TOSHIBA

32 Bit RISC Microcontroller
TX00 Series

TMPM037FWUG

TOSHIBA CORPORATION
Semiconductor & Storage Products Company

ARM

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

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

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

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

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

Base Address = 0x0000_0000

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

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

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

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

1.2.2 SAMCR(Control register)

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

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

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

R / W	READ WRITE
R	READ
W	WRITE

c. Data descriptopn

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

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

d. Register descriptoption

Registers are described as shown below.

- Register name <Bit Symbol>

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

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

- Register name [Bit]

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

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

General precautions on the use of Toshiba MCUs

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

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

1. The MCUs' operation at power-on

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

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

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

2. Unused pins

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

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

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

3. Clock oscillation stability

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

Revision History

Date	Revision	Comment
2014/11/05	1	First Release
2022/09/30	2	Contents Revised
2023/07/31	3	Contents Revised

Table of Contents

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

TMPM037FWUG

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 Pin names and Functions for each peripheral function, control pin and power supply pin....6
1.4.1.1 Peripheral functions
1.4.1.2 Debug function
1.4.1.3 Control function
1.4.1.4 Power supply pins
1.4.2 Pin names and Function of TMPM037FWUG....8
1.4.2.1 The detail for pin names and function list
1.4.2.2 PORT / Debug pin
1.4.2.3 Control pin
1.4.2.4 Power Supply pin

2. Product Information

2.1 Information of Each Peripheral Function....14
2.1.1 DMA Controller (DMAC)....14
2.1.1.1 DMA Request table
2.1.1.2 Peripheral function supported with Peripheral to Peripheral Transfer
2.1.1.3 DMA request control register (DMARQCTL)
2.1.1.4 DMACREDGE(DMAC request setting register)
2.1.1.5 DMACRCLR(DMAC request clear register)
2.1.2 16-bit Timer/Event Counter (TMRB)....16
2.1.3 16-bit Timer A(TMR16A)....17
2.1.4 Serial Channel (SIO/UART)....18
2.1.5 I2C Bus(I2C)....18
2.1.6 Analog/Digital Converter (ADC)....18
2.1.7 Debug Interface....19

3. Processor Core

3.1 Information on the processor core....21
3.2 Configurable Options....21
3.3 Exceptions/ Interruptions....22
3.3.1 Number of Interrupt Inputs....22
3.3.2 SysTick....22
3.3.3 SYSRESETREQ....22
3.3.4 LOCKUP....22
3.4 Events....22
3.5 Power Management....22

4. Memory Map

4.1 Memory map 25
4.2 Bus Matrix 27
4.2.1 Structure 28
4.2.1.1 Single chip mode
4.2.1.2 SIngle boot mode
4.2.2 Connection table 29
4.2.2.1 Code area / SRAM area
4.2.2.2 Peripheral area
4.2.3 Address lists of peripheral functions 31

5. Reset Operation

5.1 Cold Reset 34
5.1.1 Cold Reset by RESET pin 34
5.1.2 Cold Reset with power-on-reset circuit 35
5.2 Warm Reset 35
5.3 After reset 35

6. Clock/Mode control

6.1 Features 37
6.2 Registers 38
6.2.1 Register List 38
6.2.2 CGSYSCR (System control register) 39
6.2.3 CGOSCCR (Oscillation control register) 40
6.2.4 CGSTBYCR (Standby control register) 42
6.2.5 CGPLLSEL(PLL Selection Register) 43
6.2.6 CGPROTECT (Protect register) 44
6.3 Clock control 45
6.3.1 Clock Type 45
6.3.2 Initial Values after Reset 45
6.3.3 Clock system Diagram 46
6.3.4 Warm-up function 47
6.3.5 Clock Multiplication Circuit (PLL) 49
6.3.5.1 How to configure the PLL function
6.3.5.2 The sequence of PLL setting
6.3.6 System clock 51
6.3.6.1 Clock setting
6.3.6.2 When using external oscillator
6.3.7 Prescaler Clock Control 53
6.4 Modes and Mode Transitions 54
6.4.1 Mode Transitions 54
6.5 Operation mode 55
6.5.1 NORMAL mode 55
6.6 Low Power Consumption Modes 55
6.6.1 IDLE mode 55
6.6.2 STOP1 mode 56
6.6.3 Low power Consumption Mode Setting 57
6.6.4 Operational Status in Each Mode 58
6.6.5 Releasing the Low Power Consumption Mode 59
6.6.6 Warm-up 60
6.6.7 Clock Operations in Mode Transition 61
6.6.7.1 Transition of operation modes: NORMAL → STOP1 → NORMAL
6.6.8 Precaution on Transition to the Low-power Consumption Mode 62
6.6.8.1 Case when the MCU Enters IDLE or STOP1 Mode

7. Exceptions

7.1 Overview 63
7.1.1 Exception Types 63
7.1.2 Handling Flowchart 64
7.1.2.1 Exception Request and Detection
7.1.2.2 Exception Handling and Branch to the Interrupt Service Routine (Pre-emption)
7.1.2.3 Executing an ISR
7.1.2.4 Exception exit
7.2 Reset Exceptions 69
7.3 Non-Maskable Interrupts (NMI) 70
7.4 SysTick 70
7.5 Interrupts 71
7.5.1 Interrupt Sources 71
7.5.1.1 Interrupt Route
7.5.1.2 Generation
7.5.1.3 Transmission
7.5.1.4 Precautions when using external interrupt pins
7.5.2 List of Interrupt Sources 73
7.5.2.1 Active level
7.5.3 Interrupt Handling 75
7.5.3.1 Flowchart
7.5.3.2 Preparation
7.5.3.3 Detection by Clock Generator
7.5.3.4 Detection by CPU
7.5.3.5 CPU processing
7.5.3.6 Interrupt Service Routine (ISR)
7.6 Exception/Interrupt-Related Registers 80
7.6.1 Register List 80
7.6.2 NVIC registers 81
7.6.2.1 SysTick Control and Status Register
7.6.2.2 SysTick Reload Value Register
7.6.2.3 SysTick Correct Value Register
7.6.2.4 SysTick Calibration Value Register
7.6.2.5 Interrupt Set-Enable Register 1
7.6.2.6 Interrupt Clear-Enable Register 1
7.6.2.7 Interrupt Set-Pending Register 1
7.6.2.8 Interrupt Clear-Pending Register 1
7.6.2.9 Interrupt Priority Register
7.6.2.10 Application Interrupt and reset Control Register
7.6.2.11 System Handler Priority Register
7.6.2.12 System Handler Control and State Register
7.6.3 Clock generator registers 92
7.6.3.1 CGIMCGA(CG Interrupt Mode Control register A)
7.6.3.2 CGIMCGB(CG Interrupt Mode Control register)
7.6.3.3 CGICRCG(CG Interrupt Request Clear register)
7.6.3.4 CGRSTFLG (Reset Flag Register)
7.6.3.5 CGNMIFLG(NMI flag register)

8. DMA Controller(DMAC)

8.1 Overview 99
8.2 DMA transfer type 100
8.3 Block diagram 101
8.4 Description of Registers 102
8.4.1 DMAC register list 102
8.4.2 DMACxIntStatus (DMAC Interrupt Status Register) 103
8.4.3 DMACxIntTCStatus (DMAC Interrupt Terminal Count Status Register) 104
8.4.4 DMACxIntTCClear (DMAC Interrupt Terminal Count Clear Register) 105
8.4.5 DMACxIntErrorStatus (DMAC Interrupt Error Status Register) 106
8.4.6 DMACxIntErrClr (DMAC Interrupt Error Clear Register) 107
8.4.7 DMACxRawIntTCStatus (DMAC Raw Interrupt Terminal Count Status Register) 108

8.4.8 DMACxRawIntErrorStatus (DMAC Raw Error Interrupt Status Register)....109
8.4.9 DMACxEnbldChns (DMAC Enabled Channel Register)....110
8.4.10 DMACxSoftBReq (DMAC Software Burst Request Register)....111
8.4.11 DMACxSoftSReq (DMAC Software Single Request Register)....113
8.4.12 DMACxConfiguration (DMAC Configuration Register)....115
8.4.13 DMACxCnSrcAddr (DMAC Channelx Source Address Register)....116
8.4.14 DMACxCnDestAddr (DMAC Channelx Destination Address Register)....117
8.4.15 DMACxCnLLI (DMAC Channelx Linked List Item Register)....118
8.4.16 DMACxCnControl (DMAC Channeln Control Register)....119
8.4.17 DMACxCnConfiguration (DMAC Channel n Configuration Register)....121
8.5 Special Functions....123
8.5.1 Scatter/gather function....123
8.5.2 Linked list operation....124

9. Input/Output port

9.1 Registers....127
9.1.1 Register list....128
9.1.2 Port function and setting list....129
9.1.2.1 PORT A
9.1.2.2 PORT B
9.1.2.3 PORT C
9.1.2.4 PORT D
9.1.2.5 PORT E
9.1.2.6 PORT F
9.1.2.7 PORT G
9.1.3 Block Diagrams of Ports....137
9.1.3.1 Port Type
9.1.3.2 Type FT1
9.1.3.3 Type FT4
9.1.3.4 Type FT5
9.1.3.5 Type FT6

10. 16-bit Timer / Event Counters (TMRB)

10.1 Outline....143
10.2 Block Diagram....144
10.3 Registers....145
10.3.1 Register List....145
10.3.2 TBxEN (Enable Register)....146
10.3.3 TBxRUN (RUN Register)....147
10.3.4 TBxCR (Control Register)....148
10.3.5 TBxMOD (Mode Register)....149
10.3.6 TBxFFCR (Flip-Flop Control Register)....150
10.3.7 TBxST (Status Register)....151
10.3.8 TBxIM (Interrupt Mask Register)....152
10.3.9 TBxUC (Up-counter Capture Register)....153
10.3.10 TBxRG0 (Timer Register 0)....154
10.3.11 TBxRG1 (Timer Register 1)....154
10.3.12 TBxCP0 (Capture register 0)....155
10.3.13 TBxCP1 (Capture Register 1)....155
10.3.14 TBxDMA(DMA request enable register)....156
10.4 Description of Operation....157
10.4.1 Prescaler....157
10.4.2 Up-counter (UC)....157
10.4.2.1 Source clock
10.4.2.2 Counter start / stop
10.4.2.3 Counter Clear
10.4.2.4 Up-Counter Overflow
10.4.3 Timer Registers (TBxRG0, TBxRG1)....158
10.4.4 Capture Control....158
10.4.5 Capture Registers (TBxCP0, TBxCP1)....158
10.4.6 Up-Counter Capture Register (TBxUC)....159

10.4.7 Comparators (CP0, CP1)....159
10.4.8 Timer Flip-Flop (TBxFF0)....159
10.4.9 Capture Interrupt (INTTBxCAP0, INTTBxCAP1)....159
10.4.10 DMA Request....159
10.5 Description of Operation for each mode....160
10.5.1 Interval Timer Mode....160
10.5.2 Event Counter Mode....160
10.5.3 Programmable Pulse Generation (PPG) Output Mode....161
10.5.4 Programmable Pulse Generation (PPG) External Trigger Output Mode....163
10.6 Applications Using Capture Function....165
10.6.1 Frequency Measurement....165
10.6.2 Pulse Width Measurement....167

11. 16-Bit Timer A (TMR16A Ver.B)

11.1 Outline....169
11.2 Block Diagram....169
11.3 Registers....170
11.3.1 Register List....170
11.3.1.1 T16AxEN (Enable Register)
11.3.1.2 T16AxRUN (RUN Register)
11.3.1.3 T16AxCR (Control Register)
11.3.1.4 T16AxRG (Timer Register)
11.3.1.5 T16AxCP (Capture Register)
11.4 Operation Description....174
11.4.1 Timer Operation....174
11.4.2 T16AxOUT Control....174
11.4.3 Read Capture....174
11.4.4 Automatic Stop....174

12. Serial Channel with 4bytes FIFO (SIO/UART)

12.1 Overview....177
12.2 Configuration....178
12.3 Registers Description....179
12.3.1 Registers List....179
12.3.2 SCxEN (Enable Register)....180
12.3.3 SCxBUF (Buffer Register)....181
12.3.4 SCxCR (Control Register)....182
12.3.5 SCxMOD0 (Mode Control Register 0)....184
12.3.6 SCxMOD1 (Mode Control Register 1)....185
12.3.7 SCxMOD2 (Mode Control Register 2)....186
12.3.8 SCxBRCR (Baud Rate Generator Control Register)....188
12.3.9 SCxBRADD (Baud Rate Generator Control Register 2)....189
12.3.10 SCxFCNF (FIFO Configuration Register)....190
12.3.11 SCxRFC (Receive FIFO Configuration Register)....192
12.3.12 SCxTFC (Transmit FIFO Configuration Register) 193
12.3.13 SCxRST (Receive FIFO Status Register)....194
12.3.14 SCxTST (Transmit FIFO Status Register)....195
12.3.15 SCxDMA (DMA request enable register)....196
12.4 Operation in Each Mode....197
12.5 Data Format....198
12.5.1 Data Format List....198
12.5.2 Parity Control....199
12.5.2.1 Transmission
12.5.2.2 Receiption
12.5.3 STOP Bit Length....199
12.6 Clock Control....200
12.6.1 Prescaler....200

12.6.2 Serial Clock Generation Circuit....200
12.6.2.1 Baud Rate Generator
12.6.2.2 Clock Selection Circuit
12.7 Transmit/Receive Buffer and FIFO....204
12.7.1 Configuration....204
12.7.2 Transmit/Receive Buffer....204
12.7.3 Initialize Transmit Buffer....205
12.7.4 FIFO....205
12.8 Status Flag....206
12.9 Error Flag....206
12.9.1 OERR Flag....206
12.9.2 PERR Flag....207
12.9.3 FERR Flag....207
12.10 Receive....208
12.10.1 Receive Counter....208
12.10.2 Receive Control Unit....208
12.10.2.1 I/O interface mode
12.10.2.2 UART Mode
12.10.3 Receive Operation....208
12.10.3.1 Receive Buffer
12.10.3.2 Receive FIFO Operation
12.10.3.3 I/O interface mode with clock output mode
12.10.3.4 Read Received Data
12.10.3.5 Wake-up Function
12.10.3.6 Overrun Error
12.11 Transmit....212
12.11.1 Transmit Counter....212
12.11.2 Transmit Control....212
12.11.2.1 In I/O Interface Mode
12.11.2.2 In UART Mode
12.11.3 Transmit Operation....213
12.11.3.1 Operation of Transmit Buffer
12.11.3.2 Transmit FIFO Operation
12.11.3.3 Transmit in I/O interface Mode with Clock Output Mode
12.11.3.4 Level of SCxTXD pin after the last bit is output in I/O interface mode
12.11.3.5 Under-run error
12.11.3.6 Data Hold Time In the I/O interface mode with clock input mode
12.12 Handshake function....217
12.13 Interrupt/Error Generation Timing....218
12.13.1 Receive Interrupts....218
12.13.1.1 Single Buffer / Double Buffer
12.13.1.2 FIFO
12.13.2 Transmit interrupts....219
12.13.2.1 Singe Buffer / Double Buffer
12.13.2.2 FIFO
12.13.3 Error Generation....220
12.13.3.1 UART Mode
12.13.3.2 I/O Interface Mode
12.14 DMA Request....221
12.15 Software Reset....222
12.16 Operation in Each Mode....223
12.16.1 Mode 0 (I/O interface mode)....223
12.16.1.1 Transmit
12.16.1.2 Receive
12.16.1.3 Transmit and Receive (Full-duplex)
12.16.2 Mode 1 (7-bit UART mode)....234
12.16.3 Mode 2 (8-bit UART mode)....234
12.16.4 Mode 3 (9-bit UART mode)....235
12.16.4.1 Wakeup function
12.16.4.2 Protocol

13. I2C Bus Interface

13.1 Configuration....238

13.2 I2C Bus mode ... 239
13.2.1 I2C Bus Mode Data Format ... 239
13.3 Register ... 241
13.3.1 Registers for each channel ... 241
13.3.2 I2CxCR1(Control register 1) ... 241
13.3.3 I2CxDBR (Serial bus interface data buffer register) ... 243
13.3.4 I2CxAR (I2Cbus address register) ... 244
13.3.5 I2CxCR2(Control register 2) ... 245
13.3.6 I2CxSR (Status Register) ... 246
13.3.7 I2CxPRS(Prescaler Clock setting register) ... 247
13.3.8 I2CxIE(Interrupt Enable register) ... 248
13.3.9 I2CxIR(Interruppt register) ... 248
13.4 Control in the I2C Bus Mode ... 249
13.4.1 Serial Clock ... 249
13.4.1.1 Clock source
13.4.1.2 Clock Synchronization
13.4.2 Selection for a Slave Address Match Detection or General Call Detection ... 251
13.4.3 Setting the Acknowledgement Mode ... 252
13.4.4 Setting the Number of Bits per Transfer ... 252
13.4.5 Slave Addressing and Address Recognition Mode ... 252
13.4.6 Configuring the I2C as a Master or a Slave ... 253
13.4.7 Configuring the I2C as a Transmitter or a Receiver ... 253
13.4.8 Generating Start and Stop Conditions ... 254
13.4.9 Interrupt Service Request and Release ... 256
13.4.10 I2C Bus mode ... 256
13.4.11 Software Reset ... 256
13.4.12 Arbitration Lost Detection Monitor ... 256
13.4.13 Slave Address Match Detection Monitor ... 258
13.4.14 General-call Detection Monitor ... 258
13.4.15 Last Received Bit Monitor ... 258
13.4.16 Data Buffer Register (I2CxDBR) ... 259
13.5 Data Transfer Procedure in the I2C Bus Mode ... 260
13.5.1 Device Initialization ... 260
13.5.2 Generating the Start Condition and a Slave Address ... 260
13.5.2.1 Master mode
13.5.2.2 Slave mode
13.5.3 Transferring a Data Word ... 263
13.5.3.1 Master mode (<MST> = "1")
13.5.3.2 Slave mode (<MST> = "0")
13.5.4 Generating the Stop Condition ... 269
13.5.5 Restart Procedure ... 269
13.6 Notice on usage ... 271
13.6.1 Register values after a Software Reset ... 271

14. 10-bit Analog/Digital Converter (ADC)

14.1 Outline ... 273
14.2 Configuration ... 273
14.3 Registers ... 275
14.3.1 Register list ... 275
14.3.2 ADCLK (Conversion Clock Setting Register) ... 276
14.3.3 ADMOD0 (Mode Control Register 0) ... 277
14.3.4 ADMOD1 (Mode Control Register 1) ... 278
14.3.5 ADMOD2 (Mode Control Register 2) ... 279
14.3.6 ADMOD3 (Mode Control Register 3) ... 281
14.3.7 ADMOD4 (Mode Control Register 4) ... 282
14.3.8 ADMOD5 (Mode Control Register 5) ... 283
14.3.9 ADMOD6 (Mode Control Register 6) ... 284
14.3.10 ADREGn (Conversion Result Register n: n = 0 to 11) ... 285
14.3.11 ADREGSP (AD Conversion Result Register SP) ... 286
14.3.12 ADCMP0 (AD Conversion Result Comparison Register 0) ... 287
14.3.13 ADCMP1 (AD Conversion Result Comparison Register 1) ... 287
14.4 Description of Operations ... 288

14.4.1 Analog Reference Voltage....288
14.4.2 AD Conversion Mode....288
14.4.2.1 Normal AD conversion
14.4.2.2 Top-priority AD conversion
14.4.3 AD Monitor Function....289
14.4.4 Selecting the Input Channel....290
14.4.5 AD Conversion Details....290
14.4.5.1 Starting AD Conversion
14.4.5.2 AD Conversion
14.4.5.3 Top-priority AD conversion during normal AD conversion
14.4.5.4 Stopping Repeat Conversion Mode
14.4.5.5 Reactivating normal AD conversion
14.4.5.6 Conversion completion
14.4.5.7 Interrupt generation timings and AD conversion result storage register
14.4.5.8 DMA Request
14.4.5.9 Cautions

15. Low Voltage detection circuit (LVD)

15.1 Configuration....297
15.2 Registers....298
15.2.1 Register list....298
15.2.2 LVDCR1 (LVD detection control register 1)....298
15.3 Operation....300
15.3.1 Selecting detection voltage and enabling voltage detection operation....300
15.3.2 Reset by Detecting a supply voltage....300
15.3.3 Interrupt by Detecting a supply voltage....300
15.3.4 Detecting Status....300

16. Watchdog Timer (WDT)

16.1 Configuration....301
16.2 Register....302
16.2.1 Register List....302
16.2.2 WDMOD (Watchdog Timer Mode Register) 302
16.2.3 WDCR (Watchdog Timer Control Register)....303
16.3 Description of Operation....304
16.3.1 Basic Operation....304
16.3.2 Operation Mode and Status....304
16.3.3 Operation when malfunction (runaway) is detected....304
16.3.3.1 INTWDT interrupt generation
16.3.3.2 Internal Reset Generation
16.4 Control of the watchdog timer....305
16.4.1 Disable control....305
16.4.2 Enable control....305
16.4.3 Watchdog timer clearing control....305
16.4.4 Detection time of watchdog timer....305

17. Flash Memory Operation

17.1 Features....307
17.1.1 Memory Size and Configuration....307
17.1.2 Function....308
17.1.3 Operation Mode....308
17.1.3.1 Mode Description
17.1.3.2 Mode Determination
17.1.4 Memory Map....310
17.1.5 Protect/Security Function....310
17.1.5.1 Protect Function
17.1.5.2 Security Function

17.1.6 Register....312
17.1.6.1 Register List
17.1.6.2 FCSR (Flash status register)
17.1.6.3 FCSECBIT (Security bit register)
17.1.6.4 FCPSRA (Flash protect status register)
17.2 Detail of Flash Memory....314
17.2.1 Function....314
17.2.2 Operation Mode of Flash Memory....314
17.2.3 Hardware Reset....314
17.2.4 How to Execute Command....315
17.2.5 Command Description....315
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....319
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....323
17.2.7.1 Automatic Program
17.2.7.2 Automatic Erase
17.3 How to Reprogram Flash using Single Boot Mode....325
17.3.1 Mode Setting....325
17.3.2 Interface Specification....325
17.3.3 Restrictions on Internal Memories....326
17.3.4 Operation Command....326
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....327
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....334
17.3.7 Transfer Format of Flash memory Chip Erase and Protect Bit Erase....336
17.3.8 Boot Program Whole Flowchart....338
17.3.9 Reprogramming Procedure of Flash using reprogramming algorithm in the on-chip BOOT ROM....339
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....342
17.4.1 (1-A) Procedure that a Programming Routine Stored in Flash memory....342
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 346
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. Debug Interface

18.1 Specification Overview....351

18.2 SWJ-DP.....351
18.3 Peripheral Functions in Halt Mode.....351
18.4 Connection with a Debug Tool.....352
18.4.1 About connection with debug tool.....352
18.4.2 Important points of using debug interface pins used as general-purpose ports.....352

19. Port Section Equivalent Circuit Schematic

19.1 PORT pin.....354
19.2 Analog pin.....354
19.3 Control pin.....355
19.4 Clock pin.....355
19.5 Test pin.....355

20. Electrical Characteristics

20.1 Absolute Maximum Ratings.....357
20.2 DC Electrical Characteristics (1/2).....358
20.3 DC Electrical Characteristics (2/2).....360
20.4 10-bit AD Converter electrical Characteristics.....361
20.5 AC Electrical Characteristics.....362
20.5.1 Serial channel (SIO/UART).....362
20.5.1.1 AC measurement Condition
20.5.1.2 AC Electrical Characteristics (I/O Interface Mode)
20.5.2 I2C interface (I2C).....364
20.5.2.1 AC measurement Condition
20.5.2.2 AC Electrical Characteristics
20.5.3 16-bit Timer / Event counter (TMRB).....366
20.5.3.1 Event Counter
20.5.3.2 Capture
20.5.4 External Interrupt.....367
20.5.4.1 AC Measurement Condition
20.5.4.2 AC Electrical Characteristics
20.5.5 Debug Communication.....368
20.5.5.1 AC Measurement Condition
20.5.5.2 SWD interface
20.5.6 On-chip Oscillator Characteristic.....369
20.5.7 External Oscillator.....369
20.5.8 External Clock Input.....370
20.5.9 Flash Characteristic.....370
20.5.10 Noise Filter Characteristic.....370
20.6 Recommended Oscillation Circuit.....371
20.6.1 Ceramic Oscillator.....371
20.6.2 Precautions for designing printed circuit board.....371
20.7 Handling Precaution.....372
20.7.1 Voltage level of power supply at power-on.....372
20.7.2 Voltage droup during operations.....372
20.7.3 Operating Voltage on Startup 372

21. Package Dimensions

CMOS #!Undefined!#-Bit Microcontroller

TMPM037FWUG

The TMPM037FWUG is a 32-bit RISC microprocessor series with an ARM®Cortex®-M0 microprocessor core.

Features of the TMPM037FWUG are as follows.

1.1 Features

1. ARM®Cortex®-M0 microprocessor core
   a. Improved code efficiency has been realized through the use of Thumb®-2 instruction.
   b. Both high performance and low power consumption have been achieved.

      [High performance]

      - A 32-bit multiplication (32 x 32=32 bit) can be executed with one clock.

      [Low power consumptions]

      - 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. Endian: Little endian

3. On Chip program memory and data memory
   - On chip Flash ROM: 128 Kbyte
   - On chip RAM: 16 Kbyte

4. DMA controller (DMAC): 2 channels / 1 unit
   - Transfer mode: Built-in memory, peripheral function and external memory.

5. Clock generator (CG)
   - External Clock input /external oscillation (8MHz to 20MHz)
   - on Chip PLL (2x)
   - Clock gear function: The high-speed clock can be divided into 1/1, 1/2, 1/4, 1/8, 1/16.

6. Standby mode
   - IDLE, STOP1

7. Interrupt source: The order of priority can be set to 4 levels.
   - Internal: 24factors
   - External: 6 factors

8. Input/output ports (PORT): 52 pins

   Input/Output pin: 51 pins

   Output pin: 1 pin

9. 16-bit timer (TMRB): 8 channels
   - 16-bit interval timer mode
   - 16-bit event counter mode
   - 16-bit PPG output (4 Phase Synchronous mode)
   - Input capture function

10. 16-bit timer (TMR16A): 2 channels

11. Watchdog timer (WDT): 1 channel

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

12. General-purpose serial interface (SIO/UART): 5 channels
    - Either UART mode or synchronous mode can be selected
    - Transmit FIFO: 4-stage 8-bit width, receive FIFO: 4-stage 8-bit width

13. I2C bus interface (I2C): 1 channel
    - Communication rate 100kbps / 400kbps

14. 10-bit AD converter (ADC):1 unit (8 channels)
    - Fixed channel/scan mode
    - Single/repeat mode
    - support Repeat conversion
    - AD monitoring

15. LVD/POR function: 1 unit

16. Maximum operating frequency: 20MHz

17. Debug Interface
    - SWD is supported.

18. Operating voltage range: 2.3 to 3.6V

19. Temperature range
    - −40°C to 85°C (except during Flash W/E)
    - 0°C to 70°C (during Flash W/E)

20. Package

    LQFP64 (10mm× 10mm, 0.5mm pitch)

1.2 Block Diagram

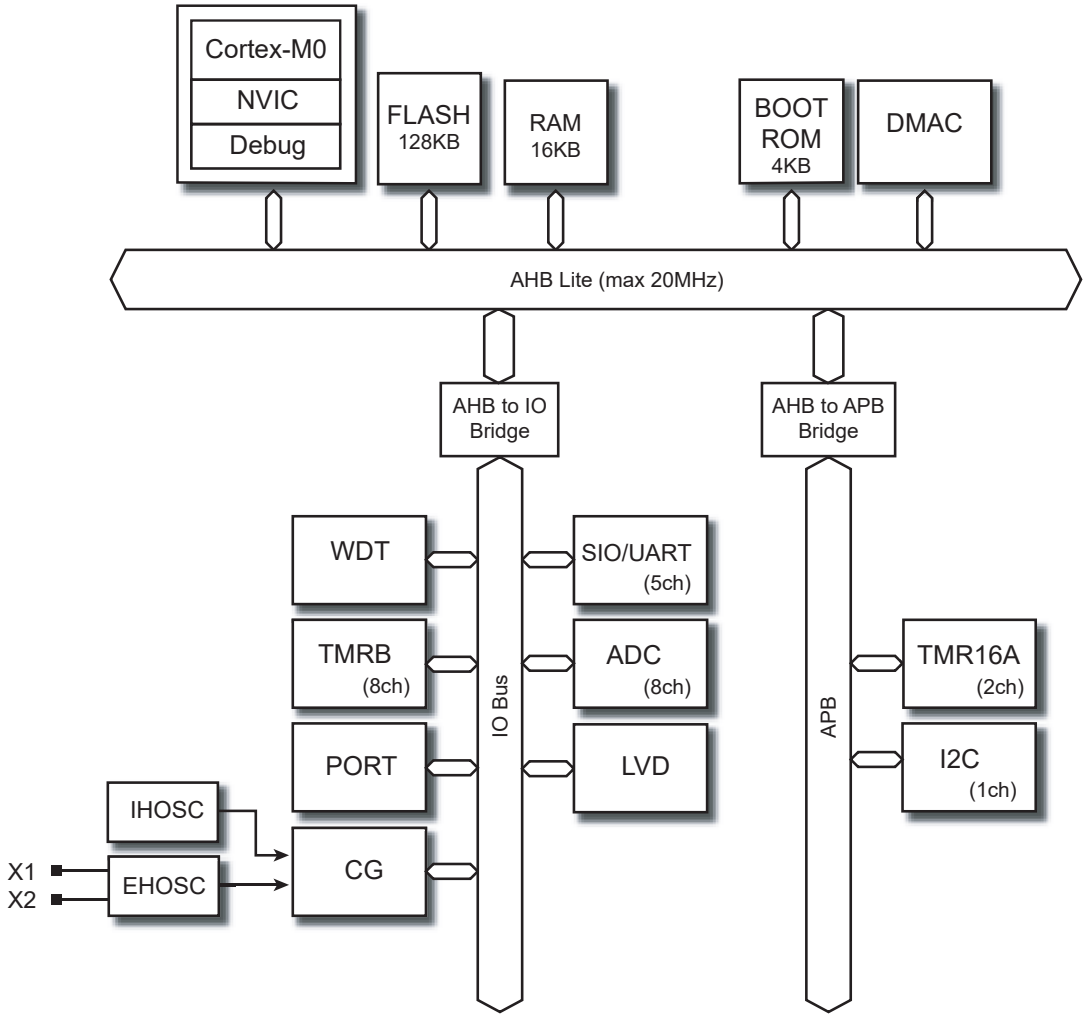


Figure 1-1 Block Diagram

1.3 Pin Layout (Top view)

Figure 1-2 shows the pin layout of TMPM037FWUG.

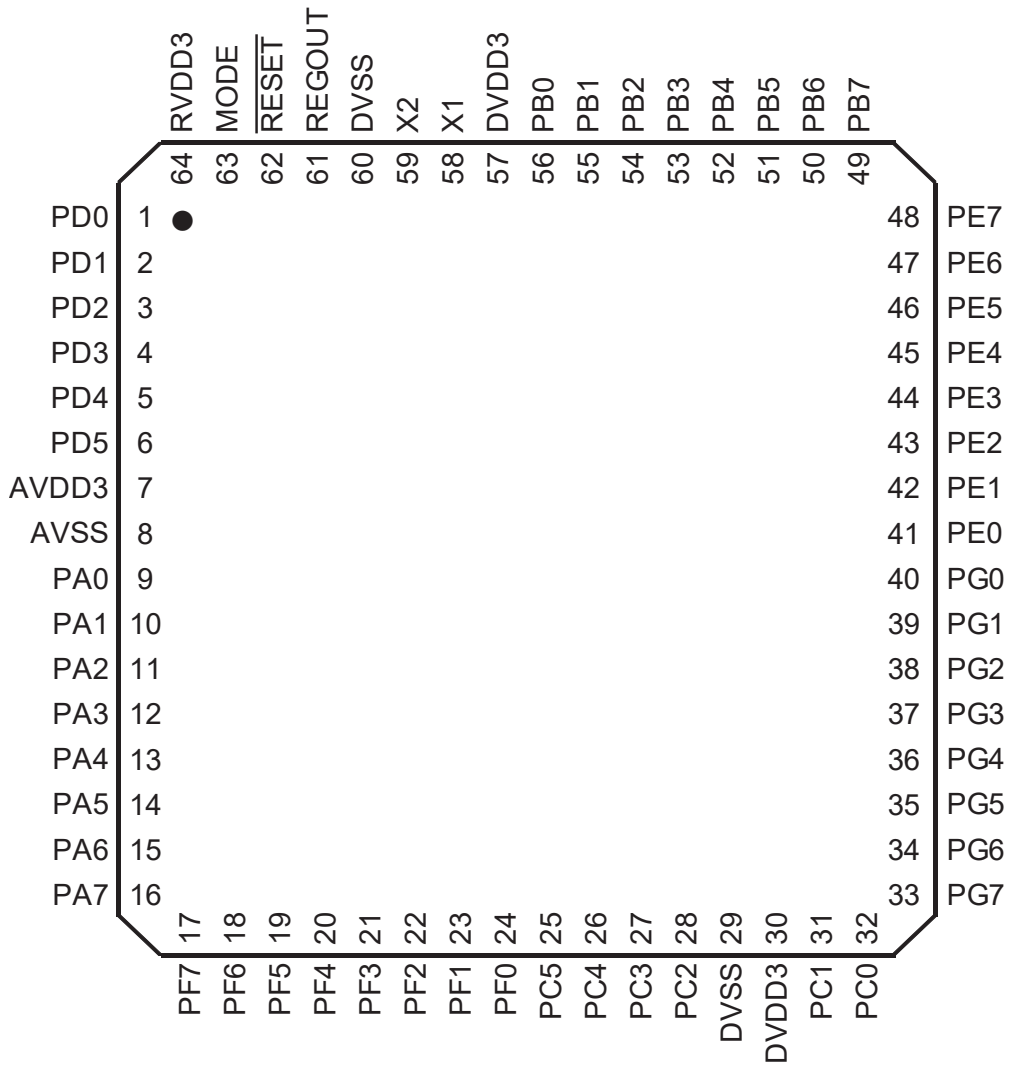


Figure 1-2 Pin Layout (LQFP64 TOP VIEW)

1.4 Pin names and Functions

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

1.4.1.1 Peripheral functions

Table 1-1 The number of pins and Pin names

Peripheral function	Pin name	Input or Output	Function
External interrupt	INTx	Input	External interrupt input pin x External interrupt input pin x has a noise filter (Filter width 30ns typ.)
16bit timer (TMRB)	TBxIN	Input	Input capture input pin
	TBxOUT	Output	output pin
16 bit timer (TMR16A)	T16AxOUT	Output	output pin
Serial channel (SIO/UART)	SCxTXD	Output	Data output pin
	SCxRXD	Input	Data input pin
	SCxSCLK	I/O	Clock input/ output pin
I2C bus interface (I2C)	I2CxSDA	I/O	Data input / output pin
	I2CxSCL	I/O	Clock input/ output pin
Analog digital converter	AINx	Input	Analog input pin

1.4.1.2 Debug function

Table 1-2 Pin name and functions

Pin name	Input or Output	Function
SWDIO	I/O	Serial wire data input and output pin
SWCLK	Input	Serial wire clock input pin

1.4.1.3 Control function

Table 1-3 Pin name and functions

Pin name	Input or Output	Function
X1	Input	Connected to a high-speed oscillator.
X2	Output	Connected to a high-speed oscillator.
MODE	Input	MODE pin MODE pin must be connected to GND.
$\overline{\text{RESET}}$	Input	Reset input pin
$\overline{\text{BOOT}}$	Input	BOOT mode control pin BOOT mode control pin is sampled at the rising edge of reset signal input pin. TMPM037FWUG changes Single Boot Mode when BOOT mode control pin is "Low". TMPM037FWUG changes Single Chip Mode when BOOT mode control pin is "High". Refer to "Flash memory" section for a detail.

1.4.1.4 Power supply pins

Table 1-4 Pin name and functions

Power supply pin name	Function
REGOUT	Pin connected with the capacitor(1μF) for the regulator.
RVDD3	Power supply pin for the regulator.
DVDD3	Power supply pin for the digital circuit.
DVSS	GND pin for the digital circuit.
AVDD3	Power supply pin for the analog circuit.
AVSS	GND pin for the analog circuit.

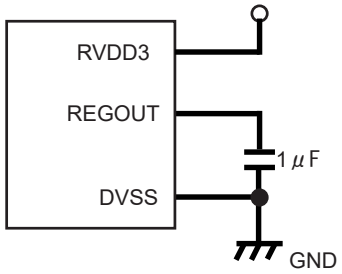


Figure 1-3 Capacitor for a regulator connection circuit

(Must be place on the shortest distance form a pin)

1.4.2 Pin names and Function of TMPM037FWUG

1.4.2.1 The detail for pin names and function list

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

1. Function A

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

2. Function B

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

3. Pin specification

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

   - SMT/CMOS: Type of input gate

     SMT: Schmitt input

     CMOS: CMOS input

   - OD: Programmable open drain output support

     Yes: supported

     N/A: not supported

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

     PU: Programmable Pull-Up supported

     PD: Programmable Pull-Down supported

   - 10mA: 10mA current drive port

     Yes: supported

     N/A: not supported

1.4.2.2 PORT / Debug pin

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

Pin-No.	PORT	Function A	Function B					Port Specification			
			1	2	3	4	5	PU/PD	OD	SMT/CMOS	10mA
PORTA											
9	PA0	AIN0						PU/PD	Yes	SMT	N/A
10	PA1	AIN1						PU/PD	Yes	SMT	N/A
11	PA2	AIN2						PU/PD	Yes	SMT	N/A
12	PA3	AIN3						PU/PD	Yes	SMT	N/A
13	PA4	AIN4						PU/PD	Yes	SMT	N/A
14	PA5	AIN5						PU/PD	Yes	SMT	N/A
15	PA6	AIN6						PU/PD	Yes	SMT	N/A
16	PA7	AIN7						PU/PD	Yes	SMT	N/A

Pin-No.	PORT	Function A	Function B					Port Specification			
			1	2	3	4	5	PU/PD	OD	SMT/CMOS	10mA
PORTB											
56	PB0	$\overline{\text{BOOT}}$						PU/PD	Yes	SMT	N/A
55	PB1		SC4 RXD	SW CLK				PU/PD	Yes	SMT	N/A
54	PB2		SC4 TXD	SWDIO				PU/PD	Yes	SMT	N/A
53	PB3		SC4 SCLK					PU/PD	Yes	SMT	N/A
52	PB4							PU/PD	Yes	SMT	N/A
51	PB5	INT0						PU/PD	Yes	SMT	N/A
50	PB6	INT1						PU/PD	Yes	SMT	N/A
49	PB7	INT2						PU/PD	Yes	SMT	N/A

Pin-No.	PORT	Function A	Function B					Port Specification			
			1	2	3	4	5	PU/PD	OD	SMT/CMOS	10mA
PORTC											
32	PC0		I2C0 SCL					PU/PD	Yes	SMT	N/A
31	PC1		I2C0 SDA					PU/PD	Yes	SMT	N/A
28	PC2		TB2 OUT					PU/PD	Yes	SMT	Yes
27	PC3		TB0 OUT					PU/PD	Yes	SMT	Yes
26	PC4		T16A0OUT					PU/PD	Yes	SMT	N/A
25	PC5		TB0IN					PU/PD	Yes	SMT	N/A

Pin-No.	PORT	Function A	Function B					Port Specification			
			1	2	3	4	5	PU/PD	OD	SMT/CMOS	10mA
PORTD											
1	PD0		TB3OUT					PU/PD	Yes	SMT	N/A
2	PD1		SC0SCLK					PU/PD	Yes	SMT	N/A
3	PD2		SC0RXD					PU/PD	Yes	SMT	N/A
4	PD3		SC0TXD					PU/PD	Yes	SMT	N/A
5	PD4		TB3IN					PU/PD	Yes	SMT	N/A
6	PD5							PU/PD	Yes	SMT	N/A

Pin-No.	PORT	Function A	Function B					Port Specification			
			1	2	3	4	5	PU/PD	OD	SMT/CMOS	10mA
PORTE											
41	PE0							PU/PD	Yes	SMT	N/A
42	PE1							PU/PD	Yes	SMT	N/A
43	PE2		SC2SCLK					PU/PD	Yes	SMT	N/A
44	PE3		SC2RXD					PU/PD	Yes	SMT	N/A
45	PE4		SC2TXD					PU/PD	Yes	SMT	N/A
46	PE5	INT5						PU/PD	Yes	SMT	N/A
47	PE6	INT4						PU/PD	Yes	SMT	N/A
48	PE7	INT3						PU/PD	Yes	SMT	N/A

Pin-No.	PORT	Function A	Function B					Port Specification			
			1	2	3	4	5	PU/PD	OD	SMT/CMOS	10mA
PORTF											
24	PF0		TB7IN					PU/PD	Yes	SMT	N/A
23	PF1		SC3SCLK					PU/PD	Yes	SMT	N/A
22	PF2		SC3RXD					PU/PD	Yes	SMT	N/A
21	PF3		SC3TXD					PU/PD	Yes	SMT	N/A
20	PF4		TB7OUT					PU/PD	Yes	SMT	N/A
19	PF5		T16A1OUT					PU/PD	Yes	SMT	N/A
18	PF6							PU/PD	Yes	SMT	N/A
17	PF7							PU/PD	Yes	SMT	N/A

Pin-No.	PORT	Function A	Function B					Port Specification			
			1	2	3	4	5	PU/PD	OD	SMT/CMOS	10mA
PORTG											
40	PG0		SC1 SCLK					PU/PD	Yes	SMT	N/A
39	PG1		SC1 RXD					PU/PD	Yes	SMT	N/A
38	PG2		SC1 TXD					PU/PD	Yes	SMT	N/A
37	PG3		TB1IN					PU/PD	Yes	SMT	N/A
36	PG4		TB1 OUT					PU/PD	Yes	SMT	N/A
35	PG5		TB4 OUT					PU/PD	Yes	SMT	N/A
34	PG6		TB5 OUT					PU/PD	Yes	SMT	Yes
33	PG7		TB6 OUT					PU/PD	Yes	SMT	Yes

1.4.2.3 Control pin

Table 1-6 The number of pin and pin names

Pin No.	Control function Pin name
58	X1
59	X2
62	$\overline{\text{RESET}}$
63	MODE
56	$\overline{\text{BOOT}}$

1.4.2.4 Power Supply pin

Table 1-7 The number of pin and pin names

Pin No.	Power supply Pin name
61	REGOUT
64	RVDD3
30, 57	DVDD3
29, 60	DVSS
7	AVDD3
8	AVSS

2. Product Information

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

- "2.1.1 DMA Controller (DMAC)"
- "2.1.2 16-bit Timer/Event Counter (TMRB)"
- "2.1.3 16-bit Timer A(TMR16A)"
- "2.1.4 Serial Channel (SIO/UART)"
- "2.1.5 I2C Bus(I2C)"
- "2.1.6 Analog/Digital Converter (ADC)"
- "2.1.7 Debug Interface"

2.1 Information of Each Peripheral Function

2.1.1 DMA Controller (DMAC)

TMPM037FWUG incorporates 1 unit of built-in DMA controller.

2.1.1.1 DMA Request table

DMA Request table are shows as follows.

Table 2-1 DMA Request table

Request No.	Corresponding peripheral	
	Ch0, Ch1	
	Burst	Single
0	SIO/UART0 reception	–
1	SIO/UART0 transmission	–
2	SIO/UART1 reception	–
3	SIO/UART1 transmission	–
4	SIO/UART2 reception	–
5	SIO/UART2 transmission	–
6	SIO/UART3 reception	–
7	SIO/UART3 transmission	–
8	I2C0 transmission / reception	–
9	–	–
10	SIO/UART4 reception	–
11	SIO/UART4 transmission	–
12	TMRB (ch0-3)	–
13	TMRB (ch4-7)	–
14	Top-priority AD conversion completion/ AD monitor 0 / AD monitor 1	–
15	AD conversion completion	–

2.1.1.2 Peripheral function supported with Peripheral to Peripheral Transfer

Peripheral function(Register) supported with Peripheral to Peripheral Transfer are shown below.

Table 2-2 Supported Peripheral function

Source	Destination
Peripheral register	SCxBUF (x=0 to 4)
	TBxRG0 to 1 (x=0 to 7)
	TBxCP0 to 1 (X=0 to 7)
ScxBUF (x=0 to 4)	Peripheral register
TBxRG0 to 1 (X=0 to 7)	
TBxCP0 to 1 (x=0 to 7)	

2.1.1.3 DMA request control register (DMARQCTL)

The DMA request Control register and address of I2C are shown as bellows.

Base Address= 0x4005_F000

Register names		Address (Base+)
DMAC request setting register	DMACREDGE	0x0000
DMAC request clear register	DMACRCLR	0x0004

2.1.1.4 DMACREDGE(DMAC request 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	-	-	-	-	-	-	-	I2CDMAC
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	−	R	Read as zero.
0	I2CDMAC	R/W	In I2C bus mode, setting for DMA request. 0: High active 1: Reserved Writing "0" only.

