillator (IHOSC) is used.

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

Multiplying	<PLLSEL>
8	0x381E

The sequence of starting PLL is shown belows.

(1) PLL Operation Start Procedure



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 12MHz	disused or 8 multiplying
	Input clock (EHCLKIN)	8 to 12,48MHz	

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

↑ Initial value after reset

Note 1: Maximum Operating Frequency of ADC is 40MHz, which is 2 dividing fc/2 frequency specified by ADxCLK<ADCLK.> register.

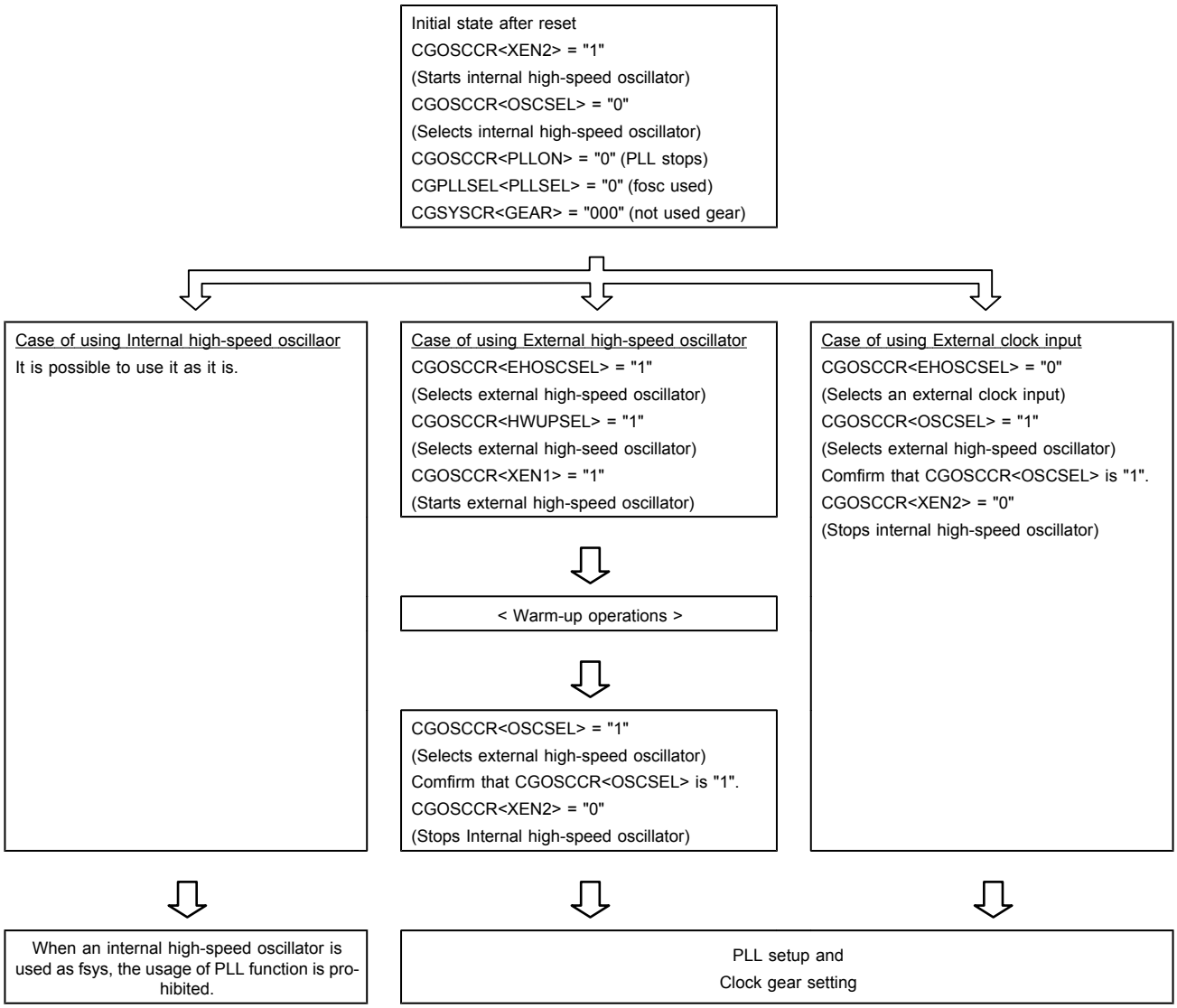
Note 2: Don't use 1/16 when SysTick is used.

8.3.6.1 System Clock setting

The system clock can be selected by setting the 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 1: 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.

Only when TBxCR<FT0SEL> is "1", φTn can be less or equal than fsys.

Note 2: Don't change prescaler for dividing until the peripheral function such as TMRB is operated.

8.3.8 System Clock Pin Output Function

The TX03 enables to output the system clock from a pin. The PD7/SCOUT pin can output the system clock fsys and fsys/2, and the prescaler input clock for peripheral I/O φT0.

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

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

Regarding to setting when port is used as SCOUT, refer to "Input/Output port".

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
<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".	

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.

Figure 8-2 shows a mode transition diagram.

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

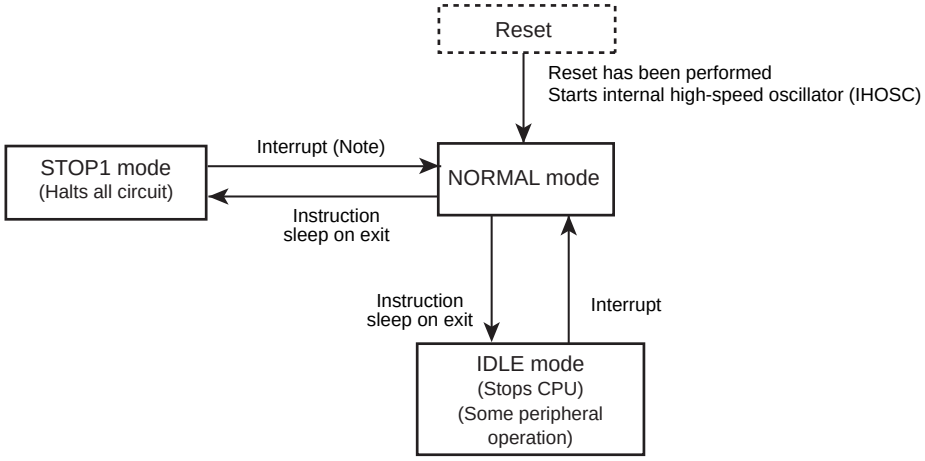


Figure 8-2 Mode Transition Diagram

Note: The warm-up is needed. The warm-up time must be set in NORMAL mode before changing to STOP1 mode. Regarding warm-up time, refer to "8.6.6 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 two low power consumption modes: IDLE and STOP1. 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 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: Please stop the clocks to the USB before the transition to the IDLE mode. (CGUSBCTL<USBCLKEN>="0")

8.6.2 STOP1 mode

When STOP1 mode is released, an internal oscilator starts and the operation mode is changed to NORMAL mode.

The STOP1 mode enables to select the pin status by setting the CGSTBYCR<DRVE>. Table 8-4 shows the pin status in the STOP1 mode.

Note 1: It is necessary to warm-up when the operation mode is changed from STOP1 mode to NORMAL mode. Warm-up is required when MCU returning. Warm-up time must be set in the previous mode (NORMAL mode) entering before STOP1 mode. For details of warm-up time, refer to "8.3.4 Warm-up function".

Note 2: When returning from STOP1 mode, related bits CGPLLSEL<PLLSEL>, CGOSCCR<HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1> and <PLLON> are initialized because of internal high-speed oscillator starting, and CGOSCCR<WUODR[11:0]> is not initialized.

Table 8-4 Pin States in the STOP1 mode

Function	Pin Name	I/O	STOP1	
			<DRVE> = "0"	<DRVE> = "1"
Control Pin	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC	Input	o	o
Oscillator	X1/EHCLKIN	Input	×	×
	X2	Output	"High" level output.	
PORT	PI3, PI6, PI7 (TCK/SWCLK, TDI, $\overline{\text{TRST}}$) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Input	Depends on PxIE[m].	
	PI4 (TMS/SWDIO) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Input	Depends on PxIE[m].	
		Output	Enabled when data is valid. Disabled when data is invalid.	
	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].	
	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
	If using other than listed above	Input	×	Depends on PxIE[m].
		Output	×	Depends on PxCR[m].

o : Valid input or output.

× : Invalid input or output.

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

8.6.3 Low power Consumption Mode Setting

The low power consumption mode is specified by the setting of 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

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

8.6.4 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)
Processor core	o	o	−	−	−
DMAC	o	o	o	o	−
I/O port	o	o	o	o	o(Note 2)
SIO/UART	o	o	Δ	Δ	−
I2C/SIO	o	o	Δ	Δ	−
TMRB	o	o	Δ	Δ	−
WDT	o	o	Δ(Note 4)	Δ(Note 4)	−
USB	o	o	o	o	−
12-bit ADC	o	o	Δ	Δ	−
CG	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 3)	o	o(Note 3)	−
Main RAM	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.

Note 1: Before transit to STOP1 mode, stop peripheral functions of "−"and "×". It is available to reduce leakage current by stopping reference voltage for AD converter.

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

Note 3: After reset or STOP1 mode released, clock is provided from internal high-speed oscillator.

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

8.6.5 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
Release source	Interrupt	INT0 to 9 (Note1)	o	o
		INTTB0 to 9	o	×
		INTCAP00 to 91	o	×
		INTRX0 to 1, INTTX0 to 1	o	×
		INTSBI0 to 1	o	×
		INTUSB	o	o
		INTUSBWKUP	o	o
		INTAD / INTADHP / INTADM0 to 1	o	×
		INTDMAC0TC, INTDMAC0ERR	o	×
	SysTick interrupt		o	×
	Non-Maskable Interrupt (INTWDT)		o	×
	Non-Maskable Interrupt ($\overline{\text{NMI}}$ pin)		o	o
	RESET ($\overline{\text{RESET}}$ pin)		o	o

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

Note 1: For shifting to the low power consumption mode, set the CPU to prohibit all the interrupts other than the release source. If not, releasing may be executed by an unspecified interrupt.

Note 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.

- 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 modes.

- Release by Non-Maskable Interrupt (NMI)

  INTWDT can only be used in the IDLE mode.

- Release by reset

  Any low power consumption mode can be released by reset from the $\overline{\text{RESET}}$ pin. 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 mode by reset does not induce the automatic warm-up. Keep the reset signal valid until the oscillator operation becomes stable.

- Release by SysTick

  SysTick interrupt is used in the IDLE mode.

Refer to "Interrupts" for details.

8.6.6 Warm-up

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

In the mode transition from STOP1 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 mode.

Note: When returning from STOP1 mode, related bits CGPLLSEL<PLLSEL>, CGOSCCR<HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1> and <PLLON> are initialized because of internal high-speed oscillator starting, and CGOSCCR<WUODR[11:0]> is not initialized.

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
IDLE → NORMAL	Not required
STOP1 → NORMAL	Auto-warm-up (Note)

Note: Returning to NORMAL mode by reset does not induce the automatic warm-up. Input the reset signal as same as cold reset.

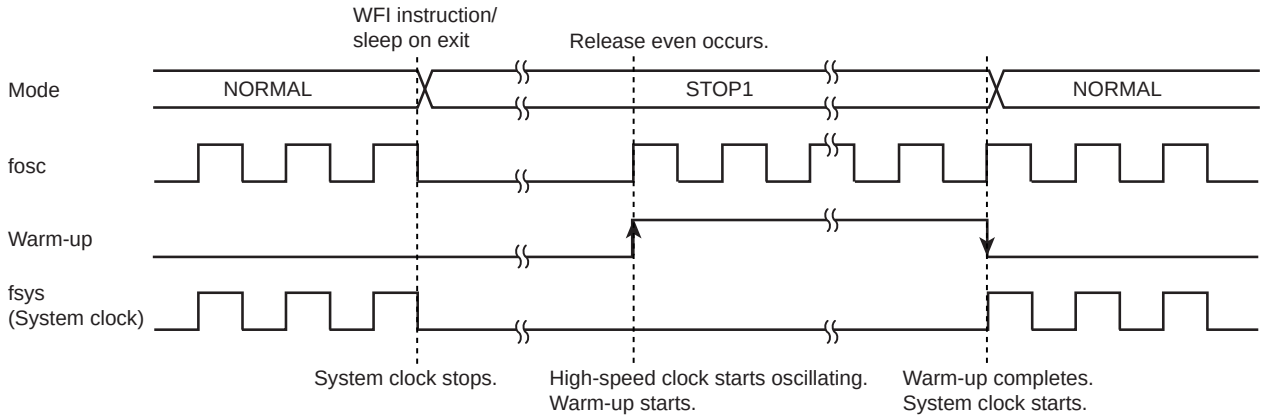
8.6.7 Clock Operations in Mode Transition

The clock operations in mode transition are described as follows.

8.6.7.1 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 to set the warm-up time 0x119 into CGOSCCR<WUODR[11:0]>, which is the stable time for internal flash, before entering the STOP1 mode.

Returning to the NORMAL mode by reset does not induce the automatic warm-up. Input the reset signal as same as cold reset.



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.

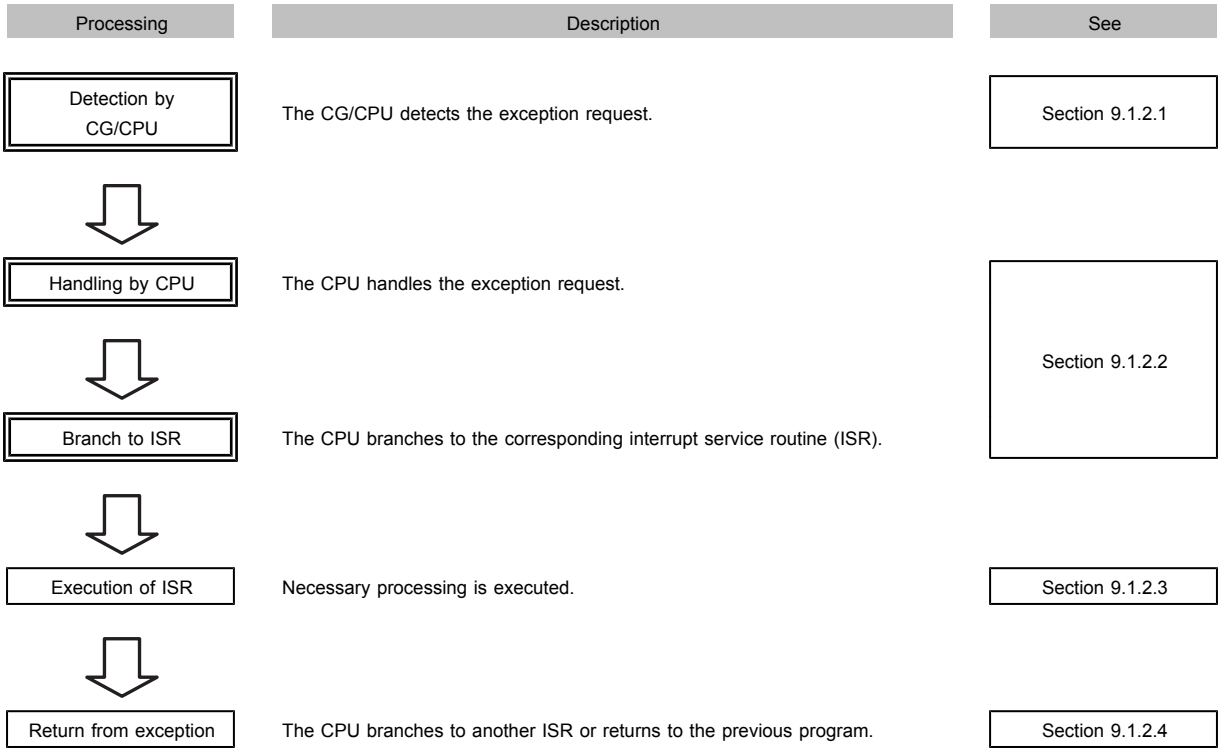
For detailed descriptions on each exception, refer to "Cortex-M3 Technical Reference Manual".

- 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.



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)

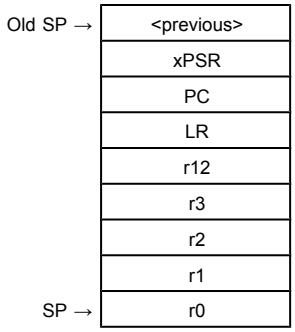
When an exception occurs, the CPU suspends the currently executing process and branches to the interrupt service routine. This is called "pre-emption".

(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).

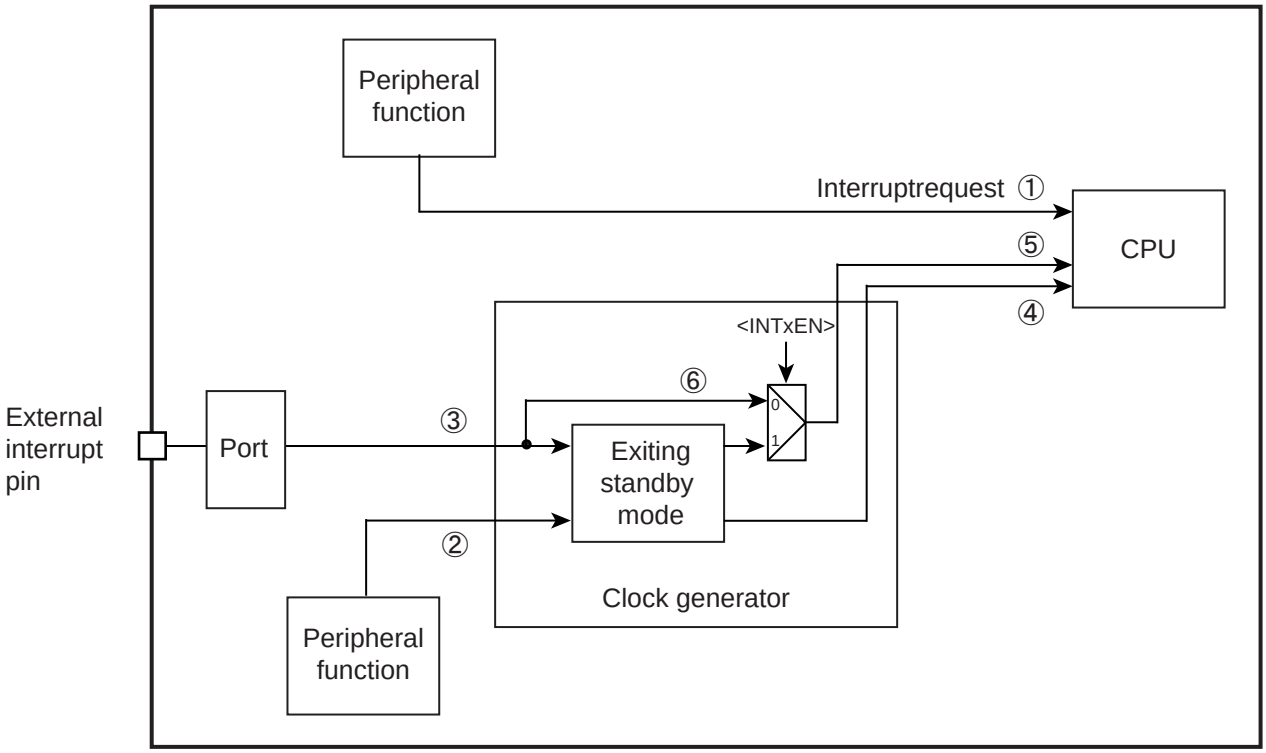


Figure 9-1 Interrupt Route

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 (Release 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 reception (channel 0)		
9	INTTX0	Serial transmission (channel 0)		
10	INTRX1	Serial reception (channel 1)		
11	INTTX1	Serial transmission (channel 1)		
12	INTUSBWKUP	USB Wake-up interrupt	Falling	CGIMCGC
13	-	Reserved		
14	INTSBI0	Serial bus interface 0		
15	INTSBI1	Serial bus interface 1		
16	INTADHP	Highest priority AD conversion complete interrupt		
17	INTAD	AD conversion completion interrupt		
18	INTADM0	AD conversion monitoring function interrupt 0		
19	INTADM1	AD conversion monitoring function interrupt 1		
20	INTTB0	TMRB0 match detection		
21	INTTB1	TMRB1 match detection		
22	INTTB2	TMRB2 match detection		
23	INTTB3	TMRB3 match detection		
24	INTTB4	TMRB4 match detection		
25	INTTB5	TMRB5 match detection		
26	INTTB6	TMRB6 match detection		
27	INTTB7	TMRB7 match detection		
28	INTTB8	TMRB8 match detection		
29	INTTB9	TMRB9 match detection		
30	INTUSB	USB interrupt		
31	-	Reserved		
32	-	Reserved		
33	-	Reserved		
34	INTUSBPON	USB Power On connection detection interrupt (Vbus Detect)	Selectable	CGIMCGC
35	-	Reserved		
36	INTCAP00	TMRB0 input capture 0		
37	INTCAP01	TMRB0 input capture 1		
38	INTCAP10	TMRB1 input capture 0		
39	INTCAP11	TMRB1 input capture 1		
40	INTCAP20	TMRB2 input capture 0		
41	INTCAP21	TMRB2 input capture 1		

Table 9-3 List of Interrupt Sources

No.	Interrupt Source		active level (Release standby and interrupt)	CG interrupt mode control register
42	INTCAP30	TMRB3 input capture 0		
43	INTCAP31	TMRB3 input capture 1		
44	INTCAP40	TMRB4 input capture 0		
45	INTCAP41	TMRB4 input capture 1		
46	INTCAP50	TMRB5 input capture 0		
47	INTCAP51	TMRB5 input capture 1		
48	INTCAP60	TMRB6 input capture 0		
49	INTCAP61	TMRB6 input capture 1		
50	INTCAP70	TMRB7 input capture 0		
51	INTCAP71	TMRB7 input capture 1		
52	INTCAP80	TMRB8 input capture 0		
53	INTCAP81	TMRB8 input capture 1		
54	INTCAP90	TMRB9 input capture 0		
55	INTCAP91	TMRB9 input capture 1		
56	INT8	Interrupt pin 8	Selectable	CGIMCGC
57	INT9	Interrupt pin 9		
58	-	Reserved		
59	-	Reserved		
60	INTDMAC0TC	DMA terminal count status interrupt 0		
61	INTABTLOSS0	I2C arbitration lost interrupt 0		
62	INTDMAC0ERR	DMA error status interrupt 0		
63	INTABTLOSS1	I2C arbitration lost 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.

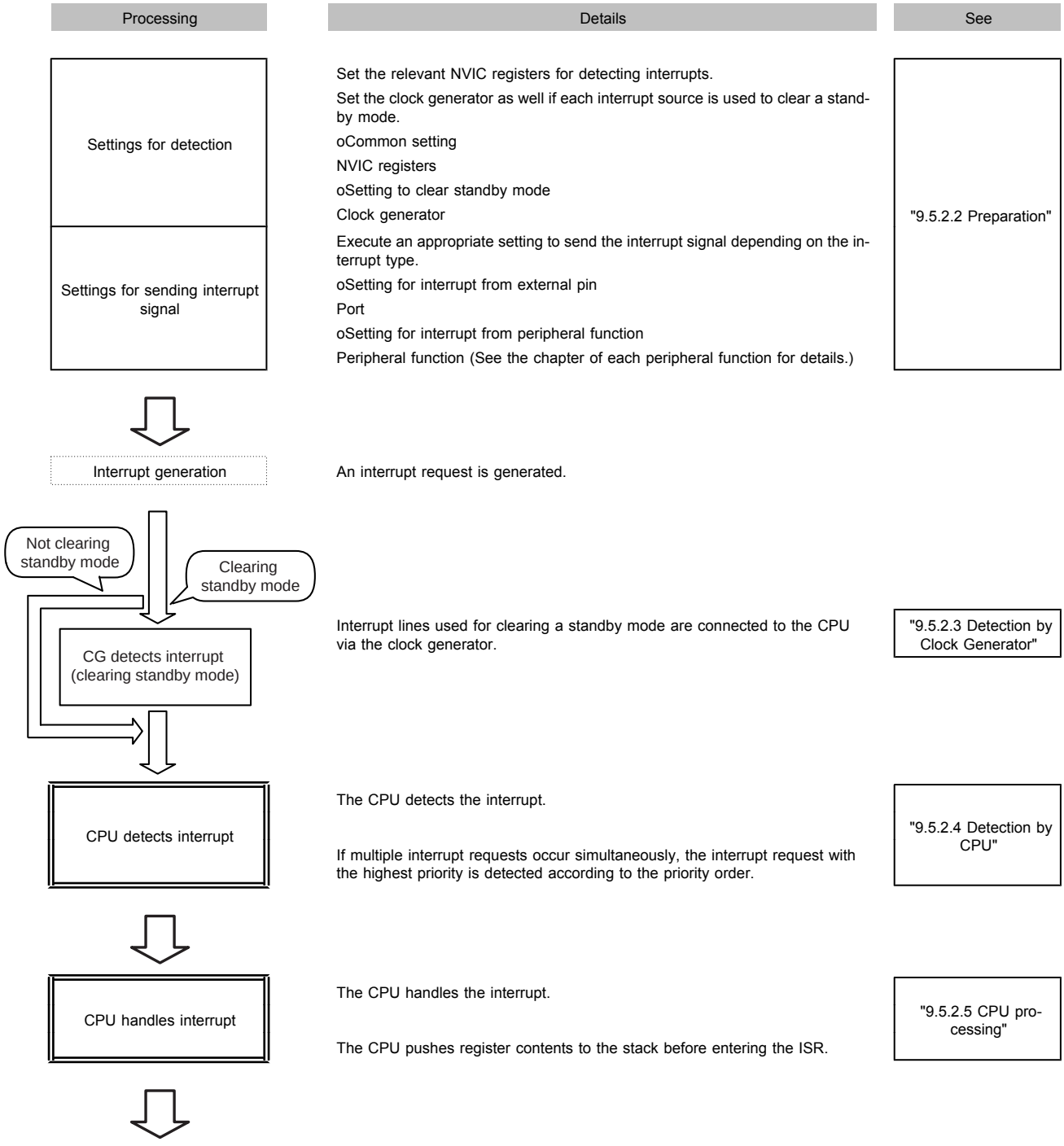
An interrupt request detected by the clock generator is notified to the CPU with a signal in "High" level.

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
Reserved	0x0108 to 0x017F
Interrupt Clear-Enable Register 1	0x0180
Interrupt Clear-Enable Register 2	0x0184
Reserved	0x0188 to 0x01FF
Interrupt Set-Pending Register 1	0x0200
Interrupt Set-Pending Register 2	0x0204
Reserved	0x0208 to 0x027F
Interrupt Clear-Pending Register 1	0x0280
Interrupt Clear-Pending Register 2	0x0284
Reserved	0x0288 to 0x03FF
Interrupt Priority Register	0x0400 to 0x0460
Reserved	0x0464 to 0x0D07
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
Reserved	-	0x004C to 0x005F
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) 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	0	0	1
	7	6	5	4	3	2	1	0
bit symbol	TENMS							
After reset	1	1	0	0	0	1	0	0

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

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

9.6.2.5 Interrupt Set-Enable Register 1

	31	30	29	28	27	26	25	24
bit symbol	-	SETENA (Interrupt 30)	SETENA (Interrupt 29)	SETENA (Interrupt 28)	SETENA (Interrupt 27)	SETENA (Interrupt 26)	SETENA (Interrupt 25)	SETENA (Interrupt 24)
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	SETENA (Interrupt 23	SETENA (Interrupt 22)	SETENA (Interrupt 21)	SETENA (Interrupt 20)	SETENA (Interrupt 19)	SETENA (Interrupt 18)	SETENA (Interrupt 17)	SETENA (Interrupt 16)
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	SETENA (Interrupt 15)	SETENA (Interrupt 14)	-	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	−	R	Read as 0.
30-14	SETENA	R/W	Interrupt number [30:14] [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.
13	−	R	Read as 0.
12-0	SETENA	R/W	Interrupt number [12:0] [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 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 34)	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-28	SETENA	R/W	Interrupt number [63:60] [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.
27-26	−	R	Read as 0.
25-4	SETENA	R/W	Interrupt number [57:36] [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.
3	−	R	Read as 0.
2	SETENA	R/W	Interrupt number [34] [Write] 1: Enable [Read] 0: Disabled 1: Enabled Each bit corresponds to the specified number of interrupts. Writing "1" to a bit in this register enables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
1-0	−	R	Read as 0.

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 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 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	−	R	Read as 0.
30-14	CLRENA	R/W	Interrupt number [30:14] [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.
13	−	R	Read as 0.
12-0	CLRENA	R/W	Interrupt number [12:0] [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 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 34)	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-28	CLRENA	R/W	Interrupt number [63:60] [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.
27-26	–	R	Read as 0.
25-4	CLRENA	R/W	Interrupt number [57:36] [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.
3	–	R	Read as 0.
2	CLRENA	R/W	Interrupt number [34] [Write] 1: Disabled [Read] 0: Disabled 1: Enable Each bit corresponds to the specified number of interrupts. It can be performed to enable interrupts and to check if interrupts are disabled. Writing "1" to a bit in this register disables the corresponding interrupt. Writing "0" has no effect. Reading the bits can see the enable/disable condition of the corresponding interrupts.
1-0	–	R	Read as 0.

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 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 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	−	R	Read as 0.
30-14	SETPEND	R/W	Interrupt number [30:14] [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.
13	−	R	Read as 0.
12-0	SETPEND	R/W	Interrupt number [12:0] [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 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 34)	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-28	SETPEND	R/W	Interrupt number [63:60] [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.
27-26	−	R	Read as 0.
25-4	SETPEND	R/W	Interrupt number [57:36] [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.
3	−	R	Read as 0.
2	SETPEND	R/W	Interrupt number [34] [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.
1-0	SETPEND	R/W	Read as 0.

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 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 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	−	R	Read as 0.
30-14	CLRPEND	R/W	Interrupt number [30:14] [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.
13	−	R	Read as 0.
12-0	CLRPEND	R/W	Interrupt number [12:0] [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 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 34)	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-28	CLRPEND	R/W	Interrupt number [63:60] [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.
27-26	−	R	Read as 0.
25-4	CLRPEND	R/W	Interrupt number [57:36] [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.
3	−	R	Read as 0.
2	CLRPEND	R/W	Interrupt number [34] [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending Each bit corresponds to the specified number can force interrupts into the pending state and determines which interrupts are currently pending. Writing "1" to a bit in this register clears the corresponding pending interrupt. However, writing "1" has no effect on an interrupt that is already being serviced. Writing "0" has no effect. Reading the bit returns the current state of the corresponding interrupts.
1-0	−	R	Read as 0.

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_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_30	PRI_29	PRI_28
0xE000_E420	-	PRI_34	-	-
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_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)/ VECTKEY-STAT(Read)	R/W	Register key [Write] Writing to this register requires 0x5FA in the <VECTKEY> field. [Read] Read as 0xFA05.
15	ENDIANESS	R/W	Endianness bit:(Note1) 1: big endian 0: little endian
14-11	−	R	Read as 0.
10-8	PRIGROUP	R/W	Interrupt priority grouping 000: seven bits of pre-emption priority, one bit of 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. (101~111: 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. (101~111: 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. (101~111: 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. (101~111: 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. (101~111: 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. (101~111: 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. (101~111: 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. (101~111: 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. (101~111: 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. (101~111: 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. (101~111: 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 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.

9.6.3.4 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	-	WDTRSTF	-	PINRSTF
After pin reset	0	0	0	0	Undefined	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	−	R/W	Read as undefined. Write as 0.
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 2: **This product is initialized by the external reset.**

10. Input/Output Ports

10.1 Port Functions

10.1.1 Function Lists

TMPM365FYXBG 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 C)

Port	pin	Input / Output	Pull-up Pull-down	Schmitt Input	Noise Filter	Program-mable Open-drain	Function pin
PORTA							
	PA0	I/O	Pull-up	-	-	o	-
	PA1	I/O	Pull-up	-	-	o	-
	PA2	I/O	Pull-up	-	-	o	-
	PA3	I/O	Pull-up	-	-	o	-
	PA4	I/O	Pull-up	-	-	o	-
	PA5	I/O	Pull-up	-	-	o	-
	PA6	I/O	Pull-up	-	-	o	-
	PA7	I/O	Pull-up	-	-	o	-
PORTB							
	PB0	I/O	Pull-up	-	-	o	-
	PB1	I/O	Pull-up	-	-	o	-
	PB2	I/O	Pull-up	-	-	o	-
	PB3	I/O	Pull-up	-	-	o	-
	PB4	I/O	Pull-up	-	-	o	-
	PB5	I/O	Pull-up	-	-	o	-
	PB6	I/O	Pull-up	-	-	o	-
	PB7	I/O	Pull-up	-	-	o	-
PORTC							
	PC0	I/O	Pull-up	o	-	o	TXD1/ TB2IN0
	PC1	I/O	Pull-up	o	-	o	RXD1/ TB2IN1
	PC2	I/O	Pull-up	o	-	o	SCLK1/ TB0OUT/ $\overline{CTS1}$

o : Exist

- : Not Exist

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

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

Port	PIn	Input /Output	Pull-up Pull-down	Schmitt Input	Noise Filter	Programmable Open-drain	Function pin
PORTD							
	PD0	I/O	Pull-up	o	-	o	TB7OUT
	PD1	I/O	Pull-up	o	-	o	TB8OUT
	PD2	I/O	Pull-up	o	-	o	TB9OUT
	PD3	I/O	Pull-up	o	-	o	$\overline{\text{ADTRG}}$
	PD4	I/O	Pull-up	-	-	o	-
	PD5	I/O	Pull-up	-	-	o	-
	PD6	I/O	Pull-up	-	-	o	-
	PD7	I/O	Pull-up	o	-	o	SCOUT
PORTE							
	PE0	I/O	Pull-up	o	-	o	TXD0
	PE1	I/O	Pull-up	o	-	o	RXD0
	PE2	I/O	Pull-up	o	-	o	SCLK0/ TB2OUT/ $\overline{\text{CTS0}}$
	PE3	I/O	Pull-up	o	o(INT5 only)	o	INT5/ TB3OUT
	PE4	I/O	Pull-up	o	-	o	SDA1/ SO1
	PE5	I/O	Pull-up	o	-	o	SCL1/ SI1
	PE6	I/O	Pull-up	o	-	o	SCK1
	PE7	I/O	Pull-up	o	o	o	INT4
PORTF							
	PF0	Output	Pull-up	o	-	o	$\overline{\text{BOOT}}$/ TB6OUT
	PF1	I/O	Pull-up	o	-	o	-
	PF2	I/O	Pull-up	o	-	o	-
	PF3	I/O	Pull-up	o	-	o	-
	PF4	I/O	Pull-up	o	o(INT6 only)	o	INT6/ TB5IN0
	PF5	I/O	Pull-up	o	o(INT7 only)	o	INT7/ TB5IN1
	PF6	I/O	Pull-up	o	-	o	-
	PF7	I/O	Pull-up	o	-	o	-
PORTG							
	PG0	I/O	Pull-up	o	-	o	SDA0/ SO0
	PG1	I/O	Pull-up	o	-	o	SCL0/ SI0/ TB3IN0
	PG2	I/O	Pull-up	o	-	o	SCK0/ TB3IN1
	PG3	I/O	Pull-up	o	o(INT0 only)	o	INT0/ TB4IN0
	PG4	I/O	Pull-up	o	-	o	TB4IN1
	PG5	I/O	Pull-up	o	o(INT1 only)	o	INT1/ USBPON

o : Exist
- : Not Exist

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

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

Port	PIn	Input /Out-put	Pull-up Pull-down	Schmitt Input	Noise Filter	Program-mable Open-drain	Function pin
PORTH							
	PH0	I/O	Pull-up	o	-	o	TRACEDATA2
	PH1	I/O	Pull-up	o	-	o	TRACEDATA3
	PH2	I/O	Pull-up	o	-	o	TB4OUT
	PH3	I/O	Pull-up	o	-	o	TB5OUT
	PH4	I/O	Pull-up	o	o	o	INT8
PORTI							
	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	After Reset, Pull-up	o	o	-	$\overline{\text{TRST}}$
PORTJ							
	PJ0	I/O	Pull-up	o	-	-	AIN00
	PJ1	I/O	Pull-up	o	-	-	AIN01
	PJ2	I/O	Pull-up	o	-	-	AIN02
	PJ3	I/O	Pull-up	o	-	-	AIN03
	PJ4	I/O	Pull-up	o	-	-	AIN04
	PJ5	I/O	Pull-up	o	-	-	AIN05
	PJ6	I/O	Pull-up	o	-	-	AIN06/ TB0IN0
	PJ7	I/O	Pull-up	o	o(INT9 only)	-	AIN07/ INT9/ TB0IN1
PORTK							
	PK0	I/O	Pull-up	o	o(INT2 only)	-	AIN08/ INT2/ TB1IN0
	PK1	I/O	Pull-up	o	o(INT3 only)	-	AIN09/ INT3/ TB1IN1
	PK2	I/O	Pull-up	o	-	-	AIN10/ TB6IN0
	PK3	I/O	Pull-up	o	-	-	AIN11/ TB6IN1

o : Exist

- : Not Exist

Note: The noise elimination width of the noise filter is approximately 30 ns 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 Mode

Input and output in STOP1 mode are enabled/disabled by the CGSTBYCR<DRVE> bit.

If PxIE or PxCR is enabled with <DRVE>="1", input or output is enabled respectively in STOP1 mode. If <DRVE>="0", both input and output are disabled in STOP1 mode except for some ports even if PxIE or PxCR are enabled.

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

Table 10-4 Port conditions in STOP 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 (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 : Input or output enabled.

× : Input or output disabled.

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

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

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
Reserved	-	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	0

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 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 1: Open drain

10.2.1.5 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

10.2.1.6 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.

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

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
Reserved	-	0x0008
Reserved	-	0x000C
Reserved	-	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 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 1: Open drain

10.2.2.5 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

10.2.2.6 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 general purpose serial interface (UART/SIO) and the 16-bits timer (TMRB).

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

The port C has three 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 other 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
Reserved	-	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 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

10.2.3.6 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: $\overline{\text{CTS1}}$
1-0	−	R	Read as 0.

10.2.3.7 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 1: Open drain

10.2.3.8 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

10.2.3.9 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 16-bits timer (TMRB), the clock output and the ADC trigger input.

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

The port D has a type 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.

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
Reserved	-	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 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

10.2.4.5 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 1: Open drain

10.2.4.6 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

10.2.4.7 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 general purpose serial interface (SIO/UART), the external interrupt input, the 16-bits timer (TMRB) and the serial bus interface (I2C/SIO).

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

The port E has three types of 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 other 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
Reserved	-	0x000C
Port E function register 3	PEFR3	0x0010
Port E function register 4	PEFR4	0x0014
Reserved	-	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 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.

10.2.5.6 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.

10.2.5.7 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 1: Open-drain

10.2.5.8 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

10.2.5.9 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 interrupt input, 16-bit timer (TMRB) and the mode setting function.

The port F has two 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 other function registers at the same time.

Reset initializes all bits of the port F as general-purpose ports. The bit of PF0 output is disabled and the pull-up setting is enabled, the bits of PF<7:1> input/output and all bits of pull-up setting are disabled.

According to the mode setting function, while a reset signal is in "Low" state, the PF0/$\overline{\text{BOOT}}$ input and pull-up are enabled. At the rising edge of the reset signal, if PF0 is "High", the device enters single chip mode and boots from the on-chip flash memory. If PF0 is "Low", the device enters single BOOT mode and boots from the internal BOOT program. For details of single BOOT mode, refer to "Flash Memory Operation".

Note: In modes other than STOP1 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
Reserved	-	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 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.

10.2.6.5 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

10.2.6.6 PFOD (Port F open drain control register)

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

10.2.6.7 PFPUP (Port F pull-up control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	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.
7-1	PF7UP-PF1UP	R/W	Pull-up 0: Disable 1: Enable
0	PF0UP	R/W	Pull-up 0: - 1: Always set to "1"

10.2.6.8 PFIE (Port F input control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	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 serial bus interface (I2C/SIO), the external interrupt input,16-bits timer (TMRB) and the Detection function of USB connector connection with V-bus(USBPON).

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

The port G has three 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 other function registers at the same time.

Note 1: 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.

Note 2: Only when input is enabled, PG5 tolerate 5V inputs. Note that these pins cannot be pulled up over the power supply voltage when using as open-drain output.

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
Reserved	-	0x000C
Port G function register 3	PGFR3	0x0010
Port G function register 4	PGFR4	0x0014
Reserved	-	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 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.

10.2.7.6 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	-	-	-	-	-
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
4-0	−	R	Read as 0.

10.2.7.7 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 1: Open-drain

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

10.2.7.8 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

10.2.7.9 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 trace interface, the external interrupt input, and the 16bits-timer(TMRB).

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

The port H has two 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 other 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
Reserved	-	0x000C
Port H function register 3	PHFR3	0x0010
Reserved	-	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 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.

10.2.8.6 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 1: Open-drain

10.2.8.7 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

10.2.8.8 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 and the debug trace interface.

PI3, PI4, PI5, PI6 and PI7 are assigned as the debug interface after reset. PI7 is initialized as the $\overline{\text{TRST}}$ pin with input and pull-up enabled. PI6 is initialized as the TDI pin with input and pull-up enabled. PI3 is initialized as the TCK/SWCLK pin with input and pull-down enabled. PI4 is initialized as the TMS/SWDIO pin with input, output and pull-up enabled. PI5 is initialized as the TDO/SWV pin with output enabled. The other pins operate as general-purpose-ports, and input, output and pull-up are disabled.

Note 1: If PI4 or PI5 is configured as the TMS/SWDIO or TDO/SWV pin, output is enabled even in STOP1 mode regardless of the CGSTBYCR<DRVE> bit setting

Note 2: If PI3 is configured as the TCK/SWCLK pin, it prevents the low power consumption mode from being fully effective. Configure PI3 to function as a general-purpose port if the TCK/SWCLK is not used.

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 1: Open-drain

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	-	PI2UP	PI1UP	PI0UP
After reset	1	1	0	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7-6	PI7UP-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	−	R	Read as 0.
2-0	PI2UP-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-down 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	1	1	0	1	1	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
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, the 16-bit timer (TMRB).

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 both 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 STOP1 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 (TMRB).

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 both function registers at the same time.

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

To use the Port K as an analog input of the AD converter, disable input on PKIE and disable pull-up on PKPUP.

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 STOP1 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	PKDATA	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	−	NoR	NoR	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	-	-	
FT6	Output	Output	−	NoR	−	o	$\overline{\text{BOOT}}$ input enabled during reset

int: Interrupt input
−: Not exist
o: Exist
R: Forced disable during reset.
NoR: Unaffected by reset.

10.3.2 Type FT1

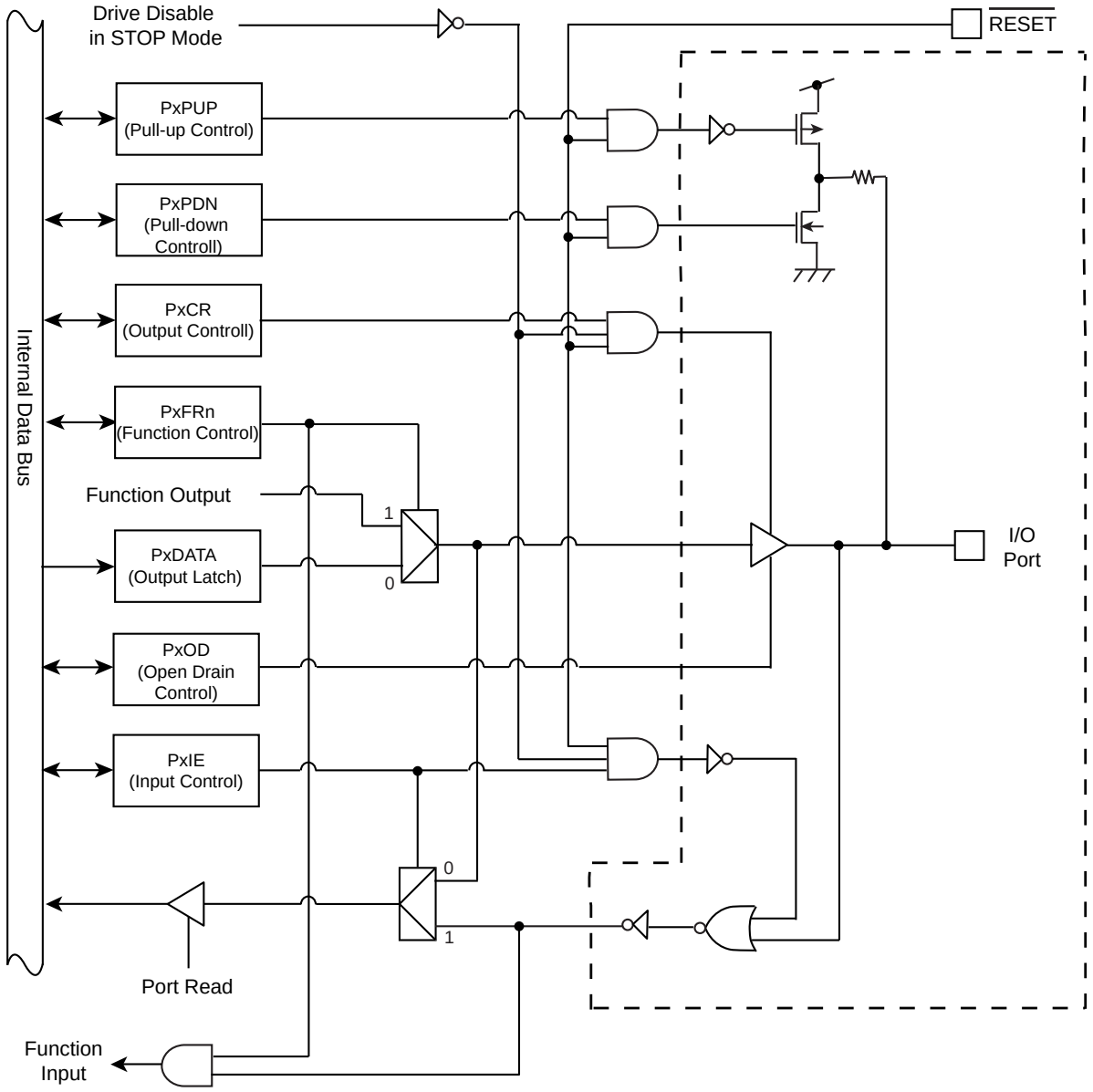


Figure 10-1 Port Type FT1

10.3.3 Type FT2

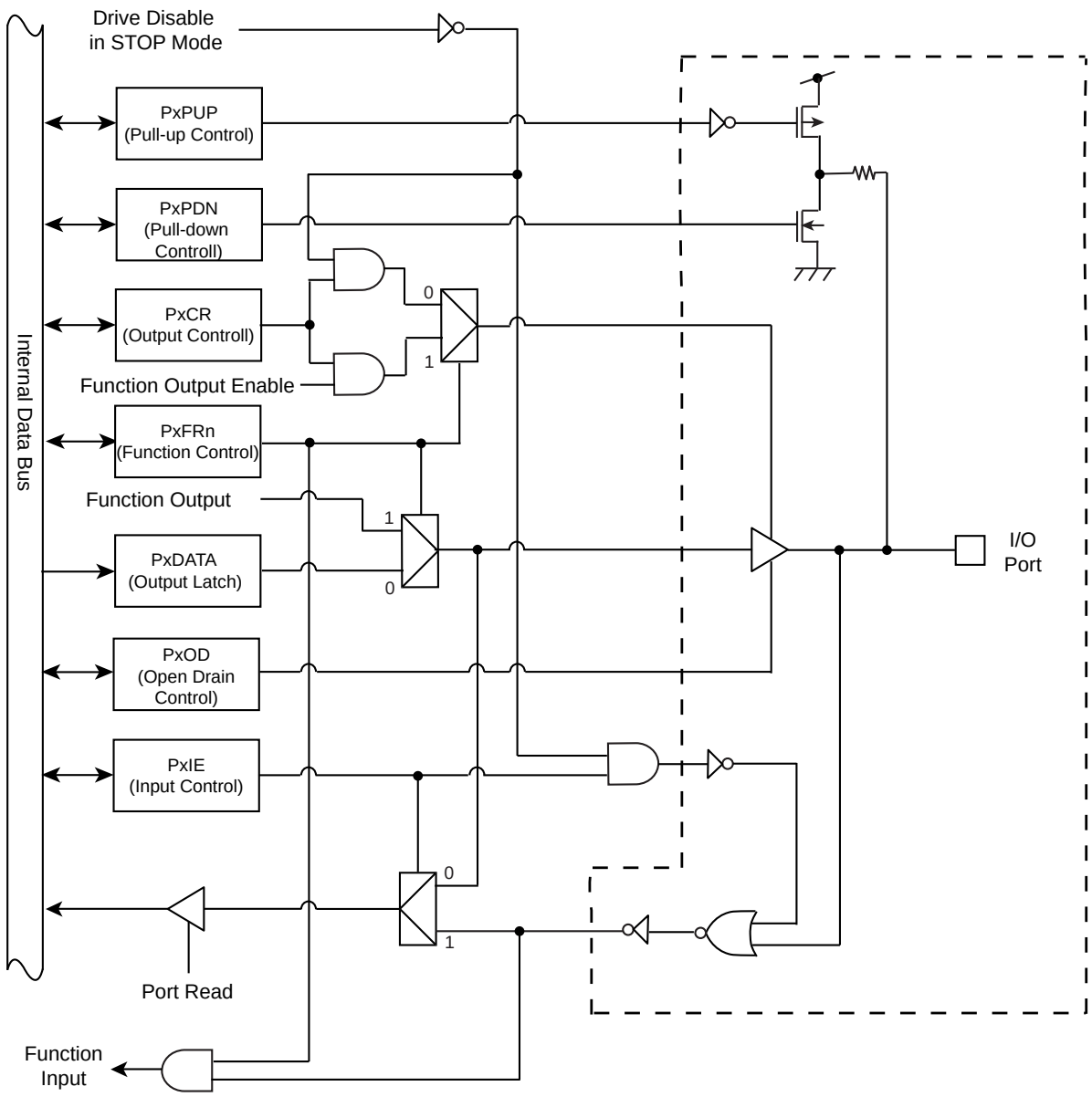


Figure 10-2 Port type FT2

Note: $\overline{\text{TRST}}$ has noise filter(30ns Typ.).