2.1.1.5 DMACRCLR(DMAC 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	1
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	-	-	DCLR0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	function
31-1	−	R	Read as zero.
0	DCLR0	W	To clear the DMA request signal in I2C bus mode. 0: - 1: DMA request clear Set to "1" in the service routine of DMAC transfer end interrupt and clear the DMA request. Read as zero.

2.1.2 16-bit Timer/Event Counter (TMRB)

TMPM037FWUG incorporates 8 channels of TMRB.

Table 2-3 Pin specifications

Channel	TBxOUT	TBxIN
TMRB0	PC3	PC5
TMRB1	PG4	PG3
TMRB2	PC2	−
TMRB3	PD0	PD4
TMRB4	PG5	−
TMRB5	PG6	−
TMRB6	PG7	−
TMRB7	PF4	PF0

Table 2-4 Synchronous start specifications

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

Table 2-5 Capture trigger specifications

Trigger input channel	Trigger output
TMRB0 TMRB1 TMRB2	TB6OUT
TMRB3 TMRB4 TMRB5	TB7OUT

This device, the CAP interrupt of each timer channel are assigned respectively to the same factors.

Therefore, to distinguish interrupt INTTBxCAP0 and INTTBxCAP1 of the same factors, it is necessary to have a corresponding such as the following.

1. When the CAP interrupt occurs, read the state of TBxIN terminal, it makes the distinction of "High" or "Low".

2. If the terminal state "High", INTTBxCAP0 interrupt occurs . To determine the terminal state "Low" if INTTBxCAP1 interrupt generation.

Note: This method can be used when the pulse width is equal to or greater than the interrupt processing time in both the High / Low width.

2.1.3 16-bit Timer A(TMR16A)

TMPM037FWUG incorporates 2 channels of TMR16A.

Table 2-6 Pin specifications

Channel	T16AxOUT
T16A0	PC4
T16A1	PF5

2.1.4 Serial Channel (SIO/UART)

TMPM037FWUG incorporates 5 channels of SIO.

Table 2-7 Pin specifications

Channel	SCxTXD	SCxRXD	SCxSCLK
SC0	PD3	PD2	PD1
SC1	PG2	PG1	PG0
SC2	PE4	PE3	PE2
SC3	PF3	PF2	PF1
SC4	PB2	PB1	PB3

Table 2-8 Transfer clock specifications

Clock input channel	Clock output
SC0	TB0OUT
SC1	TB1OUT

Note: TMPM037FWUG does not have the Handshake function(SCxCTS pin).

2.1.5 I2C Bus(I2C)

TMPM037FWUG incorporates 1 channel of I2C.

Table 2-9 Pin specifications

Channel	I2CxSDA	I2CxSCL
I2C0	PC1	PC0

2.1.6 Analog/Digital Converter (ADC)

TMPM037FWUG incorporates 1 unit of ADC.

Table 2-10 Pin specifications

Analog input	AIN0 to 7
Ports	PA0 to 7

Note: TMPM037FWUG analog input pins are 8 channels of AIN0 to AIN7.

Table 2-11 Internal start-up trigger selection

Type	Internal trigger
Normal AD conversion start-up trigger	A match with TMRB ch0
Highest priority AD conversion start-up trigger	A match with TMRB ch1

2.1.7 Debug Interface

TMPM037FWUG supports serial wire debug ports.

Table 2-12 Pin specifications

I/F	SWDIO	SWCLK
Serial wire	PB2	PB1

3. Processor Core

The TX00 series has a high-performance 32-bit processor core (the ARM Cortex-M0 processor core). For information on the operations of this processor core, please refer to the "Cortex-M0 Technical Reference Manual" issued by ARM Limited. This chapter describes the functions unique to the TX00 series that are not explained in that document.

3.1 Information on the processor core

The following table shows the revision of the processor core in the TMPM037FWUG.

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
TMPM037FWUG	r0p0-03

3.2 Configurable Options

The Cortex-M0 core has optional blocks. The following tables shows the configurable options in the TMPM037FWUG.

Configurable Options	Implementation
Interrpts	32
Data endiannes	Little-endian
SysTick timer	Present
Number of watchpoint comparators	2
Number of breakpoint comparators	4
Halting debug support	Present
Multiplier	Fast

Note: The fast multiplier provides a 32-bit × 32-bit multiply that yields the least-significant 32-bits.

3.3 Exceptions/ Interruptions

Exceptions and interruptions are described in the following section.

3.3.1 Number of Interrupt Inputs

The number of interrupt inputs can optionally be defined in the Cortex-M0 core.

TMPM037FWUG has 32 interrupt inputs.

3.3.2 SysTick

TMPM037FWUG has a SysTick timer which can generate SysTick exception.

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

3.3.3 SYSRESETREQ

The Cortex-M0 core outputs SYSRESETREQ signal when <SYSRESETREQ> bit of Application Interrupt and Reset Control Register are set.

TMPM037FWUG provides the same operation when SYSRESETREQ signal are output.

3.3.4 LOCKUP

When irreparable exception generates, the Cortex-M0 core outputs LOCKUP signal to show a serious error included in software.

TMPM037FWUG does not use this signal. To return from LOCKUP status, it is necessary to use non-maskable interruput (NMI) or reset.

3.4 Events

The Cortex-M0 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.

TMPM037FWUG does not use event output signals and event input signals. Please do not use SEV instruction and WFE instruction.

3.5 Power Management

The Cortex-M0 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.

TMPM037FWUG 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."

4. Memory Map

4.1 Memory map

The memory maps for theTMPM037FWUG are based on the ARM Cortex-M0 processor core memory map.

The internal ROM is mapped to the code of the Cortex-M0 core memory, the internal RAM is mapped to the SRAM region and the special function register (SFR) is mapped to the peripheral region respectively.

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.

TMPM037FWUG has bit-band feature equivalent to Cortex-M3 and the SRAM and SFR regions of TMPM037FWUG are all included into 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-M0 Technical Reference Manual".

Note that access to regions indicated as "Fault" causes a hard fault. Do not access the vendor-specific region and the reserved region.

Figure 4-1 shows the memory map of the TMPM037FWUG.

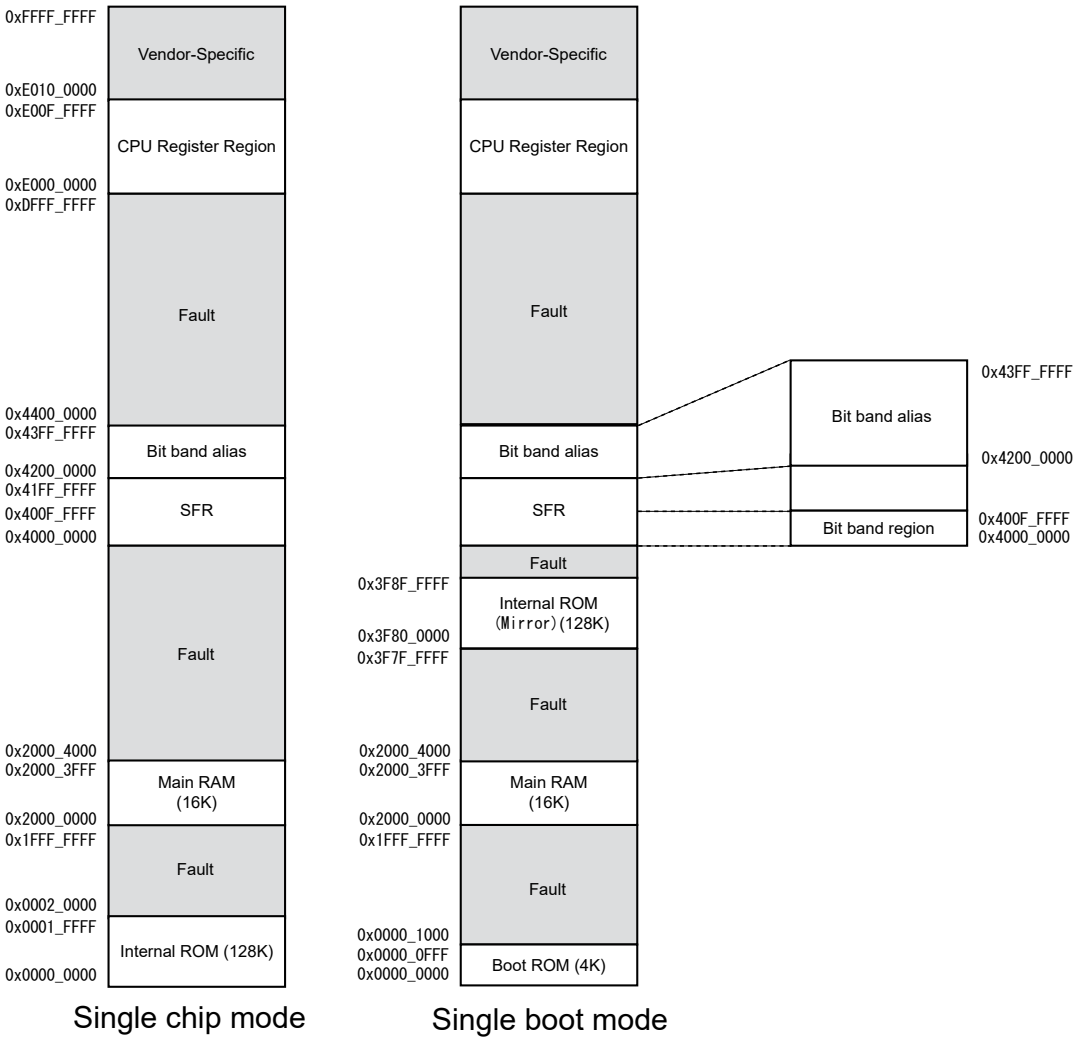


Figure 4-1 TMPM037FWUG Memory Map

4.2 Bus Matrix

The TMPM037FWUG contains two bus masters such as a CPU core and DMA controllers.

Bus masters connect to slave ports (S0,S1) of Bus Matrix. In the bus matrix, master ports(M0 to M7) connect to the peripheral functions via connections described as (o) in the following figure.

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

4.2.1 Structure

4.2.1.1 Single chip mode

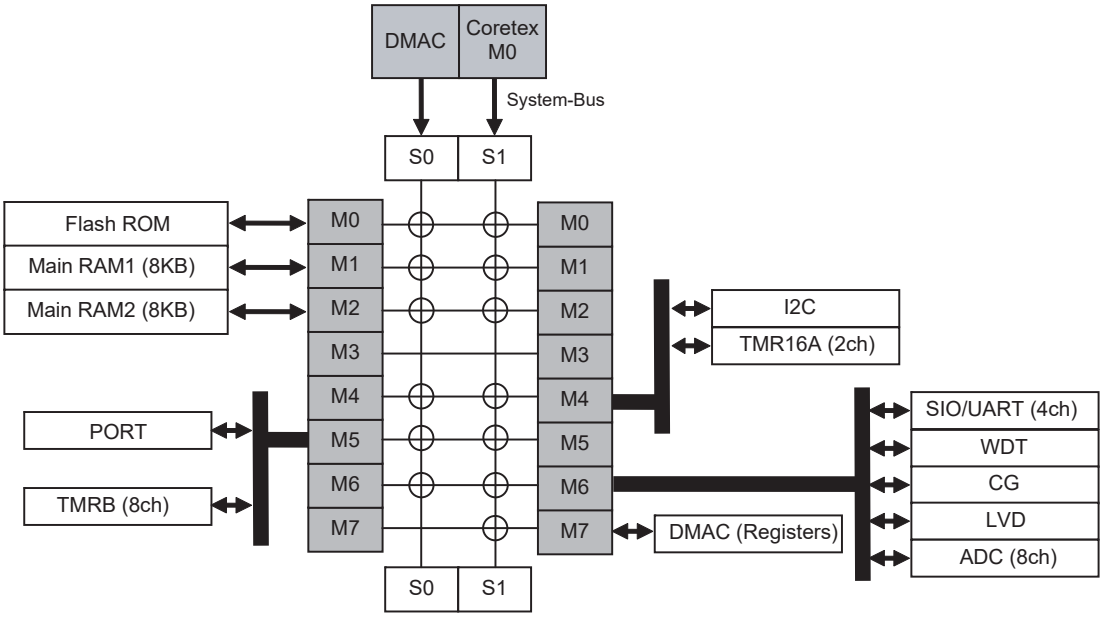


Figure 4-2 TMPM037FWUG (Single chip mode)

4.2.1.2 SIngle boot mode

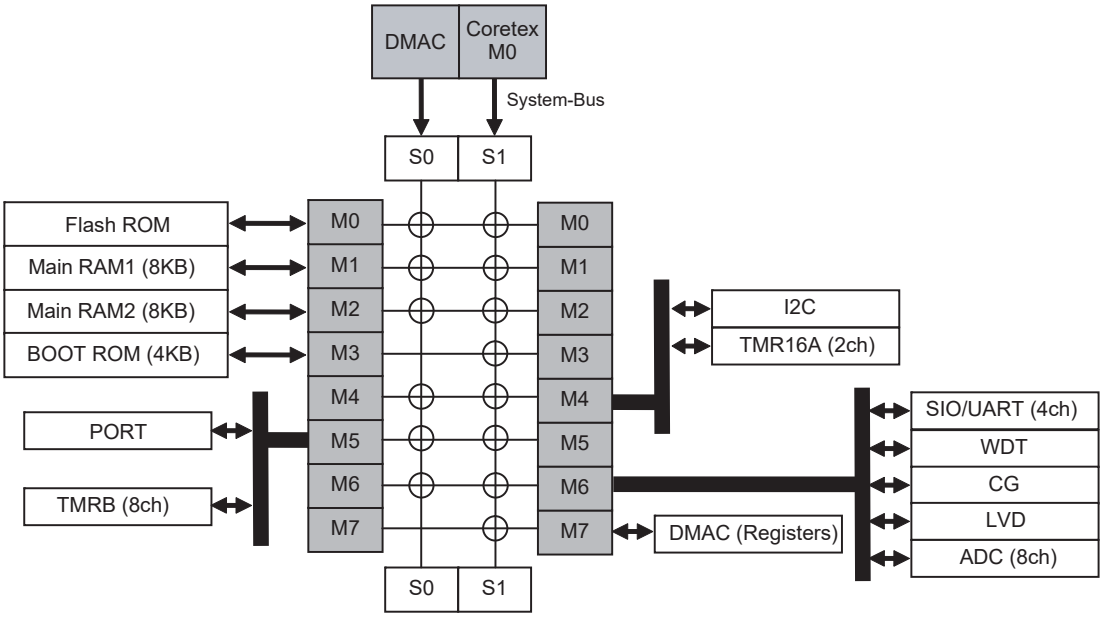


Figure 4-3 TMPM037FWUG (Single boot mode)

4.2.2 Connection table

4.2.2.1 Code area / SRAM area

(1) Single chip mode

Start Address			DMAC	Core
			S0	S1
0x0000_0000	Flash ROM	M0	o	o
0x0002_0000	Fault	-	Fault	Fault
0x2000_0000	Main RAM0	M1	o	o
0x2000_2000	Main RAM1	M2	o	o
0x2000_4000	Fault	-	Fault	Fault

(2) Single boot mode

Start Address			DMAC	Core
			S0	S1
0x0000_0000	BOOT ROM	M3	Fault	o
0x0000_1000	Fault	-	Fault	Fault
0x2000_0000	Main RAM0	M1	o	o
0x2000_2000	Main RAM1	M2	o	o
0x2000_4000	Fault	-	Fault	Fault
0x3F80_0000	Flash ROM (Mirror)	M0	o	o
0x3F82_0000	Fault	-	Fault	Fault

4.2.2.2 Peripheral area

Start Address			DMAC	Core S-Bus
			S0	S1
0x4000_0000	DMAC	M7	-	o
0x4000_1000	Fault	-	Fault	Fault
0x4005_F000	DMARC	M4	o	o
0x4006_0000	Reserved	-	-	-
0x4008_D000	TMR16A	M4	o	o
0x4008_F000	Reserved	-	-	-
0x400A_0000	I2C	M4	o	o
0x400A_1000	Fault	-	Fault	Fault
0x400C_0000	PORT	M5	o	o
0x400C_0800	Reserved	-	-	-
0x400C_4000	TMRB	M5	o	o
0x400C_4800	Fault	-	Fault	Fault
0x400E_1000	SIO/UART	M6	o	o
0x400E_1500	Reserved	-	-	-
0x400F_2000	WDT	M6	-	o
0x400F_2100	Reserved	-	-	-
0x400F_3000	CG	M6	-	o
0x400F_3100	Reserved	-	-	-
0x400F_4000	LVD	M6	-	o
0x400F_4100	Reserved	-	-	-
0x400F_C000	ADC	M6	o	o
0x400F_C100	Reserved	-	-	-
0x41FF_FF00	SFR(FLASH)	M6	-	o
0x4200_0000	Bit band alias	-	-	o

4.2.3 Address lists of peripheral functions

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

Peripheral Function		Base Address
DMA Controller (DMAC)		0x4000_0000
DMA request controller (DMARC)		0x4005_F000
16-bit Timer (TMR16A)	ch0	0x4008_D000
	ch1	0x4008_E000
I2C Serial bus interface (I2C)	ch0	0x400A_0000
Input / Output port (PORT)	Port A	0x400C_0000
	Port B	0x400C_0100
	Port C	0x400C_0200
	Port D	0x400C_0300
	Port E	0x400C_0400
	Port F	0x400C_0500
	Port G	0x400C_0600
16-bit Timer /Event Counters (TMRB)	ch0	0x400C_4000
	ch1	0x400C_4100
	ch2	0x400C_4200
	ch3	0x400C_4300
	ch4	0x400C_4400
	ch5	0x400C_4500
	ch6	0x400C_4600
	ch7	0x400C_4700
Serial Channel (SIO/UART)	ch0	0x400E_1000
	ch1	0x400E_1100
	ch2	0x400E_1200
	ch3	0x400E_1300
	ch4	0x400E_1400
Watchdog Timer (WDT)		0x400F_2000
Clock/ Mode control (CG)		0x400F_3000
Low Voltage Detection Circuit (LVD)		0x400F_4000
Analog / Digital Converter (ADC)		0x400F_C000
SFR(FLASH)		0x41FF_FF00

5. Reset Operation

The following are sources of reset operation.

- Power On Reset
- Reset Pin(RESET)
- Low voltage detection circuit(LVD)
- Watch-dog timer(WDT)
- Application interrupt by CPU and signal from the reset register bit<SYSRESETREQ>

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

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

A reset by WDT is refer to the chapter on the "Watch-dog timer".

A reset by <SYSRESETREQ> is referred to "Cortex-M0 Technical Reference Manual".

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

5.1 Cold Reset

5.1.1 Cold Reset by $\overline{\text{RESET}}$ pin

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

When turning-on power, it is necessary to take a stable time of built-in regulator into consideration. In this product ,the internal regulator requires at least approximately 1.0ms to be stable. At cold reset, $\overline{\text{RESET}}$ pin must be kept "Low" for a duration of time sufficiently long enough for the internal regulator to be stable.

Approximately 0.8ms after $\overline{\text{RESET}}$ pin becomes "High ",internal reset will be released.

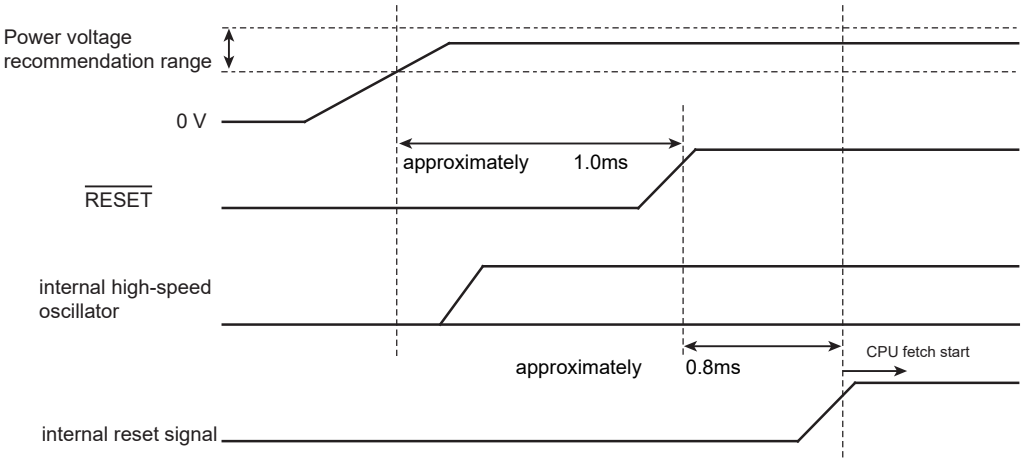


Figure 5-1 Cold reset Operation Sequence

5.1.2 Cold Reset with power-on-reset circuit

In case of Using power-on-reset circuit, there is a restriction on rising time of power line.

the power pin should start a power supply to reach the recommendation operation voltage range within 1.0ms.

Approximately 1.8ms after the power voltage becomes the recommendation operation voltage range, internal reset will be released.

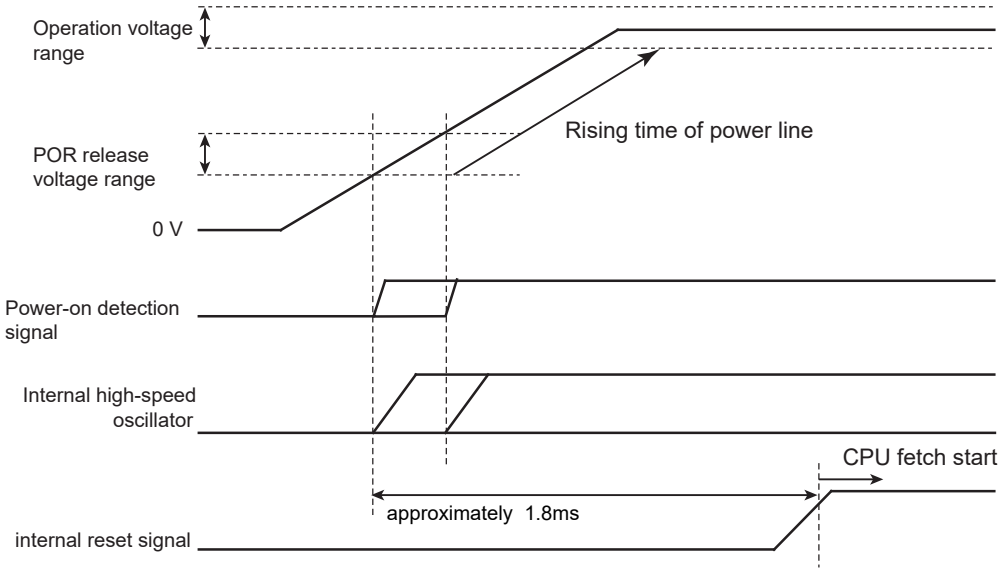


Figure 5-2 Cold reset with power-on-circuit

5.2 Warm Reset

To do reset TMPM037FWUG, the following conditions are required; power supply voltage is in the operational range ; $\overline{\text{RESET}}$ pin is kept "Low" at least for 12 internal high-speed clocks. Approximately 0.8ms after $\overline{\text{RESET}}$ pin becomes "High", internal reset will be released.

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

5.3 After reset

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

System debug component registers(FPB, DWT, and ITM)of the internal core, and FCSECBIT in the Flash related register are only initialized by cold reset.

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

6. Clock/Mode control

6.1 Features

The clock/mode control 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 TMPM037FWUG can operate low power mode to reduce power consumption according to its usage conditions.

6.2 Registers

6.2.1 Register List

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

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

Register name		Address (Base+)
System control register	CGSYSCR	0x0000
Oscillation control register	CGOSCCR	0x0004
Standby control register	CGSTBYCR	0x0008
PLL selection register	CGPLLSEL	0x000C
Protect register	CGPROTECT	0x003C

6.2.2 CGSYSCR (System control register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
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	−	R/W	Write "0".
19-18	−	R	Read as "0".
17-16	−	R/W	Write "01".
15-14	−	R	Read as "0".
13	−	R/W	Write "0".
12	FPSEL	R/W	Selects fperiph source clock. 0: fgear 1: fc Specifies the source clock to fperiph. Selecting fc fixes fperiph regardless of the clock gear.
11	−	R	Read as "0".
10-8	PRCK[2:0]	R/W	Prescaler clock 000: fperiph 100: fperiph/16 001: fperiph/2 101: fperiph/32 010: fperiph/4 110: Reserved 011: fperiph/8 111: Reserved Specifies the prescaler clock to peripheral I/O.
7-3	−	R	Read as "0".
2-0	GEAR[2:0]	R/W	High-speed gear clock (fc). 000: fc 100: fc/2 001: Reserved 101: fc/4 010: Reserved 110: fc/8 011: Reserved 111: fc/16

6.2.3 CGOSCCR (Oscillation control register)

	31	30	29	28	27	26	25	24
bit symbol	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	-	-	-	-	-	-	OSCF	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	Select High-Speed warm-up counter clock 0: Internal OSC(IHOSC) 1: External OSC (f_{eosc}) Select the OSC clock for warm-up timer. A warm-up timer is counting by the selected clock.
18	EHOSCSEL	R/W	Select external OSC source 0: external clock input (EHCLKIN) 1: external oscillator (EHOSC)
17	OSCSEL	R/W	Select High-speed oscillator (Note2) 0: internal (IHOSC) 1: external (EHOSC)
16	XEN2	R/W	Internal high-speed oscillator operation control 0: Stop 1: Oscillation
15-14	−	R/W	Write "0".
13	−	R	Read as "0".
12	−	R/W	Write "0".
11-10	−	R	Read as "0".
9	OSCF	R	Status of Selected High-speed oscillator. 0:internal high-speed oscillator 1:external high-speed oscillator
8	XEN1	R/W	External OSC operation control 0: stop 1: Oscillation
7-3	−	R/W	Write "00110" only.
2	PLLON	R/W	PLL (multiplying circuit) operation control (note3) 0: stop 1: oscillation
1	WUEF	R	Operation status of warm-up timer 0: warm-up completed 1: warm-up active Can be monitored the status of the warm-up timer.
0	WUEON	W	Operation of warm-up timer (WUP) 0: don't care 1: warm-up timer start Enables to start the warm-up timer. Read as "0".

Note 1: Refer to "6.3.4 Warm-up function" about Warm-up setup.

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

Note 3: When the PLL value is set, the CGOSCCR<PLLON> is set to "0" after approximately 100 μs elapses as initialization time of PLL, and the state of PLL starts.

Note 4: After changed CGOSCCR<PLLON> is set to "1", CGPLLSEL<PLLSEL > should be set to "1" with Warp-up time finished.

Note 5: Returning from the STOP1 mode, related bits <HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1>,<PLLON> of the register CGOSCCR and CGPLLSEL<PLLSEL> are initialized and internal high-speed oscillator starts up.

Note 6: When using internal high-speed oscillator (IHOSC) as system clock, do not use PLL multiplying.

Note 7: When using internal high-speed oscillator (IHOSC), do not use it as system clock which high accuracy assurance is required.

6.2.4 CGSTBYCR (Standby control register)

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

Bit	Bit Symbol	Type	Function
31-20	−	R	Read as "0".
19-18	−	R/W	Write "0".
17	−	R/W	Write "0".
16	−	R	Read as "0".
15-3	−	R	Read as "0".
2-0	STBY[2:0]	R/W	Low power consumption mode control. 000: Reserved 001: STOP1 010: Reserved 011: IDLE 100: Reserved 101: Reserved 110: Reserved 111: Reserved

6.2.5 CGPLLSEL(PLL Selection Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	PLLST	-	-
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
31-19	−	R	Read as "0".
18	PLLST	R	Status of selected clock in PLL 0:fosc 1:f_{PLL}
17-16	−	R	Read as "0".
15-1	PLLSET	R/W	PLL multiplying value (Do not use except below) 0x609F: 2 times
0	PLLSEL	R/W	Use of PLL 0: fosc USE 1: f_{PLL} USE Specifies use or disuse of the clock multiplied by the PLL. fosc is automatically set after reset. Setting is required when using the PLL.

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

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

Note 3: When the PLL value is set, the CGOSCCR<PLLON> is set to "0" after approximately 100 μs elapses as initialization time of PLL, and the state of PLL starts.

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

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

6.2.6 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
31-8	–	R	Read as "0".
7-0	CGPROTECT	R/W	Register protection control 0xC1: Register write enable Except 0xC1:Register write disable Initial value is "0xC1" as writing enable to each register and when writing except "0xC1",each register except CGPROTECT register ca not be written.

6.3 Clock control

6.3.1 Clock Type

Each clock is defined as follows.

fosc	: Clock from the internal oscillator, or input from X1&X2 pin.
f_{PLL}	: Clock multiplied by PLL(2 x).
fc	: Clock specified by CGPLLSEL<PLLSEL> (high-speed clock)
fgear	: Clock specified by CGSYSCR<GEAR[2:0]>.(system clock)
fsys	: Clock specified by CGSYSCR<GEAR[2:0]>.(system clock)
fperiph	: Clock specified by CGSYSCR<FPSEL[2:0]>
φT0	: Clock specified by CGSYSCR<PRCK[2:0]> (prescaler clock)

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

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

6.3.2 Initial Values after Reset

Reset operation initializes the clock configuration as follows.

internal high-speed oscillator	: oscillating
external high-speed oscillator	: stop
PLL (phased locked loop circuit)	: stop
High-speed gear clock	: fc (no frequency dividing)

Reset operation causes all the clock configurations to be the same as fosc.

fc = fosc
fsys = fosc
φT0 = fosc

6.3.3 Clock system Diagram

Figure 6-1 shows the clock system diagram.

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

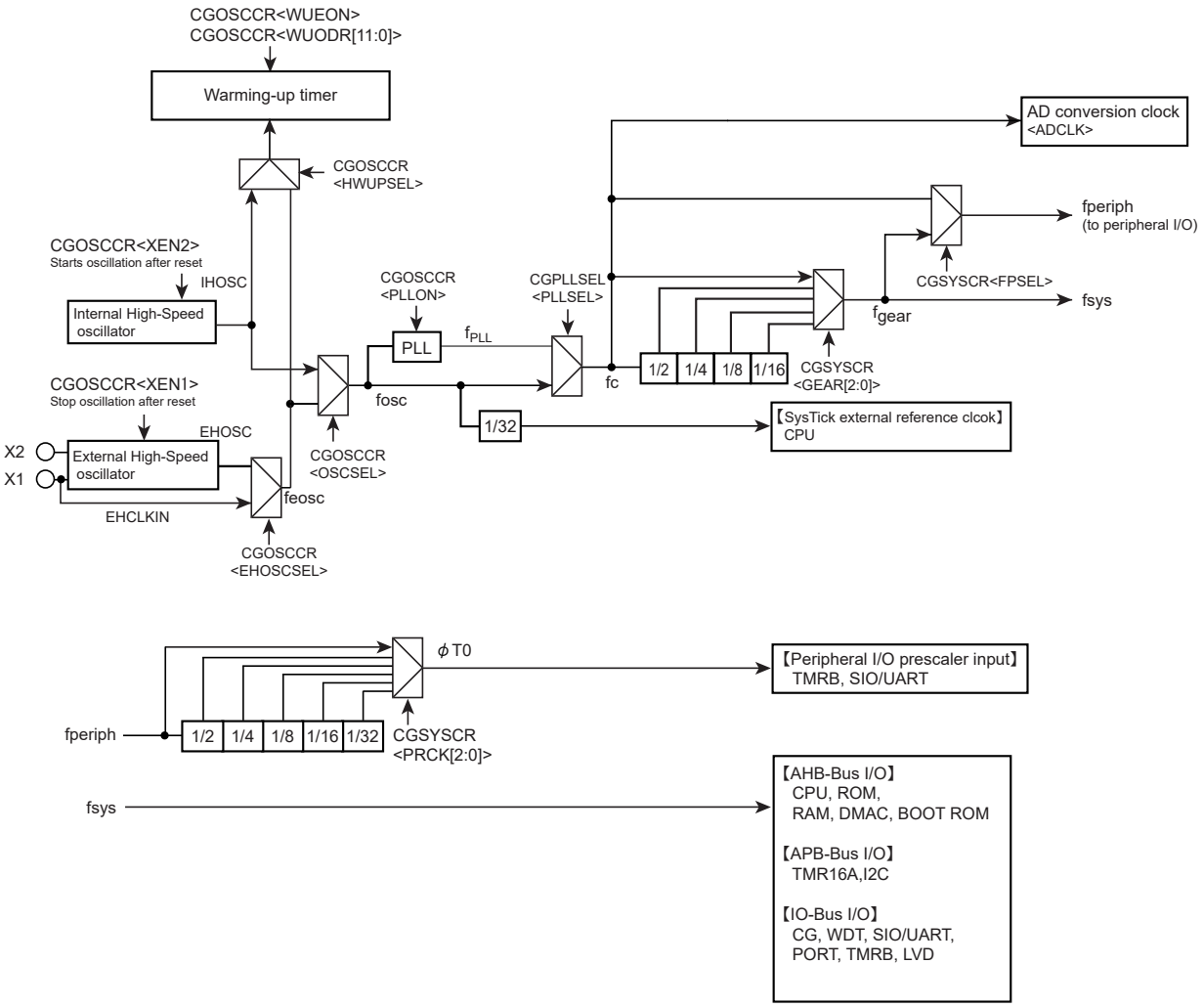


Figure 6-1 Clock Block Diagram

6.3.4 Warm-up function

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

Refer to "6.6.6 Warm-up" for a detail.

How to configure the warm-up function is described.

1. Specify the count up clock

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

2. Specify the warm-up counter value

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

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

   When using high-speed oscillator 8MHz, and set warm-up time 5ms as follows.

$$\frac{\text{Warm-up time to set}}{\text{Input frequency cycle(s)}} = \frac{5\text{ms}}{1/8\text{MHz}} = 40{,}000 \text{ cycle} = 0\text{x}9\text{C}40$$

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

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

   When the warm-up is started by software, by setting CGOSCCR<WUEON> to "1", the warm-up start a count up. The CGOSCCR <WUEF> is used to confirm the start and completion of warm-up.

   <WUEF> "1" shows under warm-up and <WUEF> "0" shows completion of warm-up

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

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

The example of warm-up function setup is shown below.

Table 6-1 The example of warm-up function setup

	CGOSCCR<WUODR[11:0]> = "0x9C4"	:Specify the warm-up time
↻	CGOSCCR<WUODR[11:0]> read	: Confirm warm-up time reflecting Repeat until the read data is "0x9C4".
	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. Therefor, the warm-up time should be taken as approximate time.

Note 3: After setting warm-up count value to CGOSCCR<WUODR[11:0]>,wait until confirming of the value to be reflected, then change to the standby mode by WFI instruction.

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

6.3.5 Clock Multiplication Circuit (PLL)

This circuit outputs the f_{PLL} clock that is multiplied by 2 of the high-speed oscillator output clock. As a result, the input frequency to oscillator can be low frequency, and the internal clock be made high-speed.

6.3.5.1 How to configure the PLL function

The PLL is disabled after reset.

To enable the PLL, set CGPLLSEL<PLLSET> to multiplying value while CGOSCCR<PLLON> is "0". And set <PLLON> to "1" after 100μs for initialize time of PLL. After 100μs for lock-up time elap-ses, set CGPLLSEL<PLLSEL> to "1", f_{PLL} which is multiplied by 2 from fosc is used.

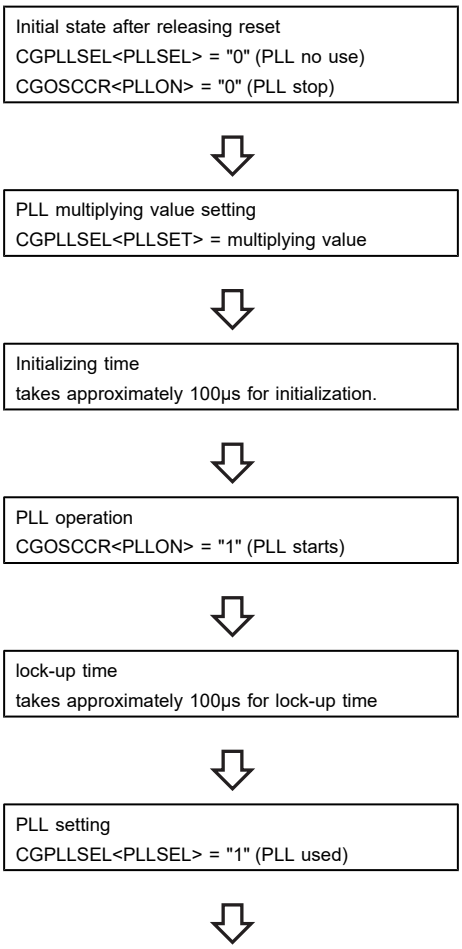
The PLL requires a certain amount of time to be stabilized, which should be secured using the warm-up function or other methods.

Note 1: When using internal high-speed oscillator (IHOSC) as system clock, do not use PLL multiplying.

As for the 2 multiplying value, only the following setting are permitted.

Multiplying	<PLLSET>
2	0x609F

6.3.5.2 The sequence of PLL setting



6.3.6 System clock

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

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 oscillation clock, the PLL function can be used by multiplying.

Source clock		Frequency	Using PLL
Internal High-speed oscillation (f_{IHOSC})		10MHz	can not use
External high-speed oscillation	Oscillator (f_{EHOSC})	8 ≤ fOSC ≤ 10MHz	2 multiplying
		10 < fOSC ≤20MHz	can not use
	Input clock ($f_{EHCLKIN}$)	8 ≤ fOSC ≤ 10MHz	2 multiplying
		10 < fOSC ≤ 20MHz	can not use

The clock generated with PLL by multiplying 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 ~ 20	20 (Max.)

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

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

Table 6-2 System clock frequency when PLL is 2 times

External oscillator (MHz)	External clock input (MHz)	PLL multiplying	max. operation frequency (fc) (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	2	16	16	8	4	2	1	8	4	2	1	-
10	10		20	20	10	5	2.5	1.25	10	5	2.5	1.25	-
12	12	-	12	-	-	-	-	-	12	6	3	1.5	-
16	16		16	-	-	-	-	-	16	8	4	2	1
20	20		20	-	-	-	-	-	20	10	5	2.5	1.25

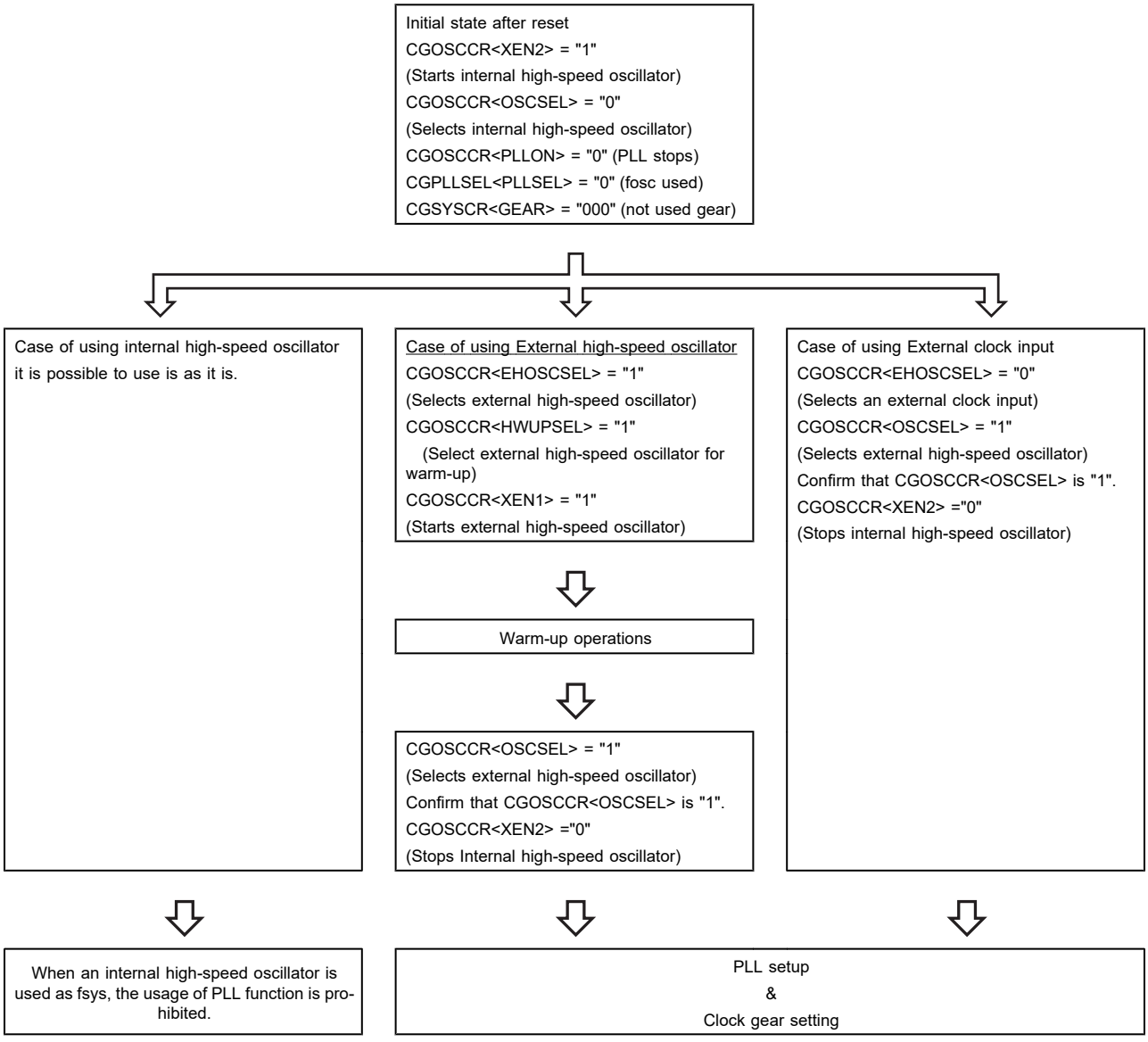
↑ initial value after reset

6.3.6.1 Clock setting

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

The clock setup sequence is shown as follow.

Clock setup sequence



6.3.6.2 When using external oscillator

This product activates from an internal high-speed oscillator after releasing reset.When an external high-speed oscillator and Clock multiplication circuit (PLL) are used, it sets it according to the procedure"6.3.5 Clock Multiplication Circuit (PLL)" and "6.3.6.1 Clock setting".

The following figures show the transition when external high-speed oscillator and clock multiplication circuit are used.

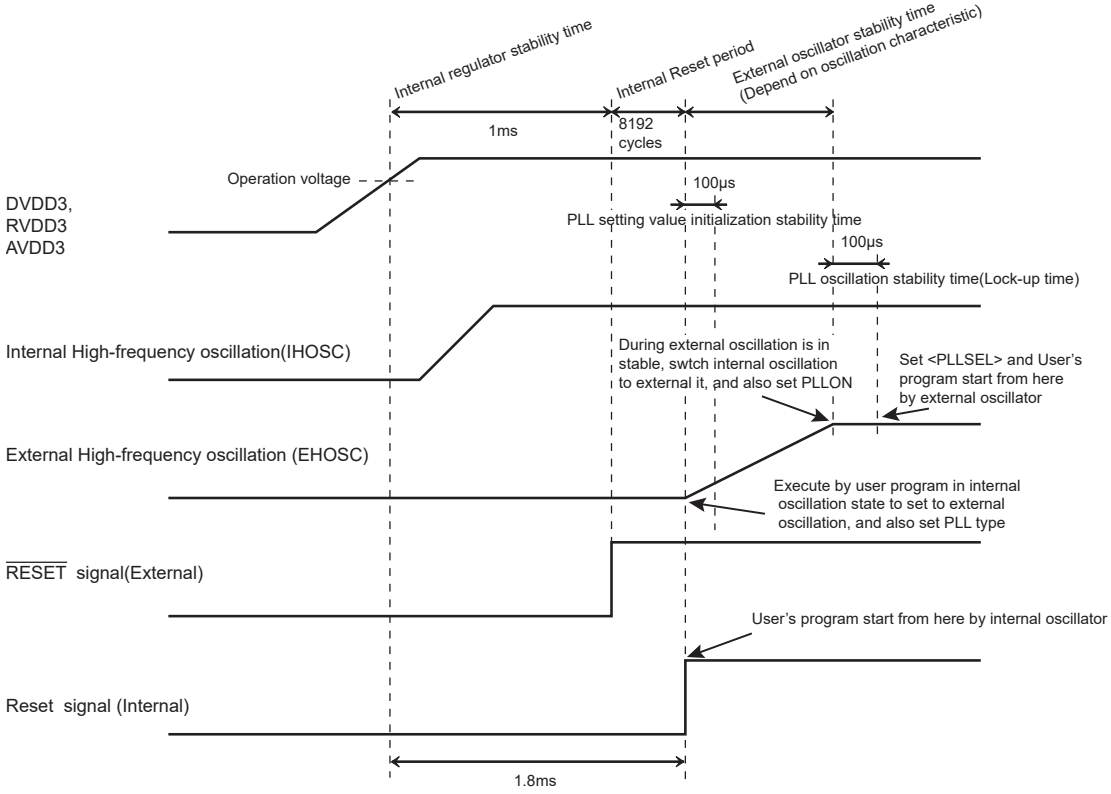


Figure 6-2 Transition when high-speed oscillator is set with PLL

6.3.7 Prescaler Clock Control

Peripheral function (TMRB,SIO/UART) has a prescaler for dividing a clcok.As a 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 reset, fperiph/1 is selected as φT0.

Note: Do not switch the clock gear while the timer counter or other peripheral function is operating.

6.4 Modes and Mode Transitions

6.4.1 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 6-3 shows a mode transition diagram.

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

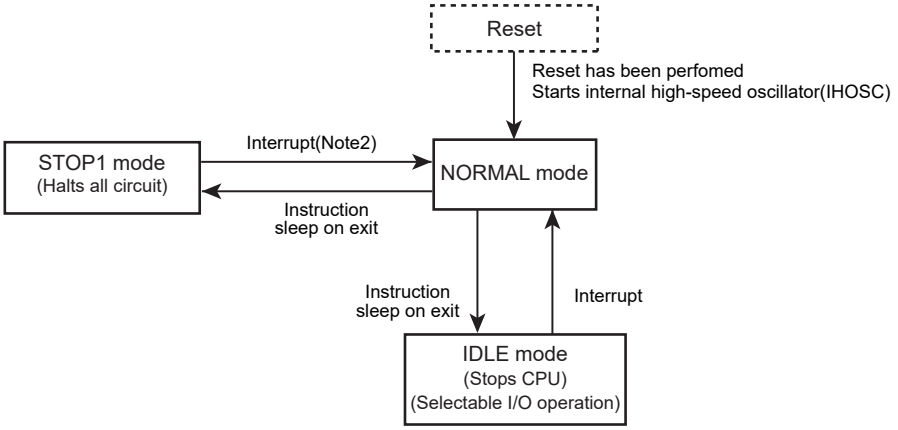


Figure 6-3 Mode Transition Diagram

Note 1: Returning form the STOP1 mode, related bits CGPLLSEL<PLLSEL>,CGOSCCR<HWUPSEL>,<OSCSEL>,<XEN2>,<XEN1>and <PLLON> are initialized, but CGOSCCR<WUODR[11:0]> is not initialized.

Note 2: it branches to interrupt service routine of interrupt factor when returning from the STOP1 mode.

6.5 Operation mode

6.5.1 NORMAL mode

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

It is shifted to the NORMAL mode after reset.

6.6 Low Power Consumption Modes

The TMPM037FWUG has low power consumption modes: IDLE, 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 TMPM037FWUG 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 TMPM037FWUG does not support the low power consumption mode configured with the SLEEPDEEP bit in the Cortex-M0 core. Setting the <SLEEPDEEP> bit of the system control register is prohibited.

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

6.6.1 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)
- 16-bit timer (TMR16A)
- Serial channel (SIO/UART)
- Analog Digital converter (ADC)
- Watch dog timer (WDT)

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

6.6.2 STOP1 mode

All the internal circuits including the internal oscillator are brought to a stop in STOP1 mode. And the internal oscillator activates the clock after releasing the STOP1 mode then transit to the Normal mode.

The STOP1 mode enables to select the pin status by setting the port register.Table 6-3 shows the pin status in the STOP1 mode.

Table 6-3 Pin States in the STOP1 mode

Function	Pin name	I/O	STOP1
Control pin	$\overline{\text{RESET}}$, MODE	Input	o
Oscillator	X1/ EHCLKIN	Input	×
	X2	Output	"High" level output.
Port	PB1 (SWCLK) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Input	Depends on PxIE[m]
	PB2 (SWDIO) (Debug I/F setting, case of PxFRn<PxmFn>="1")	Input	Depends on PxIE[m]
		Output	Enable when data is valid. Disabled when data is invalid
	PB5, PB6, PB7, PE5, PE6, PE7 (INT0 to 5) (Interrupt setting, case of PxFRn<PxmFn>="1" and PxIE<PxmIE>="1")	Input	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

6.6.3 Low power Consumption Mode Setting

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

Table 6-4 shows the mode setting in the <STBY[2:0]>.

Table 6-4 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]>.

6.6.4 Operational Status in Each Mode

Table 6-5 show the operational status in each mode.

Table 6-5 Operational Status in Each Mode

Block	NORMAL Internal high-speed oscillator (IHOSC)	NORMAL external high-speed oscillator (EHOSC)	IDLE Internal high-speed oscillator (IHOSC)	IDLE external high-speed oscillator (EHOSC)	STOP1 (Note 1)
Processor core	o	o	−	−	−
DMAC	o	o	o	o	−
IO port	o	o	o	o	o
SIO/UART	o	o	Δ	Δ	−
I2C	o	o	o	o	−
TMRB	o	o	Δ	Δ	−
TMR16A	o	o	Δ	Δ	−
WDT	o	o	Δ(Note3)	Δ(Note3)	−
10-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	−
LVD	o	o	o	o	−
Internal high-speed oscillator (IHOSC)	o	o(Note 2)	o	o(Note 2)	−
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 "−". it is available to reduce leakage current by stopping reference voltage for AD converter.

Note 2: After reset or STOP1 mode released, clock is provided from internal high-speed oscillator.

Note 3: Pay attention that the counter of watch dog timer function can not be cleared by CPU while in IDLE mode.

6.6.5 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 6-6.

Table 6-6 Release Source in Each Mode

Low power consumption mode			IDLE	STOP1
release source	Interrupt	INT0 to 5 (Note2)	o	o
		INTTB0 to 7	o	×
		INTTTMR16A0 to 1	o	×
		INTCAP00 to 71	o	×
		INTRX0 to 4, INTTX0 to 4	o	×
		INTI2C0	o	×
		INTAD/INTADHP/INTADM0 to 1	o	×
		INTDMAC0TC, INTDMAC0ERR	o	×
	SysTick interrupt		o	×
	Non-Maskable Interrupt (INTWDT)		o	×
	Non-Maskable Interrupt (INTLVD)		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 by interrupting level mode from ILDE,STOP1 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)

  There are two kinds of NMI sources: WDT interrupt (INTWDT), LVD interrupt (INTLVD).

  INTWDT,INTLVD 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. The reset signal as same as cold reset should be inputted.

- Release by SysTick interrupt

  SysTick interrupt can only be used in the IDLE mode.

Refer to "Interrupts" for details.

6.6.6 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: Returning form the STOP1 mode, related bits CGPLLSEL<PLLSEL>,CGOSCCR<HWUPSEL>,<OSCSEL>,<XEN2>,<XEN1>and <PLLON> are initialized, but CGOSCCR<WUODR[11:0]> is not initialized.

Table 6-7 shows whether the warm-up setting of each mode transition is required or not.

Table 6-7 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: When releasing by reset is executed, automatic warm-up is not performed. Input a reset until the oscillator becomes stable.

6.6.7 Clock Operations in Mode Transition

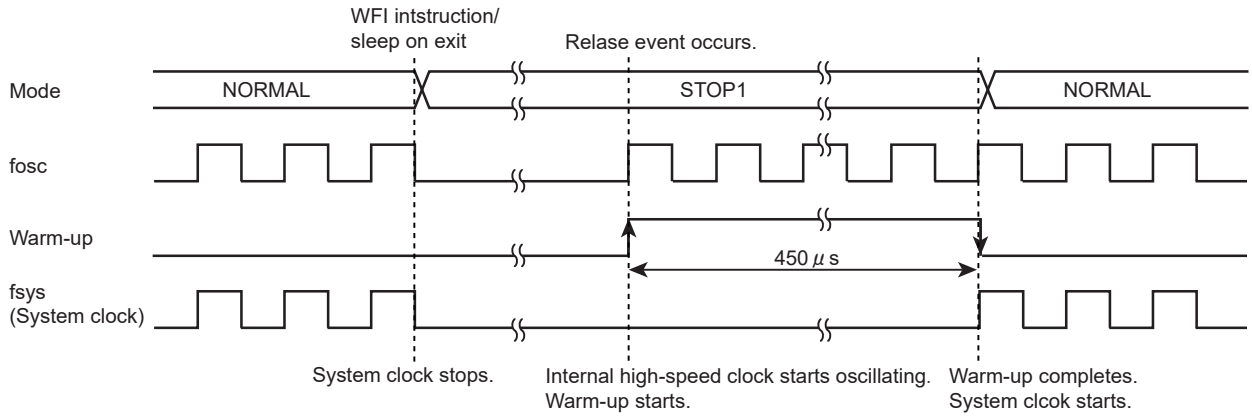
The clock operations in mode transition are described as follows.

6.6.7.1 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 for the warm-up to be set CGOSCCR<WUODR[11:0]>=0x119 in this case as a stability time (450μs) of on chip Flash ROM before entering the STOP1 mode.

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



6.6.8 Precaution on Transition to the Low-power Consumption Mode

6.6.8.1 Case when the MCU Enters IDLE or STOP1 Mode

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

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

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

   (Stop the watch-dog timer, Stop the LVD)

4. If the MCU did not enter STOP1 mode, CGOSCCR<HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1>,<PLLSEL> and CGPLLSEL <PLLON> are not initialized. These registers maintain the former condition before entering the STOP1 mode.

7. Exceptions

This chapter describes features, types and handling of exceptions.

Exceptions have close relation to the CPU core. Refer to "Cortex-M0 Technical Reference Manual" if needed.

7.1 Overview

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

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

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

7.1.1 Exception Types

The following types of exceptions exist in the Cortex-M0

For detailed descriptions on each exception, refer to "Cortex-M0 Technical Reference Manual".

- Reset
- Non-Maskable Interrupt (NMI)
- Hard Fault
- SVCall (Supervisor Call)
- PendSV
- SysTick
- External Interrupt

7.1.2 Handling Flowchart

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

Each step described later in this chapter.

Processing	Description	See
Detection by CG/CPU	The CG/CPU detects the exception request.	Section 7.1.2.1
Handling by CPU	The CPU handles the exception request.	Section 7.1.2.2
Branch to ISR	The CPU branches to the corresponding interrupt service routine (ISR).	
Execution of ISR	Necessary processing is executed.	Section 7.1.2.3
Return from exception	The CPU branches to another ISR or returns to the previous program.	Section 7.1.2.4

7.1.2.1 Exception Request and Detection

(1) Exception occurrence

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

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

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

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

(2) Exception detection

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

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

If a disabled exception occurs, it is handled as Hard Fault.

Table 7-1 Exception Types and Priority

No.	Exception type	Priority	Description
1	Reset	−3 (highest)	Reset pin, WDT, POR, LVD or SYSRETREQ
2	Non-Maskable Interrupt	−2	WDT
3	Hard Fault	−1	Fault that cannot activate because a higher-priority fault is being handled or it is disabled
4to10	Reserved	−	
11	SVCall	Configurable	System service call with SVC instruction
12to13	Reserved	−	
14	PendSV	Configurable	Pending system service request
15	SysTick	Configurable	Notification from system timer
From16	External interrupt	Configurable	External interrupt pin or peripheral function (Note2)

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

(3) Priority setting

- Priority level

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

  TMPM037FWUG has 2-bit of <PRI_n>

  The configuration of <PRI_n> is two bit, so the priority can be configured in the range from 0 to 3. The highest priority is "0". If multiple elements with the same priority exist, the smaller the number, the higher the priority becomes.

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

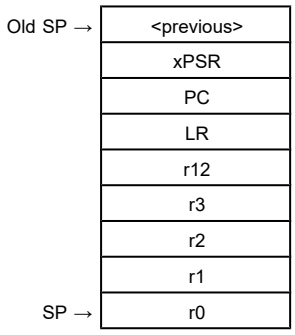
When an exception occurs, the CPU suspends the currently executing process and branches to the interrupt service routine. This is called "pre-emption".

(1) Stacking

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

1. Program Counter (xPSR)
2. Program Status register (PC)
3. Link register (LR)
4. r12
5. r3 to r0

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.

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.

It should always be 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 to 0x28	Reserved	-	-
0x2C	SVCall	ISR address	Optional
0x30 to 0x34	Reserved	-	-
0x38	PendSV	ISR address	Optional
0x3C	SysTick	ISR address	Optional
0x40	External Interrupt	ISR address	Optional

7.1.2.3 Executing an ISR

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

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

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

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

7.1.2.4 Exception exit

(1) Execution after returning from an ISR

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

- Tail-chaining

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

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

- Returning to the last stacked ISR

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

- Returning to the previous program

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

(2) Exception exit sequence

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

- Pop registers

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

- Load current active interrupt number

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

- Select SP

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

7.2 Reset Exceptions

Reset exceptions are generated from the following 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 POR

  The POR has a reset generating feature. For details, see the chapter on the POR.

- 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.

- Reset exception by LVD

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

7.3 Non-Maskable Interrupts (NMI)

Non-maskable interrupts are generated from the following sources.

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

- Non-maskable interrupt by WDT

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

- Non-maskable interrupt by LVD

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

7.4 SysTick

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

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

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

7.5 Interrupts

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

The CPU is notified of interrupt requests by the interrupt signal from each interrupt source. It sets priority on interrupts and handles an interrupt request with the highest priority.

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

7.5.1 Interrupt Sources

7.5.1.1 Interrupt Route

Figure 7-1 shows an interrupt request route.

The interrupts issued by the peripheral function that is not used to release standby are directly input to the CPU (route 1)

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

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

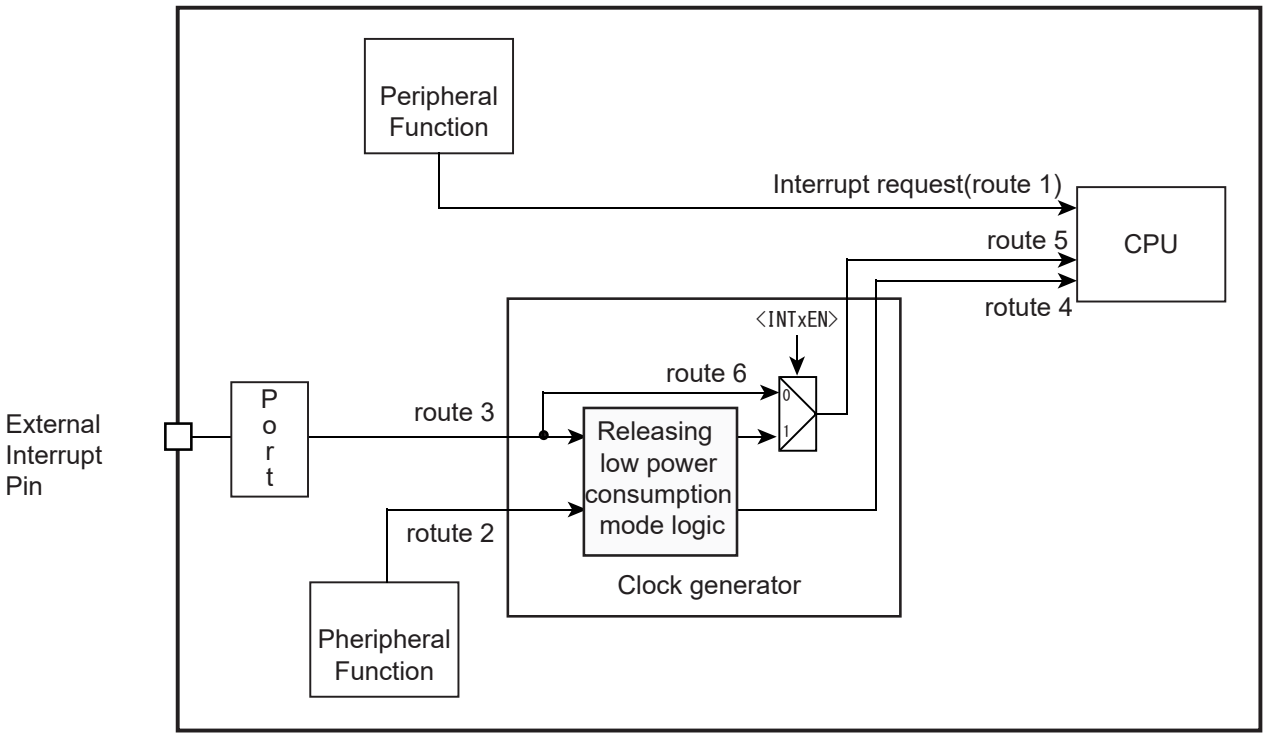


Figure 7-1 Interrupt Route

7.5.1.2 Generation

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

- From external pin

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

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

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

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

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

7.5.1.3 Transmission

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

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

7.5.1.4 Precautions when using external interrupt pins

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

If input disabled (PxIE<PxmIE>="0"), inputs from external interrupt pins are "High". Also, if external interrupts are not used as a trigger to release standby (route 6 of "Figure 7-1 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.

7.5.2 List of Interrupt Sources

Table 7-2 shows the list of interrupt sources.

Table 7-2 List of Interrupt Sources

No	Interrupt Source		The active level to release the low power consumption mode					CG interrupt mode control register
			"Low" level	"High" level	Rising edge	Falling edge	Both edge	
0	INT0	External interrupt pin 0	o	o	o	o	o	CGIMCGA
1	INT1	External interrupt pin 1	o	o	o	o	o	
2	INT2	External interrupt pin 2	o	o	o	o	o	
3	INT3	External interrupt pin 3	o	o	o	o	o	
4	INT4	External interrupt pin 4	o	o	o	o	o	CGIMCGB
5	INT5	External interrupt pin 5	o	o	o	o	o	
6	INTRX0	Serial reception (channel.0)						
7	INTTX0	Serial transmission (channel.0)						
8	INTRX1	Serial reception (channel.1)						
9	INTTX1	Serial transmission (channel.1)						
10	Reserved	-						
11	Reserved	-						
12	INTI2C0	I2C0 interrupt						
13	INTDMAC INTDMACTC INTDMACERR	 DMAC transmission completion interrupt DMAC transmission error interrupt						
14	INTT16A0	16-bit TMR16A match detection (channel.0)						
15	INTT16A1	16-bit TMR16A match detection (channel.1)						
16	INTTMRB0	16-bit TMRB (channel0)						
	INTTB0 INTTB0CAP0 INTTB0CAP1	match detection/0ver flow input capture0 input capture1						
17	INTTMRB1	16-bit TMRB (channel1)						
	INTTB1 INTTB1CAP0 INTTB1CAP1	match detection/0ver flow input capture0 input capture1						
18	INTTMRB2	16-bit TMRB (channel2)						
	INTTB2 INTTB2CAP0 INTTB2CAP1	match detection/0ver flow input capture0 input capture1						
19	INTTMRB3	16-bit TMRB (channel3)						
	INTTB3 INTTB3CAP0 INTTB3CAP1	match detection/0ver flow input capture0 input capture1						
20	INTTMRB4	16-bit TMRB (channel4)						
	INTTB4 INTTB4CAP0 INTTB4CAP1	match detection/0ver flow input capture0 input capture1						
21	INTTMRB5	16-bit TMRB (channel5)						
	INTTB5 INTTB5CAP0 INTTB5CAP1	match detection/0ver flow input capture0 input capture1						

Table 7-2 List of Interrupt Sources

No	Interrupt Source		The active level to release the low power consumption mode					CG interrupt mode control register
			"Low" level	"High" level	Rising edge	Falling edge	Both edge	
22	INTTMRB6	16-bit TMRB (channel6)						
	INTTB6 INTTB6CAP0 INTTB6CAP1	match detection/0ver flow input capture0 input capture1						
23	INTTMRB7	16-bit TMRB (channel7)						
	INTTB7 INTTB7CAP0 INTTB7CAP1	match detection/0ver flow input capture0 input capture1						
24	INTRX2	Serial reception (channel.2)						
25	INTTX2	Serial transmission (channel.2)						
26	INTRX3	Serial reception (channel.3)						
27	INTTX3	Serial transmission (channel.3)						
28	INTRX4	Serial reception (channel.4)						
29	INTTX4	Serial transmission (channel.4)						
30	INTADC							
	INTADHP INTADM0 INTADM1	Highest priority AD conversion complete interrupt AD conversion monitoring function interrupt 0 AD conversion monitoring function interrupt 1						
31	INTAD	AD conversion complete interrupt						

7.5.2.1 Active level

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

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

If an interrupt source is used for clearing a standby mode, setting the relevant clock generator register is also required. Enable the CGIMCGx<INTxEN> bit and specify the active level in the CGIMCGx<EMCGx> bits. You must set the active level for interrupt requests from each peripheral function as shown in Table 7-2.

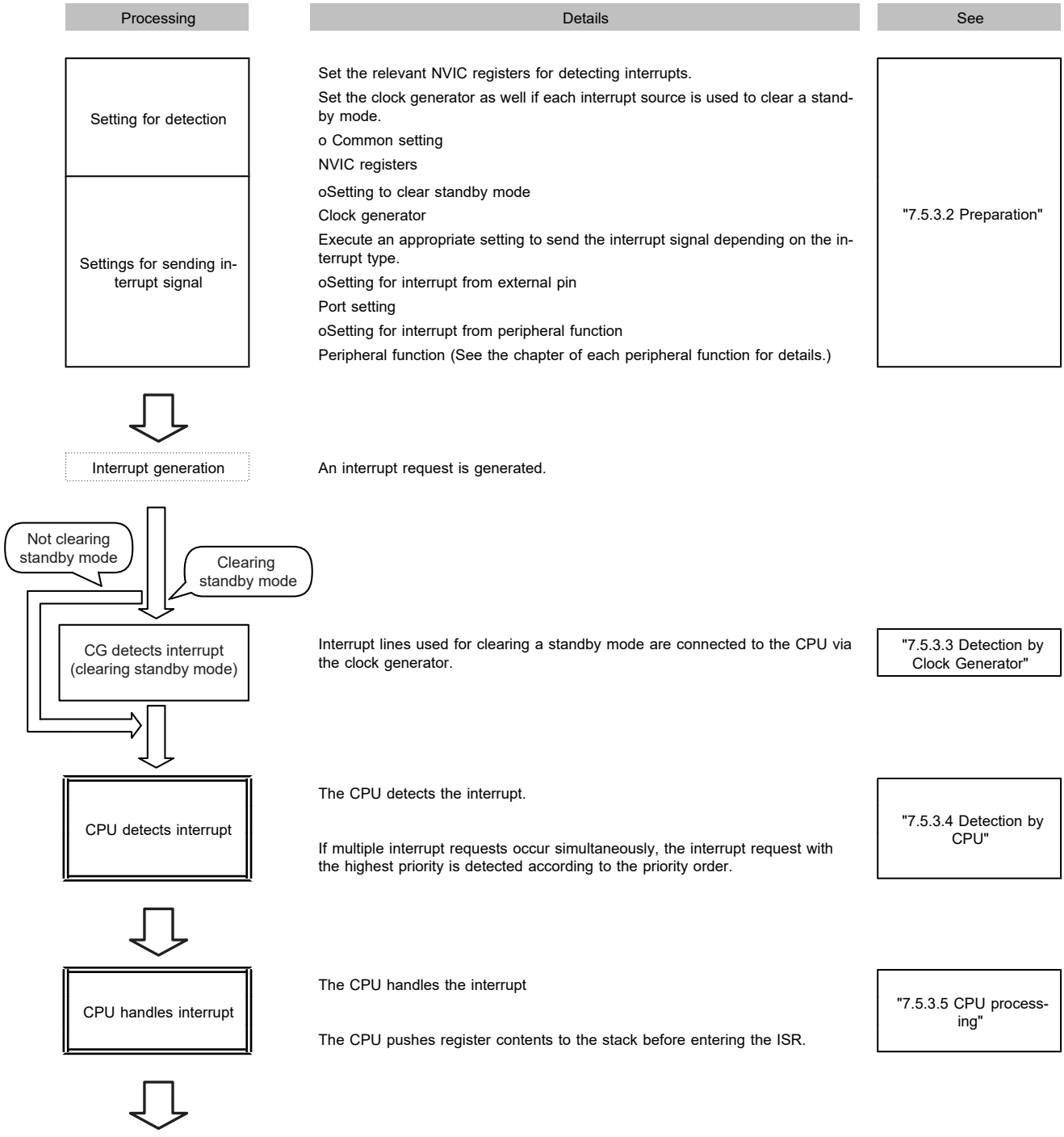
An interrupt request detected by the clock generator is notified to the CPU with a signal in "High" level.

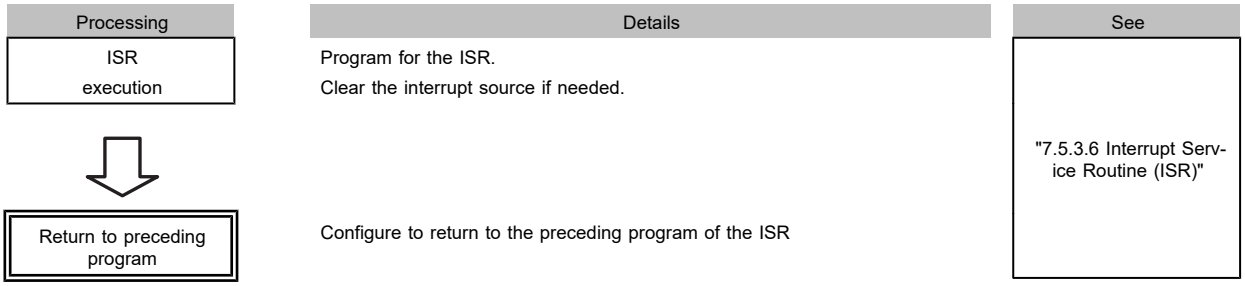
7.5.3 Interrupt Handling

7.5.3.1 Flowchart

The following shows how an interrupt is handled.

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





7.5.3.2 Preparation

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

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

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

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

1. Disabling interrupt by CPU
2. CPU register setting
3. Preconfiguration(1) (Interrupt form 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 register 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 two bits for assigning a priority level from 0 to 3. Priority level 0 is the highest priority level. If multiple sources have the same priority, the smallest-numbered interrupt source has the highest priority.

NVIC register		
<PRI_n>	←	"priority"

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

(3) Pre configuration (1) (Interrupt from external pin)

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

Port register		
PxIE<PxmIE>	←	"1"

Note: x: port number / m: corresponding bit.

Setting PxIE to enable input enables the corresponding interrupt input. Be careful not to enable interrupts that are not used.

(4) Pre configuration (2) (Interrupt from peripheral function)

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

(5) Pre configuration (3) (Interrupt Set-Pending Register)

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

NVIC register		
<SETPEND[m]>	←	"1"

Note: m: corresponding bit

(6) Configuring the clock generator

For an interrupt source to be used for exiting a standby mode, you need to set the active level and enable interrupts in the CGIMCG register of the clock generator. The CGIMCG register is capable of configuring each source. According to the active level, refer to"Table 7-2 List of Interrupt Sources".

Before enabling an interrupt, clear the corresponding interrupt request already held. This can avoid unexpected interrupt.To clear corresponding interrupt request, write a value corresponding to the interrupt to be used to the CGICRCG register.See"7.6.3.3 CGICRCG(CG Interrupt Request Clear register)"for each value.

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

Also, be aware of the description of "7.5.1.4 Precautions when using external interrupt pins".

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

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

(7) Enabling interrupt by CPU

Enable the interrupt by the CPU as shown below.

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

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

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

At the end, PRIMASK register is zero cleared.

NVIC register		
Interrupt clear-Pending <CLRPEND[m]>	←	"1"
Interrupt Set-Enable <SETENA[m]>	←	"1"
Interrupt mask register		
PRIMASK	←	"0"

Note 1: m: corresponding bit

7.5.3.3 Detection by Clock Generator

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

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

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

7.5.3.4 Detection by CPU

The CPU detects an interrupt request with the highest priority.

7.5.3.5 CPU processing

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

7.5.3.6 Interrupt Service Routine (ISR)

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

(1) Procedure during ISR

An ISR normally pushes register contents to the stack and handles an interrupt as required. The Cortex-M0 core automatically pushes the contents of xPSR, PC, LR, R12 and r3 to r0 to the stack. No extra programming is required for them.

Push the contents of other registers if needed.

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

(2) Clearing an interrupt source

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

If an interrupt 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.

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.

7.6 Exception/Interrupt-Related Registers

7.6.1 Register List

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

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

NVIC registers — Base Address = 0xE000_E000

Register name	Address
SysTick Control and Status Register	0x0010
SysTick Reload Value Register	0x0014
SysTick Current Value Register	0x0018
SysTick Calibration Value Register	0x001C
Interrupt Set-Enable Register	0x0100
Interrupt Clear-Enable Register	0x0180
Interrupt Set-Pending Register	0x0200
Interrupt Clear-Pending Register	0x0280
Interrupt Priority Register	0x0400 ~ 0x041F
Application Interrupt and Reset Control Register	0x0D0C
System Handler Priority Register	0x0D1C, 0x0D20
System Handler Control and State Register	0x0D24

peripheral function name: CG

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

7.6.2 NVIC registers

7.6.2.1 SysTick Control and Status Register

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

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

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

7.6.2.2 SysTick Reload Value Register

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

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

7.6.2.3 SysTick Correct Value Register

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

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

7.6.2.4 SysTick Calibration Value Register

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

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

Note: This product does not prepare the calibration value.

7.6.2.5 Interrupt Set-Enable Register 1

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

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

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.2 List of Interrupt Sources".

7.6.2.6 Interrupt Clear-Enable Register 1

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

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

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.2 List of Interrupt Sources".

7.6.2.7 Interrupt Set-Pending Register 1

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

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

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.2 List of Interrupt Sources".

7.6.2.8 Interrupt Clear-Pending Register 1

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

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

Note: For descriptions of interrupts and interrupt numbers, see Section "7.5.2 List of Interrupt Sources".

7.6.2.9 Interrupt Priority Register

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

	31 24	23 16	15 8	7 0
0xE000_E400	PRI_3	PRI_2	PRI_1	PRI_0
0xE000_E404	PRI_7	PRI_6	PRI_5	PRI_4
0xE000_E408	PRI_11	PRI_10	PRI_9	PRI_8
0xE000_E40C	PRI_15	PRI_14	PRI_13	PRI_12
0xE000_E410	PRI_19	PRI_18	PRI_17	PRI_16
0xE000_E414	PRI_23	PRI_22	PRI_21	PRI_20
0xE000_E418	PRI_27	PRI_26	PRI_25	PRI_24
0xE000_E41C	PRI_31	PRI_30	PRI_29	PRI_28

Cortex-M0 core uses two 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
31-30	PRI_3	R/W	Priority of interrupt number 3
29-24	–	R	Read as 0,
23-22	PRI_2	R/W	Priority of interrupt number 2
21-16	–	R	Read as 0,
15-14	PRI_1	R/W	Priority of interrupt number 1
13-8	–	R	Read as 0,
7-6	PRI_0	R/W	Priority of interrupt number 0
5-0	–	R	Read as 0,

7.6.2.10 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	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	SYSRESET REQ	VECTCLR ACTIVE	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	VECTKEY (Written) / VECTKEYSTAT (Read)	R/W	Register key [Write] Writing to this register requires 0x5FA in the <VECTKEY> field. [Read] Read as 0xFA05.
15	ENDIANESS	R/W	Endianness bit: (Note1) 1: Big endian 0: Little endian
14-3	−	R	read as "0".
2	SYSRESET REQ	R/W	System Reset Request 1=CPU outputs a SYSRESETREQ signal. (note2)
1	VECTCLR ACTIVE	R/W	Clear active vector bit 1: clear all state information for active NMI, fault, and interrupts. 0: do not clear. This bit self-clears. It it the responsibility of the application to reinitialize the stack.
0	−	R	Read as "0".

Note 1: This product can be used as the little-endian memory format only.

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

7.6.2.11 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_ED1C	PRI_11 (SVCall)	PRI_10	PRI_9	PRI_8
0xE000_ED20	PRI_15 (SysTick)	PRI_14 (PendSV)	PRI_13	PRI_12

Cortex-M0 core uses two 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_15		-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	PRI_14		-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	PRI_13		-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	PRI_12		-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-30	PRI_15	R/W	Priority of SysTick
29-24	−	R	Read as "0".
23-22	PRI_14	R/W	Priority of PendSV.
21-16	−	R	Read as "0".
15-14	PRI_13	R/W	Reserved.
13-8	−	R	Read as "0".
7-6	PRI_12	R/W	Reserved.
5-0	−	R	Read as "0".

7.6.2.12 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	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	SVCALL PENDED	-	-	-	-	-	-	-
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 "0".
15	SVCALL PENDED	R/W	SVCall 0: Not pended. 1: Pended.
14-0	−	R	Read as "0".

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

7.6.3 Clock generator registers

7.6.3.1 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	-	0
	23	22	21	20	19	18	17	16
bit symbol	-	EMCG2			EMST2		-	INT2EN
After reset	0	0	1	0	0	0	-	0
	15	14	13	12	11	10	9	8
bit symbol	-	EMCG1			EMST1		-	INT1EN
After reset	0	0	1	0	0	0	-	0
	7	6	5	4	3	2	1	0
bit symbol	-	EMCG0			EMST0		-	INT0EN
After reset	0	0	1	0	0	0	-	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 requeset.(101 to 111:setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
27-26	EMST3[1:0]	R	active level of INT3 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges
25	−	R	Read 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 requeset.(101 to 111:setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
19-18	EMST2[1:0]	R	active level of INT2 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges
17	−	R	Read 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 requeset.(101 to 111:setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
11-10	EMST1[1:0]	R	active level of INT1 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges
9	−	R	Read 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 requeset.(101 to 111:setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
3-2	EMST0[1:0]	R	active level of INT0 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges
1	−	R	Read as undefined.
0	INT0EN	R/W	INT0 clear input 0: Disable 1: Enable

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

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

7.6.3.2 CGIMCGB(CG Interrupt Mode Control register)

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

Bit	Bit Symbol	Type	Function
31	−	R	Read as "0"
30-28	−	R/W	Write "000".
27-25	−	R	Read as undefined.
24	−	R/W	Write "0".
23	−	R	Read as "0"
22-20	−	R/W	write "000".
19-17	−	R	Read as undefined.
16	−	R/W	Write "0".
15	−	R	Read as "0".
14-12	EMCG5[2:0]	R/W	active level setting of INT5 standby clear requeset.(101 to 111:setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
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	Read 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 requeset.(101 to 111:setting prohibited) 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edges
3-2	EMST4[1:0]	R	active level of INT4 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges
1	−	R	Read as undefined.

Bit	Bit Symbol	Type	Function
0	INT4EN	R/W	INT4 clear input 0: Disable 1: Enable

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

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

7.6.3.3 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 request. 0_0000: INT0 0_0001: INT1 0_0010: INT2 0_0011: INT3 0_0100: INT4 0_0101: INT5 0_0110Å`1_1111: setting prohibited. Read as "0".

7.6.3.4 CGRSTFLG (Reset Flag Register)

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

Bit	Bit Symbol	Type	Function
31-7	−	R	Read as "0".
6	LVDRSTF	R/W	LVD reset flag [Read] 0: - 1: Reset from LVD. [Write] 0: Clear reset flag (Note2) 1: Don't care
5	−	R	Read as "0".
4	SYSRSTF	R/W	Debug reset flag(Note1) [Read] 0: - 1: Reset from SYSRESETREQ. [Write] 0: Clear reset flag (Note2) 1: Don't care
3	−	R	Read as "0".
2	WDTRSTF	R/W	WDT reset flag [Read] 0: - 1: Reset from WDT. [Write] 0: Clear reset flag (Note2) 1: Don't care
1	PINRSTF	R/W	$\overline{\text{RESET}}$ pin flag [Read] 0: - 1: Reset from $\overline{\text{RESET}}$ pin. [Write] 0: Clear reset flag (Note2) 1: Don't care
0	PONRSTF	R/W	Power-On-reset flag [Read] 0: - 1: Reset from Power-On-Reset [Write] 0: Clear reset flag (Note2) 1: Don't care

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

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

7.6.3.5 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	-	-	-	-	NMIFLG3	NMIFLG2	-	NMIFLG0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Functions
31-5	−	R	Read as "0".
4	−	R	Read as "0".
3	NMIFLG3	R	NMI source generation flag 0: not applicable 1: generated from LVD when returning from low voltage.
2	NMIFLG2	R	NMI source generation flag 0: not applicable 1: generated from LVD, Only lower than the setting voltage when voltage decreasing.
1	−	R	Read as "0".
0	NMIFLG0	R	NMI source generation flag 0: not applicable 1: generated from WDT.

8. DMA Controller(DMAC)

8.1 Overview

The table below lists its major functions.

Table 8-1 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 "ProductoInformation" chapter. .
Interrupt function	Transfer end interrupt (INTDMACxTC) Error interrupt (INTDMACxERR)		-
Special Function	Scatter/gather function		-

8.2 DMA transfer type

Table 8-2 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.

8.3 Block diagram

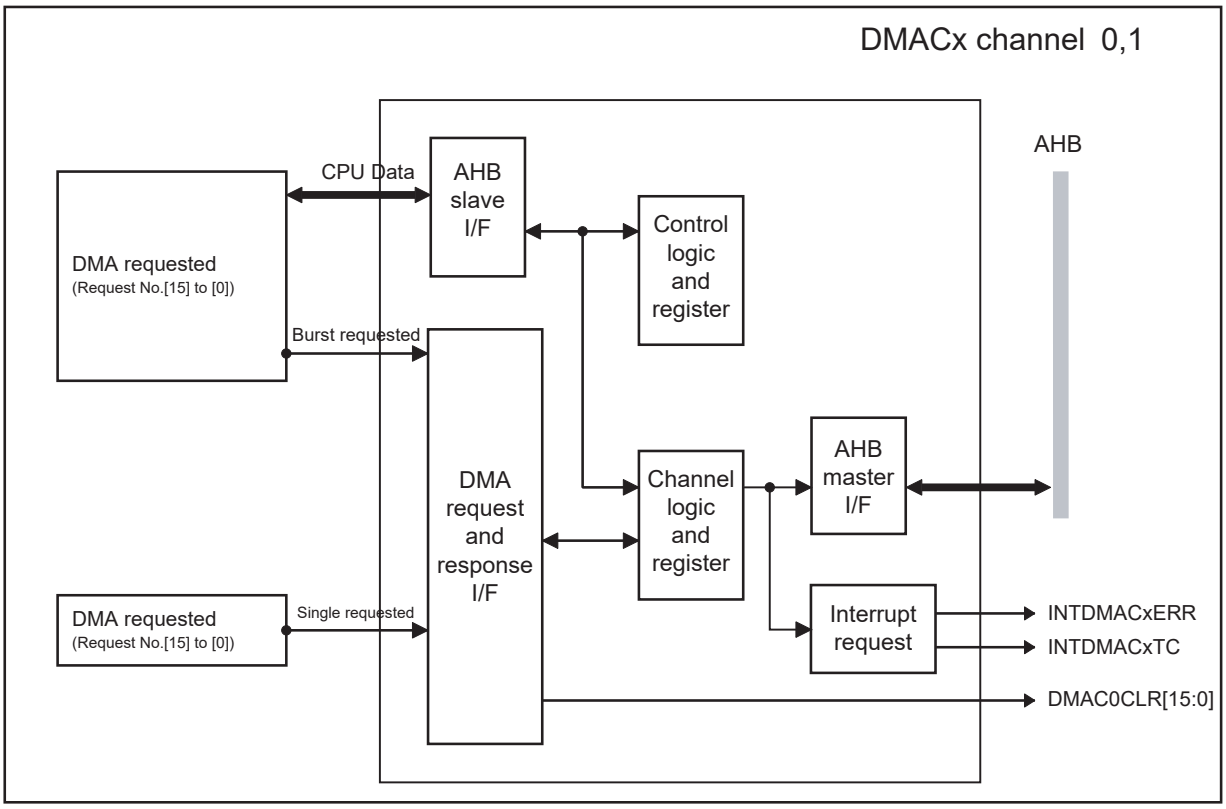


Figure 8-1 DMAC Block diagram

8.4 Description of Registers

8.4.1 DMAC register list

The function and address for each register are shown bellow.

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

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
DMAC Configuration Register	DMACxConfiguration	0x0030
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: 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 3: 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.

8.4.2 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.

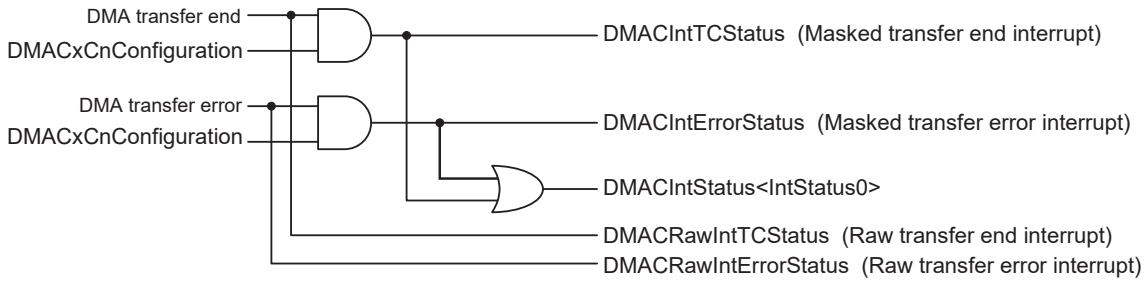


Figure 8-2 Interrupt-related block diagram

8.4.3 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.

8.4.4 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.

8.4.5 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.

8.4.6 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.

8.4.7 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

8.4.8 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

8.4.9 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.

8.4.10 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 "Product information" chapter for DMA request number. Clear "0" to bit corresponded with the DMA request number which has no burst request.

8.4.11 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 "Produnct information" chapter for DMA request number. Clear "0" to bit corresponded with the DMA request number which has no single request.

8.4.12 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	-	-	-	-	-	-	-	E
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Write as zero.
1	-	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".

8.4.13 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.

8.4.14 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.

8.4.15 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	Description
31-2	LLI[29:0]	R/W	Sets the first address of the next transfer information. Set a value within 0xFFFF_FFF0. When <LLI> = 0, LLI is the last chain. After DMA transfer finishes, the DMA channel is disabled.
1-0	-	W	Write as zero.

Note: For <LLI> detailed operation, see "8.5 Special Functions".

8.4.16 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 8-3 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 8-3 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 8-3 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 8-3 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 in DMA running, Read as Zero in DMA stopping. 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 8-3 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.

8.4.17 DMACxCnConfiguration (DMAC Channel n Configuration Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	Halt	Active	Lock
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	ITC	IE	FlowCntrl			-	DestPeripheral	
After reset	0	0	0	0	0	Undefined	0	0
	7	6	5	4	3	2	1	0
bit symbol	DestPeripheral		-	SrcPeripheral				E
After reset	0	0	Undefined	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-19	-	W	Write as zero.
18	Halt	R/W	Controls accepting a DMA request 0 : Accept a DMA request 1 : Ignore a DMA request
17	Active	R	Indicates whether data is present in the channel FIFO. 0 : No data exists in the FIFO 1 : Data exists in the FIFO
16	Lock	R/W	Sets a locked transfer (Non-divided transfer). 0 : Disable locked transfer 1: Enable locked transfer (Note3) When locked transfer is enabled, as many burst transfers as specified are consecutively executed without re-leasing the bus.
15	ITC	R/W	Transfer end interrupt enable register. 0 : Disable interrupt 1 : Enable interrupt
14	IE	R/W	Error interrupt enable register 0 :Disable interrupt 1: Enable interrupt
13-11	FlowCntrl[2:0]	R/W	Sets transfer method <FlowCntrl[2:0]> setting value / Transfer method 000: Memory to Memory (Note) 001: Memory to Peripheral 010: Peripheral to Memory 011: Peripheral to Peripheral 100 to 111: Reserved
10	-	W	Write as zero.
9-6	DestPeripheral [3:0]	R/W	DMA request number of transfer destination For DMA request number, refer to "Table 2-1 DMA request table". When a memory is the transfer destination, this setting is ignored.
5	-	W	Write as zero.
4-1	SrcPeripheral [3:0]	R/W	DMA request number of transfer source For DMA request number, refer to "Table 2-1 DMA request table". 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.

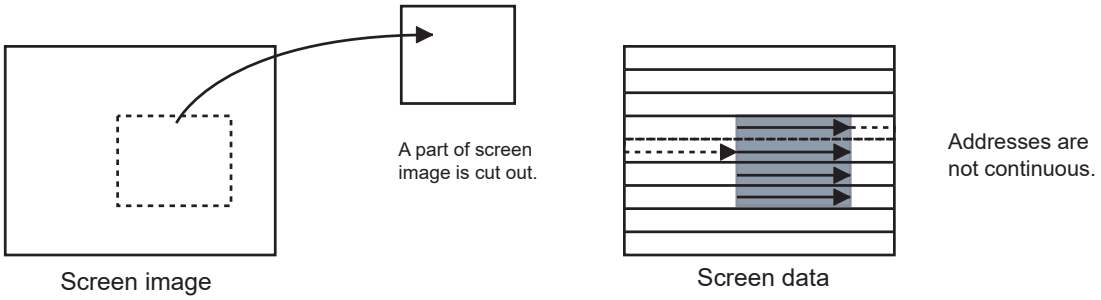
Note 2: When DMACxENableChns<EnabledCHx> is enabled and the corresponding DMACxCnConfiguration<Halt> is set to "1", write them after channel enable bit (E:bit0) is clear to "0". Without this, in the case of the slave error is occurred when writing them, the error is recovered by reset. Regarding slave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

8.5 Special Functions

8.5.1 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.