10.3.4 Type FT3

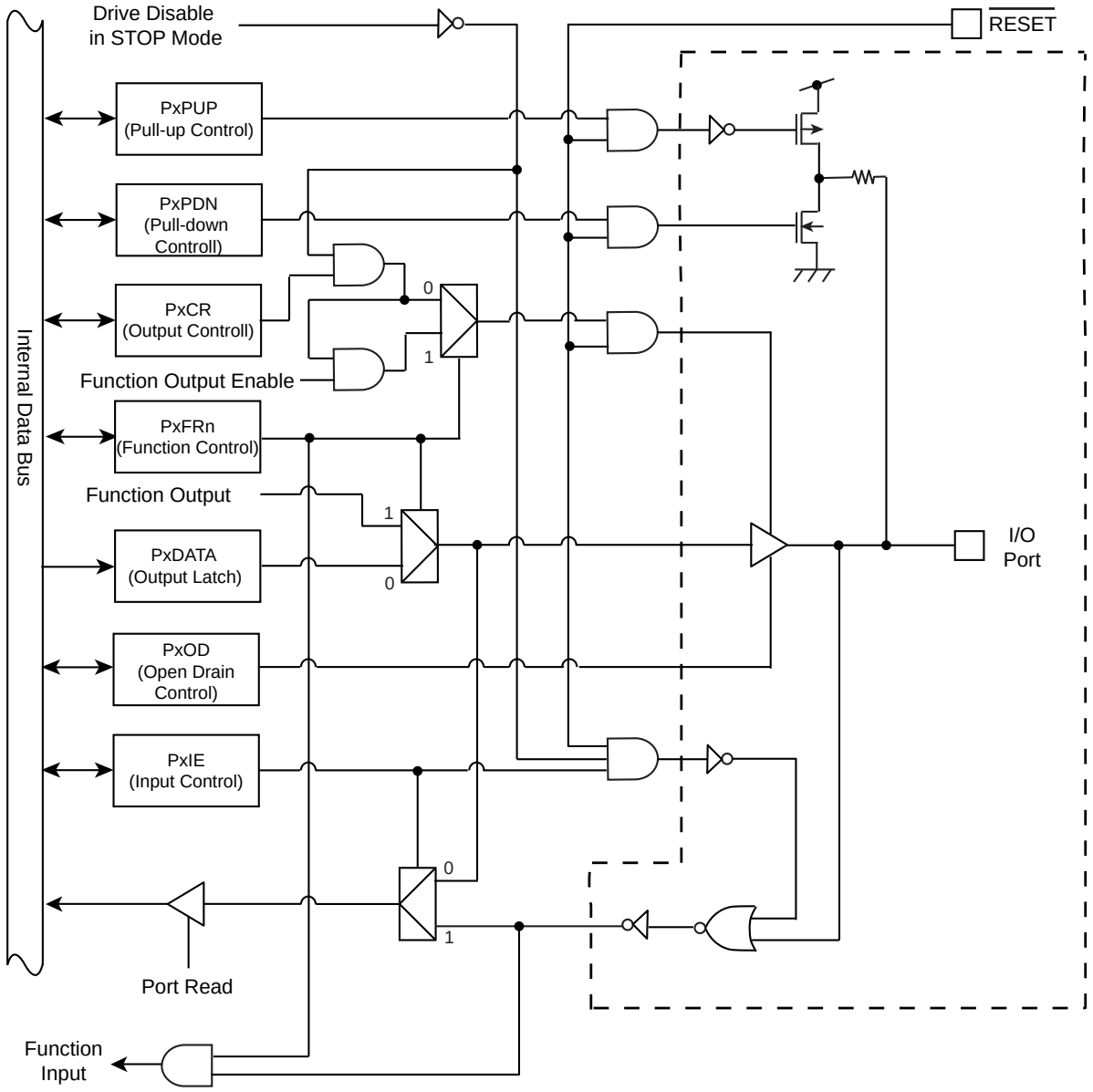


Figure 10-3 Port Type FT3

10.3.5 Type FT4

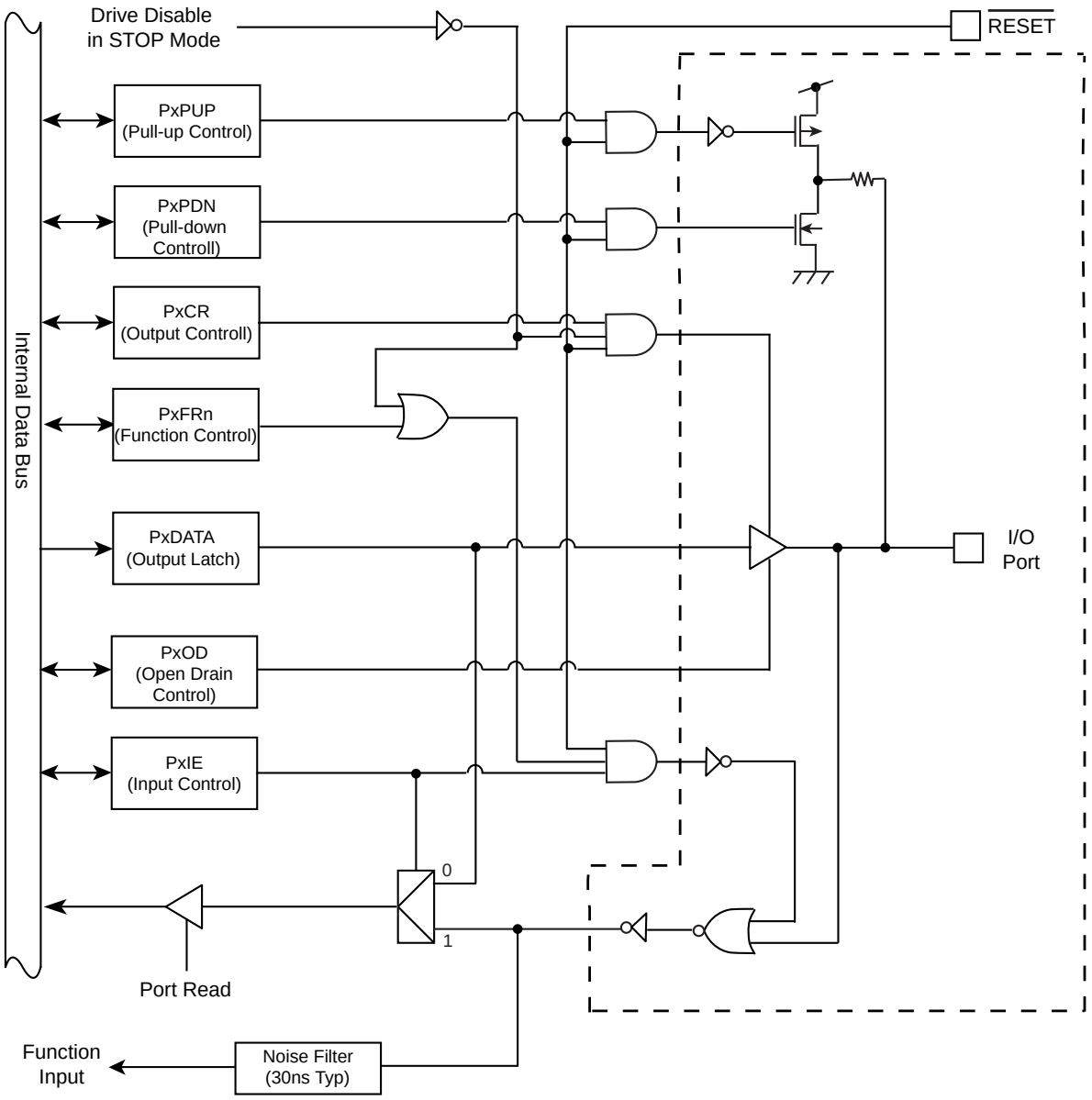


Figure 10-4 Port Type FT4

10.3.6 Type FT5

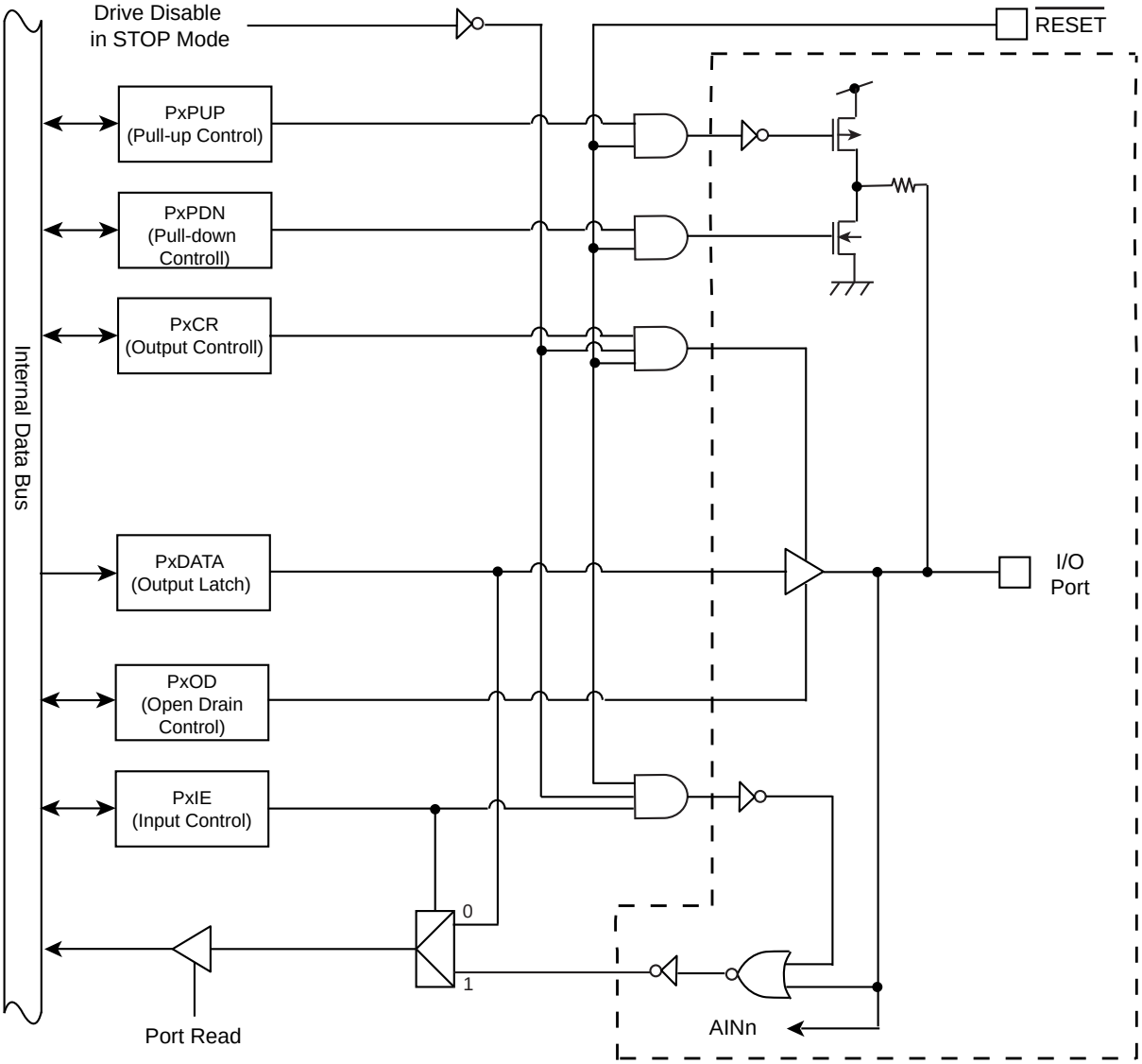


Figure 10-5 Port Type FT5

10.3.7 Type FT6

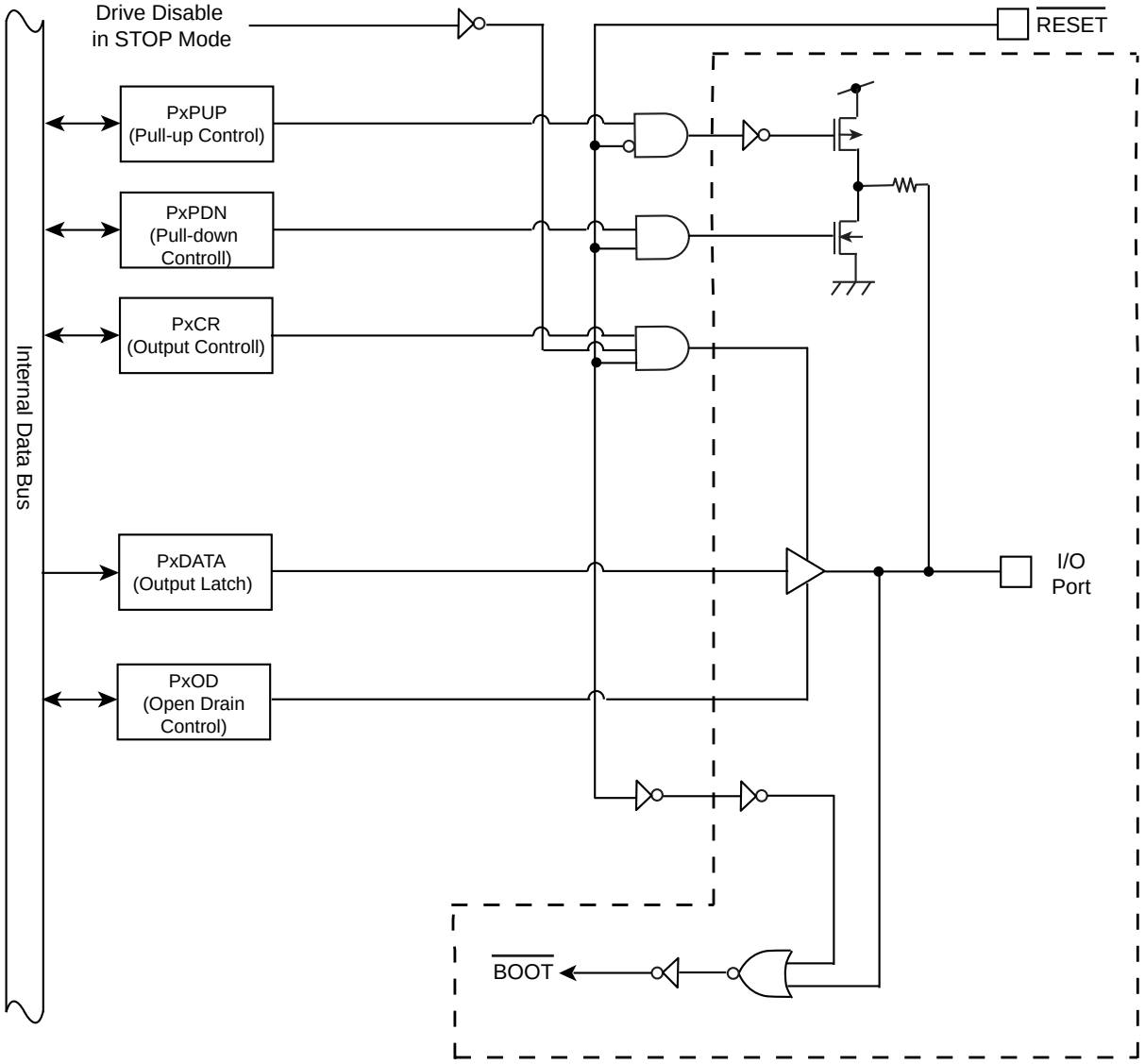


Figure 10-6 Port Type FT6

10.4 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	PAOD	PAPUP	PAIE
PA0	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PA1	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PA2	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PA3	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PA4	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PA5	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PA6	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PA7	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0

10.4.2 Port B Setting

Table 10-7 Port Setting List (Port B)

Pin	Port Type	Function	After re-set	PBCR	PBOD	PBPUP	PBIE
PB0	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PB1	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PB2	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PB3	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PB4	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PB5	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PB6	FT1	Input Port		0	x	x	1
		Output Port		1	x	x	0
PB7	FT1	Input Port		0	x	x	1
		Output Port		1	x	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	PCFR3	PCFR4	PCOD	PCPUP	PCIE
PC0	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	TXD1 (Output)		1	1	0	0	x	x	0
		TB2IN0 (Input)		0	0	1	0	x	x	1
PC1	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	RXD1 (Input)		0	1	0	0	x	x	1
		TB2IN1 (Input)		0	0	1	0	x	x	1
PC2	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	SCLK1 (Input)		0	1	0	0	x	x	1
		SCLK1 (Output)		1	1	0	0	x	x	0
		TB0OUT (Output)		1	0	1	0	x	x	0
		$\overline{\text{CTS1}}$ (Input)		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	PDFR3	PDOD	PDPUP	PDIE
PD0	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT1	TB7OUT (Output)		1	1	x	x	0
PD1	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT1	TB8OUT (Output)		1	1	x	x	0
PD2	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT1	TB9OUT (Output)		1	1	x	x	0
PD3	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT1	$\overline{\text{ADTRG}}$ (Input)		0	1	x	x	1
PD4	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
PD5	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
PD6	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
PD7	FT1	Input Port		0	0	x	x	1
		Output Port		1	0	x	x	0
	FT1	SCOUT (Output)		1	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	PEFR3	PEFR4	PEOD	PEPUP	PEIE
PE0	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	TXD0 (Output)		1	1	0	0	x	x	0
PE1	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	RXD0 (Input)		0	1	0	0	x	x	1
PE2	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	SCLK0 (Input)		0	1	0	0	x	x	1
		SCLK0 (Output)		1	1	0	0	x	x	0
		TB2OUT (Output)		1	0	1	0	x	x	0
		$\overline{\text{CTS0}}$ (Input)		0	0	0	1	x	x	1
PE3	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT4	INT5 (Input)		0	1	0	0	x	x	1
	FT1	TB3OUT (Output)		1	0	1	0	x	x	0
PE4	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	SO1 (Output)		1	1	0	0	x	x	0
		SDA1 (I/O)		1	1	0	0	1	x	1
PE5	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	SI1 (Input)		0	1	0	0	x	x	1
		SCL1 (I/O)		1	1	0	0	1	x	1
PE6	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	SCK1 (Input)		0	1	0	0	x	x	1
		SCK1 (Output)		1	1	0	0	x	x	0
PE7	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT4	INT4 (Input)		0	1	0	0	x	x	1

10.4.6 Port F Setting

Table 10-11 Port Setting List (Port F)

Pin	Port Type	Function	After reset	PFCR	PFFR2	PFFR3	PFOD	PFPUP	PFIE
PF0	FT6	Output Port		1	0	0	x	1	0
	FT1	TB6OUT (output)		1	0	1	x	1	0
PF1	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
PF2	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
PF3	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
PF4	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT4	INT6 (Input)		0	1	0	x	x	1
	FT1	TB5IN0 (Input)		0	0	1	x	x	1
PF5	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT4	INT7 (Input)		0	1	0	x	x	1
	FT1	TB5IN1 (Input)		0	0	1	x	x	1
PF6	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
PF7	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0

Note: The PF0 input and pull-up are enabled and act as $\overline{\text{BOOT}}$ input pin while a $\overline{\text{RESET}}$ is in "Low" state.

10.4.7 Port G Setting

Table 10-12 Port Setting List (Port G)

Pin	Port Type	Function	After reset	PGCR	PGFR1	PGFR3	PGFR4	PGOD	PGPUP	PGIE
PG0	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	SO0 (Output)		1	1	0	0	x	x	0
	FT1	SDA0 (I/O)		1	1	0	0	1	x	1
PG1	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	SI0 (Input)		0	1	0	0	x	x	1
		SCL0 (I/O)		1	1	0	0	1	x	1
		TB3IN0 (Input)		0	0	1	0	x	x	1
PG2	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	SCK0 (Input)		0	1	0	0	x	x	1
		SCK0 (Output)		1	1	0	0	x	x	0
		TB3IN1 (Input)		0	0	1	0	x	x	1
PG3	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT4	INT0 (Input)		0	1	0	0	x	x	1
	FT1	TB4IN0 (Input)		0	0	1	0	x	x	1
PG4	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT1	TB4IN1 (Input)		0	0	1	0	x	x	1
PG5	FT1	Input Port		0	0	0	0	x	x	1
		Output Port		1	0	0	0	x	x	0
	FT4	INT1 (Input)		0	1	0	0	x	x	1
	FT1	USBPON (Input)		0	0	0	1	x	x	1

Note: PG5 is 5 voltage tolerant input pin. However, the programmable pull-up voltage is up to DVDD3A voltage level. So the pull-up of 5 voltage tolerant input pin should be designed by outer pull-up resistance.

10.4.8 Port H Setting

Table 10-13 Port Setting List (Port H)

Pin	Port Type	Function	After reset	PHCR	PHFR1	PHFR3	PHOD	PHPUP	PHIE
PH0	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT1	TRACEDATA2 (Output)		1	1	0	0	0	0
PH1	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT1	TRACEDATA3 (Output)		0	1	0	0	0	0
PH2	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT1	TB4OUT (Output)		1	0	1	x	x	0
PH3	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT1	TB5OUT (Output)		1	0	1	x	x	0
PH4	FT1	Input Port		0	0	0	x	x	1
		Output Port		1	0	0	x	x	0
	FT4	INT8 (Input)		0	0	1	x	x	1

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
	FT1	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
	FT1	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
	FT1	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
PJ1	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
PJ2	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
PJ3	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
PJ4	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
PJ5	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	0
PJ6	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	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
	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
	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
	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
	FT1	TB6IN0 (Input)		0	0	1	x	1
PK3	FT1	Input Port		0	0	0	x	1
		Output Port		1	0	0	x	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 : CH0 Low : CH1		Fixed
FIFO	4word × 2ch (1word = 32bit)		-
Bus width	8/16/32bit		Settable individually for transfer source and destination.
Burst size	1/4/8/16/32/64/128/256		-
Number of transfers	up to 4095		-
Address	Transfer source address	increment not increment	It is possible to specify whether Source and Destination addresses should increment or should not increment. (Address wrapping is not supported.)
	Transfer destination address	increment not increment	
Endian	Littleendian is supported.		-
Transfer type	Peripheral to Memory Memory to Peripheral Memory to Memory Peripheral to Peripheral		When "Memory to Memory" is selected, hardware start for DMA startup is not supported. Refer to the DMACxCnConfiguration for more information. Particular peripheral can be assigned as Source or Distination when "Peripheral to Peripheral" is selected. Regarding to peripheral assigned, refer to "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

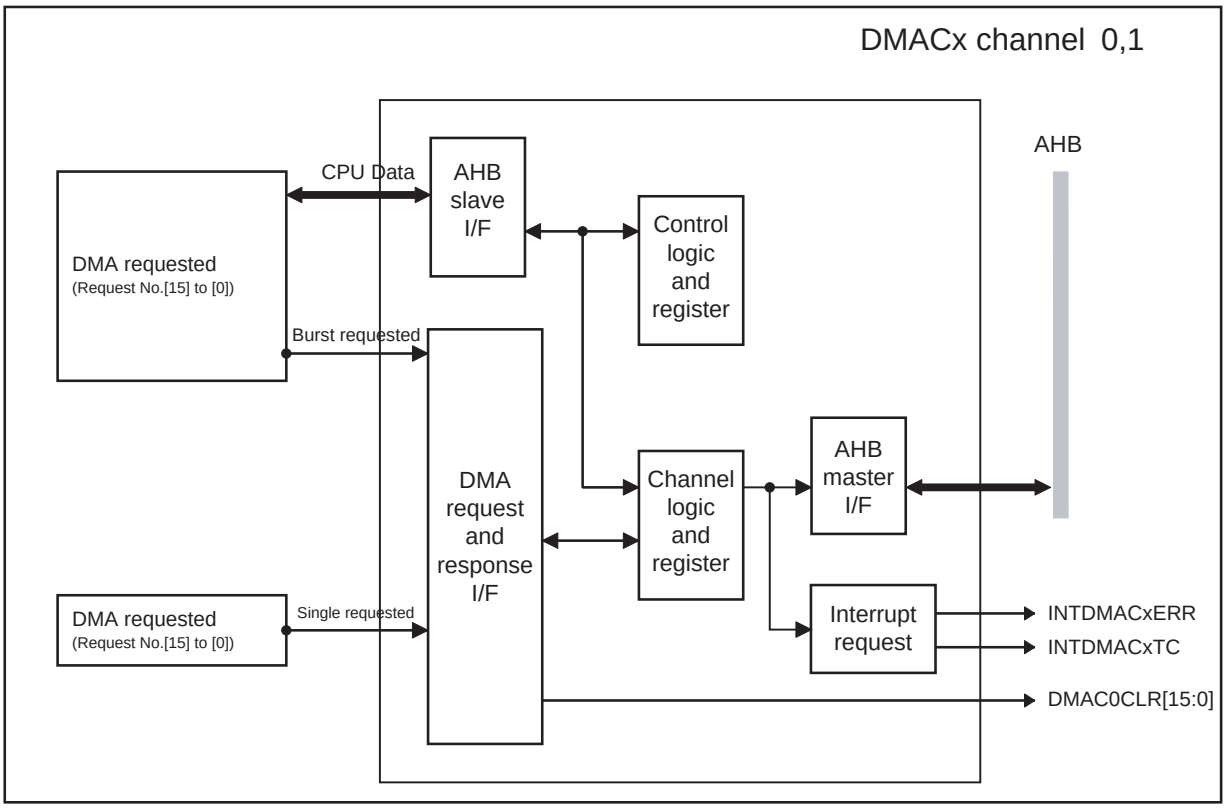


Figure 11-1 DMAC Block diagram

11.4 Product information of TMPM365FYXBG

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

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	Normal AD Conversion End	-
12	I2C0 / SIO0 Reception (Note)	-
13	I2C0 / SIO0 Transmission (Note)	-
14	I2C1 / SIO1 Reception (Note)	-
15	I2C1 / SIO1 Transmission (Note)	-

Note: DMAC can be used in I2C mode. It must not be used in SIO mode.

11.4.3 Interrupt request

Transfer complete interrupt	Error interrupt
INTDMAC0TC	INTDMAC0ERR

11.4.4 Base address of registers

Base Address
0x4000_0000

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.

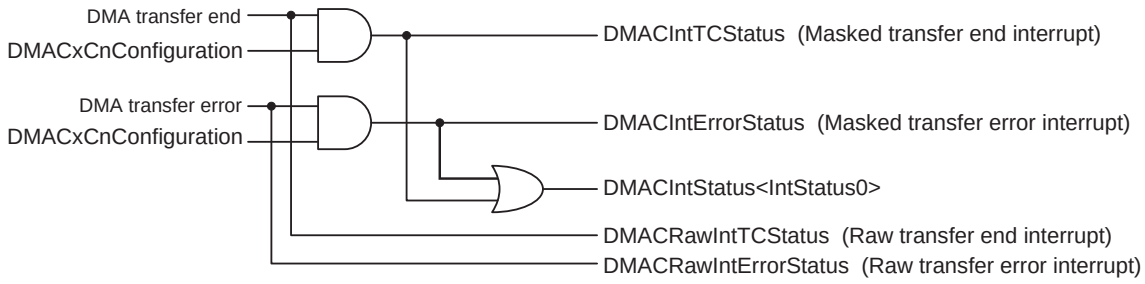


Figure 11-2 Interrupt-related block diagram

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 11-4 for the setting value.
20-18	Swidth[2:0]	R/W	Transfer source bit width 000: Byte (8 bits) 001: Half-word (16 bits) 010 : Word (32 bits) other: Reserved Refer to Table 11-4 for the setting value.
17-15	DBSize[2:0]	R/W	Transfer destination burst size: (Note 1) 000: 1 beat 100: 32 beats 001: 4 beats 101: 64 beats 010: 8 beats 110: 128 beats 011: 16 beats 111: 256 beats Refer to Table 11-4 for the setting value.

Bit	Bit Symbol	Type	Function
14-12	SBSize[2:0]	R/W	Transfer source burst size: (Note 1) 000: 1 beat 100: 32 beats 001: 4 beats 101: 64 beats 010: 8 beats 110: 128 beats 011: 16 beats 111: 256 beats Refer to Table 11-4 for the setting value.
11-0	TransferSize [11:0]	R/W	Set the total number of transfers. Amount of transfer data per a bit width specified by transfer source bit witdth (4 bytes / 2 bytes / 1byte) is set into <TransderSize[11:0]>. Because the burst size shows amount of transfer data at every DMA request internally, amount of transfer data is never changed even if any burst size is set when transfer source bit width and total number of transfers are not changed. The value of <TransferSize[11:0]> is decremented to 0 by the DMA transferring. If this is read, the value which is the number of data not to transfer. The total number of transfers is used as the unit for the transfer source bit width. For examples: When <Swidth>="000" (8bit), the number of transfers is expressed in the units of byte. When <Swidth>="001" (16bit), the number of transfers is expressed in the units of half word. When <Swidth>="010" (32bit), the number of transfers is expressed in the units of word.

Note: The burst size to be set with DBsize and SBsize has no connections with the HBURST for the AHB bus.

Table 11-4 How to decide the value of <Dwidth[2:0]>,<Swidth[2:0]>, <DBSize[2:0]>, <SBSize[2:0]>

<Dwidth[2:0]> / <Swidth[2:0]>	Set the number so that the following expression is satisfied: Transfer source bit width × Total number of transfers = Transfer destination bit width × N (N : Integer number)
	(ex.1) Bit width of transfer source:8 bit, bit width of transfer destination:32 bit, total number of transfers:25 times 8 bit × 25 times = 200 bit (25 byte) N = 200 ÷ 32 = 6.25 word Since 6.25 is not an integer number, the above setting is invalid. If the transfer source bit width is smaller than the transfer destination bit width, care must be taken when setting the total number of transfers.
	(ex.2) Bit width of transfer source :32 bit, bit width of transfer destination:16 bit, total number of transfers: 13 times 32 bit × 13 times = 416 bit (13 word) N = 416 ÷ 16 = 26 half_word Since 26 is an integer number, the above setting is valid.
<DBSize[2:0]> / <SBSize[2:0]>	When "Peripheral to Memory" or "Memory to Peripheral" is performed, peripheral circuits generates DMA request signal to indicate the preparation is ready. This signal triggers to execute data transfers. (In the case of "Memory to Memory", only software start is used.) Set the burst size to define the amount of data transferred from peripherals per DMA request signal. This register is used with FIFO buffer that can be contained multiple data.

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 releasing 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 (Note 2) 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 (Note 2) 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.

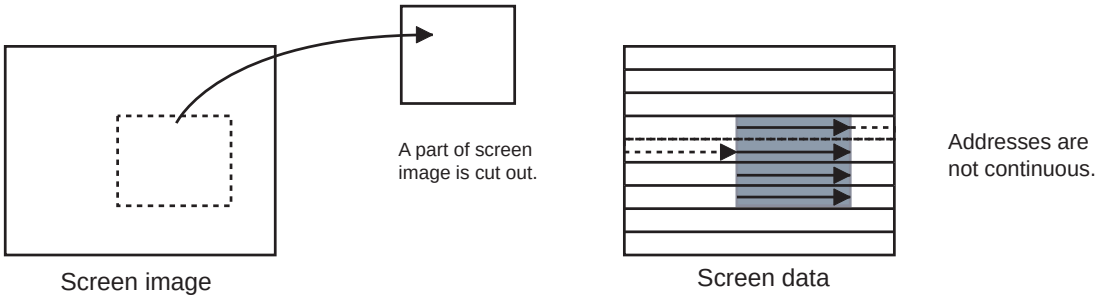
Note 2: When DMACxENableChns<EnabledCHx> is enabled and the corresponding DMACxCnConfiguration<Halt> is set to "1", write them after channel enable bit (E:bit0) is clear to "0". Without this, in the case of the slave error is occurred when writing them, the error is recovered by reset. Regarding slave 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

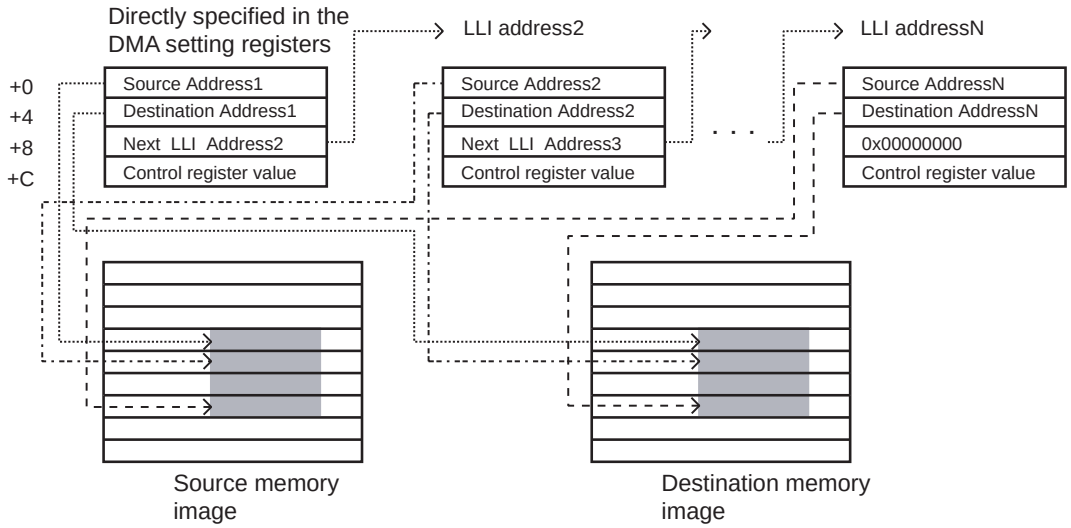
To operate the scatter/gather function, a transfer source and source data areas need to be defined by creating a set of Linked Lists first.

Each setting is called LLI (LinkedList).

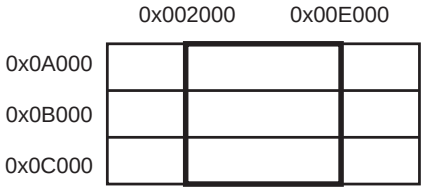
Each LLI controls the transfer of one block of data. Each LLI indicates normal DMA setting and controls transfer of successive data. Each time each DMA transfer is complete, the next LLI setting will be loaded to continue the DMA operation (Daisy Chain).

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.

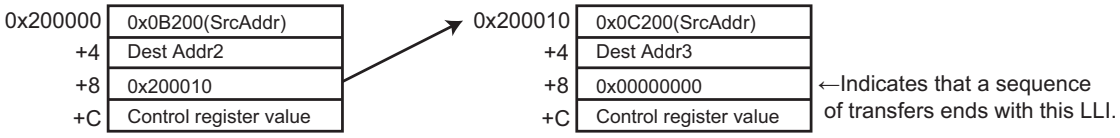


When transferring data in the area enclosed by the square



	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. 16-bit Timer/Event Counters(TMRB)

12.1 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.

12.2 Differences in the Specifications

TMPM365FYXBG 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 12-1.

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 12-1 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	ADC 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	–	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	–	–	

12.3 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.

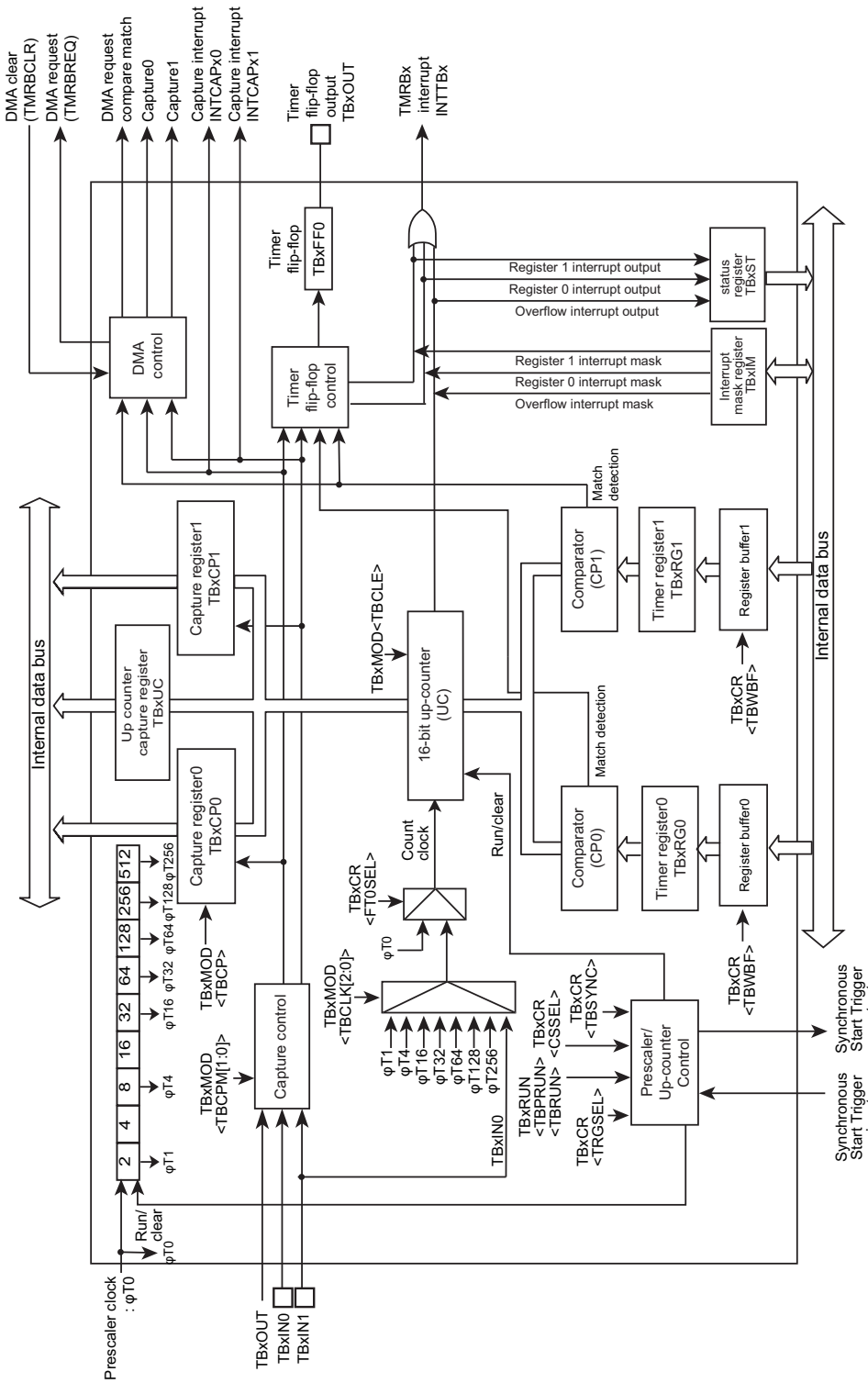


Figure 12-1 TMRBx Block Diagram(x= 0 to 9)

12.4 Registers

12.4.1 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.

12.4.2 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.

12.4.3 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.

12.4.4 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	FT0SEL	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	FT0SEL	R/W	Select source clock φT0 0: Select except φT0 (clock selection at TBxMOD<TBCLK[2:0]) 1:Select φT0
3	I2TB	R/W	Operation at IDLE mode 0: Stop 1:Operation
2	TBINSEL	R/W	Write "0".
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

12.4.5 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: TB9OUT)
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. As for the source clock of TMRBx, φT0 is chosen if it is set as TBxCR<FT0SEL>="1" irrespective of the pre-set value of <TBCLK [2:0]>. As for the source clock of TMRBx, the following clocks are chosen if it is set as TBxCR<FT0SEL>="0". 000: TBxIN0 pin input 001: φT1 010: φT4 011: φT16 100: φT32 101: φT64 110: φT128 111: φT256

Note 1: Do not make any changes of TBxMOD register while corresponding TMRBx is running.

Note 2: When the channel of TBxMOD register is 7, 8 or 9, the setting of <TBCPM[1:0]>="11" is prohibited.

12.4.6 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".

12.4.7 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.

12.4.8 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.

12.4.9 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.

12.4.10 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.

12.4.11 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.

12.4.12 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.

12.4.13 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.

12.4.14 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 1: When mask configuration by TBxIM register is valid, DMA request does not issue even if it is enabled.
Note 2: The assignment of DMA request factor differs by channel, TMRB0 to 9 . Refer to Chapter "DMAC " for details.

12.5 Description of Operations for Each Circuit

The channels operate in the same way, except for the differences in their specifications as shown in Table 12-1 .

12.5.1 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 12-2 and Table 12-3 show prescaler output clock resolutions.

Table 12-2 Prescaler Output Clock Resolutions(fc = 48MHz,1/fc = 0.02μs)

Select peripheral clock CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function			
			TBxCR<FT0SEL>="1" (Note2)	TBxCR<FT0SEL>="0" (Note1)		
			φT0	φT1	φT4	φT16
0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^0$ (0.02 μs) (Note2)	$fc/2^1$ (0.04 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)
		001 (fperiph/2)	$fc/2^1$ (0.04 μs)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
		010 (fperiph/4)	$fc/2^2$ (0.08 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		011 (fperiph/8)	$fc/2^3$ (0.17 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		100 (fperiph/16)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
	100 (fc/2)	000 (fperiph/1)	$fc/2^1$ (0.04 μs) (Note2)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
		001 (fperiph/2)	$fc/2^2$ (0.08 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		010 (fperiph/4)	$fc/2^3$ (0.17 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		100 (fperiph/16)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
	101 (fc/4)	000 (fperiph/1)	$fc/2^2$ (0.08 μs) (Note2)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		001 (fperiph/2)	$fc/2^3$ (0.17 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		010 (fperiph/4)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		011 (fperiph/8)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		100 (fperiph/16)	$fc/2^6$ (1.33 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
		101 (fperiph/32)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{12}$ (85.33 μs)
	110 (fc/8)	000 (fperiph/1)	$fc/2^3$ (0.17 μs) (Note2)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		001 (fperiph/2)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		010 (fperiph/4)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		011 (fperiph/8)	$fc/2^6$ (1.33 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
		100 (fperiph/16)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{12}$ (85.33 μs)
		101 (fperiph/32)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{13}$ (170.67 μs)
	111 (fc/16)	000 (fperiph/1)	$fc/2^4$ (0.33 μs) (Note2)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		001 (fperiph/2)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		010 (fperiph/4)	$fc/2^6$ (1.33 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
		011 (fperiph/8)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{12}$ (85.33 μs)
		100 (fperiph/16)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{13}$ (170.67 μs)
		101 (fperiph/32)	$fc/2^9$ (10.7 μs)	$fc/2^{10}$ (21.34 μs)	$fc/2^{12}$ (85.34 μs)	$fc/2^{14}$ (341.34 μs)

Table 12-2 Prescaler Output Clock Resolutions(fc = 48MHz,1/fc = 0.02μs)

Select peripheral clock CGSYSCR <FPSEL>	Clock gear value CGSYSCR <GEAR[2:0]>	Select prescaler clock CGSYSCR <PRCK[2:0]>	Prescaler output clock function			
			TBxCR<FT0SEL>="1" (Note2)	TBxCR<FT0SEL>="0" (Note1)		
			φT0	φT1	φT4	φT16
1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^0$ (0.02 μs) (Note2)	$fc/2^1$ (0.04 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)
		001 (fperiph/2)	$fc/2^1$ (0.04 μs)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
		010 (fperiph/4)	$fc/2^2$ (0.08 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		011 (fperiph/8)	$fc/2^3$ (0.17 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		100 (fperiph/16)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (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)	$fc/2^1$ (0.04 μs) (Note2)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
		010 (fperiph/4)	$fc/2^2$ (0.08 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		011 (fperiph/8)	$fc/2^3$ (0.17 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		100 (fperiph/16)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (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)	$fc/2^2$ (0.08 μs) (Note2)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		011 (fperiph/8)	$fc/2^3$ (0.17 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		100 (fperiph/16)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (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)	$fc/2^3$ (0.17 μs) (Note2)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		100 (fperiph/16)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (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)	–	–	$fc/2^4$ (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)	$fc/2^4$ (0.33 μs) (Note2)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (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: Only when it is set as TBxCR<FT0SEL>="1", the prescaler output clock φTn can be used also on condition of φTn ≤ fsys.

Note 3: Do not change the clock gear while the timer is operating.

Note 4: "–" denotes a setting prohibited.

Table 12-3 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^{15} (682.66 μs)	fc/2^{16} (1365.34 μs)	fc/2^{17} (2730.67 μs)	fc/2^{18} (5461.34 μs)

Table 12-3 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.

12.5.2 Up-counter (UC)

UC is a 16-bit binary counter.

- Source clock

  UC source clock, specified by TBxMOD<TBCLK[2:0]> and TBxCR<FT0SEL>, can be selected from φT0, φT1, φT4, φT16, φT32, φT64, φT128 or φT256 or the external clock of the TBxIN0 pin.

  Only when it is set as TBxCR<FT0SEL>="1", the prescaler output clock φTn ≤ fsys.

- 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.

12.5.3 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.

12.5.4 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>.

12.5.5 Capture registers (TBxCP0, TBxCP1)

This register captures an up-counter (UC) value.

12.5.6 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.

12.5.7 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.

12.5.8 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.

12.5.9 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.

12.6 Description of Operations for Each Mode

12.6.1 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

12.6.2 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

12.6.3 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)

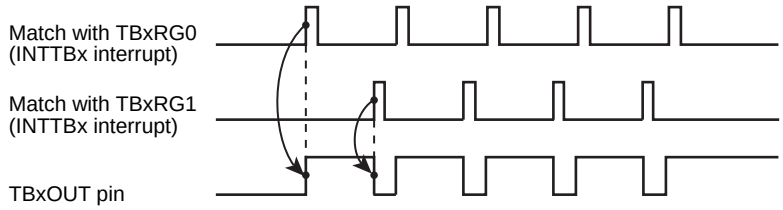


Figure 12-2 Example of Output of Programmable Pulse Generation (PPG)

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.

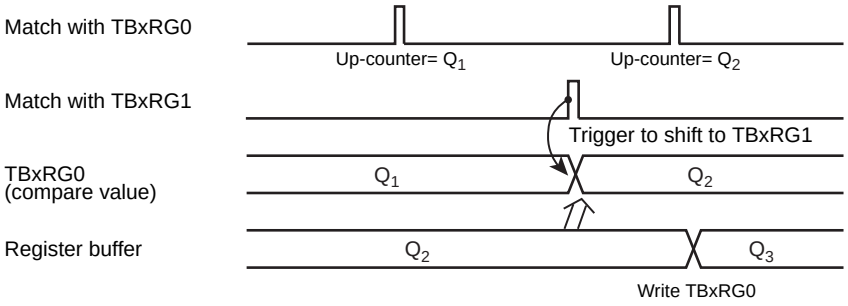


Figure 12-3 Register Buffer Operation

The block diagram of this mode is shown below.

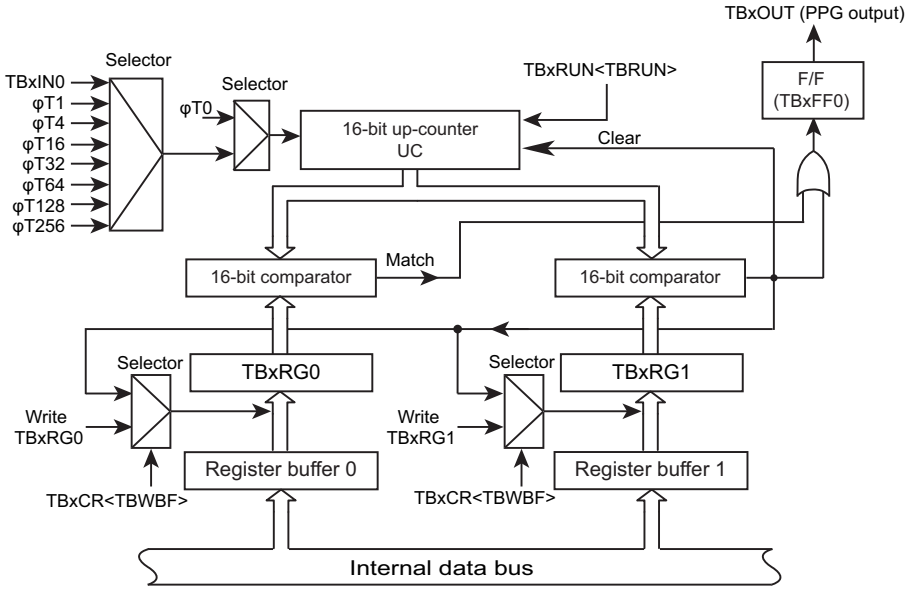


Figure 12-4 Block Diagram of 16-bit PPG Mode

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

12.6.4 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 TMPM365FYXBG, 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: 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 2: TMRB0 and TMRB4 are the master clocks of the timer synchronous mode. Therefore, their <TBSYNC> bit must be set to "0".

Note 3: This mode cannot be applied to TMRB8 and TMRB9.

12.6.5 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.

12.7 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

12.7.1 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 12-5 One-shot Pulse Output (With Delay)".

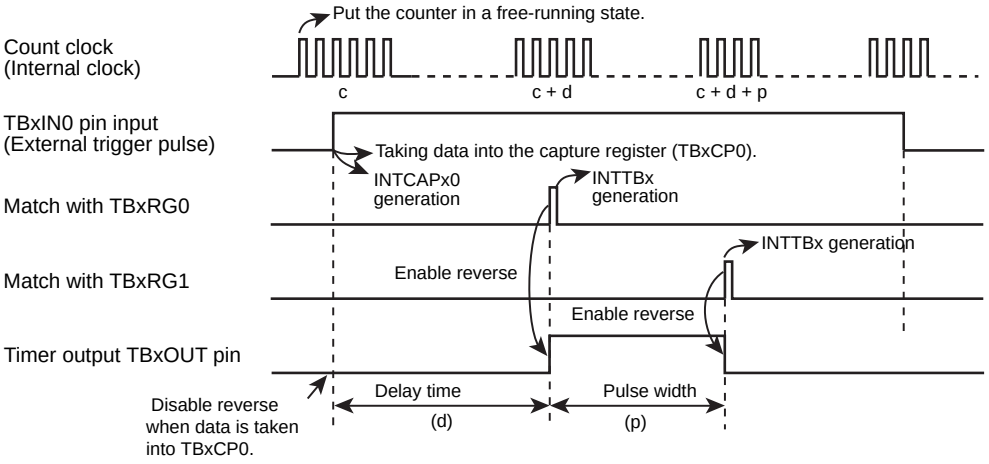


Figure 12-5 One-shot Pulse Output (With Delay)

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.

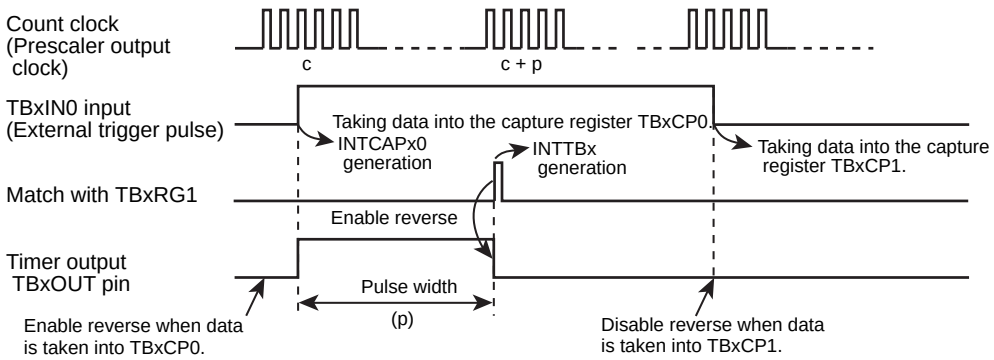


Figure 12-6 One-shot Pulse Output Triggered by an External Pulse (Without Delay)

12.7.2 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).

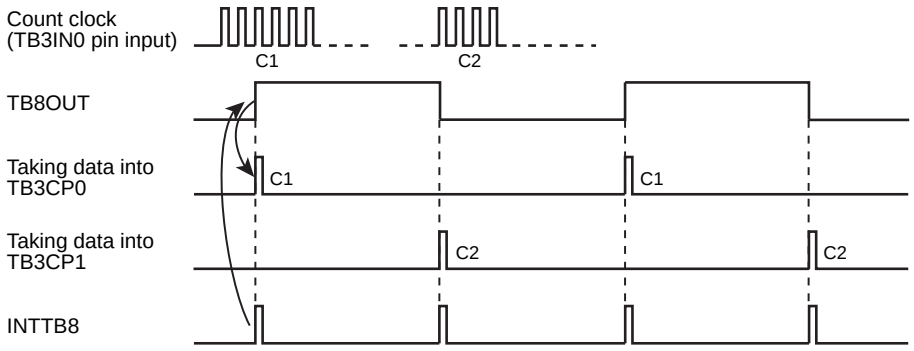


Figure 12-7 Frequency Measurement

12.7.3 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 12-8 Pulse Width Measurement" and this difference is multiplied by the cycle of the prescaler output clock to obtain the "Low" level width.

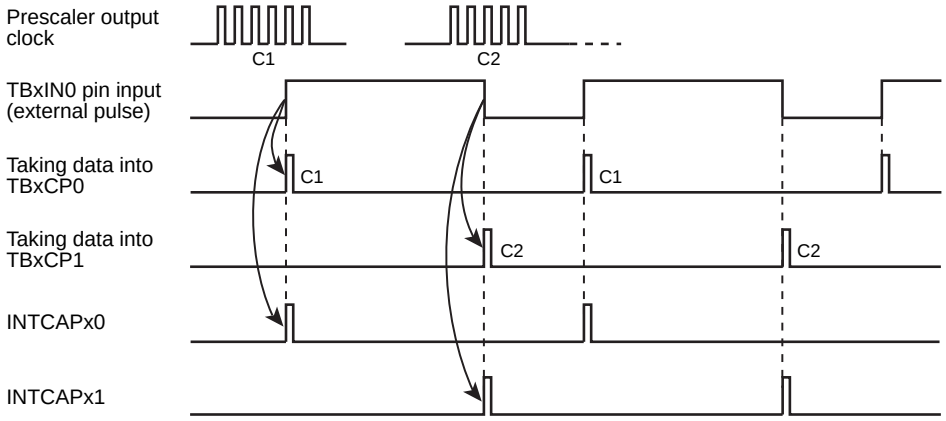


Figure 12-8 Pulse Width Measurement

12.7.4 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.

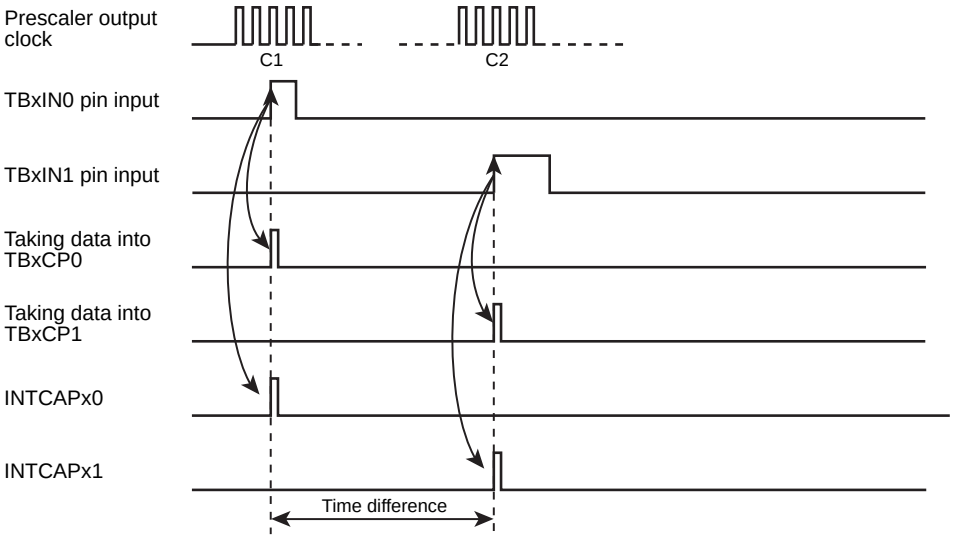


Figure 12-9 Time Difference Measurement

13. USB Device Controller (USBD)

This section describes about USB device controller.

An endpoint is described as EP in this section.

13.1 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 13-1 Endpoints

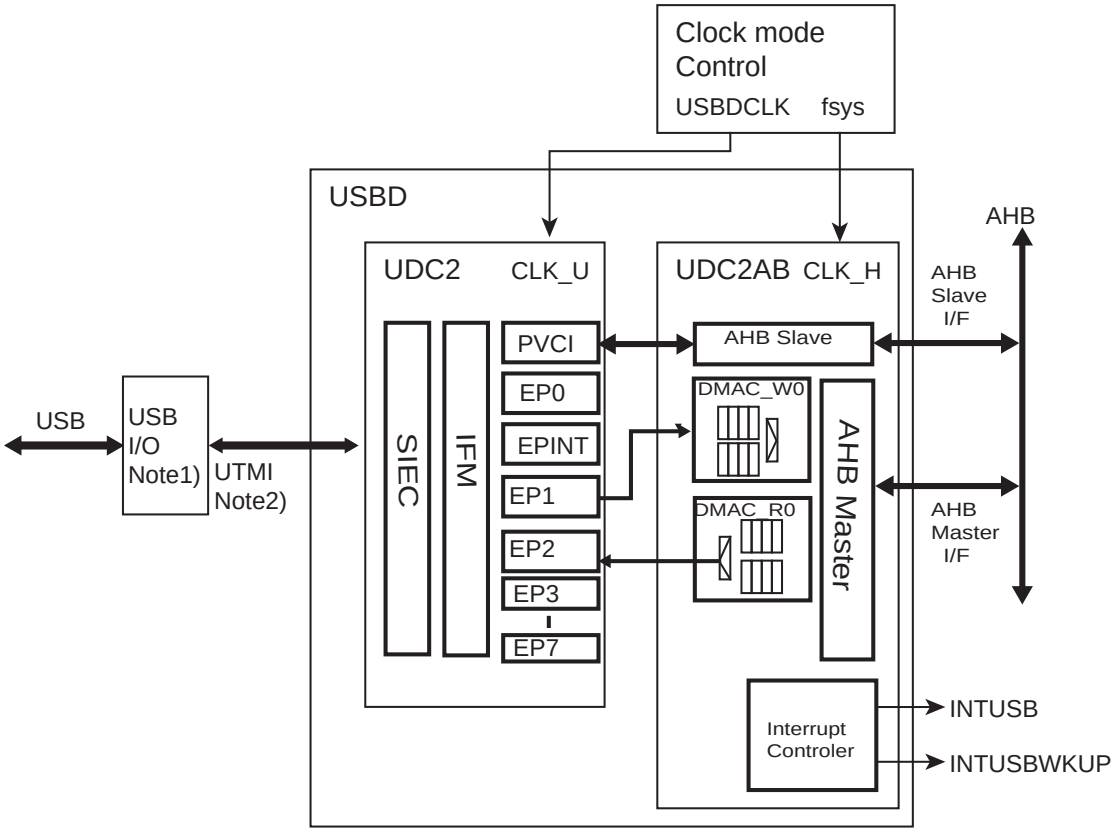
EP0:	Control	64byte × 1 FIFO
EP1:	Control / Interrupt / Bulk / Isochronous (IN)	64byte × 2 FIFO
EP2:	Control / Interrupt / Bulk / Isochronous (OUT)	64byte × 2 FIFO
EP3:	Control / Interrupt / Bulk / Isochronous (IN)	64byte × 2 FIFO
EP4:	Control / Interrupt / Bulk / Isochronous (OUT)	64byte × 2 FIFO
EP5:	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

13.2 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, "13.2.1 AHB Bus Bridge (UDC2AB)" describes the configuration of the UDC2AB, and "13.2.2 Toshiba USB-Spec2.0 Device Controller (UDC2)" describes that of the UDC2.



Note1) TMPM365FYXBG 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 13-1 Block diagram of the USB device controller

13.2.1 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.

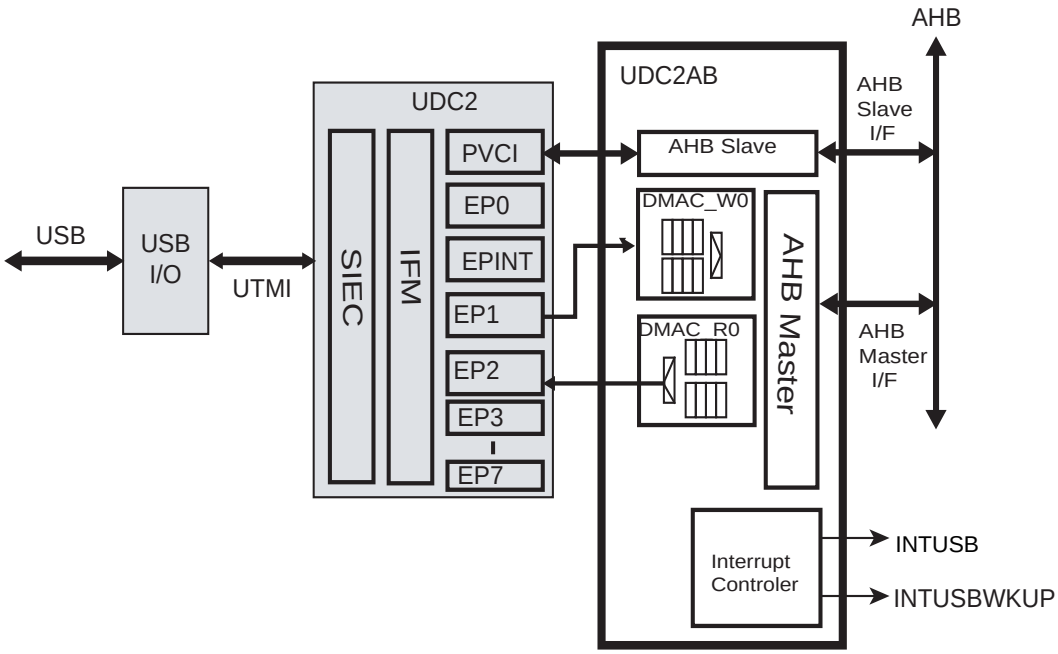


Figure 13-2 UDC2AB Block Diagram

13.2.1.1 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 13-2 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.

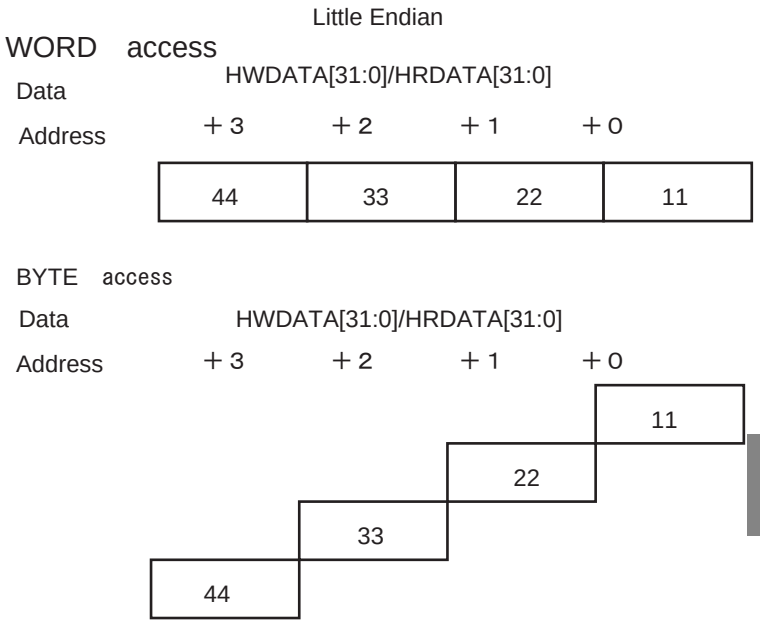


Figure 13-3 Image of Endian conversion in AHB Master function

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.

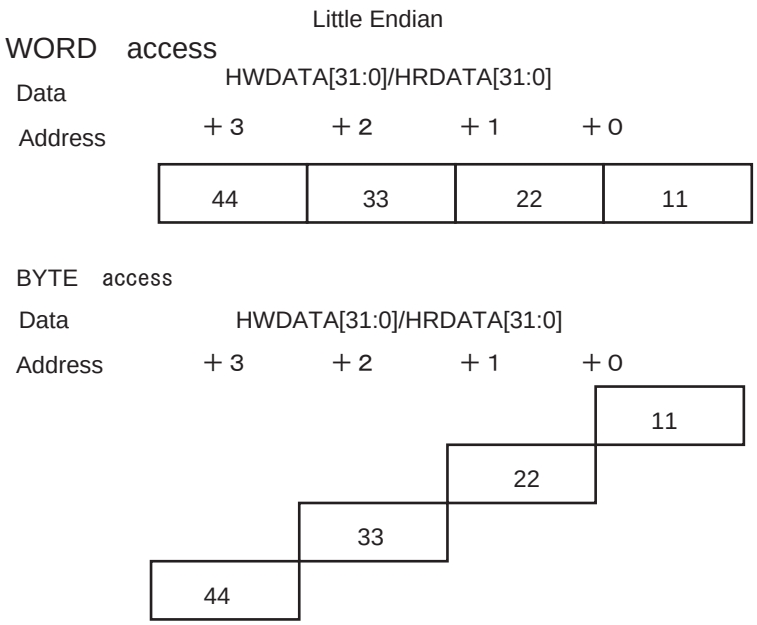


Figure 13-4 Image of Endian conversion in AHB Slave function

13.2.1.2 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).

13.2.1.3 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 TMPM365FYXBG.

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 "13.5.5.2 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.

13.2.2 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.

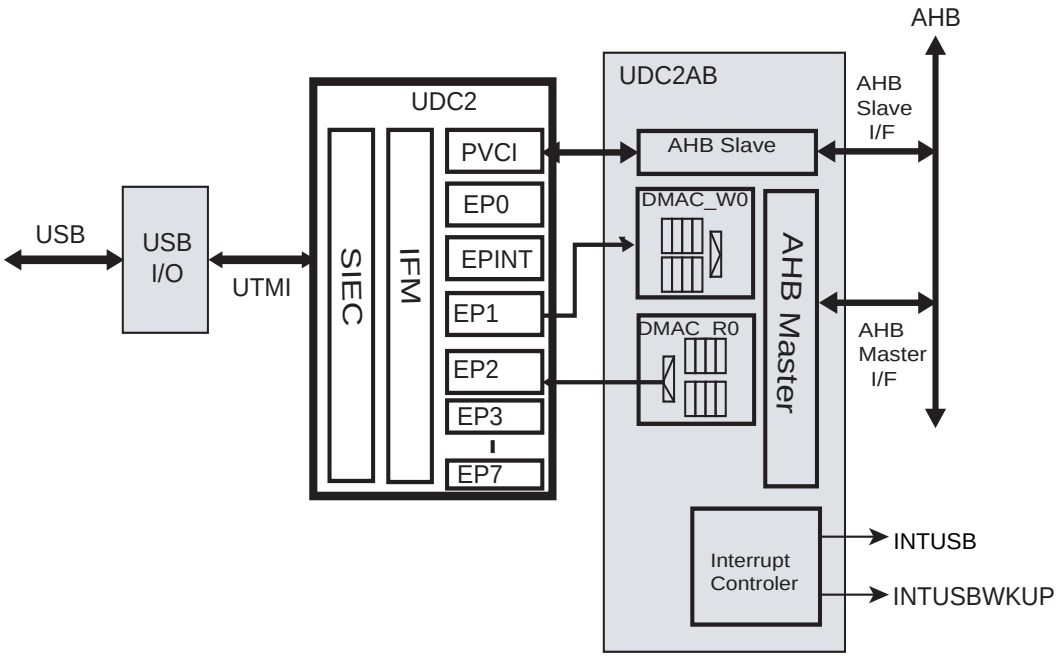


Figure 13-5 Block diagram of UDC2

13.2.2.1 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)

13.2.2.2 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.

13.2.2.3 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.

13.3 How to connect with the USB bus

The circuit below shows how to connect TMPM365FYXBG 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.

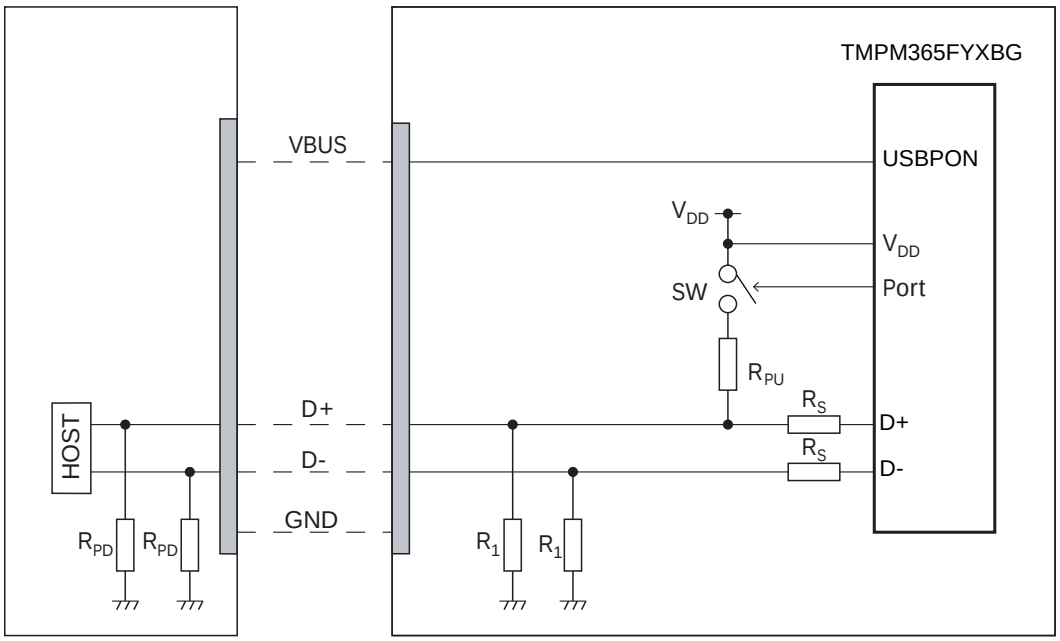


Figure 13-6 Connection example of the USB bus and TMPM365FYXBG.

Note: R_1=500kΩ or higher (recommended value), R_S=33Ω (recommended value), R_{PU}=1.5kΩ (recommended value)

13.4 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.

13.4.1 UDC2AB Register

13.4.1.1 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.

13.4.1.2 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.

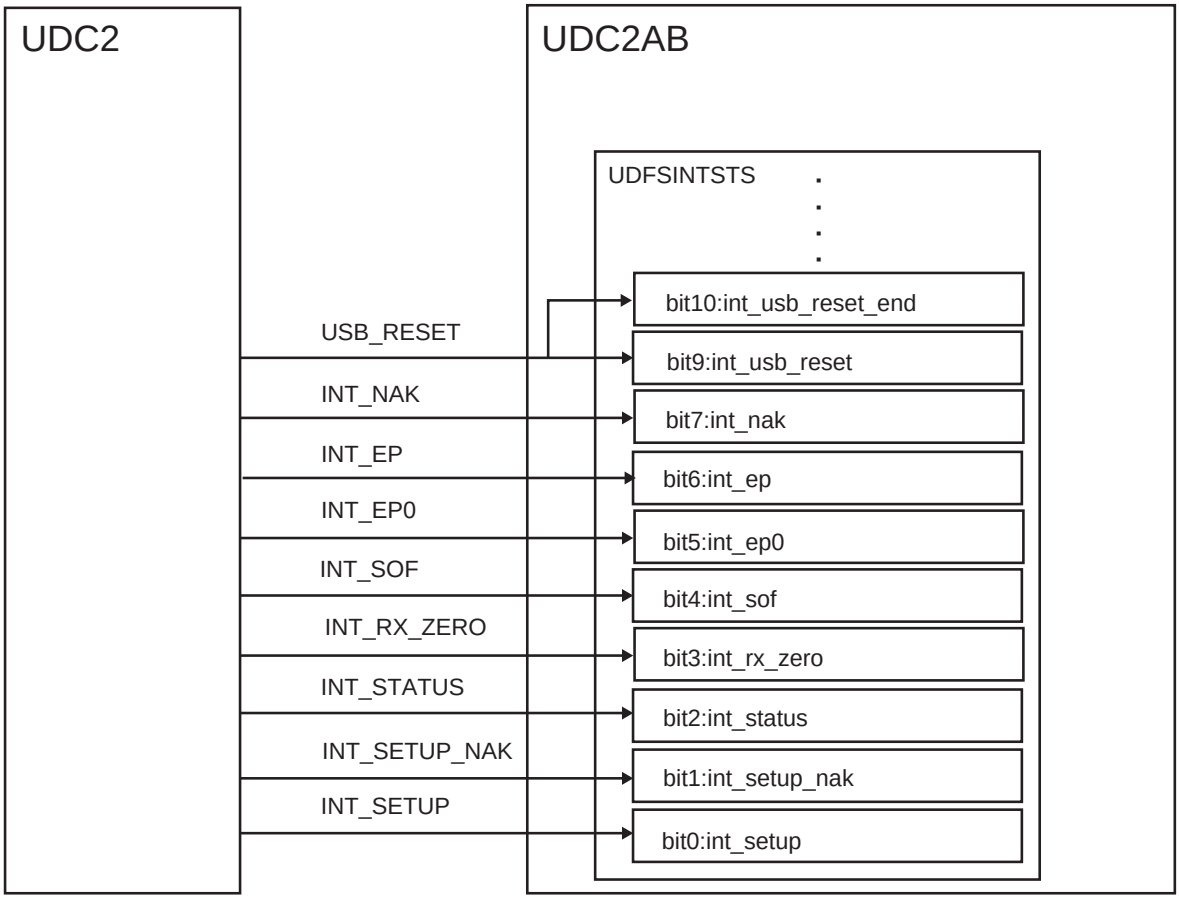


Figure 13-7 Connection between the flag output signals and interrupt bits.

13.4.1.3 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.

13.4.1.4 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

13.4.1.5 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"13.5.4.1 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.

13.4.1.6 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

13.4.1.7 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.

13.4.1.8 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.

13.4.1.9 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 "13.4.1.1 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.

13.4.1.10 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.

13.4.1.11 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.

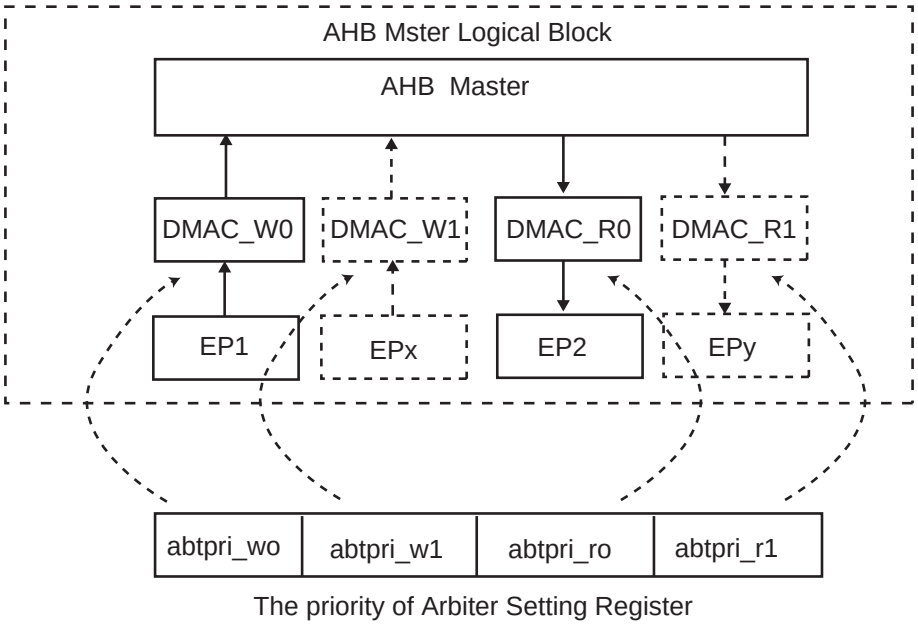


Figure 13-8 Relationship between DMAC and priority areas

13.4.1.12 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.

13.4.1.13 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.

13.4.1.14 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.

13.4.1.15 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.

13.4.1.16 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.

13.4.1.17 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.

13.4.1.18 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.

13.4.1.19 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.

13.4.1.20 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 TMPM365FYXBG 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 TMPM365FYXBG to resuming from the low-power consumption mode using INTUSBWKUP. Refer to "13.5.7 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 "13.5.7 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 "13.5.7 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 "13.5.1 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

13.4.1.21 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.

13.4.1.22 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.

13.4.2 UDC2 Register

13.4.2.1 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.

13.4.2.2 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 "13.4.2.1 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.

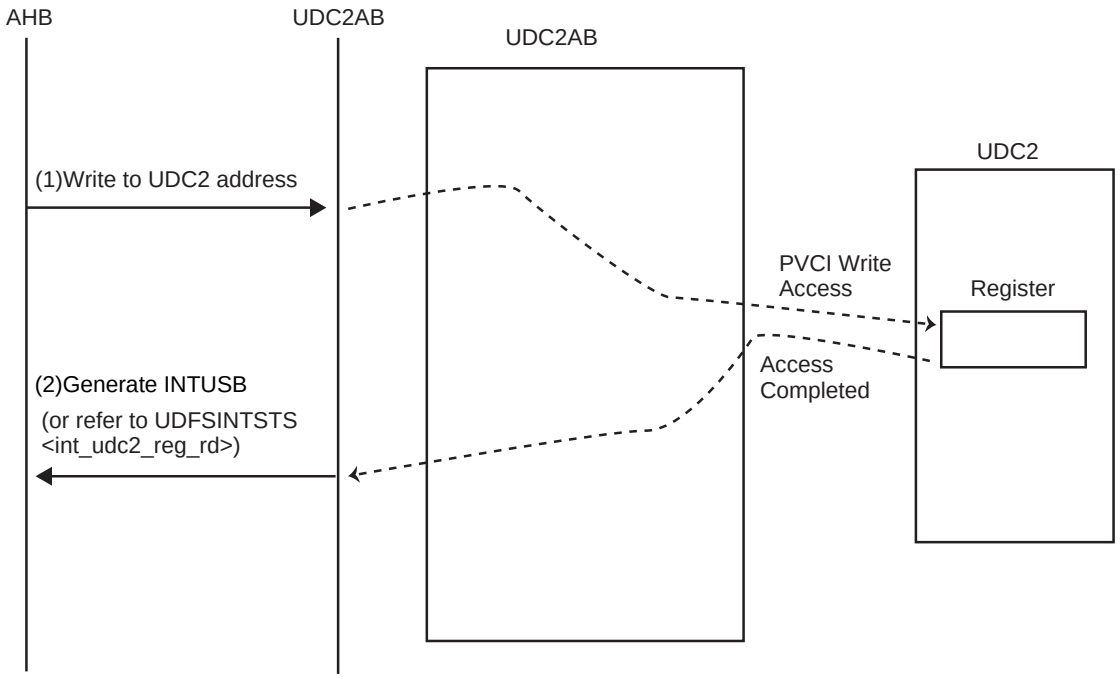


Figure 13-9 Write Access Flow Diagram of the UDC2 Register

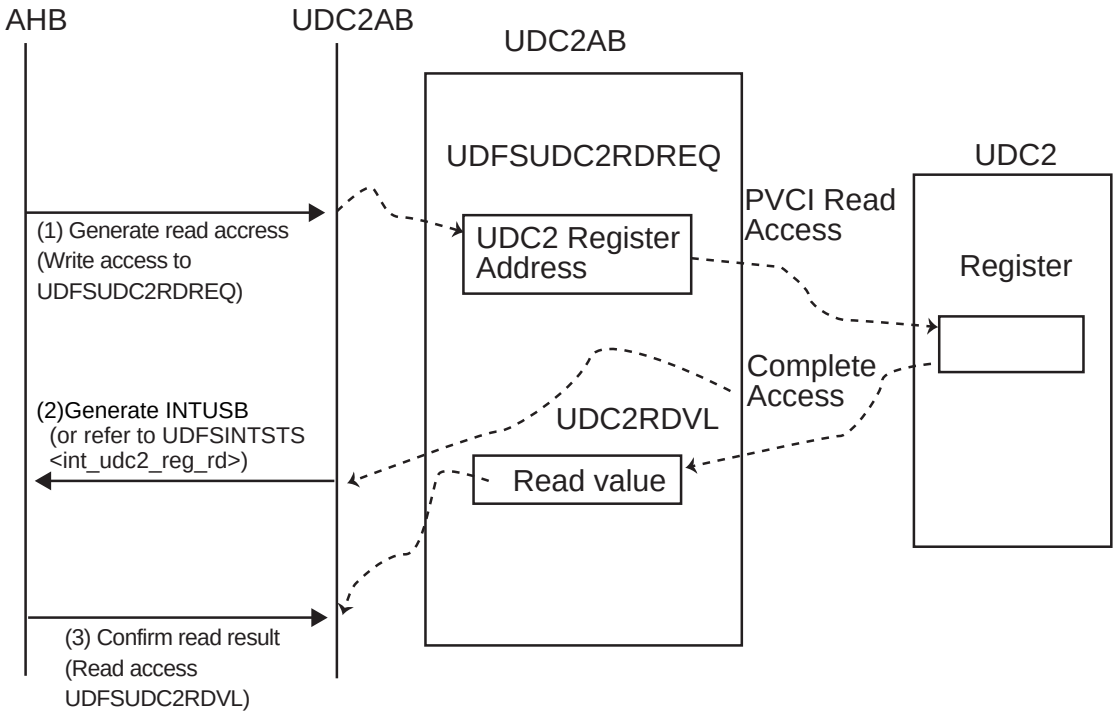


Figure 13-10 Read Access Flow Diagram of the UDC2 Register

13.4.2.3 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).

13.4.2.4 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.

13.4.2.5 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 "13.2.2.3 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

13.4.2.6 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

13.4.2.7 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).

13.4.2.8 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).

13.4.2.9 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).

13.4.2.10 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.

13.4.2.11 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.

13.4.2.12 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".

13.4.2.13 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.

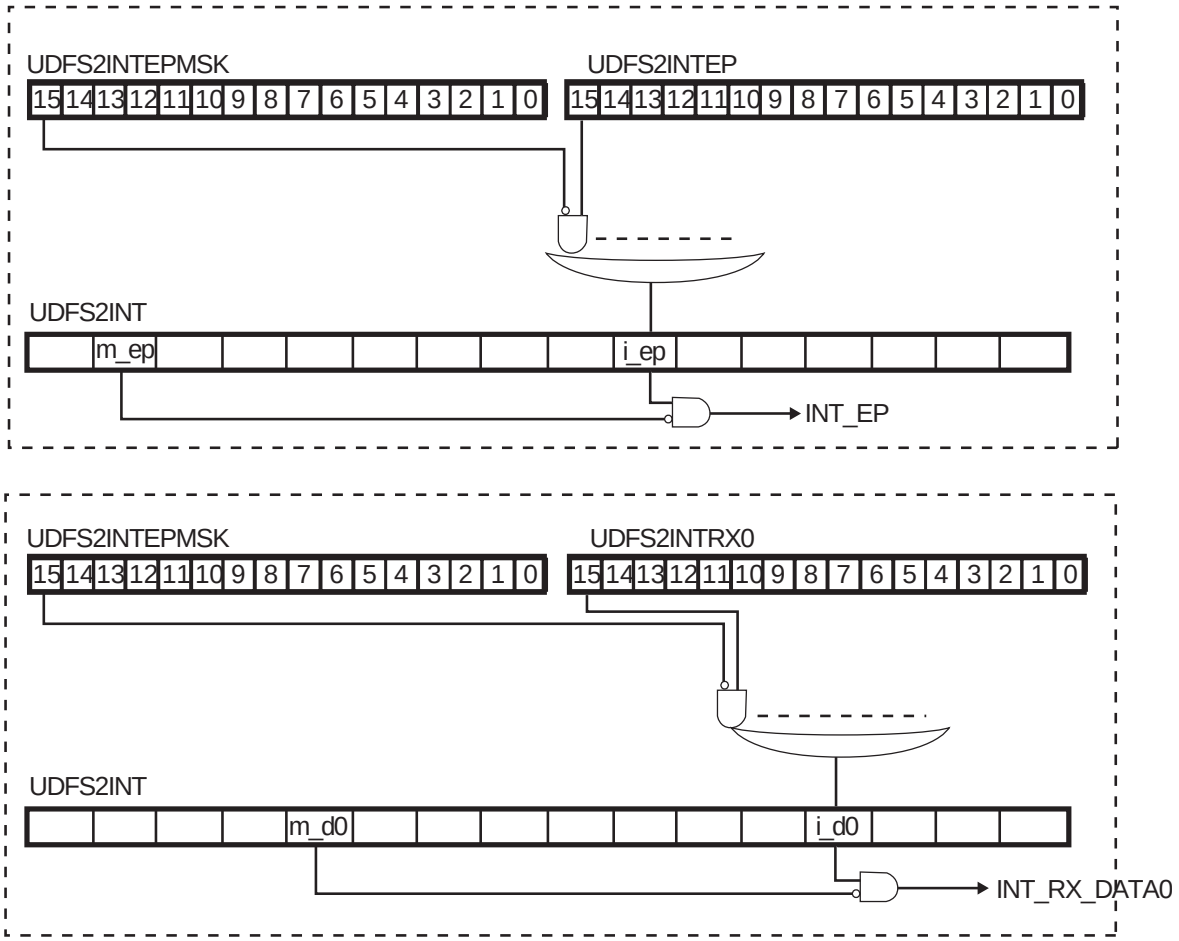


Figure 13-11 Interrupt Status and Mask Register

13.4.2.14 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.

13.4.2.15 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.

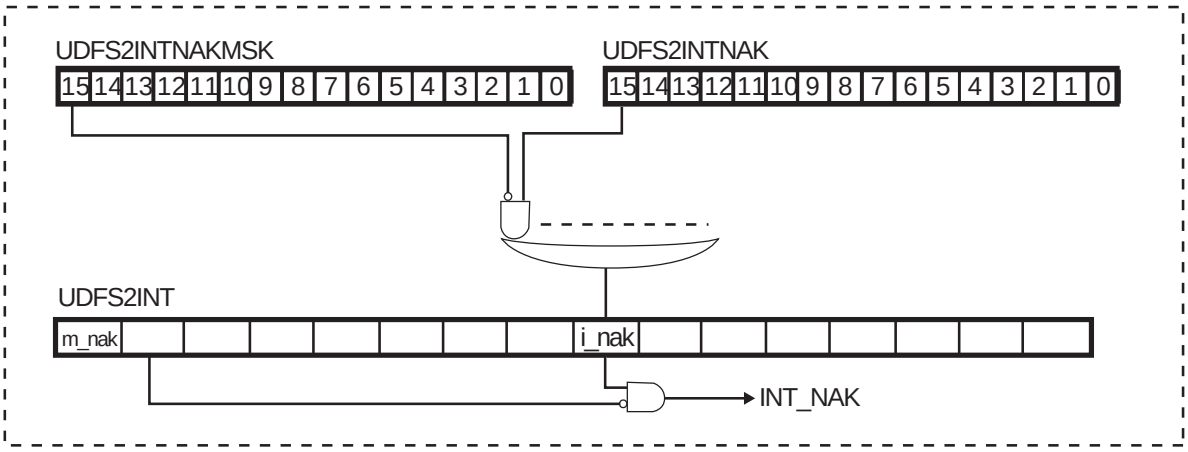


Figure 13-12 Interrupt and Status Register

13.4.2.16 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.

13.4.2.17 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.

13.4.2.18 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.

13.4.2.19 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 "13.7.1.1 Control-RD transfer" , "13.7.1.2 Control-WR transfer (without DATA-Stage)" and "13.7.1.3 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).

13.4.2.20 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 "13.9.2 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

13.4.2.21 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.

13.4.2.22 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

13.4.2.23 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

13.5 Description of UDC2AB operation

13.5.1 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 "13.4.1.1 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 "13.4.1.6 UDFSMSTSET(DMAC Setting Register)".

13.5.2 Interrupt Signals

There are two interrupt output signals of UDC2AB, INTUSB and INTUSBWKUP.

13.5.2.1 INTUSB Interrupt Signal

Interrupt output signal of INTUSB consists of interrupts generated by UDC2 and that generated by other sources.

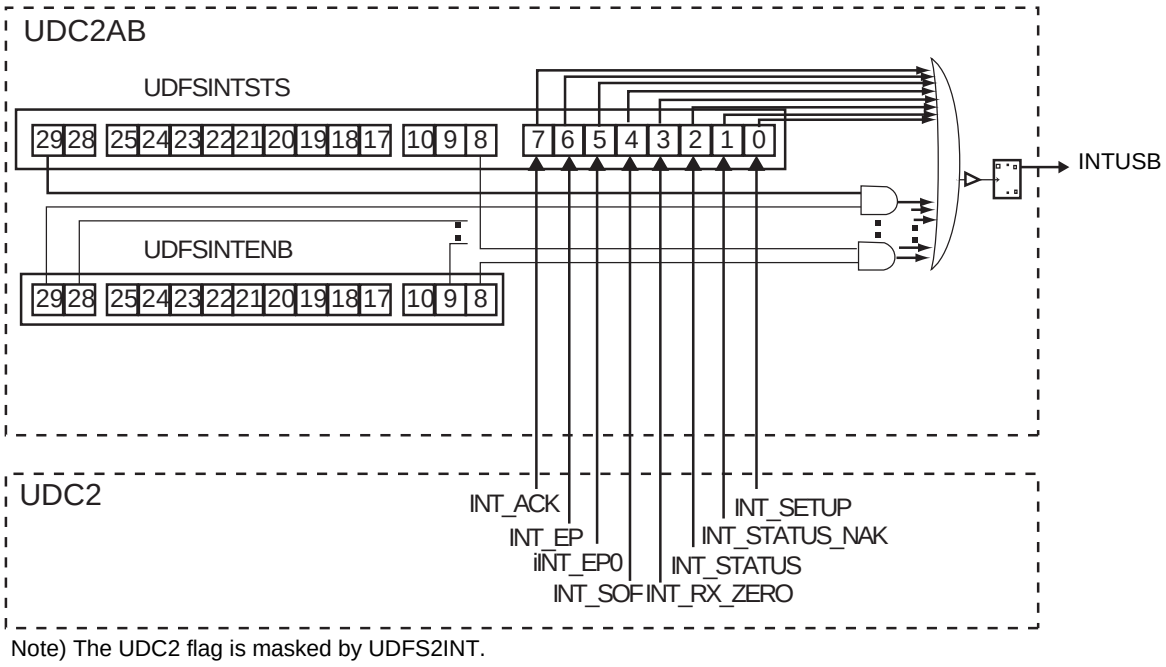
Once the interrupt condition is met, UDC2AB sets the corresponding bit of its UDFSINTSTS. When that bit is set, INTUSB will be asserted if the relevant bit of UDFSINTENB has been set to "Enable."

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 13-13 Relationship of INTUSB and registers

13.5.2.2 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.

13.5.3 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 "13.5.5.2 Sequence of USB Bus Power (VBUS) Connection/Disconnection" and "13.5.1 Reset" for details.
4. USB enumeration response

   Refer to "13.6 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 "13.5.4.1 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 "13.5.4.2 Master Write transfer" for details.
6. VBUS detection (disconnect)

   The USB bus power supply may be disconnected at any time.

   Refer to "13.5.5.2 Sequence of USB Bus Power (VBUS) Connection/Disconnection" for details.

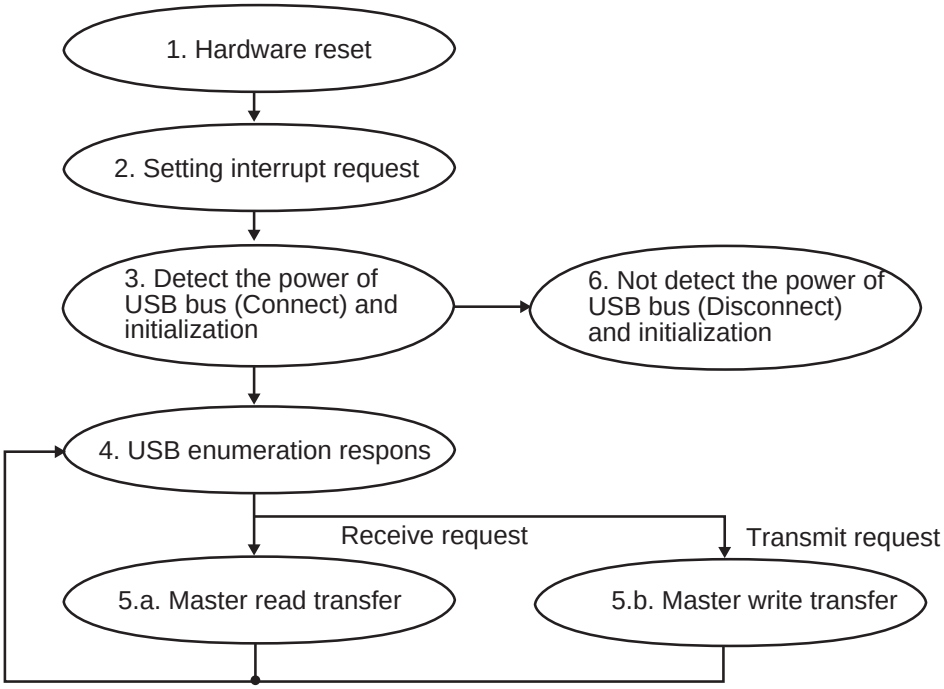


Figure 13-14 Operation Sequence

13.5.4 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."

13.5.4.1 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 "13.9.2 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.

13.5.4.2 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 13-15). 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.

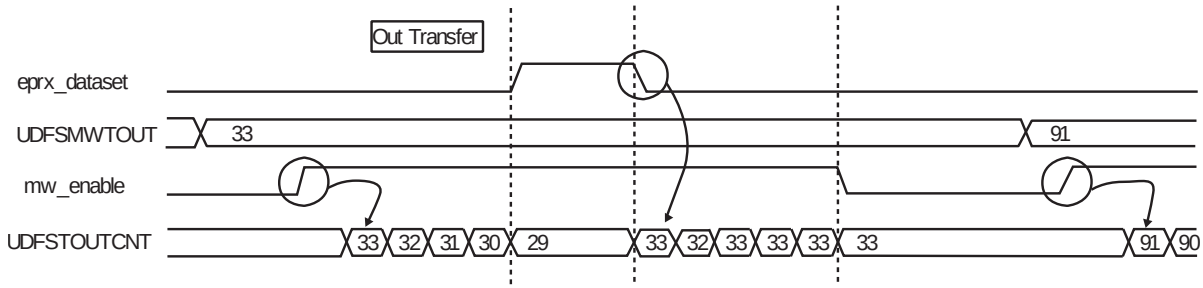


Figure 13-15 Example of MW timeout count

(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).

13.5.5 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.

13.5.5.1 Connection Diagram of Power Management Control Signal

Below is a connection diagram of signals related to power management control.

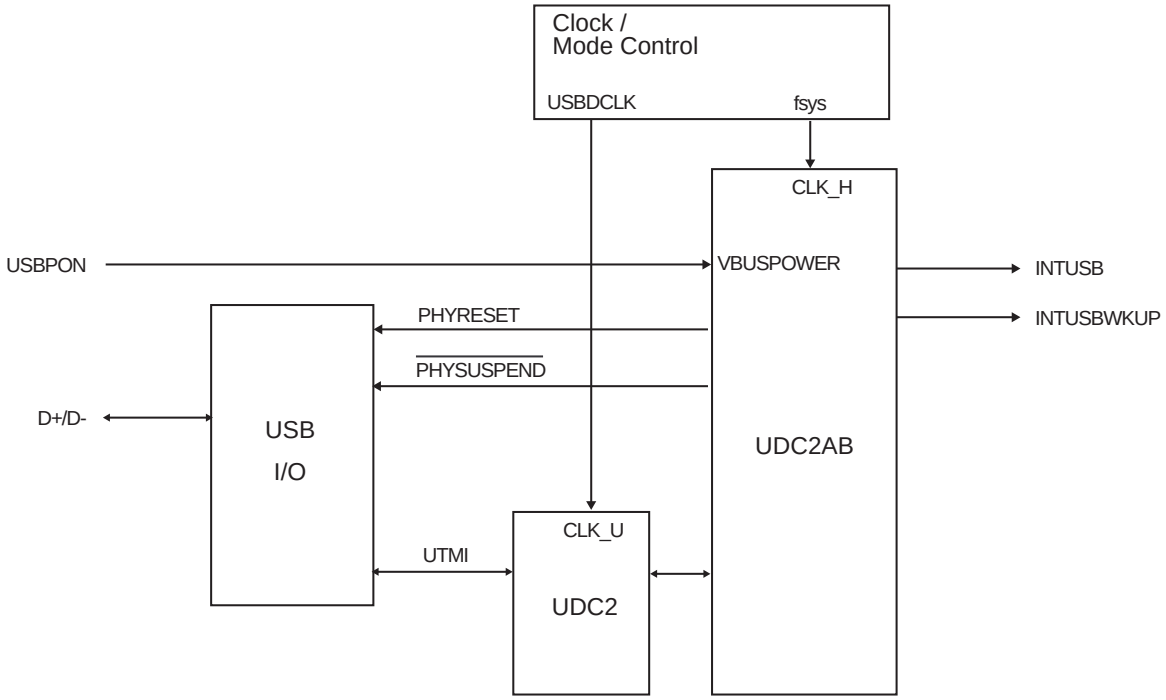


Figure 13-16 Connection Diagram of Power Management Control Signal

13.5.5.2 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.

13.5.6 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.

13.5.7 Suspend / Resume

13.5.7.1 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 $\overline{\text{PHYSUSPEND}}$ which will put PHY in suspended state.

13.5.7.2 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.

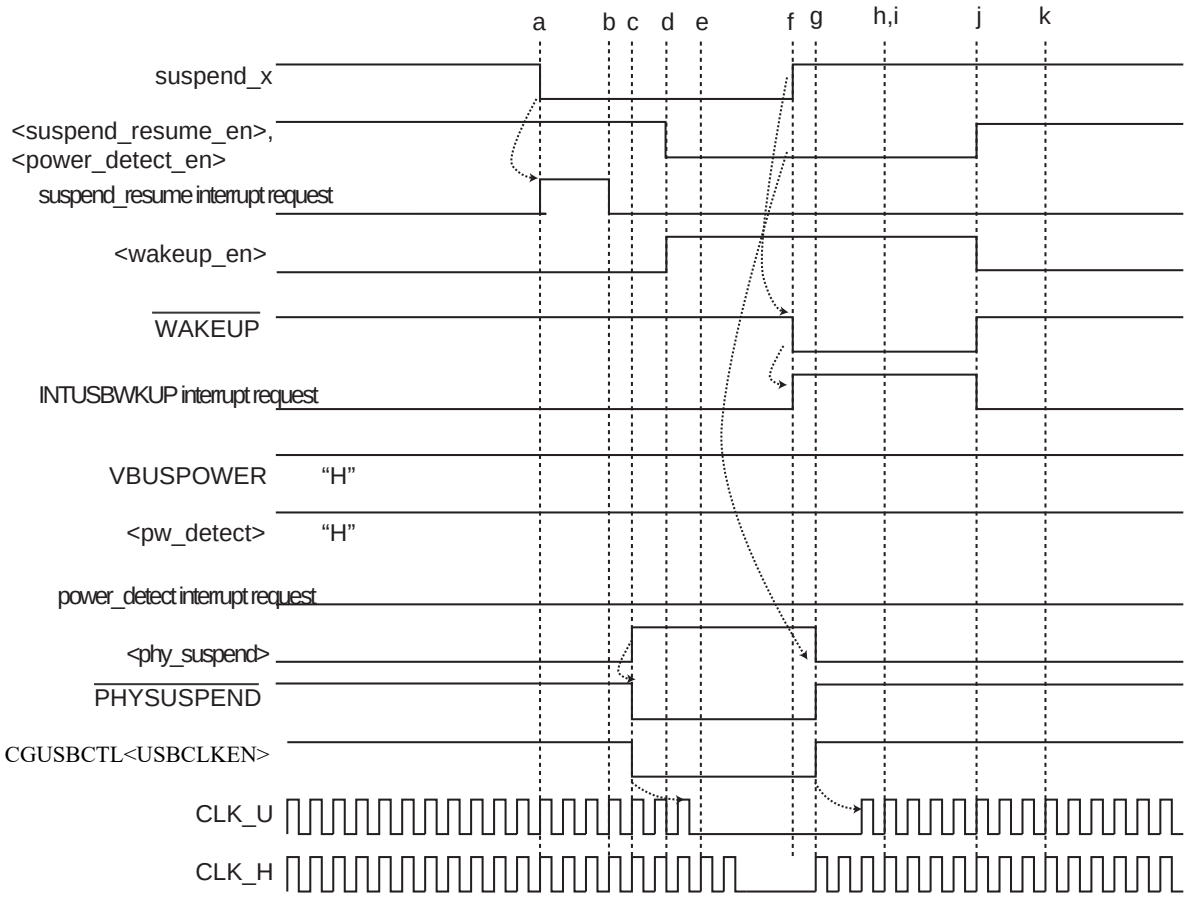


Figure 13-17 Signal operations when suspended and resumed (when CLK_H is stopped)

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 "13.5.7.3 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.

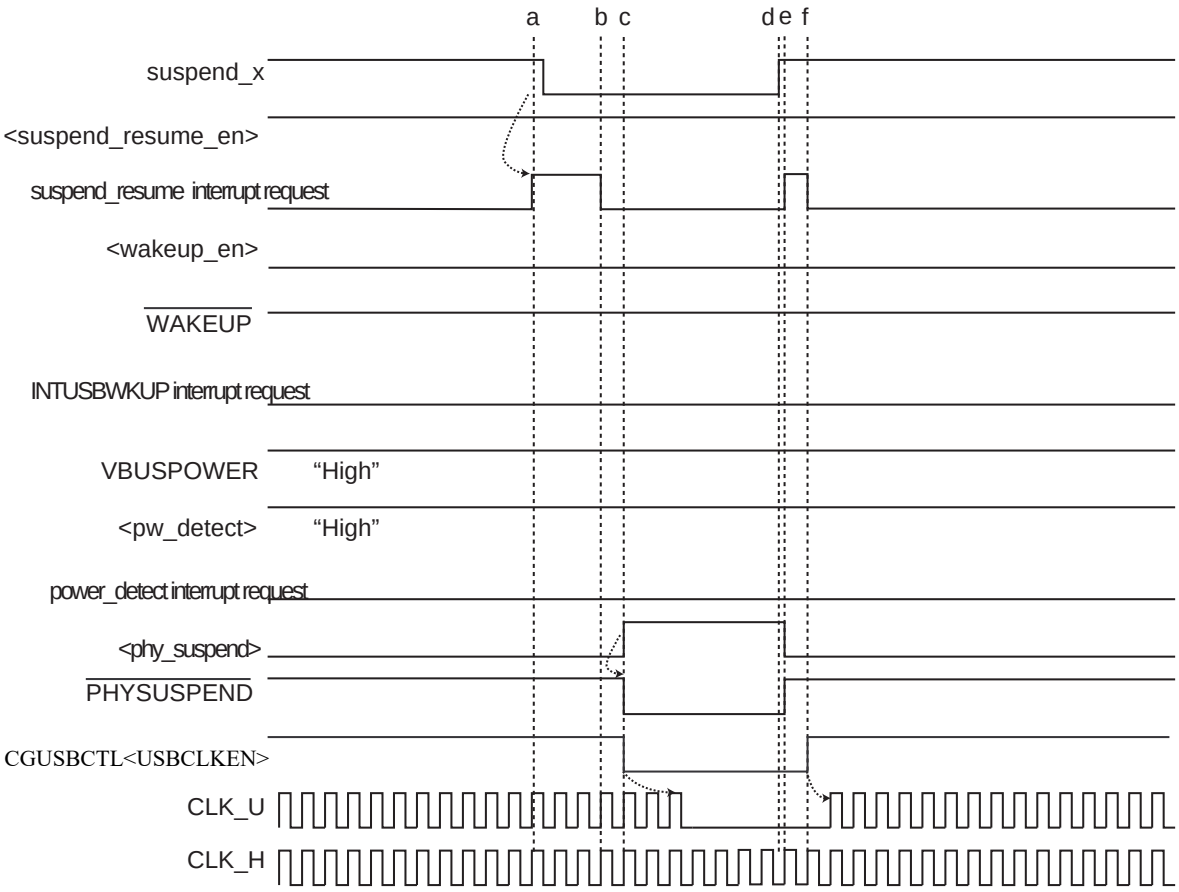


Figure 13-18 Operation of suspend/resume signals (to get the CLK_H work)

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.

13.5.7.3 Resuming from the suspend state (disconnect)

The procedures to resume from the suspended state (disconnection) and the signal change are shown as below.

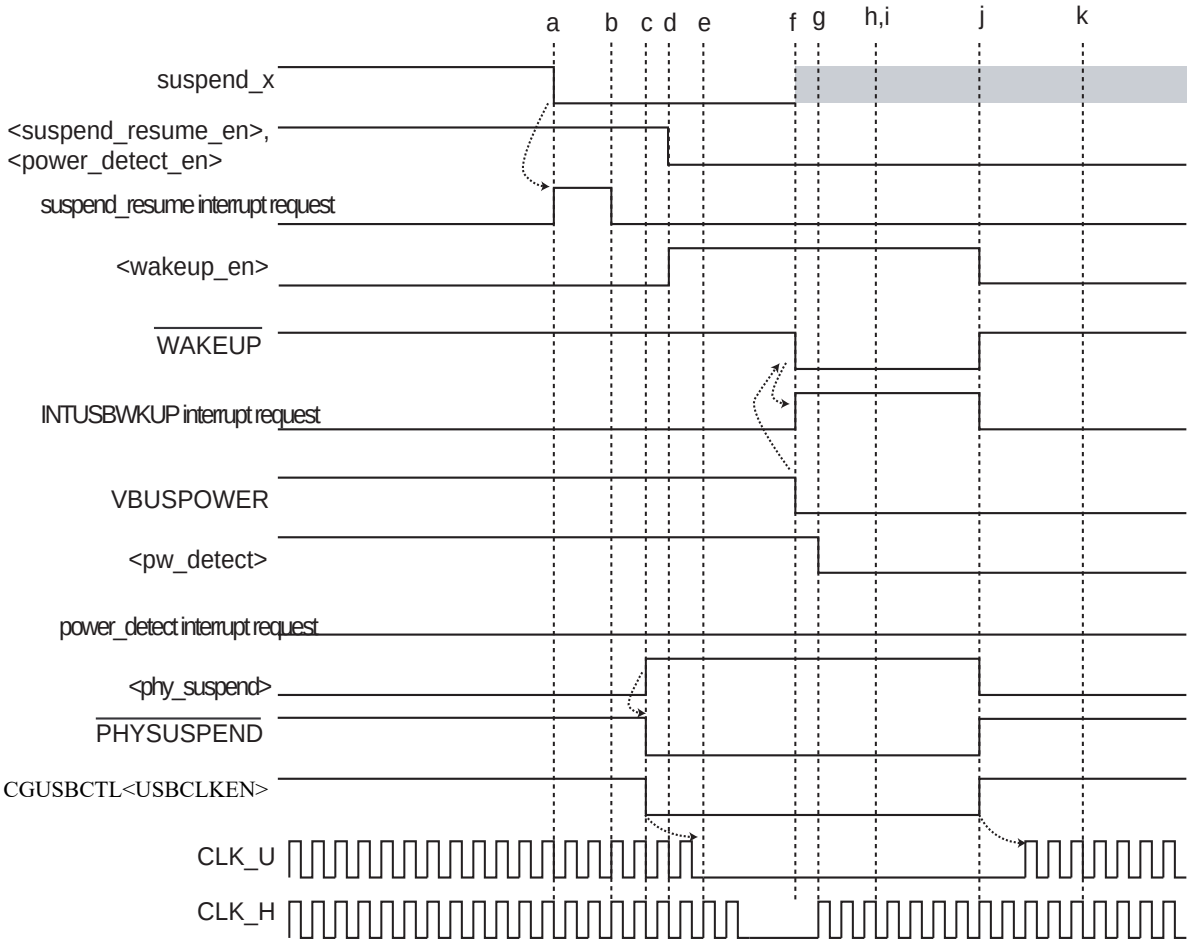


Figure 13-19 Operation of suspend/disconnect signals (to stop CLK_H)

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 "13.5.7.2 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.

13.5.7.4 Remote wakeup from the suspended state

The procedure of remote wakeup from the suspended state and the signal change are shown below.

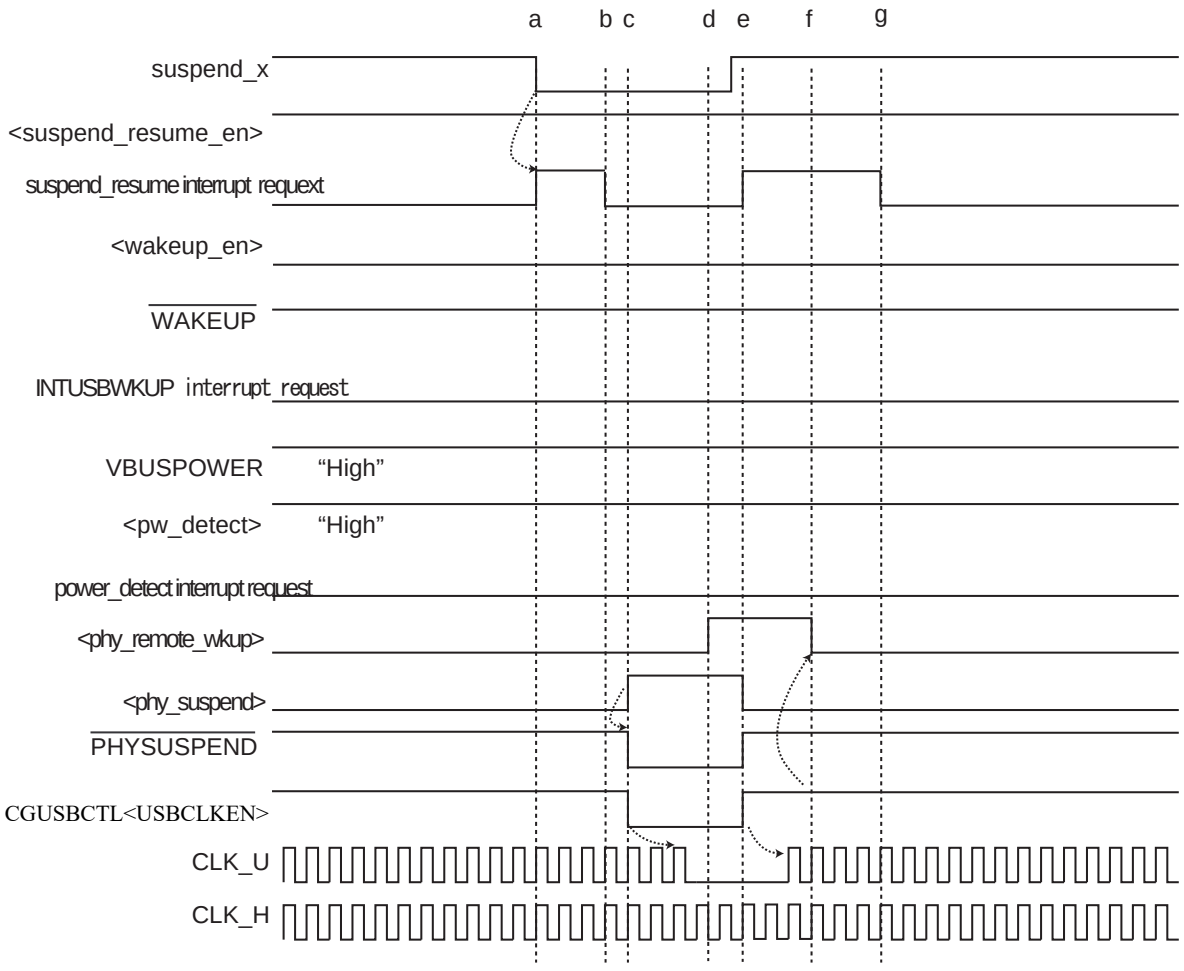


Figure 13-20 Operation of suspend/remote wakeup signals

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.

13.6 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 13-21 shows the "Device State Diagram".

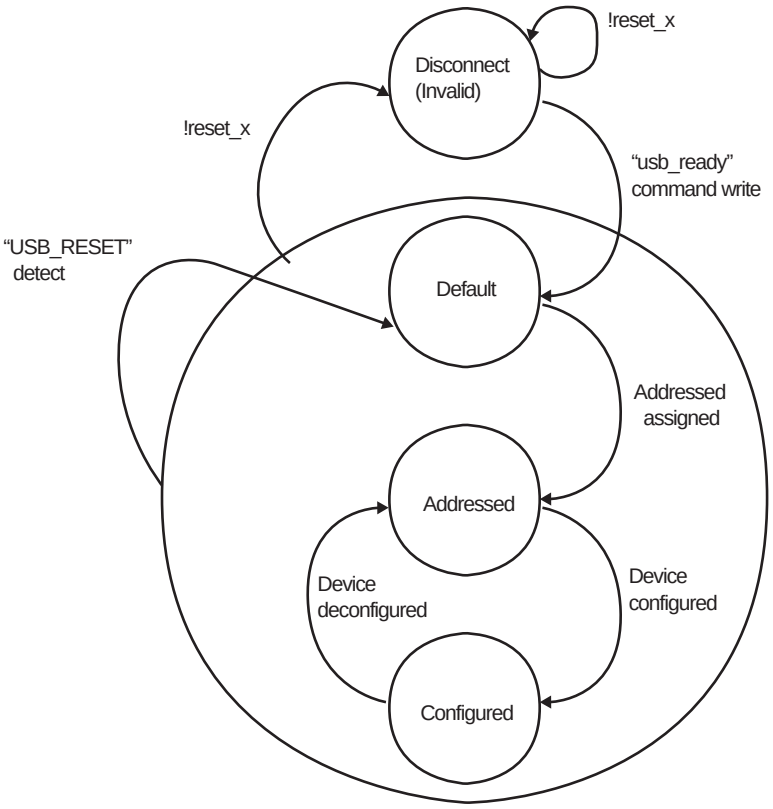


Figure 13-21 Device state diagram

13.7 Flow of Control in Transfer of EPs

13.7.1 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.

13.7.1.1 Control-RD transfer

The flow of control in Control-RD transfers is shown below.

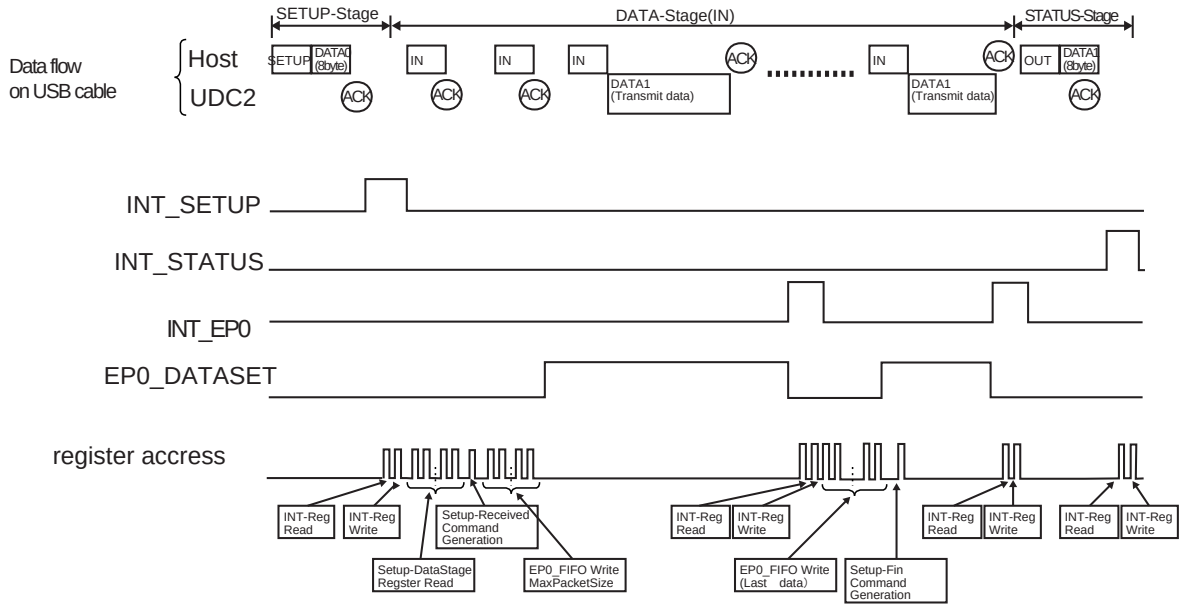


Figure 13-22 Flow of the control in Control-RD transfer

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.

13.7.1.2 Control-WR transfer (without DATA-Stage)

The flow of control in Control-WR transfer (without DATA-Stage) is shown below.

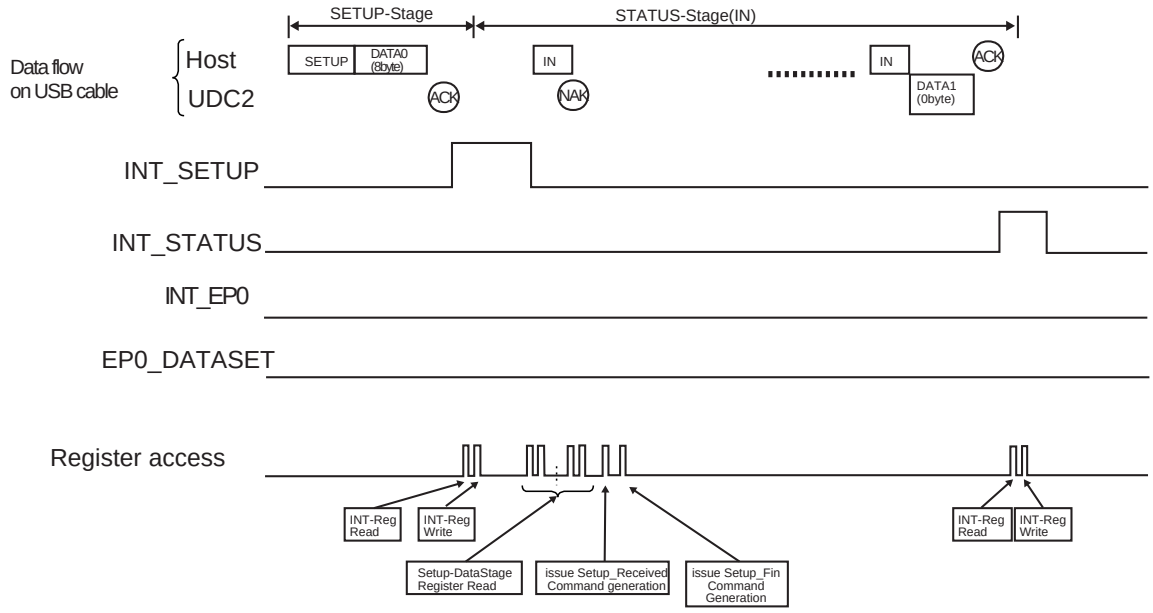


Figure 13-23 Flow of control in Control-WR transfer (without DATA-Stage)

(1) SETUP-Stage

Perform the same procedure described in "13.7.1.1 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 "13.7.1.1 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 "13.7.1.5 Processing when standard request".

13.7.1.3 Control-WR transfer (with DATA-Stage)

The flow of control in Control-WR transfer (with DATA-Stage) is shown below.

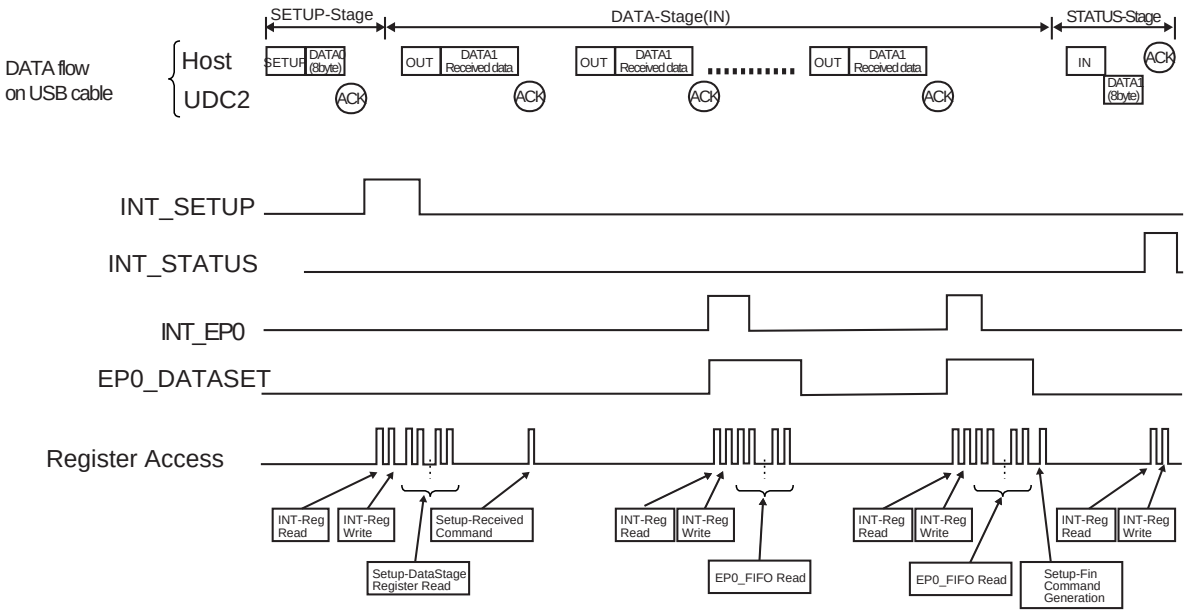


Figure 13-24 Flow of control in Control-WR transfers (with DATA-Stage)

(1) SETUP-Stage

To be processed in the same way of SETUP-Stage as described in"13.7.1.1 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 "13.7.1.1 Control-RD transfer".