8.5.2 Linked list operation

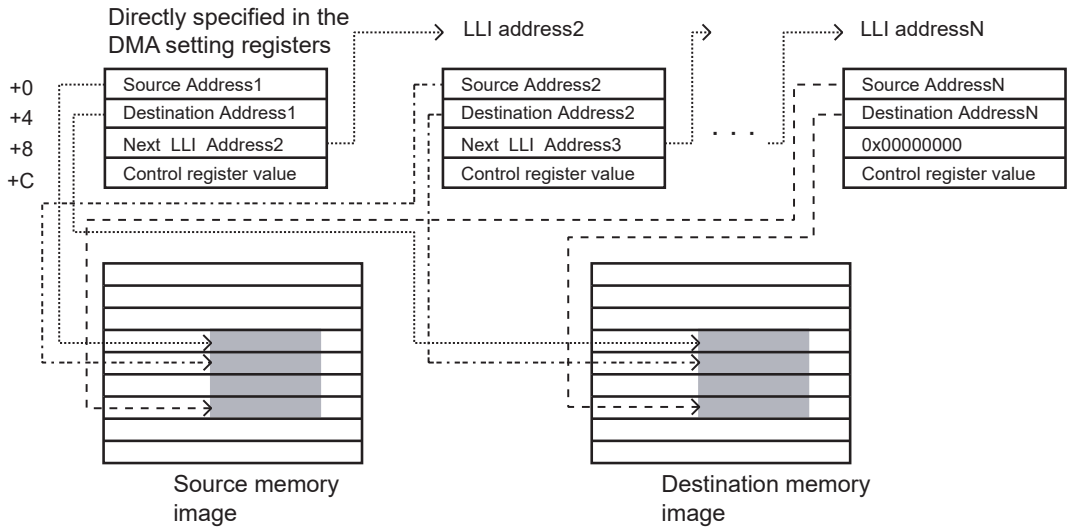
To operate the scatter/gather function, a transfer source and source data areas need to be defined by creating a set of Linked Lists first.

Each setting is called LLI (LinkedList).

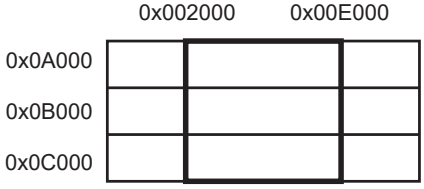
Each LLI controls the transfer of one block of data. Each LLI indicates normal DMA setting and controls transfer of successive data. Each time each DMA transfer is complete, the next LLI setting will be loaded to continue the DMA operation (Daisy Chain).

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.

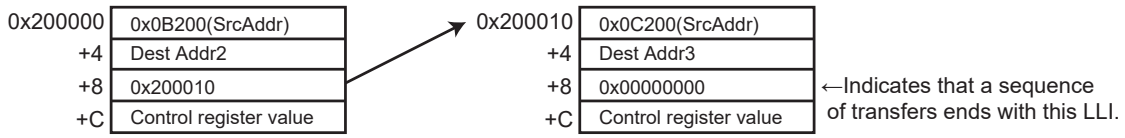


When transferring data in the area enclosed by the square



	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



9. Input/Output port

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

9.1 Registers

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

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

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

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

9.1.1 Register list

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

A bit without function is read as "0" and the writing a data to this bit is invalid

Register name	Address (Base+)	Port A	Port B	Port C	Port D	Port E
Data register	0x0000	PADATA	PBDATA	PCDATA	PDDATA	PEDATA
Output control register	0x0004	PACR	PBCR	PCCR	PDCR	PECR
Function register 1	0x0008	-	PBFR1	PCFR1	PDFR1	PEFR1
Function register 2	0x000C	-	PBFR2	-	-	-
Open-drain control register	0x0028	PAOD	PBOD	PCOD	PDOD	PEOD
Pull-up control register	0x002C	PAPUP	PBPUP	PCPUP	PDPUP	PEPUP
Pull-down control register	0x0030	PAPDN	PBPDN	PCPDN	PDPDN	PEPDN
Input control register	0x0038	PAIE	PBIE	PCIE	PDIE	PEIE

Register name	Address (Base+)	Port F	Port G
Data register	0x0000	PFDATA	PGDATA
Output control register	0x0004	PFCR	PGCR
Function register 1	0x0008	PFFR1	PGFR1
Function register 2	0x000C	-	-
Open-drain control register	0x0028	PFOD	PGOD
Pull-up control register	0x002C	PFPUP	PGPUP
Pull-down control register	0x0030	PFPDN	PGPDN
Input control register	0x0038	PFIE	PGIE

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

9.1.2 Port function and setting list

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

- "Table 9-1 Port A Setting List"
- "Table 9-2 Port B Setting List"
- "Table 9-3 Port C Setting List"
- "Table 9-4 Port D Setting List"
- "Table 9-5 Port E Setting List"
- "Table 9-6 Port F Setting List"
- "Table 9-7 Port G Setting List"

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

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

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

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

9.1.2.1 PORT A

Table 9-1 Port A Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PADATA	PACR	PAFRn	PAOD	PAPUP	PAPDN	PAIE
PA0	After reset			0	0		0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	AIN0	Input	FT5	0/1	0		0/1	0	0	0
PA1	After reset			0	0		0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	AIN1	Input	FT5	0/1	0		0/1	0	0	0
PA2	After reset			0	0		0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	AIN2	Input	FT5	0/1	0		0/1	0	0	0
PA3	After reset			0	0		0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	AIN3	Input	FT5	0/1	0		0/1	0	0	0
PA4	After reset			0	0		0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	AIN4	Input	FT5	0/1	0		0/1	0	0	0
PA5	After reset			0	0		0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	AIN5	Input	FT5	0/1	0		0/1	0	0	0
PA6	After reset			0	0		0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	AIN6	Input	FT5	0/1	0		0/1	0	0	0
PA7	After reset			0	0		0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	AIN7	Input	FT5	0/1	0		0/1	0	0	0

9.1.2.2 PORT B

Table 9-2 Port B Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PBDATA	PBCR	PBFRn	PBOD	PBPUP	PBPDN	PBIE
PB0 (note)	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
PB1	After reset			0	0	PBFR2	0	0	1	1
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC4RXD	Input	FT1	0/1	0	PBFR1	0/1	0/1	0/1	1
	SWCLK	Input	FT2	0/1	0	PBFR2	0	0	1	1
PB2	After reset			0	1	PBFR2	0	1	0	1
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC4TXD	Output	FT1	0/1	1	PBFR1	0/1	0/1	0/1	0
	SWDIO	I/O	FT2	0/1	1	PBFR2	0/	1	0	1
PB3	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC4SCLK	Input	FT1	0/1	0	PBFR1	0/1	0/1	0/1	1
		Output		0/1	1	PBFR1	0/1	0/1	0/1	0
PB4	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
PB5	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	INT0	Input	FT4	0/1	0		0/1	0/1	0/1	1
PB6	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	INT1	Input	FT4	0/1	0		0/1	0/1	0/1	1
PB7	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	INT2	Input	FT4	0/1	0		0/1	0/1	0/1	1

Note: PB0 works as a BOOT function. It is enabled to be input and pulled-up while $\overline{\text{RESET}}$ pin is "Low". At the rising edge of the reset signal, if PB0 is "High", the device enters single chip mode and boots from the on-chip flash memory. If PB0 is "Low", the device enters single BOOT mode and boots from the internal BOOT program.

9.1.2.3 PORT C

Table 9-3 Port C Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PCDATA	PCCR	PCFRn	PCOD	PCPUP	PCPDN	PCIE
PC0	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	I2C0SCL	I/O	FT1	0/1	1	PCFR1	1	0/1	0/1	1
PC1	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	I2C0SDA	I/O	FT1	0/1	1	PCFR1	1	0/1	0/1	1
PC2	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB2OUT	Output	FT1	0/1	1	PCFR1	0/1	0/1	0/1	0
PC3	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB0OUT	Output	FT1	0/1	1	PCFR1	0/1	0/1	0/1	0
PC4	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	T16A0OUT	Output	FT1	0/1	1	PCFR1	0/1	0/1	0/1	0
PC5	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB0IN	Input	FT1	0/1	0	PCFR1	0/1	0/1	0/1	1

9.1.2.4 PORT D

Table 9-4 Port D Setting List

PORT	Reset status	Input/Output	PORT Type	Control register						
				PDDATA	PDCR	PDFRn	PDOD	PDPUP	PDPDN	PDIE
PD0	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB3OUT	Output	FT1	0/1	1	PDFR1	0/1	0/1	0/1	0
PD1	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC0SCK	Input	FT1	0/1	0	PDFR1	0/1	0/1	0/1	1
		Output		0/1	1	PDFR1	0/1	0/1	0/1	0
PD2	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC0RXD	Input	FT1	0/1	0	PDFR1	0/1	0/1	0/1	1
PD3	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC0TXD	Output	FT1	0/1	1	PDFR1	0/1	0/1	0/1	0
PD4	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB3IN	Input	FT1	0/1	0	PDFR1	0/1	0/1	0/1	1
PD5	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0

9.1.2.5 PORT E

Table 9-5 Port E Setting List

PORT	Reset status	Input/Output	PORT Type	Control register						
				PEDATA	PECR	PEFRn	PEOD	PEPUP	PEPDN	PEIE
PE0	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
PE1	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
PE2	After reset			0	1	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC2SCK	Input	FT1	0/1	0	PEFR1	0/1	0/1	0/1	1
		Output		0/1	1	PEFR1	0/1	0/1	0/1	0
PE3	After reset			0	1	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC2RXD	Input	FT1	0/1	0	PEFR1	0/1	0/1	0/1	1
PE4	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC2TXD	Output	FT1	0/1	1	PEFR1	0/1	0/1	0/1	0
PE5	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	INT5	Input	FT4	0/1	0		0/1	0/1	0/1	1
PE6	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	INT4	Input	FT4	0/1	0		0/1	0/1	0/1	1
PE7	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	INT3	Input	FT4	0/1	0		0/1	0/1	0/1	1

9.1.2.6 PORT F

Table 9-6 Port F Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PFDATA	PFCR	PFFRn	PFOD	PFPUP	PFPDN	PFIE
PF0	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB7IN	Input	FT1	0/1	0	PFFR1	0/1	0/1	0/1	1
PF1	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC3SCK	Input	FT1	0/1	0	PFFR1	0/1	0/1	0/1	1
		Output		0/1	1	PFFR1	0/1	0/1	0/1	0
PF2	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC3RXD	Input	FT1	0/1	0	PFFR1	0/1	0/1	0/1	1
PF3	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC3TXD	Output	FT1	0/1	1	PFFR1	0/1	0/1	0/1	0
PF4	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB7OUT	Output	FT1	0/1	1	PFFR1	0/1	0/1	0/1	0
PF5	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	T16A1OUT	Output	FT1	0/1	1	PFFR1	0/1	0/1	0/1	0
PF6	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
PF7	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0

9.1.2.7 PORT G

Table 9-7 Port G Setting List

PORT	Reset status	Input/Output	PORT Type	Control registers						
				PGDATA	PGCR	PGFRn	PGOD	PGPUP	PGPDN	PGIE
PG0	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC1SCK	Input	FT1	0/1	0	PGFR1	0/1	0/1	0/1	1
		Output		0/1	1	PGFR1	0/1	0/1	0/1	0
PG1	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC1RXD	Input	FT1	0/1	0	PGFR1	0/1	0/1	0/1	1
PG2	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	SC1TXD	Output	FT1	0/1	1	PGFR1	0/1	0/1	0/1	0
PG3	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB1IN	Input	FT1	0/1	0	PGFR1	0/1	0/1	0/1	1
PG4	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB1OUT	Output	FT1	0/1	1	PGFR1	0/1	0/1	0/1	0
PG5	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB4OUT	Output	FT1	0/1	1	PGFR1	0/1	0/1	0/1	0
PG6	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB5OUT	Output	FT1	0/1	1	PGFR1	0/1	0/1	0/1	0
PG7	After reset			0	0	0	0	0	0	0
	Input Port	Input		0/1	0		0/1	0/1	0/1	1
	Output Port	Output		0/1	1		0/1	0/1	0/1	0
	TB6OUT	Output	FT1	0/1	1	PGFR1	0/1	0/1	0/1	0

9.1.3 Block Diagrams of Ports

9.1.3.1 Port Type

The ports are classified as shown below. Please refer to the following pages for the block diagrams of each port type. A dotted box in the figure indicates the part of the equivalent circuit described in the "Block diagrams of ports".

Table 9-8 Function List

Type	Function Input /Output / Input/ Output	Pull-up	Pull-down	Analog
FT1	Input /Output	R	R	−
FT4	Input (INT)	R	R	−
FT5	Input (AIN)	R	R	o
FT6	Input ($\overline{\text{BOOT}}$)	EnR	R	−

int: interrupt input

−: no exist

o: Exist

R: Forced disable during reset

EnR: Forced enable during reset

9.1.3.2 Type FT1

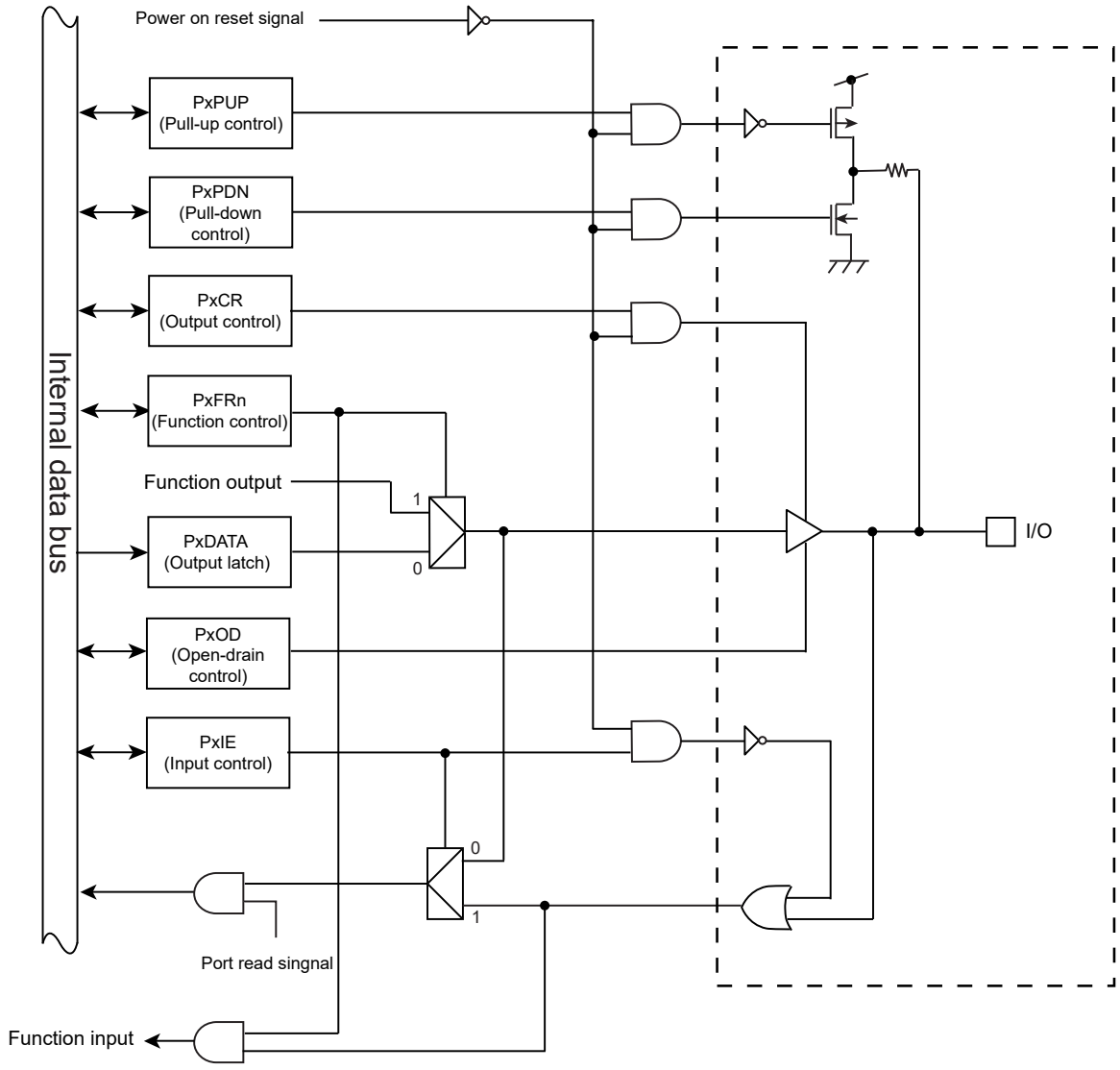


Figure 9-1 Port Type FT1

9.1.3.3 Type FT4

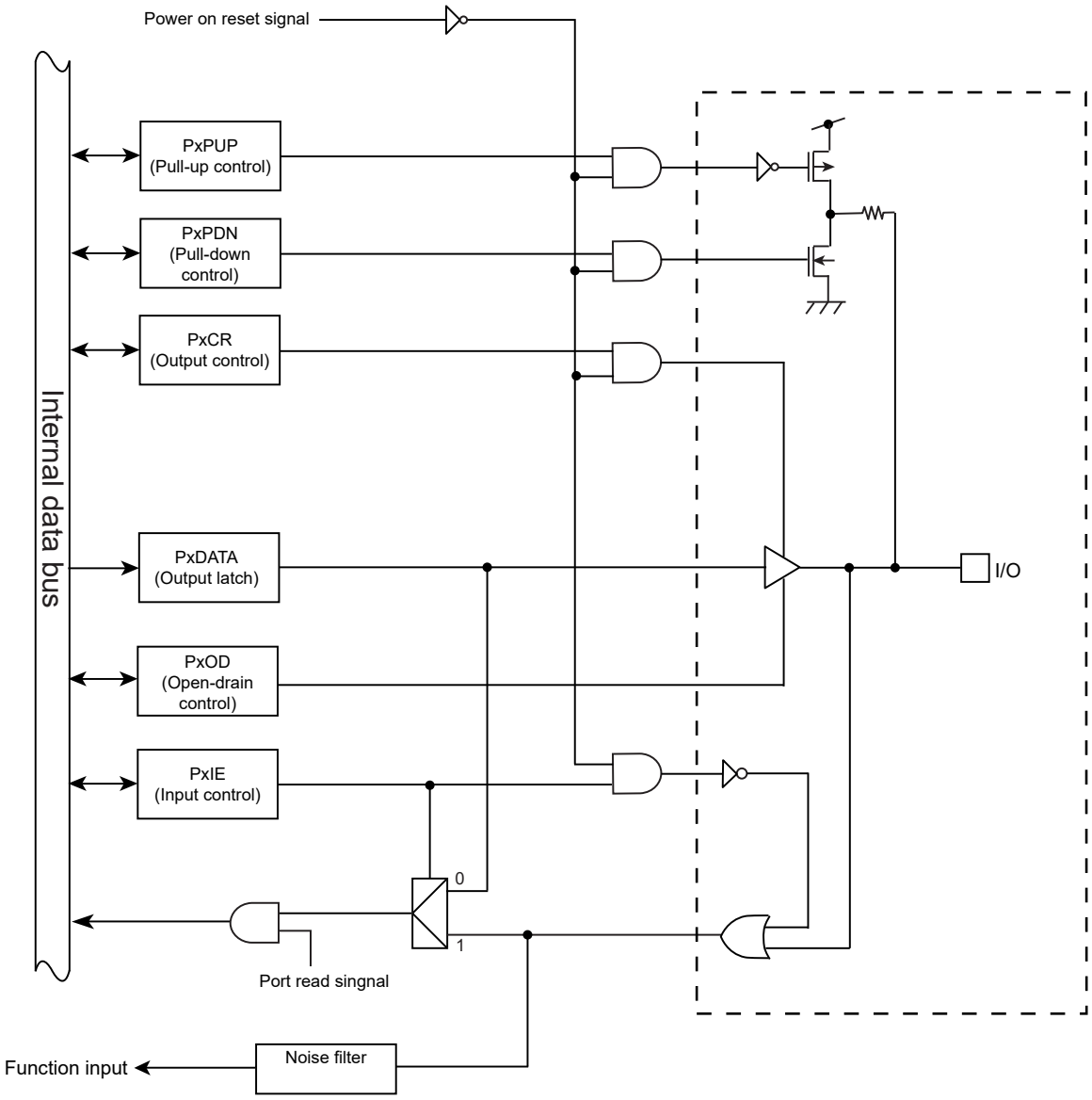


Figure 9-2 Port Type FT4

9.1.3.4 Type FT5

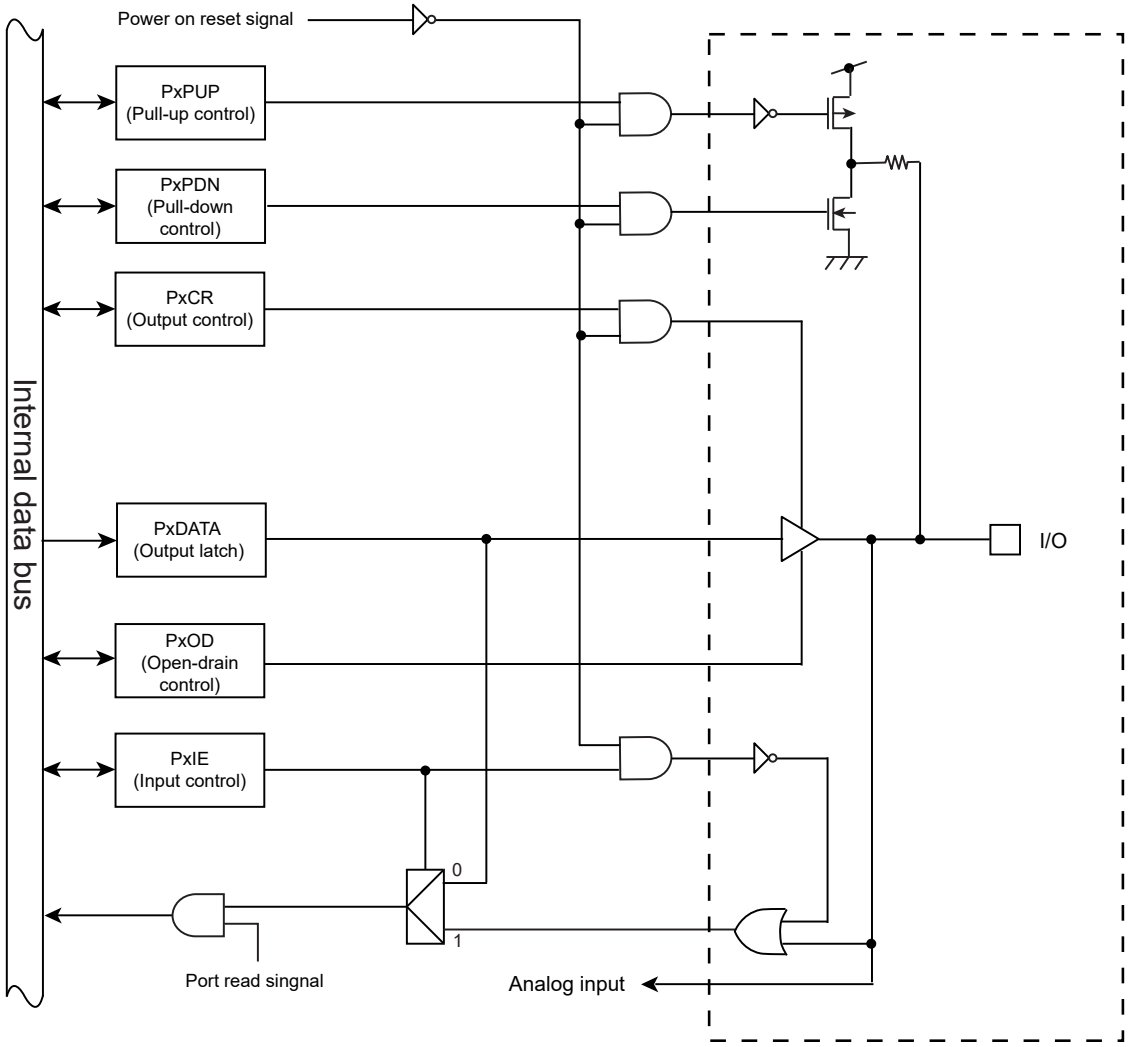


Figure 9-3 Port Type FT5

9.1.3.5 Type FT6

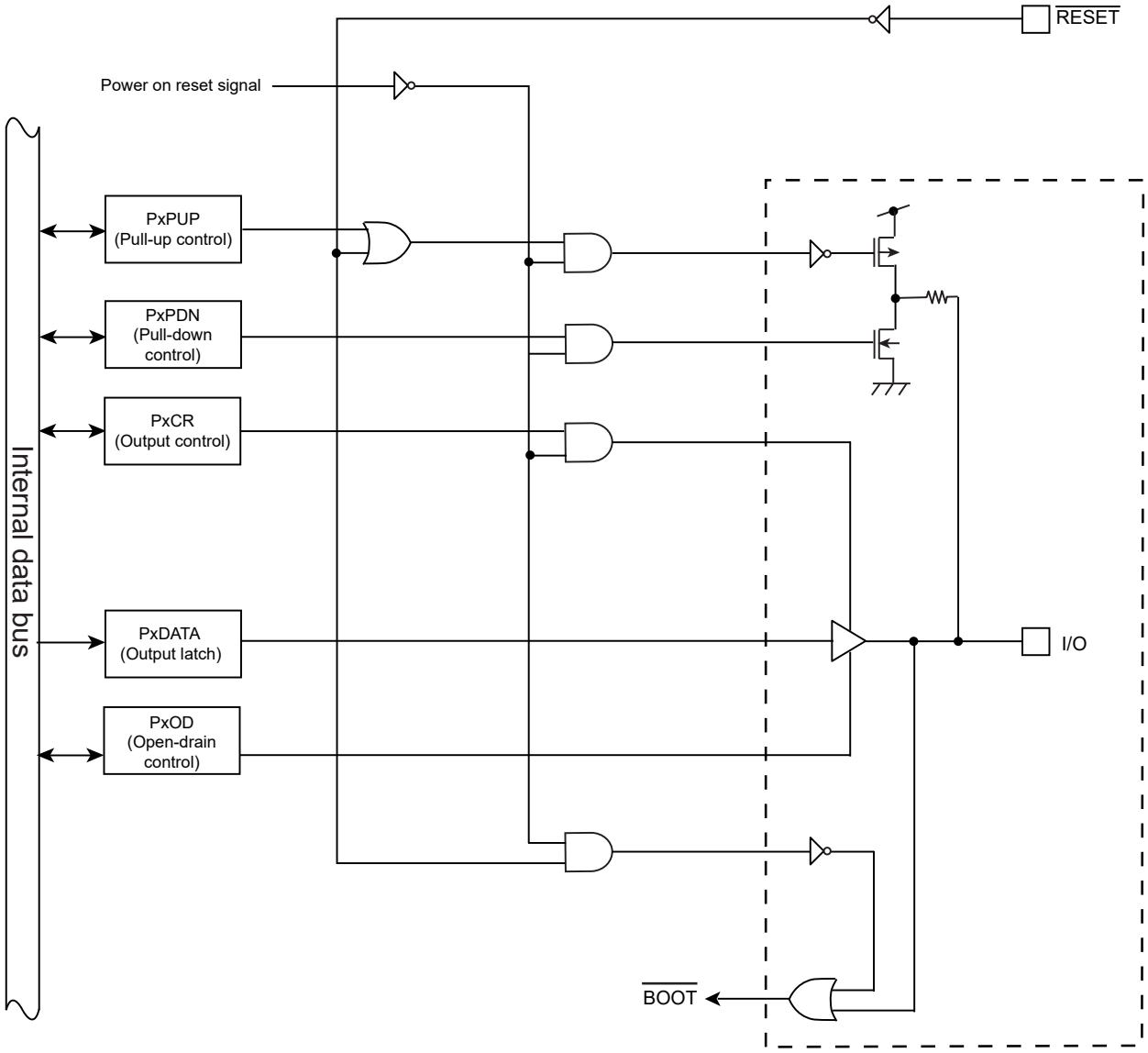


Figure 9-4 Port Type FT6

10. 16-bit Timer / Event Counters (TMRB)

10.1 Outline

TMRBx has the operation modes shown below.

- Interval timer mode
- Event counter mode
- Programmable pulse generation (PPG) mode
- Programmable pulse generation (PPG) external trigger mode

The use of the capture function allows TMRBx to perform the following measurements.

- Frequency measurement
- Pulse width measurement

10.2 Block Diagram

TMRBx consists of a 16-bit up-counter, two 16-bit timer register (Double-buffered), two 16-bit capture registers, two comparators, a capture input control, a timer flip-flop and its associated control circuit. Timer operation modes and the timer flip-flop are controlled by registers.

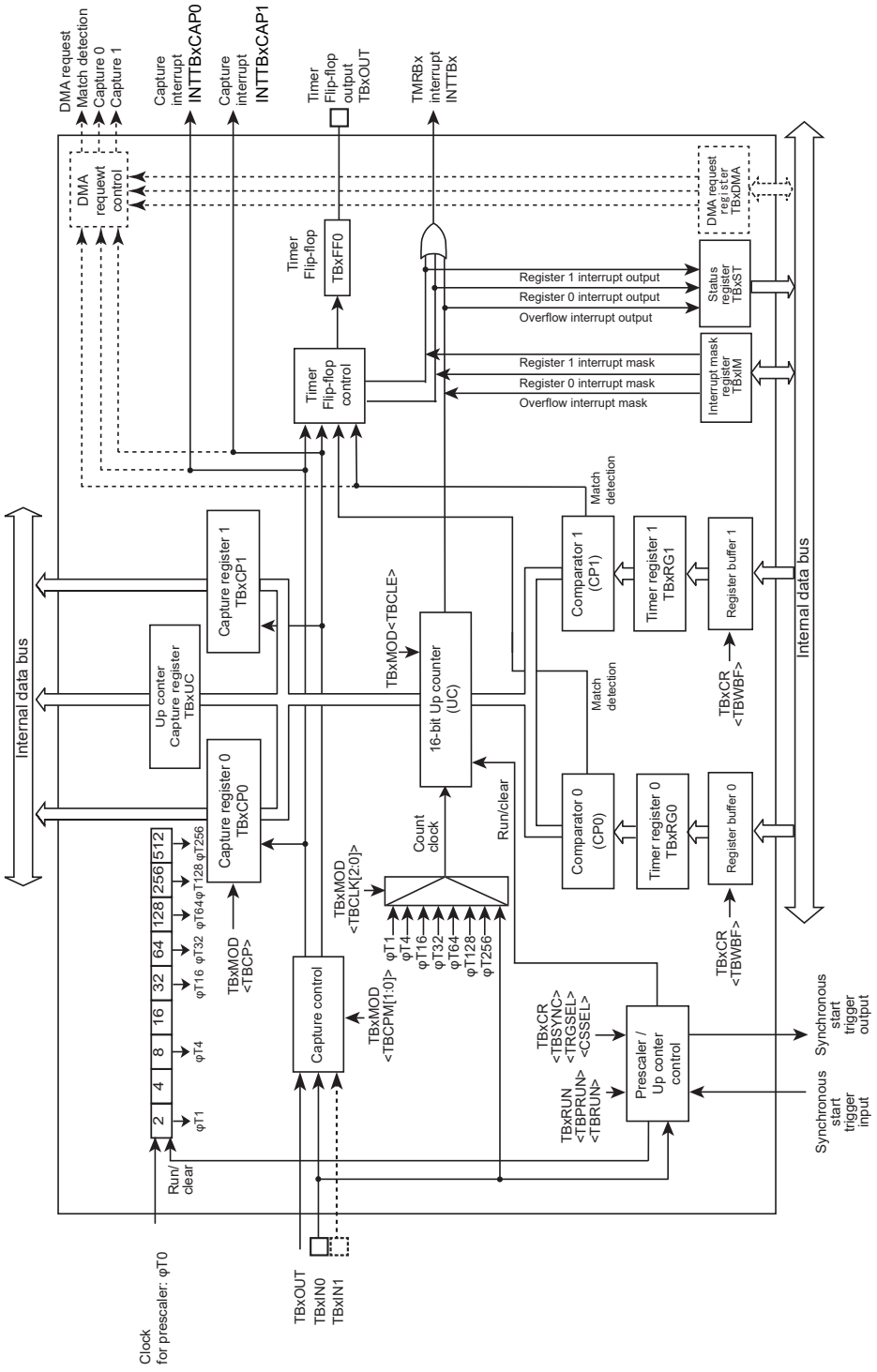


Figure 10-1 TMRBx Block Diagram

10.3 Registers

10.3.1 Register List

The table below shows control registers and their addresses.

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

Register name		Address (Base+)
Enable register	TBxEN	0x0000
RUN register	TBxRUN	0x0004
Control register	TBxCR	0x0008
Mode register	TBxMOD	0x000C
Flip-flop control register	TBxFFCR	0x0010
Status register	TBxST	0x0014
Interrupt mask register	TBxIM	0x0018
Up counter capture register	TBxUC	0x001C
Timer register 0	TBxRG0	0x0020
Timer register 1	TBxRG1	0x0024
Capture register 0	TBxCP0	0x0028
Capture register 1	TBxCP1	0x002C
DMA request enable register	TBxDMA	0x0030

Note: Do not modify the timer control register, timer mode register and timer flip-flop control register during timer operation. Users can modify them after stopping timer operation.

10.3.2 TBxEN (Enable Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TBEN	R/W	TMRBx operation 0: Disabled 1: Enabled Specifies the TMRBx operation. When the operation is disabled, no clock is supplied to the other registers in the TMRBx. This can reduce power consumption. (This disables reading from and writing to the other registers except TBxEN register.) To use the TMRB, enable the TMRBx operation (set to "1") before programming each register in the TMRBx. If the TMRBx operation is executed and then disabled, the settings will be maintained in each register.
6	TBHALT	R/W	Clock operation during debug HALT 0: run 1: stop Specifies the TMRBx clock setting to run or stop when the debug tool transits to HALT mode while in use.
5-0	−	R	Read as "0".

10.3.3 TBxRUN (RUN Register)

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

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	TBPRUN	R/W	Prescaler operation 0: Stop & clear 1: Count
1	−	R	Read as "0".
0	TBRUN	R/W	Count operation 0: Stop & clear 1: Count

10.3.4 TBxCR (Control Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBWBF	-	TBSYNC	-	I2TB	-	TRGSEL	CSSEL
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TBWBF	R/W	Double Buffer 0: Disabled 1: Enabled
6	−	R/W	Write "0".
5	TBSYNC	R/W	Synchronous mode switching 0: individual (Each channel) 1: synchronous
4	−	R	Read as "0".
3	I2TB	R/W	Operation at IDLE mode 0: Stop 1:Operation
2	−	R/W	Write "0".
1	TRGSEL	R/W	Selects the edge when the external trigger is used. 0: rising 1: falling Selects the edge when counting is started by external triggers (TBxIN).
0	CSSEL	R/W	Selects the count start 0: starts by software 1: starts by external trigger

10.3.5 TBxMOD (Mode Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
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) captures a count value. Read as "1".
5-4	TBCPM[1:0]	R/W	Capture timing 00: Disabled 01: Reserved 10: TBxIN↑ TBxIN↓ Captures a counter value on rising edge of TBxIN input into Capture register 0 (TBxCP0). Captures a counter value on falling edge of TBxIN input into Capture register 1 (TBxCP1). 11: TBxFF0↑ TBxFF0↓ Captures a counter value on rising edge of TBxFF0 input into Capture register 0 (TBxCP0). Captures a counter value on falling edge of TBxFF0 input into Capture register 1 (TBxCP1).
3	TBCLE	R/W	Up-counter control 0: Disables clearing of the up-counter. 1: Enables clearing of the up-counter. Clears and controls the up-counter. When "0" is written, it disables clearing of the up-counter. When "1" is written, it clears up-counter when up-counter matches with timer regsiter1 (TBxRG1).
2-0	TBCLK[2:0]	R/W	Selects the TMRBx source clock. 000: TBxIN pin input 001: φT1 010: φT4 011: φT16 100: φT32 101: φT64 110: φT128 111: φT256

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

10.3.6 TBxFFCR (Flip-Flop Control Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	TBC1T1	TBC0T1	TBE1T1	TBE0T1	TBFF0C	
After reset	1	1	0	0	0	0	1	1

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-6	−	R	Read as "1".
5	TBC1T1	R/W	TBxFF0 reverse trigger when the up-counter value is taken into the TBxCP1. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when the up-counter value is taken into the capture register 1 (TBxCP1).
4	TBC0T1	R/W	TBxFF0 reverse trigger when the up-counter value is taken into the TBxCP0. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when the up-counter value is taken into the capture register 0 (TBxCP0).
3	TBE1T1	R/W	TBxFF0 reverse trigger when the up-counter value is matched with TBxRG1. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when the up-counter value is matched with the timer register 1 (TBxRG1).
2	TBE0T1	R/W	TBxFF0 reverse trigger when the up-counter value is matched with TBxRG0. 0: Disable trigger 1: Enable trigger By setting "1", the timer-flip-flop reverses when an up-counter value is matched with the timer register 0 (TBxRG0).
1-0	TBFF0C[1:0]	R/W	TBxFF0 control 00: Invert Reverses the value of TBxFF0 (reversed by software). 01: Set Sets TBxFF0 to "1". 10: Clear Clears TBxFF0 to "0". 11: Don't care This is always read as "11".

10.3.7 TBxST (Status Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	INTTBOF	INTTB1	INTTB0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	INTTBOF	R	Overflow interrupt request flag 0:No overflow occurs 1:Overflow occurs When an up-counter is overflow, "1" is set.
1	INTTB1	R	Match (TBxRG1) interrupt request flag 0:No match is detected. 1:Detects a match with TBxRG1 When a match with the timer register 1 (TBxRG1) is detected, "1" is set.
0	INTTB0	R	Match(TBxRG0) interrupt request flag 0:No match is detected 1:Detects a match with TBxRG0 When a match with the timer register 0 (TBxRG0) is detected, "1" is set.

Note 1: Even if mask configuration by TBxIM register is valid, the status is set to TBxST register.

Note 2: When the interrupt mask configuration is disabled by the corresponding bit of TBxIM register, the interrupt is issued to the CPU.

Note 3: To clear the flag, read TBxST register.

10.3.8 TBxIM (Interrupt Mask Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	-	-	-	-	-	TBIMOF	TBIM1	TBIM0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-3	−	R	Read as "0".
2	TBIMOF	R/W	Overflow interrupt request mask 0:Disable 1:Enable Sets the up-counter overflow interrupt to disable or enable.
1	TBIM1	R/W	Match (TBxRG1) interrupt request mask 0:Disable 1:Enable Sets the match interrupt request mask with the timer register 1 (TBxRG1) to enable or disable.
0	TBIM0	R/W	Match (TBxRG0) interrupt request mask 0:Disable 1:Enable Sets the match interrupt request mask with the Timer register 0 (TBxRG0) to enable or disable.

Note: Even if mask configuration by TBxIM register is valid, the status is set to TBxST register.

10.3.9 TBxUC (Up-counter Capture Register)

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

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-0	TBUC[15:0]	R	Captures a value by reading up-counter out. If TBxUC is read during the counter operation, the current value of up-counter will be captured.

10.3.10 TBxRG0 (Timer Register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBRG0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBRG0							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-0	TBRG0[15:0]	R/W	Sets a value comparing to the up-counter.

10.3.11 TBxRG1 (Timer Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBRG1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TBRG1							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-0	TBRG1[15:0]	R/W	Sets a value comparing to the up-counter.

10.3.12 TBxCP0 (Capture register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBCP0							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	TBCP0							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-0	TBCP0[15:0]	R	A value captured from the up-counter is read.

10.3.13 TBxCP1 (Capture Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	TBCP1							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	7	6	5	4	3	2	1	0
bit symbol	TBCP1							
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-0	TBCP1[15:0]	R	A value captured from the up-counter is read.

10.3.14 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 with TBxRG1. (Note 1) 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<TBIM1> register is valid, DMA request does not occur even if a DMA request is enabled with <TBDMAEN2>.

Note 2: The assignment of DMA request factor differs by channel. Refer to "Product Information" Chapter

10.4 Description of Operation

10.4.1 Prescaler

TMRBx has 4-bit prescaler to generate the source clock for up-counter.

A input clock φT0 to the prescaler is either among fperiph/1, fperiph/2, fperiph/4, fperiph/8, fperiph/16 or fperiph/32 selected by CGSYSCR<PRCK[2:0] in the CG circuit. The peripheral clock is either fgear (a clock selected by CGSYSCR<FPSEL> in the CG circuit) or fc (a clock before divided by the clock gear).

The operation or the stoppage of prescaler is set by TBxRUN<TBPRUN>. Writing "1" to the bit is to start counting; writing "0" to the bit is to stop counting.

10.4.2 Up-counter (UC)

UC is a 16-bit binary counter.

10.4.2.1 Source clock

The up-counter's source clock is specified by TBxMOD<TBCLK[2:0]>.

It can be selected from the prescaler output clock - φT1, φT4, φT16, φT32, φT64, φT128 and φT256 - or the external clock of the TBxIN pin.

10.4.2.2 Counter start / stop

There are software start, external trigger start and synchronous start to start the counter.

1. Software start

   If <TBRUN> is set to "1", the counter will start. If "0" is set to the <TBRUN>, the counter will stop and the up-counter will be cleared at the same time.

2. External trigger start

   In the external trigger mode, the counter will be started by external signals.

   If TBxCR<CSSEL> is set to "1", the external trigger start mode is set. At this time, if <TBRUN> is set to "1", the condition of the counter will be trigger wait. The counter will start on the rising/falling edge of TBxIN.

   TBxCR<TRGSEL> bit specifies the switching external trigger edges.

   <TRGSEL>="0": Rising edge of TBxIN is selected.

   <TRGSEL>="1": Falling edge of TBxIN is selected.

   If <TBRUN> is set to "0", the counter will stop and the up-counter will be cleared at the same time.

3. Synchronous start

   In the timer synchronous mode, synchronous start timers can be possible. If timer synchronous mode is used in the PPG output mode, motor drive application can be achieved.

   Depending on products, the combination of master channels and slave channels have already been determined. For the combination of master channels and slave channels of this product, refer to Chapter Product Information.

TBxCR<TBSYNC> bit specifies the switching of synchronous mode. If <TBSYNC> bit of a slave channel is set to "1", the counter will start/stop synchronously with the software or external trigger start of a master channel. TBxRUN<TBPRUN, TBRUN> bit of a slave channel is not required to set. <TBSYNC> bit of the master channel must be set to "0".

Note that if the external trigger counter mode and timer synchronous mode are both set, the timer synchronous mode gains a higher priority.

10.4.2.3 Counter Clear

The up-counter is cleared at the timings below:

1. When a match with TBxRG1 is detected

   By setting TBxMOD<TBCLE> = "1", up-counter is cleared if the comparator detects a match between UC and TBxRG1.

2. When up-counter stops

   Up-counter stops and is cleared when TBxRUN<TBRUN> = "0".

10.4.2.4 Up-Counter Overflow

If up-counter overflows, the INTTBx overflow interrupt is generated.

10.4.3 Timer Registers (TBxRG0, TBxRG1)

TBxRG0 and TBxRG1 are registers to compare with a value of up-counter. The two registers are built into each channel. If the comparator detects a match between a value set in timer register and a value in the up-counter, the comparator outputs the match detection signal.

TBxRG0 and TBxRG1 are double buffered configuration that are paired with register buffers. The double buffering is disabled in the initial state.

Disabling or Enabling of double buffering is specified by TBxCR<TBWBF>. If <TBWBF> = 0, double buffering becomes disable; if <TBWBF> ="1", it becomes enable.

When double buffering is enabled, data is transferred from the register buffer to the timer register (TBxRG0/1) when up-counter is matched with TBxRG1.

When up-counter is stopped even if double buffering is enabled, the double buffering operates as a single buffer, and data can be written to the TBxRG0 and TBxRG1 directly.

10.4.4 Capture Control

This is a circuit that controls the timing of latching values from UC into the TBxCP0 and TBxCP1. The capture timing of up-counter is specified by TBxMOD<TBCPM[1:0]>.

Software can also capture the value of up-counter into capture registers. The value of up-counter are captured into the TBxCP0 in each time "0" is written to TBxMOD<TBCP>.

10.4.5 Capture Registers (TBxCP0, TBxCP1)

These registers capture the value of up-counter.

10.4.6 Up-Counter Capture Register (TBxUC)

If TBxUC register is read during the counter operation, the current value of up-counter will be captured and the value will be read. The value captured at the end is held while the counter is stopping.

10.4.7 Comparators (CP0, CP1)

These circuits compare up-counter and values set to TBxRG0/1 to detect a match. If a match is detected, INTTBx occurs.