13.7.1.4 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.)

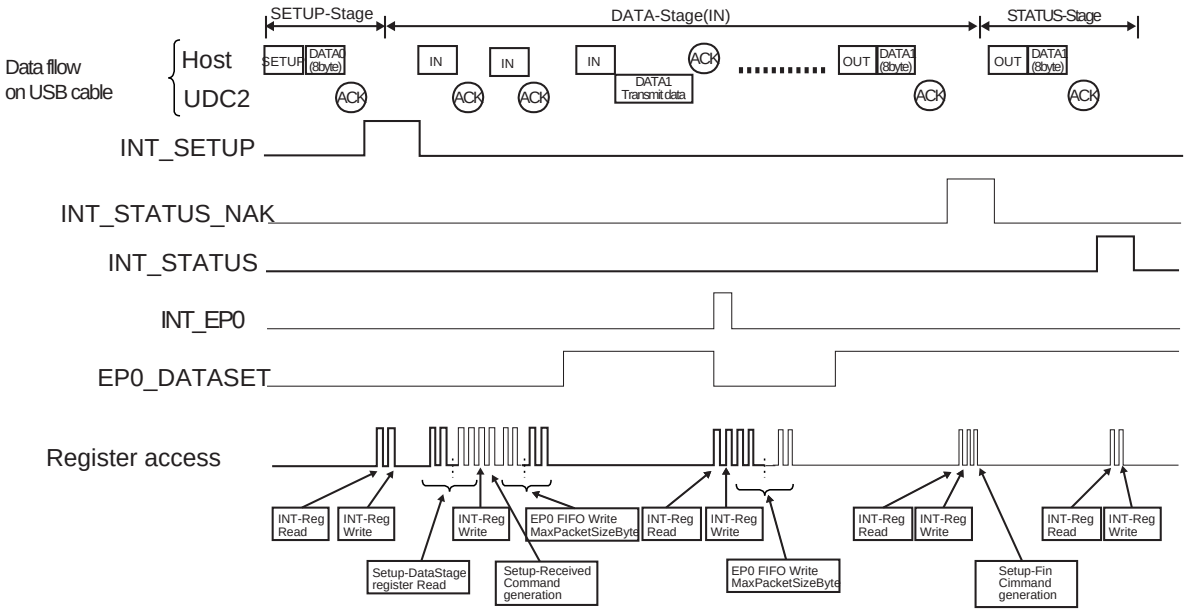


Figure 13-25 Example of using the INT_STATUS_NAK flag in Control-RD transfers

(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.

13.7.1.5 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 13.7.1.1 , 13.7.1.2 and 13.7.1.3.

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 "13.7.1.1 Control-RD transfer".

  Get Status Get Description Get Configuration

  Get Interface Get Frame
- Standard requests for "13.7.1.2 Control-WR transfer (without DATA-Stage)".

  Clear Feature Set Feature Set Address

  Set Configuration Set Interface
- Standard requests for "13.7.1.3 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 13-3) to UDFS2EP0FIFO.

<recipient> = Interface: Issue the EP-Stall command to EP0

<recipient> = EP : If wIndex=0(EP0), write the information on EP0 (Table 13-5) to UDFS2EP0FIFO. If wIndex≠0(EPx), issue the EP-Stall command to EP0.

- Configured state:

<recipient> = Device : Write the information on the device (Table 13-3) to UDFS2EP0FIFO.

<recipient> = Interface: If the interface specified by lwIndex, write the information on the interface (Table 13-4) to UDFS2EP0FIFO.

<recipient> = EP : If the EP specified by wIndex, write the information on the relevant EP(Table 13-5) to UDFS2EP0FIFO.

Table 13-3 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 13-4 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 13-5 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 "13.7.1.1 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.

13.7.2 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.

13.8 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.

13.8.1 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.

13.8.2 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.

13.8.2.1 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.

13.8.2.2 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".

13.9 USB-Spec2.0 Device Controller Appendix

13.9.1 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.

13.9.1.1 Connect / Disconnect Operations

(1) Connect Operation

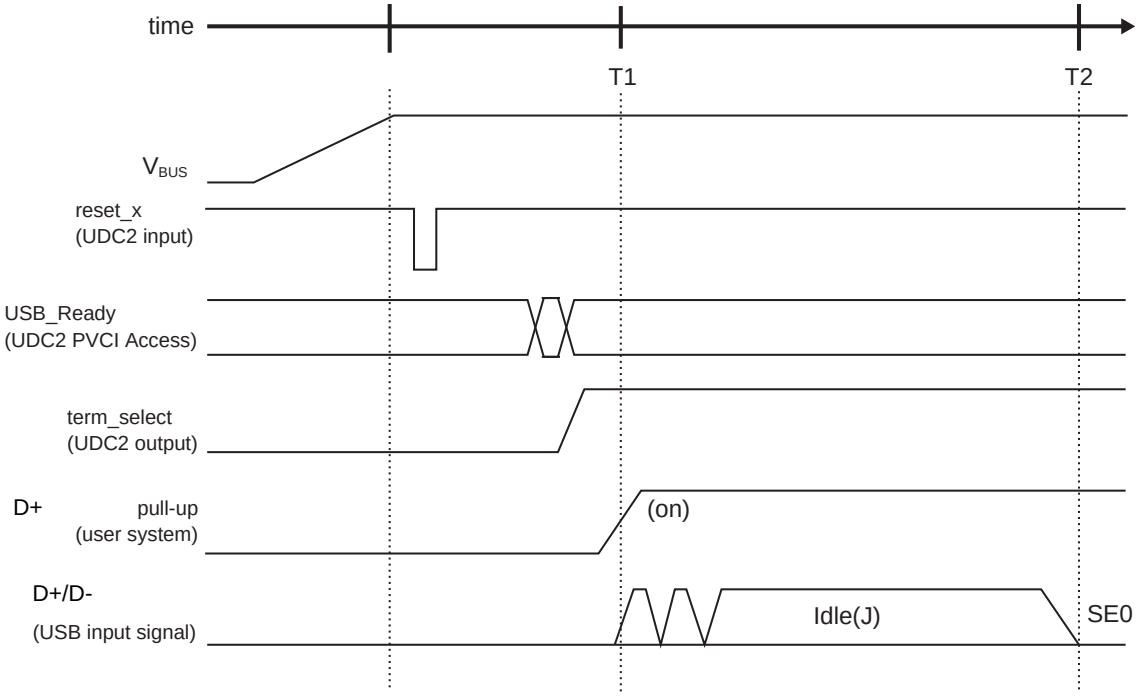


Figure 13-26 Connect operation timing

- 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.

13.9.1.2 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

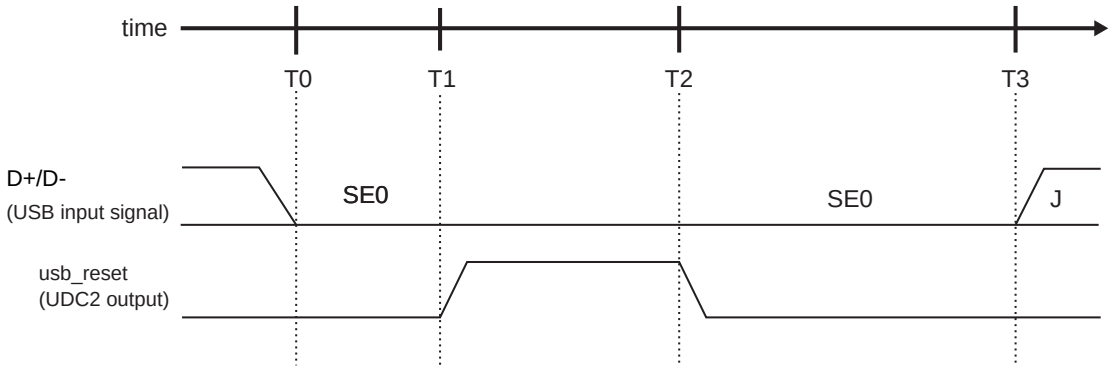


Figure 13-27 Reset Operation Timing

- 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 "13.4 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.

13.9.1.3 Suspend Operation

(1) Suspend operation

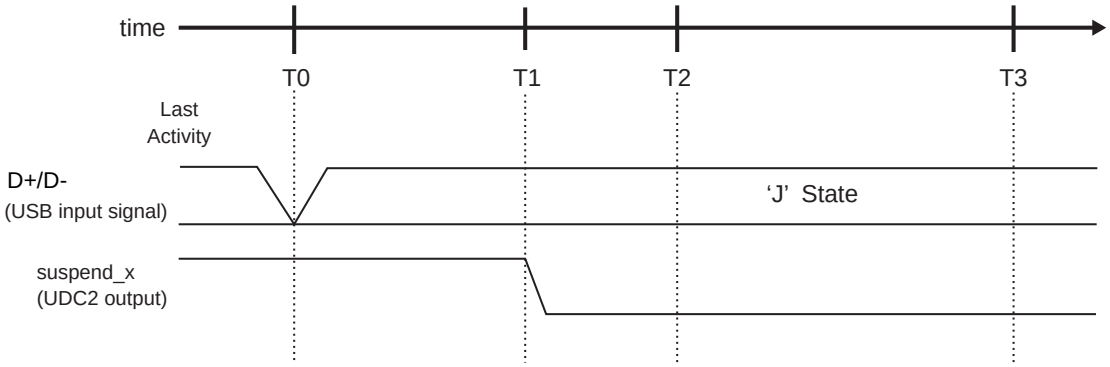


Figure 13-28 Suspend operation timing

- 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.

13.9.1.4 Resume Operation

(1) Resume Operation by the Host

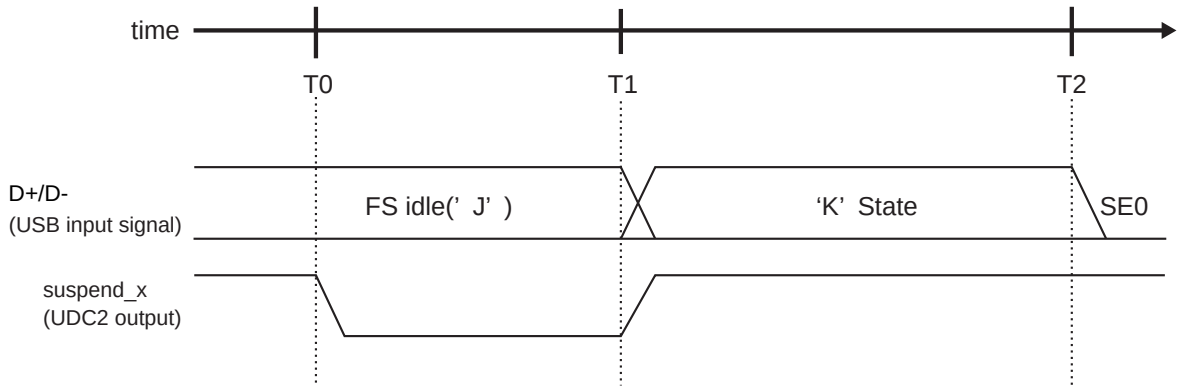


Figure 13-29 Resume operation timing by the host

- 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)

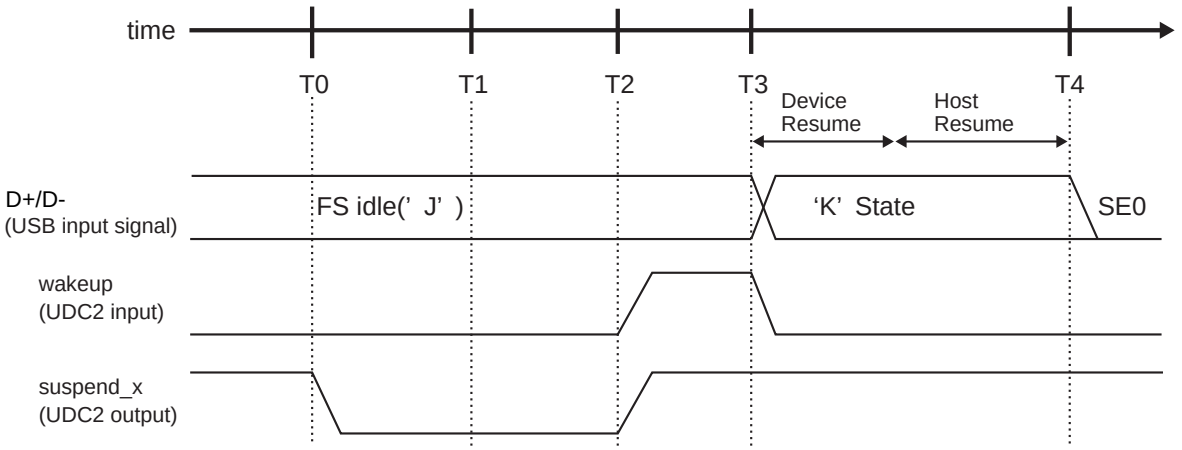


Figure 13-30 Remote wakeup operation timing

- 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 13.8 of the USB 2.0 Specification which offers detailed description.

13.9.2 Appendix B About Setting an Odd Number of Bytes as MaxPacketSize

13.9.2.1 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 13-6 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 13-6 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.

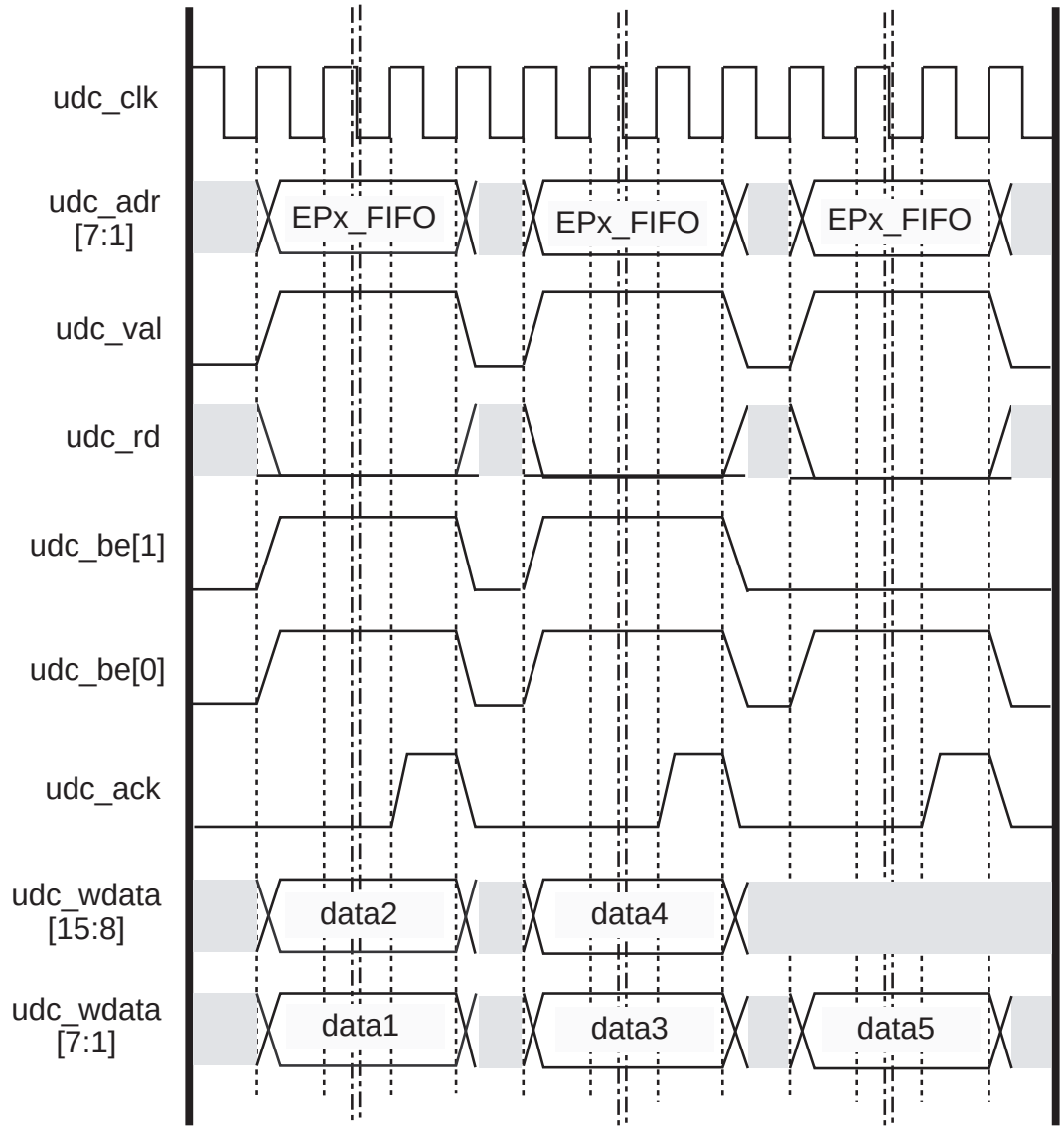


Figure 13-31 MPS write access with max_pkt = odd number (common bus access)

(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.

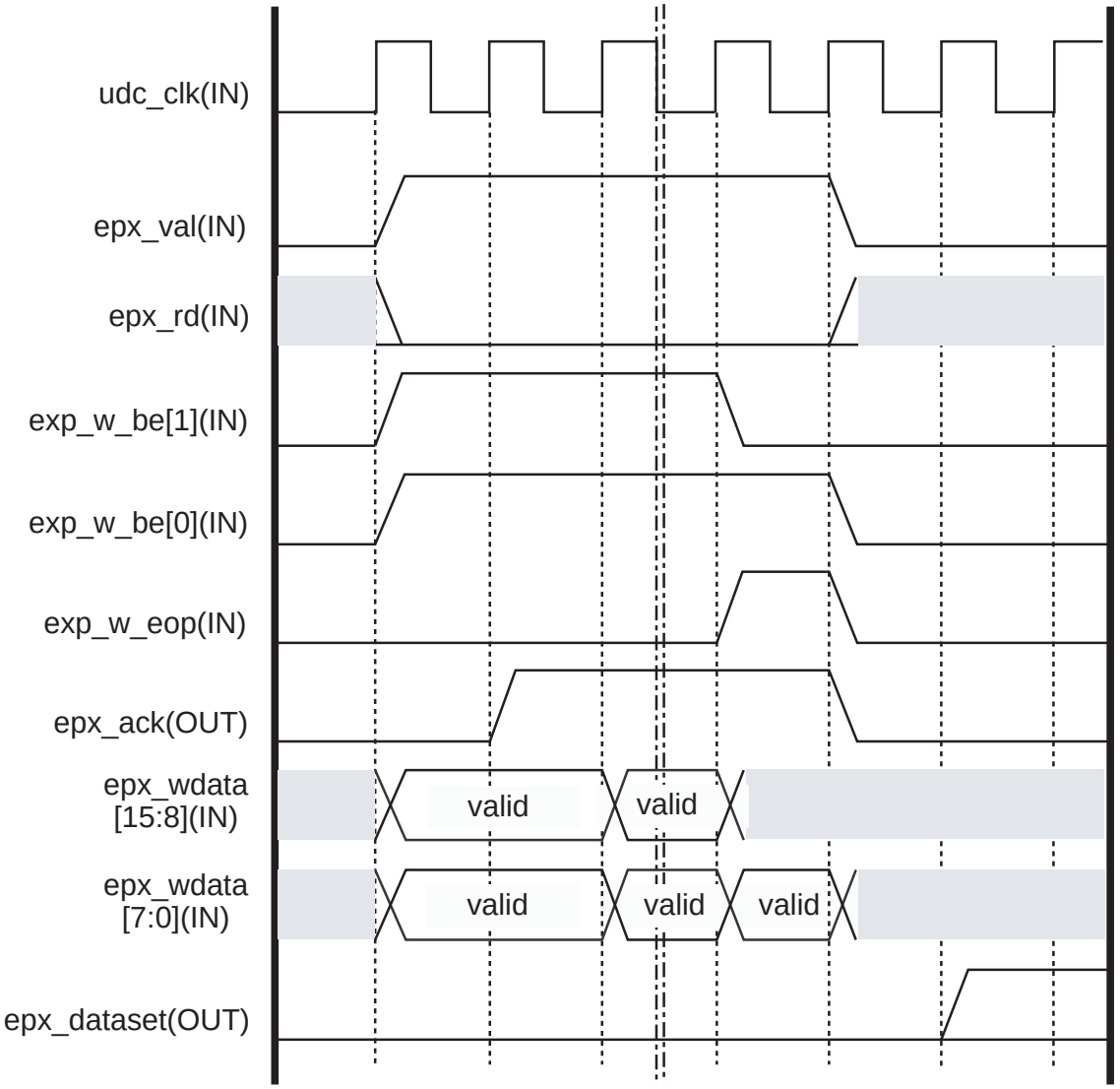


Figure 13-32 MPS (odd number) write access with max_pkt = even number (EP direct access)

13.9.3 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.

13.9.3.1 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.)

13.9.3.2 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).

14. Serial Channel (SIO/UART)

14.1 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.

14.2 Difference in the Specifications of SIO Modules

TMPM365FYXBG 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 14-1 Difference in the Specifications of SIO Modules

	Pin name			Interrupt		DMA request	UART source clock
	TXD	RXD	$\overline{\text{CTS}}$x/ SCLKx	Receive Interrupt	Transmit Interrupt		
Channel 0	PE0	PE1	PE2	INTRX0	INTTX0	Support	TB8OUT
Channel 1	PC0	PC1	PC2	INTRX1	INTTX1	Support	TB8OUT

14.3 Configuration

Figure 14-1 shows SIO block diagram.

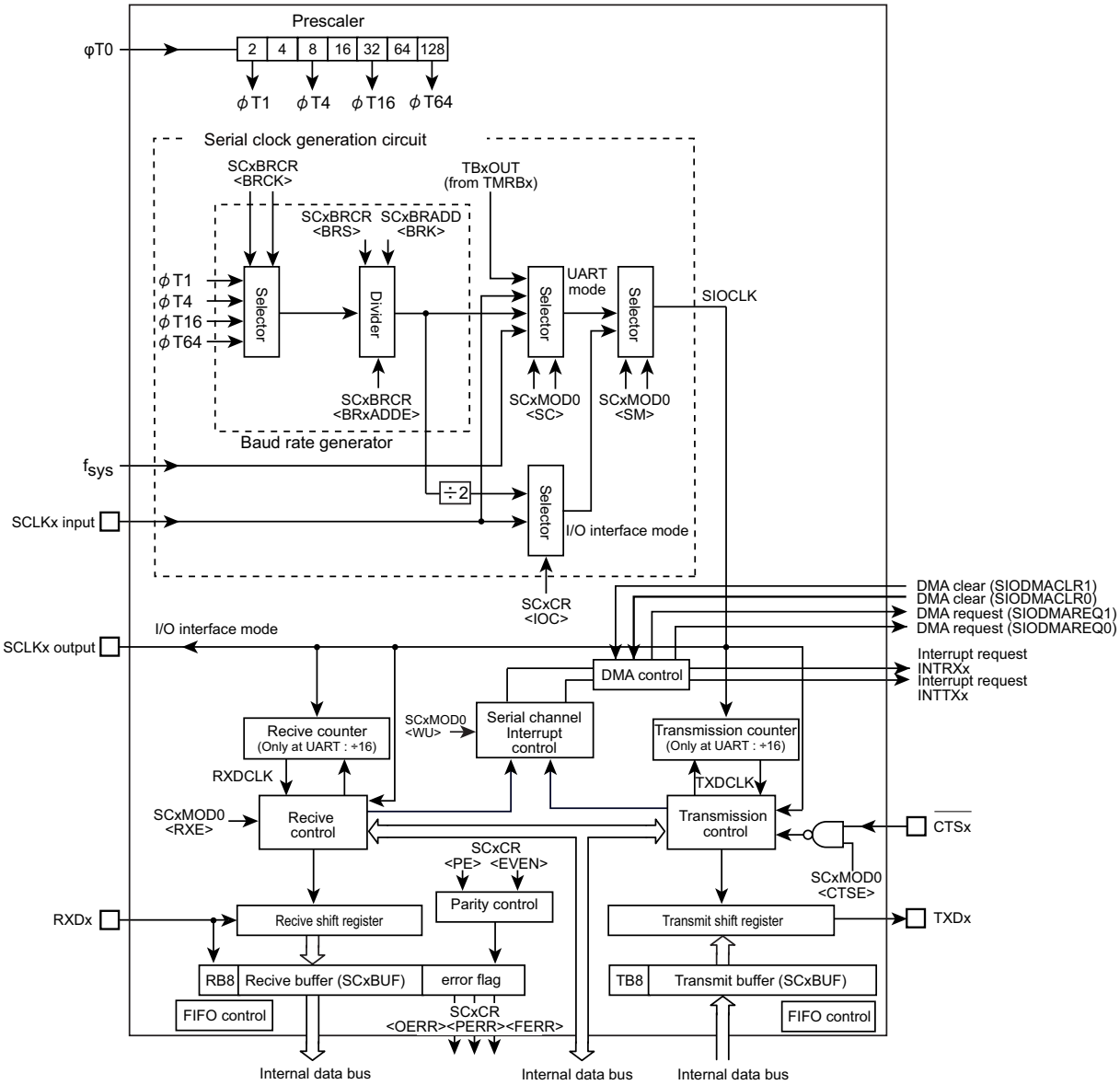


Figure 14-1 SIO Block Diagram

14.4 Registers Description

14.4.1 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.

14.4.2 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.**

14.4.3 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

14.4.4 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.**

14.4.5 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 $\overline{CTS}$ 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.

14.4.6 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.**

14.4.7 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

<table>
<tr><th>Bit</th><th>Bit Symbol</th><th>Type</th><th>Function</th></tr>
<tr><td>2</td><td>WBUF</td><td>R/W</td><td>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.</td></tr>
<tr><td>1-0</td><td>SWRST[1:0]</td><td>R/W</td><td>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).
<table>
<tr><th>Register</th><th>Bit</th></tr>
<tr><td>SCxMOD0</td><td><RXE></td></tr>
<tr><td>SCxMOD1</td><td><TXE></td></tr>
<tr><td>SCxMOD2</td><td><TBEMP>, <RBFLL>, <TXRUN></td></tr>
<tr><td>SCxCR</td><td><OERR>, <PERR>, <FERR></td></tr>
<tr><td>SCxDMA (note2)</td><td><DMAEN1>, <DMAEN0></td></tr>
</table></td></tr>
</table>

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.

14.4.8 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 14-2 lists the settings of baud rate generator division ratio.

Table 14-2 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.**

14.4.9 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.**

14.4.10 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").

14.4.11 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.

14.4.12 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.**

14.4.13 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.**

14.4.14 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.**

14.5 Operation in Each Mode

Table 14-3 shows the modes and data formats.

Table 14-3 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.

14.6 Data Format

14.6.1 Data Format List

Figure 14-2 shows data format.

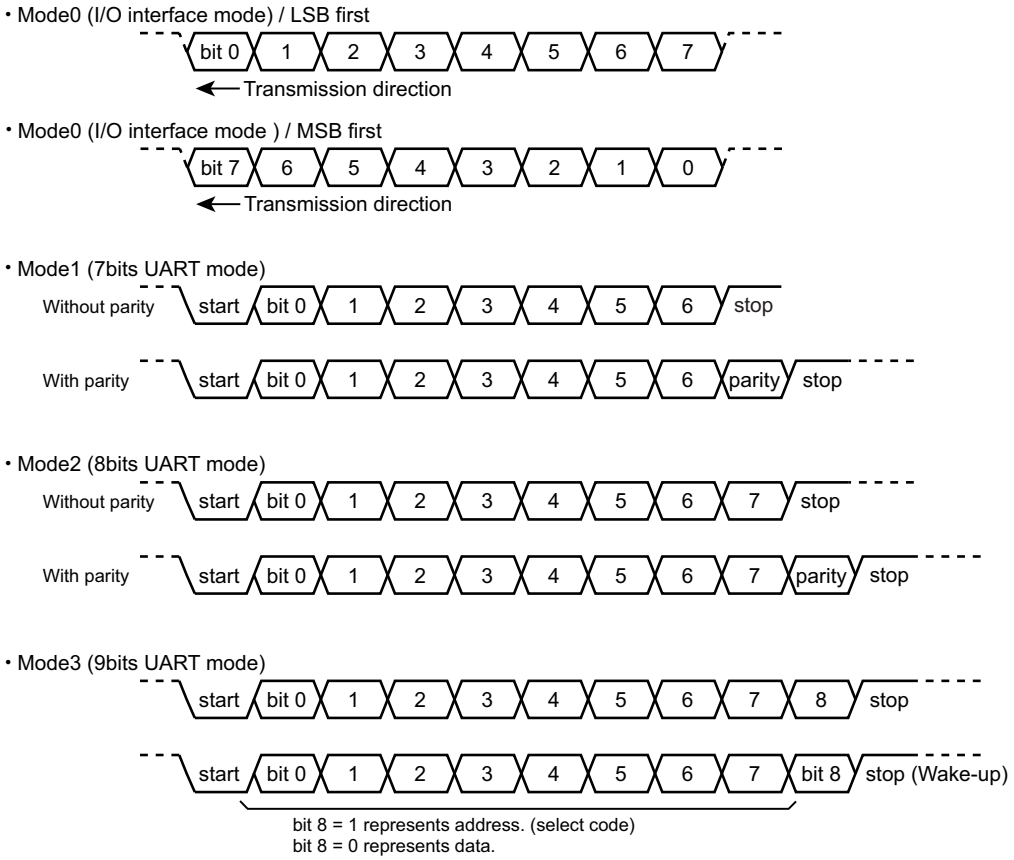


Figure 14-2 Data Format

14.6.2 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.

14.6.2.1 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.

14.6.2.2 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.

14.6.3 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.

14.7 Clock Control

14.7.1 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 14-4 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 14-4 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 14-5 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} (342 μs)	fc/2^{16} (1366 μs)

Table 14-5 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.**

14.7.2 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.

14.7.2.1 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.

14.7.2.2 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 14-6 shows clock selection in I/O interface mode.

Table 14-6 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 14-7 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 14-7 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 14-8 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 14-8 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 14-9 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 14-9 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

14.8 Transmit/Receive Buffer and FIFO

14.8.1 Configuration

Figure 14-3 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.

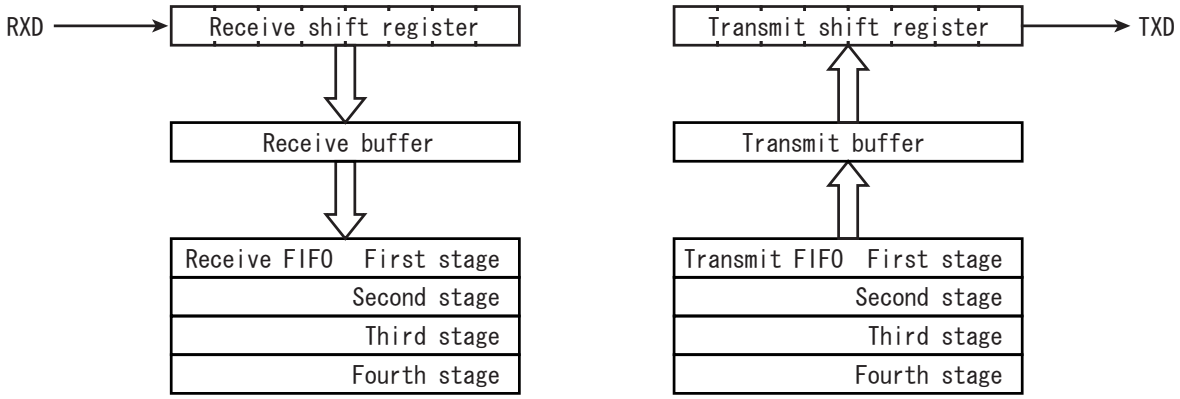


Figure 14-3 The Configuration of Buffer and FIFO

14.8.2 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 14-10 shows correlation between modes and buffers.

Table 14-10 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

14.8.3 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 14-11 shows correlation between modes and FIFO.

Table 14-11 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

14.9 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".

14.10 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

14.10.1 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.

14.10.2 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.

14.10.3 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.

14.11 Receive

14.11.1 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.

14.11.2 Receive Control Unit

14.11.2.1 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.

14.11.2.2 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.

14.11.3 Receive Operation

14.11.3.1 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.

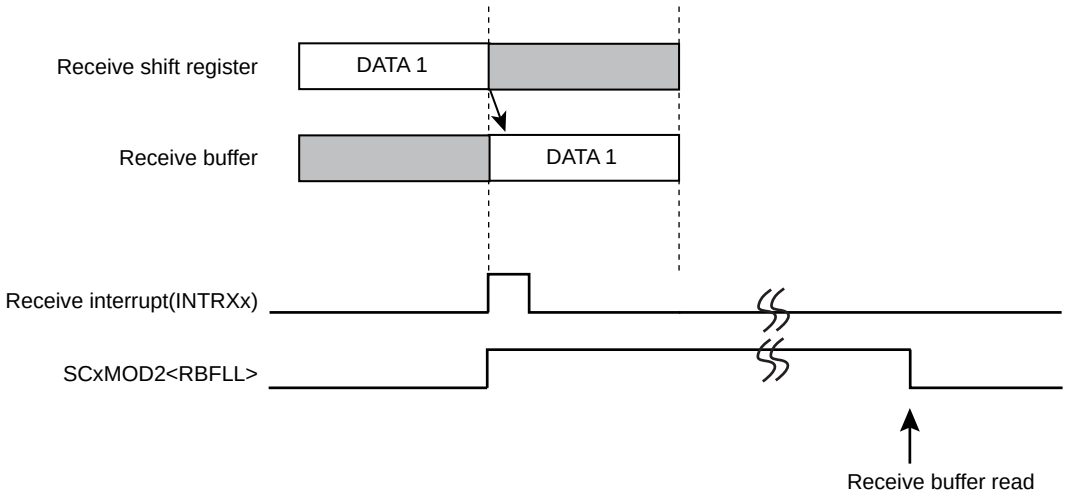


Figure 14-4 Receive Buffer Operation

14.11.3.2 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.

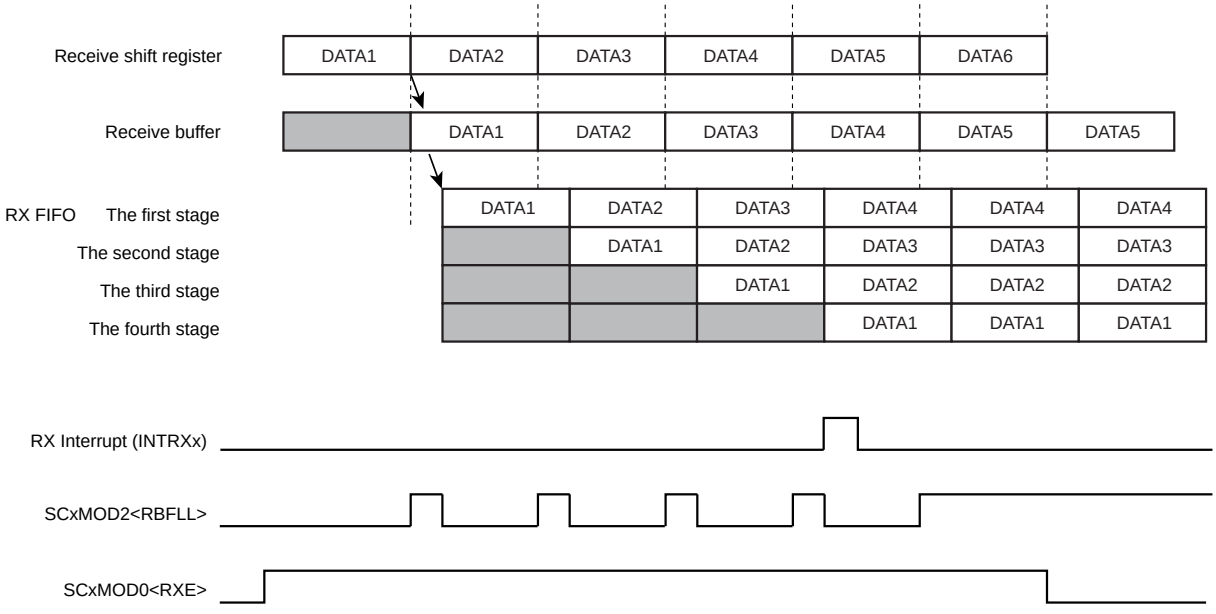


Figure 14-5 Receive FIFO Operation

14.11.3.3 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.

14.11.3.4 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>.

14.11.3.5 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."

14.11.3.6 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.

14.12 Transmission

14.12.1 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.

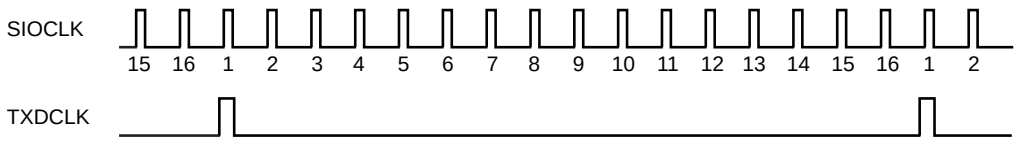


Figure 14-6 Generation of Transmission Clock

14.12.2 Transmission Control

14.12.2.1 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.

14.12.2.2 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.

14.12.3 Transmit Operation

14.12.3.1 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".

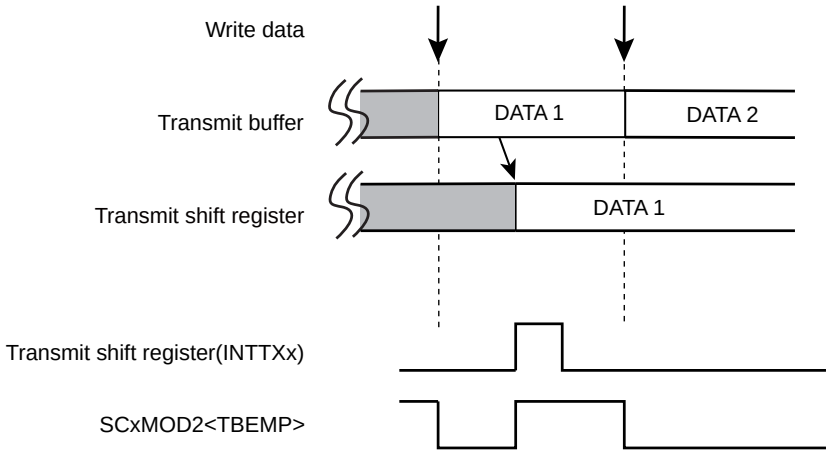


Figure 14-7 Operation of Transmission Buffer (Double-buffer is enabled)

14.12.3.2 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".

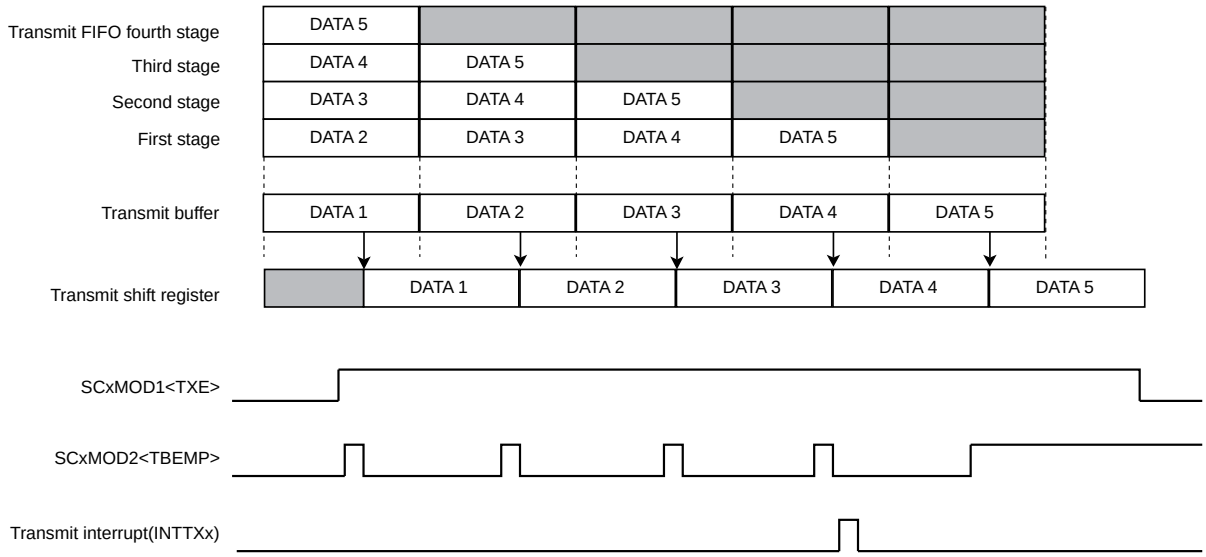
Note: **To use TX FIFO buffer, TX FIFO must be cleared after setting the SIO transfer mode (half duplex/full duplex) and enabling FIFO (SCxFCNF<CNFG> = "1").**

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.



14.12.3.3 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.

14.12.3.4 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.**

14.13 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.

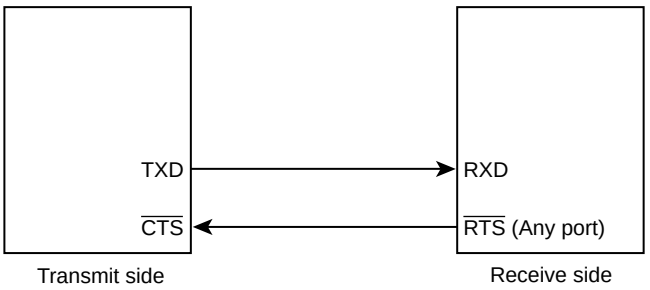


Figure 14-8 Handshake Function

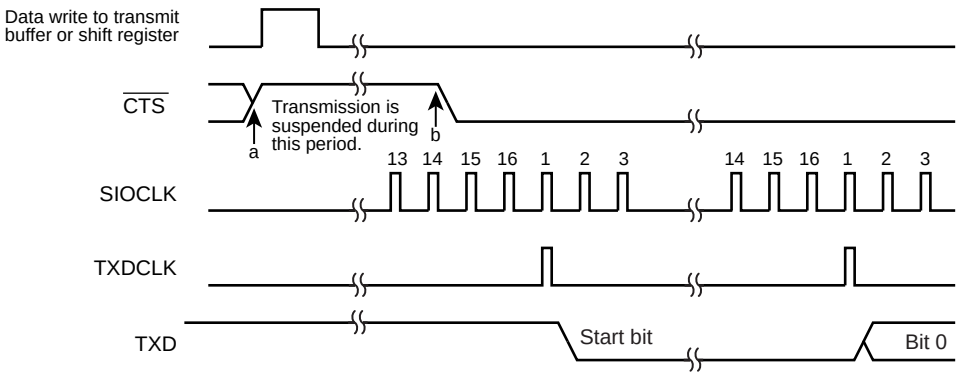


Figure 14-9 $\overline{\text{CTS}}$ Signal timing

14.14 Interrupt/Error Generation Timing

14.14.1 RX Interrupts

Figure 14-10 shows the data flow of receive operation and the route of read.

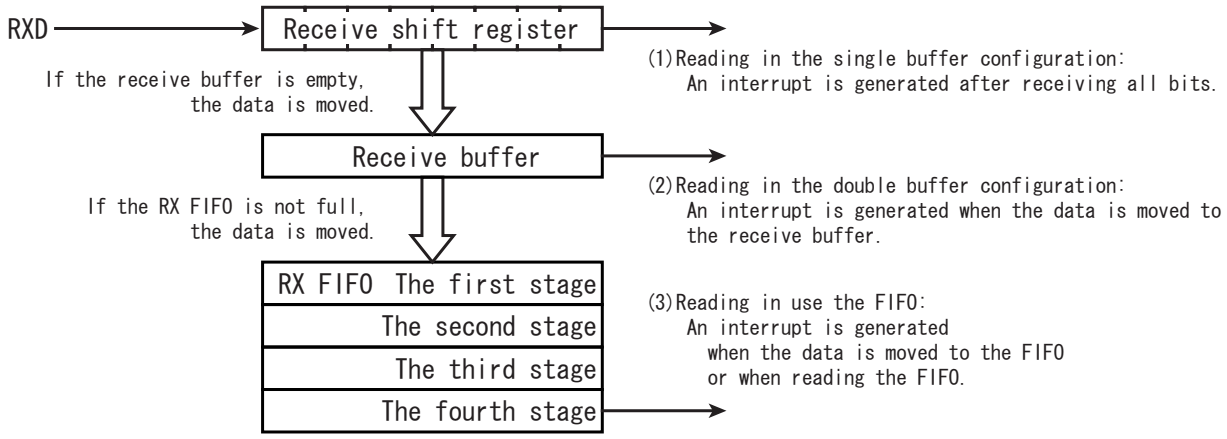


Figure 14-10 Receive Buffer/FIFO Configuration Diagram

14.14.1.1 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.

14.14.1.2 FIFO

When the FIFO is used, a receive interrupt occurs depending on the timing described in Table 14-12 and the condition specified with SCxRFC<RFIS>.

Table 14-12 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

14.14.2 TX interrupts

Figure 14-11 shows the data flow of transmit operation and the route of read.

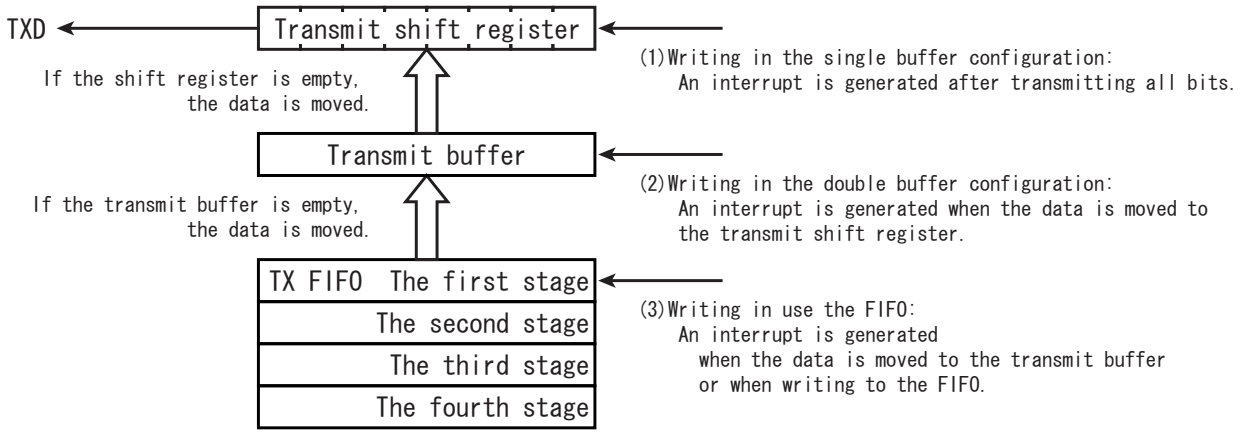


Figure 14-11 Transmit Buffer/FIFO Configuration Diagram

14.14.2.1 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.

14.14.2.2 FIFO

When the FIFO is used, a transmit interrupt occurs depending on the timing described in Table 14-13 and the condition specified with SCxTFC<TFIS>.

Table 14-13 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

14.14.3 Error Generation

14.14.3.1 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

14.14.3.2 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.

14.15 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.

14.16 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,).

14.17 Operation in Each Mode

14.17.1 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.

14.17.1.1 Transmitting Data

(1) SCLK Output Mode

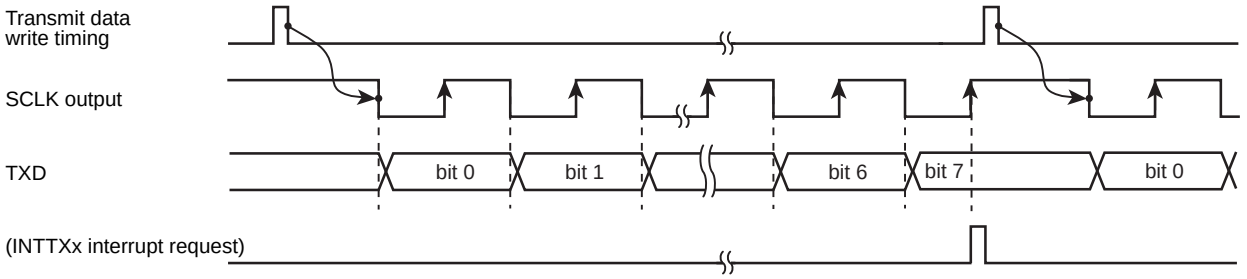
- If the transmit double buffer is disabled (SCxMOD2<WBUF> = "0")

  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.

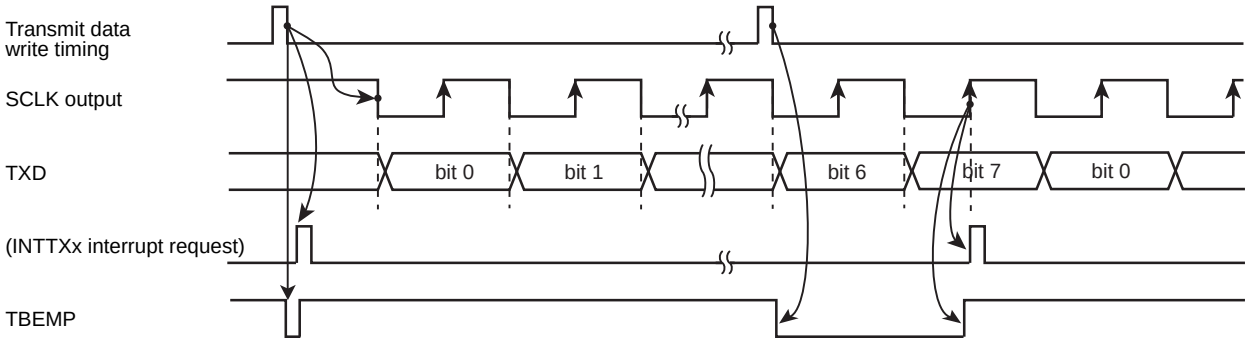
- If the transmit double buffer is enabled (SCxMOD2<WBUF> = "1")

  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.

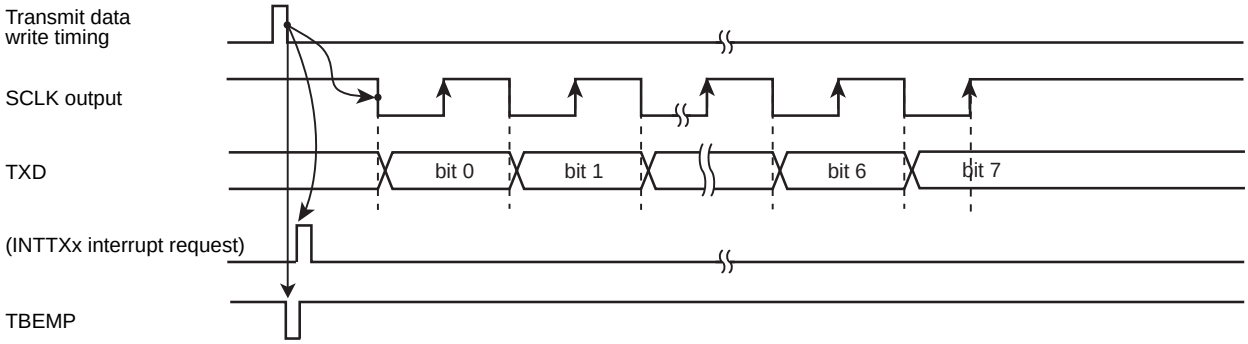
  When data is moved from the transmit buffer to the transmit shift register, if the transmit buffer has no data to be moved to the transmit shift register, INTTXx interrupt is not generated and the SCLK output stops.



<WBUF> = "0" (if double buffering is disabled)



<WBUF> = "1" (if double buffering is enabled and there is data in buffer)



<WBUF> = "1" (if double buffering is enabled and threre is no data in buffer)

Figure 14-12 Transmit Operation in the I/O Interface Mode (SCLK Output Mode)

(2) SCLK Input Mode

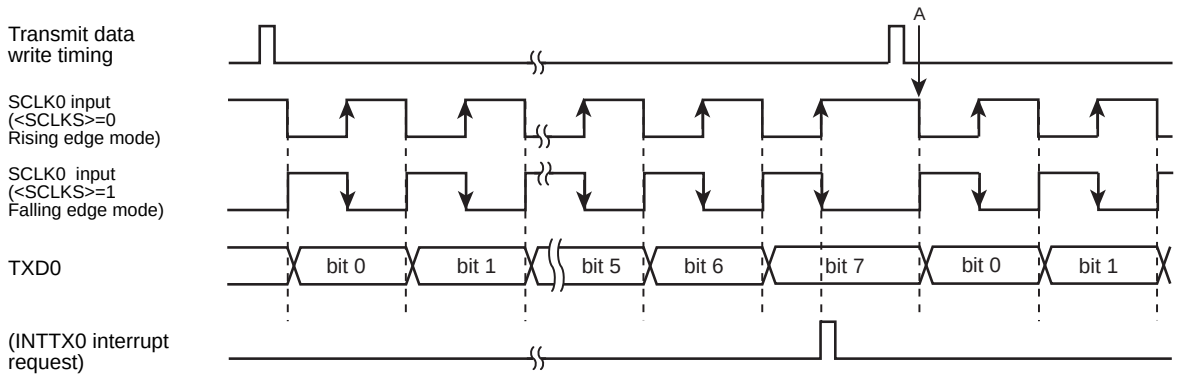
- If double buffering is disabled (SCxMOD2<WBUF> = "0")

  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 14-13.

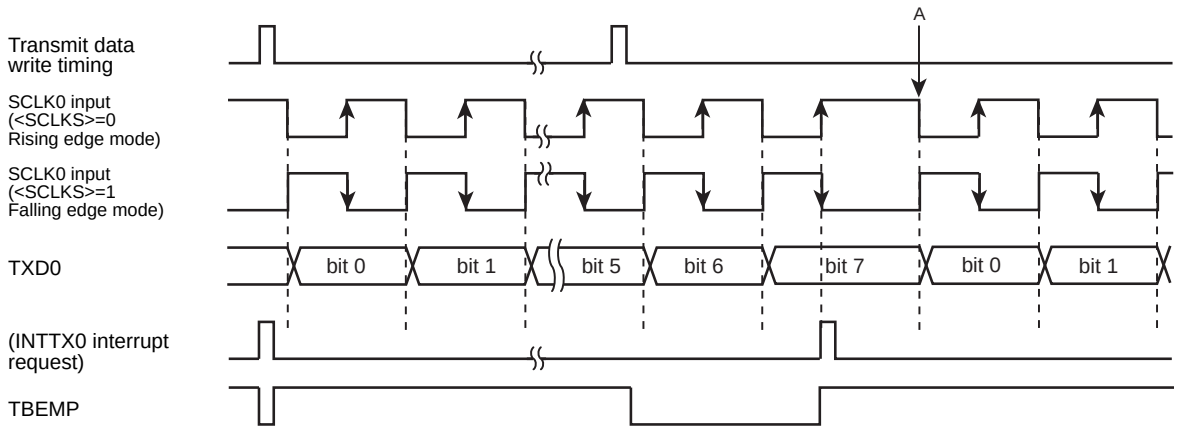
- If double buffer is enabled (SCxMOD2<WBUF> = "1")

  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.

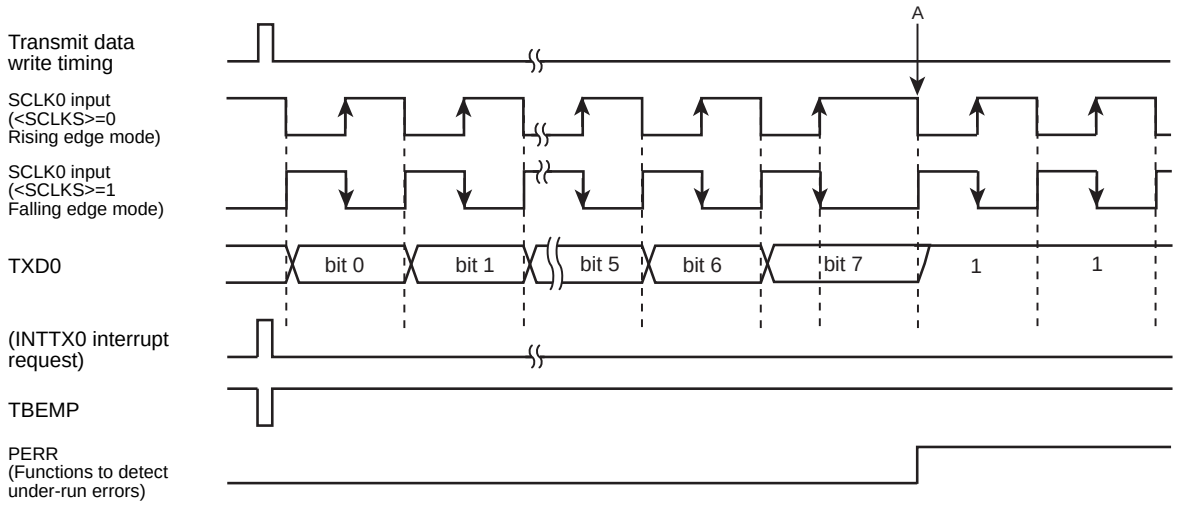
  If the SCLK input becomes active while no data is in the transmit buffer, although the internal bit counter is started, an under-run error occurs and 8-bit dummy data (0xFF) is sent.



<WBUF> = "0" (if double buffering is disabled)



<WBUF> = "1" (if double buffering is enabled and there is data in buffer2)



<WBUF> = "1" (if double buffering is enabled and there is no data in buffer2)

Figure 14-13 Transmit Operation in the I/O Interface Mode (SCLK Input Mode)

14.17.1.2 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.

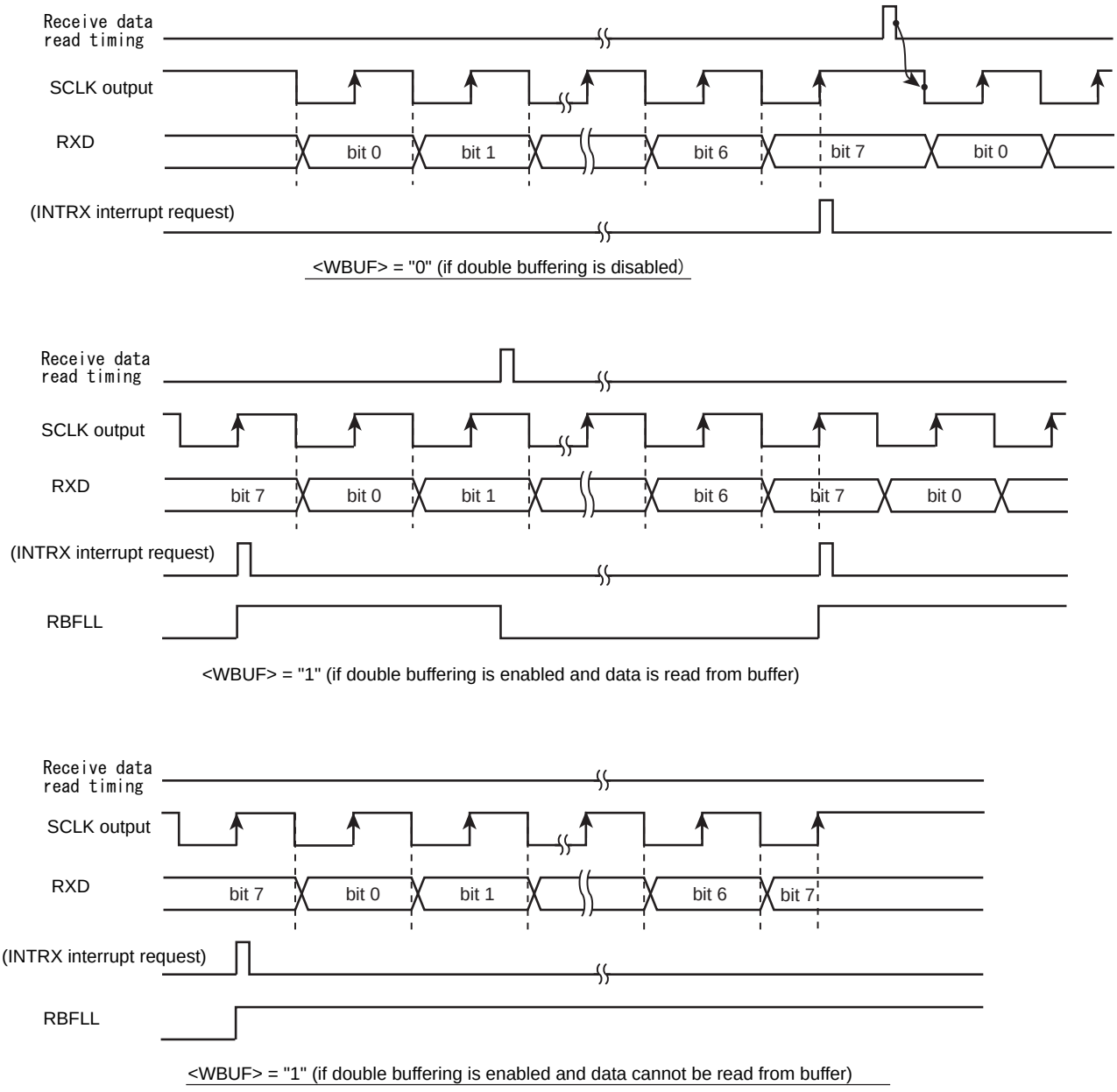


Figure 14-14 Receive Operation in the I/O Interface Mode (SCLK Output Mode)

(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.

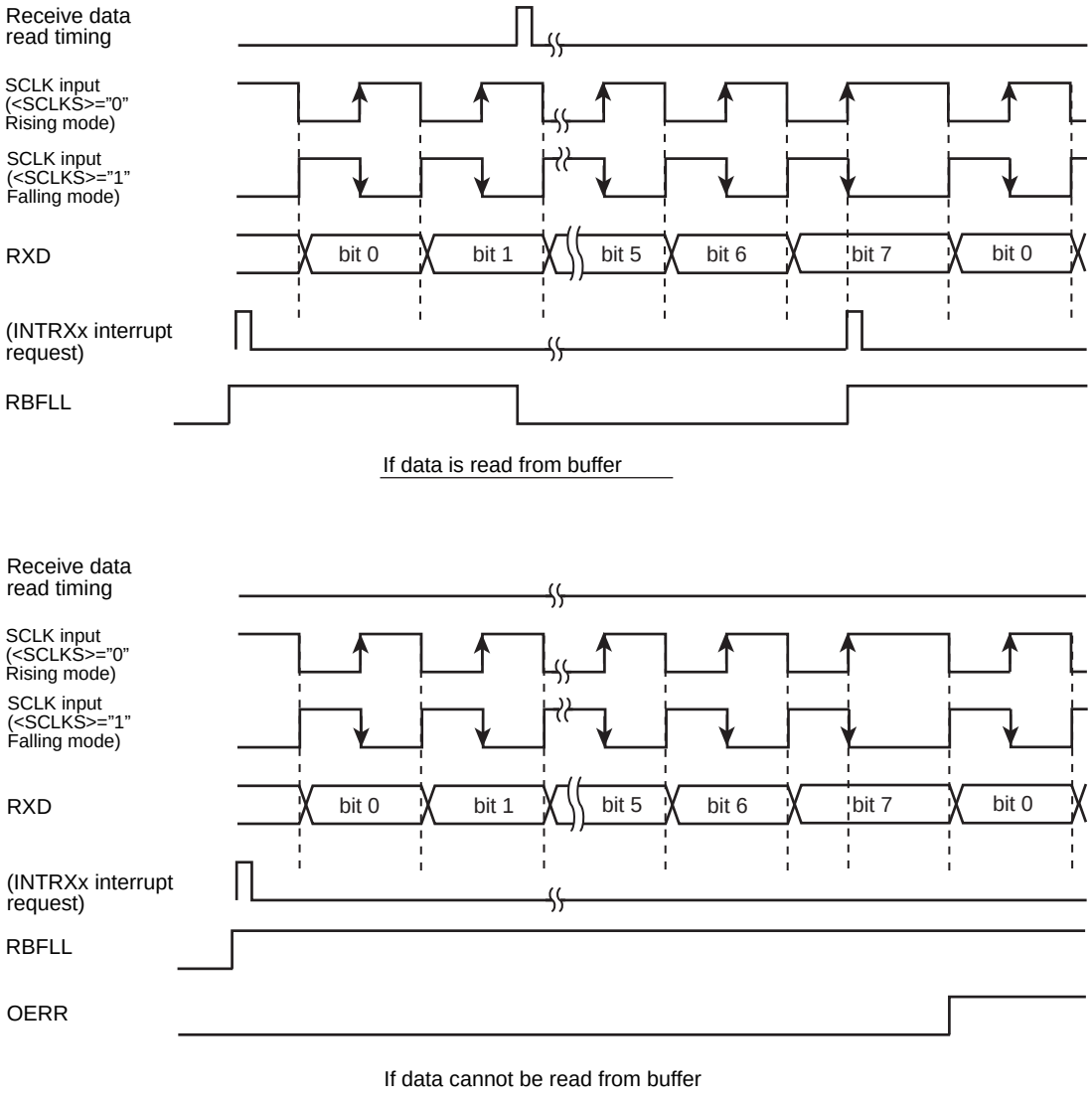


Figure 14-15 Receive Operation in the I/O Interface Mode (SCLK Input Mode)

14.17.1.3 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.

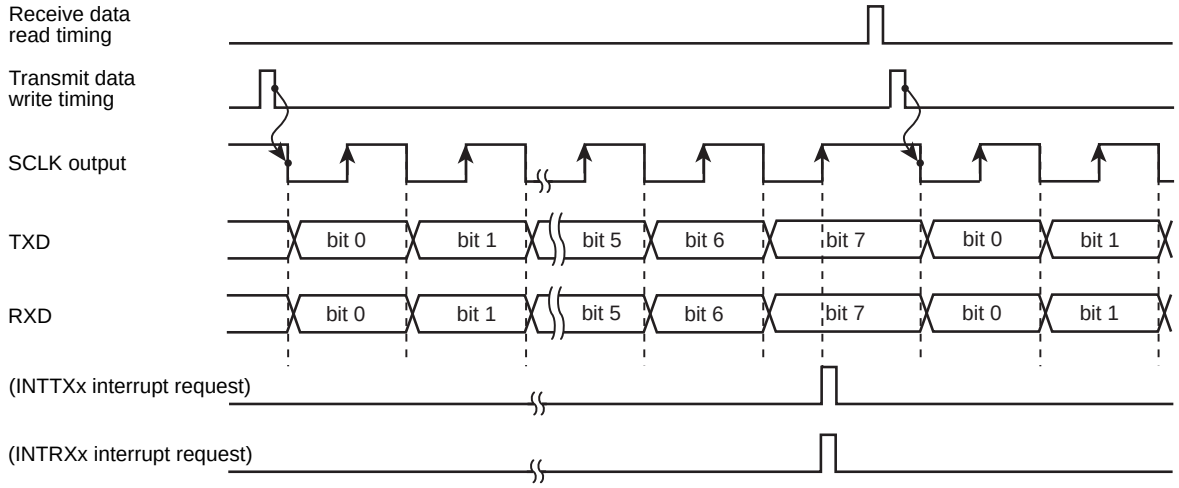
  The next round of data transmission and reception starts when the data is read from the receive buffer and the next transmit data is written to the transmit buffer by the CPU. The order of reading the receive buffer and writing to the transmit buffer can be freely determined. Data transmission is resumed only when both conditions are satisfied.

- If SCxMOD2<WBUF> is set to "1" and the double buffers are enabled

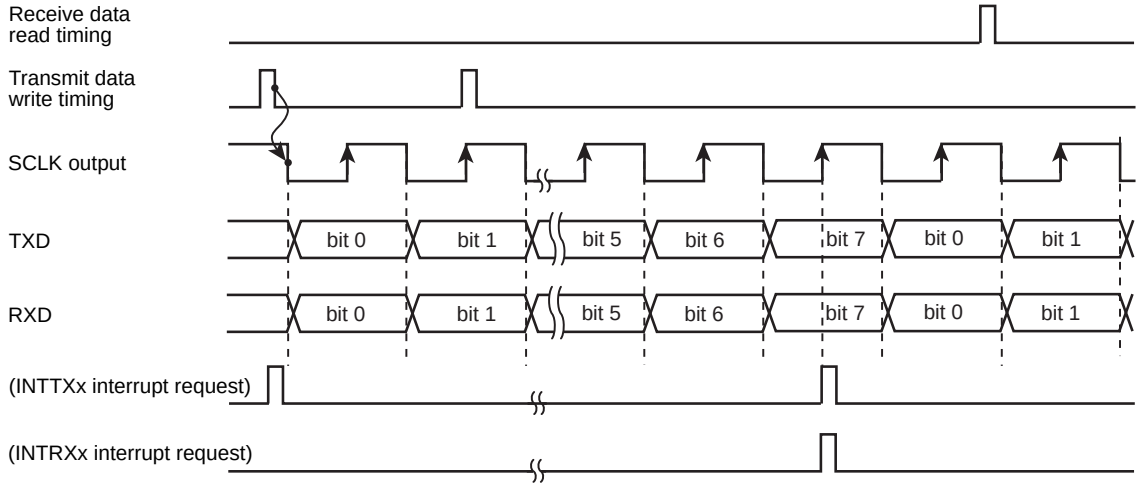
  SCLK is outputted when the CPU writes data to the transmit buffer.

  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.

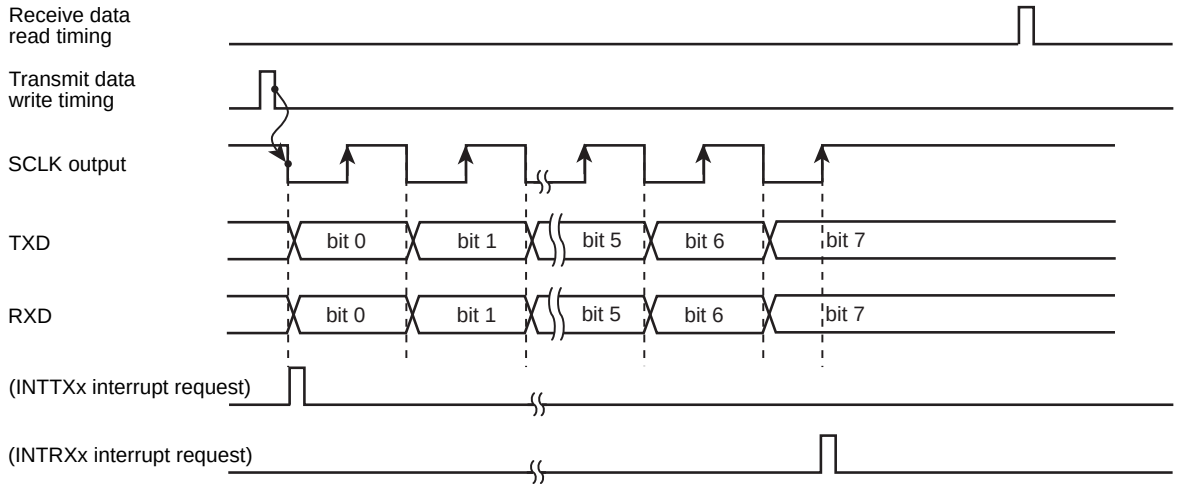
  If the transmit buffer has no data to be moved to the transmit buffer (SCxMOD2<TBEMP> = 1) or when the receive buffer is full (SCxMOD2<RBFULL> = 1), the SCLK output is stopped. When both conditions, receive data is read and transmit data is written, are satisfied, the SCLK output is resumed and the next round of data transmission and reception is started.



<WBUF> = "0" (if double buffering is disabled)



<WBUF> = "1" (if double buffering is enabled)



<WBUF> = "1" (if double buffering is enabled)

Figure 14-16 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 14-17). 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 14-17). 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.

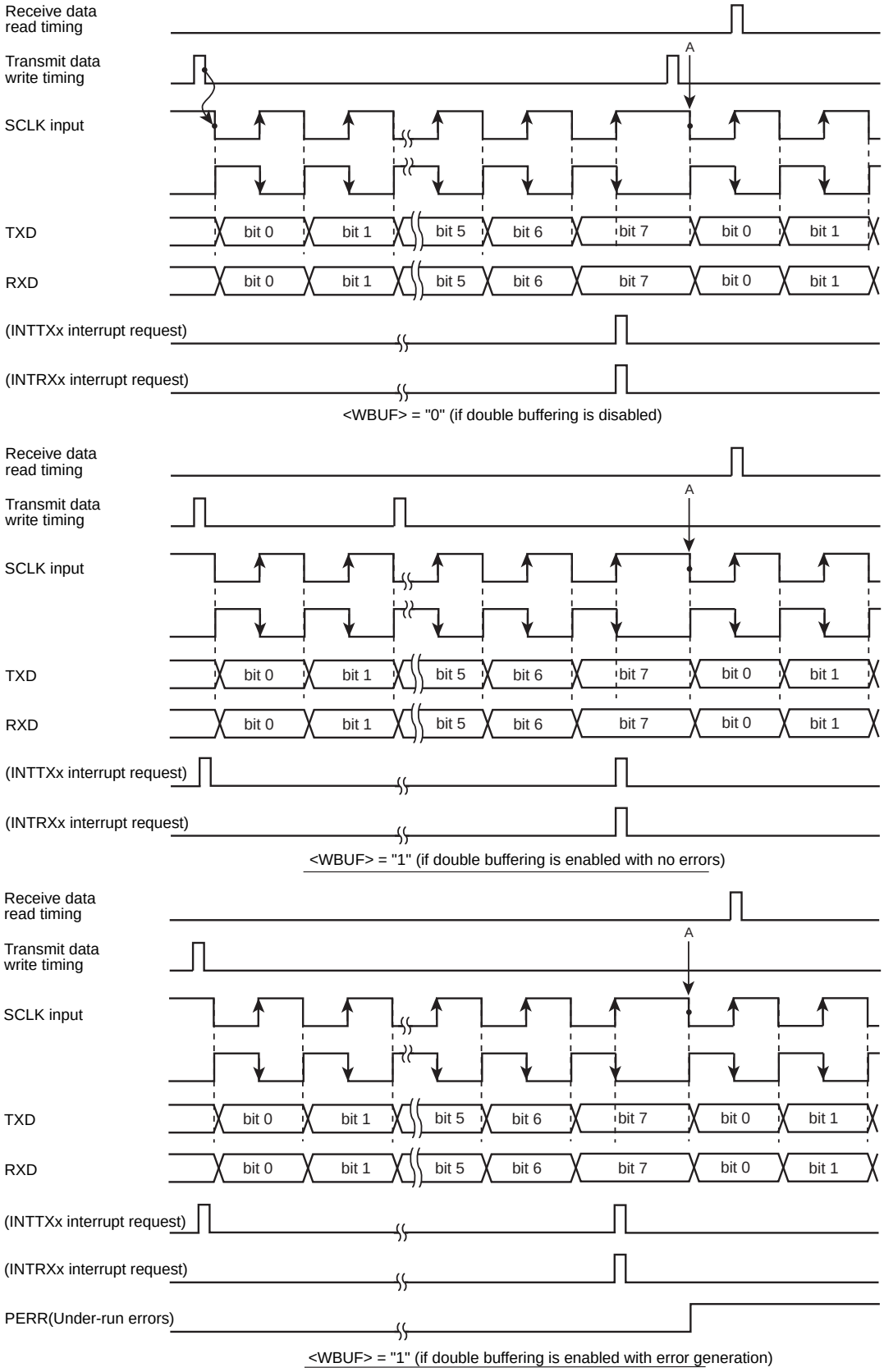


Figure 14-17 Transmit/Receive Operation in the I/O Interface Mode (SCLK Input Mode)

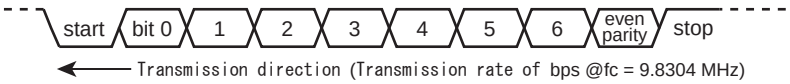
14.17.2 Mode 1 (7-bit UART mode)

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.



Transmission direction (Transmission rate of bps @fc = 9.8304 MHz)

Clocking conditions	system clock :	High-speed (fc)
	High-speed clock gear:	X1 (fc)
	Prescaler clock:	fperiph/2 (fperiph = fsys)

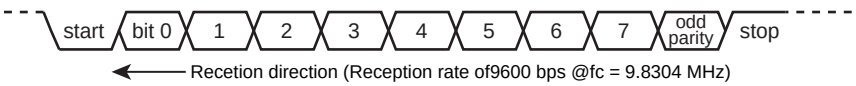
		7	6	5	4	3	2	1	0	
SCxMOD0	←	x	0	-	0	0	1	0	1	Set 7-bit UART mode
SCxCR	←	x	1	1	x	x	x	0	0	Even parity enabled
SCxBRCR	←	0	0	1	0	0	1	0	0	Set 2400bps
SCxBUF	←	*	*	*	*	*	*	*	*	Set transmit data

x : don't care - : no change

14.17.3 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:



Recetion direction (Reception rate of9600 bps @fc = 9.8304 MHz)

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

14.17.4 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>.

14.17.4.1 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.

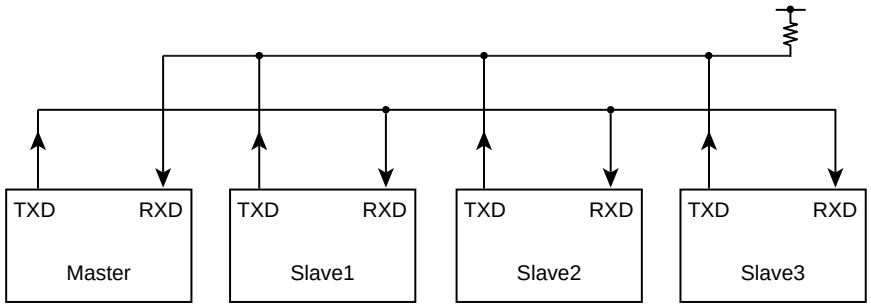


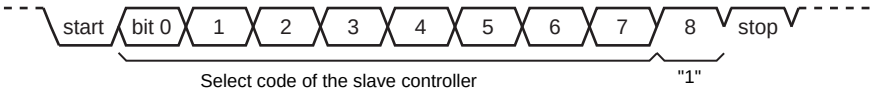
Figure 14-18 Serial Links to Use Wake-up Function

14.17.4.2 Protocol

1. Select the 9-bit UART mode for the master and slave controllers.

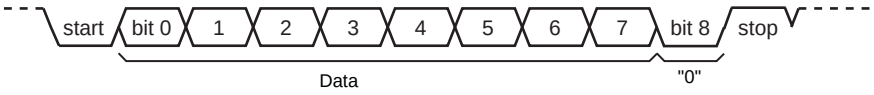
2. Set SCxMOD<WU> to "1" for the slave controllers to make them ready to receive data.

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.

15. Serial Bus Interface (I2C/SIO)

The TMPM365FYXBG 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 15-1 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

15.1 Configuration

The configuration is shown in Figure 15-1.

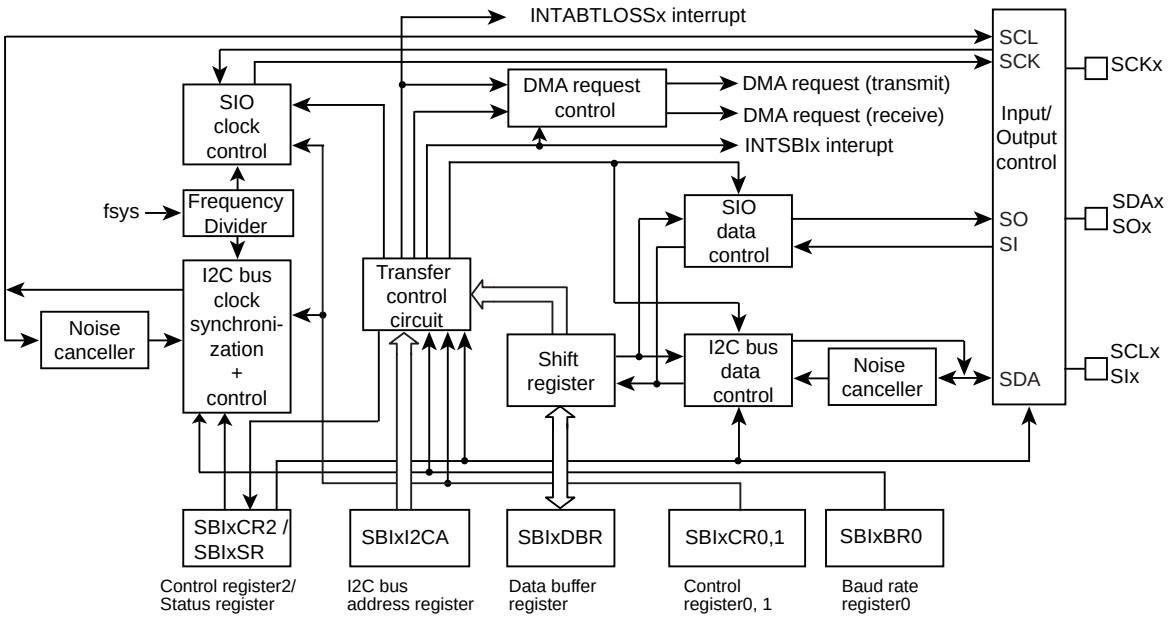


Figure 15-1 (I2C/SIO) Block Interface

15.2 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 "15.4 Control Registers in the I2C Bus Mode" and "15.7 Control register of SIO mode".

15.2.1 Registers for each channel

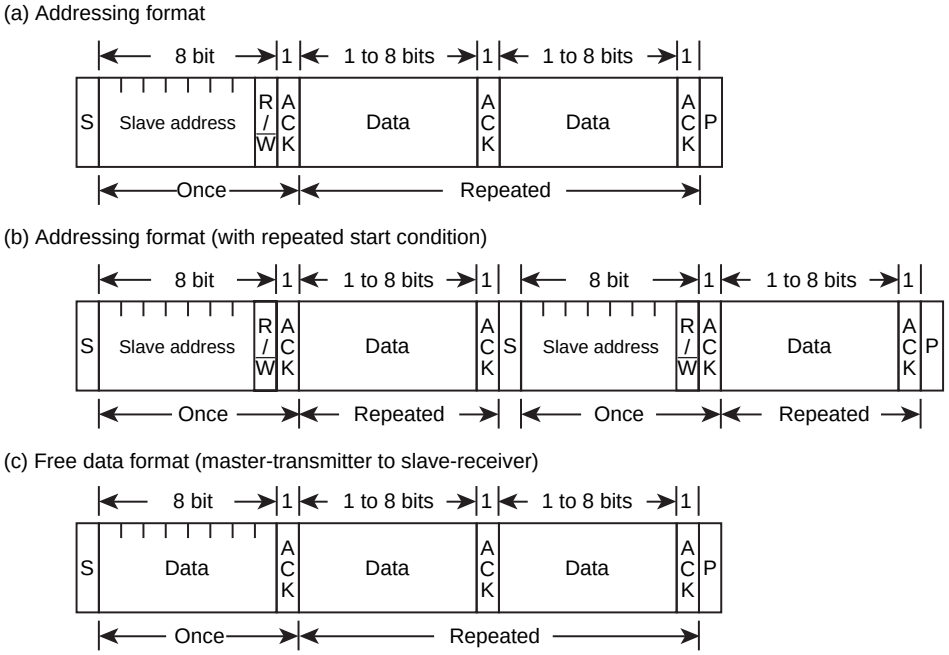
The tables below show the registers and register addresses for each channel.

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

15.3 I2C Bus Mode Data Format

Figure 15-2 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 15-2 I2C Bus Mode Data Formats

15.4 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.

15.4.1 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.

15.4.2 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
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

System Clock: fsys (= 48MHz)
Clock gear : fc/1
Frequency = $\frac{fsys}{2^n + 72}$ [Hz]

Note 1: **Clear <BC[2:0]> to "000" before switching the operation mode to the SIO mode.**

Note 2: **For details on the SCL line clock frequency, refer to "15.5.1 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 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.

15.4.3 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.

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.**

15.4.4 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"

15.4.5 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".

15.4.6 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)	Receive data
		W (Transmit)	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.**

15.4.7 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.**

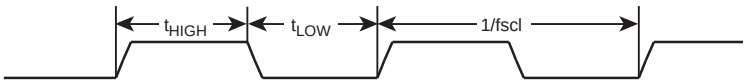
Note 2: **Do not set SBIxI2CAR to "0x00" in slave mode. (If SBIxI2CAR is set to "0x00", it's recognized that the slave address matches the START byte ("0x01") of the I2C standard received in slave mode.)**

15.5 Control in the I2C Bus Mode

15.5.1 Serial Clock

15.5.1.1 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 15-3 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.**

15.5.1.2 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.

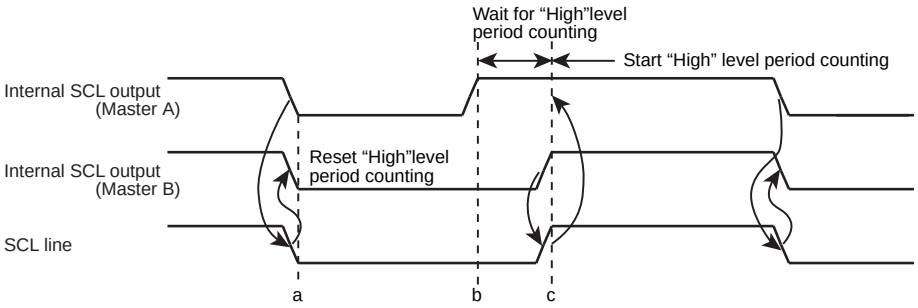


Figure 15-4 Example of Clock Synchronization

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.

15.5.2 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 not counted.

15.5.3 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.

15.5.4 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.

15.5.5 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.

15.5.6 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.

15.5.7 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.

15.5.8 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.

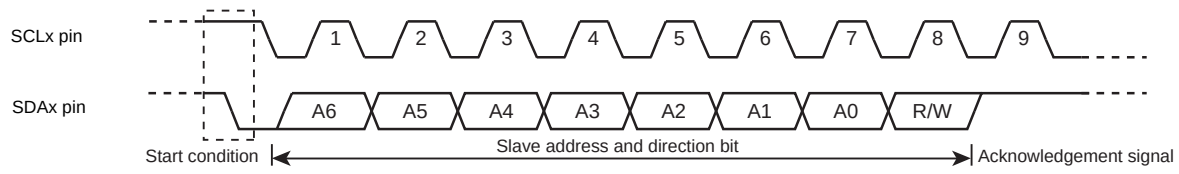


Figure 15-5 Generating the Start Condition and a Slave Address

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.

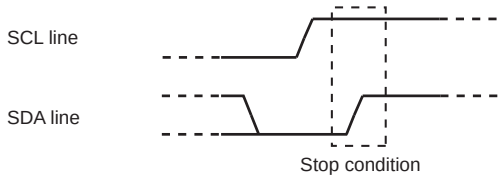


Figure 15-6 Generating the Stop Condition

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).

15.5.9 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).

Note: When the INTSBIx interrupt occurs, a DMA request occurs at the same time. If DMA is used, set the corresponding registers of the DMA controller.

15.5.10 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. When a start condition is output under the bus busy state, a start condition is not output to SCL or SDA line, thus arbitration lost occurs.

The arbitration procedure for two masters on a bus is shown below.

Up until the point a, Master A and Master B output the same data. At the point a, Master A outputs the "Low" level and Master B outputs the "High" level.

Then Master A pulls the SDA bus line to the "Low" level because the line has the wired-AND connection. When the SCL line goes high at the point b, the slave device reads the SDA line data, i.e., data transmitted by Master A. At this time, data transmitted by Master B becomes invalid.

This condition of Master B is called "Arbitration Lost". Master B releases its SDA pin, so that it does not affect the data transfer initiated by another master. If two or more masters have transmitted exactly the same first data word, the arbitration procedure continues with the second data word.

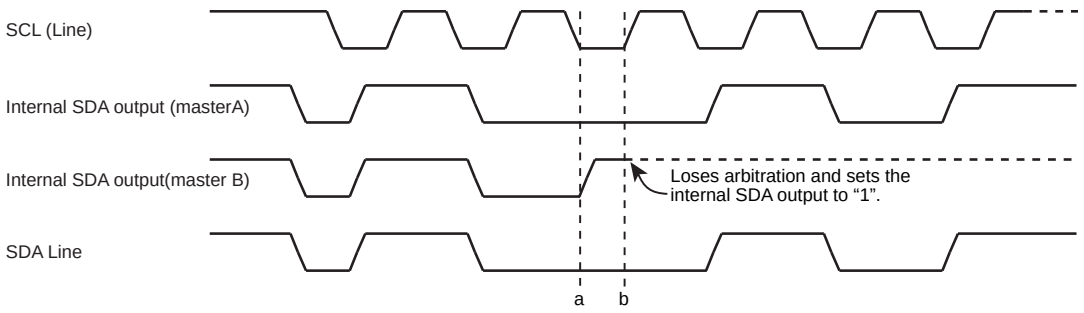


Figure 15-7 Lost Arbitration

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" and the INTABLOSSx interrupt occurs.

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.

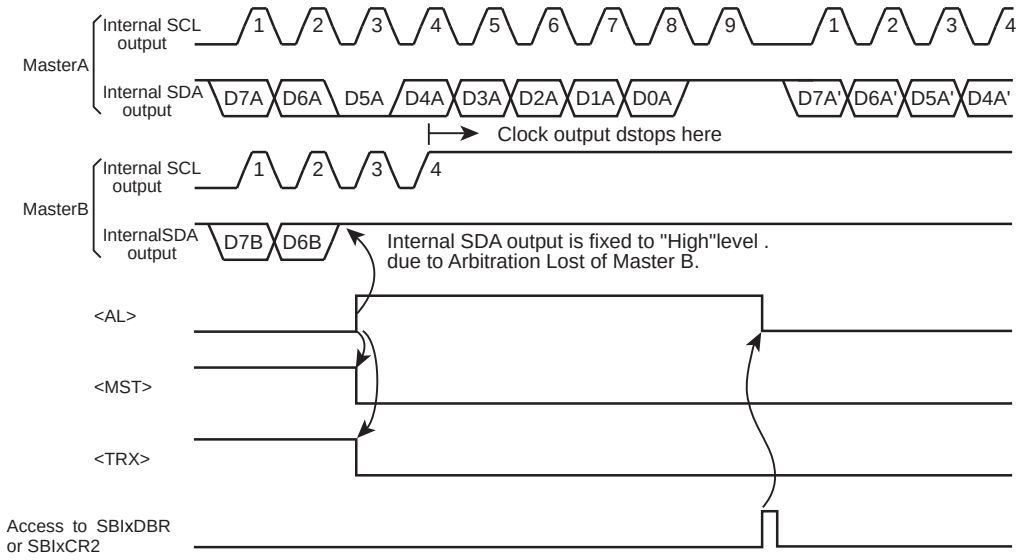


Figure 15-8 Example of Master B Lost Arbitration (D7A = D7B, D6A = D6B)

15.5.11 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.

15.5.12 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.

15.5.13 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.

15.5.14 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.

15.5.15 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.

15.5.16 Software Reset

If the serial bus interface circuit locks up due to external noise, it can be initialized by using a software reset.

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 serialbus 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 [1:0]> is automatically cleared to "0".

Note: A software reset causes the SBI operating mode to switch from the I2C mode to the port mode.