10.4.8 Timer Flip-Flop (TBxFF0)

The timer flip-flop (TBxFF0) is reversed by a match signal from the comparator and a latch signal to the capture registers. Reversing is enabled or disabled by setting the TBxFFCR<TBC1T1, TBC0T1, TBC1T1, TBC1T0>.

The value of TBxFF0 becomes undefined after a reset. The flip-flop can be reversed by writing "00" to TBxFFCR<TBFF0C[1:0]>. It can be set to "1" by writing "01", and can be cleared to "0" by writing "10".

The value of TBxFF0 can be output to the timer output pin (TBxOUT). If the timer output is performed, program the corresponding port settings beforehand.

10.4.9 Capture Interrupt (INTTBxCAP0, INTTBxCAP1)

INTTBxCAP0 and INTTBxCAP1 can be generated at the timing of latching value from the up-counter into TBxCP0 and TBxCP1.

10.4.10 DMA Request

A DMA request is issued to the DMAC at the timing of match interrupts or capture interrupts generation. When DMA transfer is performed, set DMA request to be enabled by the corresponding bit of TBxDMA register

Note: Even if DMA request is enabled by TBxDMAREQ<TBDMAEN2>, if interrupt mask configuration has been set by TBxIM<TBIM1>, a DMA request does not occur.

10.5 Description of Operation for each mode

10.5.1 Interval Timer Mode

When interrupts is generated in constant intervals, set the interval time to the timer register (TBxRG1) to generate the INTTBx interrupt.

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	Enable TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops prescaler and counter.
Interrupt set-enable register	←	*	*	*	*	*	*	*	*	Permits INTTBx interrupt by setting the corresponding bit to "1".
TBxFFCR	←	X	X	0	0	0	0	1	1	Disable to TBÇòFF0 reverse trigger
TBxMOD	←	X	1	0	0	0	*	*	*	Changes to prescaler output clock as input clock. Specifies capture function to disable. (*** = 001 to 111)
TBxRG1	←	*	*	*	*	*	*	*	*	Specifies a time interval. (16 bits)
	←	*	*	*	*	*	*	*	*	
TBxRUN	←	X	X	X	X	X	1	X	1	Starts prescaler and counter.

Note:X; Don’t care, *; optional value, -; Don't change

10.5.2 Event Counter Mode

It is possible to make TMRBx the event counter by using a source clock as an external clock (TBxIN pin input).

The up-counter counts up on the rising edge of TBxIN pin input. The value of up-counter can be captured by software. It is possible to read the count value by reading capture values.

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	Enable TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops prescaler and counter.
Assigns the corresponding port to TBxIN.										
TBxFFCR	←	X	X	0	0	0	0	1	1	Disable to TBxFF0 reverse trigger.
TBxMOD	←	X	1	0	0	0	0	0	0	Set to a source clock as TBxIN pin input.
TBxRUN	←	X	X	X	X	X	1	X	1	Starts prescaler and counter.
TBxMOD	←	X	0	-	-	-	-	-	-	Software capture is done.

Note:X; Don’t care, *; optional value, -; Don't change

10.5.3 Programmable Pulse Generation (PPG) Output Mode

Square wave can be output in any frequency and duty. The output pulse can be either low-active or high-active.

TBxFF0 is reversed when the up-counter matches the set value of TBxRG0 and TBxRG1. TBxFF0 can be output from TBxOUT pin.

Note that the set value of TBxRG0 and TBxRG1 must be satisfy the following requirement.

Set value of TBxRG0 < Set value of TBxRG1

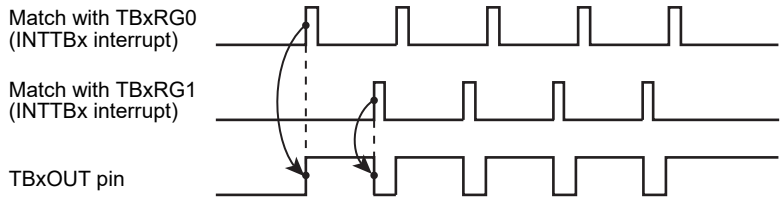


Figure 10-2 Example of programmable pulse generation output

In this mode, by enabling the double buffering, the value of register buffer 0 and 1 are shifted into TBxRG0 and TBxRG1 when UC matches the value of TBxRG1.

This makes possible to modify frequency and duty without a concern of a change timing of TBxRG0 and TBxRG1.

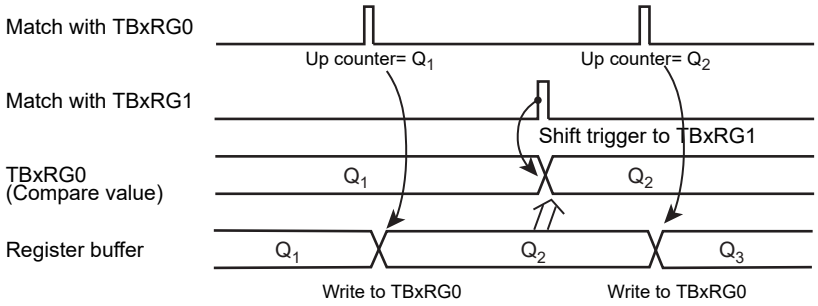


Figure 10-3 Register buffer operation

The block diagram of this mode is shown below.

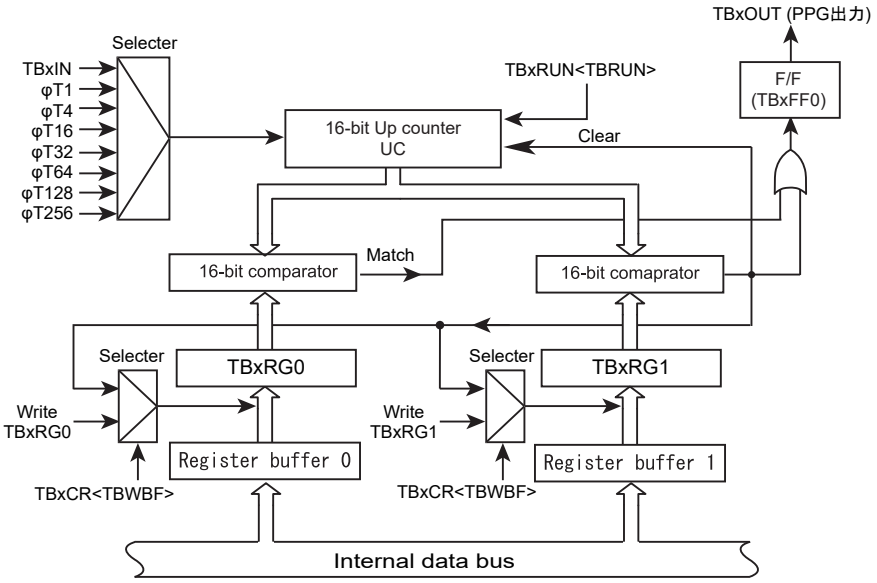


Figure 10-4 Block diagram of 16-bit PPG mode

Each register in the 16-bit PPG output mode should 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 prescaler and counter.
TBxCR	←	1	0	X	X	X	0	X	X	Enables double-buffering.
TBxRG0	←	*	*	*	*	*	*	*	*	set a duty.
	←	*	*	*	*	*	*	*	*	
TBxRG1	←	*	*	*	*	*	*	*	*	Set a cycle.
	←	*	*	*	*	*	*	*	*	
TBxFFCR	←	X	X	0	0	1	1	1	0	Sets TBxFF0 to invert the signal by detection of a match between TBxRG0 or TBxRG1 and the up-counter; sets TBxFF0 not to reverse the signal by capturing TBxCP0 or TBxCP1.Sets an initial value of TBxFF0 to "0".
TBxMOD	←	X	1	0	0	1	*	*	*	Designates the prescaler output clock as the input clock, and disable the capture function.
							(*** = 001 to 111)			
Assigns the corresponding port to TBxOUT.										
TBxRUN	←	X	X	X	X	X	1	X	1	Starts prescaler and counter.

Note: X; Don't care, *; optional value, -; Don't change

10.5.4 Programmable Pulse Generation (PPG) External Trigger Output Mode

A PPG wave with a short delay time can be output when TMRBx is started by the external trigger count start mode in the PPG (Programmable Pulse Generation) output mode.

The example of one-shot pulse output (with delay) using external trigger count start is shown below.

To start TMRB in the external trigger count start mode, set "1" to TBxCR<CSSEL> and set "0" to TBxCR<TRGSEL> to count up TBxIN on the rising edge while 16-bit counter has been stopped.

TBxRG0 is set the delay time (d) from an external trigger signal. TBxRG1 is set the value (d)+(p) of which the delay time (d) is added to the width (p) of one-shot pulse.

To reverse TBxFF0 when the up-counter matches TBxRG0/1, set "1" to TBxFFCR<TBE1T1>, <TBE0T1>.

To start the up-counter, set "1" to TBxRUN<TBPRUN>, <TBRUN>.

At this time, if the external trigger pulse is input to TBxIN, the up-counter is started on the rising edge of external trigger pulse.

TBxFF0 is reversed when the up-counter counts up to (d). It matches TBxRG0 and then TBxFF0 becomes "High" level.

TBxFF0 is reversed when the up-counter counts up to (d)+(p). It matches TBxRG1 and then TBxFF0 becomes "Low" level.

To avoid changing the level of TBxFF0 by INTTBx that occurs when the up-counter matches TBxRG1, clear TBxFFCR<TBE1T1>, <TBE0T1> to "0" or stop the up-counter by TBxRUN<TBPRUN>, <TBRUN>.

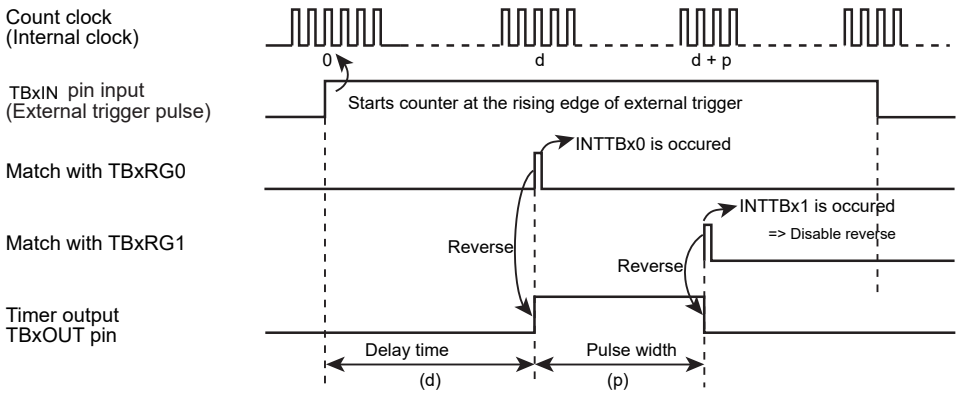


Figure 10-5 One-shot pulse output with delay using external trigger count start

The following shows the case that 2 ms width one-shot pulse is output by triggering TBxIN input on the rising edge after 3 ms has been elapsed. In this example, the source clock is φT1.

		7	6	5	4	3	2	1	0	
[Main processing]										
Assigns the corresponding port to TBxIN.										
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	0	Stops prescaler and counter.
TBxRG0	←	*	*	*	*	*	*	*	*	Set count value. (3ms/φT1)
TBxRG0	←	*	*	*	*	*	*	*	*	
TBxRG1	←	*	*	*	*	*	*	*	*	Set count value. (3+2)ms/φT1)
TBxRG1	←	*	*	*	*	*	*	*	*	
TBxFFCR	←	X	X	0	0	1	1	1	0	Sets TBxFF0 to invert the signal by detection of a match between TBxRG0 or TBxRG1 and the up-counter; sets TBxFF0 not to reverse by capturing TBxCP0 or TBxCP1. Sets an initial value of TBxFF0 to "0".
TBxMOD	←	X	1	0	0	1	0	0	1	Starts the up-counter as free-running. Selects φT1 for the source clock. Disable capture a value of the up-counter.
Assigns the corresponding port to TBxOUT.										
TBxIM	←	X	X	X	X	X	1	0	1	Masks except TBxRG1 interrupt.
Interrupt set-enable register	←	*	*	*	*	*	*	*	*	Sets "1" to the bit corresponding to INTTBx interrupt and permits to generate the interrupt.
TBxRUN	←	X	X	X	X	X	1	X	1	Starts prescaler and counter.
[Processing of INTTBx interrupt service routine] Output disable										
TBxFFCR	←	X	X	−	−	0	0	−	−	Clears TBxFF0 reverse trigger setting
TBxRUN	←	X	X	X	X	X	0	X	0	Stops prescaler and counter.

Note: X; Don't care, *; optional value, -; Don't change

10.6 Applications Using Capture Function

The capture function can be used in many applications.

The applications are shown below.

1. Frequency measurement
2. Pulse width measurement

10.6.1 Frequency Measurement

The following are examples of measuring a frequency of external clock.

In this section, TMRBm is used as 16-bit interval timer mode and TMRBn is used as 16-bit event counter mode.

To count the up-counter of TMRBn freely by an external clock, set TMnMOD<TBCLK> to "000" and set TBnRUN<TBE1T1><TBE0T1> to "11".

To reverse TBmFF0 when the up-counter of TMRBm matches TBmRG0 and TBmRG1, set TBmFFCR<TBE1T1> <TBE0T1> to "11".

To capture a value of the up-counter into TBnCP0 on the rising edge of TBmFF0 and to capture a value of the up-counter into TBnCP1 on the falling edge of TBmFF0, set TBnMOD<TBCPM[1:0]> to "11".

Set the number of counting external clocks to TBmRG0/1 and start TMRBm.

When a value the up-counter of TMRBm matches TBmRG0, TBmFF0 rises and a value of the up-counter of TMRBn is captured into TBnCP0. When the up-counter of TMRBm matches TBmRG1, TBmFF0 falls and a value of the up-counter of TMRBn is captured into TBnCP1.

A frequency can be measured from (TBnCP1 - TBnCP0) ÷ (TBmRG1 - TBmRG0) using INTTBm.

For example, the difference between TBmRG1 and TBmRG0 is 0.5 s and the difference between TBnCP1 and TBnCP0 is 100, the frequency is 200 Hz (100 ÷ 0.5 s = 200Hz)

Depending on the changing timing of TBmFF0, the result of TBnCP1 - TBnCP0 will be minus. Correct the value if TBnCP1 - TBnCP0 is minus.

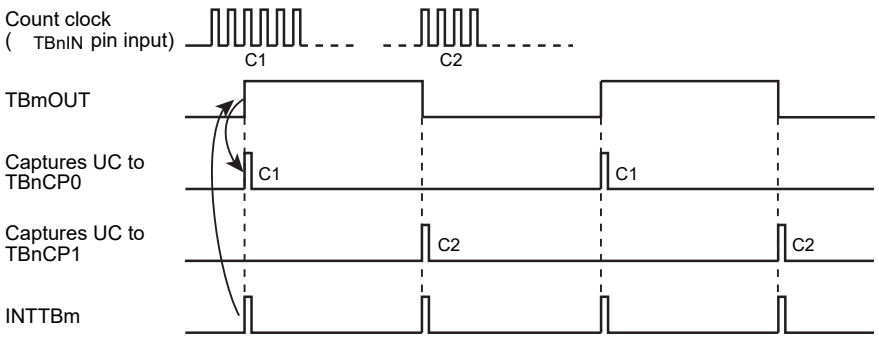


Figure 10-6 Frequency measurement

The following shows in the case that the measured pulse is input to TBnIN. In this example, the source clock is φT1.

7 6 5 4 3 2 1 0

[Main processing] Capture setting by TBmFF0

		7	6	5	4	3	2	1	0	
Assigns a corresponding port to TBnIN.										
TBmEN	←	1	X	X	X	X	X	X	X	Enables TMRBm operation.
TBmRUN	←	X	X	X	X	X	0	X	0	Stops prescaler and counter.
TBnEN	←	1	X	X	X	X	X	X	X	Enables TMRBn operation.
TBnRUN	←	X	X	X	X	X	0	X	0	Stops prescaler and counter.
TBmCR	←	1	0	X	X	X	0	X	X	Enables double-buffering.
TBmRG0	←	*	*	*	*	*	*	*	*	Sets the external clock measured time 1.
	←	*	*	*	*	*	*	*	*	
TBmRG1	←	*	*	*	*	*	*	*	*	Sets the external clock measured time 2.
	←	*	*	*	*	*	*	*	*	
TBmFFCR	←	X	X	0	0	1	1	1	0	Sets TBmFF0 to invert the signal by detection of a match between TBmRG0 or TBmRG1 and the up-counter. Sets TBmFF0 not to reverse the signal by capturing TBmCP0 or TBmCP1. Sets an initial value of TBmFF0 to "0".
TBnMOD	←	0	1	1	1	0	0	0	0	Captures on the rising / falling edge. Clears and disables the up-counter. Set input clock to TBnIN.
TBmIM	←	X	X	X	X	X	1	0	1	Masks except TBxRG1 interrupt.
Interrupt set-enable register	←	*	*	*	*	*	*	*	*	Sets "1" to the bit corresponding to INTTBm interrupt to enable the interrupt.
TBnRUN	←	X	X	X	X	X	1	X	1	Starts prescaler and counter.
TBmRUN	←	X	X	X	X	X	1	X	1	Starts prescaler and counter.
[Processing of INTTBm interrupt service routine]										
TBmFFCR	←	X	X	−	−	0	0	−	−	Clears TBxFF0 reverse trigger setting
Interrupt enable clear register	←	*	*	*	*	*	*	*	*	Sets "1" to the bit corresponding to INTTBm to disable the interrupt.
TBnCP0 and TBnCP1 are read out and the frequency is calculated.										

Note: X; Don't care, *; optional value, -; Don't change, m,n; Optional channel number

10.6.2 Pulse Width Measurement

"High" level width of the external pulse can be measured.

To capture a value of the up-counter into TBxCP0 on the rising edge of TBxIN and to capture a value of the up-counter into TBxCP1 on falling edge of TBxIN, set TBxMOD<TBCPM> to "10".

Enables INTTBxCAP1 interrupt.

Enables TMRBx operation.

If rising edge of external pulse is input to TBxIN, a value of the up-counter is captured into TBxCP0. If falling edge of external pulse is input to TBxIN, a value of the up-counter is captured into TBxCP1 and INTTBxCAP1 interrupt occurs.

The "High" level width of the external pulse can be calculated by multiplying the difference between TBxCP0 and TBxCP1 by the clock cycle of a prescaler output clock.

For example, if the difference between TBxCP0 and TBxCP1 is 100 and the cycle of the prescaler output clock is 0.5 μs, the pulse width is 100 × 0.5 μs = 50 μs.

If a pulse width is beyond the maximum count time of the up-counter, make a correction.

In addition, "Low" level width of an external pulse can be measured.

To calculate the "Low" level width, enable INTTBxCAP0 interrupt and multiply the time difference between first C2 and second C1 in "Figure 10-7 Pulse width measurement" at the second time of INTTBxCAP0 by the cycle of the prescaler output clock.

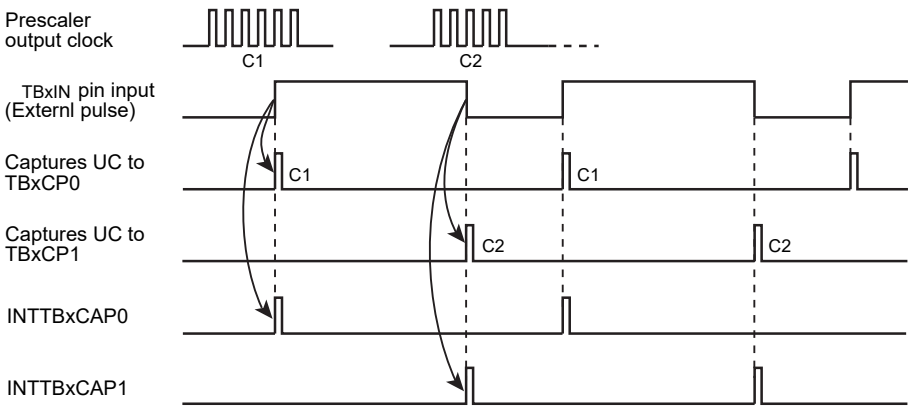


Figure 10-7 Pulse width measurement

The following describes the example of the measurement of "High" level width of the external pulse into TBxIN. In this example, the source clock is φT1.

		7	6	5	4	3	2	1	0	
[Main processing] Capture setting TBxIN.										
Assigns a corresponding port to TBxIN.										
TBxEN	←	1	X	X	X	X	X	X	X	Enables TMRBx operation
TBxRUN	←	X	X	X	X	X	0	X	0	Stops prescaler and counter.
TBxFFCR	←	X	X	0	0	0	0	1	0	Sets TBxFF0 not to invert the signal if TMRBx detects a match between TBxRG0 or TBxRG1 and the up-counter, or when capturing TBxCP0 or TBxCP1. Sets an initial value of TBxFF0 to "0".
TBxMOD	←	X	1	1	0	0	0	0	1	Starts the up-counter as free-running. Selects φT1 for the source clock. the up-counter is captured to TBxCP0 on the rising edge of TBxIN. the up-counter is captured to TBxCP1 on the falling edge of TBxIN.
Interrupt set-enable register	←	*	*	*	*	*	*	*	*	Sets "1" to the bit corresponding to INTTBxCAP1 interrupt to enable the interrupt.
TBxRUN	←	X	X	X	X	X	1	X	1	Starts prescaler and counter.
[Processing INTTBxCAP1 interrupt service routine] Calculate the width of "High" level.										
Interrupt enable clear register	←	*	*	*	*	*	*	*	*	Sets "1" to the bit corresponding to INTTBxCAP1 interrupt to disable the interrupt.
Calculated the width of "High" level by reading TBxRG0 and TBxRG1.										

Note: X; Don’t care, *; optional value, -; Don't change

11. 16-Bit Timer A (TMR16A Ver.B)

11.1 Outline

TMR16A contains the following functions:

- Match interrupt
- Square waveform output
- Read capture

In this chapter, "x" indicates a channel number.

11.2 Block Diagram

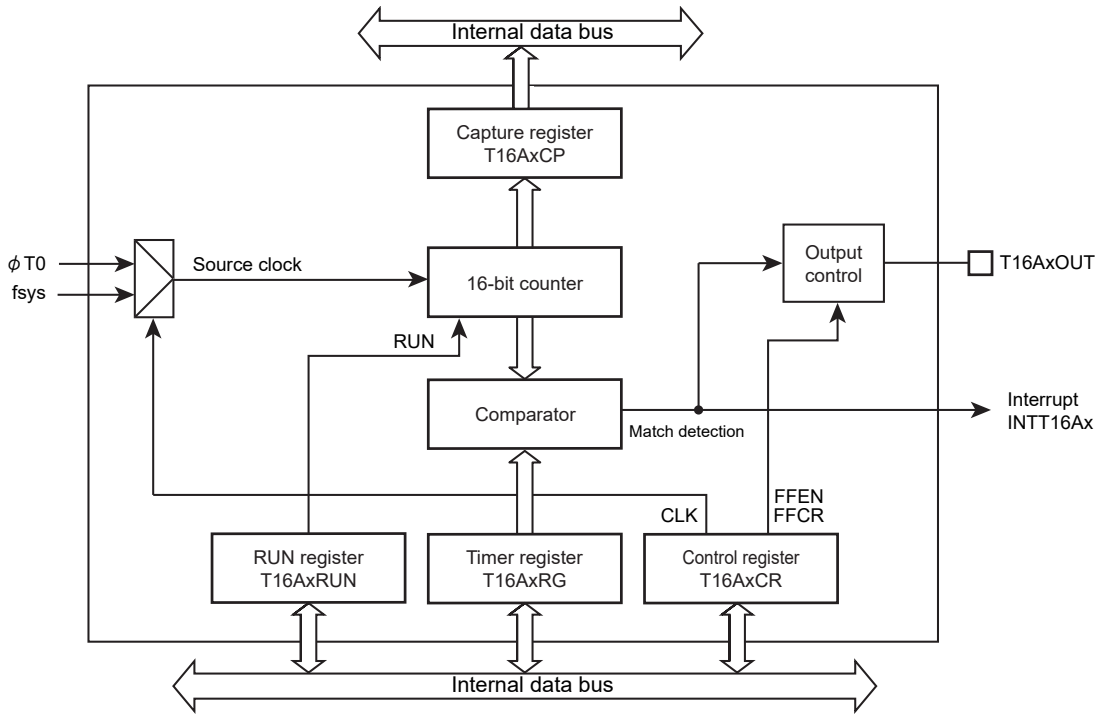


Figure 11-1 Block diagram of TMR16A

11.3 Registers

11.3.1 Register List

The table below shows control registers and their addresses.

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

Register name		Address(Base+)
Enable register	T16AxEN	0x0000
RUN register	T16AxRUN	0x0004
Control register	T16AxCR	0x0008
Timer register	T16AxRG	0x000C
Capture register	T16AxCP	0x0010

Note: When T16ARUN<RUN> is set to "1", do not modify T16AxEN, T16AxCR, T16AxRG and T16AxCP.

11.3.1.1 T16AxEN (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	-	-	-	-	-	-	HALT	I2T16A
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as "0".
1	HALT	R/W	Operation during halt mode debug 0: Operating 1: Stop Specifies the operation during halt mode debug. Write "1" to the bit to stop the operation.
0	I2T16A	R/W	Operation during the IDLE mode 0: Stop 1: Operating Specifies the operation during the IDLE mode. Write "1" to the bit to continue the operation.

11.3.1.2 T16AxRUN (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	-	-	-	-	-	-	-	RUN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	−	R	Read as "0".
0	RUN	R/W	Counter operation 0: Stop 1: Operating

11.3.1.3 T16AxCR (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	FFEN	-	FFCR		-	-	-	CLK
After reset	0	0	1	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15	−	R/W	Write to "0"
14-8	−	R	Read as "0".
7	FFEN	R/W	Inverse of T16AxOUT 0: Disabled 1: Enabled Write "1" to the bit to invert T16AxOUT when the counter matches with T16ARG.
6	−	R	Read as "0".
5-4	FFCR[1:0]	W	T16AxOUT control 00: Invert 01: Set 10: Clear 11: No operation Write a value to the bit to control T16AxOUT by software. Read as "11".
3-1	−	R	Read as "0".
0	CLK	R/W	Source clock 0: fsys 1: ΦT0 Specifies a source clock.

11.3.1.4 T16AxRG (Timer Register)

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

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-0	RG[15:0]	R/W	Set a value to compare with a counter

Note: Do not set "0x0000".

11.3.1.5 T16AxCP (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	CP[15:8]							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	CP[7:0]							
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as "0".
15-0	CP[15:0]	R	Counter value [Read] Reads a current counter value.

11.4 Operation Description

11.4.1 Timer Operation

1. Preparation

   Select a source clock with T16AxCR<CLK>. Write "0" to set fsys or write "1" to set ΦT0. Set a counter value to T16AxRG<RG[15:0]>.

2. Counter operation

   Before starting counter operation, set "0x0000" to T16AxCP<CP> to clear the counter.

   To start count-up, set "1" to T16AxRUN<RUN>. If the counter value matches with a value of T16AxRG<RG[15:0]>, it will be cleared to "0x0000" and continued to count-up.

3. Match detection interrupt generation

   If a counter value matches with a value of T16AxRG<RG[15:0]>, a match detection interrupt INTT16Ax will be output.

4. Stop

   To stop counts, set "0" to T16AxRUN<RUN>. The counter value is held. Then clear the counter before counting is started by setting "1" to <RUN>.

Note: Modification of T16AxCR, T16AxRG and T16AxCP must be performed while the counter is stopping (T16AxRUN<RUN> is set to "0").

11.4.2 T16AxOUT Control

T16AxOUT is modified by register setting or by matching the counter with T16AxRG.

An initial state of T16AxOUT is "0".

1. Control by software

   With T16AxCR<FFCR[1:0]> setting, T16AxOUT can be specified; "1" is to set, "0" is to clear, and also the inverted setting is possible.

   Modify T16AxCR while the counter stops (T16AxRUN<RUN> is "0").

2. Inverse due to matching counter

   Write "1" to T16ACR<FFEN> to invert T16AxOUT. When T16AxRG<RG[15:0]> matches with a counter value, T16AxOUT will invert. When the counter stops, a state of T16AxOUT will remain.

11.4.3 Read Capture

A current value of the counter can be captured by reading T16AxCP<[15:0]>.

11.4.4 Automatic Stop

With the setting of T16AxEN<I2T16A> or <HALD>, TMR16A automatically stops in the following conditions:

1. Transition to/from IDLE mode

   With T16AxEN<I2T16A> setting, TMR16Ax operation during IDLE mode can be specified. If "1" is set, TMR16A automatically stops count-up when the transition to the IDLE mode occurs. If TMR16Ax returns from IDLE mode, it will restart counting-up operation.

2. Debug halt

   With T16AxEN<HALT> setting, TMR16Ax operation during debug halt can be specified. If "0" is set, TMR16Ax automatically stops count-up when the transition to the debug halt mode occurs. If debug halt mode of the core is cancelled, count-up will restart.

12. Serial Channel with 4bytes FIFO (SIO/UART)

12.1 Overview

Serial channel (SIO/UART) has the modes shown below.

- Synchronous communication mode (I/O interface mode)
- Asynchronous communication mode (UART mode)

Their features are given in the following.

- Transfer Clock
  - Dividing by the prescaler, from the peripheral clock (φT0) frequency into 1/1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, 1/128.
  - Make it possible to divide from the prescaler output clock frequency into 1 to 16.
  - Make it possible to divide from the prescaler output clock frequency into N+m/16 (N=2 to 15, m=1 to 15). (only UART mode)
  - The usable system clock (fsys) (only UART mode).
- Buffer
  - The usable double buffer function.
  - Make it possible to clear the transmit buffer.
- FIFO

  The usable 4 byte FIFO including transmit and receive.
- I/O Interface Mode
  - Transfer Mode: the half duplex (transmit/receive), the full duplex
  - Clock: Output (fixed rising edge) /Input (selectable either rising or falling edge)
  - Make it possible to specify the interval time of continuous transmission.
  - The state of SCxTXD pin after output of the last bit can be selected as follow:

    Keep a "High" level, "Low" level or the state of the last bit
  - The state of SCxTXD pin when an under run error is occurred in clock input mode can be selected as follow:

    Keep a "High" level or "Low" level
  - The last bit hold time of SCxTXD pin can be specified in clock input mode.
- UART Mode
  - Data length: 7 bits, 8bits, 9bits
  - Add parity bit (to be against 9bits data length)
  - Serial links to use wake-up function
  - Handshaking function with $\overline{\text{SCxCTS}}$ pin
  - Noise cancel for SCxRXD pin

In the following explanation, "x" represents channel number.

12.2 Configuration

Serial channel block diagram and serial clock generator circuit diagram are shown in bellows.

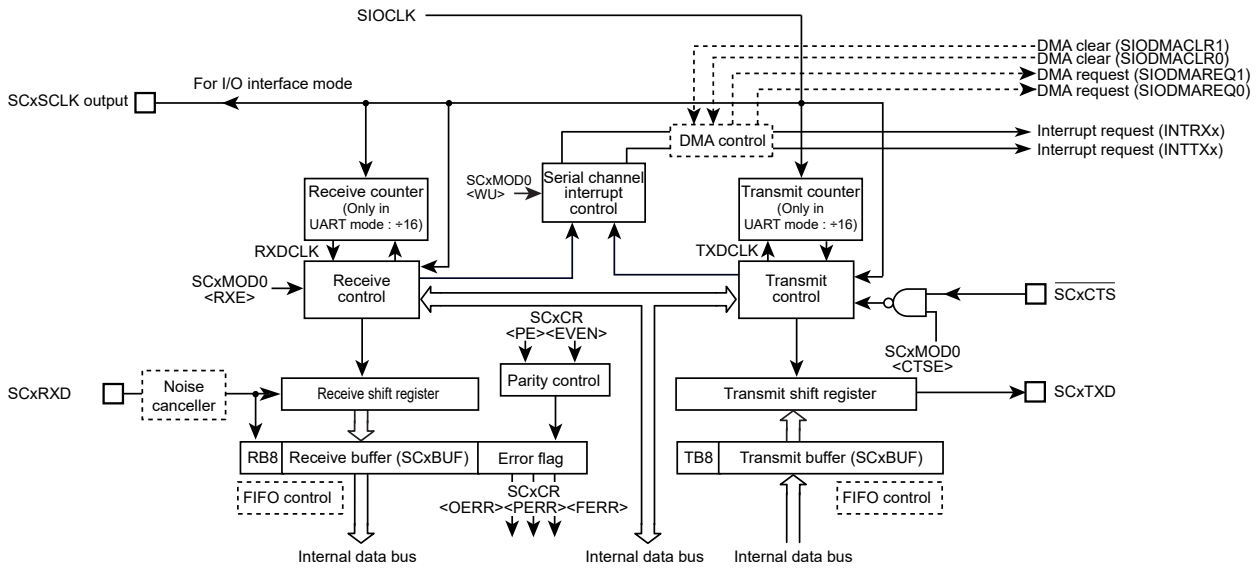


Figure 12-1 Serial Channel Block Diagram

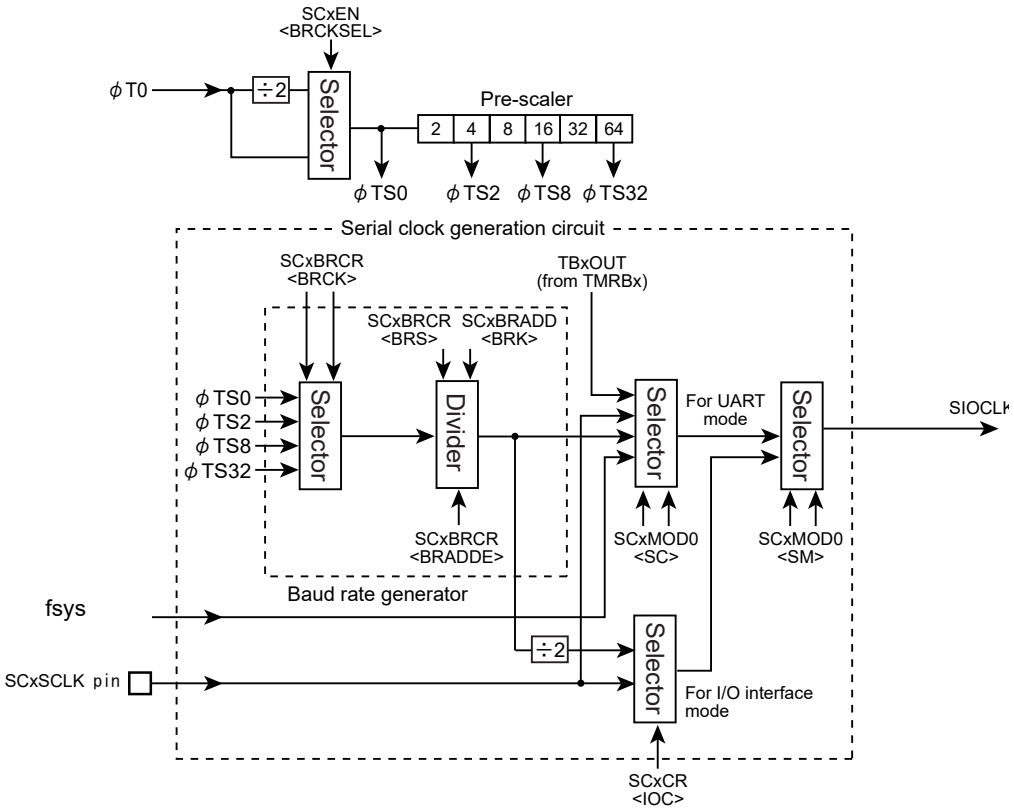


Figure 12-2 Serial clock generation circuit block diagram

12.3 Registers Description

12.3.1 Registers List

The table below shows control registers and their addresses.

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

Register name		Address (Base+)
Enable register	SCxEN	0x0000
Buffer register	SCxBUF	0x0004
Control register	SCxCR	0x0008
Mode control register 0	SCxMOD0	0x000C
Baud rate generator control register	SCxBRCR	0x0010
Baud rate generator control register 2	SCxBRADD	0x0014
Mode control register 1	SCxMOD1	0x0018
Mode control register 2	SCxMOD2	0x001C
Receive FIFO configuration register	SCxRFC	0x0020
Transmit FIFO configuration register	SCxTFC	0x0024
Receive FIFO status register	SCxRST	0x0028
Transmit FIFO status register	SCxTST	0x002C
FIFO configuration register	SCxFCNF	0x0030
DMA request enable register	SCxDMA	0x0034

Note: Do not modify any control register when data is being transmitted or received.

12.3.2 SCxEN (Enable Register)

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

Bit	Bit Symbol	Type	Function
31-2	−	R	Read as "0".
1	BRCKSEL	R/W	Selects input clock for prescaler. 0: φT0/2 1: φT0
0	SIOE	R/W	Serial channel operation 0: Disabled 1: Enabled Specified the Serial channel operation. To use the Serial channel, set <SIOE> = "1". When the operation is disabled, no clock is supplied to the other registers in the Serial channel module. This can reduce the power consumption. If the Serial channel operation is executed and then disabled, the settings will be maintained in each register.

12.3.3 SCxBUF (Buffer Register)

SCxBUF works as a transmit buffer or FIFO for write operation and as a receive buffer or FIFO for read operation.

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7-0	TB[7:0] / RB [7:0]	R/W	[write] TB: Transmit buffer or FIFO [read] RB: Receive buffer or FIFO

12.3.4 SCxCR (Control Register)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	EHOLD			-	TXDEMP	TIDLE	
After reset	0	0	0	0	0	1	1	0
	7	6	5	4	3	2	1	0
bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-15	−	R	Read as "0".
14-12	EHOLD[2:0]	R/W	The last bit hold time of a SCxTXD pin in clock input mode (For only I/O interface mode) Set the last bit hold time and SCLK cycle to keep the last bit hold time equal or less than SCLK cycle/2. 000: 2/fsys 100: 32/fsys 001: 4/fsys 101: 64/fsys 010: 8/fsys 110: 128/fsys 011: 16/fsys 111: Reserved
11	-	R	Read as "0".
10	TXDEMP	R/W	The state of SCxTXD pin when an under run error is occurred in clock input mode. (For only I/O interface mode) 0: "Low" level output 1: "High" level output
9-8	TIDLE[1:0]	R/W	The state of SCxTXD pin after output of the last bit (For only I/O interface mode) When <TIDLE[1:0]> is set to "10", set "000" to <EHOLD[2:0]>. 00: Keep a "Low" level output 01 :Keep a "High" level output 10: Keep a last bit 11: Reserved
7	RB8	R	Receive data bit 8 (For only UART mode) 9th bit of the received data in the 9-bit UART mode.
6	EVEN	R/W	Parity (For only UART mode) Selects even or odd parity. The parity bit may be used only in the 7- or 8-bit UART mode. 0: Odd 1: Even Selects even or odd parity.
5	PE	R/W	Add parity (For only UART mode) Controls disabled or enabled parity. The parity bit may be used only in the 7- or 8-bit UART mode. 0: Disabled 1: Enabled
4	OERR	R	Over-run error flag (Note) 0: Normal operation 1: Error
3	PERR	R	Parity / Under-run error flag (Note) 0: Normal operation 1: Error
2	FERR	R	Framing error flag (Note) 0: Normal operation 1: Error

Bit	Bit Symbol	Type	Function
1	SCLKS	R/W	Selecting input clock edge (For I/O Interface mode) Set to "0" in the clock output mode. 0: Data in the transmit buffer is sent to SCxTXD pin every one bit on the falling edge of SCxRXD pin. Data from SCxRXD pin is received in the receive buffer every one bit on the rising edge of SCxRXD pin. In this case, the state of a SCxRXD pin starts from "High" level. (Rising edge mode) 1: Data in the transmit buffer is sent to SCxTXD pin every one bit on the rising edge of SCxSCLK pin. Data from SCxRXD pin is received in the receive buffer every one bit on the falling edge of SCxSCLK pin. In this case, the state of a SCxSCLK starts from "Low" level.
0	IOC	R/W	Selecting clock (For I/O Interface mode) 0: Clock output mode (A transfer clock is output from SCxSCLK pin.) 1: Clock input mode (A transfer clock is input to SCxSCLK pin.)

Note:<OERR>, <PERR> and <FERR> are cleared to "0" when read.

12.3.5 SCxMOD0 (Mode Control Register 0)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	TB8	CTSE	RXE	WU	SM		SC	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TB8	R/W	Transmit data bit 8 (For only UART mode) Writes the 9th bit of transmit data in the 9-bit UART mode.
6	CTSE	R/W	Handshake function control (For only UART mode) 0: CTS disabled 1: CTS enabled Controls handshake function. Setting "1" enables handshake function using $\overline{\text{SCxCTS}}$ pin.
5	RXE	R/W	Receive control (Note1)(Note2) 0: Disabled 1: Enabled
4	WU	R/W	Wake-up function (For only UART mode) 0: Disabled 1: Enabled This function is available only at 9-bit UART mode. In other mode, this function has no meaning. When it is enabled, interrupt is occurred only when RB9 = "1" in a 9-bit UART mode.
3-2	SM[1:0]	R/W	Specifies transfer mode. 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode
1-0	SC[1:0]	R/W	Serial transfer clock (For only UART mode) 00: TMRB output 01: Baud rate generator 10: System clock (fsys) 11: External clock (SCxSCLK pin input) (For the I/O interface mode, the transfer clock in I/O interface mode is selected by SCxCR<IOC>.)

Note 1: Specify the all mode control registers first and then the <RXE>.

Note 2: Do not stop the receive operation (by setting SCxMOD0<RXE> to "0") when data is being received.

12.3.6 SCxMOD1 (Mode Control Register 1)

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	I2SC	FDPX		TXE	SINT			-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	I2SC	R/W	IDLE 0: Stop 1: Operate Specifies operation in the IDLE mode.
6-5	FDPX[1:0]	R/W	Transfer mode setting 00: Transfer prohibited 01: Half duplex (Receive) 10: Half duplex (Transmit) 11: Full duplex Configures the transfer mode in the I/O interface mode. And when FIFO is enabled, specify the configuration of FIFO. In UART mode, specify the only configuration of FIFO.
4	TXE	R/W	Transmit control (Note1)(Note2) 0 :Disabled 1: Enabled This bit enables transmission and is valid for all the transfer modes.
3-1	SINT[2:0]	R/W	Interval time of continuous transmission (For I/O interface mode) 000: None 001: 1 x SCLK cycle 010: 2 x SCLK cycle 011: 4 x SCLK cycle 100: 8 x SCLK cycle 101: 16 x SCLK cycle 110: 32 x SCLK cycle 111: 64 x SCLK cycle This parameter is valid only for the I/O interface mode when SCLK output mode is selected. In other modes, this parameter has no meaning. Specifies the interval time of continuous transmission when double buffering or FIFO is enabled in the I/O interface mode.
0	−	R/W	Write a "0".

Note 1: Specify the all mode control registers first and then enable the <TXE>.

Note 2: Do not stop the transmit operation (by setting <TXE> to "0") when data is being transmitted.

12.3.7 SCxMOD2 (Mode Control Register 2)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TBEMP	R	Transmit buffer empty flag 0: Full 1: Empty If double buffering is disabled, this flag is insignificant. This flag shows that the transmit double buffers are empty. When data in the transmit double buffers is moved to the transmit shift register and the double buffers are empty, this bit is set to "1". Writing data again to the double buffers sets this bit to "0".
6	RBFLL	R	Receive buffer full flag 0: Empty 1: Full If double buffering is disabled, this flag is insignificant. This is a flag to show that the receive double buffers are full. When a receive operation is completed and received data is moved from the receive shift register to the receive double buffers, this bit changes to "1" .When reading the receive buffer ,this bit is cleared to "0".
5	TXRUN	R	In transmission flag 0: Stop 1: Operate This is a status flag to show that data transmission is in progress. <TXRUN> and <TBEMP> bits indicate the following status.
4	SBLEN	R/W	STOP bit length (for UART mode) 0: 1-bit 1: 2-bit This specifies the length of transmission stop bit in the UART mode. On the receive side, the decision is made using only a single bit regardless of the <SBLEN>.
3	DRCHG	R/W	Setting transfer direction 0: LSB first 1: MSB first Specifies the direction of data transfer. In the UART mode, set this bit to LSB first.
2	WBUF	R/W	Enable double-buffer 0: Disabled 1: Enabled This parameter enables or disables the transmit/receive double buffers to transmit (in both SCLK output/input modes) and receive (in SCLK output mode) data i n the I/O interface mode and to transmit in the UART mode. When receiving data in the I/O interface mode (in clock input mode) and UART mode, double buffering is enabled regardless of the <WBUF>.

<TXRUN>	<TBEMP>	Status
1	−	Transmission in progress
0	1	Transmission is 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>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 a software reset is executed, the following bits are initialized and the transmit/receive circuit and FIFO become initial state (Note1)(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>
</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.