15.6 Data Transfer Procedure in the I2C Bus ModeI2C

15.6.1 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

15.6.2 Generating the Start Condition and a Slave Address

15.6.2.1 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 slave 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
```

15.6.2.2 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".

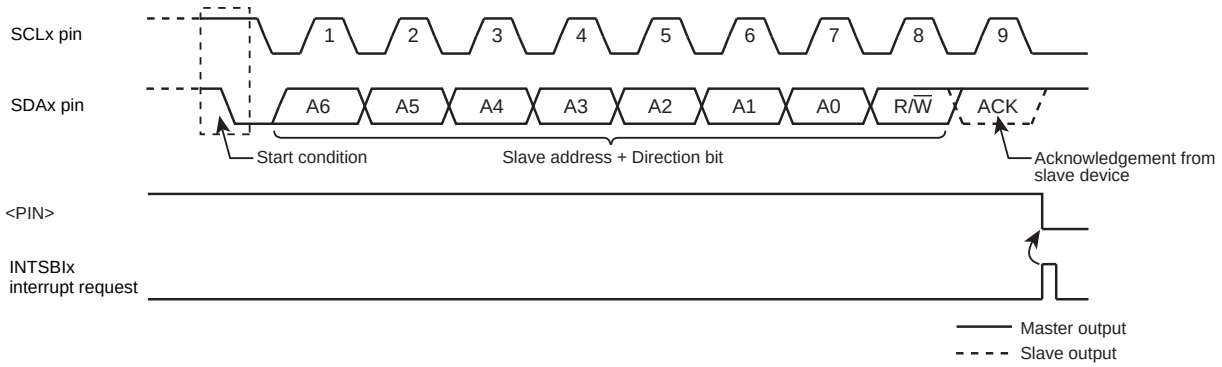


Figure 15-9 Generation of the Start Condition and a Slave Address

15.6.3 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.

15.6.3.1 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**

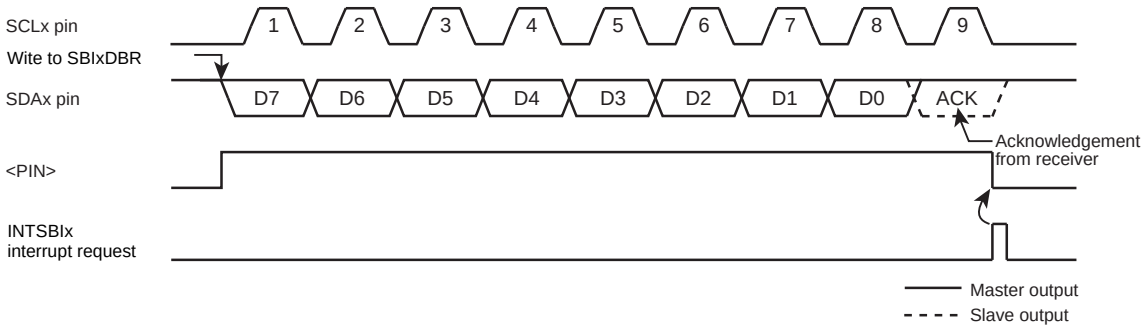


Figure 15-10 <BC[2:0]>= "000",<ACK>= "1" (Transmitter Mode)

(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.

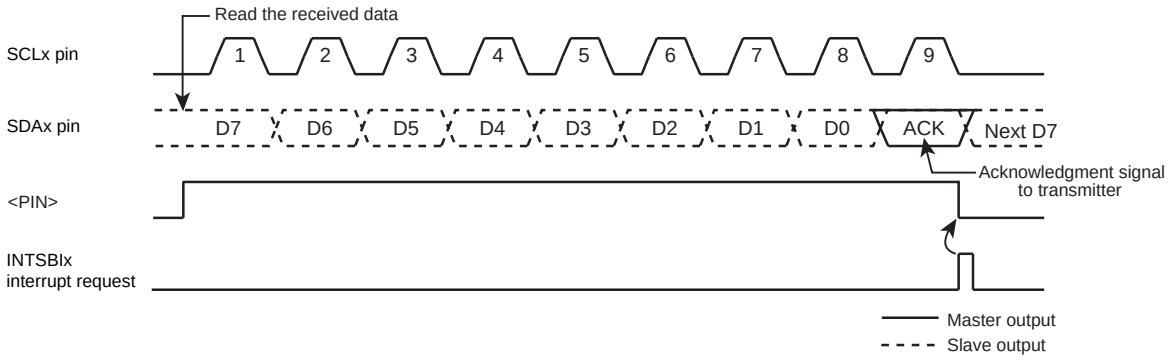


Figure 15-11 <BC[2:0]>= "000",<ACK>= "1" (Receiver Mode)

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.

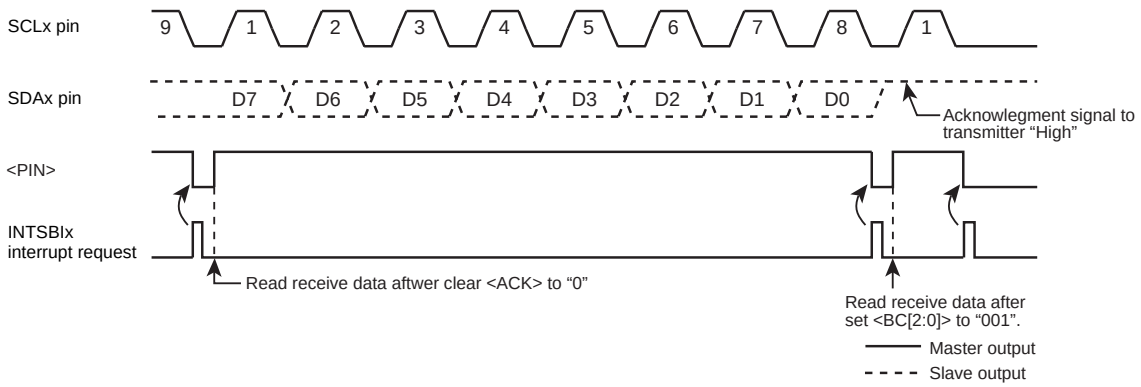


Figure 15-12 Terminating Data Transmission in the Master Receiver Mode

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

15.6.3.2 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 15-2 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 15-2 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.

15.6.4 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.

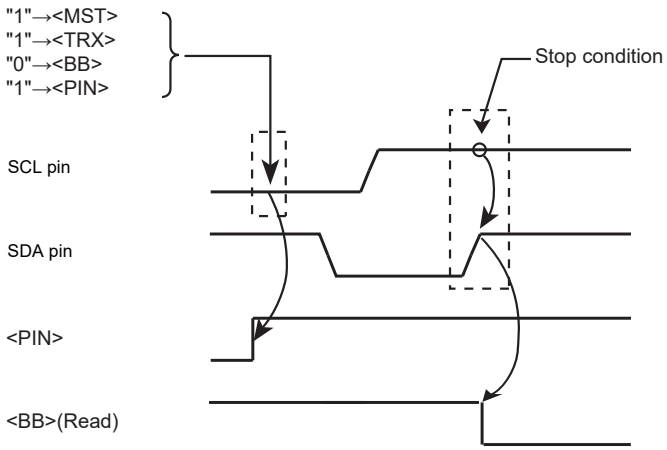


Figure 15-13 Generating the Stop Condition

15.6.5 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 "15.6.2 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

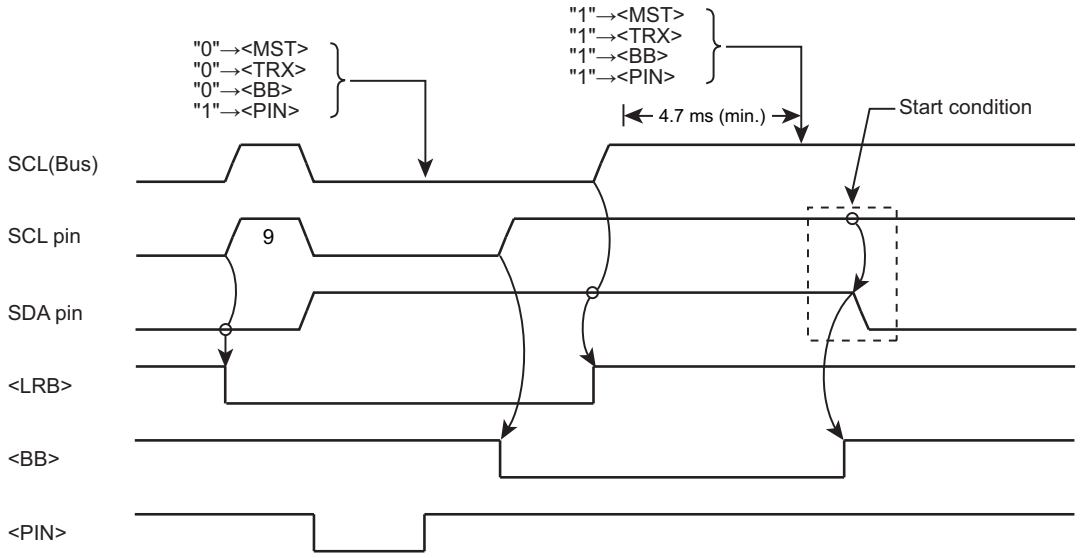


Figure 15-14 Timing Chart of Generating a Restart

15.6.6 Data Transfer using DMA

Data can be transfterred using DMA in I2C mode.But one master device and one slave device are connected to a bus line and the number of transfer data is decied beforehand.

Since a DMA request occurs at the same time when the INTSBIx request occurs in I2C mode, data can be transferred continuously by executing DMA transfer between SBIxDBR (data buffer) and the memory.

A DMA request for transmission and reception are separately executed.

Set enable or disable to the DMAC control register for transmission and reception before using the DMA.

To transfer data using DMA, the number of bytes to be transferred should be preliminarily specified on the both transmission and reception.

In the receive mode, data received for the last and the second-to-last reception should be read using software.

If arbitration lost occurs during I2C transfer, the INTABTLOSSx interrupt and the INTSBIx interrupt occurs, then the communication stops. A DMA request does not occur.

Note: **DMA transfer cannot be performed in SIO mode.**

Table 15-3 DMA request number

Channel	Transmitssion / 1reception	DMA request number
Channel 0	Reception	12
	Transmission	13
Channel 0	Reception	14
	Transmission	15

15.6.6.1 How to transfer data in master mode

1. Generate start condition and slave addres.
2. Check <LRB> and <TRX> in interrupt service ruoutine of INTSBIx after output slave adress.
3. When <LRB> is "1", output a stop condition and complete the transferring because no slave device which has the sent slave addrress is on the bus.
4. When <LRB> is "0", check <TRX> because slave device which has the sent slave addrress is on the bus. The proceeding is depended on <TRX>.

(1) When <TRX> is "1" (Tramsmitter mode)

a. Set DMA such as transfer bit width, burst size and the total transferring number, etc. Subtract 1 from the number of data to be transmitted. And enable to receive DMA request of INTSBI.
b. Write the first data into SBIDBRx. Start tranmit the first data by this writing.
c. Start DMA transfer by the DMA request of INTSBIx which is after the completed transferring. And transmit the 2nd data and following data.
d. When the specified the number of DMA transfer (N-1 times) ends, a DMA transfer end interrupt occurs. Change the settings to ignore a DMA request reception in this interrupt service routine. (Configure the setting not to generate a DMA request before INTSBIx interrupt occurs.)
e. Generate stop condition to stop transmition at the interrupt service routine of INTSBIx.

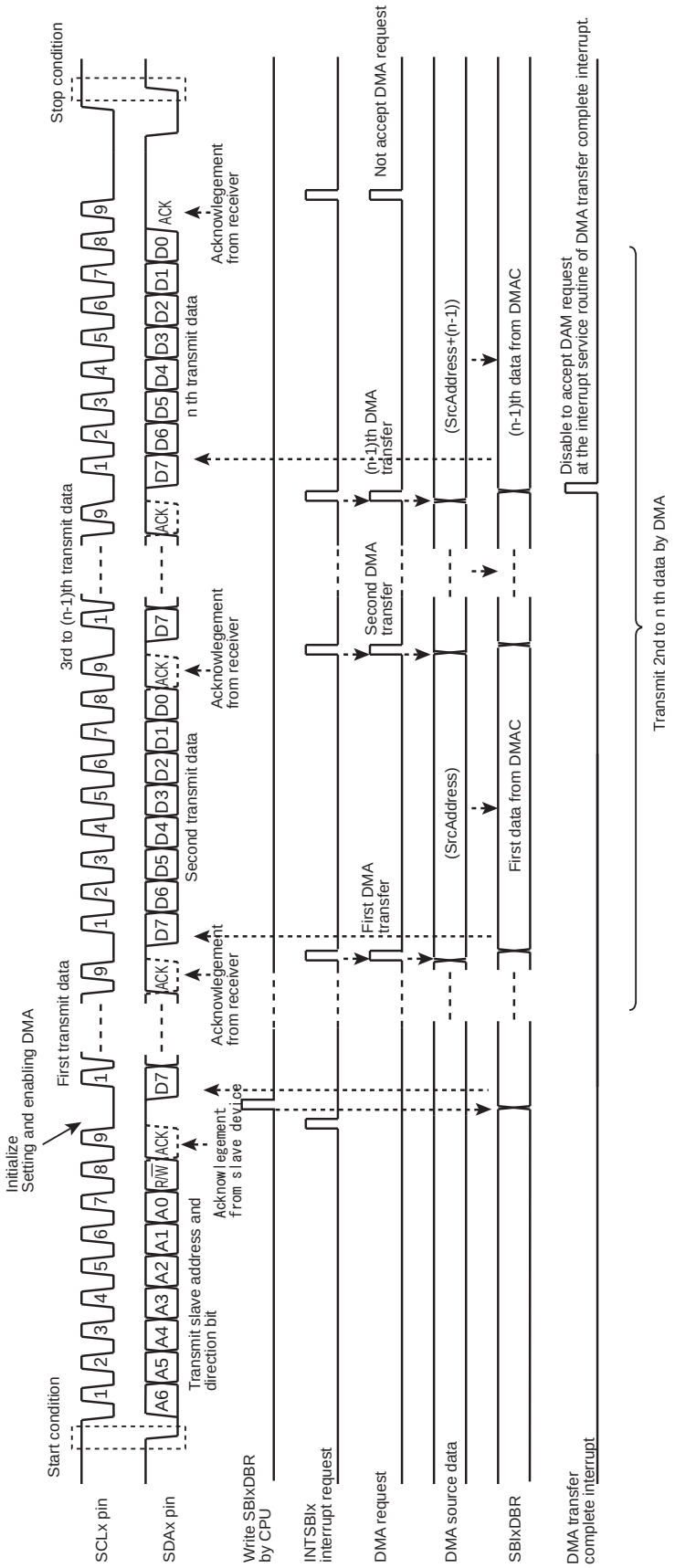


Figure 15-15 Master Transmit Mode

(2) When <TRX> is "0" (Receiver mode)

a. Set DMA such as transfer bit width, burst size and the total transferring number, etc. Subtract 2 from the number of data to be transmitted. And enable to receive DMA request of INTSBI.

b. Read dummy data from SBIDBRx. Start receive the first data by this reading.

c. Start DMA transfer by the DMA request of INTSBIx which is after the completed receiving. And receive the 2nd data and following data.

d. When the specified the number of DMA transfer (N-2 times) ends, a DMA transfer end interrupt occurs. Change the settings to ignore a DMA request reception in this interrupt service routine. (Configure the setting not to generate a DMA request before INTSBIx interrupt occurs.)

e. Clear <ACK> not to generate the clock for acknowlegement at the interrupt service routine of INTSBIx. Read received data from SBIDBRx.

f. Set <BC> to "001" and read received data at the interrupt service routine of INTSBIx. Receive one bit data with high level's SDA by this setting. Transmitter receives a negative acknowlegement.

g. Generate stop condition to stop transmition at the interrupt service routine of INTSBIx.

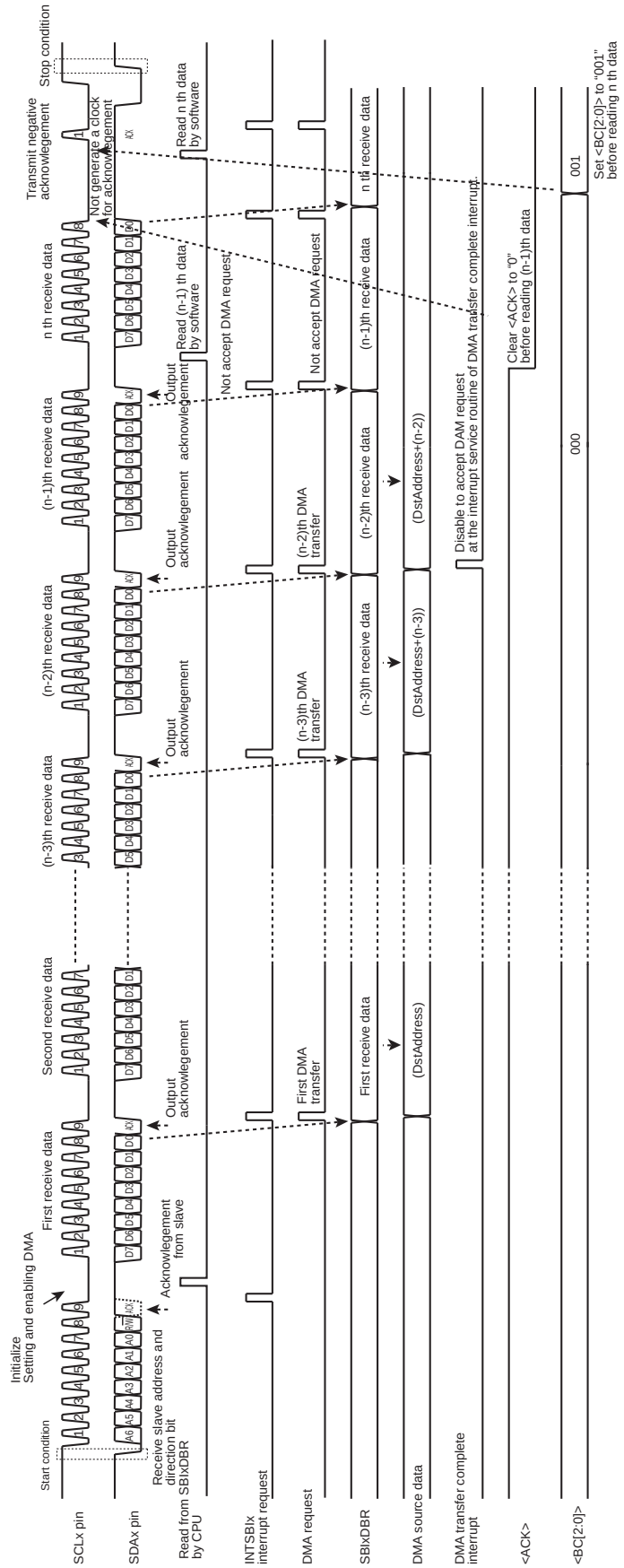


Figure 15-16 Master Receiver Mode

15.6.6.2 How to transfer data in slave mode

1. Receive start condition and slave addres.
2. Check <TRX> in interrupt service ruoutine of INTSBIx after receive slave adress.
3. Check <TRX>. The proceeding is depended on <TRX>.

(1) When <TRX> is "1" (Tramsmitter mode)

a. Set DMA such as transfer bit width, burst size and the total transferring number, etc. Subtract 1 from the number of data to be transmitted. And enable to receive DMA request of INTSBI.
b. Write the first data into SBIDBRx. TMPM365FYXBG can be receive a clock from a master device and start tranmit the first data by this writing.
c. Start DMA transfer by the DMA request of INTSBIx which is after the completed transferring. And transmit the 2nd data and following data.
d. When the specified the number of DMA transfer (N-1 times) ends, a DMA transfer end interrupt occurs. Change the settings to ignore a DMA request reception in this interrupt service routine. (Configure the setting not to generate a DMA request before INTSBIx interrupt occurs.)
e. Wait stop condition from a master device without writing a data at the service routine of INTSBIx.

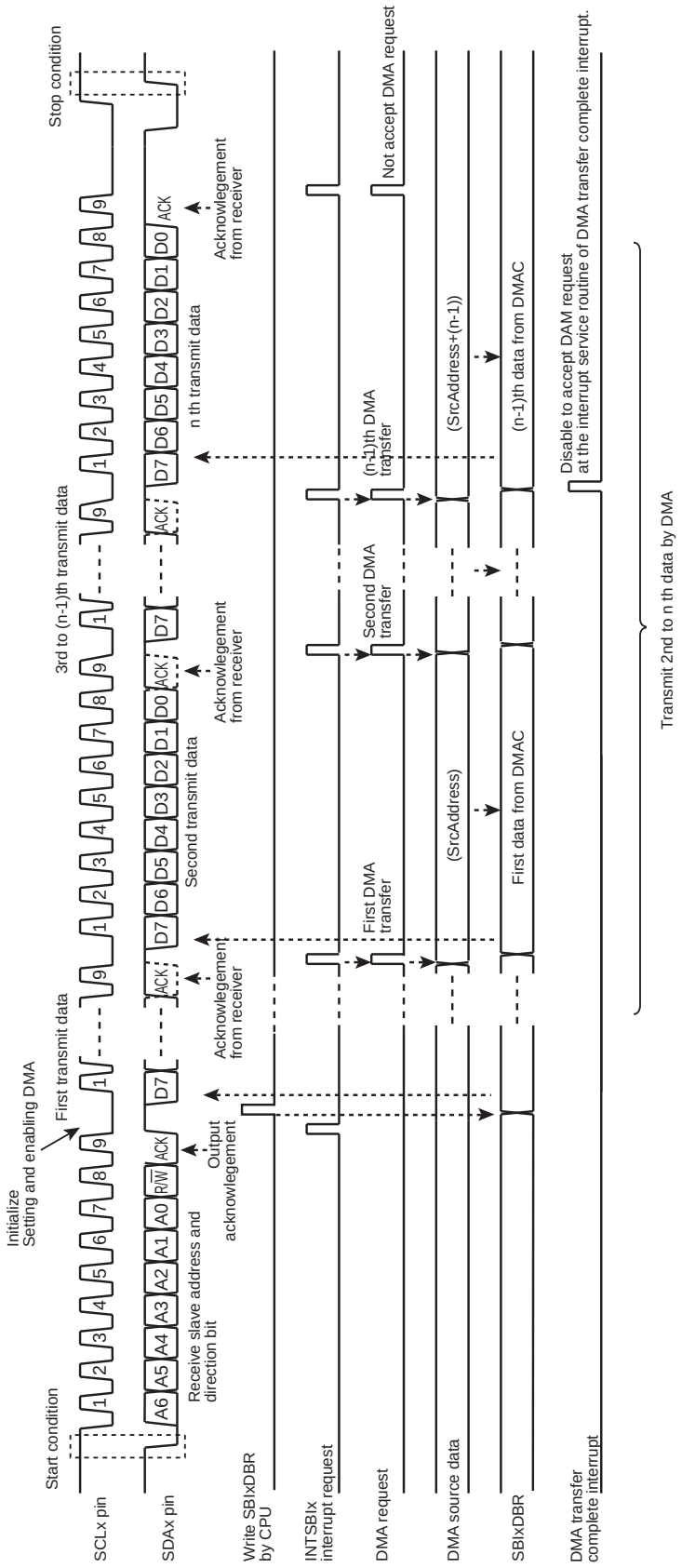


Figure 15-17 Slave Trasmit Mode

(2) When <TRX> is "0" (Receiver mode)

a. Set DMA such as transfer bit width, burst size and the total transferring number, etc. Subtract 2 from the number of data to be transmitted. And enable to receive DMA request of INTSBI.

b. Read dummy data from SBIDBRx. Start receive the first data by this reading.

c. Start DMA transfer by the DMA request of INTSBIx which is after the completed receiving. And receive the 2nd data and following data.

d. When the specified the number of DMA transfer (N-2 times) ends, a DMA transfer end interrupt occurs. Change the settings to ignore a DMA request reception in this interrupt service routine. (Configure the setting not to generate a DMA request before INTSBIx interrupt occurs.)

e. Clear <ACK> not to generate the clock for acknowlegement at the interrupt service routine of INTSBIx. Read received data from SBIDBRx.

f. Set <BC> to "001" and read received data at the interrupt service routine of INTSBIx. Receive one bit data with high level's SDA by this setting. Transmitter receives a negative acknowlegement.

g. Wait stop condition from a master device without writing a data at the service routine of INTSBIx.

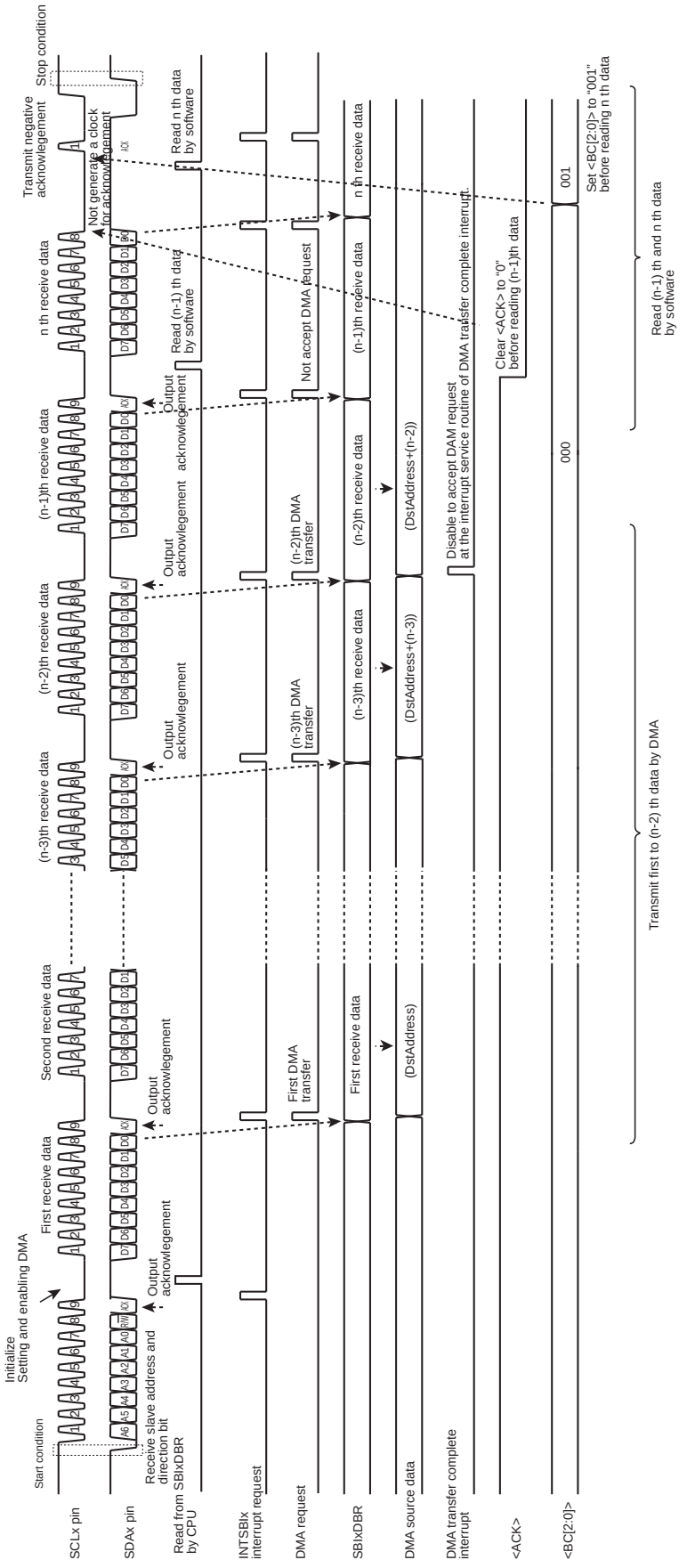


Figure 15-18 Slave Receive Mode

15.7 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.

15.7.1 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.

15.7.2 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.**

Note 3: **In the master mode, when <BC[2:0]>="001" and <ACK>="1" are set, the SCL line may be fixed to "L" on the falling edge of the SCL line after a stop condition occurs. So another bus master device cannot use the bus. If multiple master devices are connected to the bus, set two or more to the number of transfer bit before a stop condition occurs.**

15.7.3 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.**

15.7.4 SBIxCR2(Control register 2)

This register serves as SBIxSR register by reading 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.**

15.7.5 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(Note)	1(Note)	1(Note)	0	0	1(Note)	1(Note)

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-4	-	R	Read as 1.(Note)
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)

Note: **In this document, the value written in the column "after reset" is the value after setting the SIO mode in the initial state.**

15.7.6 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".

15.8 Control in SIO mode

15.8.1 Serial Clock

15.8.1.1 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.

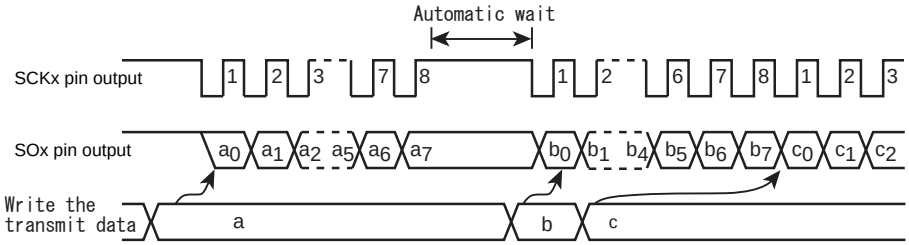


Figure 15-19 Automatic Wait

(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.

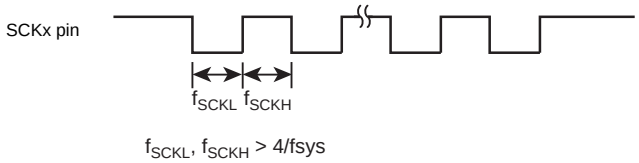


Figure 15-20 Maximum Transfer Frequency of External Clock Input

15.8.1.2 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).

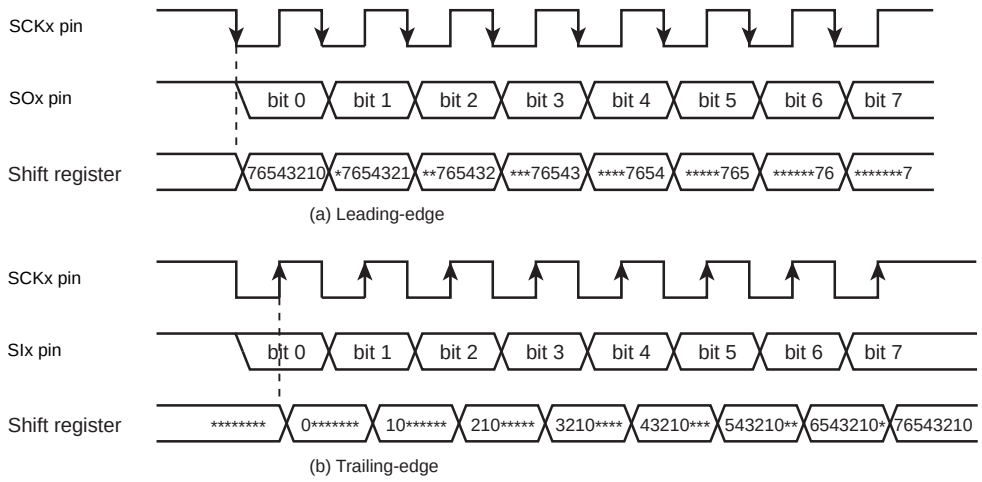


Figure 15-21 Shift Edge

15.8.2 Transfer Modes

The transmit mode, the receive mode or the transmit/receive mode can be selected by programming SBIxCR1<SIOM[1:0]>.

15.8.2.1 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.

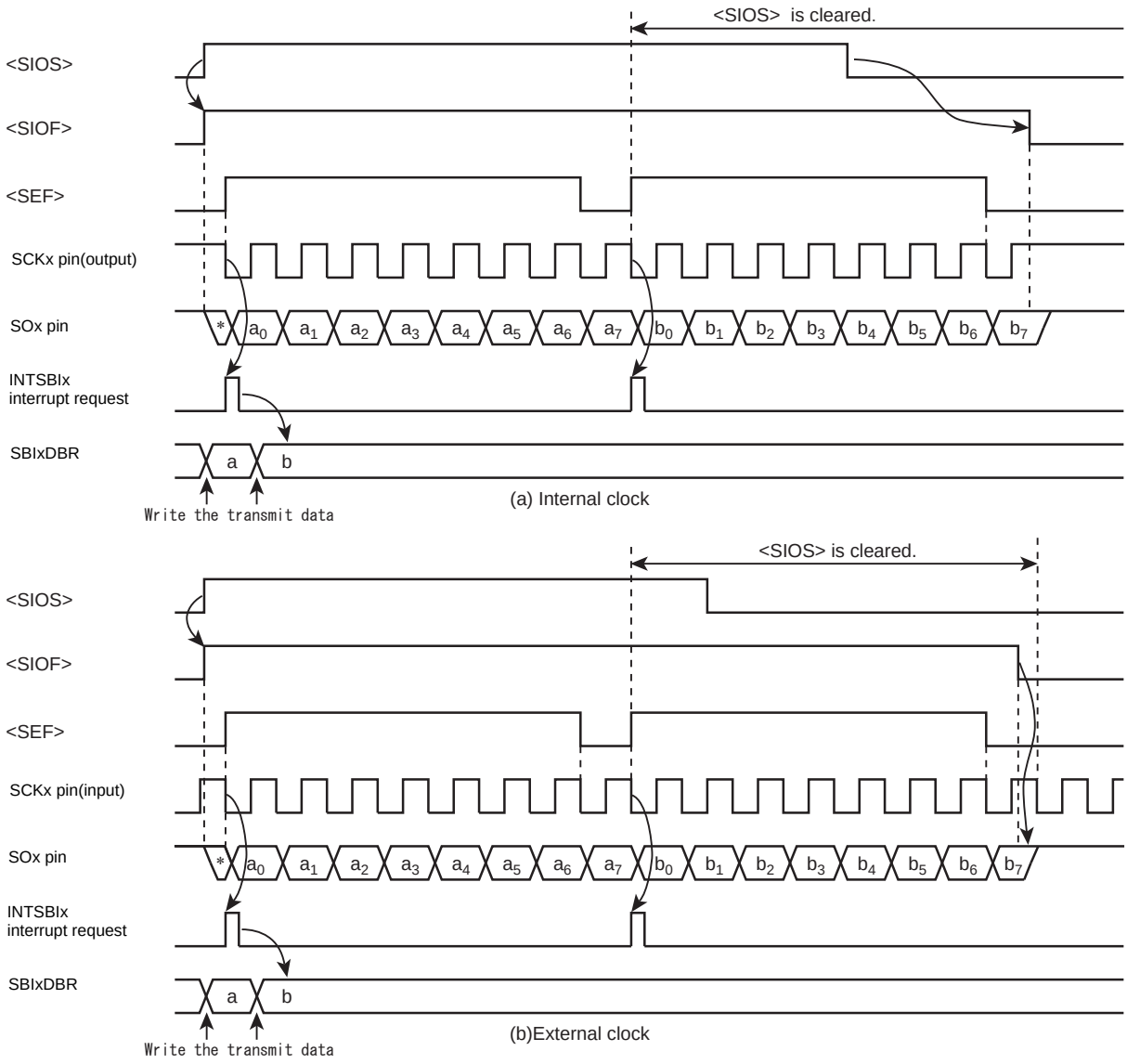


Figure 15-22 Transmit Mode

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.

15.8.2.2 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.

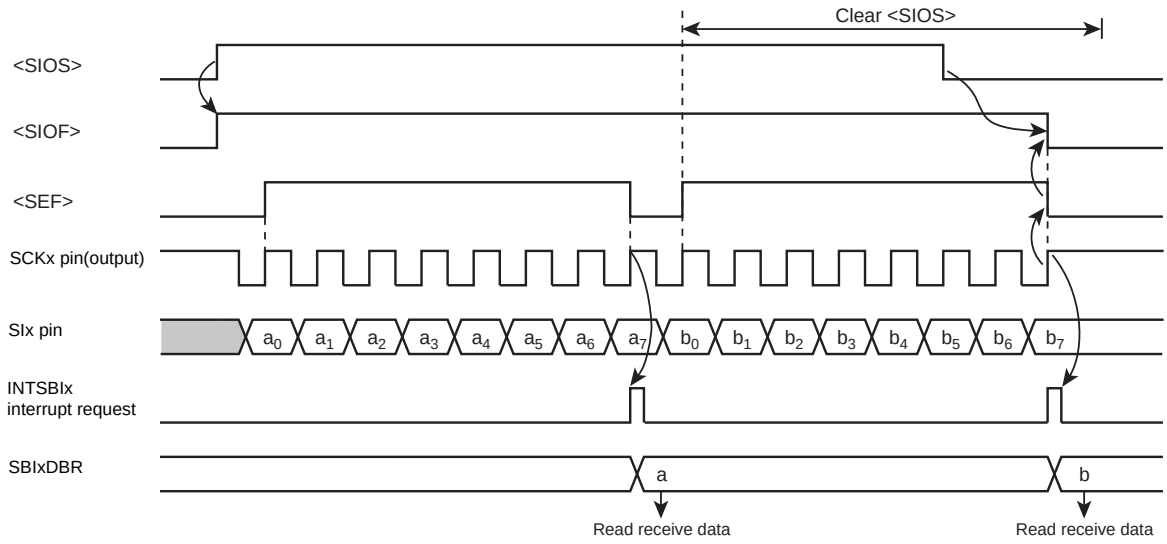


Figure 15-23 Receive Mode (Example: Internal Clock)

15.8.2.3 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.**

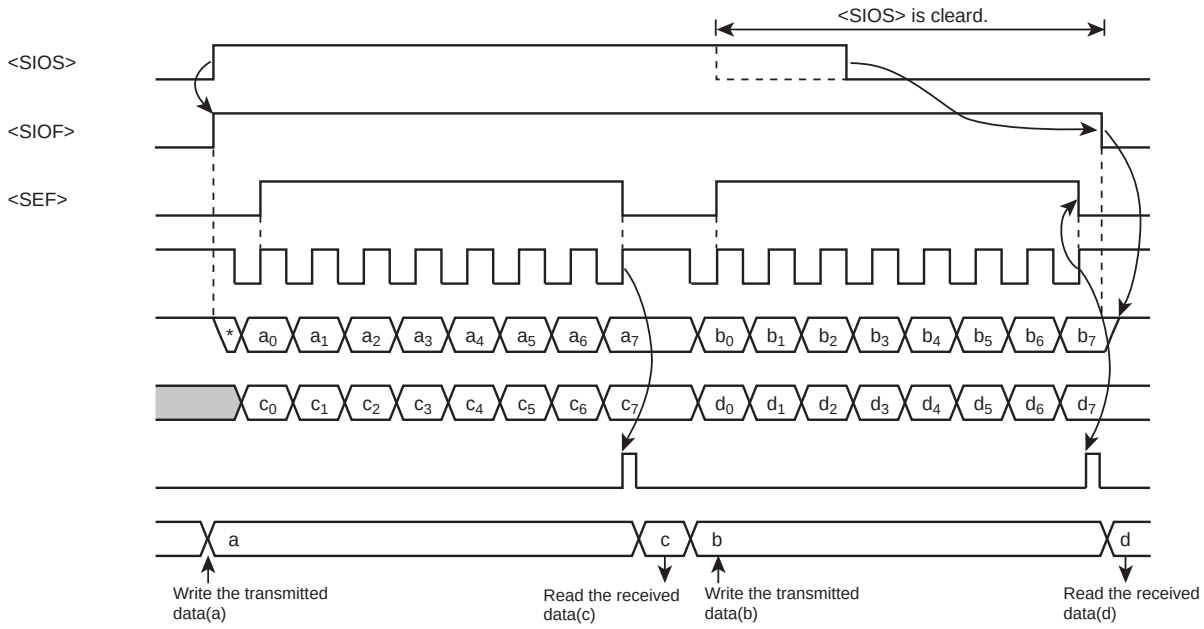


Figure 15-24 Transmit/Receive Mode (Example: Internal Clock)

		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.

15.8.2.4 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.

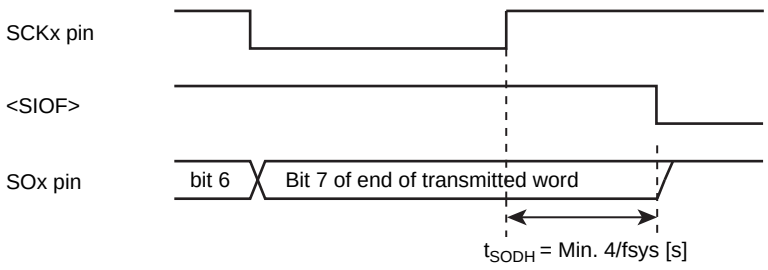


Figure 15-25 Data retention time of the last bit at the end of transmission

16. Analog/Digital Converter (ADC)

16.1 Outline

TMPM365FYXBG 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.

16.2 Configuration

Figure 16-1 shows the block diagram of the AD converter.

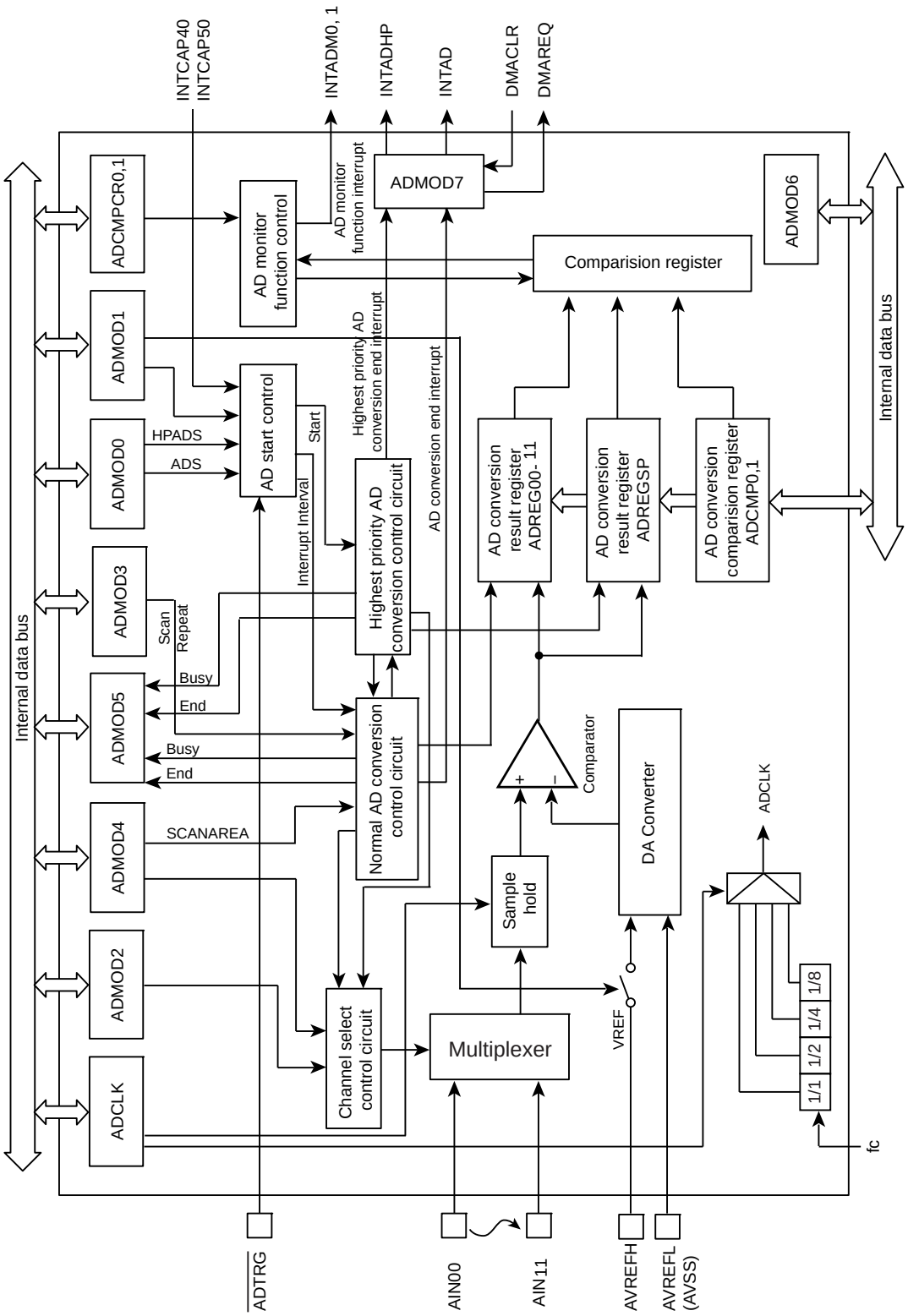


Figure 16-1 AD Converter Block Diagram

16.3 Registers

16.3.1 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.

16.3.2 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	1

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".

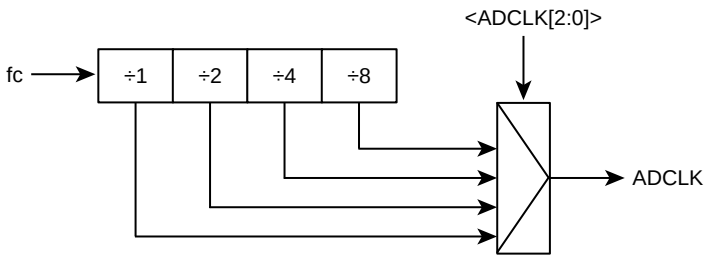


Figure 16-2 AD conversion clock (ADCLK)

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.

16.3.3 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.

16.3.4 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 STOP1 mode.

16.3.5 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 16-1.)
3-0	ADCH[3:0]	R/W	Select analog input channels in normal AD conversion. (See Table 16-1.)

Table 16-1 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

16.3.6 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 16-2.
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 16-2 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.

16.3.7 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 16-3 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.

16.3.8 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 mode.

16.3.9 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. A software reset initializes all the registers except for ADCLK<ADCLK>.

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: When executing the software reset, the bit of ADMOD1<VREFON> would be "1" is a valid.

Note 3: Initialization takes 3μs in case of the software reset.

16.3.10 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	-	-	-	-	-	-	INTADHPD-MA	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

16.3.11 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	-	CMPCOND0	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 : each time　0110 : Over 7 times　1100 : Over 13 times 0001 : Over 2 times　0111 : Over 8 times　1101 : Over 14 times 0010 : Over 3 times　1000 : Over 9 times　1110 : Over 15 times 0011 : Over 4 times　1001 : Over 10 times　1111 : Over 16 times 0100 : Over 5 times　1010 : Over 11 times 0101 : Over 6 times　1011 : Over 12 times
7	CMP0EN	R/W	AD monitor function 0 0: Disable 1: Enable Setting the condition <CMP0EN>="0" (disabled) clears the number of counts.
6	−	R	Read as "0".
5	CMPCOND0	R/W	Sets the condition for judgement count. 0: Serial 1: Cumulative Using serial method, an AD monitor interrupt occurs when the condition set to the <ADBIG0> continue-sand counts up to the number set to the <CMPCNT0>. After exceeding the setting value, an AD monitor interrupt occurs every time when the judgement condition is true. If the condition is different from the condition-set to the <ADBIG0>, the counter is cleared. Using cumulative method, an AD monitor interrupt occurs and the counter is cleared when the conditionset to the <ADBIG0> is accumulated and reaches the number set to the <CMPCNT0>. Even if the condition is different from the value set to the <ADBIG0>, the value of the counter is held.
4	ADBIG0	R/W	Set the AD monitor function interrupt 0 (INTADM0) 0: If the value of the conversion result register is bigger than the comparison register 0, an interrupt is generated. 1: If the value of the conversion result register is smaller than the comparison register 0, an interrupt is generated. 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 1100 to 1111:

Note: AD monitor function is used the fixed repeat mode and the scan repeat mode.

16.3.12 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	-	CMPCOND1	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 : each time 1000 : Over 9 times 0001 : Over 2 times 1001 : Over 10 times 0010 : Over 3 times 1010 : Over 11 times 0011 : Over 4 times 1011 : Over 12 times 0100 : Over 5 times 1100 : Over 13 times 0101 : Over 6 times 1101 : Over 14 times 0110 : Over 7 times 1110 : Over 15 times 0111 : Over 8 times 1111 : Over 16 times
7	CMP1EN	R/W	AD monitor function 1 0: Disable 1: Enable
6	−	R	Read as 0.
5	CMPCOND1	R/W	Sets the condition for judgement count. 0: Serial 1: Cumulative Using serial method, an AD monitor interrupt occurs when the condition set to the <ADBIG0> continue-sands counts up to the number set to the <CMPCNT0>. After exceeding the setting value, an AD monitor interrupt occurs every time when the judgement condition is true. If the condition is different from the condition-set to the <ADBIG0>, the counter is cleared. Using cumulative method, an AD monitor interrupt occurs and the counter is cleared when the conditionset to the <ADBIG0> is accumulated and reaches the number set to the <CMPCNT0>. Even if the condition is different from the value set to the <ADBIG0>, the value of the counter is held.
4	ADBIG1	R/W	Set the AD monitor function interrupt 1(INTADM1) 0: If the value of the conversion result is bigger than the comparison register 1, an interrupt is generated. 1: If the value of the conversion result is smaller than the comparison register 1, an interrupt is generated. 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	AINS1[3:0]	R/W	Select a target conversion result register when using the AD monitor function 1. 0000 : ADREG00 A value other than 0000 : Not to be set.

Note: AD monitor function is used the fixed repeat mode and the scan repeat mode.

16.3.13 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 the comparison value of AD conversion.

Note: To write values into this register, the AD monitor function must be disabled (ADCMPCR0<CMP0EN> ="0").

16.3.14 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 the comparison value of AD conversion.

Note: To write values into this register, the AD monitor function must be disabled (ADCMPCR1<CMP1EN> ="0").

16.3.15 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 16-5, chapter 16.4.5.7.

Note: Do not do the output switching during AD conversion, when other analog / input-output ports are used as output port.

16.3.16 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	-	-	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	−	R	Read as an undefined value.
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. For information about the correlation between the conversion channel and the conversion result register, refer to Table 16-5, chapter 16.4.5.7.

16.4 Description of Operations

16.4.1 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.

By writing "0" to the ADMOD1<RCUT> bit, a switched-on state of AVREFH − AVREFL can be turned into a switched -off state.

16.4.2 AD Conversion Mode

Two types of AD conversion are supported: normal AD conversion and highest-priority AD conversion.

For normal AD conversion, the following four operation modes are supported.

16.4.2.1 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 request (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.

16.4.2.2 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". <HPEOCF> flag is cleared to "0" upon read.

Highest-priority AD conversion activated while highest-priority AD conversion is under way is ignored.

16.4.3 AD monitor function

This function is used for configuring the fixed channel repeat mode and the scan repeat mode.

If ADCMPCR0<CMP0EN> and ADCMPCR1<CMP1EN> are set to "1", AD monitor function is enabled. Two monitor function can be enabled concurrently.

Here is an example of ADCMPCR0 (same as ADCMPCR1).

Analog input for comparison is set to ADACMPCR0/ADBCMPCR0<AINS0[3:0]>. Large or small judgement is set to <ADBIG0>. Conditions of this comparison count is set to <CMPCOND0>. The number of comparison count is set to <CMPCNT0[3:0]>.

Once AD conversion starts, the AD converter checks comparison conditions (smaller/larger than values of the compare register) for each AD conversion. If the result of the comparison meets the settings of <ADBIG0>, the AD converter counts up the counter value.

There are two types of comparison condition: sequential method and cumulative method.

In the sequential method, an AD monitor interrupt (INTADM0) occurs if the condition set to <ADBIG0> continues and reaches the number of counts set to <CMPCNT0[3:0]>. An interrupt occurs without clearing the counter if the result of comparison meets the condition even after reaching the the number of counts set to <CMPCNT0[3:0]>. The value of the counter is zero cleared only when the condition does not meet the set condition of <ADBIG0>.

In the cumulative method, the counter is zero cleared when the total number of times that the condition set to <ADBIG0> meets reaches the number set to <CMPCNT0[3:0]>. An AD monitor interrupt (INTADM0) occurs at this time. The counter value is kept even if the result of the comparison is different from the set value

of the counter. If values of the Conversion Result Register configured by the ADCMPCR0 register are the same as those of the tar get register, the AD converter does not count up. An AD conversion interrupt (INTADM0) does not occur either.

This comparison operation is performed each time a result is stored in a corresponding conversion result register. An interrupt (INTADM0) occurs when conditions meet (including counting). Since the conversion result register assigned to perform the AD monitor function is usually not read by software, the conversion result storage flag ADREG<ADRF> and the overrun flag ADREG<ADOVRF> remain being set. Do not use the conversion result register when you use the AD monitor function.

1. AIN00 input is set to the fixed channel repeat conversion mode and compare values of the AD Conversion Result Compare Register (0x0888).
   - ADMOD3=0x0002: fixed channel repeat conversion

     AD conversion completion interrupt (INTAD) is disabled.
   - ADCMPCR0 =0x0280: compare target channel: AIN00, size determination: larger than a value of the compare register. Compare counting condition: sequential method. AD monitor function: enabled. Size determination count: 3 counts.
   - ADCMP0=0x0888: AD Conversion Result Compare Register (compared value 0x0888)

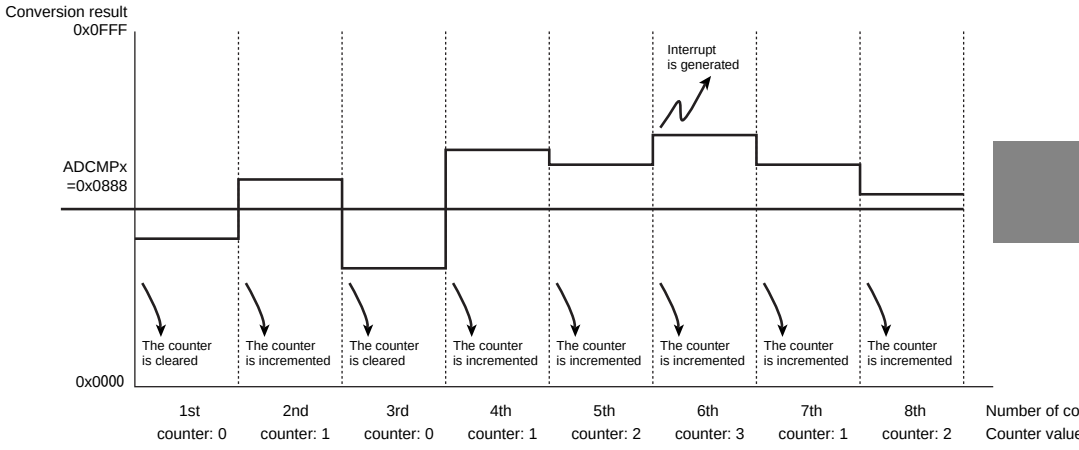


Figure 16-3 AD monitor function (fixed channel repeat and sequential method)

2. AIN00 is set to fixed channel repeat conversion and compare the value of the AD Conversion Result Compare Register (0x0888).
   - ADMOD3=0x0002: fixed channel repeat conversion

     AD conversion completion interrupt (INTAD) is disabled.
   - ADCMPCR0 =0x02A0: compare target channel: AINA00, size determination: larger than a value of the compare register. compare counting condition: cumulative method. AD monitor function: enabled. size determination count: 3 counts
   - ADCMP0=0x0888: AD Conversion Result Compare Register (compared value 0x0888)

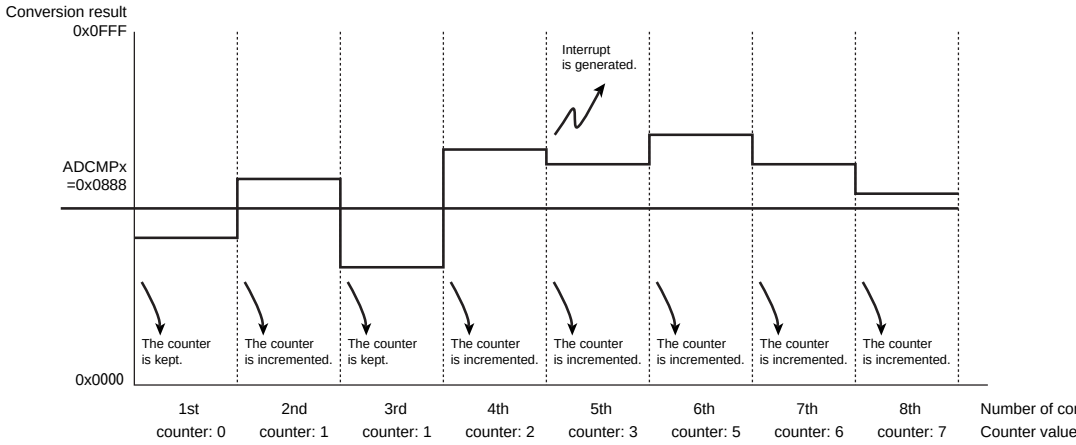


Figure 16-4 AD Monitor Function (fixed channel repeat and cumulative method)

16.4.4 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.

16.4.5 AD Conversion Details

16.4.5.1 Starting AD Conversion

Two types of AD 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 $\overline{\text{ADTRG}}$ 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.

16.4.5.2 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 AD conversion cannot be cleared.

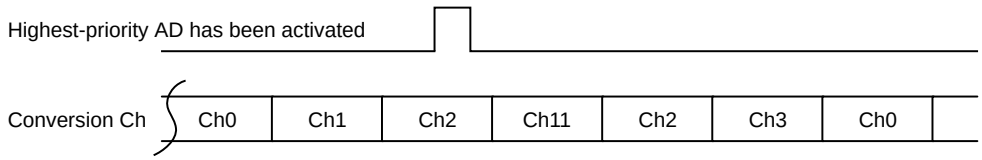
16.4.5.3 Highest-priority AD conversion requests during normal AD conversion

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.



16.4.5.4 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".

16.4.5.5 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)

16.4.5.6 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 SP.

(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.

16.4.5.7 Interrupt generation timings and AD conversion result storage register

Table 16-4 shows a relation in the following three items: AD conversion modes, interrupt generation timings and flag operations. Table 16-5 shows a relation between analog channel inputs and AD conversion result registers.

Table 16-4 Relations in conversion modes, interrupt generation timings and flag operations

Conversion mode		Scan/repeat mode setting (ADMOD3)			Interrupt generation timing	(ADMOD5)		
		<REPEAT>	<SCAN>	<ITM[2:0]>		<EOCF>/ <HPEOCF> set timing (See note1)	<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 16-5 Relations between analog channel inputs and AD conversion result registers

Fixed-channel single mode		Fixed-channel repeat mode		
Channel	Storage register	ADMOD3<ITM[2:0]>		Storage register
AIN00	ADREG00	000	Interrupt by each time AD/C	ADREG00
AIN01	ADREG01	001	Interrupt by each time 2 AD/C	ADREG00 to ADREG01
AIN02	ADREG02	010	Interrupt by each time 3 AD/C	ADREG00 to ADREG02
AIN03	ADREG03	011	Interrupt by each time 4 AD/C	ADREG00 to ADREG03
AIN04	ADREG04	100	Interrupt by each time 5 AD/C	ADREG00 to ADREG04

Table 16-5 Relations between analog channel inputs and AD conversion result registers

Fixed-channel single mode	
Channel	Storage register
AIN05	ADREG05
AIN06	ADREG06
AIN07	ADREG07
AIN08	ADREG08
AIN09	ADREG09
AIN10	ADREG10
AIN11	ADREG11

Fixed-channel repeat mode		
ADMOD3<ITM[2:0]>		Storage register
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 (ex. ADREG03 to depend on the scan channel range.)		
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

Notes on designing for AD converter inputs

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.

Notes of the use of the AD converter

The result value of AD conversion may vary depending on the fluctuation of the supply voltage, or may be affected by noise.

When using analog input pins and ports alternately, do not read and write ports during conversion because the conversion accuracy may be reduced. Also the conversion accuracy may be reduced if the output ports current fluctuate during AD conversion.

Please take counteractive measures with the program such as averaging the AD conversion results.

17. Flash Memory Operation

This section describes the hardware configuration and operation of Flash memory. In this section, "1-word" means 32 bits.

17.1 Features

17.1.1 Memory Size and Configuration

Table 17-1 and Figure 17-1 show a built-in memory size and configuration of TMPM365FYXBG.

Table 17-1 Memory size and configuration

Memory size	Block configuration				# of words per page	# of pages	Write time		Erase time	
	128 KB	64 KB	32 KB	16 KB			1 page	Total area	Block erase	Chip erase
256 KB	−	3	1	2	64	1024	1.25ms	1.28 sec	0.1sec	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 used by a user.

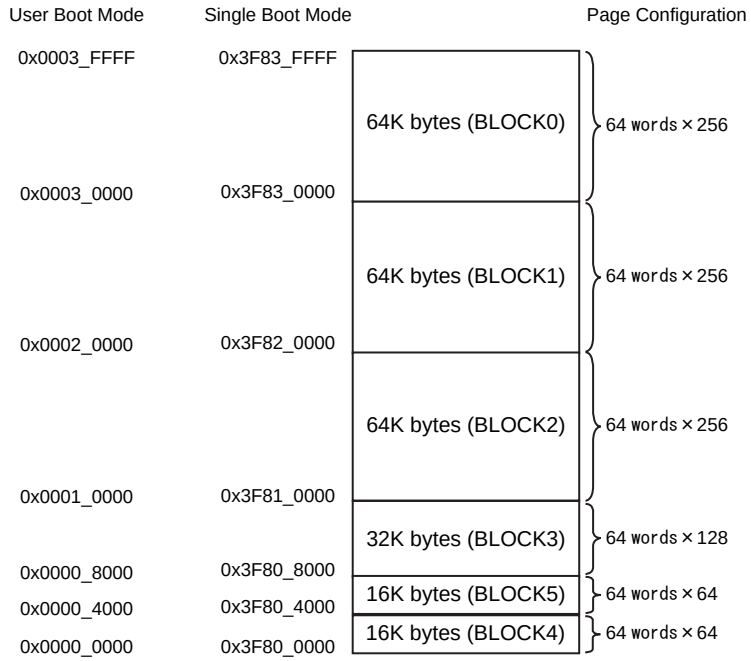


Figure 17-1 Block configuration

Flash memory configuration units are described as "block" and "page".

- Page

  One page is 64 words. Same address [31:8] is used in a page. First address of the group is [7:0] = 0x00 and the last address of the group is [7:0] = 0xFF.

- Block

  There is size of 16 KB, 32 KB and 64 KB in a block, and a flash memory consists of blocks of some different sizes.

Write operation is performed per page. The write time per page is 1.25ms. (Typ.)

Erase is performed per block (auto block erase command use) or performed on entire flash memory (use of auto chip erase command). Erase time varies on commands. If auto block command is used, the erase time will be 0.1 sec per block (Typ.). If the auto chip erase command is used to erase entire area, the time will be 0.2 sec (Typ.).

In addition, the protect function can be used per block. For detail of the protect function, refer to "17.1.5 Protect/Security Function".

17.1.2 Function

Flash memory built-in this device is generally compliant with the JEDEC standards except for some specific functions. Therefore, if a user is currently using a flash memory as an external memory, it is easy to implement the functions into this device. Furthermore, to provide easy write or erase operation, this product contains a dedicated circuit to perform write or chip erase automatically.

JEDEC compliant functions	Modified, added, or deleted functions
・Automatic programming ・Automatic chip erase ・Automatic block erase ・Data polling/toggle bit	<Modified> Block write/erase protect (only software protection is supported) <Deleted> Erase resume - suspend function

17.1.3 Operation Mode

17.1.3.1 Mode Description

This device provides the single chip mode and single boot mode. The single chip mode contains the normal mode and user boot mode. Figure 17-2 shows the mode transition.

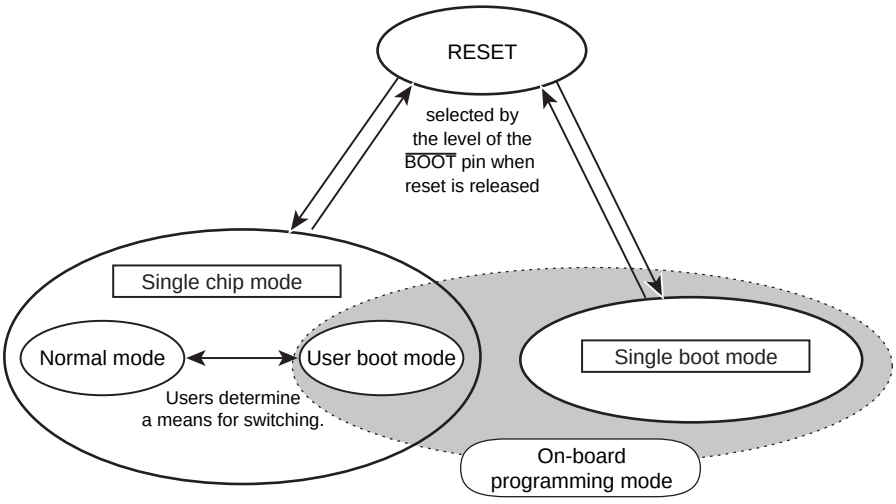


Figure 17-2 Mode transition

(1) Single chip mode

The single chip mode is a mode where the device can boot-up from Flash memory after reset. The mode contains two sub-modes in below.

- Normal mode

  The mode where user application program is executed.
- User boot mode

  The mode where flash memory is re-programmed on the user's set.

Users can switch the normal mode to user boot mode freely. For example, a user can set if PA0 of port A is "1", the mode is the normal mode. If PA0 of port A is "0", the mode is the user boot mode. The user must prepare a routine program in the application program to determine the switching.

(2) Single boot mode

The mode where flash memory can boot-up from the built-in BOOT ROM (Mask ROM) after reset.

The BOOT ROM contains the algorithm that can rewrite Flash memory via serial port of this device on the user's set. With connecting the serial port to external host, data transfer is performed in above-mentioned protocol and re-programed Flash memory.

(3) On-board programming mode

The user boot mode and single boot mode are the modes where flash memory can be re-programmable on the user's set. These two modes are called "on-board programming mode".

17.1.3.2 Mode Determination

Either the single chip or single boot operation mode can be selected by the level of the $\overline{\text{BOOT}}$ pin when reset is released.

Table 17-2 Operation mode setting

Operation mode	Pin	
	$\overline{\text{RESET}}$	$\overline{\text{BOOT}}$
Single chip mode	0 → 1	1
Single boot mode	0 → 1	0

17.1.4 Memory Map

Figure 17-3 shows a comparison of the memory map in the single chip mode and single boot mode. In the single boot mode, built-in Flash memory is mapped to 0x3F80_0000 and subsequent addresses, and the built-in BOOT ROM is mapped to 0x0000_0000 through 0x0000_0FFF.

Flash memory and RAM addresses are shown below.

FLASH size	RAM size	FLASH address	RAM address
256 KB	24 KB	0x0000_0000 to 0x0003_FFFF(single mode) 0x3F80_0000 to 0x3F83_FFFF(single boot mode)	0x2000_0000 to 0x2000_5FFF

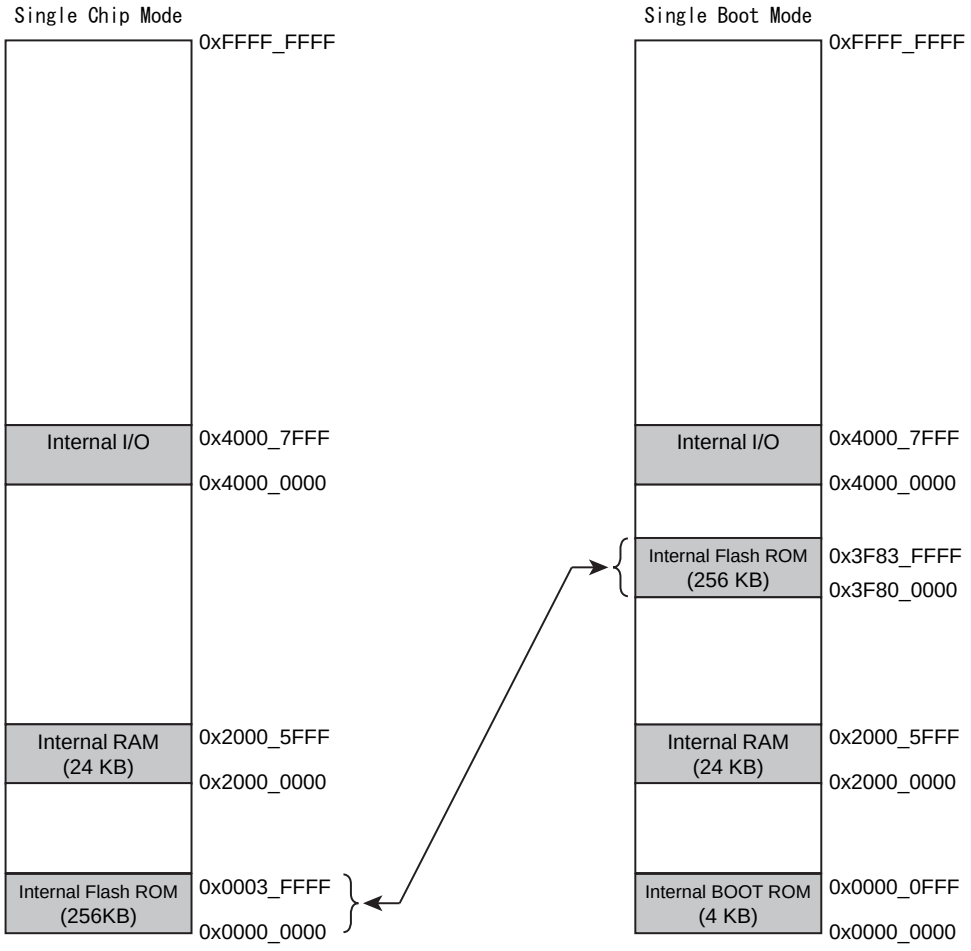


Figure 17-3 Comparison of memory map

17.1.5 Protect/Security Function

This device has the protect and security functions for Flash memory.

1. Protect function

   The write/erase operation can be inhibited per block.

2. Security function

   The read operation from a flash writer can be inhibited.

   Usage restrictions on debug functions

17.1.5.1 Protect Function

This function inhibits the write/erase operation per block.

To enable the protect function, a protect bit corresponding to a block is set to "1" using the protect bit program command. If a protect bit is set to "0" using the protect bit erase command, a block protect can be canceled. The protect bit can be monitored with FCFLCS<BLPRO[3:0]>.

A program of protect bit can be programmed by 1-bit unit and can be erased by 4-bit unit. For detail of programming/erasing of protect bits, refer to "17.2.5 Command Description".

17.1.5.2 Security Function

Table 17-3 shows operations when the security function is enabled.

Table 17-3 Operations when the security function is enabled.

Item	Description
Read flash memory	CPU can read flash memory.
Debug port	JTAG, serial wire or trace communication is disabled.
Command execution to Flash memory	Command write to flash memory is not accepted. If a user tries to erase a protect bit, chip erase is executed and all protect bits are erased.

The security function is enabled under the following conditions;

1. FCSECBIT<SECBIT> is set to "1".
2. All protect bits (FCFLCS<BLPRO>) are set to "1".

FCSECBIT<SECBIT> is set to "1" by the cold reset. Rewriting of FCSECBIT <SECBIT> is described in below.

Note: Use a 32-bit transfer instruction when the following writing operations, item1 and 2.

1. Write the specified code (0xa74a9d23) to FCSECBIT

2. Write data within 16 clocks after the operation of item 1.

17.1.6 Register

17.1.6.1 Register List

Base Address = 0x41FF_F000

Register name		Address(Base+)
Security bit register	FCSECBIT	0x0010
Flash control register	FCFLCS	0x0014

Note: Do not access the address of 0x41FF_F000 to 0x41 FF_FFFF other than the above-mentioned address list.

17.1.6.2 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	(Note 2)	(Note 2)	(Note 2)	(Note 2)	(Note 2)	(Note 2)
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	RDY/BSY
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-22	−	R	Read as "0".
19-21	BLPRO3- BLPRO0	R	Protection status of Block5 to 0 0: Not protected 1: Protected Protect bit values correspond to protect status of each block. If corresponding bit indicates "1", corresponding block is in the protection status. A block in the protection status cannot be re-programmable.
15-1	−	R	Read as "0".
0	RDY/BSY	R	Ready/Busy (Note 1) 0: Busy (during auto operation) 1: Ready (auto operation ends) This bit is a function bit to monitor flash memory from CPU. While flash memory is in auto operation, this bit outputs "0" to indicate that flash memory is busy. Once auto operation is finished, this bit becomes ready state and outputs "1". Then next command is accepted. If a result of auto operation is failed, this bit outputs "0" continuously. The bit returns to "1" by hardware reset.

Note 1: Make sure that flash memory is ready before commands are issued. If a command is issued during busy, not only the command is not sent but also subsequent commands may not be accepted. In that case, use hardware reset to return. Hardware reset needs 0.5 μs or more reset period regardless of system clock. At this time, it takes approximately 2 ms until enabling to read after reset.

Note 2: A value will correspond to the protection status.

17.1.6.3 FCSECBIT (Security bit register)

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

Bit	Bit Symbol	Type	Function
31-1	−	R	Read as "0".
0	SECBIT	R/W	Security bit 0: Security function setting is disabled. 1: Security function setting is enabled.

Note: This register is initialized by cold reset.

17.2 Detail of Flash Memory

In on-board programming, the CPU executes commands for reprogramming or erasing Flash memory. This reprogramming/erase control program should be prepared by the user beforehand. Since Flash memory content cannot be read while Flash memory is being written or erased, it is necessary to run the reprogram/erase control program on the built-in RAM. Do not generate interrupt/fault except reset to avoid abnormal program termination.

17.2.1 Function

Flash memory is generally compliant with the JEDEC standards except for some specific functions. However; a method of address designation of operation command is different from standard commands.

If write/erase operation is executed, commands are input to flash memory using 32-bit (1-word) store instruction command. After command input, write or erase operation is automatically executed in inside.

Table 17-4 Flash memory function

Main function	Description
Automatic page program	Writes data automatically.
Automatic chip erase	Erases the entire area of Flash memory automatically.
Automatic block erase	Erases a selected block automatically.
Write/erase protect	The write or erase operation can be individually inhibited for each block.

17.2.2 Operation Mode of Flash Memory

Flash memory provides mainly two types of operation modes;

- The mode to read memory data (Read mode)
- The mode to erase or rewrite memory data automatically (Automatic operation mode)

After power-on, after rest or after automatic operation mode is finished normally, Flash memory becomes read mode. Instruction stored in Flash memory or data read is executed in the read mode.

If commands is input during the read mode, the operation mode becomes the automatic operation. If the command process is normally finished, the operation mode returns to the read mode except the ID-Read command. During the automatic operation, data read and instruction execution stored in Flash memory cannot be performed.

If command process is abnormally finished then the operation mode should forcibly return to read mode. In this case, use the read command, read/reset command or hardware reset.

17.2.3 Hardware Reset

A hardware reset means a cold reset or warm reset to use returning to the read mode when the automatic programming/erase operation is forced cancel, or automatic operation abnormally ends.

If the hardware reset occurs during the automatic operation, Flash memory stops the automatic operation and returns to the read mode. If a hardware reset is generated during Flash memory automatic program/erase operation, the hardware reset needs 0.5 μs or more reset period regardless of system clock. At this time, it takes approximately 2 ms until enabling to read after reset . Note that if a hardware reset occurs during the automatic operation, data write operation is not executed properly. Set write operation again.

For detail of the reset operation, refer to "Reset". After a given reset input, CPU will read the reset vector data and then starts the routine after reset.

17.2.4 How to Execute Command

The command execution is performed by writing command sequences to Flash memory with a store instruction. Flash memory executes each automatic operation command according to the combination of input addresses and data. For detail of the command execution, refer to "17.2.5 Command Description".

An execution of store instruction to the Flash memory is called "bus write cycle". Each command consists of some bus write cycles. In Flash memory, when address and data of bus write cycle are performed in the specified order, the automatic command operation is performed. When the cycle is performed in non-specified order, Flash memory stops command execution and returns to the read mode.

If you cancel the command during the command sequence or input a different command sequence, execute the read command or read/reset command. Then Flash memory stops command execution and returns to the read mode. The read command and read/reset command are called "software reset".

When write command sequence ends, the automatic operation starts and FCFLCS<RDY/BSY> is set to "0". When the automatic operation normally ends, FCFLCS<RDY/BSY> = "1" is set and Flash memory returns to the read mode.

New command sequences are not accepted during the automatic operation. If you want to stop the command operation, use a hardware reset. In case that the automatic operation abnormally ends (FCFLCS<RDY/BSY> remains "0"), Flash memory remains locked and will not return to the read mode. To returns to the read mode, use a hardware reset. If the hardware reset stops the command operation, commands are not normally executed.

Notes on the command execution;

1. To recognize command, command sequencer need to be in the read mode before command starting. Confirm FCFLCS<RDY/BSY> = 1 is set prior to the first bus write cycle of each command. Consecutively, it is recommended that the read command is executed.
2. Execute each command sequence from outside of Flash memory.
3. Execute sequentially each bus write cycle by data transfer instruction in 1-word (32-bit).
4. Do not access Flash memory during the each command sequence. Do not generate any interrupt or fault except reset.
5. Upon issuing a command, if any address or data is incorrectly written, make sure to return to the read mode by using software reset.

17.2.5 Command Description

This section explains each command content. For detail of specific command sequences, refer to "17.2.6 Command Sequence".

17.2.5.1 Automatic Page Program

(1) Operation Description

The automatic page program writes data per page. When the program writes data to multiple pages, a page command need to be executed in page by page. Writing across pages is not possible.

Writing to Flash memory means that data cell of "1" becomes data of "0". It is not possible to become data cell of "1" from data of "0". To become data cell of "1" from "0", the erase operation is required.

The automatic page program is allowed only once to each page already erased. Either data cell of "1" or "0" cannot be written data twice or more. If rewriting to a page that has already been written once, the automatic page program is needed to be set again after the automatic block erase or automatic chip erase command is executed.

Note 1: Page program execution to the same page twice or more without erasing operation may damage the device.

Note 2: Writing to the protected block is not possible.

(2) How to Set

The 1st to 3rd bus write cycles indicate the automatic page program command.

In the 4th bus write cycle, the first address and data of the page are written. On and after 5th bus cycle, one page data will be written sequentially. Data is written in 1-word unit (32-bit).

If a part of the page is written, set "0xFFFFFFFF" as data, which means not required to write, for entire one page.

No automatic verify operation is performed internally in the device. So, be sure to read the data programmed to confirm that it has been correctly written.

If the automatic page program is abnormally terminated, that page has been failed to write. It is recommended not to use the device or not to use the block including the failed address.

17.2.5.2 Automatic Chip Erase

(1) Operation Description

The automatic chip erase is executed to the memory cell of all addresses. If protected blocks are contained, these blocks will not be erased. If all blocks are protected, the automatic chip erase operation will not performed and will return to the read mode after a command sequence is input.

(2) How to Set

The 1st to 6th bus write cycles indicate the automatic chip erase command. After the command sequence is input, the automatic chip erase operation starts.

No automatic verify operation is performed internally in the device. So, be sure to read the data to confirm that it has been correctly erased.

If the automatic chip erase is abnormally terminated, specify the failed block with the block erase function. On and after it is recommended not to use the failed block.

17.2.5.3 Automatic Block Erase

(1) Operation Description

The automatic erase command performs erase operation to the specified block. If the specified block is protected, erase operation is not executed.

(2) How to Set

The 1st to 5th bus write cycles indicate the automatic block erase command. In the 6th bus write cycle, the block to be erased is specified. After the command sequence is input, the automatic block erase operation starts.

No automatic verify operation is performed internally in the device. So, be sure to read the data to confirm that it has been correctly erased.

If the automatic chip erase is abnormally terminated, on and after it is recommended not to use the faild block.

17.2.5.4 Automatic Protect Bit Program

(1) Operation Description

The automatic protect bit program writes "1" to a protect bit at a time. To set "0" to a protect bit, use the automatic protect bit erase command.

For detail of the protect function, refer to "17.1.5 Protect/Security Function".

(2) How to Set

The 1st to 6th bus write cycles indicate the automatic protect bit program command. In the 7th bus write cycle, the protect bit to be written is specified. After the command sequence is input, the automatic protect bit program starts. Check whether write operation is normally terminated with FCFLCS<BLPRO>.

17.2.5.5 Auto Protect Bit Erase

(1) Operation Description

The automatic protect bit erase command operation depends on the security status. For detail of security status, refer to "17.1.5 Protect/Security Function".

- Non-security status

  Clear the specified protect bit to "0". Protect bit erase is performed in 4-bit unit.
- Security status

  Erase all protect bits after all addresses of Flash memory are erased.

(2) How to Set

The 1st to 6th bus write cycles indicate the automatic protect bit erase command. In the 7th bus write cycle, the protect bit to be erased is specified. After the command sequence is input, the automatic protect bit erase operation starts.

In the non-security status, specified protect bit is erased. Check whether erase operation is normally terminated with FCFLCS<BLPRO>.

In the security status, all addresses and all protect of Flash memory bits are erased. Confirm if data and protect bits are erased normally. If necessary, execute the automatic protect bit erase, automatic chip erase or automatic block erase.

All cases are the same as other commands, FCFLCS<RDY/BSY> becomes "0" during the automatic protect bit erase command operation. After the operation is complete, FCFLCS<RDY/BSY> becomes "1" and Flash memory will return to the read mode. To abort the operation, a hardware reset is required.

17.2.5.6 ID-Read

(1) Operation Description

The ID-Read command can read information including Flash memory type and three types of codes such as a maker code, device code and macro code.

(2) How to Set

The 1st to 3rd bus write cycles indicate the ID-Read command. In the 4th bus write cycle, the code to be read is specified. After the 4th bus write cycle, read operation in the arbitrary flash area acquires codes.

The ID-Read can be executed successively. The 4th bus write cycle and reading ID value can be executed repeatedly.

The ID-Read command does not automatically return to the read mode. To return to the read mode, execute the read command, read/reset command or hardware reset.

17.2.5.7 Read Command and Read/reset Command (Software Reset)

(1) Operation Description

A command to return Flash memory to the read mode.

When the ID-Read command is executed, macro stops at the current status without automatically return to the read mode. To return to the read mode from this situation, use the read command or read/reset command. It is also used to cancel the command when commands are input to the middle.

(2) How to Set

The 1st bus cycle indicates the read command. The 1st to 3rd bus write cycles indicate the read/reset command. After either command sequence is executed, Flash memory returns to the read mode.

17.2.6 Command Sequence

17.2.6.1 Command Sequence List

Table 17-5 shows addresses and data of bus write cycle in each command.

All command cycles except the 5th bus cycle of ID-Read command are "bus write cycles". A bus write cycle is performed by 32-bit (1-word) data transfer instruction. (Following table shows only lower 8 bits of data.)

For detail of addresses, refer to Table 17-6. Use below values to "command" described in a column of Addr[15:9] in the Table 17-6.

Note 1) Always set to "0" to the address bit [1:0].

Note 2) Set below values to the address bit [19] according to Flash memory size.
Memory size is 1MB or less : Always set to "0"
Memory size is over 1MB : If bus write to 1MB area or less, the bit is set to "0".
If bus write to over 1MB area, the bit is set to "1".

Table 17-5 Command Sequence

Command	1st bus cycle	2nd bus cycle	3rd bus cycle	4th bus cycle	5th bus cycle	6th bus cycle	7th bus cycle
	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.
	Data	Data	Data	Data	Data	Data	Data
Read	0xXX	–	–	–	–	–	–
	0xF0	–	–	–	–	–	–
Read/reset	0x54XX	0xAAXX	0x54XX	–	–	–	–
	0xAA	0x55	0xF0	–	–	–	–
ID-Read	0x54XX	0xAAXX	0x54XX	IA	0xXX	–	–
	0xAA	0x55	0x90	0x00	ID	–	–
Automatic page program	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 protect bit program	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x9A	0xAA	0x55	0x9A	0x9A
Automatic protect bit erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	0xXX
	0xAA	0x55	0x6A	0xAA	0x55	0x6A	0x6A

Supplementary explanation

- IA: IDAddress
- ID: ID data
- PA: Program page address
- PD: Program data (32-bit data)

  After the 4th bus cycle, input data in the order of the addresses per page
- BA: Block address (see Table 17-7)
- PBA: Protect bit address (see Table 17-8, Table 1-9)

17.2.6.2 Address Bit Configuration in the Bus Cycle

Table 17-6 is used in conjunction with "Table 17-5 Command Sequence".

Set the address setting according to the normal bus write cycle address configuration from the first bus cycle.

Table 17-6 Address bit configuration in the bus write cycle

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]

Normal Command	Normal bus write cycle address configuration				
	Flash area	"0" is recommended.	Command		Addr[1:0] = "0" (fixed) Other bits = "0" (recommended)
ID-READ	IA: ID address (Setting of the 4th bus write cycle address for ID-READ)				
	Flash area	"0" is recommended.	ID Address	Addr[1:0] = "0" (fixed) Other bits= "0" (recommended)	
Block erase	BA: Block address(Setting of the 6th bus write cycle address for block erase)				
	Block address (Table 17-7)			Addr[1:0] = "0" (fixed) Other bits= "0" (recommended)	
Automatic page program	PA: Program page address (Setting of the 4th bus write cycle address for page program)				
	Page address				Addr[1:0] = "0" (fixed) Other bits= "0" (recommended)
Protect bit program	PBA: Protect bit address (Setting of the 7th bus write cycle address for protect bit program)				
	Flash area	Protect bit selection (Table 17-8)	Fix to "0"	Protect bit selection (Table 17-8)	Addr[1:0] = "0" (fixed) Other bits= "0" (recommended)
Protect bit erase	PBA: Protect bit address (Setting of the 7th bus write cycle address for protect bit erase)				
	Flash area	Protect bit selection (Table 17-9)	Fix to "0"	Addr[1:0] = "0" (fixed) Other bits= "0" (recommended)	

17.2.6.3 Block Address(BA)

Table 17-7 shows block addresses. Specify any address included in the block to be erased in the 6th bus write cycle of the automatic block erase command.

Table 17-7 Block address

Block	Address (User boot mode)	Address (Single boot mode)	Size (Kbyte)
4	0x0000_0000 to 0x0000_3FFF	0x3F80_0000 to 0x3F80_3FFF	16
5	0x0000_4000 to 0x0000_7FFF	0x3F80_4000 to 0x3F80_7FFF	16
3	0x0000_8000 to 0x0000_FFFF	0x3F80_8000 to 0x3F80_FFFF	32
2	0x0001_0000 to 0x0001_FFFF	0x3F81_0000 to 0x3F81_FFFF	64
1	0x0002_0000 to 0x0002_FFFF	0x3F82_0000 to 0x3F82_FFFF	64
0	0x0003_0000 to 0x0003_FFFF	0x3F83_0000 to 0x3F83_FFFF	64

17.2.6.4 How to Specify Protect Bit (PBA)

The protect bit is specified in 1-bit unit in programming and in 4-bit unit in erasing.

Table 17-8 and Table 17-9 shows a protect bit selection table of the automatic protect bit program. The column of address example indicates an address described in upper side is used in the single mode and the lower side is used in the single boot mode.

Four protect bits are erased by the automatic protect bit erase command in all.

Table 17-8 Protect bit program address

Block	Protect bit	Address of 7th bus write cycle							Address example [31:0]
		Address [18]	Address [17]	Address [16]	Address [15:11]	Address [10]	Address [9]	Address [8]	
Block0	<BLPRO[0]>	0	0	Fix to "0"			0	0	0x0000_0000 0x3F80_0000
Block1	<BLPRO[1]>	0	0				0	1	0x0000_0100 0x3F80_0100
Block2	<BLPRO[2]>	0	0				1	0	0x0000_0200 0x3F80_0200
Block3	<BLPRO[3]>	0	0				1	1	0x0000_0300 0x3F80_0300
Block4	<BLPRO[4]>	0	1				0	0	0x0002_0000 0x3F82_0000
Block5	<BLPRO[5]>	0	1				0	1	0x0002_0100 0x3F82_0100

Table 17-9 Protect bit erase address

Block	Protect bit	Address of 7th bus write cycle		Address example [31:0]
		Address [18]	Address [17]	
Block0 - 3	<BLPRO[0:3]>	0	0	0x0000_0000 0x3F80_0000
Block4 - 5	<BLPRO[4:5]>	0	0	0x0002_0000 0x3F82_0000

17.2.6.5 ID-Read Code (IA, ID)

Table 17-10 shows how to specify a code and the content using ID-Read command.

The column of address example indicates an address described in the upper side is used in the single mode and the lower side is used in the single boot mode

Table 17-10 ID-Read Command codes and contents

Code	ID[7:0]	IA[13:12]	Address Example [31:0]
Manufacture code	0x98	0y00	0x0000_0000 0x3F80_0000
Device code	0x5A	0y01	0x0000_4000 0x3F80_4000
–	Reserved	0y10	–
Macro code	0x13	0y11	0x0000_C000 0x3F80_C000

17.2.6.6 Example of Command Sequence

(1) Single mode

Command		Bus cycle						
		1	2	3	4	5	6	7
Read	Address	0x0000_0000	–	–	–	–	–	–
	Data	0x0000_00F0	–	–	–	–	–	–
Read/reset	Address	0x0000_5400	0x0000_0AA0	0x0000_5400	–	–	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_00F0	–	–	–	–
ID-Read	Address	0x0000_5400	0x0000_0AA0	0x0000_5400	IA	0x0000_00XX	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0090	0x0000_0000	ID	–	–
Automatic page program	Address	0x0000_5400	0x0000_0AA0	0x0000_5400	PA	In the following cycles, write addresses and data successively per page.		
	Data	0x0000_00AA	0x0000_0055	0x0000_00A0	PD			
Automatic chip erase	Address	0x0000_5400	0x0000_0AA0	0x0000_5400	0x0000_5400	0x0000_0AA0	0x0000_5400	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0010	–
Automatic block erase	Address	0x0000_5400	0x0000_0AA0	0x0000_5400	0x0000_5400	0x0000_0AA0	BA	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0030	–
Automatic protect bit program	Address	0x0000_5400	0x0000_0AA0	0x0000_5400	0x0000_5400	0x0000_0AA0	0x0000_5400	PBA
	Data	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_009A
Automatic protect bit erase	Address	0x0000_5400	0x0000_0AA0	0x0000_5400	0x0000_5400	0x0000_0AA0	0x0000_5400	PBA
	Data	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_006A

(2) Data single boot mode

Command		Bus cycle						
		1	2	3	4	5	6	7
Read	Address	0x3F80_0000	–	–	–	–	–	–
	Data	0x0000_00F0	–	–	–	–	–	–
Read/reset	Address	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	–	–	–	–
	Data	0x0000_00AA	0x3F80_0055	0x3F80_00F0	–	–	–	–
ID-Read	Address	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	IA	0x0000_00XX	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0090	0x0000_0000	ID	–	–
Automatic page program	Address	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	PA	In the following cycles, write addresses and data successively per page.		
	Data	0x0000_00AA	0x0000_0055	0x0000_00A0	PD			
Automatic chip erase	Address	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0010	–
Automatic block erase	Address	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	0x3F80_5400	0x3F80_0AA0	BA	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0030	–
Automatic protect bit program	Address	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	PBA
	Data	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_009A
Automatic protect bit erase	Address	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	PBA
	Data	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_006A

17.2.7 Flowchart

17.2.7.1 Automatic Program

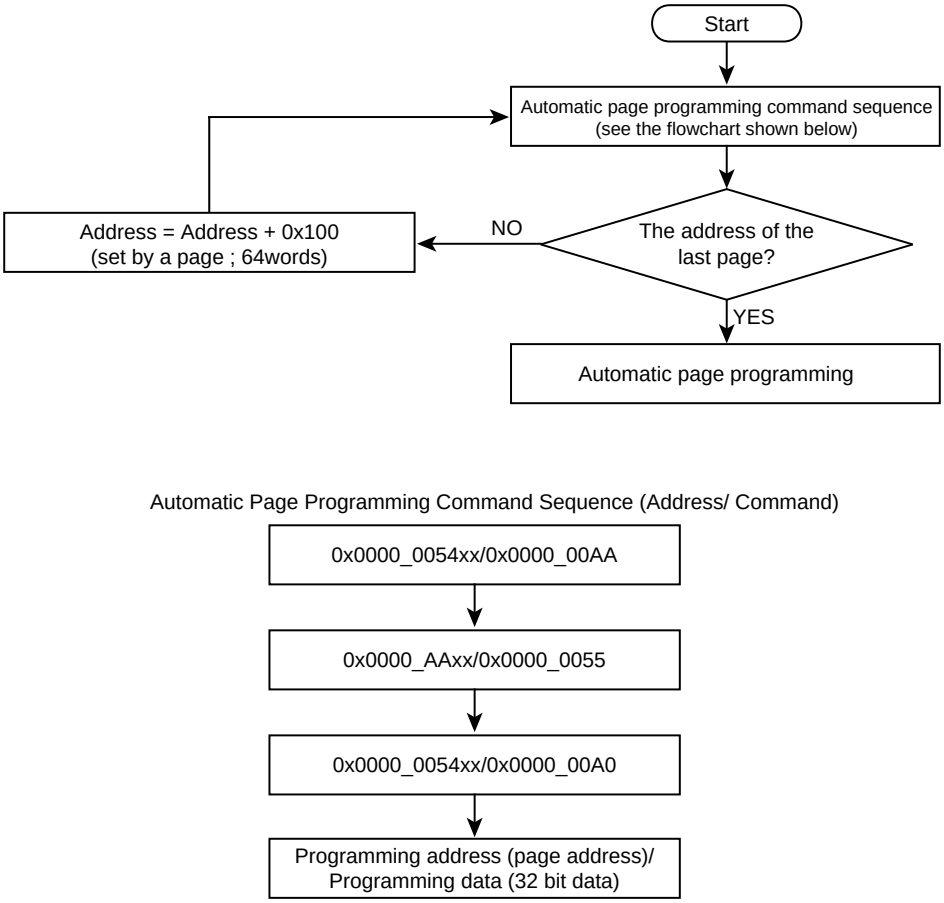


Figure 17-4 Flowchart of automatic program

17.2.7.2 Automatic Erase

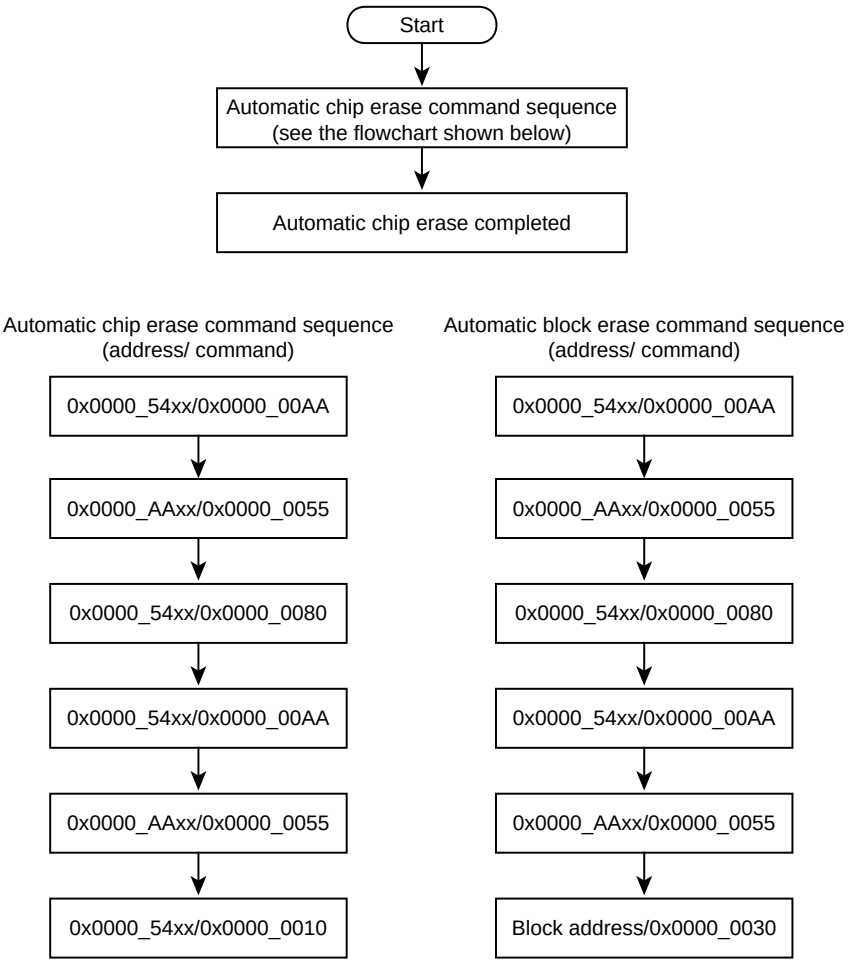


Figure 17-5 Flowchart of automatic erase

17.3 How to Reprogram Flash using Single Boot Mode

The single boot mode utilizes a program contained in built-in BOOT ROM for reprogrammig Flash memory. In this mode, BOOT ROM is mapped to the area containg interrupt vector tables and Flash memory is mapped to another address area other than BOOT ROM area.

In the boot mode, Flash memory is reprogrammed using serial command/data transfer. With connecting serial channel (SIO/UART) of this device to the external host, a reprogramming program is copied from the external host to the built-in RAM. A reprogramming routine in the RAM is executed to reprogram Flash memory. For details of communication with host, follow the protocol described later.

Even in the single boot mode, do not generate interrupt/fault except reset to avoid abnormal program termination.

To secure the contents of Flash memory in the single chip mode (normal operation mode), once re-programming is complete, it is recommended to protect relevant flash blocks against accidental erasure during subsequent single chip operations.

17.3.1 Mode Setting

In order to execute the on-board programming, this device is booted-up in the single boot mode. Below setting is for the single boot mode setting.

$\overline{\text{BOOT}}$ = 0
$\overline{\text{RESET}}$ = 0 → 1

While $\overline{\text{BOOT}}$ pin is set to the above in advance, set $\overline{\text{RESET}}$ pin to "0". Then release $\overline{\text{RESET}}$ pin, the device will boot-up in the single boot mode.

17.3.2 Interface Specification

This section describes SIO/UART communication format in the single boot mode. The serial operation supports both UART (asynchronous communication) and I/O interface modes. In order to execute the on-board programming, set the communication format of the programming controller as well.

- UART communication

  Communication channel: channel 0

  Serial transfer mode: UART (asynchronous), half-duplex, LSB first

  Data length: 8-bit

  Parity bit: None

  STOP bit: 1-bit

  Baud rate: Arbitrary baud rate

- I/O interface mode

  Communication channel: channel 0

  Serial transfer mode: I/O interface, full-duplex, LSB first

  Synchronous signal (SCLK0): Input mode, rising edge setting

  Handshaking signal: PE4 (output mode)

  Baud rate: Arbitrary baud rate

The boot program operates the clock/mode control block setting as an initial condition. For detail of the initial setting of the clock, refer to "Clock/Mode control".

As explained in the "17.3.5.1 Serial Operation Mode Determination", a baud rate is determined by the 16-bit timer (TMRB). When determining the baud rate, communication is executed by 1/16 of a desired baud rate. Therefore, the communication baud rate must be within the measurable range. The timer count clock operates at ΦT1 (fc/2).

A handshaking pin of I/O interface mode outputs "Low" waiting in receive state and outputs "High" in transmission state. Check the handshaking pin before communications and must follow the communication protocol.

Table 17-11 shows the pins used in the boot program. Other than these pins are not used by the boot program.

Table 17-11 Pin connection

Pin		Interface	
		UART	I/O interface mode
Mode setting pin	$\overline{\text{BOOT}}$	o	o
Reset pin	$\overline{\text{RESET}}$	o	o
Communication pin	TXD0 (PE0)	o	o
	RXD0 (PE1)	o	o
	SCLK0 (PE2)	×	o (Input mode)
	PE4	×	o (Output mode)

o:used ×:unused

17.3.3 Restrictions on Internal Memories

Note that the single boot mode places restrictions on the built-in RAM and built-in flash memory as shown in Table 17-12.

Table 17-12 Restrictions on the memories in the single boot mode

Memory	Restrictions
Internal RAM	Boot program uses the memory as a work area through 0x2000_0000 to 0x2000_03FF. Store the program 0x2000_0400 through the end address of RAM.
Internal flash memory	The following addresses are assigned for storing software ID information and passowerds. Storing program in below addresses is not recommendable. 0x3F83_FFF0 to 0x3F83_FFFF

Note: If a password is erased data (0xFF), it is difficult to protect data secure due to an easy-to-guess password. Even if the single boot mode is not used, it is recommended to set a unique value as a password.

17.3.4 Operation Command

The boot program provides the following operation commands.

Table 17-13 Operation command data

Operation command data	Operation mode
0x10	RAM transfer
0x40	Flash memory chip erase and protect bit erase

17.3.4.1 RAM Transfer

The RAM transfer is to store data from the controller to the built-in RAM . When the transfer is complete normally, a user program starts. User program can use the memory address of 0x2000_0400 or later except 0x2000_0000 to 0x2000_03FF for the boot program. CPU will start execution from RAM store start address.

This RAM transfer function enables user-specific on-board programming control. In order to execute the on-board programming by a user program, use Flash memory command sequence explained in 17.2.6.

17.3.4.2 Flash Memory Chip Erase and Protect Bit Erase

Flash memory chip erase and protect bit erase commands erase the entire blocks of Flash memory and write/erase protects of all blocks regardless of write/erase protect or security status. When the command is complete, the FCSECBIT<SECBIT> is set to "1".

17.3.5 Common Operation regardless of Command

This section describes common operation under the boot program execution.

17.3.5.1 Serial Operation Mode Determination

When the controller communicates via UART, set the 1st byte to 0x86 at the desired baud rate. When the controller communicate via I/O interface mode, set the 1st byte to 0x30 at 1/16 of the desired baud rate. Figure 17-6 shows waveforms in each case.

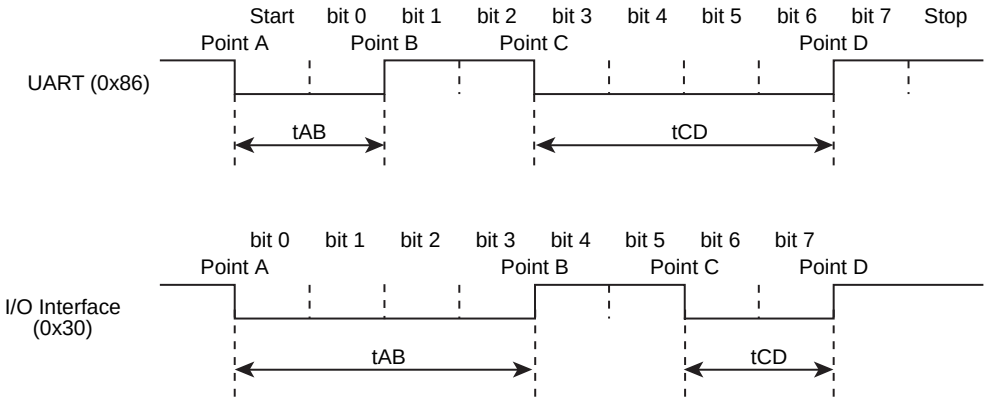


Figure 17-6 Serial operation mode determination data

Figure 17-7 shows a flowchart of boot program. Using 16-bit timer (TMRB) with the time of tAB, tAC and tAD, the 1st byte of serial operation mode determination data (0x86, 0x30) after reset is provided. In Figure 17-7, the CPU monitors level of the receive pin, and obtains a timer value at the moment when the receive pin's level is changed. Consequently, the timer values of tAB, tAC and tAD have a margin of error. In addition, note that if the transfer goes at a high baud rate, the CPU may not be able to determine the level of receive pin. In particular, I/O Interface tends to generate this problem since its baud rate is generally much higher than those of UART. To avoid this, the controller should send data at 1/16 of the desired baud rate in the I/O interface mode.

The flowchart in Figure 17-8 shows the serial operation mode is determined that the time length of the receive pin is long or short. If the length is tAB ≤ tCD, the serial operation mode is determined as UART mode. The time of tAD is used whether the automatic baud rate setting is enable or not. If the length is tAB > tCD, the serial operation mode is determined as I/O Interface mode. Note that timer values of

tAB, tAC and tAD have a margin of error. If the baud rate is high and operation frequency is low, each timer value becomes small. This may generates unexpected determination occurs. (To prevent this problem, re-set UART 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 where the time is expected 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, it is not necessary that the first byte is 0x30 as long as tAB >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>tCD is established 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).

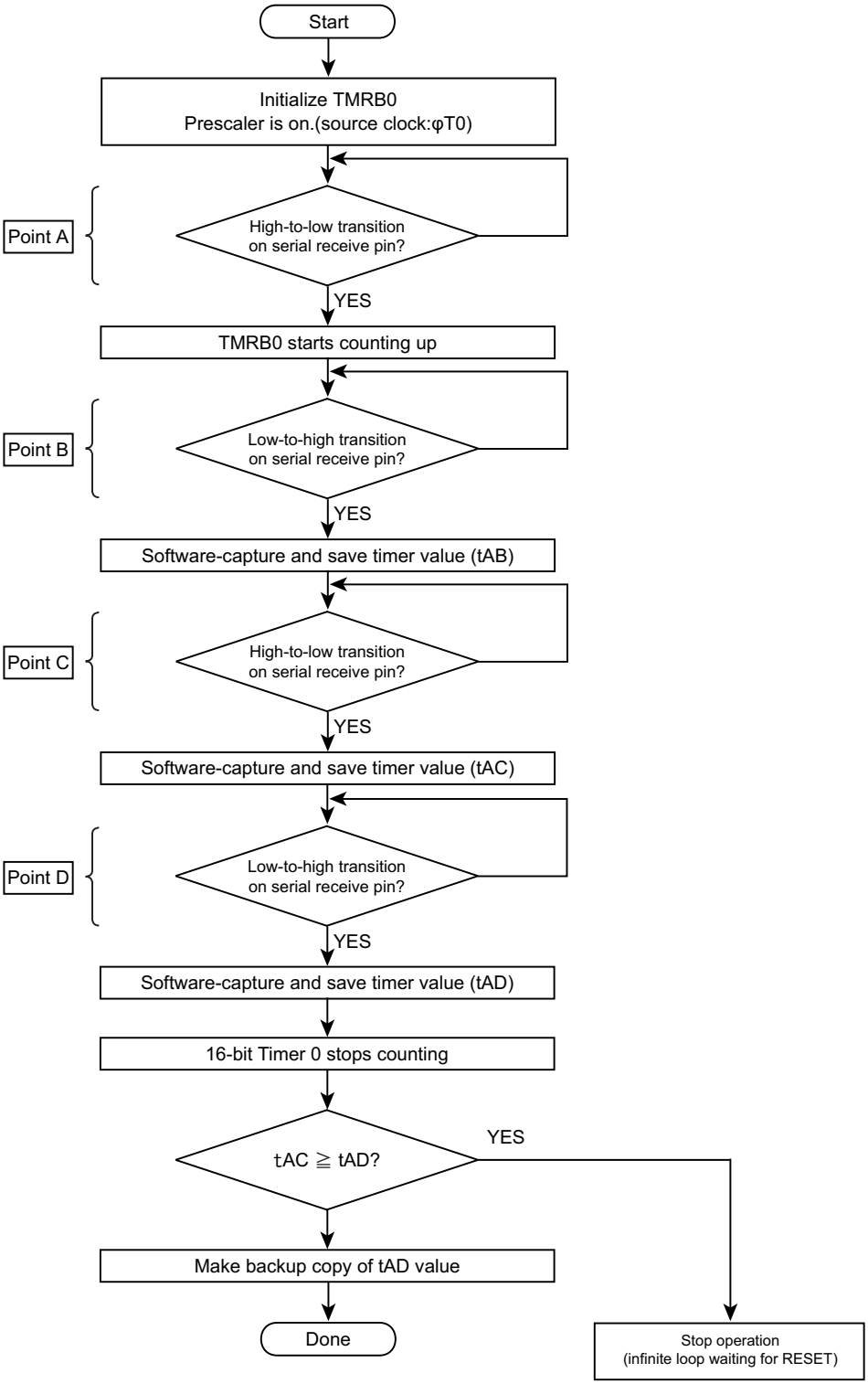


Figure 17-7 Serial operation mode receive flowchart

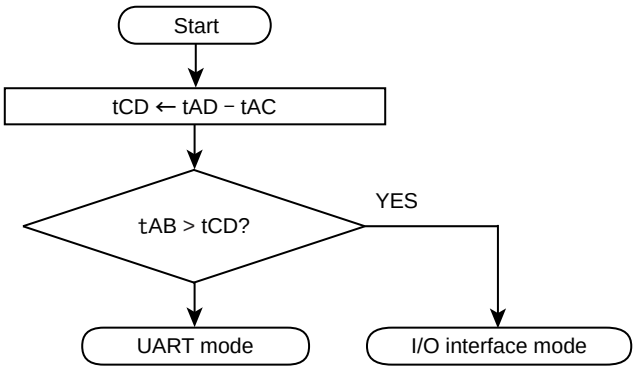


Figure 17-8 Serial operation mode determination flowchart

17.3.5.2 Acknowledge Response Data

The boot program represents processing states in specific codes and sends them to the controller. Table 17-14 to Table 17-17 show the values of acknowledge responses to each receive data.

In Table 17-15 to Table 17-17, the upper four bits of the acknowledge response are equal to those of the operation command data. The 3rd bit indicates a receive error. The 0th bit indicates an invalid operation command error, a checksum error or a password error. The 1st bit and 2nd bit are always "0". Receive error checking is not performed in I/O Interface mode.

Table 17-14 ACK response to the serial operation determination data

Transmit data	Description
0x86	Determined that UART communication is possible. (Note)
0x30	Determined that I/O interface communication is possible.

Note: When the serial operation is determined as UART, if the baud rate setting is determined as unacceptable, the boot program aborts without sending back any response.

Table 17-15 ACK response to the operation command data

Transmit data	Description
0x?8 (Note)	A receive error occurrs in the operation command data
0x?1 (Note)	An undefined operation command data is received normally.
0x10	Determined as a RAM transfer command
0x20	Determined as a Flash Memory SUM command
0x30	Determined as a Device code read command
0x40	Determined as a flash memory chip erase command

Note: The upper 4 bits of the ACK response data are the same as those of the previous command data.

Table 17-16 ACK response to the CHECK SUM data

Transmit data	Description
0xN8 (Note)	A receive error occurs.
0xN1 (Note)	A CHECK SUM or a password error occurs.
0xN0 (Note)	The CHECK SUM value is correct.

Note: The upper 4 bits of the ACK response data are the same as those of the operation command data.

Table 17-17 ACK response to Flash memory chip erase and protect bit erase operation

Transmit data	Description
0x54	Determined as a erase enable command
0x4F	Erase command is complete.
0x4C	Erase command is abnormally terminated.

17.3.5.3 Password Determination

The boot program use the below area to determine whether a password is required or use as a password.

Area	Address
Password requirement determination	0x3F83_FFF0 (1byte)
Password area	0x3F83_FFF4 to 0x3F83_FFFF (12byte)

The RAM Transfer command performs a password verification regardless of necessity judging data. Flash memory chip erase or protect bit erase command performs a password verification only when necessity judging is determined as "required".

Password requirement setting	Data
Need password	Other than 0xFF
No password	0xFF

If a password is set to 0xFF (erased data), it is difficult to protect data securely due to an easy-to-guess password. Even if Single Boot mode is not used, it is recommended to set a unique value as a password.

(1) Password verification using RAM transfer command

If all these address locations contain the same bytes of data other than 0xFF, this condition is determined as a password area error as shown in Figure 17-9. In this case, the boot program returns an error acknowledge (0x11) in response to the 17th byte of checksum value regardless of the password verification.

The boot program verifies 5th byte to 16th byte of receive data (password data). A password error occurs if all 12 bytes do not match. If the password error is determined, an ACK response data to the 17th of CHECK SUM data is a password error.

The password verification is performed even if the security function is enabled.

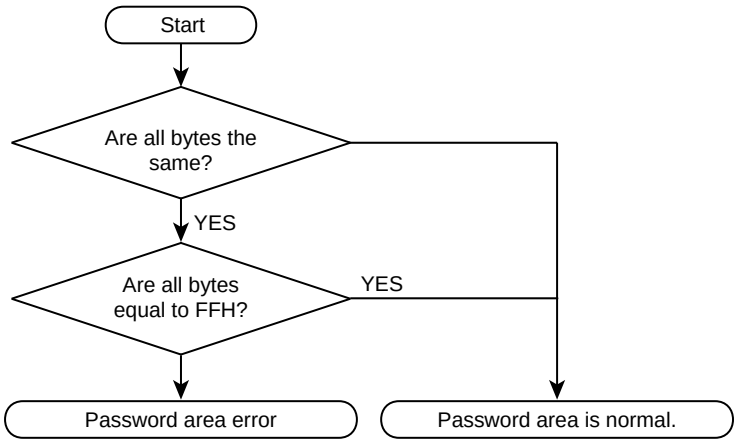


Figure 17-9 Password area check flowchart

(2) Password verification to Flash memory chip erase and protect bit erase command

When a password is enable in the erase password necessity determination area as shown in Figure 17-10 and the passwords are identical data, a password area error occurs. If a password area error is determined, an ACK response to the 17th byte of CHECK SUM sends 0x41 regardless of the password verification.

The boot program verifies 5th byte to 16th byte of receive data (password data). A password error occurs if all 12 bytes do not match. If the password error is determined, an ACK response data to the 17th of CHECK SUM data is a password error.

The password verification is performed even if the security function is enabled.

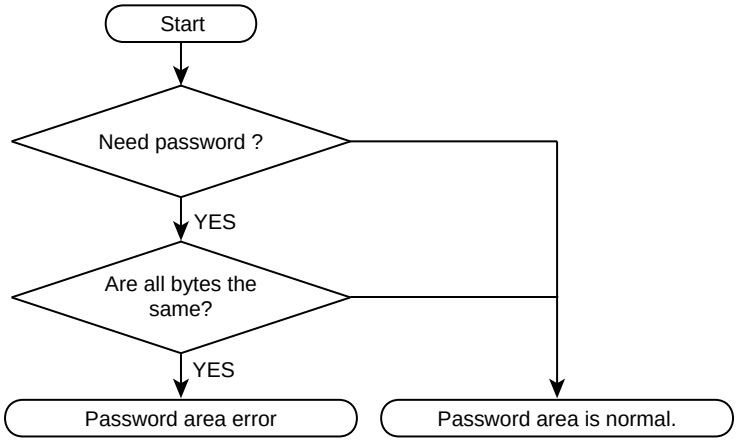


Figure 17-10 Password area check flowchart

17.3.5.4 CHECK SUM Calculation

The checksum is calculated by 8-bit addition to transmit data, dropping the carries, and taking the two's complement of the total sum. The controller must perform the same checksum operation in transmitting checksum bytes.

Example of CHECK SUM

To calculate the checksum for a series of 0xE5 and 0xF6, perform 8-bit addition.

0xE5 + 0xF6 = 0x1DB

Take the two's complement of the sum to the lower 8-bit, and that is a checksum value. So the boot program sends 0x25 to the controller.

0 − 0xDB = 0x25

17.3.6 Transfer Format at RAM Transfer

This section shows a RAM transfer command format. Transfer directions in the table are indicated as follows:

Transfer direction (C→T): Controller to TMPM365FYXBG

Transfer direction (C←T): TMPM365FYXBG to Controller

Number of transfer bytes	Transfer direction	Transfer data	Description
1	C→T	Serial operation mode and baud rate setting	Sends data to determine the serial operation mode. For detail of mode determination, refer to "17.3.5.1 Serial Operation Mode Determination".