12.3.8 SCxBRCR (Baud Rate Generator Control Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	−	R/W	Write "0".
6	BRADDE	R/W	N + (16 − K)/16 divider function (Only for UART mode) 0: disabled 1: enabled
5-4	BRCK[1:0]	R/W	Select input clock to the baud rate generator. 00:φTS0 01:φTS2 10:φTS8 11:φTS32
3-0	BRS[3:0]	R/W	Division ratio "N" 0000: N = 16 0001: N = 1 0010: N = 2 ... 1111: N = 15

Note 1: As a division ratio, 1 ("0001") or 16 ("0000") can not be applied to N when using the "N + (16 - K)/16" division function in the UART mode.

Note 2: The division ratio "1" of the baud rate generator can be specified only when the double buffering is used in the I/O interface mode.

12.3.9 SCxBRADD (Baud Rate Generator Control Register 2)

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

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-0	BRK[3:0]	R/W	Specify K for the "N + (16 − K)/16" division (For UART mode) 0000: Prohibited 0001: K = 1 0010: K = 2 ... 1111: K = 15

Table 12-1 lists the settings of baud rate generator division ratio.

Table 12-1 Setting division ratio

	<BRADDE> = "0"	<BRADDE> = "1" (Note1) (Only in the UART mode)
<BRS>	Specify "N"	
<BRK>	No setting required	Specify "K" (Note2)
Division ratio	Divide by N	$N + \frac{(16 - K)}{16}$ division.

Note 1: To use the "N + (16 - K)/16" division function, be sure to set <BRADDE> to "1" after setting the K value to <BRK>. The "N + (16 - K)/16" division function can only be used in the UART mode.

Note 2: Specifying "K = 0" is prohibited.

12.3.10 SCxFCNF (FIFO Configuration Register)

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

Bit	Bit Symbol	Type	Function
31-8	–	R	Read as "0".
7-5	-	R/W	Be sure to write "000".
4	RFST	R/W	Bytes used in receive FIFO. 0: Maximum 1: Same as FILL level of receive FIFO The number of receive FIFO bytes to be used is selected. (Note1) 0: The maximum number of bytes of the FIFO configured (see also <CNFG>). 1: Same as the fill level for receive interrupt generation specified by SC0RFC <RIL[1:0]>.
3	TFIE	R/W	Specify transmit interrupt for transmit FIFO. 0: Disabled 1: Enabled When transmit FIFO is enabled, transmit interrupts are enabled or disabled by this parameter.
2	RFIE	R/W	Specify receive interrupt for receive FIFO. 0: Disabled 1: Enabled When receive FIFO is enabled, receive interrupts are enabled or disabled by this parameter.
1	RXTXCNT	R/W	Automatic disable of RXE/TXE. 0: None 1: Auto disable Controls automatic disabling of transmission and reception. Setting "1" enables to operate as follows. Half duplex Receive: When the receive shift register, receive fuffers and receive FIFO are filled up to the specified number of valid bytes, SCxMOD0<RXE> is automatically set to "0" to inhibit further reception. Half duplex Transmit: When the transimit shift register, transmit buffers and the transmit FIFO are empty, SCxMOD1<TXE> is automatically set to "0" to inhibit further transmission. Full duplex: When either of the above two conditions is satisfied, <TXE> and <RXE> are automatically set to "0" to inhibit further transmission and reception.
0	CNFG	R/W	FIFO enable. 0: Disabled 1: Enabled Enables FIFO.(Note2) When <CNFG> is set to "1", FIFO is enabled. If FIFO is enabled, the SCOMOD1 <FDPX[1:0]> setting automatically configures FIFO as follows: Half duplex Receive: Receive FIFO 4bytes Half duplex Transmit: Transmit FIFO 4bytes Full duplex: Receive FIFO 2bytes and Transmit FIFO 2bytes

Note 1: Regarding Transmit FIFO, the maximum number of bytes being configured is always available. (See also <CNFG>.)

Note 2: The FIFO can not be used in 9 bit UART mode.

12.3.11 SCxRFC (Receive FIFO Configuration Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	RFCS	W	Receive FIFO clear (Note1) 1: Clear When SCxRFC<RFCS> is set to "1", the receive FIFO is cleared and SCxRST<RLVL[2:0]> is "000". And also the read pointer is initialized. Read as "0".
6	RFIS	R/W	Select interrupt generation condition. 0: When FIFO fill level (SCxRST<RLVL[2:0]>) = Receive FIFO fill level to generate receive interrupt (<RIL[1:0]>) 1: When FIFO fill level (SCxRST<RLVL[2:0]>) ≥ Receive FIFO fill level to generate receive interrupt (<RIL[1:0]>) For the detail of interrupt condition, refer to "12.13.1.2 FIFO"
5-2	-	R	Read as "0".
1-0	RIL[1:0]	R/W	FIFO fill level to generate receive interrupts.

	Half duplex	Full duplex
00	4 bytes	2 bytes
01	1 byte	1 byte
10	2 bytes	2 bytes
11	3 bytes	1 byte

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

12.3.12 SCxTFC (Transmit FIFO Configuration Register)

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

Bit	Bit Symbol	Type	Function
31-9	−	R	Read as "0".
8	TBCLR	W	Transmit buffer clear 0: Don't care 1: Clear When SCxTFC<TBCLR> is set to "1", the transmit buffer is cleared. Read as "0".
7	TFCS	W	Transmit FIFO clear (Note1) 0: Don't care 1: Clear When SCxTFC<TFCS> is set to "1", the transmit FIFO is cleared and SCxTST<TLVL[2:0]> is "000". And also the write pointer is initialized. Read as "0".
6	TFIS	R/W	Selects interrupt generation condition. 0: When FIFO fill level (SCxTST<TLVL[2:0]>) = Transmit FIFO fill level to generate transmit interrupt (<TIL[1:0]>) 1: When FIFO fill level (SCxTST<TLVL[2:0]>) ≤ Transmit FIFO fill level to generate transmit interrupt (<TIL[1:0]>) For the detail of interrupt condition, refer to "12.13.2.2 FIFO"
5-2	−	R	Read as "0".
1-0	TIL[1:0]	R/W	Fill level which transmit interrupt is occurred.

	Half duplex	Full duplex
00	Empty	Empty
01	1 byte	1 byte
10	2 bytes	Empty
11	3 bytes	1 byte

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

Note 2: In case that SCxEN<SIOE>="0" (Stop SIO/UART operation) or the operation mode is changed to IDLE mode with SCxMOD<I2SC>="0" (Stop SIO/UART operation in IDLE mode), SCxTFC is initialized again.After you perform the following operations, configure the SCxTFC register again.

12.3.13 SCxRST (Receive FIFO Status Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	ROR	R	Receive FIFO Overrun. (Note) 0: Not generated 1: Generated
6-3	−	R	Read as "0".
2-0	RLVL[2:0]	R	Status of Receive FIFO fill level. 000: Empty 001: 1 byte 010: 2 bytes 011: 3 bytes 100: 4 bytes

Note: **<ROR> is cleared to "0" when receive data is read from the SCxBUF.**

12.3.14 SCxTST (Transmit FIFO Status Register)

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

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as "0".
7	TUR	R	Transmit FIFO Under run. (Note) 0: Not generated 1: Generated
6-3	−	R	Read as "0".
2-0	TLVL[2:0]	R	Status of Transmit FIFO level 000: Empty 001: 1 byte 010: 2 byte 011: 3 byte 100: 4 byte

Note: **<TUR> is cleared to "0" when transmit data is written to the SCxBUF.**

12.3.15 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 transmit interrupt INTTX. 0: disable 1: enable

Note: When DMA request is generated during DMA transfer is being, it is not kept and nesting.

12.4 Operation in Each Mode

Table 12-2 shows the modes.

Table 12-2 Modes

Mode	type	Data length	Transfer direction	Specifies whether to use parity bits.	STOP bit length (transmit)
Mode 0	Synchronous communication mode (I/O interface mode)	8 bits	LSB first/MSB first	-	-
Mode 1	Asynchronous communication mode (UART mode)	7 bits	LSB first	o	1 bit or 2 bits
Mode 2		8 bits		o	
Mode 3		9 bits		×	

The Mode 0 is a synchronous communication and can be used to extend I/O. This mode transmits and receives data in synchronization with SCLK clock. SCLK clock can be used for both input and output modes. The direction of data transfer can be selected from LSB first or MSB first. This mode is not allowed either to use parity bits or STOP bits.

The mode 1, mode 2 and mode 3 are asynchronous modes and the transfer directions can be selected as only the LSB first.

Parity bits can be added in the mode 1 and mode 2. The mode 3 has a wakeup function in which the master controller can start up slave controllers via the serial link (multi-controller system). STOP bit in transmission can be selected from 1 bit and 2 bits. The STOP bit length in reception is fixed to a one bit.

12.5 Data Format

12.5.1 Data Format List

Figure 12-3 shows data format.

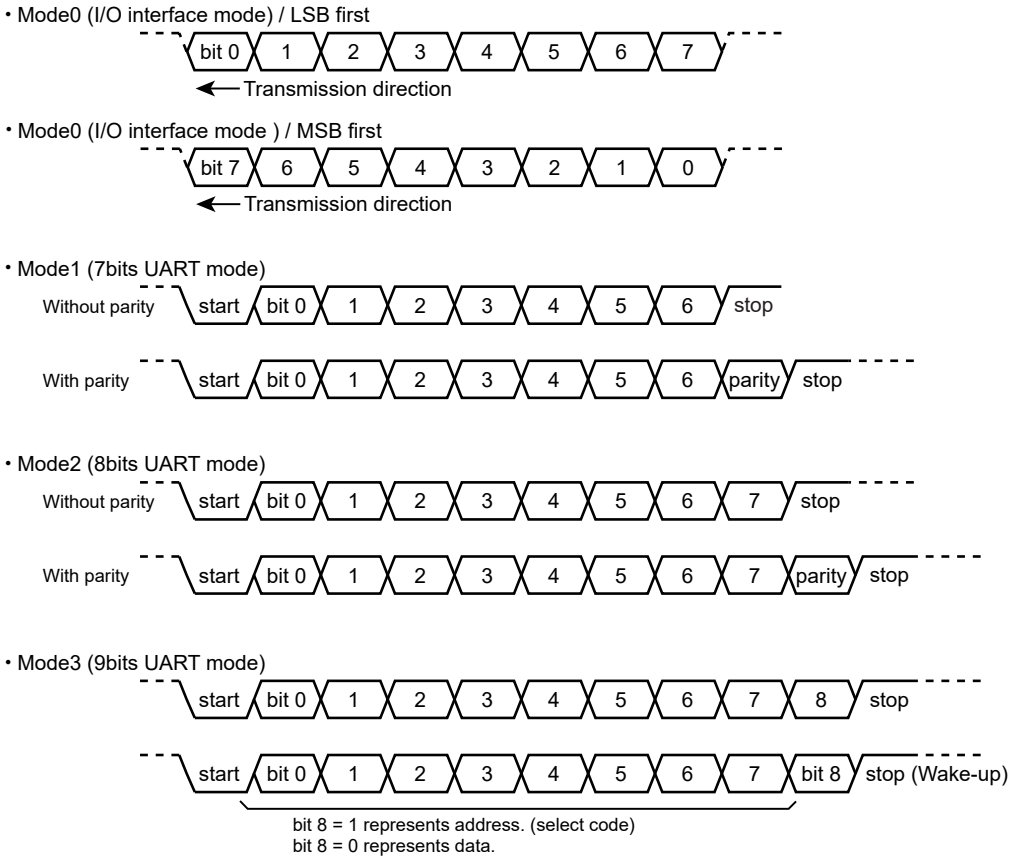


Figure 12-3 Data Format

12.5.2 Parity Control

The parity bit can be added with a transmitted data only in the 7- or 8-bit UART mode. And the received parity bit can be compared with a generated one.

Setting "1" to SCxCR<PE> enables the parity. SCxCR<EVEN> selects either even or odd parity.

12.5.2.1 Transmission

Upon data transmission, the parity control circuit automatically generates the parity with the data in the transmit buffer. The parity bit will be stored in SCxBUF<TB7> in the 7-bit UART mode and SCxMOD<TB8> in the 8-bit UART mode.

The <PE> and <EVEN> settings must be completed before data is written to the transmit buffer.

12.5.2.2 Receiption

If the received data is moved from the receive shift register to the receive buffer, a parity is generated.

In the 7-bit UART mode, the generated parity is compared with the parity stored in SCxBUF<RB7>, in the 8-bit UART mode, it is compared with the one in SCxCR<RB8>.

If there is any difference, a parity error occurs and the SCxCR<PERR> is set to "1".

In use of the FIFO, <PERR> indicates that a parity error was generated in one of the received data.

12.5.3 STOP Bit Length

The length of the STOP bit in the UART transmission mode can be selected from one bit or two bits by setting the SCxMOD2<SBLEN>. The length of the STOP bit data is determined as one-bit when it is received regardless of the setting of this bit.

12.6 Clock Control

12.6.1 Prescaler

There is a 7-bit prescaler to divide a prescaler input clock φT0 by 1, 2, 4, 8, 16, 32, 64 and 128.

Use the CGSYSCR and SCxEN<BRCKSEL> in the clock/mode control block to select the input clock of the prescaler.

The prescaler becomes active only when the baud rate generator is selected as a transfer clock by SCxMOD0<SC[1:0]> = "01".

12.6.2 Serial Clock Generation Circuit

The serial clock generation circuit is a block to generate transmit and receive clocks (SIOCLK) and consists of the circuits in which clocks can be selected by the settings of the baud rates generator and modes.

12.6.2.1 Baud Rate Generator

The baud rate generator generates transmit and receive clocks to determine the serial channel transfer rate.

(1) Baud Rate Generator input clock

The input clock of the baud rate generator is selected from the prescaler outputs divided by 1, 4, 16 and 64.

This input clock is selected by setting the SCxEN<BRCKSEL> and SCxBRCR<BRCK>.

SCxEN<BRCKSEL>	SCxBRCR<BRCK>	Baud rate generator input clock φTx
0	00	φT0/2
0	01	φT0/8
0	10	φT0/32
0	11	φT0/128
1	00	φT0
1	01	φT0/4
1	10	φT0/16
1	11	φT0/64

(2) Baud Rate Generator output clock

The frequency division ratio of the output clock in the baud rate generator is set by SCxBRCR and SCxBRADD.

The following frequency divide ratios can be used; 1/N frequency division in the I/O interface mode, either 1/N or 1/(N + (16-K)/16) in the UART mode.

The table below shows the frequency division ratio which can be selected.

Mode	Divide Function Setting SCxBRCR<BRADDE>	Divide by N SCxBRCR<BRS[3:0]>	Divide by K SCxBRADD<BRK[3:0]>
I/O interface	Divide by N	1 to 16 (Note)	-
UART	Divide by N	1 to 16	-
	N + (16-K)/16 division	2 to 15	1 to 15

Note: 1/N (N=1) frequency division ratio can be used only when a double buffer is enabled.

The input clock to the divider of baud rate generator is φTx, the baud rate generator output clock in the case of 1/N and N + (16-K)/16 is shown below.

- Divide by N

$$\text{Baud rate generator output clock} = \frac{\phi Tx}{N}$$

- N + (16-K)/16 division

$$\text{Baud rate generator output clock} = \frac{\phi Tx}{N + \frac{(16 - K)}{16}}$$

12.6.2.2 Clock Selection Circuit

A clock can be selected by setting the modes and the register.

Modes can be specified by setting the SCxMOD0<SM[1:0]>

The clock in I/O interface mode is selected by setting SCxCR<IOC><SCLKS>.

The clock in UART mode is selected by setting SCxMOD0<SC[1:0]>.

(1) Transfer Clock in I/O interface mode

Table 12-3 shows clock selection in I/O interface mode.

Table 12-3 Clock Selection in I/O Interface Mode

Mode SCxMOD0<SM[1:0]>	Input/Output selection SCxCR<IOC>	Clock edge selection SCxCR<SCLKS>	Clock of use
"00" (I/O interface mode)	"0" (Clock output mode)	"0" (Transmit : falling edge, Receive : rising edge)	Divided by 2 of the baud rate generator output.
	"1" (Clock input mode)	"0" (Transmit : falling edge, Receive : rising edge)	SCxSCLK pin input
		"1" (Transmit : rising edge, Receive : falling edge)	SCxSCLK pin input

To use SCxSCLK input, the following conditions must be satisfied.

- If double buffer is used
  - SCLK cycle > 6/fsys
- If double buffer is not used
  - SCLK cycle > 8/fsys

(2) Transfer clock in the UART mode

Table 12-4 shows the clock selection in the UART mode. In the UART mode, selected clock is divided by 16 in the receive counter or the transmit counter before use.

Table 12-4 Clock Selection in UART Mode

Mode SCxMOD0<SM[1:0]>	Clock selection SCxMOD0<SC[1:0]>
UART Mode ("01", "10", "11")	"00" : TMRB output
	"01" : Baud rate generator
	"10" : fsys
	"11" : SCxSCLK pin input

To use SCxSCLK pin input, the following conditions must be satisfied.

- SCLK cycle > 2/fsys

To enable the timer output, a timer flip-flop output inverts when the value of the counter and that of TBxRG1 match. The SIOCLK clock frequency is "Setting value of TBxRG1 × 2".

Baud rates can be obtained by using the following formula.

Baud rate calculation

$$\text{Transfer rate} = \frac{\text{Clock frequency selected by CGSYSCR<PRCK[1:0]>}}{(\text{TBxRG1} \times 2) \times 2 \times 16}$$

In the case the timer prescaler clock ΦT1 (2division ratio) is selected.

One clock cycle is a period that the timer flip-flop is inverted twice.

12.7 Transmit/Receive Buffer and FIFO

12.7.1 Configuration

Figure 12-4 shows the configuration of transmit buffer, receive buffer and FIFO.

Appropriate settings are required for using buffer and FIFO. The configuration may be predefined depending on the mode.

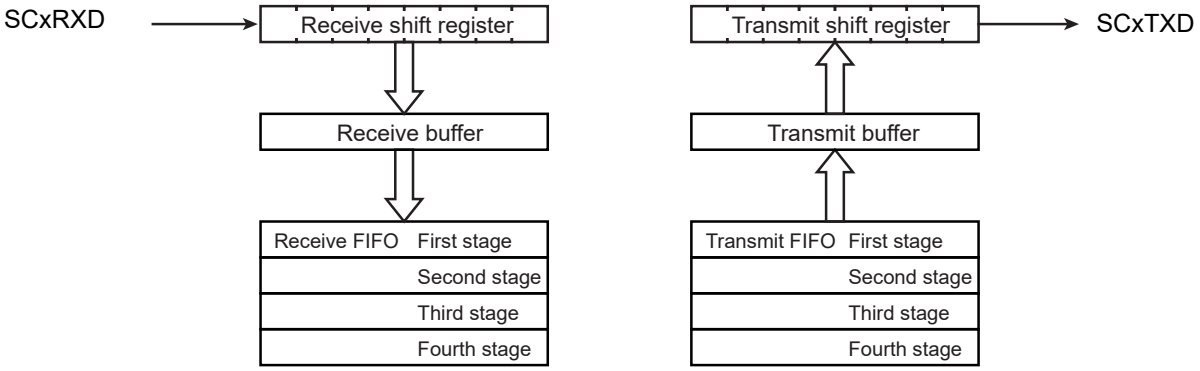


Figure 12-4 The Configuration of Buffer and FIFO

12.7.2 Transmit/Receive Buffer

Transmit buffer and receive buffer are double-buffered. The buffer configuration is specified by SCxMOD2<WBUF>.

When serial channel is operated as receive, if it is operated as clock input mode in the I/O interface mode or it is operated as the UART mode, it's double buffered regardless of <WBUF> settings.

In other modes, it's according to the <WBUF> settings.

Table 12-5 shows correlation between modes and buffers.

Table 12-5 Mode and buffer Composition

Mode		SCxMOD2<WBUF>	
		"0"	"1"
UART mode	Transmit	Single	Double
	Receive	Double	Double
I/O interface mode (Clock input mode)	Transmit	Single	Double
	Receive	Double	Double
I/O interface mode (Clock output mode)	Transmit	Single	Double
	Receive	Single	Double

12.7.3 Initialize Transmit Buffer

When transmission is stopped with a data in the transmit buffer, it is necessary to initialize the transmit buffer before new transmit data is written to transmit buffer.

The transmit buffer must be initialized when the transmit operation is stopped. To stop the transmit operation can be confirmed by reading SCxMOD2<TXRUN>. After confirming to stop the transmit operation, SCxTFC<TBCLR> is set to "1" and initialize the transmit buffer.

When a transmit FIFO is enabled, the initialize operation is depend on the data in a transmit FIFO. If transmit FIFO has data, a data is transferred from a transmit FIFO to a transmit buffer. If is does not have data, SCxMOD2<RBEMP> is set to "1".

Note: In the I/O interface mode with clock input mode is input asynchronously. When transmit operation is stopped, do not input the clock.

12.7.4 FIFO

In addition to the double buffer function above described, 4-byte FIFO can be used.

To enable FIFO, enable the double buffer by setting SCxMOD2<WBUF> to "1" and SCxFCNF<CNFG> to "1". The FIFO buffer configuration is specified by SCxMOD1<FDPX[1:0]>.

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

Table 12-6 shows correction between modes and FIFO.

Table 12-6 Mode and FIFO Composition

	SCxMOD1<FDPX[1:0]>	Receive FIFO	Transmit FIFO
Half duplex Receive	"01"	4byte	-
Half duplex Transmit	"10"	-	4byte
Full duplex	"11"	2byte	2byte

12.8 Status Flag

The SCxMOD2 has two types of flag. This bit is significant only when the double buffer is enabled.

<RBFLL> is a flag to show that the receive buffer is full. When one frame of data is received and the data is moved from the receive shift register to the receive buffers, this bit changes to "1". When reading the receive buffer is read, this bit is cleared to "0".

<TBEMP> shows that the transmit buffer is empty. When data in the transmit buffers is moved to the transmit shift register, this bit is set to "1". When data is set to the transmit buffers, the bit is cleared to "0".

12.9 Error Flag

Three error flags are provided in the SCxCR. The meaning of the flags is changed depending on the modes. The table below shows the meanings in each mode.

These flags are cleared to "0" after reading the SCxCR.

<table>
<tr><th rowspan="2">Mode</th><th colspan="3">Flag</th></tr>
<tr><th><OERR></th><th><PERR></th><th><FERR></th></tr>
<tr><td>UART mode</td><td>Over-run error</td><td>Parity error</td><td>Framing error</td></tr>
<tr><td rowspan="2">I/O Interface mode
(Clock input mode)</td><td rowspan="2">Over-run error</td><td>Under-run error
(When a double buffer and FIFO are used)</td><td rowspan="2">Fixed to 0</td></tr>
<tr><td>Fixed to 0
(When a double buffer and FIFO are not used)</td></tr>
<tr><td>I/O Interface mode
(Clock output mode)</td><td>Undefined</td><td>Undefined</td><td>Fixed to 0</td></tr>
</table>

12.9.1 OERR Flag

In both UART and I/O interface modes, this bit is set to "1" when an error is generated by completing the reception of the next frame before the receive buffer has been read.

If the receive FIFO is enabled, the received data is automatically moved to the receive FIFO and no over-run error will be generated until the receive FIFO is full (or until the usable bytes are fully occupied).

In the I/O interface mode with clock output mode, the SCxSCLK pin output stops upon setting the flag.

Note: **To switch from the I/O interface mode with clock output mode to other modes, read the SCxCR and clear the overrun flag.**

12.9.2 PERR Flag

This flag indicates a parity error in the UART mode and an under-run error or completion of transmit in the I/O interface mode.

In the UART mode, <PERR> is set to "1" when the parity generated from the received data is different from the received parity bit.

In the I/O interface mode, <PERR> is set to "1" under the following conditions when a double buffer is enabled.

In the clock input mode, <PERR> is set to "1" when the clock is input after completing data output of the transmit shift register with no data in the transmit buffer.

In the clock output mode, <PERR> is set to "1" after completing output of all data and the clock output stops.

Note: To switch from the I/O interface mode with clock output mode to other modes, read the SCxCR and clear the under-run flag.

12.9.3 FERR Flag

A framing error is generated if the corresponding stop bit is determined to be "0" by sampling the bit at around the center. Regardless of the stop bit length settings in the SCxMOD2<SBLEN>, the stop bit status is determined by only 1’st STOP bit.

This bit is fixed to "0" in the I/O interface mode.

12.10 Receive

12.10.1 Receive Counter

The receive counter is a 4-bit binary counter and is up-counted by SIOCLK.

In the UART mode, sixteen SIOCLK clock pulses are used in receiving a single data bit and the data symbol is sampled at the eighth pulse.

12.10.2 Receive Control Unit

12.10.2.1 I/O interface mode

In the clock output mode with SCxCR <IOC> set to "0", the SCxRXD pin is sampled on the rising edge of SCxSCLK pin.

In the clock input mode with SCxCR <IOC> set to "1", the SCxRXD pin is sampled on the rising or falling edge of SCxSCLK pin depending on the SCxCR <SCLKS>.

12.10.2.2 UART Mode

The receive control unit has a start bit detection circuit, which is used to initiate receive operation when a normal start bit is detected.

12.10.3 Receive Operation

12.10.3.1 Receive Buffer

The received data is stored by 1 bit in the receive shift register. When a complete set of bits has been stored, the interrupt INTRXx is generated.

When the double buffer is enabled, the data is moved to the receive buffer (SCxBUF) and the receive buffer full flag (SCxMOD2<RBFLL>) is set to "1". The receive buffer full flag is cleared to "0" by reading the receive buffer. When the double buffer is disabled, the receive buffer full flag has no meaning.

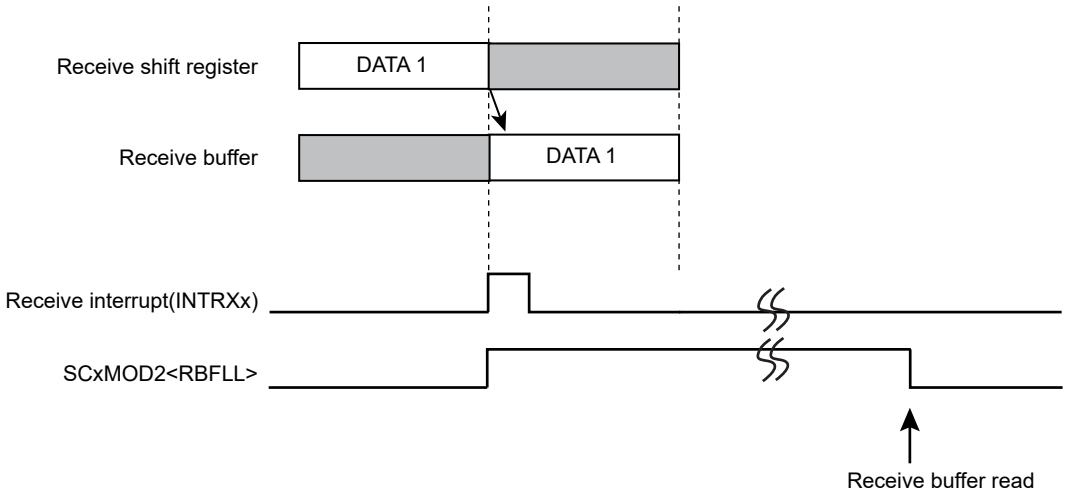


Figure 12-5 Receive Buffer Operation

12.10.3.2 Receive FIFO Operation

When FIFO is enabled, the received data is moved from receive buffer to receive FIFO and the receive buffer full flag is cleared immediately. An interrupt will be generated according to the SCxRFC<RIL[1:0]>.

Note: **When the data with parity bit are received in UART mode by using the FIFO, the parity error flag is shown the occurring the parity error in the received data.**

The configurations and operations in the half duplex Receive mode are described as follows.

SCxMOD1<FDPX[1:0]> = "01"	:Transfer mode is set to half duplex mode
SCxFCNF<RFST><TFIE><RFIE>	:Automatically inhibits continuous reception after reaching the fill level.
<RXTCNT><CNFG> = "10111"	:The number of bytes to be used in the receive FIFO is the same as the interrupt generation fill level.
SCxRFC<RIL[1:0]> = "00"	:The fill level of FIFO in which generated receive interrupt is set to 4 bytes
SCxRFC<RFCS><RFIS> = "01"	:Clears receive FIFO and sets the condition of interrupt generation.

After setting of the above FIFO configuration, the data reception is started by writing "1" to the SCxMOD0<RXE>. When the data is stored all in the receive shift register, receive buffer and receive FIFO, SCxMOD0<RXE> is automatically cleared and the receive operations completed.

In the above condition, if the continuous reception after reaching the fill level is enabled, it becomes possible to receive a data continuously by reading the data in the FIFO.

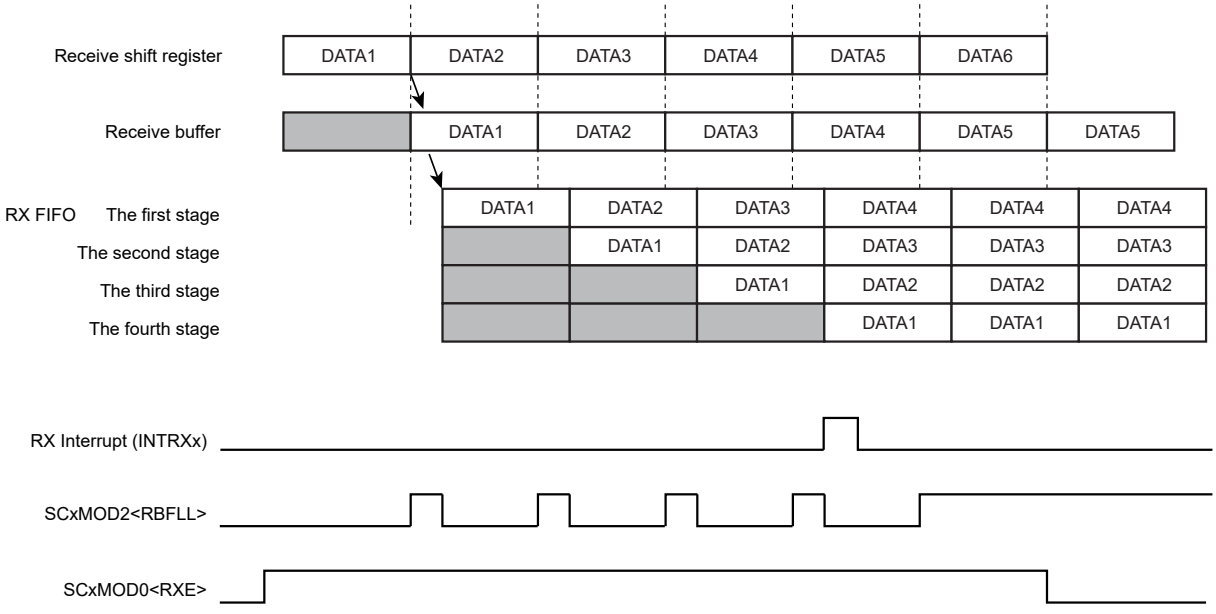


Figure 12-6 Receive FIFO Operation

12.10.3.3 I/O interface mode with clock output mode

In the I/O interface mode with clock output mode setting, clock stops when all received data is stored in the receive buffer and FIFO. So, in this mode, the over-run error flag has no meaning.

The timing of SCLK output stop and re-output depends on receive buffer and FIFO.

(1) Case of single buffer

Stop clock output after receiving a data. In this mode, I/O interface can transfer each data with the transfer device by hand-shake.

When the data in a buffer is read, clock output is restarted.

(2) Case of double buffer

Stop clock output after receiving the data into a receive shift register and a receive buffer.

When a data is read, clock output is restarted.

(3) Case of FIFO

Stop clock output after receiving the data into a shift register, received buffer and FIFO.

When one byte data is read, the data in the received buffer is transferred into FIFO and the data in the receive shift register is transferred into the received buffer and clock output restarts.

And if SCxFCNF<RXTXCNT>is set to "1", clock stops and receive operation stops with clearing SCxMOD0<RXE>.

12.10.3.4 Read Received Data

In spite of enabling or disabling FIFO, read the received data from the receive buffer (SCxBUF).

When receive FIFO is disabled, the buffer full flag SCxMOD2<RBFLL> is cleared to "0" by this reading. The next data can be received in the receive shift register before reading a data from the receive buffer. The parity bit to be added in the 8-bit UART mode as well as the most significant bit in the 9-bit UART mode will be stored in SCxCR<RB8>.

When the receive FIFO is enabled, the 9-bit UART mode is prohibited because up to 8-bit data can be stored in receive FIFO. In the 8-bit UART mode, the parity bit is lost but parity error is determined and the result is stored in SCxCR<PERR>.

12.10.3.5 Wake-up Function

In the 9-bit UART mode, the slave controller can be operated in the wake-up mode by setting the wake-up function SCxMOD0 <WU> to "1". In this case, the interrupt INTRXx will be generated only when SCxCR <RB8> is set to "1".

12.10.3.6 Overrun Error

When receive FIFO is disabled, the overrun error occurs without completing reading data before receiving the next data. When an overrun error occurs, a content of receive buffer and SCxCR<RB8> is not lost, but a content of receive shift register is lost.

When receive FIFO is enabled, overrun error is occurred and set overrun flag by no reading receive FIFO before moving the next data into received buffer when receive FIFO is full. In this case, the contents of receive FIFO are not lost.

In the I/O interface mode with clock output mode, the clock output automatically stops, so this flag has no meaning.

Note: When the mode is changed from I/O interface mode with clock output mode to the other modes, read SCxCR and clear overrun flag.

12.11 Transmit

12.11.1 Transmit Counter

The transmit counter is a 4-bit binary counter and is counted by SIOCLK as in the case of the receive counter. In UART mode, it generates a transmit clock (TXDCLK) on every 16th clock pulse.

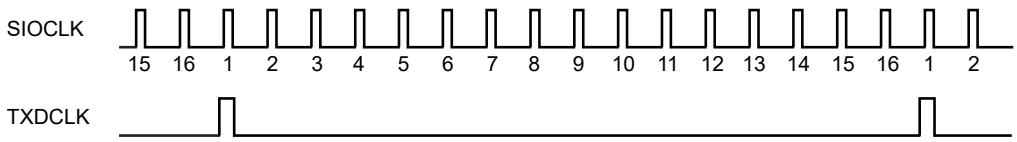


Figure 12-7 Generation of Transmission Clock in UART mode

12.11.2 Transmit Control

12.11.2.1 In I/O Interface Mode

In the clock output mode with SCxCR<IOC> set to "0", each bit of data in the transmit buffer is outputted to the SCxTXD pin on the falling edge of SCxSCLK pin.

In the clock input mode with SCxCR<IOC> set to "1", each bit of data in the transmit buffer is outputted to the SCxTXD pin on the rising or falling edge of the SCxSCLK pin according to the SCxCR<SCLKS>.

12.11.2.2 In UART Mode

When the transmit data is written in the transmit buffer, data transmission is initiated on the rising edge of the next TXDCLK and the transmit shift clock signal is also generated.

12.11.3 Transmit Operation

12.11.3.1 Operation of Transmit Buffer

If double buffering is disabled, the CPU writes data only to transmit shift register and the transmit interrupt INTTXx is generated upon completion of data transmission.

When double buffering is enabled (including the case where the transmit FIFO is enabled), if "1" is set to SCxMOD1<TXE>, data in the transmit buffer is transferred to the transmit shift register. The INTTXx interrupt is generated at the same time and the transmit buffer empty flag (SCxMOD2<TBEMP>) is set to "1". This flag indicates that the next transmit data can be written. When the next data is written to the transmit buffer, the <TBEMP> flag is cleared to "0".

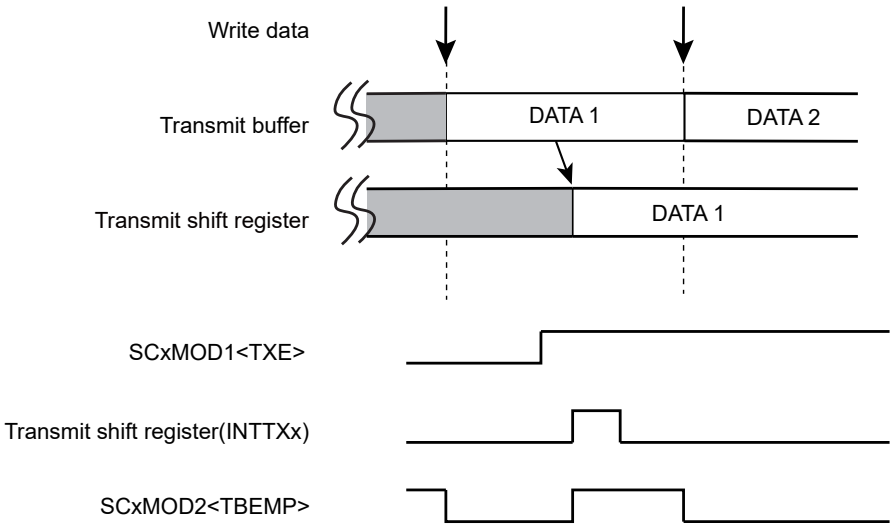


Figure 12-8 Operation of Transmit Buffer (Double-buffer is enabled)

12.11.3.2 Transmit FIFO Operation

When FIFO is enabled, the maximum 5-byte data can be stored using the transmit buffer and FIFO. Once transmission is enabled, data is transferred to the transmit shift register from the transmit buffer and start transmission. If data exists in the FIFO, the data is moved to the transmit buffer immediately, and the <TBEMP> flag is cleared to "0".

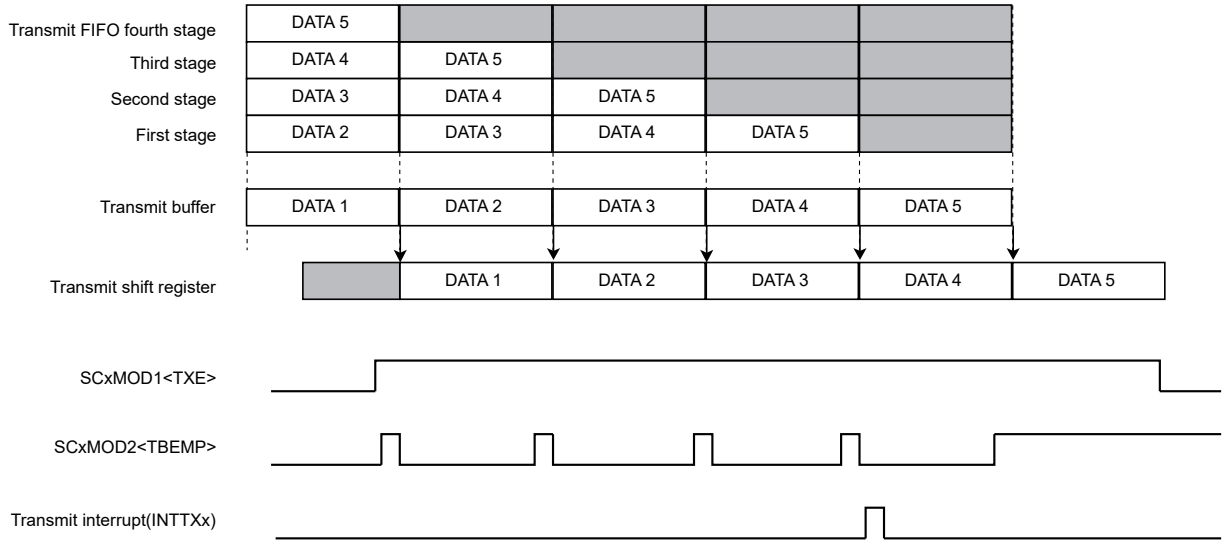
Note: To use Transmit FIFO buffer, Transmit FIFO must be cleared after setting the SIO transfer mode (half duplex/ full duplex) and enabling FIFO (SCxFCNF<CNFG>="1").

Settings and operations to transmit 5 bytes data stream by setting the transfer mode to half duplex are shown as below.

SCxMOD1<FDPX[1:0]> = "10"	:Transfer mode is set to half duplex.
SCxFCNF<RFST><TFIE><RFIE>	:Transmission is automatically disabled if FIFO becomes empty.
<RXTXCNT><CNFG> = "11011"	:The number of bytes to be used in the receive FIFO is the same as the interrupt :generation fill level.
SCxTFC<TIL[1:0]> = "00"	:Sets the interrupt generation fill level to "0".
SCxTFC<TFCS><TFIS> = "11"	:Clears receive FIFO and sets the condition of interrupt generation.
SCxFCNF<CNFG> = "1"	:Enable FIFO

After above settings are configured, data transmission can be initiated by writing 5 bytes of data to the transmit buffer and FIFO, and setting the SCxMOD1<TXE> bit to "1". When the last transmit data is moved to the transmit buffer, the transmit interrupt is generated. When transmission of the last data is completed, the clock is stopped and the transmission sequence is terminated.

Once above settings are configured, if the transmission is not set as auto disabled, the transmission should lasts writing transmit data.



12.11.3.3 Transmit in I/O interface Mode with Clock Output Mode

In the I/O interface mode with clock output mode, the clock output automatically stops when all data transmission is completed and underrun error will not occur.

The timing of suspension and resume of clock output is different depending on the buffer and FIFO usage.

(1) Single Buffer

The clock output stops each time one frame of data is transferred. Handshaking for each data with the other side of communication can be enabled. The clock output resumes when the next data is written in the buffer.

(2) Double Buffer

The clock output stops upon completion of data transmission in the transmit shift register and the transmit buffer. The clock output resumes when the next data is written in the buffer.

(3) FIFO

The transmission of all data stored in the transmit shift register, transmit buffer and FIFO is completed, the SCLK output stops. The next data is written, clock output resumes.

If SCxFCNF<RXTXCNT> is configured, SCxMOD0<TXE> bit is cleared at the same time as clock stops and the transmission stops.

12.11.3.4 Level of SCxTXD pin after the last bit is output in I/O interface mode

The level of SCxTXD pin after the data hold time is passed after the last bit is output is specified by SCxCR<TIDLE>.

When SCxCR<TIDLE> is "00", the level of SCxTXD pin is output "Low" level. When SCxCR<TIDLE> is "01", the level of SCxTXD pin is output "High" level. When SCxCR<TIDLE> is "10", the level of SCxTXD pin is output the level of the last bit.

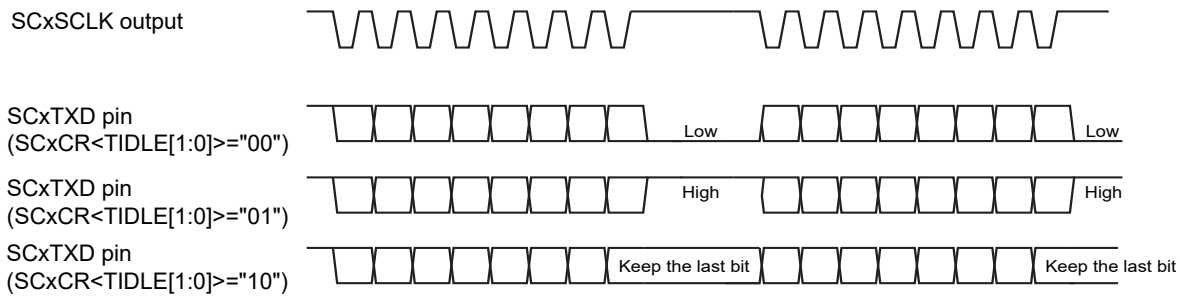


Figure 12-9 Level of SCxTXD pin After the last bit is output

12.11.3.5 Under-run error

In the I/O interface mode with clock input mode and if FIFO is empty and if no data is set in transmit buffer before the next frame clock input, which occurs upon completion of data transmission from transmit shift register, an under-run error occurs and SCxCR<PERR> is set to "1".

The level of a SCxTXD pin can be specified by SCxCR<TXDEMP>. When SCxCR<TXDEMP> is "0", a SCxTXD pin outputs "Low" level during data output period. When SCxCR<TXDEMP> is "1", a SCxTXD pin outputs "High" level.