		[UART mode] 0x86	Sends 0x86. If UART mode is determined, the program determines whether a baud setting is possible. If not, the program stops and communication is shutdown.
		[I/O interface mode] 0x30	Sends 0x30 at 1/6 of desired baud rate.The 2nd byte is also sent at 1/6 of desired baud rate. From the 3rd byte or later, you can send the data at a desirable baud rate.
2	C←T	ACK response to serial operation mode	The 2nd byte of transmit data is a ACK response data to the 1st byte that corresponds to the serial operation setting mode data. If the setting is possible, sets SIO/UART. A receive enable timing is set before transmit buffer is written to the data.
		[UART mode] Normal state: 0x86	If the setting is determined to be possible, sends 0x86. If not, the operation aborts without sending back any response. When the controller finished to send the 1st byte of data, requires a time-out time (5 seconds). If data (0x86) is not normally received within a time-out time, communication is not possible.
		[I/O interface mode] Normal state:0x30	Writes data (0x30) to the transmit buffer and waits for SCLK0 clock. When the controller finished to send the 1st byte of data and several ms (idle time) later, outputs SCLK clock. At this time, the baud rate is set to 1/16 of desired baud rate. If receive data is 0x30, communications are possible. After the 3rd byte, sets a desired baud rate to communicate.
3	C→T	Operation command data (0x10)	Sends RAM transfer command data (0x10).
4	C←T	ACK response to operation command Normal state: 0x10 Abnormal state: 0xX1 Communication error: 0xX8	ACK response data to the operation command. First, checks if 3rd byte of receive data has errors. (UART mode only) If receive errors exist, sends a ACK response data 0xX8 that means abnormal communications and waits for a next operation command (3rd byte). Upper 4 bits of transmit data are undefined. (same as upper 4 bits of immediately before operation command.) Note that in the I/O interface, receive error check is not performed. Then, if the 3rd byte of receive data corresponds to either operation command data in Table 17-13, receive data is echoed back. In the case of RAM transfer, 0x10 is echoed back and the transfer data branches to the RAM transfer service routine. If the data does not correspond to the command in Table 17-13, sends a ACK response data 0xX1 that means operation command errors, and waits for next operation command (3rd byte). Upper 4 bits of transmit data are undefined. (same as upper 4 bits of immediately before operation command data.)
5 to 16	C→T	Password data (12-byte) 0x3F83_FF04 to 0x3F83_FF0F	Checks data in the password area. For detail of password area checking, refer to "17.3.5.3 Password Determination". Compares 5th to 16th byte of receive data with 0x3F83_FFF0 to 0x3F83_FFFF of data of Flash memory. If the data does not match the address, a password error flag is set.
17	C→T	5th to 16th byte of CHECK SUM values	Send 5th to 16th byte of CHECK SUM values. For detail of CHECK SUM calculation, refer to 17.3.5.4.

Number of transfer bytes	Transfer direction	Transfer data	Description
18	C←T	ACK response to CHECK SUM value Normal state: 0x10 Abnormal state: 0x11 Communication error: 0x18	First, checks if 5th to 17th byte of receive data have errors.(UART mode only) If receive errors exist, sends a ACK response data 0x18 that means abnormal communications and waits for a next operation command (3rd byte). Then checks 17th byte of CHECK SUM data. If errors exist, sends 0x11 and waits for a next operation command (3rd byte). Finally, checks the result of password verification. If a password error exists, sends a ACK response data 0x11 that means a password error and waits for a next operation command (3rd byte). If all procedure normally ends, sends a normal ACK response data 0x10.
19	C→T	RAM store start Address 31 to 24	Sends a start address of block transfer for RAM store. The 19th byte corresponds to 31st to 24th bit of address.The 22nd byte corresponds to 7th to 0th bit of address. Specify the address to the address 0x2000_0400 through the last address of RAM.
20	C→T	RAM store start Address 23 to 16	
21	C→T	RAM store start Address 15 to 8	
22	C→T	RAM store start Address 7 to 0	
23	C→T	Number of RAM store bytes 15 to 8	Set the number of bytes to perform block transfer. The 23rd byte corresponds to the15th bit to 8th bit of transfer bytes. The 24th byte corresponds to 7th bit to 0th bit of transfer bytes. Specify the data to be stored in the address from 0x2000_0400 through the last address of RAM.
24	C→T	Number of RAM store bytes 7 to 0	
25	C→T	19th to 24th byte of CHECK SUM value	Send 19th byte to 24th byte of CHECK SUM values
26	C←T	ACK response to CHECK SUM value Normal state: 0x10 Abnormal state: 0x11 Communication error: 0x18	First, checks if 19th byte to 25th byte of receive data have errors.(UART mode only) If receive errors exist, sends a ACK response data 0x18 that means abnormal communications and waits for a next operation command (3rd byte). Then checks 25th byte of CHECK SUM data. If errors exist, sends 0x11 and waits for a next operation command (3rd byte). If all procedure normally ends, sends a normal ACK response data 0x10.
27 to m	C→T	RAM stored data	Sends same bytes of data specified in 23th bytes to 24 byte for RAM stored data.
m+1	C→T	27 to m byte of CHECK SUM value	Sends 27th byte to m byte of CHECK SUM value
m+2	C←T	ACK response to CHECK SUM value Normal state:0x10 Abnormal state: 0x11 Communication error: 0x18	First, checks if 27th byte to m+1 byte of receive data have errors. (UART mode only) If receive errors exist, sends a ACK response data 0xX8 that means abnormal communications and waits for a next operation command (3rd byte). Then checks m+1 byte of CHECK SUM data, if errors exist, sends 0x11 and waits for a next operation command (3rd byte). If all procedure normally ends, sends a normal ACK response data 0x10.
–	–	–	If m + 2nd byte of ACK response data is normal ACK response data, the transfer data branches to the address specified in 19th byte to 22 byte.

17.3.7 Transfer Format of Flash memory Chip Erase and Protect Bit Erase

This section shows a transfer format of Flash memory chip erase and protect bit erase commands. Transfer directions in the table are indicated as follows:

Transfer direction (C→T): Controller to TMPM365FYXBG

Transfer direction (C←T): TMPM365FYXBG to Controller

Number of transfer bytes	Transfer direction	Transfer data	Description
1	C→T	Serial operation mode and baud rate setting	Sends data to determine the serial operation mode. For detail of mode determination, refer to"17.3.5.1 Serial Operation Mode Determination".
		[UART mode] 0x86	Sends 0x86. If UART mode is determined, checks if baud rate setting can be done. If not, operation stops communications.
		[I/O interface mode] 0x30	Sends 0x30 at 1/16 of desired baud rate. Same as the 1st byte, sends the 2nd byte at 1/16 of desired baud rate. Desired baud rate can be used after 3rd byte.
2	C←T	ACK response to serial operation mode	The 2nd byte of transmit data is a ACK response data to the 1st byte that corresponds to the serial operation setting mode data. If the setting is possible, sets SIO/UART. A receive enable timing is set before transmit buffer is written to the data.
		[UART mode] Normal state: 0x86	If the setting is determined to be possible, sends 0x86. If not, the operation aborts without sending back any response. When the controller finished to send the 1st byte of data, requires a time-out time (5 seconds). If data (0x86) is not normally received within a time-out time, communication is not possible.
		[I/O interface mode] Normal state:0x30	Writes data (0x30) to the transmit buffer and waits for SCLK0 clock. When the controller finished to send the 1st byte of data and several ms (idle time) later, outputs SCLK clock. At this time, the baud rate is set to 1/16 of desired baud rate. If receive data is 0x30, communications are possible. After the 3rd byte, sets a desired baud rate to communicate.
3	C→T	Operation command data (0x40)	Sends Flash memory chip erase and protect bit erase command data (0x40).
4	C←T	ACK response to the operation command Normal state: 0x40 Abnormal state: 0xX1 Communication error: 0xX8	ACK response data to the operation command. First, checks if 3rd byte of receive data has errors. (UART mode only) If receive errors exist, sends a ACK response data 0xX8 that means abnormal communications and waits for a next operation command (3rd byte). Upper 4 bits of transmit data are undefined. (same as upper 4 bits of immediately before operation command data.) Note that in the I/O interface, receive error check is not performed. Then, if the 3rd byte of receive data corresponds to either operation command data in Table 17-13, receive data is echoed back. If the data does not correspond to the command in Table 17-13, sends a ACK response data 0xX1 that means operation command errors, and waits for next operation command. (3rd byte) Upper 4 bits of transmit data are undefined. (Upper 4 bits of immediate before operation command data are used.)
5 to 16	C→T	Password data (12-byte) 0x3F83_FF04 to 0x3F83_FF0F	If password necessity is set to "none", this data is dummy data. If password necessity is set to "necessary", checks data in the password area. For a method of password area data checking, refer to "17.3.5.3 Password Determination". Compares 5th to 16th byte of receive data with 0x3F83_FFF0 to 0x3F83_FFFF of data of Flash memory in order. If the data does not match, a password error flag is set.
17	C→T	5 to 16 th byte CHECK SUMvalue	Sends 5th byte to 16 byte of CHECK SUM value. For a method of CHECK SUM calculation, refer to"17.3.5.4 CHECK SUM Calculation".
18	C←T	ACK response to the CHECK SUM value Normal state: 0x40 Abnormal state: 0x41 Communication error: 0x48	If password necessity is set to "none", sends a normal ACK response data 0x40. If password necessity is set to "necessary", first checks if receive errors exist in the 5th byte to 17th byte receive data. (UART mode only) If a receive error exists, sends a ACK response data 0x48 that means abnormal communications and waits for next operation command. (3rd byte) Then checks 17th byte of CHECK SUM data. If error occurs, sends 0x41 and waits for a next operation command (3rd byte) Finally, checks the result of password verification. If a password error exists, sends a ACK response data 0x41that means a password error and waits for a next operation command (3rd byte) If all procedure normally ends, sends a normal ACK response data 0x40.
19	C→T	Erase enable command data (0x54)	Sends an enable command data (0x54).
20	C←T	ACK response to the erase enable command Normal state: 0x54 Abnormal state: 0xX1 Communication error: 0x58	First, checks if 19th byte of receive data has errors. If receive errors exist, sends a ACK response data (bit 3) 0x58 that means abnormal communication and waits for next operation command (3rd byte). Then, if 19th byte of receive data corresponds to the erase enable command, receive data is echoed back (normal ACK response data). In this case, 0x54 is echoed back and the transfer data branches into Flash memory chip erase process routine. If the data does not correspond to the erase enable command, sends a ACK response data (bit 0) 0xX1 and waits for next operation command. Upper 4 bits of transmit data are undefined. (Upper 4 bits of immediate before operation command data are used.)
21	C→T	ACK response to the erase command Normal state: 0x4F Abnormal state: 0x4C	If the operation is normally complete, the end code (0x4F) is returned. If erase error occurs, an error code (0x4C) is returned.
–	–	–	Waits for a next operation command.

17.3.8 Boot Program Whole Flowchart

This section shows a boot program whole flowchart.

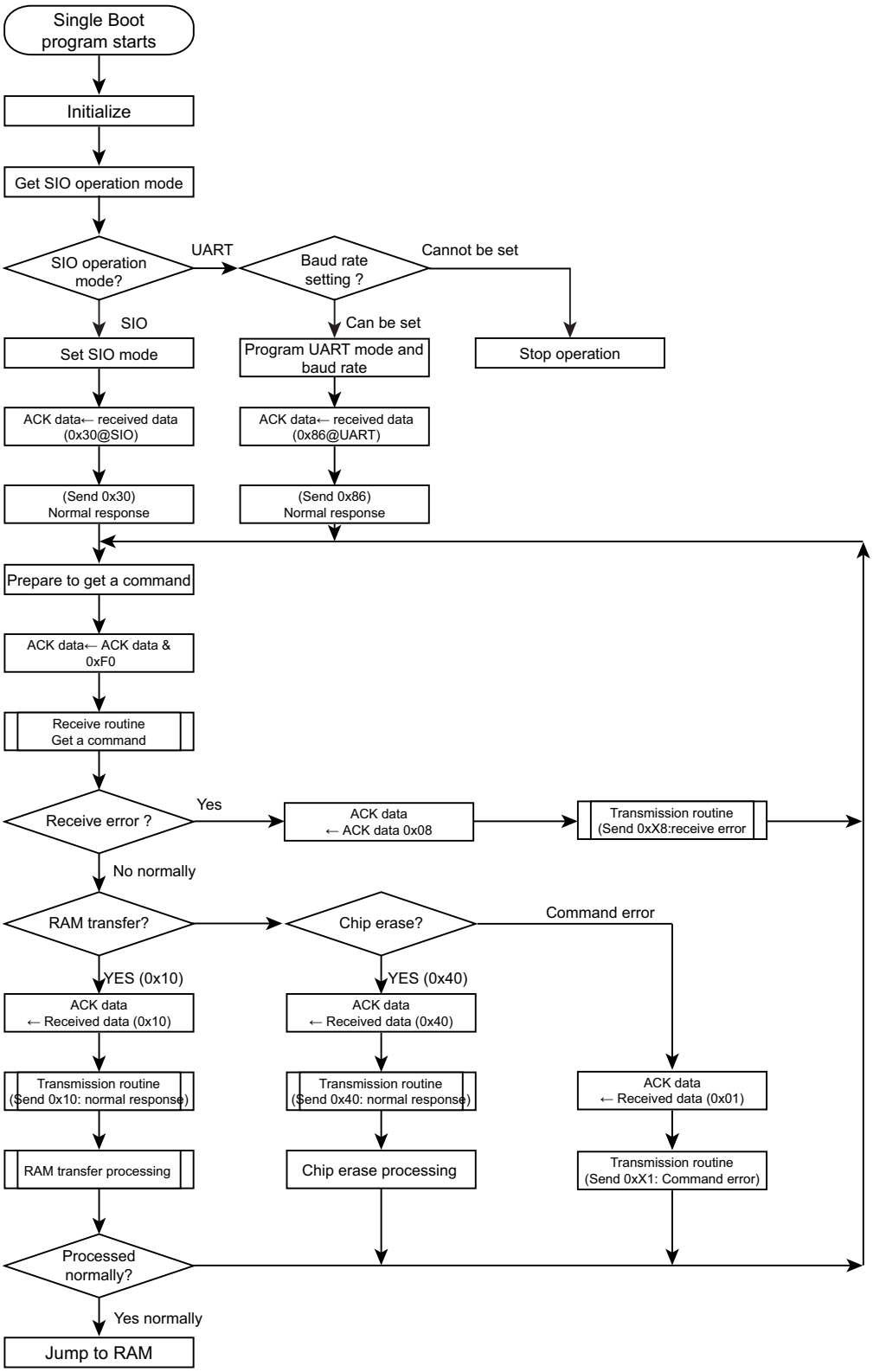


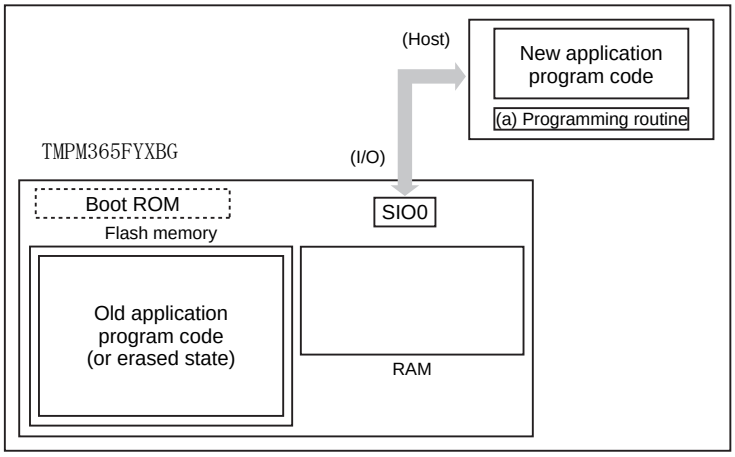
Figure 17-11 Boot program whole flowchart

17.3.9 Reprogramming Procedure of Flash using reprogramming algorithm in the on-chip BOOT ROM

This section describes the reprogramming procedure of the flash using reprogramming algorithm in the on-chip boot ROM.

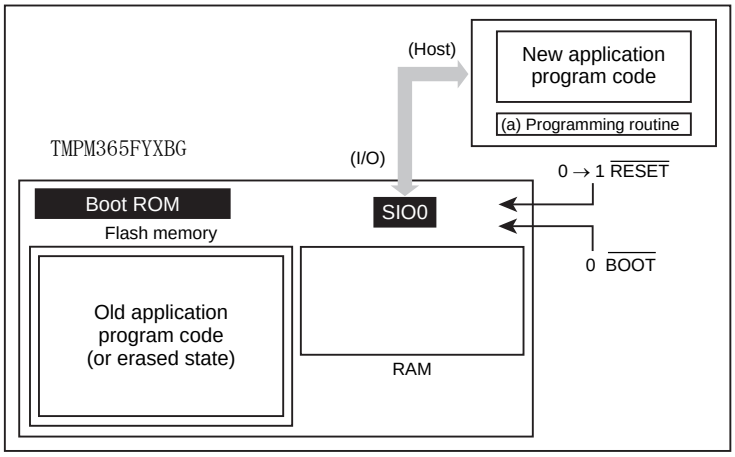
17.3.9.1 Step-1

The condition of Flash memory does not care whether a user program made of former versions has been written or erased. Since a programming routine and programming data are transferred via the SIO (SIO0), the SIO0 must be connected to an external host. A programming routine (a) is prepared on the host.



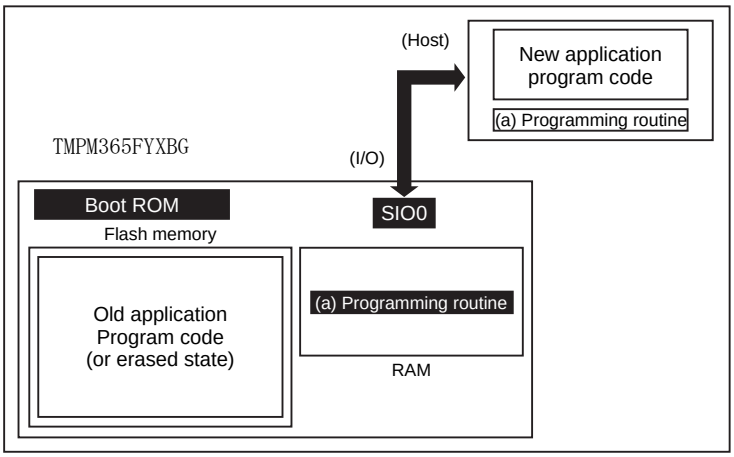
17.3.9.2 Step-2

Release the reset by pin condition setting in the boot mode and boot-up the BOOT ROM. According to the procedure of boot mode, transfer the programming routine (a) via SIO0 from the source (host). A password verification with the the password in the user application program is perfomed. (If Flash memory is erased, an erase data (0xFF) is dealt with a password.)



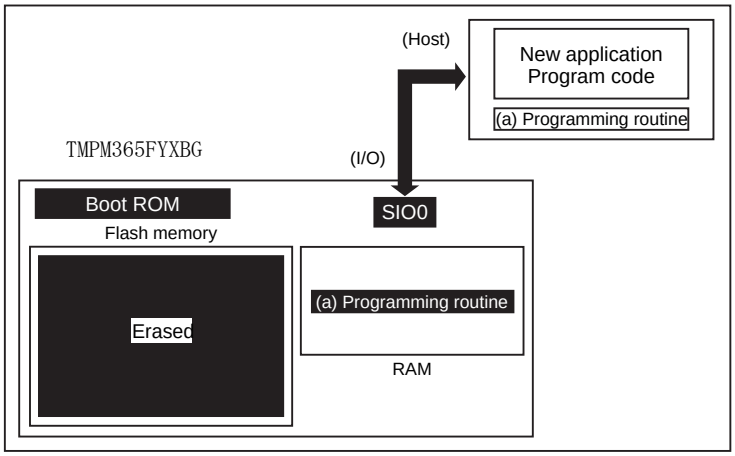
17.3.9.3 Step-3

If the password verification is complete, the boot program transfer a programming routine (a) from the host into the on-chip RAM. The programming routine must be stored in the range from 0x2000_0400 to the end address of RAM.



17.3.9.4 Step-4

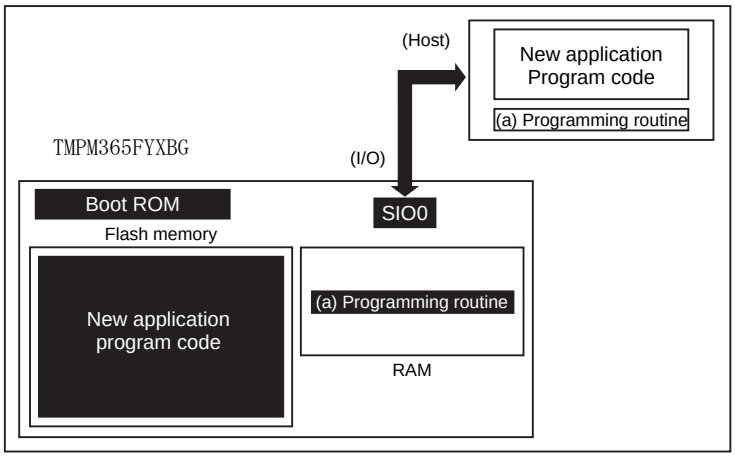
The boot program jumps to the programming routine (a) in the on-chip RAM to erase the flash block containing old application program codes. The Block Erase or Chip Erase command is be used.



17.3.9.5 Step-5

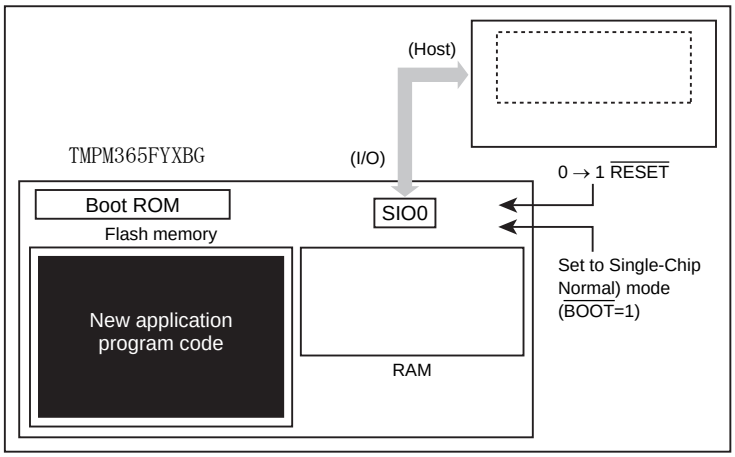
The boot program executes the programming routine (a) to download new application program codes from the host and programs it into the erased flash area. When the programming is complete, the writing or erase protection of that flash area in the user's program must be set.

In the example below, new program code comes from the same host 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 a hardware board and programming routine to suit your particular needs.



17.3.9.6 Step-6

When programming of Flash memory is complete, power off the board and disconnect the cable leading from the host to the target board. Turn on the power again so that the device re-boots in the single-chip (Normal) mode to execute the new program.



17.4 Programming in the User Boot Mode

A 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 user application is different from the serial I/O. It operates in the single chip mode; therefore, a switch from normal mode in which user application is activated in the single chip mode to the user boot mode for programming flash is required. Specifically, add a mode judgment routine to the reset service routine in the user application program.

The condition to switch the modes needs to be set according to the user's system setup condition. Also, a flash memory programming routine that the user uniquely makes up needs to be set in the new application. This routine is used for programming after being switched to the user boot mode. The data in built-in Flash memory cannot be read out during erase/reprogramming mode. Thus, reprogramming routine must be take place while it is stored in the area outside of Flash memory area. Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental reprogramming. Be sure not to generate interrupt/fault except reset to avoid abnormal termination during the user boot mode.

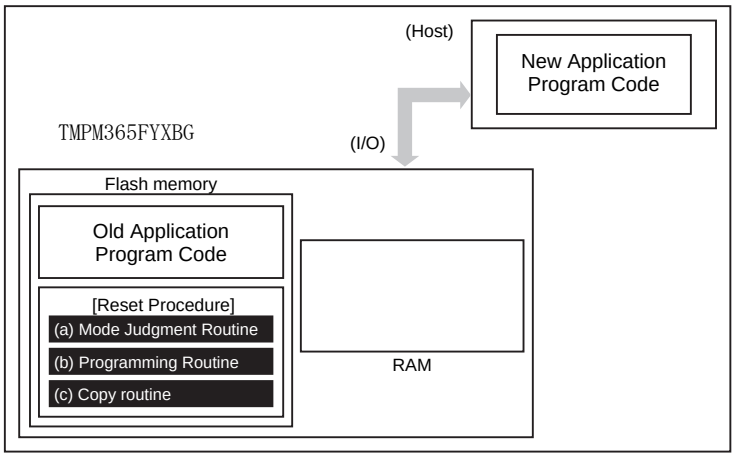
Taking examples from two cases such as the method that reprogramming routine stored in Flash memory (1-A) and transferred from the external device (1-B), the following section explains the procedure. For a detail of the program/erase to Flash memory, refer to "17.2 Detail of Flash Memory".

17.4.1 (1-A) Procedure that a Programming Routine Stored in Flash memory

17.4.1.1 Step-1

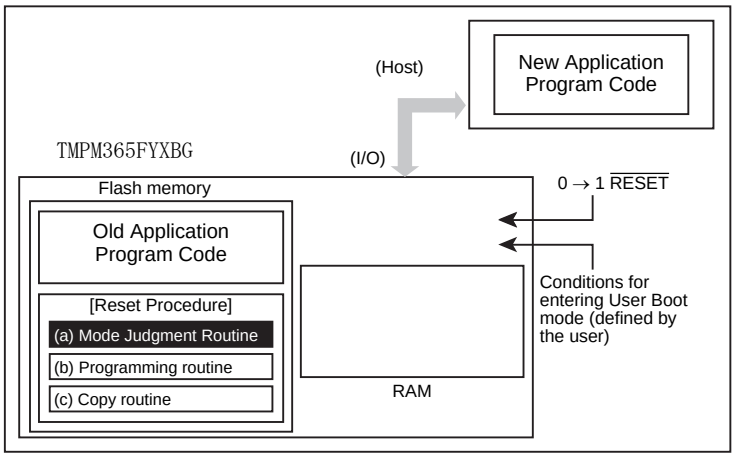
A user determines the conditions (e.g., pin status) to enter the user boot mode and the I/O bus to be used to transfer data. Then suitable circuit design and program are created. Before installing the device on a printed circuit board, write the following three program routines into an arbitrary flash block using programming equipment such as a flash writer.

(a) Mode judgment routine:	A program to determine to switch to user boot mode or not
(b) Flash programming routine:	A program to download new program from the host controller and re-program Flash memory
(c) Copy routine:	A program to copy the data described in (b) to the built-in RAM or external memory device



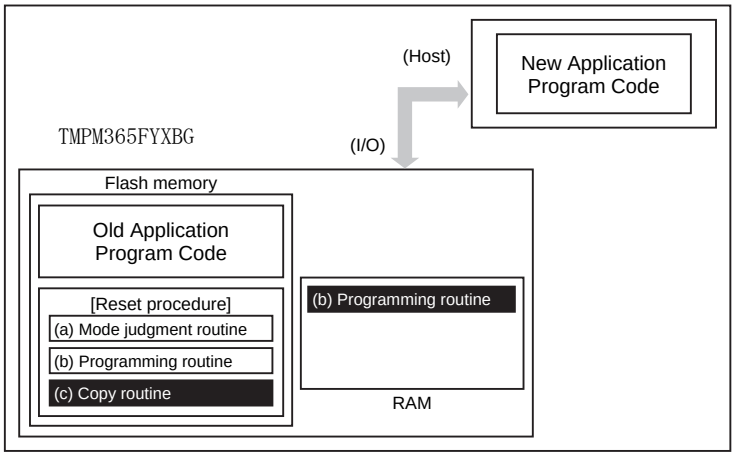
17.4.1.2 Step-2

This section explains the case that a programming routine storied in the reset routine. First, the reset routine determines to enter the user boot mode. If mode switching conditions are met, the device enters the user boot mode to reprogram data.



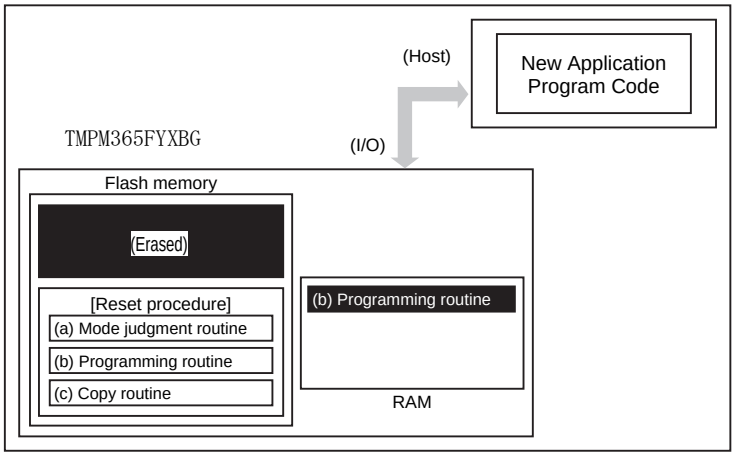
17.4.1.3 Step-3

Once the device enters the user boot mode, execute the copy routine (C) to download the flash programming routine (b) from the host controller to the built-in RAM .



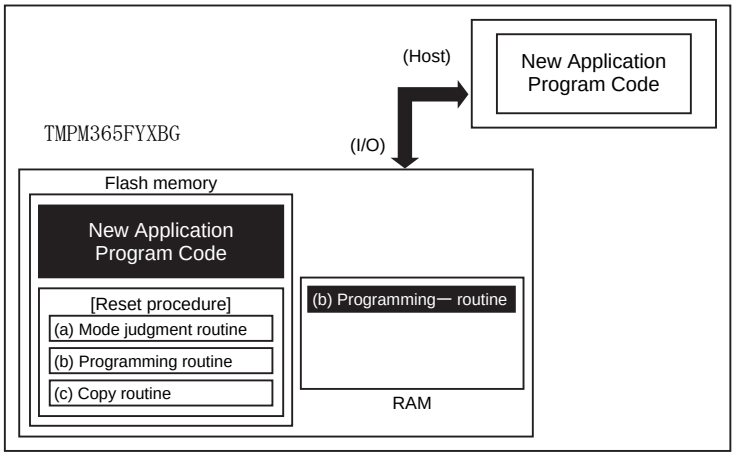
17.4.1.4 Step-4

Jump to the the reprogramming routine in the built-in RAM to release the write/erase protection for the old application program, and to erase a flash in block unit.



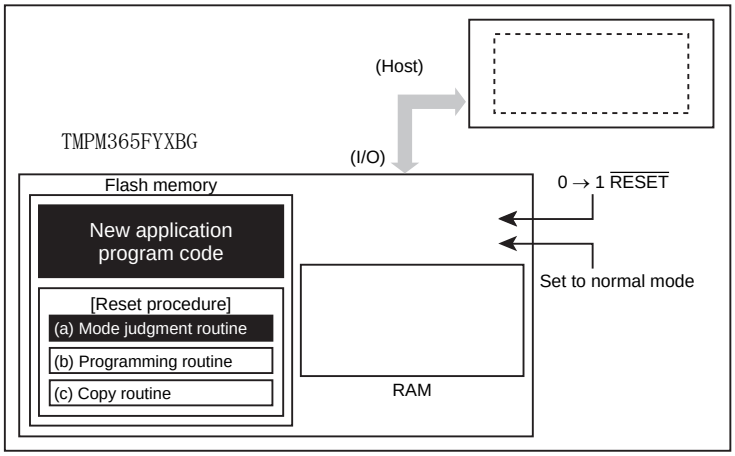
17.4.1.5 Step-5

Continue to execute the flash programming routine to download new program data from the host controller and program it into the erased flash block. When the programming is complete, the write/erase protection of that flash block in the user program area must be set.



17.4.1.6 Step-6

Set $\overline{\text{RESET}}$ to "0". Upon reset, Flash memory is set to the normal mode. After reset, the CPU will start along with the new application program.



17.4.2 (1-B) Procedure that a Programming Routine is transferred from External Host

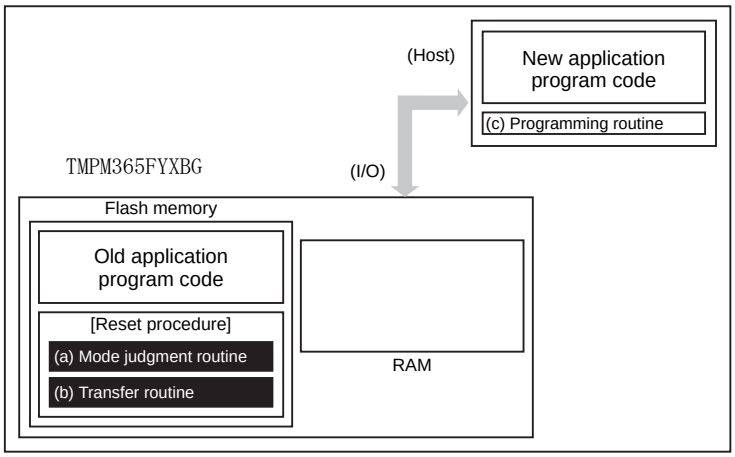
17.4.2.1 Step-1

A user determines the conditions (e.g., pin status) to enter the user boot mode and the I/O bus to be used to transfer data. Then suitable circuit design and program are created. Before installing the device on a printed circuit board, write the following two program routines into an arbitrary flash block using programming equipment such as a flash writer.

(a) Mode judgment routine: A program to determine to switch to reprogramming operation
(b) Transfer routine: A program to obtain a reprogramming program from the external device.

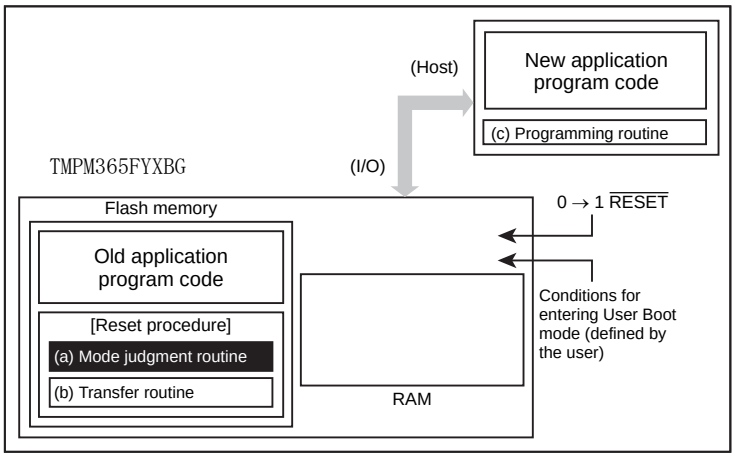
In addition, prepare a reprogramming routine shown below must be stored on the host controller.

(c) Reprogramming routine: A program to reprogram data



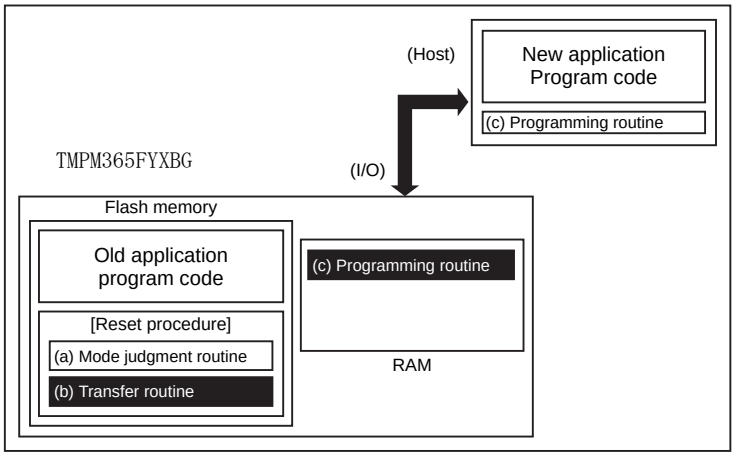
17.4.2.2 Step-2

This section explains the case that a programming routine stored in the reset routine. First, the reset routine determines to enter the user boot mode. If mode switching conditions are met, the device enters the user boot mode to reprogram data.



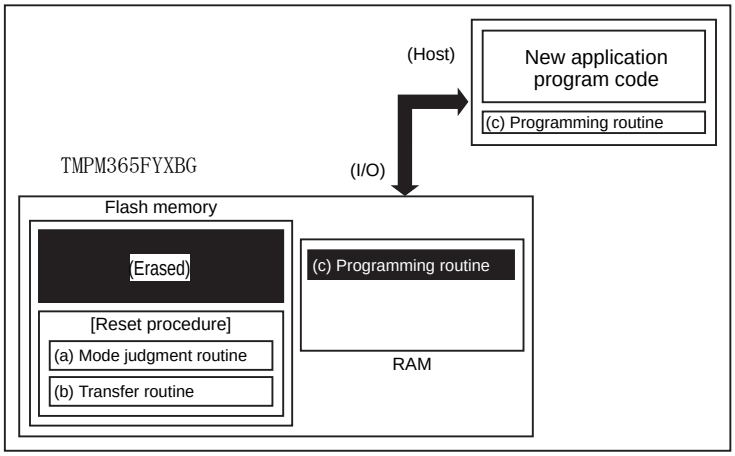
17.4.2.3 Step-3

Once the device enters the user boot mode, execute the transfer routine (b) to download the programming routine (c) from the host controller to the built-in RAM .



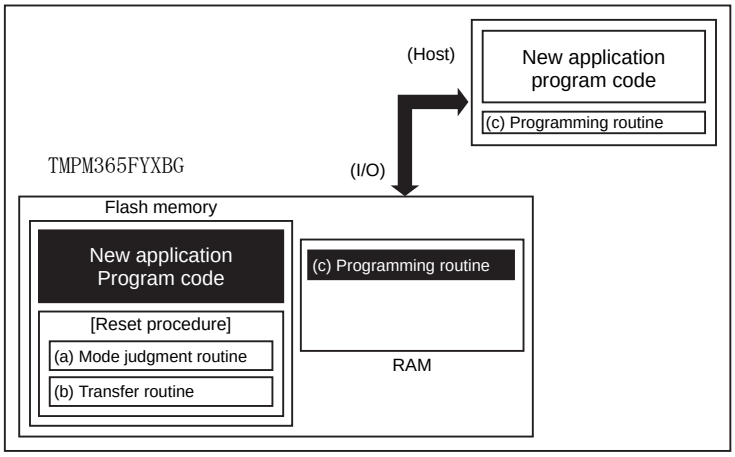
17.4.2.4 Step-4

Jump to the the reprogramming routine in the built-in RAM to release the write/erase protection for the old application program, and to erase a flash in block unit.



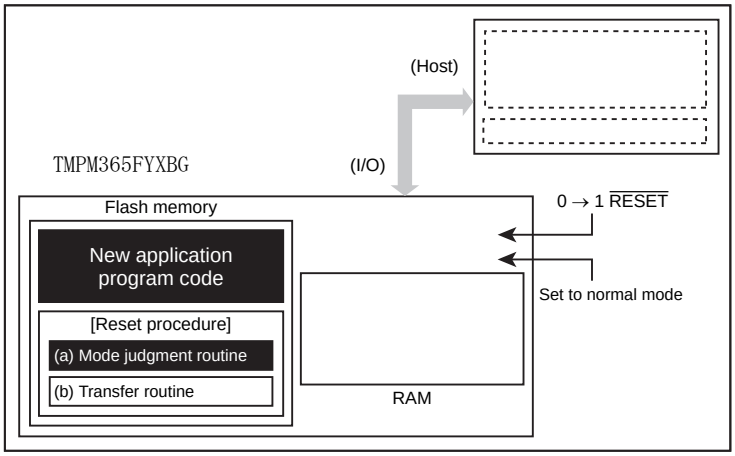
17.4.2.5 Step-5

Continue to execute the flash programming routine (c) to download new program data from the host controller and program it into the erased flash block. When the programming is complete, the write/erase protection of that flash block in the user program area must be set.



17.4.2.6 Step-6

Set $\overline{\text{RESET}}$ to "0". Upon reset, Flash memory is set to the normal mode. After reset, the CPU will start along with the new application program.

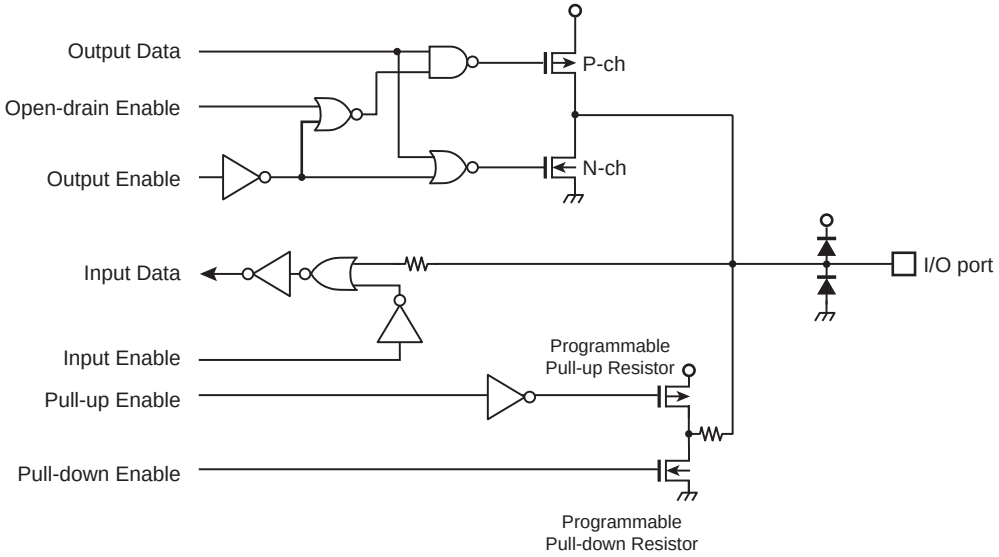


18. Port Section Equivalent Circuit Schematic

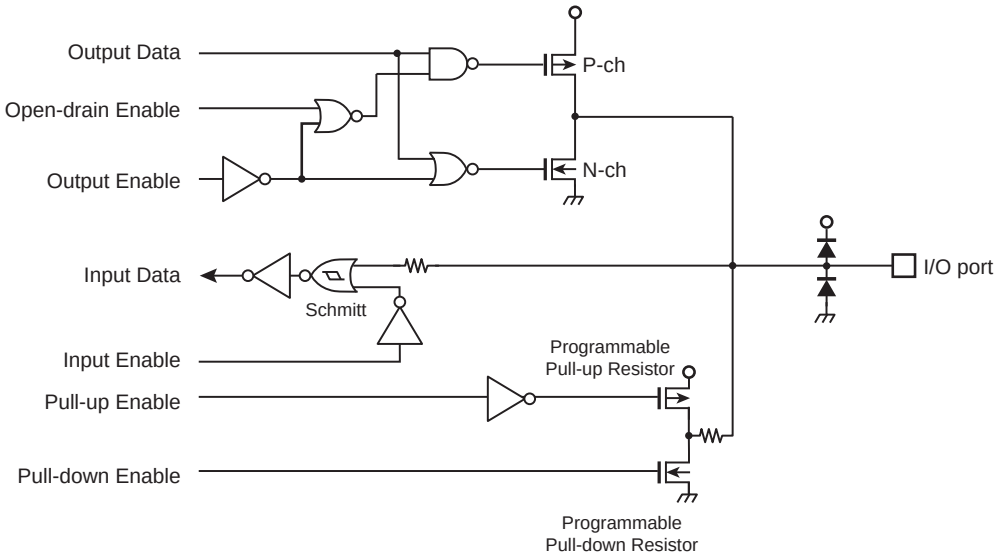
Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

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.

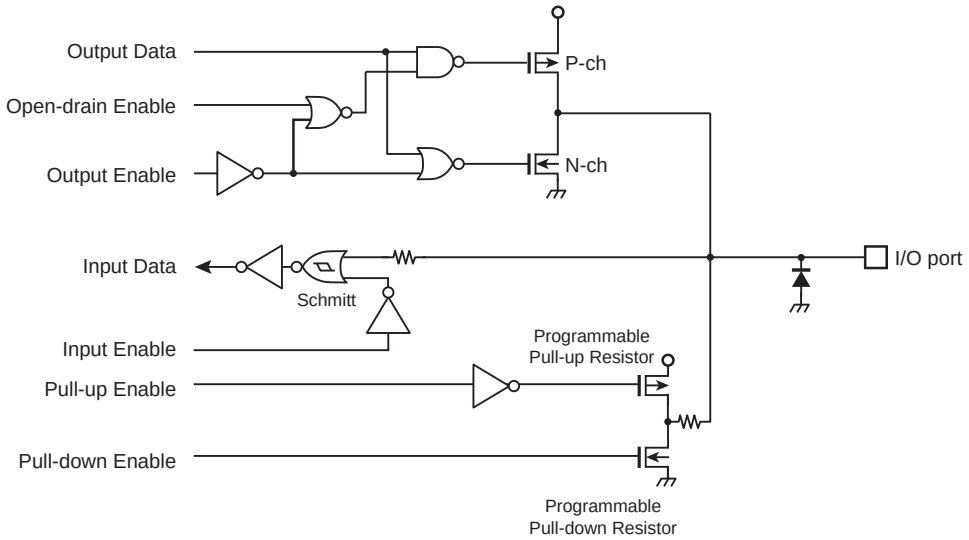
18.1 PA0 to 7,PB0 to 7



18.2 PC0 to 2, PD0 to 7, PE0 to 7, PF1 to 7, PG0 to 4, PH0 to 4, PI0 to 7

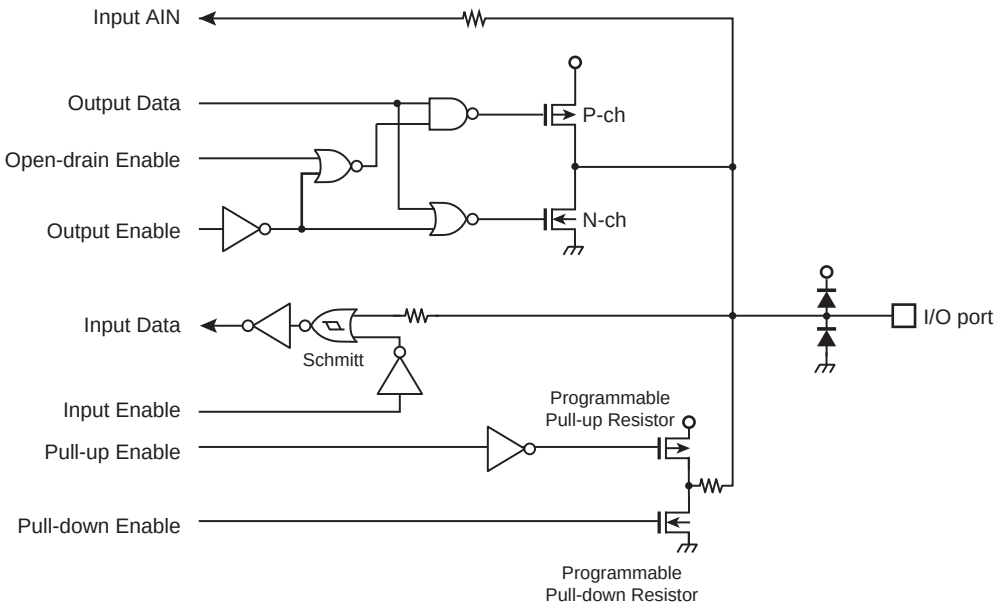


18.3 PG5

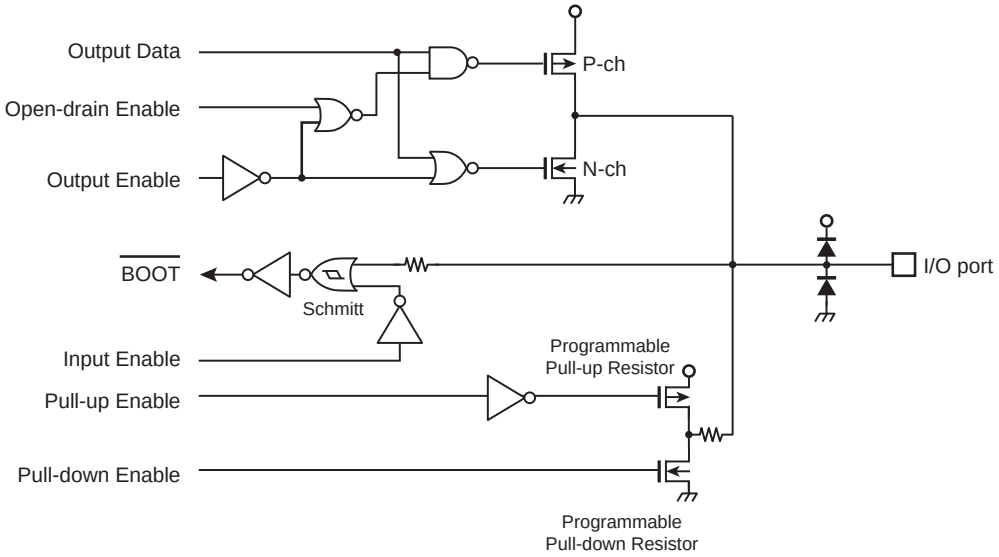


Note: Only when input is enabled, these pins tolerate 5V inputs.

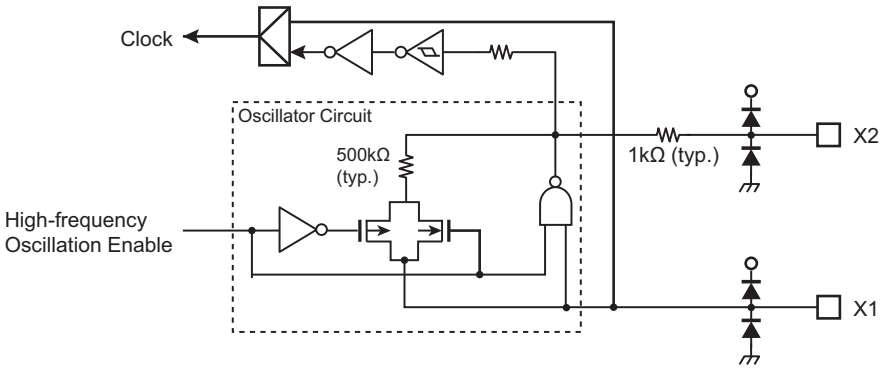
18.4 PJ0 to 7,PK0 to 3



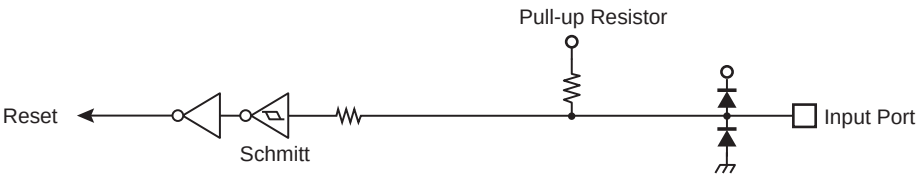
18.5 PF0



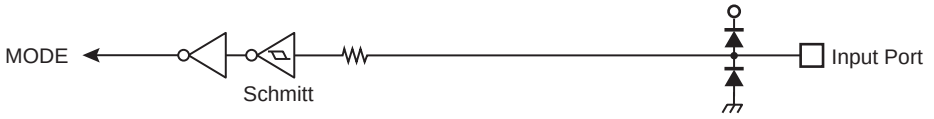
18.6 X1,X2



18.7 $\overline{\text{RESET}}$,$\overline{\text{NMI}}$

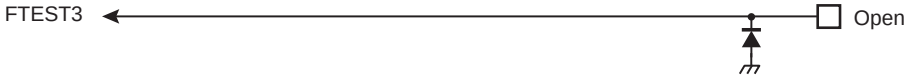


18.8 MODE



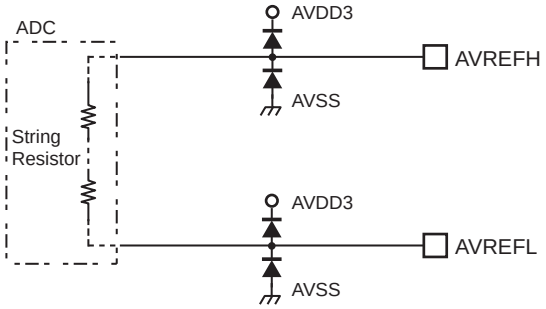
(Note)MODE pin is fixed to GND.

18.9 FTEST3



(Note)FTEST3 pin is fixed to Open.

18.10 AVREFH,AVREFL



19. Electrical Characteristics

19.1 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 port	V_{IN1}	−0.3 to VDD + 0.3	V
Input voltage	PG5 (5V tolerant input)	V_{IN2}	−0.3 to 5.5	V
Low-level output current	Per pin	I_{OL}	5	mA
	Total	ΣI_{OL}	50	
High-level output current	Per pin	I_{OH}	−5	
	Total	ΣI_{OH}	−50	
Power consumption (Ta = 85 °C)		PD	600	mW
Soldering temperature (10 s)		T_{SOLDER}	260	°C
Storage temperature		T_{STG}	−40 to 125	°C
Operating Temperature	Except during Flash W/E	T_{OPR}	−40 to 85	°C
	During Flash W/E		0 to 70	

Note: **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.**

19.2 DC Electrical Characteristics (1/3)

DVSSA = DVSSB = AVSS = RVSS = DVSSC = 0V
Ta = −40 to 85 °C

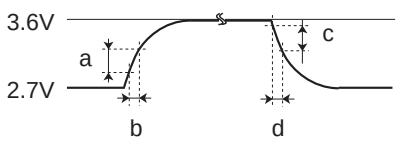
Parameter		Symbol	Condition		Min.	Typ. (Note 1)	Max.	Unit
Supply voltage	DVDD3A DVDD3C AVDD3 RVDD3 (Note 2)	DVDD3A AVDD3 RVDD3 DVDD3C	f_{OSC} = 8 to 12 MHz fsys = 1 to 48 MHz	without USB	2.7	−	3.6	V
				with USB	3.0	−	3.45	V
Low-level input voltage	PA, PB, PC, PD, PE, PF, PG, PH, PI, PJ, PK, $\overline{RESET}$, $\overline{NMI}$, MODE, BSC	V_{IL1}	2.7 V ≤ DVDD3A ≤ 3.6 V		−0.3	−	0.2 DVDD3A	V
	X1	V_{IL2}	2.7 V ≤ RVDD3 ≤ 3.6 V				0.2 RVDD3	
High-level input voltage	PA, PB, PC, PD, PE, PF, PG (Except PG5), PH, PI, PJ, PK, $\overline{RESET}$, $\overline{NMI}$, MODE, BSC	V_{IH1}	2.7 V ≤ DVDD3A ≤ 3.6 V		0.8 DVDD3A	−	DVDD3A + 0.3	V
	PG5	V_{IH3}					5.5	
	X1	V_{IH2}	2.7 V ≤ RVDD3 ≤ 3.6 V		0.8 RVDD3		RVDD3 + 0.3	
Low-level output voltage		V_{OL}	I_{OL} = 2 mA DVDD3A ≥ 2.7 V		−	−	0.4	V
High-level output voltage		V_{OH}	I_{OH} = −2 mA DVDD3A ≥ 2.7 V		2.4	−	DVDD3A	V
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		VTH	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
Power supply variation rate in operation range		VRS	DVDD3A = DVDD3C = AVDD3 = RVDD3		−	−	10.0	mV/μs
		VFS			−	−	-3.33	

Note 1: **Ta = 25 °C, DVDD3A = DVDD3C = RVDD3 = AVDD3 = RVDD3 = 3.3 V, unless otherwise noted.**

Note 2: **The same voltage must be supplied to DVDD3A, DVDD3C, AVDD3 and RVDD3.**

Note 3: **Ensure that all power suply source, including AVDD3, is power-off and then power-on again when DVDD3A, DVDD3C, AVDD3 and RVDD3 fall below 2.7V (3.0V when USB is used) which is minimum operation voltage.**

Note 4: VRS(Rising), VFS(Falling) should be measured at a strict level against a characteristics.



VRS = a / b , VFS = c / d

19.3 DC Electrical Characteristics (2/3)

AVDD3 = RVDD3 = 2.7 V to 3.6 V (Nore2)
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 <PA0 to PA7, PB0 to PB7, PC0 to PC7, PD0 to PD7, PE0 to PE7, PF0 to PF7, PG0 to PG5, PH0 to PH4, PI0 to PI7, PJ0 to PJ7, PK0 to PK7>	−	−	2	mA
	ΣI_{OL1}	Per group, Port A	−	−	10	
	ΣI_{OL2}	Per group, Port B	−	−	10	
	ΣI_{OL3}	Per group, Port C	−	−	10	
	ΣI_{OL4}	Per group, Port D	−	−	10	
	ΣI_{OL5}	Per group, Port E	−	−	10	
	ΣI_{OL6}	Per group, Port F	−	−	10	
	ΣI_{OL7}	Per group, PortG	−	−	20	
	ΣI_{OL8}	Per group, Port H	−	−	10	
	ΣI_{OL9}	Per group, Port I	−	−	10	
	ΣI_{OL10}	Per group, Port J	−	−	10	
	ΣI_{OL11}	Per group, Port K	−	−	10	
	ΣI_{OL}	Total, all Port	−	−	35	
High-level output current	I_{OH1}	Per pin 2.7 V ≤ DVDD3A, DVDD3C ≤ 3.6 V <PA0 to PA7, PB0 to PB7, PC0 to PC7, PD0 to PD7, PE0 to PE7, PF0 to PF7, PG0 to PG5, PH0 to PH4, PI0 to PI7, PJ0 to PJ7, PK0 to PK7>	−	−	-2	mA
	ΣI_{OH1}	Per group, Port A	−	−	-10	
	ΣI_{OH2}	Per group, Port B	−	−	-10	
	ΣI_{OH3}	Per group, Port C	−	−	-10	
	ΣI_{OH4}	Per group, Port D	−	−	-10	
	ΣI_{OH5}	Per group, Port E	−	−	-10	
	ΣI_{OH6}	Per group, Port F	−	−	-10	
	ΣI_{OH7}	Per group, PortG	−	−	-10	
	ΣI_{OH8}	Per group, Port H	−	−	-10	
	ΣI_{OH9}	Per group, Port I	−	−	-10	
	ΣI_{OH10}	Per group, Port J	−	−	-10	
	ΣI_{OH11}	Per group, Port K	−	−	-10	
	ΣI_{OH}	Total, all Port	−	−	-35	

Note 1: **The same voltage must be supplied to DVDD3A, DVDD3C, AVDD3, and RVDD3.**
Note 2: 3.0 V to 3.45 V (when USB is used)

19.4 DC Electrical Characteristics (3/3)

DVDD3A = DVDD3C = AVDD3 = RVDD3 = 2.7 V to 3.6 V (Note6)
Ta = −40 to 85 °C

Parameter	Symbol	Condition	Min.	Typ. (Note1)	Max.	Unit
NORMAL (Note2) (Note3)	I_{DD}	Gear 1/1 fsys = 48 MHz The TMRB, ADC, SIO/ UART, I2C/SIO and USB operates.	−	31.8	42.0	mA
NORMAL (Note2) (Note3)		Gear 1/1 fsys = 48 MHz The TMRB, ADC, SIO / UART and I2C/SIO operates.	−	22.3	32.0	
IDLE (Note4)		Gear 1/1 fsys = 48 MHz All peripherals are disabled.	−	10.0	16.0	
STOP1 (Note5)		−	−	14.1	800	μA

Note 1: **Ta = 25 °C, DVDD3A = DVDD3C = AVDD3 = RVDD3 = 3.3 V, unless otherwise noted.**

Note 2: **I_{DD} NORMAL: Measured with the dhrystone ver. 2.1 operated in FLASH.**

Note 3: I_{DD} NORMAL: The currents flow through DVDD3A, DVDD3C and AVDD3 are not included.

Note 4: **I_{DD} IDLE: Measured with all functions stopped. The currents flow through DVDD3A, DVDD3C, AVDD3 and RVDD3 are included.**

Note 5: **I_{DD}STOP1: The currents flow through DVDD3A, DVDD3C, AVDD3 and RVDD3 are included.**

Note 6: 3.0 V to 3.45 V (when USB is used)

19.5 12-bit ADC Electrical Characteristics

DVDD3A = DVDD3C = AVDD3 = RVDD3 = AVREFH = 2.7 V to 3.6 V
DVSSA = DVSS3C = AVSS = RVSS = DVSSC = 0 V
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	−	−	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	8	LSB
DNL error				−	2	8	
Offset error				−	3	8	
Full-scale error				−	3	8	
INL error		−	AIN resistance ≤ 600 Ω AIN load capacitance ≥33 pF Conversion time ≥ 1.66 μs	−	3	8	
DNL error				−	2	8	
Offset error				−	4	8	
Full-scale error				−	2	8	
Conversion time		Tconv	−	1.0	−	10	μs

Note 1: **1LSB = (AVREFH − AVSS) / 4096 [V]**
Note 2: All **Peripheral functions except ADC are disabled.**

19.6 AC Electrical Characteristics

19.6.1 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

19.6.2 Serial Channel (SIO/UART)

19.6.2.1 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		fsys = 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 / fall (Note1)	t_{SRD}	30	−	30.0	−	30.0	−	
SCLK rise / fall (Note1) → Input Data hold	t_{HSR}	x + 30	−	55.0	−	50.8	−	

[Data output]

Parameter	Symbol	Equation		fsys = 40 MHz		48 MHz		Unit
		Min	Max.	Min	Max.	Min.	Max.	
SCLK Clock High width (input)	t_{SCH}	4x	−	120 (Note3)	−	107.5 (íç3)	−	ns
SCLK Clock Low width (input)	t_{SCL}	4x	−	120 (Note3)	−	107.5 (íç3)	−	
SCLK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	−	240	−	215	−	
Output Data → SCLK rise / fall (Note1)	t_{OSS}	$t_{SCY}/2 - 3x - 45$	−	0.00 (Note2)	−	0.00 (íç2)	−	
SCLK rise / fall (Note1) → Output Data hold	t_{OHS}	$t_{SCY}/2$	−	120	−	107.5	−	

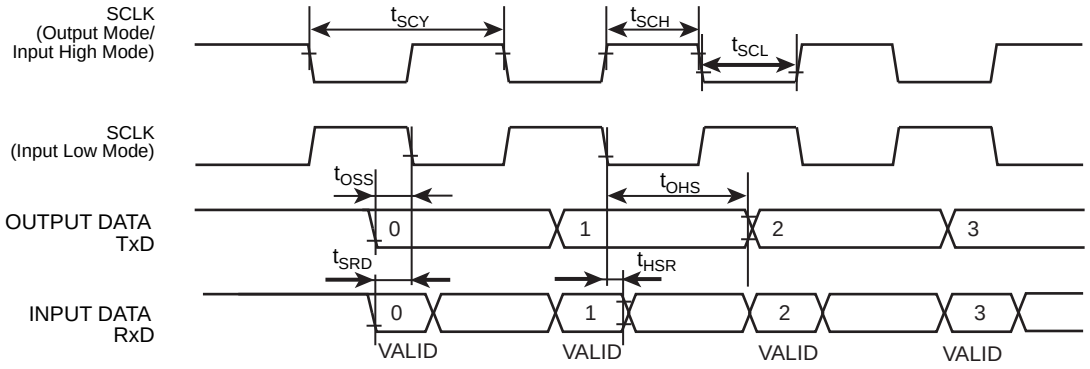
Note 1: SCLK rise/fall : SCLK rise mode uses the rise timing of SCLK. SCLK fall mode uses the fall timing of SCLK.

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	−	



19.6.3 Serial Bus Interface (I2C/SIO)

19.6.3.1 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) = $(2^{n-1} + 58)/x$**

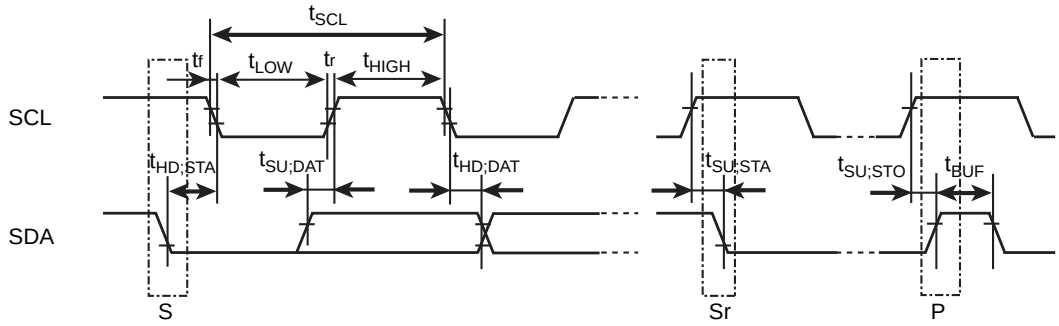
Note 2: **SCL clock High width (output) = $(2^{n-1} + 14)/x$ On I2C-bus specification, Maximum Speed of Standard Mode is 100kHz, Fast mode is 400khz. Internal SCL Frequency setting should comply with Note1 & Note2 shown above.**

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

19.6.3.2 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		fsys = 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]

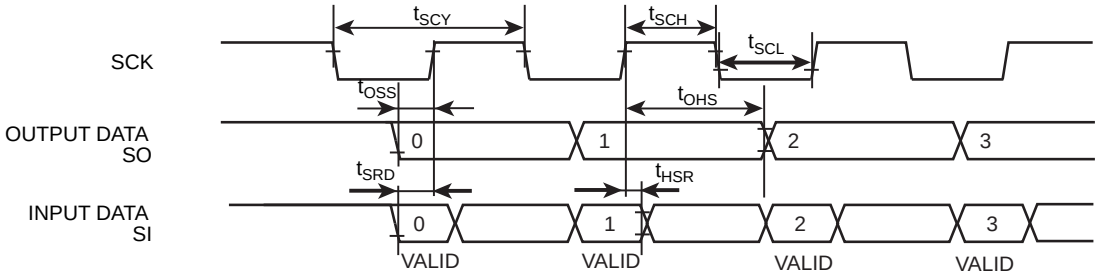
Parameter	Symbol	Equation		fsys = 40 MHz		48 MHz		Unit
		Min	Max.	Min	Max.	Min.	Max.	
SCK Clock High width (input)	t_{SCH}	4x	−	120 (Note2)	−	108 (Note2)	−	ns
SCK Clock Low width (input)	t_{SCL}	4x	−	120 (Note2)	−	108 (Note2)	−	
SCK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	−	240	−	215	−	
Output Data → SCLK 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.**



19.6.4 Event Counter

In the table below, the letter x represents the TMRB 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

19.6.5 Capture

In the table below, the letter x represents the TMRB 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

19.6.6 External Interrupt

In the table below, the letter x represents the fsys cycle time.

1. Except STOP1 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

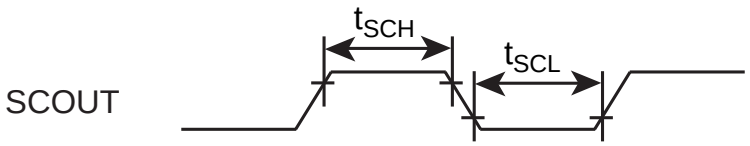
19.6.7 $\overline{NMI}$

Parameter	Symbol	Min.	Max.	Unit
$\overline{NIM}$ low level pulse width	t_{INTCL}	100	−	ns

19.6.8 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.**



19.6.9 $\overline{ADTRG}$ Tringger Input Pin AC Characteristic

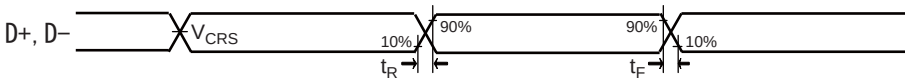
In the table below, the letter x represents fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		40MHz		48MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
Low level pulse width	T_{adl}	2 x + 20	−	32.5	−	32.5	−	ns
High level pulse interval	T_{adh}	2 x + 20	−	32.5	−	32.5	−	

19.6.10 USB Timing

DVDD3A = DVDD3C = AVDD3 = RVDD3 = 3.0 to 3.45V, fsys = 48MHz

Parameter	Symbol	Min.	Max.	Unit
D+,D−Supply rise time	t_R	4	20	ns
D+,D−Supply withdraw time	t_F	4	20	
Data Line crossover voltage	V_{CRS}	1.3	2.0	V



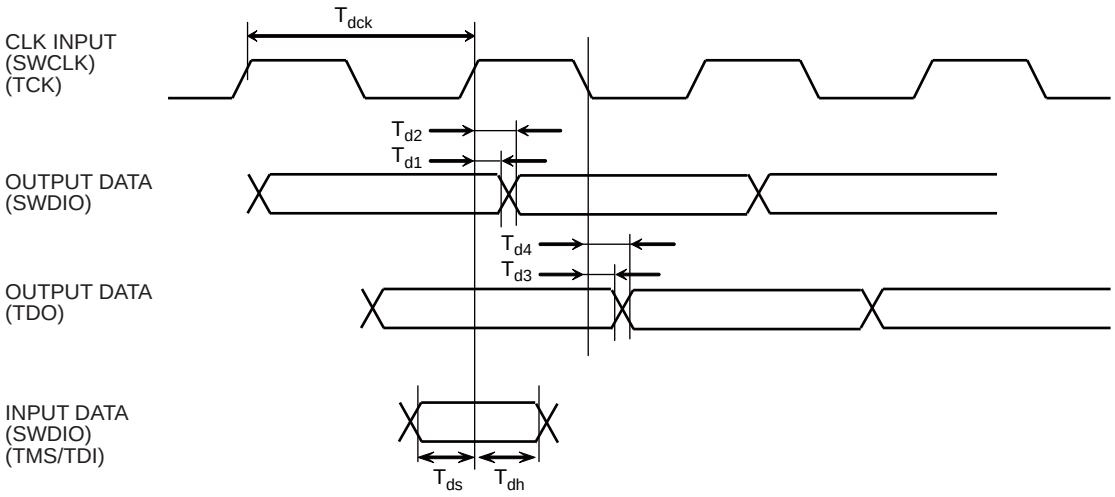
19.6.11 Debug Communication

19.6.11.1 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	−	

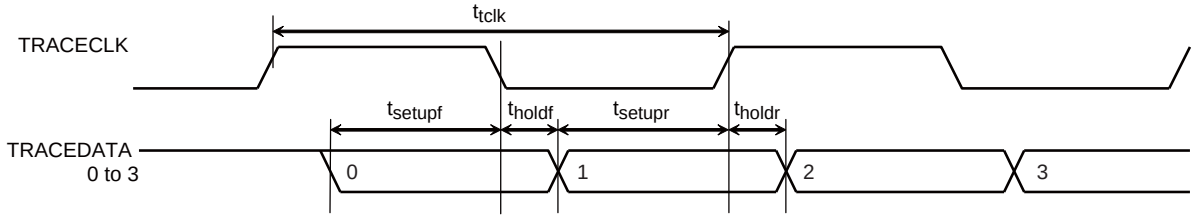
19.6.11.2 JTAG Interface

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	−	



19.6.12 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



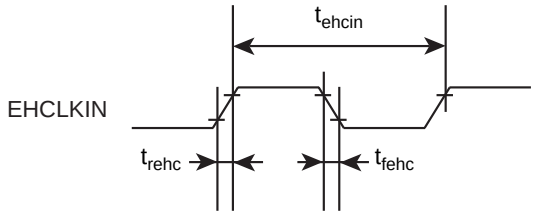
19.6.13 On chip oscillator

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillating frequency	IHOSC	Ta = -40 to 85°C	9	10	11	MHz

Note: The on-chip-oscillator can not be used as system clock (fsys) which is required oscillation accuracy.

19.6.14 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



19.6.15 Flash Memory Characteristics

Parameter	condition	Min.	Typ.	Max.	unit
Guarantee on Flash-Mmory Rewriting	DVDD3A = AVDD3 = RVDD3 = 2.7 V to 3.6 V, DVDD3C = 2.7 V to 3.6 V, Ta = 0 to 70 °C	-	-	100	times

19.7 Recommended Oscillation Circuit

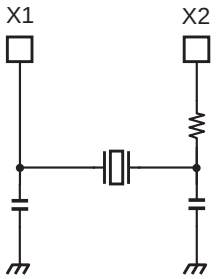


Figure 19-1 High-frequency oscillation connection

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 TMPM365FYXBG has been evaluated by the oscillator vender below. Please refer this information when selecting external parts

19.7.1 Ceramic oscillator

The TMPM365FYXBG recommends the high-frequency oscillator by Murata

Manufacturing Co., Ltd.

Please refer to the Murata Website for details.

19.7.2 Crystal oscillator

The TMPM365FYXBG recommends the high-frequency oscillator by KYOCERA Corporation.

Please refer to the KYOCERA Website for details.

19.7.2.1 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 .

19.8 Handling Precaution

19.8.1 Voltage level of power supply at power-on

The rising of power supply in power-on must be less than the value in below tabel.

TMPM365FYXBG has some power supply pin. They must be supplied the power at same time.

And the external reset pin ($\overline{\text{RESET}}$) must be "Low" level at power-on.

Power supply pin = DVDD3A, DVDD3C, AVDD3, RVDD3
C = 0.1μF
Ta = -40 ~ 85°C

Parameter	condition	Min.	Typ.	Max.	unit
The rising of power supply in power-on	0V → 2.7 V to 3.6 V	-	-	10	mV/μs

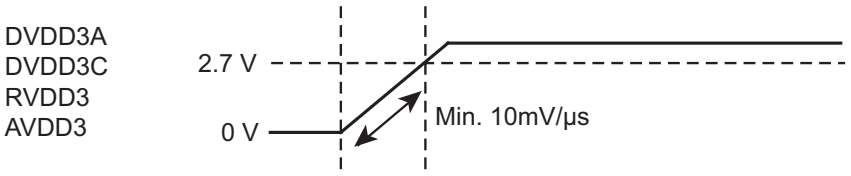
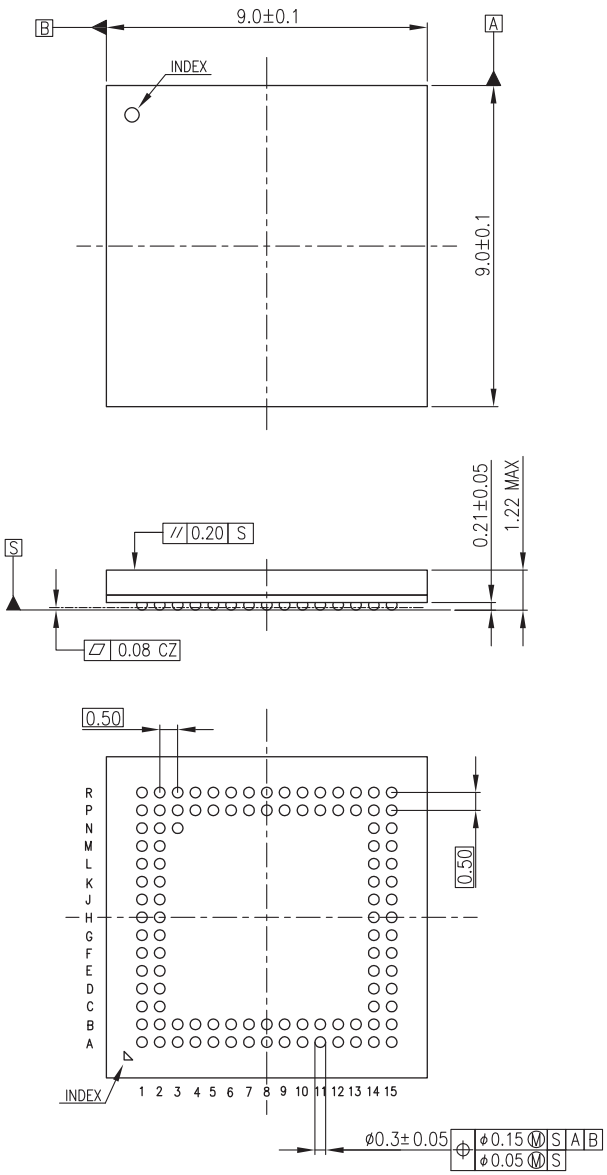


Figure 19-2 Voltage level of power supply at power-on

20. Package Dimensions

Type: P-LFBGA105-0909-050-001

"Unit:mm"



• RESTRICTIONS ON PRODUCT USE

Toshiba Corporation and its subsidiaries and affiliates are collectively referred to as “TOSHIBA”.
Hardware, software and systems described in this document are collectively referred to as “Product”.

- TOSHIBA reserves the right to make changes to the information in this document and related Product without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating a operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS' PRODUCT DESIGN OR APPLICATIONS.**
- **PRODUCT IS NEITHER INTENDED NOR WARRANTED FOR USE IN EQUIPMENTS OR SYSTEMS THAT REQUIRE EXTRAORDINARILY HIGH LEVELS OF QUALITY AND/OR RELIABILITY, AND/OR A MALFUNCTION OR FAILURE OF WHICH MAY CAUSE LOSS OF HUMAN LIFE, BODILY INJURY, SERIOUS PROPERTY DAMAGE AND/OR SERIOUS PUBLIC IMPACT ("UNINTENDED USE").** Except for specific applications as expressly stated in this document, Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, lifesaving and/or life supporting medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, and devices related to power plant. **IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT.** For details, please contact your TOSHIBA sales representative or contact us via our website.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. **TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.**