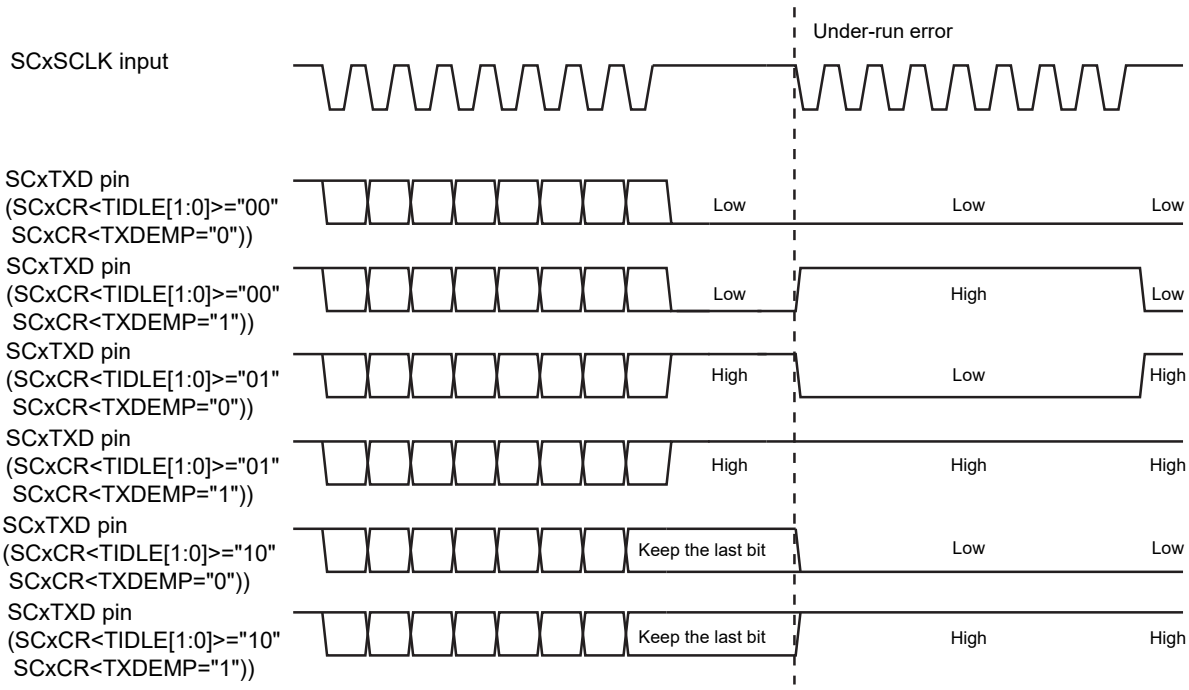


Figure 12-10 Level of SCxTXD pin when Under-run Error is Occurred

In the I/O interface mode with SCLK output setting, the clock output automatically stops, so SCxCR<PERR> has no meaning.

Note: Before switching the I/O interface mode with clock output mode to other modes, read the SCxCR and clear the under-run flag.

12.11.3.6 Data Hold Time In the I/O interface mode with clock input mode

In the I/O interface mode with clock input mode, a data hold time of the last bit can be adjusted by SCxCR<EHOLD[2:0]>. Specify a data hold time and the period of the SCLK to satisfy the following formula.

The data hold time of the last bit ≤ The period of SCLK / 2

12.12 Handshake function

The function of the handshake is to enable frame-by-frame data transmission by using the $\overline{\text{SCxCTS}}$ (Clear to send) pin and to prevent over-run errors. This function can be enabled or disabled by SCxMOD0<CTSE>.

When the $\overline{\text{SCxCTS}}$ pin is set to "High" level, the current data transmission can be completed but the next data transmission is suspended until the $\overline{\text{SCxCTS}}$ pin returns to the "Low" level. The INTTXx interrupt is generated in the normal timing, the next transmit data is written in the transmit buffer, and it waits until it is ready to transmit data.

Note 1: If the $\overline{\text{CTS}}$ signal is set to "High" level during transmission, the next data transmission is suspended after the current transmission is completed.

Note 2: Data transmission starts on the first falling edge of the TXDCLK clock after $\overline{\text{CTS}}$ is set to "Low" level.

Although no $\overline{\text{RTS}}$ pin is provided, a handshake control function can easily implemented by assigning one bit of the port for the $\overline{\text{RTS}}$ function. By setting the port to "High" level upon completion of data reception (in the receive interrupt routine), the transmit side can be requested to suspend data transmission.

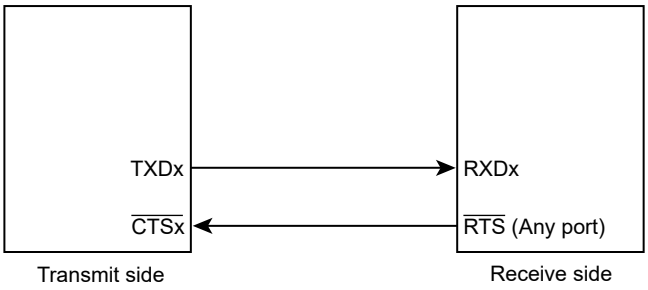


Figure 12-11 Handshake Function

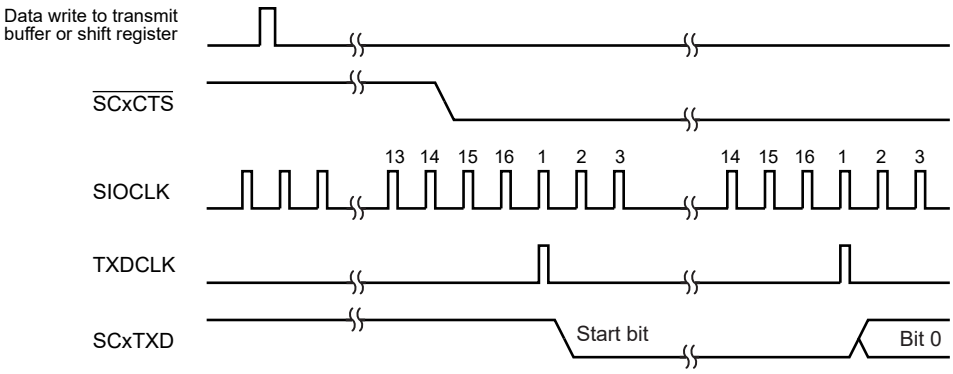


Figure 12-12 $\overline{\text{SCxCTS}}$ Signal timing

12.13 Interrupt/Error Generation Timing

12.13.1 Receive Interrupts

Figure 12-13 shows the data flow of receive operation and the route of read.

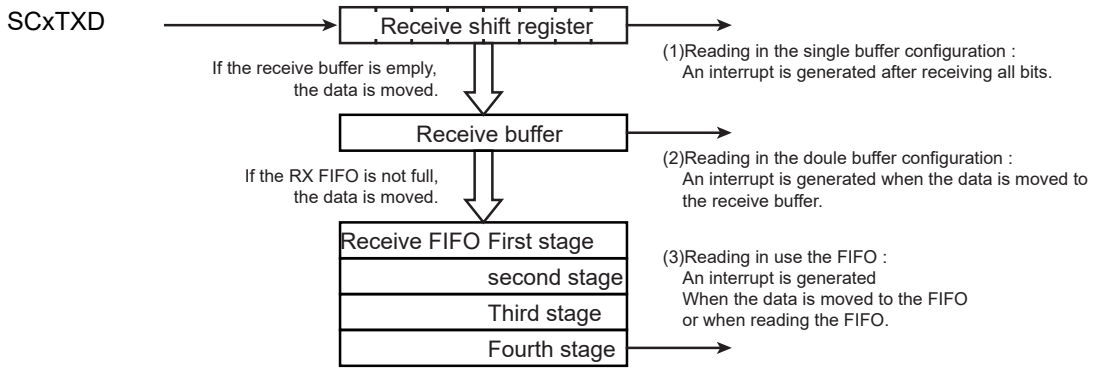


Figure 12-13 Receive Buffer/FIFO Configuration Diagram

12.13.1.1 Single Buffer / Double Buffer

Receive interrupts are generated at the time depends on the transfer mode and the buffer configurations, which are given as follows.

Table 12-7 Receive Interrupt Conditions in use of Single Buffer / Double Buffer

Buffer Configurations	UART modes	IO interface modes
Single Buffer	–	Immediately after the raising / falling edge of the last SCxSCLK pin (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Double Buffer	A receive interrupt occurs when data is transferred from the receive shift register to the receive buffter.Specific timings are : ・If data does not exist in the receive buffer, a receive interrupt occurs in the vicinity of the center of the 1st stop bit. ・If data exists in both the receive shift register and the receive buffer, a receive interrupt occurs when the buffer is read.	A receive interrupt occurs when data is transferred from the receive shift register to the receive buffer.Specific timings are: ・If data does not exit int the receive buffer, a receive interrupt occurs immediately after on rising/falling edge of SCxSCLK pin of the last bit. (The setting of rising edge or falling edge is specified with SCxCR<SCLKS>.) ・If data exists in both the receive shift register and the receive buffer, a receive interrupt occurs when the buffer is read.

Note: Interrupts are not generated when an over-run error is occurred.

12.13.1.2 FIFO

When the FIFO is used, a receive interrupt occurs on depending on the timing described in Table 12-8 and the condition specified with SCxRFC<RFIS>.

Table 12-8 Receive Interrupt Conditions in use of FIFO

SCxRFC<RFIS>	Interrupt conditions	Interrupt generation timing
"0"	When FIFO fill level (SCxRST<RLVL[2:0]>) = Receive FIFO fill level to generate receive interrupt <RIL[1:0]>	・When transfer a received data from receive buffer to receive FIFO
"1"	When FIFO fill level (SCxRST<RLVL[2:0]>) ≥ Receive FIFO fill level to generate receive interrupt <RIL[1:0]>	・When transfer a received data from receive buffer to receive FIFO ・When read a receive data from receive FIFO

12.13.2 Transmit interrupts

Figure 12-14 shows the data flow of transmit operation and the route of read.

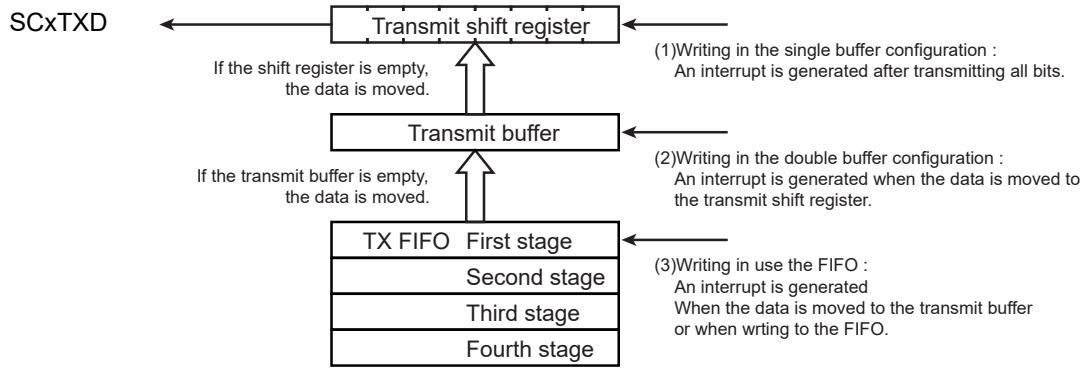


Figure 12-14 Transmit Buffer / FIFO Configuration Diagram

12.13.2.1 Singe Buffer / Double Buffer

Transmit interrupts are generated at the time depends on the transfer mode and the buffer configurations, which are given as follows.

Table 12-9 Transmit Interrupt conditions in use of Single Buffer/Double Buffer

Buffer Configurations	UART modes	IO interface modes
Single Buffer	Just before the stop bit is sent	Immediately after the raising / falling edge of the last SCxSCLK pin (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Double Buffer	When a data is moved from the transmit buffet to the transmit shift register. If "1" is set to SCxMOD1<TXE> and the transmit shift register is empty, a transmit interrupt occurs because data is immediately transferred to the transmit shift register from the transmit buffer.	

12.13.2.2 FIFO

When the FIFO is used, a transmit interrupt occurs depending on the timing described in Table 12-10 and the condition specified with SCxTFC<TFIS>.

Table 12-10 Transmit Interrupt conditions in use of FIFO

SCxTFC<TFIS>	Interrupt condition	Interrupt generation timing
"0"	When FIFO fill level (SCxTST<TLVL[2:0]>) = Transmit FIFO fill level to generate transmit interrupt <TIL[1:0]>	・When transmitted data is transferred from transmit FIFO to transmit buffer
"1"	When FIFO fill level (SCxTST<TLVL[2:0]>) ≤ Transmit FIFO fill level to generate transmit interrupt <TIL[1:0]>	・When transmitted data is transferred from transmit FIFO to transmit buffer ・When transmit data is write into transmit FIFO

12.13.3 Error Generation

12.13.3.1 UART Mode

Error	9 bits	7 bits 8 bits 7 bits + Parity 8 bits + Parity
Framing Error over-run Error	Around the center of stop bit	
Parity Error	−	Around the center of parity bit

12.13.3.2 I/O Interface Mode

over-run Error	Immediately after the raising / falling edge of the last SCxSCLK pin (Rising or falling is determined according to SCxCR<SCLKS> setting.)
Under-run Error	Immediately after the rising or falling edge of the next SCxSCLK pin. (Rising or falling is determined according to SCxCR<SCLKS> setting.)

Note: Over-run error and Under-run error have no meaning in clock output mode.

12.14 DMA Request

DMA transfer can be started at the timing of interrupt request.

When you perform a DMA transfer, please set up the bit of a SCxDMA.

Please refer to the chapter of "product information" for the channel which can be used for a DMA request with this product.

Note 1: In case using DMA transfer by transmit or receive interrupt request, enabled DMA and set transmit and receive registers after generating software reset by SCxMOD<SWRST>.

Note 2: When the DMA transfer is used, the FIFO cannot be used.

12.15 Software Reset

Software reset is generated by writing SCxMOD2<SWRST[1:0]> as "10" followed by "01".

As a result, SCxMOD0<RXE>, SCxMOD1<TXE>, SCxMOD2<TBEMP><RBFLL><TXRUN>, SCxCR <OERR><PERR><FERR> are initialized. And the receive circuit and the transmit circuit become initial state.

Other states are maintained.

12.16 Operation in Each Mode

12.16.1 Mode 0 (I/O interface mode)

The I/O interface mode is selected by setting SCxMOD<SM[1:0]> to "00".

Mode 0 consists of two modes, the clock output mode to output synchronous clock (SCLK) and the clock input mode to accept synchronous clock (SCLK) from an external source.

The operation with disabling a FIFO in each mode is described below. Regarding a FIFO, refer to a receive FIFO and a transmit FIFO which are described before.

12.16.1.1 Transmit

(1) Clock Output Mode

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

  Data is output from the SCxTXD pin and the clock is output from the SCxSCLK pin each time the CPU writes data to the transmit buffer. When all data is output, an interrupt (INTTXx) is generated.

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

  When data is written to the transmit buffer and the shift register is empty, or when data transmission from the shift register is completed, data is transferred to the shift register from the transmit buffer. Simultaneously, SCxMOD2<TBEMP> is set to "1", and the INTTXx interrupt is generated.

  If the transmit buffer has no data to be moved to the transmit shift register, INTTXx interrupt is not generated and the clock output stops.

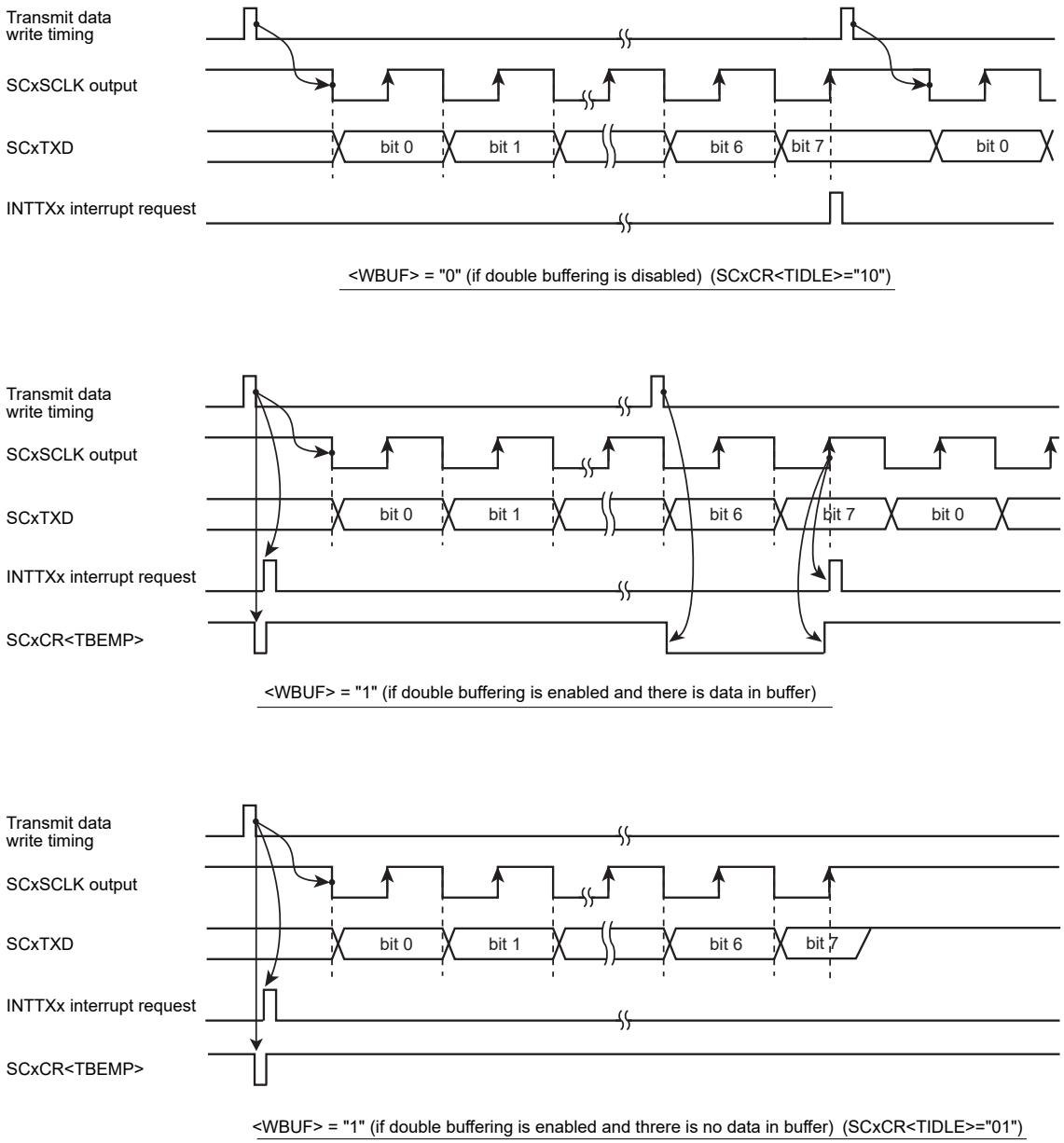


Figure 12-15 Transmit Operation in the I/O Interface Mode (Clock Output Mode)

(2) Clock Input Mode

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

  If the clock is input in the condition where data is written in the transmit buffer, 8-bit data is output from the SCxTXD pin. When all data is output, an interrupt INTTXx is generated. The next transmit data must be written before the timing of point "A" as shown in Figure 12-16.

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

  Data is moved from the transmit buffer to the transmit shift register when the CPU writes data to the transmit buffer before the clock input becomes active or when data transmission from the transmit shift register is completed. Simultaneously, SCxMOD2<TBEMP> is set to "1", and the INTTXx interrupt is generated.

  If the clock input becomes active while no data is in the transmit buffer, although the internal bit counter is started, an under-run error occurs and the level which is specified by SCxCR<TXDEMP> is output to SCxTXD pin.

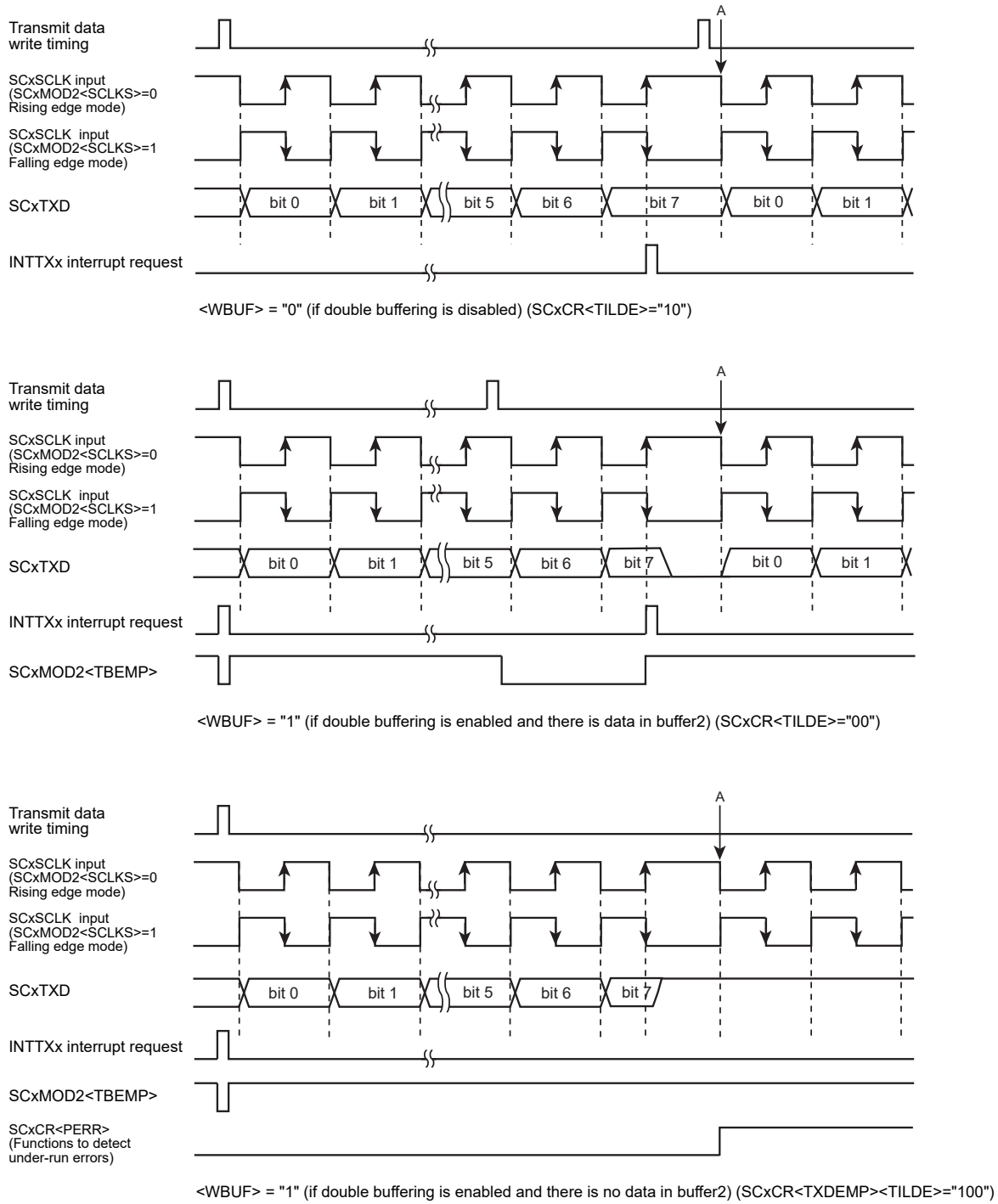


Figure 12-16 Transmit Operation in the I/O Interface Mode (Clock Input Mode)

12.16.1.2 Receive

(1) Clock Output Mode

The clock output starts by setting the receive enable bit SCxMOD0<RXE> to "1".

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

  A clock is output from the SCxSCLK pin and the next data is stored into the shift register each time the CPU reads received data. When all the 8 bits are received, the INTRXx interrupt is generated.

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

  Data stored in the shift register is moved to the receive buffer and the receive buffer can receive the next frame. A data is moved from the shift register to the receive buffer, SCxMOD2<RBFLL> is set to "1" and the INTRXx is generated.

  When a data is in the receive buffer, if the data is not read from the receive buffer before completing reception of the next 8 bits, the INTRXx interrupt is not generated and the clock output stops. In this state, reading data from the receive buffer allows data in the shift register to move to the receive buffer and thus the INTRXx interrupt is generated and data reception resumes.

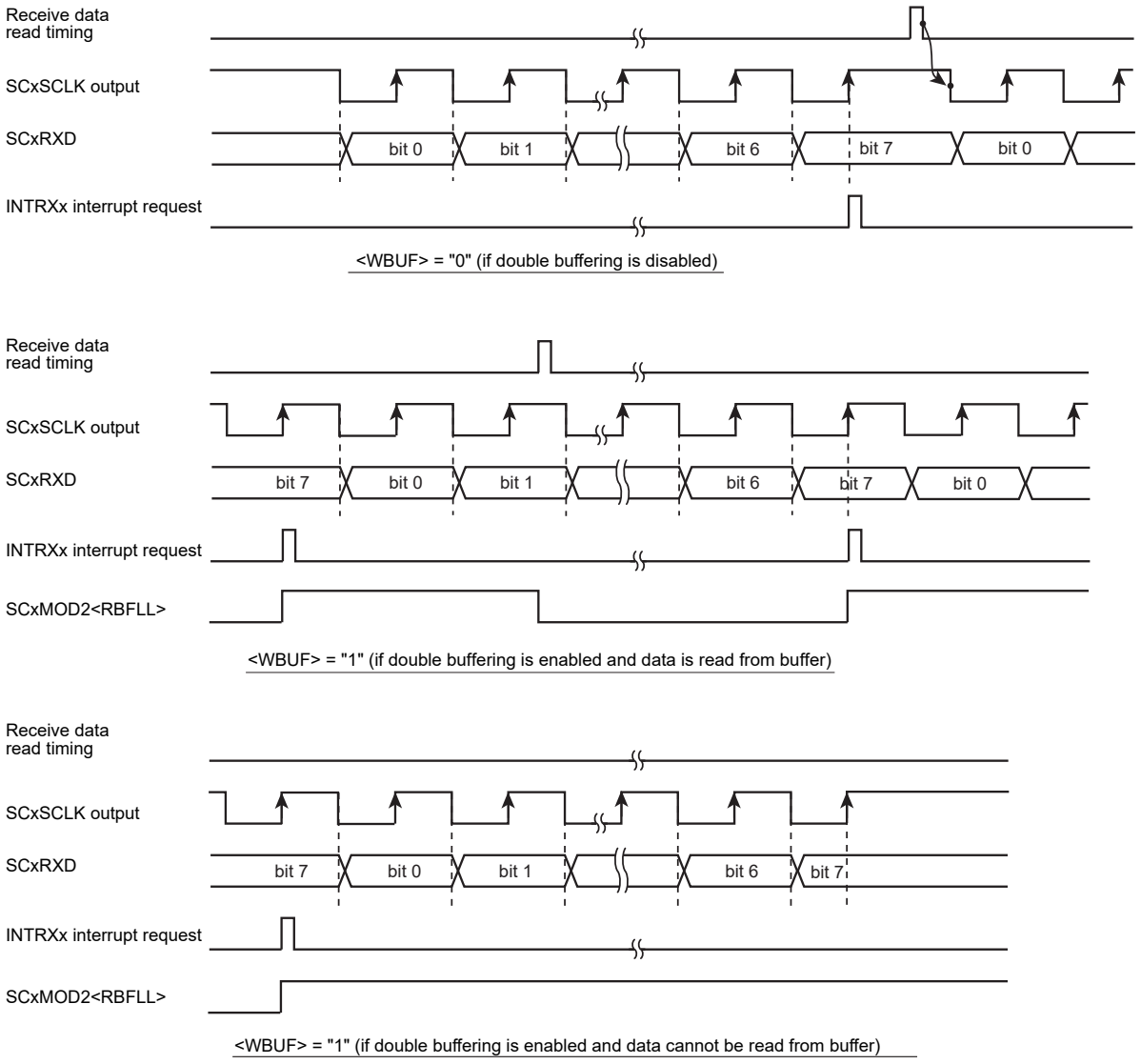


Figure 12-17 Receive Operation in the I/O Interface Mode (Clock Output Mode)

(2) clock input mode

In the clock input mode, receiving double buffering is always enabled, the received data can be moved to the receive buffer from the shift register, and the receive buffer can receive the next data successively.

The INTRXx receive interrupt is generated each time received data is moved to the receive buffer.

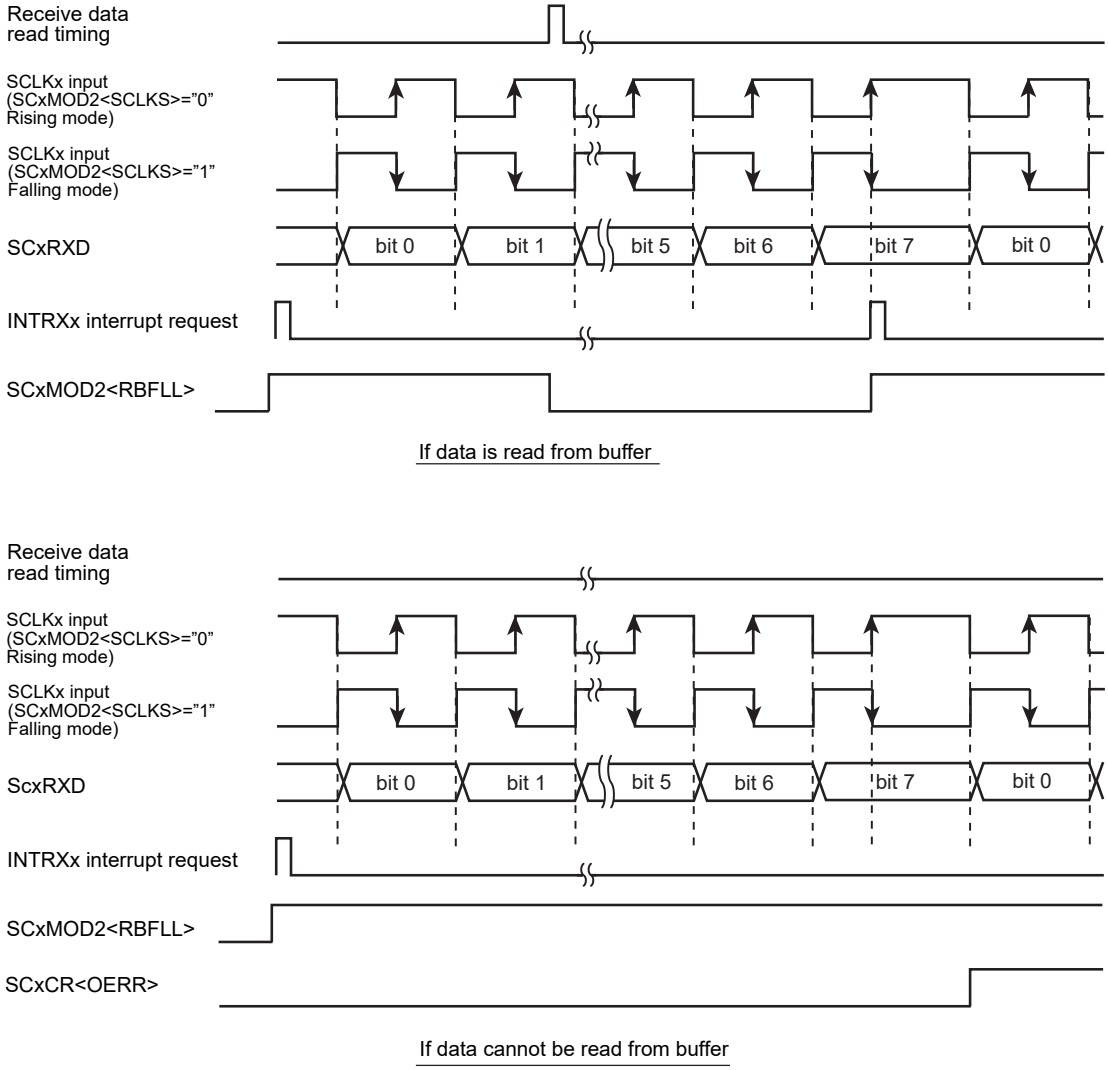


Figure 12-18 Receive Operation in the I/O Interface Mode (Clock Input Mode)

12.16.1.3 Transmit and Receive (Full-duplex)

(1) Clock Output Mode

- If double buffers are disabled (SCxMOD2<WBUF> = "0")

  Clock is output when the CPU writes data to the transmit buffer.

  Subsequently, a data is shifted into receive buffer and the INTRXx is generated. Concurrently, a data written to the transmit buffer is output from the SCxTXD pin, the INTTXx is generated when transmission of all data has been completed. Then, the clock output stops.

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

- If double buffers are enabled (SCxMOD2<WBUF> = "1")

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

  A data is shifted into the receive shift register, moved to the receive buffer, and the INTRXx is generated. While a data is received, a transmit data is output from the SCxTXD pin. When all data are sent out, the INTTXx is generated and the next data is moved from the transmit buffer to the transmit shift register.

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

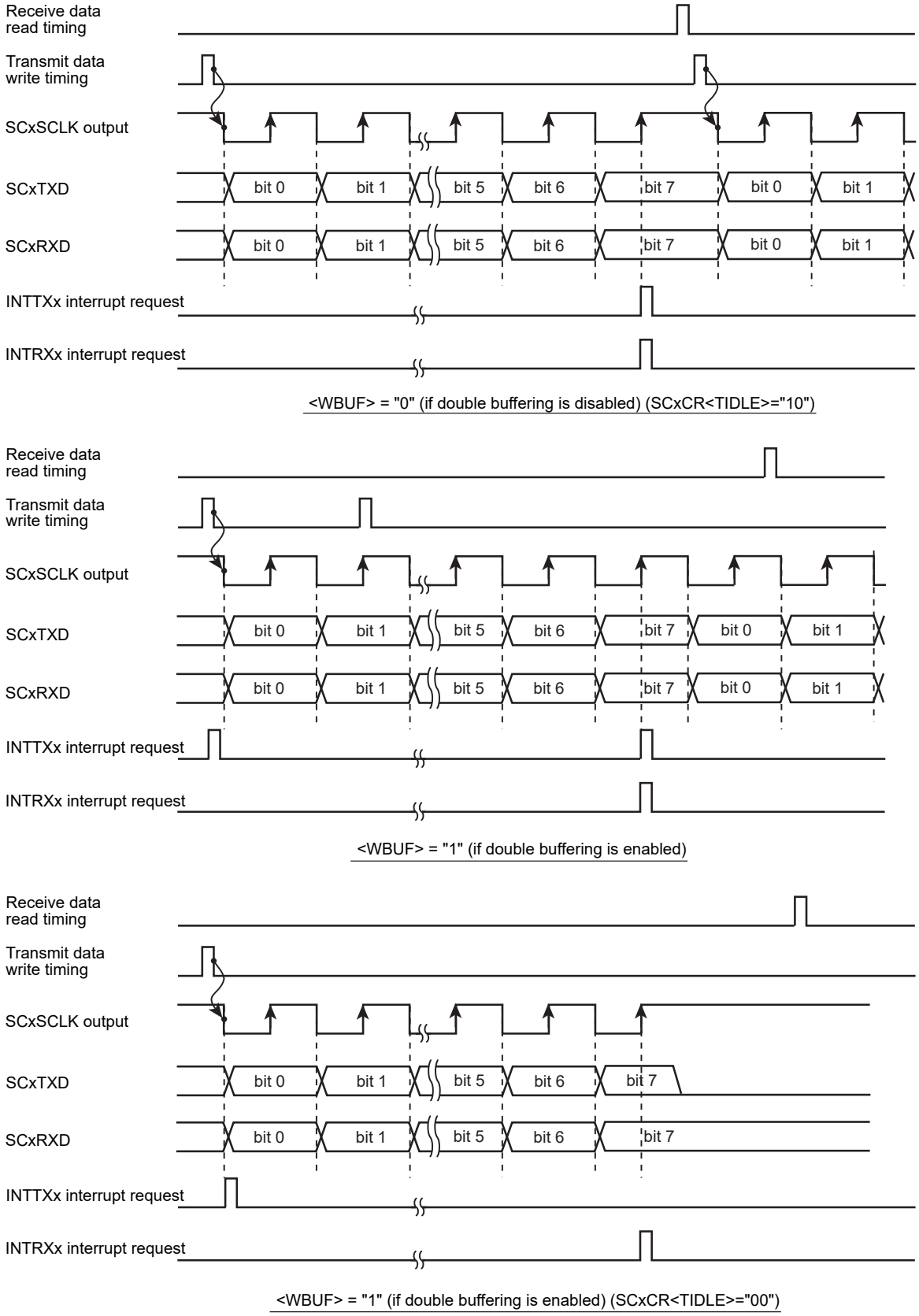


Figure 12-19 Transmit/Receive Operation in the I/O Interface Mode (SCLK Output Mode)

(2) Clock Input Mode

- If double buffers are disabled. (SCxMOD2<WBUF> = "0")

  When receiving data, double buffer is always enabled regardless of the SCxMOD2 <WBUF> settings.

  A data written in the transmit buffer is outputted from the SCxTXD pin and a data is shifted into the receive buffer when the clock input becomes active. The INTTXx is generated upon completion of data transmission. The INTRXx is generated when the data is moved from shift register to receive buffer after completion of data reception.

  Note that transmit data must be written into the transmit buffer before the clock input for the next data (data must be written before the point A in Figure 12-20). Data must be read before completing reception of the next data.

- If double buffers are enabled. (SCxMOD2<WBUF> = "1")

  The INTTXx is generated at the timing the transmit buffer data is moved to the transmit shift register after completing data transmission from the transmit shift register. At the same time, data received is shifted to the shift register, it is moved to the receive buffer, and the INTRXx is generated.

  Note that transmit data must be written into the transmit buffer before the clock input for the next data (data must be written before the point A in Figure 12-20). Data must be read before completing reception of the next data.

  Upon the clock input for the next data, transmission from transmit shift register (in which data has been moved from transmit buffer) is started while receive data is shifted into receive shift register simultaneously.

  If data in receive buffer has not been read when the last bit of the data is received, an overrun error occurs.

  If there is no data written to transmit buffer when clock for the next data is input, an underrun error occurs. The level which is specified by SCxCR<TXDEMP> is output to SCxTXD pin.

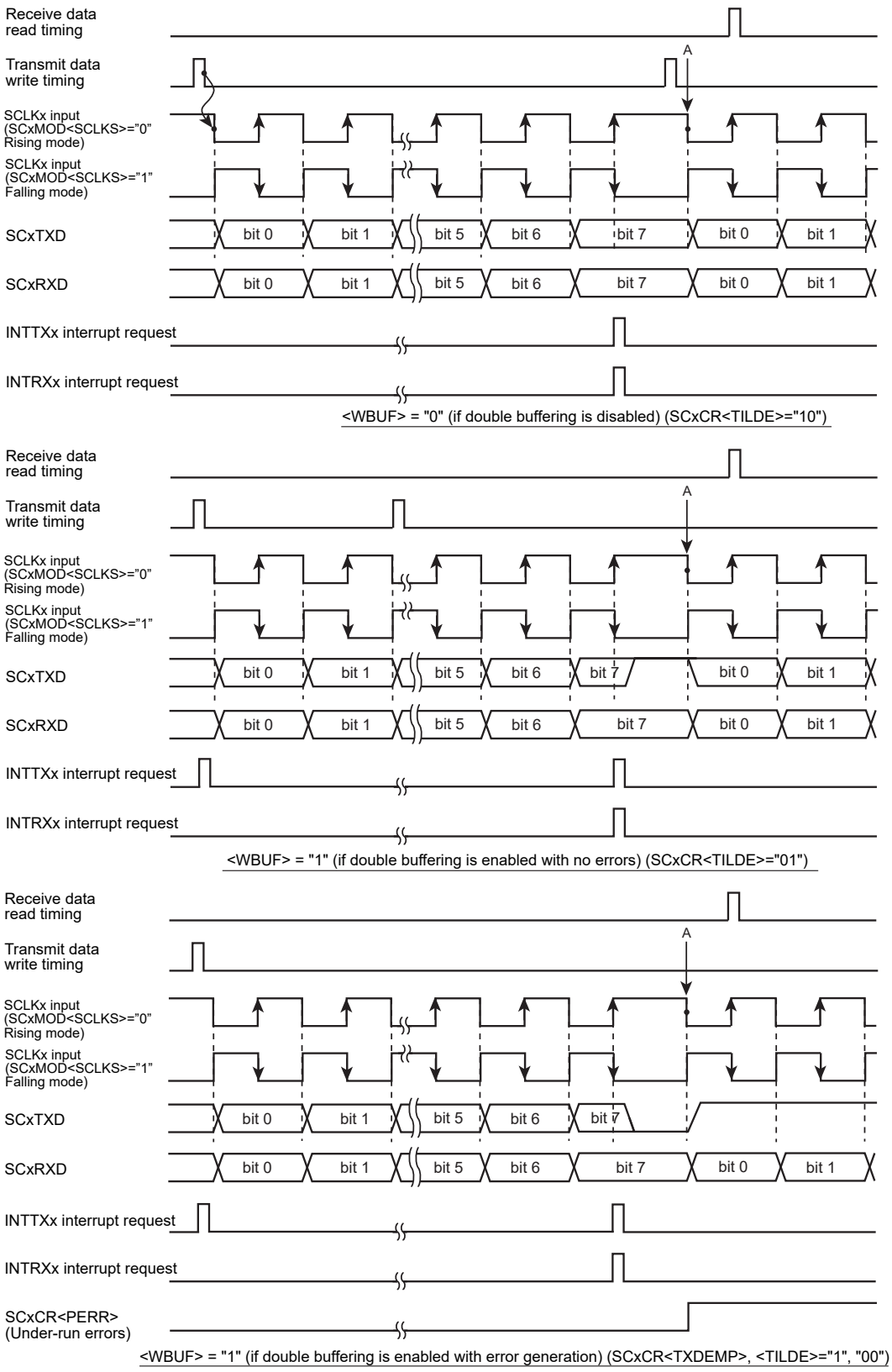


Figure 12-20 Transmit/Receive Operation in the I/O Interface Mode (Clock Input Mode)

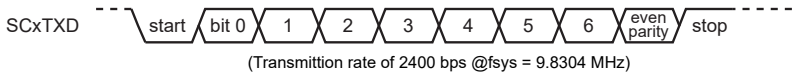
12.16.2 Mode 1 (7-bit UART mode)

The 7-bit UART mode is selected by setting SCxMOD<SM[1:0]> to "01".

In this mode, parity bits can be added to the transmit data stream; SCxCR<PE> controls the parity enable/disable setting.

When <PE> is set to "1" (enable), either even or odd parity may be selected using the SCxCR<EVEN>. The length of the stop bit can be specified using SCxMOD2<SBLEN>.

The following table shows the control register settings for transmitting in the following data format.



(Transmittion rate of 2400 bps @fsys = 9.8304 MHz)

Clocking conditions	system clock:	High-speed (fc)
	High-speed clock gear:	x 1 (fc)
	Prescaler clock:	fperiph/2 (fperiph = fsys)

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

x: don't care - : no change

12.16.3 Mode 2 (8-bit UART mode)

The 8-bit UART mode is selected by setting SCxMOD0<SM[1:0]> to "10". In this mode, parity bits can be added and parity enable/disable is controlled using SCxCR<PE>. If <PE> = "1" (enabled), either even or odd parity can be selected using SCxCR<EVEN>.

The control register settings for receiving data in the following format are as follows:

(Reception rate of 9600 bps @fc = 9.8304 MHz)

Clocking conditions	System clock:	High-speed (fc)
	High-speed clock gear:	x 1 (fc)
	Prescaler clock:	fperiph/2 (fperiph = fsys)

		7	6	5	4	3	2	1	0	
SCxMOD0	←	x	0	0	0	1	0	0	1	Set 8-bit UART mode
SCxCR	←	x	0	1	x	x	x	0	0	Odd parity enabled
SCxBRCR	←	0	0	0	1	0	1	0	0	Set 9600bps
SCxMOD0	←	-	-	1	-	-	-	-	-	Reception enabled

x: don't care - : no change

12.16.4 Mode 3 (9-bit UART mode)

The 9-bit UART mode is selected by setting SCxMOD0<SM[1:0]> to "11". In this mode, parity bits must be disabled (SCxCR<PE> = "0").

The most significant bit (9th bit) is written to SCxMOD0<TB8> for transmitting data. The data is stored in SCxCR<RB8> for receiving data.

When writing or reading data to/from the buffers, the most significant bit must be written or read first before writing or reading to/from SCxBUF.

The stop bit length can be specified using SCxMOD2<SBLEN>.

12.16.4.1 Wakeup function

In the 9-bit UART mode, slave controllers can be operated in the wake-up mode by setting SCxMOD0<WU> to "1".

In this case, the interrupt INTRXx will be generated only when SCxCR<RB8> is set to "1".

Note: The SCxTXD pin of the slave controller must be set to the open drain output mode using the PxOD.

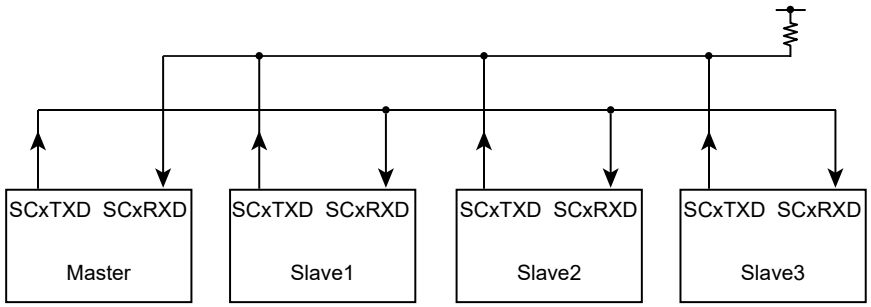


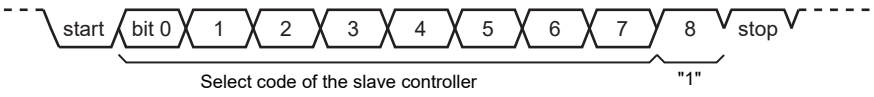
Figure 12-21 Serial Links to Use Wake-up Function

12.16.4.2 Protocol

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

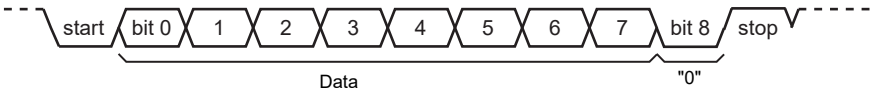
2. Set SCxMOD<WU> to "1" for the slave controllers to make them ready to receive data.

3. The master controller is to transmit a single frame of data that includes the slave controller select code (8 bits). In this, the most significant bit (bit 8) <TB8> must be set to "1".



4. Each slave controller receives the above data frame; if the code received matches with the controller's own select code, it clears the <WU> to "0".

5. The master controller transmits data to the designated slave controller (the controller of which SCxMOD<WU> is cleared to "0"). In this, the most significant bit (bit 8) <TB8> must be set to "0".



6. The slave controllers with the <WU> set to "1" ignore the receive data because the most significant bit (bit 8) <RB8> is set to "0" and thus no interrupt (INTRXx) is generated. Also, the slave controller with the <WU> set to "0" can transmit data to the master controller to inform that the data has been successfully received.

13. I2C Bus Interface

The TMPM037FWUG contains I2C Bus Interface with typical I2C bus standard (Philips specifications).

The main features are as follows.

- Allows selection between master and slave.
- Allows selection between transmission and reception.
- Support multiple masters (arbitration, clock synchronization recognition).
- Baud rate (Support standard mode and fast mode)
- Supports the addressing format of 7 bit only.
- Supports transfer data sizes of 1 to 8 bits.
- Provides one transfer (transmission or reception) complete interrupt (level sensitive).
- Enable or disable the interrupts.

This module also supports Toshiba's proprietary data format called " Free data format".

Table 13-1 I2C Bus Standard specifications

I2C bus feature	I2C Standard	TMPM037FWUG
STANDARD mode (up to 100KHz)	Required	Supported
FAST mode (up to 400KHz)	Required	Supported
Hs (High speed) mode (up to 3.4Mbps)	Required	Not Supported
7-bit addressing	Required	Supported
10-bit addressing	Required	Not Supported
START byte	Required	Supported
Noise canceller	Required	Supported (digital)
Slope control	Required	Not Supported
I/O at power off	Required	Supported
Schmidt (VIL/VHL)	VDD × 0.3 / VDD × 0.7	Not Supported
Output current at ¨VOL = 0.4V, VDD > 2V	3mA	Not Supported

13.1 Configuration

The configuration is shown in Figure 13-1.

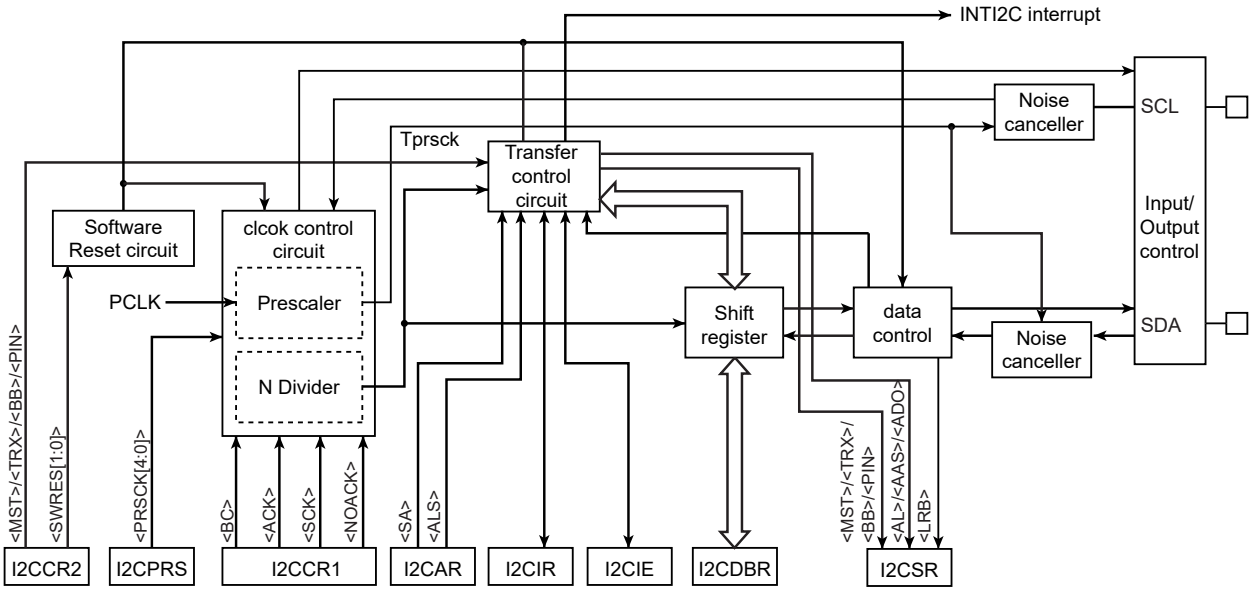


Figure 13-1 I2C Bus Block

13.2 I2C Bus mode

The I2C bus is connected to devices via the SDA and SCL pins and can communicate with multiple devices.

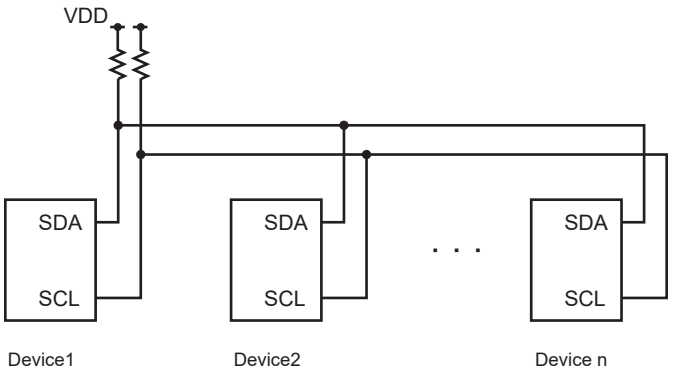


Figure 13-2

This module operates as a mater or slave device on the I2C bus.The master device drives the serial clock line (SCL) of the bus, send 8-bits address, and sends or receives data of 1 to 8bits.

The slave device send a 8-bits addresses and sends or receives serial data of 1 to 8 bits in synchronization with serial clock on the bus.

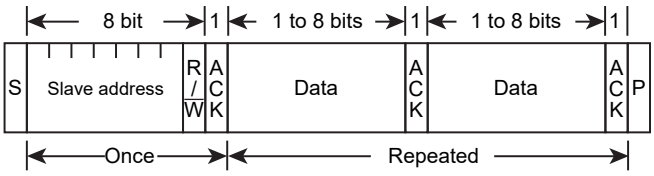
The device that operates as a receiver can output an acknowledge signal after reception of serial data and the device the operates as a transmitter can receive that acknowledge signal, regardless of whether the device is a master or slave. The master device can output a clock for the acknowledge signal.

In multimaster mode in which multiple masters exist on the same bus, serial clock synchronization and arbitration lost to maintain consistency of serial data are supported.

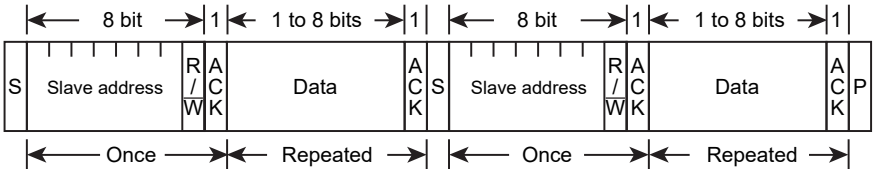
13.2.1 I2C Bus Mode Data Format

Figure 13-3 shows the data formats used in the I2C bus mode.

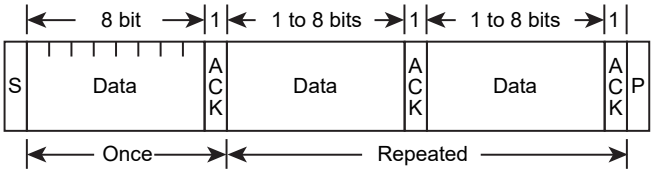
(a) Addressing format



(b) Addressing format (with repeated start condition)



(c) Free data format (master-transmitter to slave-receiver)



Note) S : Start condition
R/W : Direction bit
ACK : Acknowledge bit
P : Stop condition

Figure 13-3 I2C Bus Mode Data Formats

13.3 Register

13.3.1 Registers for each channel

The following registers table and address of the I2C bus interface.

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

Register name		Address(Base+)
Control register 1	I2CxCR1	0x0000
Data buffer register	I2CxDBR	0x0004
I2C bus address register	I2CxAR	0x0008
Control register 2	I2CxCR2 (writing)	0x000C
Status register	I2CxSR (reading)	
Prescaler Clock setting register	I2CxPRS	0x0010
Interrupt Enable Register	I2CxIE	0x0014
interrupt Register	I2CxIR	0x0018

Note: These registers can be read or written by an only word access.

13.3.2 I2CxCR1(Control register 1)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-5	BC[2:0]	R/W	Select the number of bits per transfer (Note 1)

<BC>	When <ACK> = 0		When <ACK> = 1	
	Number of clock cycles	Data length	Number of clock cycles	Data length
000	8	8	9	8
001	1	1	2	1
010	2	2	3	2
011	3	3	4	3
100	4	4	5	4
101	5	5	6	5
110	6	6	7	6
111	7	7	8	7

<table>
<tr><th>Bit</th><th>Bit Symbol</th><th>Type</th><th>Function</th></tr>
<tr><td>4</td><td>ACK</td><td>R/W</td><td>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.</td></tr>
<tr><td>3</td><td>NOACK</td><td>R/W</td><td>Slave address match detection and general call detection.
0:the slave address match or general call detection are enabled.
1:the slave address match or general call detection are disabled.
When I2CxAR<ALS>="1", this bit has no meaning.</td></tr>
<tr><td>2-0</td><td>SCK[2:0]</td><td>R/W</td><td>Select internal SCL output clock frequency (Note 2).
<table>
<tr><td>000</td><td>n = 0</td><td>000</td><td>n = 4</td></tr>
<tr><td>001</td><td>n = 1</td><td>001</td><td>n = 5</td></tr>
<tr><td>010</td><td>n = 2</td><td>010</td><td>n = 6</td></tr>
<tr><td>011</td><td>n = 3</td><td>011</td><td>n = 7</td></tr>
</table></td></tr>
</table>

Note 1: Writing to this register must be done before a start condition is generated or after a stop condition is generated or between the instant when an address or data transfer interrupt occurs and the instant when the internal iterrupt is released. Do not write to this register during address or data transfer.

Note 2: For details on the SCL line clock frequency, refer to "13.4.1 Serial Clock".

Note 3: The initial value for selecting a frequency is <SCK[2:0]>=000 and is independent of the read initial value.

13.3.3 I2CxDBR (Serial bus interface data buffer register)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-0	DB[7:0]	R (Receive)	Receive data
		W (Transmit)	Transmit data The transmission data must be written in to the register from the MSB (bit 7). The received data is stored in the LSB.

When the master needs to transmit a slave address, the transfer target address is written to I2CxDBR<DB [7:1]> and the transfer direction is specified in I2CxDBR<DB[0]> as follows:

<DB[0]>=0:Master(transmission) →Slave/reception

<DB[0]>=1:Master(reception) ←Slave/transmission

When all the bits in the I2CxDBR register are written as "0",a general call can be sent out on the bus.

In both transmission and reception modes, a write to the I2CxDBR register or read from the I2CxDBR register release the internal interrupt after the current transfer and initiates the next transfer.

I2CxDBR is provided as a transmit/receive buffer, it should be used as a dedicated transmit buffer in transmit mode and as a dedicated receive buffer in receive mode. This register in transmit mode and as a dedicated receive buffer in receive mode. this register should be accessed on a transfer -by - transfer basis.

Note: This register are initialized by Hardware RESET only. Software RESET can not initialize and are keeping the last data.

13.3.4 I2CxAR (I2Cbus address register)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7-1	SA[6:0]	R/W	Set the slave address when the I2C acts as a slave device.
0	ALS	R/W	Specify address recognition mode. 0: Recognize its slave address. 1: Do not recognize its slave address (free-data format).

Note 1: Please set the bit 0 <ALS> of I2C bus address register I2CxAR to "0", except when you use a free data format. It operates as a free data format when setting it to "1". Selecting the master fixes to transmission. Selecting the slave fixes to reception.

Note 2: Do not set I2CxAR to "0x00" in slave mode. (If I2CxAR is set to "0x00", it's recognized that the slave address matches the START byte ("0x01") of the I2C standard received in slave mode.)

13.3.5 I2CxCR2(Control register 2)

This register serves as I2CxSR register by reading it.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MST	TRX	BB	PIN	I2CM	-	SWRES	
After reset	0	0	0	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	MST	W	Select master/slave 0: Slave mode 1: Master mode
6	TRX	W	Select transmit/ receive 0: Receive 1: Transmit
5	BB	W	Start/stop condition generation 0: Stop condition generated 1: Start condition generated
4	PIN	W	Clear INTI2CIx interrupt request 0: − 1: Clear interrupt request
3	I2CM	W	I2C operation control 0: Disable 1: Enable
2	-	R	Read as 0.
1-0	SWRES[1:0]	W	Software reset generation Write "10" followed by "01" to generate a reset. For detail, refer to "13.4.11 Software Reset".

Note 1: Make sure that modes are not changed during a communication session. Ensure that the bus is free before switching the operating mode to the port mode. Ensure that the port is at the "High" level before switching the operating mode from the port mode to the I2C bus mode.

Note 2: The I2CxCR2<I2CM> bit cannot be cleared to 0 to disable I2C operation while transfer operation is being performed. before clearing this bit, mask sure that transfer operation is completely stopped by reading the status register.

13.3.6 I2CxSR (Status Register)

This register serves as I2CxCR2 by writing to it.

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	MST	TRX	BB	PIN	AL	AAS	AD0	LRB
After reset	0	0	0	1	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as 0.
7	MST	R	Master/slave selection monitor 0: Slave mode 1: Master mode
6	TRX	R	Transmit/receive selection monitor 0: Receive 1: Transmit
5	BB	R	I2C bus state monitor 0: Free 1: Busy
4	PIN	R	INTI2Cx interrupt request monitor 0:Interrupt request generated 1: Interrupt request cleared
3	AL	R	Arbitration lost detection 0: − 1:Detected
2	AAS	R	Slave address match detection 0: − 1: Detected (This bit is set when the general call is detected as well.)
1	AD0	R	General call detection 0: − 1:Detected
0	LRB	R	Last received bit monitor 0:Last received bit "0" 1:Last received bit "1"

13.3.7 I2CxPRS(Prescaler Clock setting register)

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

Bit	Bit Symbol	Type	Function
31-5	-	R	Read as 0.
4-0	PRSCK	R/W	Prescaler clock frequency for generating the Serial clock 00000: P = divided by 32 00001: P = divided by 1 - - - - - - - - - - 11111: P = divided by 31

Note: Refer to Section "13.3.2 I2CxCR1(Control register 1)", "13.4.1 Serial Clock".

13.3.8 I2CxIE(Interrupt Enable register)

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

Bit	Bit Symbol	Type	Function
31-1	-	R	Read as 0.
0	IE	R/W	I2C interrupt enable or disable setting. 0: Disable 1: Enable

13.3.9 I2CxIR(Interruppt register)

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

Bit	Bit Symbol	Type	Function
31-1	-	R	Read as 0.
0	ISIC	R	I2C interrupt status. 0: not interrupt 1: interrupt generated. This bit indicates the I2C interrupt status prior to masking by I2CxIE<IE>
		W	Clear the I2C interrupt. 0: Invalid 1: Clear the I2C interrupt. Writing "1" to this bit clears the I2C interrupt output(INTI2Cx). Writing "0" is invalid.

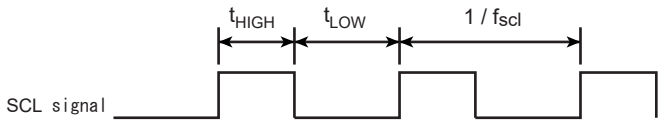
13.4 Control in the I2C Bus Mode

13.4.1 Serial Clock

13.4.1.1 Clock source

I2CxCR1<SCK[2:0]> is used to set the high and low periods of the serial clock to be output in master mode.

<SCK[2:0]>	t_{HIGH} (i / T_{prsck})	t_{LOW} (j / T_{prsck})
	i	j
000	8	12
001	10	14
010	14	18
011	22	26
100	38	42
101	70	74
110	134	138
111	262	266



t_{LOW} = i / Tprsck
t_{HIGH} = j / Tprsck
fscl = 1 / (t_{LOW} + t_{HIGH})

Figure 13-4 Clock source

Note: The tHIGH period may differ from the specified value if the rising edge becomes blunt depending on the combination of bus load capacitance and pull-up register. If the clock synchronization function for synchronizing clocks from multiple clocks is used, the actual clock period may differ from the specified setting.

In master mode, the hold time when a start conditions generated and the setup time when a stop condition is generated are defined as t_{HIGH}[S].

When I2CxCR2<PIN> is set to "1" in slave mode, the time to the release of SCLx is defined as tLOW [S].

In both master and slave modes, the high level period must be 4/Tprsck[s] or longer and the low level period must be 5/Tprsck[s] or longer for externally input serial clocks, regardless of the I2CxCR1<SCK>. setting.

The serial clock rete to be output from the master is set through I2CxCR1<SCK[2:0]> and the I2CxPRS<PRSCK[4:0]>. The prescaler clock which is divided according to I2CxPRS<PRSK[4:0]> is used as the reference clock for generating the serial clock. The prescaler clock is further divided according to I2CxCR1<SCk[2:0]> and used as the serial clock. The default setting of the prescaler clock is divide by 1(=fsys).

<Serial transfer rate>

The serial clock rate (fSCL) is determined by prescaler setting value "p" (I2CxPRS<PRSCK[4:0]>,p=1 to 32) and serial clock setting value "n" (I2CxCR1<SCK[2:0]>,n=0 to 7) based on the operating frequency (fsys) as follows:

$$\text{Serial clock rate : } f_{scl}\ (\text{kHz}) = \frac{\text{fsys(MHz)}}{p\times(2^{n+2}+16)} \times 1000$$

p: prescaler setting I2CxPRS<PRSCK[4:0]>, 1 to 32
n: serial clock setting I2CxCR1<SCK[2:0]>, 0 to 7

The allowed range of prescaler setting value "p" (I2CxPRS<PRSCK[4:0]>) varies depending on the operating frequency (fsys) and must satisfy the following condition.

50ns < Prescaler clock width:Tprsck (ns) ≤ 150ns

Note: Setting the prescaler clock width out of this range is prohibited in both master and slave modes.

The serial clock rate may not be constant due to the clock synchronization function.

n: <SCK[2:0]>			p: <PRSCK[4:0]>		
			00001 (divide by1)	01101 (divide by 13)	00000 (divide by 32)
			Ratio to fsys		
0	0	0	20	260	640
0	0	1	24	312	768
0	1	0	32	416	1024
0	1	1	48	624	1536
1	0	0	80	1040	2560
1	0	1	144	1872	4608
1	1	0	272	3536	8704
1	1	1	528	6864	16896

Note: Writing to these bits must be done before a start condition is generated or after a stop condition is benerated.Writing to these bits during transfer will cause unexpected operation.

<Prescaler clock width (=noise cancellation width)

The prescaler clock width (Tprsck)(= noise cancellation width) is determined by prescaler setting value "p" (I2CxPRS<PRSCK[4:0]>,P=1 to 32) based on the operating frequency (fsys) as follows:

$$\text{Prescaler clock width : Tprsck (ns)} = \frac{1}{\text{fsys(MHz)}} \times 1000 \times p$$
(=Noise cancellation width)

13.4.1.2 Clock Synchronization

The I2C bus is driven by using the wired-AND connection due to its pin structure. The first master that pulls its clock line to the "Low" level overrides other masters producing the "High" level on their clock lines. This must be detected and responded by the masters producing the "High" level.

I2C has a clock synchronization function to ensure proper transfer operation even when multiple master exist on a bus.

For example, the clock synchronization procedure for a bus with two masters is shown below.

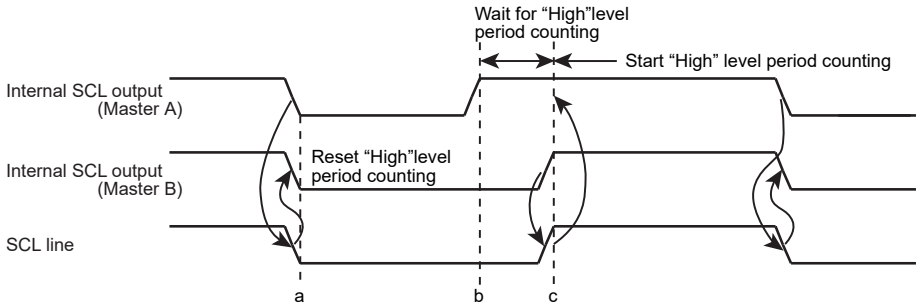


Figure 13-5 Example of Clock Synchronization

At the point a, Master A pulls its internal SCL output to the "Low" level, bringing the SCL bus line to the "Low" level. Master B detects this transition, resets its "High" level period counter, and pulls its internal SCL output level to the "Low" level.

Master A completes counting of its "Low" level period at the point b, and brings its internal SCL output to the "High" level. However, Master B still keeps the SCL bus line at the "Low" level, and Master A stops counting of its "High" level period counting.After Master A detects that Master B brings its internal SCL output to the "High" level and brings the SCL bus line to the "High" level at the point c, it starts counting of its "High" level period.

After that Master finishes counting the "High" level period, the Master pulls the SCL pin to "Low" and the SCL bus line becomes "Low".

This way, the clock on the bus is determined by the master with the shortest "High" level period and the master with the longest "Low" level period among those connected to the bus.

13.4.2 Selection for a Slave Address Match Detection or General Call Detection

For a slave device, the I2CxCR1<NOACK> is done to enable or disable for a slave address match detection and general call detection in slave mode.

when I2CxCR1<NOACK>=0,it is enable for a slave address match detection and general call detection.

If the received address matches its slave address specified at I2CxAR or is equal to the general-call address, the I2C pulls the SDA line to the "Low" level during the ninth clock and outputs an acknowledgment signal.

when I2CxCR1<NOACK>=1,it is disable for a slave address match detection and general call detection.

If the received address matches its slave address specified at I2CxAR or is equal to the general-call address, the I2C release the SDA line to the "High" level and outputs an non-acknowledgment signal.

The slave device ignores a slave address and general calls sent from the master and returns non-acknowledgment. the INTI2Cx interrupt request are not generated.

In master mode, the bit of I2CxCR1<NOACK> is ignored and has no effect on operation.

Note: if I2CxCR1<NOACK> is cleared to "0" during data transfer in slave mode, I2CxCR1<NOACK> remains "1" and an acknowledge signal is returned for the transferred data.

13.4.3 Setting the Acknowledgement Mode

Setting I2CxCR1<ACK> to "1" selects the acknowledge mode. When operating as a master, the I2C adds one clock for acknowledgment signal. In slave mode, the clock for acknowledgement signals is counted. In transmitter mode, the I2C releases the SDAx pin during clock cycle to receive acknowledgement signals from the receiver. In receiver mode, the I2C pulls the SDAx pin to the "Low" level during the clock cycle and generates acknowledgement signals. Also in slave mode, if a general-call address is received, the I2C pulls the SDAx pin to the "Low" level during the clock cycle and generates acknowledgement signals.

However, the second byte is necessary to be controlled by software to generate an ACK signal on the contents of the second byte.

By setting <ACK> to "0", the non-acknowledgment mode is activated. When operating as a master, the I2C does not generate clock for acknowledgement signals. In slave mode, the clock for acknowledgement signals is not counted.

13.4.4 Setting the Number of Bits per Transfer

I2CxCR1<BC[2:0]> specifies the number of bits of the next data to be transmitted or received.

Under the start condition, <BC[2:0]> is set to "000", causing a slave address and the direction bit to be transferred in a packet of eight bits. At other times, <BC[2:0]> keeps a previously programmed value.

Note: A slave address must be trasmitted / received with I2CxCR1<ACK> set to "1". if I2CxCR1<ACK> is cleared, the slave address match detection and direction bit detection cannot be performed properly.

13.4.5 Slave Addressing and Address Recognition Mode

Setting "0" to I2CxAR<ALS> and a slave address in I2CxAR<SA[6:0]> sets addressing format, and then the I2C recognizes a slave address transmitted by the master device and receives data in the addressing format.

If <ALS> is set to "1", the I2C does not recognize a slave address and receives data in the free data format. In the case of free data format, a slave address and a direction bit are not recognized; they are recognized as data immediately after generation of the start condition.

13.4.6 Configuring the I2C as a Master or a Slave

Setting I2CxCR2<MST> to "1" configures the I2C to operate as a master device.

Setting <MST> to "0" configures the I2C as a slave device. <MST> is cleared to "0" by the hardware when it detects the stop condition on the bus or the arbitration lost.

13.4.7 Configuring the I2C as a Transmitter or a Receiver

Setting I2CxCR2<TRX> to "1" configures the I2C as a transmitter. Setting <TRX> to "0" configures the I2C as a receiver.

At the slave mode:

- when data is transmitted in the addressing format.
- when the received slave address matches the value specified at I2CxCAR.
- when a general-call address is received (the eight bits following the start condition are all zeros.)

If the value of the direction bit (R/$\overline{W}$) is "1", <TRX> is set to "1" by the hardware. If the bit is "0", <TRX> is set to "0".

As a master device, the I2C receives acknowledgement from a slave device. If the direction bit of "1" is transmitted, <TRX> is set to "0" by the hardware. If the direction bit is "0", <TRX> changes to "1". If the I2C does not receive acknowledgement, <TRX> retains the previous value.

<TRX> is cleared to "0" by the hardware when it detects the stop condition on the bus or the arbitration lost.

Note: When I2CxCR1<NOACK>=1, the slave address detection and general call detection are disabled, and thus I2CxSR<TRX> remains unchanged.

Table 13-2 I2CxSR<TRX> operation

mode	direction bit	condition for state change	Changed <TRX> after state change
Slave mode	0	Received Slave address = I2CxAR<SA>	0
	1		1
Master mode	0	ACK received	1
	1		0

When I2C is used in free data format, the salve address and direction bit are not recognized and bits immediately following a start condition are handled as data. Therefore, <TRX> is not changed by the hardware.

13.4.8 Generating Start and Stop Conditions

When I2CxSR<BB> is "0", writing "1" to I2CxCR2<MST, TRX, BB, PIN> causes the I2C to start a sequence for generating the start condition and to output the slave address and the direction bit prospectively written in the data buffer register. <ACK> must be set to "1" in advance.

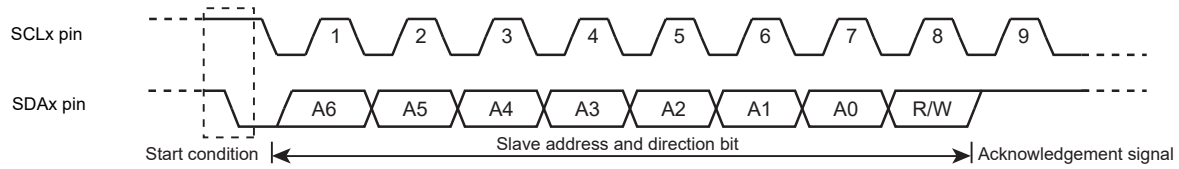


Figure 13-6 Generating the Start Condition and a Slave Address

When <BB> is "1", writing "1" to <MST, TRX, PIN> and "0" to <BB> causes the I2C to start a sequence for generating the stop condition on the bus. The contents of <MST, TRX, BB, PIN> should not be altered until the stop condition appears on the bus.

If SCL bus line is pulled "Low" by other devices when the stop condition is generated, the stop condition is generated after the SCL line is released.

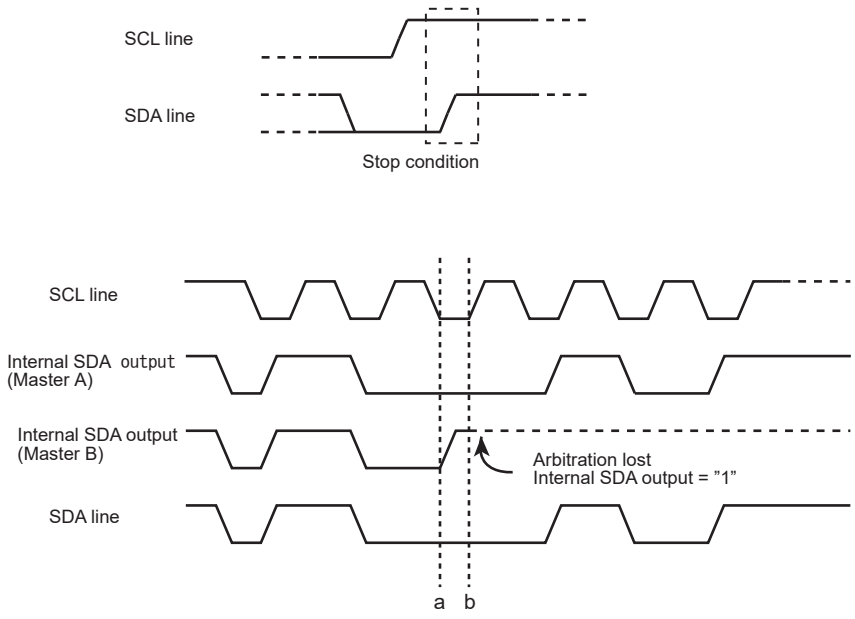


Figure 13-7 Generating the Stop Condition

I2CxSR<BB> can be read to check the bus state. <BB> is set to "1" when the start condition is detected on the bus (the bus is busy), and cleared to "0" when the stop condition is detected (the bus is free).

The following table shows typical setting examples according to the I2CxSR state.

Although the I2CxCR2<MST>,<TRX>,<BB>,<PIN> bits are given independent functions, they are used in typical combinations, as shown below, according to the I2CxSR setting.

I2CxSR			I2CxCR2				Operation
<MST>	<BB>	<PIN>	<MST>	<TRX>	<BB>	<PIN>	
0	0	1	0	0	0	0	Wait for a start condition as a slave.
			1	1	1	1	Generate a start condition.
1	1	0	1	1	0	1	Generate a stop condition.
			0	0	0	1	Release the internal interrupt for restart.

Note: When writing to these bits, do not change I2CxCR2<I2CM> by mistake.

13.4.9 Interrupt Service Request and Release

In master mode, a I2C bus interface request (INTI2Cx) is generated when the transfer of the number of clock cycles set by I2CxCR1<BC> and I2CxCR1<ACK> is completed.

In slave mode, INTI2Cx is generated under the following conditions.

- When I2CxCR1<NOACK> is "0",after output of the acknowledge signal which is generated when the received slave address matches the slave address set to I2CxAR<SA[6:0]>.
- When I2CxCR1<NOACK> is "0",after the acknowledge signal is generated when a general-call address is received.
- When the slave address matches or a data transfer is completed after receiving a general-call address.

When an interrupt request (INTI2Cx) is generated, I2CxCR2<PIN> is cleared to "0". While <PIN> is cleared to "0", the I2C pulls the SCL line to the "Low" level.

<PIN> is set to "1" when data is written to or read from I2CxDBR. It takes a period of t_{LOW} for the SCL line to be released after <PIN> is set to "1". When the program writes "1" to <PIN>, it is set to "1". However, writing "0" does not clear this bit to "0".

Note: When arbitration lost occurs while a slave address and direction bit are transferred in the master mode, <PIN> is cleared to "0" and INTI2Cx occurs. This does not relate to whether a slave address matches <SA>.

13.4.10 I2C Bus mode

When I2CxCR2<I2CM> is set to "1". I2C bus mode is selected. ensure that the I2C bus interface pins are at "High" level before setting <I2CM> to "1". Also, ensure that the bus is free before switching the operating mode to the port mode.

Note: When I2CxCR2<I2CM> = "0", no value can be written to bits in the I2CxCR2 register other than I2CxCR2<I2CM> bit. Before setting I2CxCR2, write "1" into I2CxCR2<I2CM> to select the I2C bus mode.

13.4.11 Software Reset

If the I2C bus interface circuit locks up due to external noise, it can be initialized by using a software reset.

If the I2C bus interface circuit locks up due to external noise, it can be initialized by using a software reset.Writing "10" followed by "01" into I2CxCR2<SWRES[1:0]> generates a reset signal that initializes the I2C bus interface circuit. After a reset, all control registers and status flags except I2CxCR2<I2CM> and I2CxDBR register are initialized to their reset values. When the I2C bus interface is initialized, <SWRES[1:0] > is automatically cleared to "0".

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

13.4.12 Arbitration Lost Detection Monitor

The I2C bus has the multi-master capability (there are two or more masters on a bus), and requires the bus arbitration procedure to ensure correct data transfer.

A master that attempts to generate the start condition while the bus is busy loses bus arbitration, with no start condition occurring on the SDA and SCL lines. The I2C-bus arbitration takes place on the SDA line. When a start condition is output under the bus busy state, a start condition is not output to SCL or SDA line, thus arbitration lost occurs.

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

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

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

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

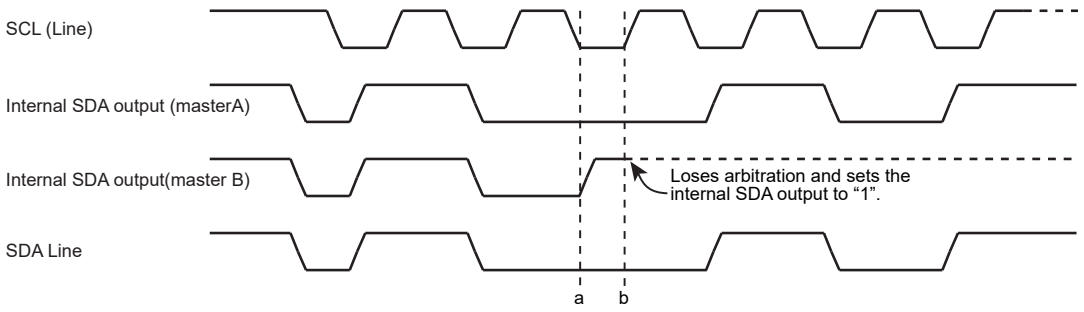


Figure 13-8 Lost Arbitration

A master compares the SDA bus line level and the internal SDA output level at the rising of the SCL line. If there is a difference between these two values, Arbitration Lost occurs and I2CxSR<AL> is set to "1".

When <AL> is set to "1", I2CxSR<MST, TRX> are cleared to "0", causing the I2C to operate as a slave receiver. Therefore, the serial bus interface circuit stops the clock output during data transfer after <AL> is set to "1".

<AL> is cleared to "0" when data is written to or read from I2CxDBR or data is written to I2CxCR2.

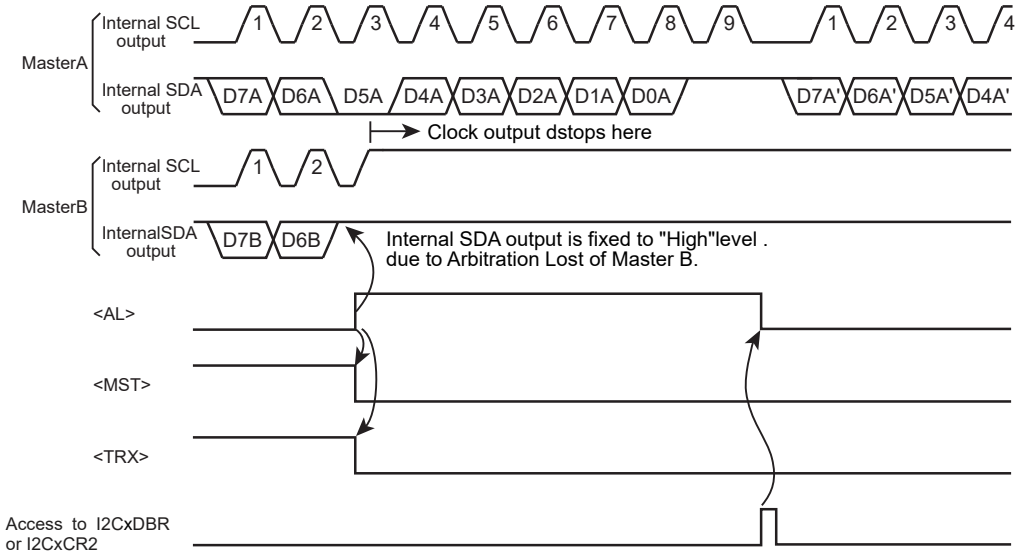


Figure 13-9 Example of Master B Lost Arbitration (D7A = D7B, D6A = D6B)

13.4.13 Slave Address Match Detection Monitor

When the I2C operates as a slave device in the address recognition mode (I2CxAR<ALS>="0"), I2CxSR<AAS> is set to "1" on receiving the general-call address or the slave address that matches the value specified at I2CxAR.

Clearing I2CxCR1<NOACK> to "0" enable the slave address match detection.When a general call is received or the slave address sent form the master matches the slave address set in I2CxAR<SA>,I2CxSR<AAS> is set to "1".

Setting I2CxCR1<NOACK> to "1" disable the slave address match detection. Even if a general call is received or the slave address sent form the master matches the slave address set in I2CxAR<SA>,I2CxSR<AAS> remains "0".

When Free data format mode <ALS> is set to "1", <AAS> is set to "1" when the first data word has been received. <AAS> is cleared to "0" when data is written to or read from I2CxDBR.

13.4.14 General-call Detection Monitor

When the I2C operates as a slave device, I2CxSR<AD0> is set to "1" when it receives the general-call address (i.e.; the eight bits following the start condition are all zeros).

<AD0> is cleared to "0" when the start or stop condition is detected on the bus.

13.4.15 Last Received Bit Monitor

I2CxSR<LRB> is set to the SDA line value that was read at the rising of the SCL line.

In the acknowledgment mode, reading I2CxSR<LRB> immediately after generation of the INTI2Cx interrupt request causes ACK signal to be read.

13.4.16 Data Buffer Register (I2CxDBR)

Reading or writing I2CxDBR initiates reading received data or writing transmitted data.

When the I2C is acting as a master, setting a slave address and a direction bit to this register generates the start condition.

13.5 Data Transfer Procedure in the I2C Bus Mode

13.5.1 Device Initialization

After ensuring that the SDA and SCL pins are high (Bus free), set I2CxCR2<I2CM> to "1" for enable the I2C.

Next, write "1" to I2CxCR1<ACK]> and "0" to I2CxCR1<NOACK>.Writing "000" to I2CxCR1<BC[2:0] > at the time.

These setting enable acknowledge operation, slave address match detection and general call detection and set the data length to 8 bits.

and set "tHIGH" and "tLOW" in I2CxCR1<SCK>.

then, program I2CxAR by specifying a slave address at <SA[6:0]> and an address recognition mode at <ALS>. (<ALS> must be cleared to "0" when using the addressing format).

To configure the I2C Bus Interface as a slave receiver, ensure that the I2C bus interface pin is at "High" first. Finally, write "0" to I2CxCR2<MST, TRX, BB>, "1" to I2CxCR2<PIN>, "00" to I2CxCR2<SWRES[1:0] > to configure the device as a slave receiver.

Note: Initialization of the serial bus interface circuit must be completed within a period that any device does not generate start condition after all devices connected to the bus were initialized. If this rule is not followed, data may not be received correctly because other devices may start transfer before the initialization of the serial bus interface circuit is completed.

		7	6	5	4	3	2	1	0	
I2CxCR2	←	0	0	0	1	1	0	0	0	enable I2C
I2CxCR1	←	0	0	0	1	0	1	1	0	Specifies ACK and SCL clock.
I2CxAR	←	X	X	X	X	X	X	X	X	Specifies a slave address and an address recognition mode.
I2CxCR2	←	0	0	0	1	1	0	0	0	Configures the I2C as a slave receiver.

Note: X; Don’t care

13.5.2 Generating the Start Condition and a Slave Address

13.5.2.1 Master mode

In the master mode, the following steps are required to generate the start condition and a slave address.

First, ensure that the bus is free (<BB> = "0"). Then, write "1" to I2CxCR1<ACK> to select the acknowledgment mode. Write to I2CxDBR a slave address and a direction bit to be transmitted.

When <BB> = "0", writing "1111" to I2CxCR2<MST, TRX, BB, PIN> generates the start condition on the bus. Following the start condition, the I2C generates nine clocks from the SCL pin. The I2C outputs the slave address and the direction bit specified at I2CxDBR with the first eight clocks, and releases the SDA line in the ninth clock to receive an acknowledgment signal from the slave device.

The INTI2Cx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to "0". In the master mode, the I2C holds the SCL line at the "Low" level while <PIN> is = "0".<TRX> changes its value according to the transmitted direction bit at generation of the INTI2Cx interrupt request, provided that an acknowledgment signal has been returned from the slave device.

Note 1: To output slave address, check with software that the bus is free before writing to I2CxDBR. If this rule is not followed, data being output on the bus may get ruined.

Note 2: When other master device may transfer while the device generate a start condition after writing the slave address.Therefor,To generate a start condition again after confirmed the bus is free by your program. the above procedure is done within 98.0μs (Minimum transmitting time on the STANDARD mode of I2C standard) or 23.7μs (Minimum transmitting time on the FAST mode of I2C standard) after writing the slave address.

Settings in main routine

```
                        7  6  5  4  3  2  1  0
Reg.            ←   I2CxSR
Reg.            ←   Reg. AND 0x20
if Reg.         ≠   0x00                          Ensures that the bus is free.
Then
I2CxCR1         ←   X  X  X  1  0  X  X  X        Selects the acknowledgement mode.
I2CxDBR         ←   X  X  X  X  X  X  X  X        Specifies the desired slave address and direction.
I2CxCR2         ←   1  1  1  1  1  0  0  0        Generates the start condition.
Example of INTI2C0 interrupt routine
    Clears the interrupt request.
    Processing
    End of interrupt
```

13.5.2.2 Slave mode

In the slave mode, the I2C receives the start condition and a slave address.

After receiving the start condition from the master device, the I2C receives a slave address and a direction bit from the master device during the first eight clocks on the SCL line.

If the received address matches its slave address specified at I2CxAR or is equal to the general-call address, the I2C pulls the SDA line to the "Low" level during the ninth clock and outputs an acknowledgment signal.

The INTI2Cx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to "0". In the slave mode, the I2C holds the SCL line at the "Low" level while <PIN> is "0".

Note: The I2C bus with DMA transfer is possible as following condition.

- Master and slave, one to one communication
- Sequential and continuous transmission or received

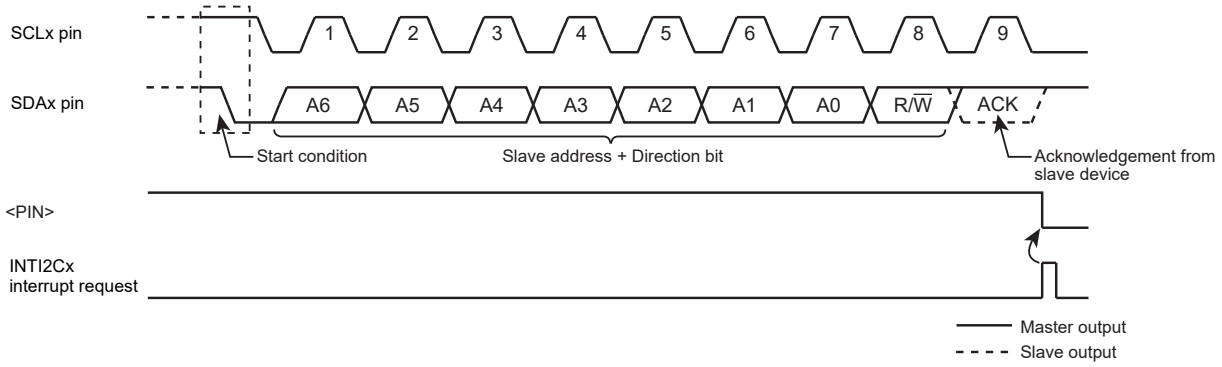


Figure 13-10 Generation of the Start Condition and a Slave Address

13.5.3 Transferring a Data Word

At the end of a data word transfer, the INTI2Cx interrupt is generated to test I2CxSR<MST> to determine whether the I2C is in the master or slave mode.

13.5.3.1 Master mode (<MST> = "1")

Test <TRX> to determine whether the I2C is configured as a transmitter or a receiver.

(1) Transmitter mode (<TRX> = "1")

Test <LRB>. If <LRB> is "1", that means the receiver requires no further data.

The master then generates the stop condition as described later to stop transmission.

If <LRB> is "0", that means the receiver requires further data. If the next data to be transmitted has eight bits, the data is written into I2CxDBR. If the data has different length, <BC[2:0]> and <ACK> are programmed and the transmit data is written into I2CxDBR. Writing the data makes <PIN> to "1", causing the SCL pin to generate a serial clock for transferring a next data word, and the SDA pin to transfer the data word.

After the transfer is completed, the INTI2Cx interrupt request is generated, <PIN> is cleared to "0", and the SCL pin is pulled to the "Low" level.

To transmit more data words, test <LRB> again and repeat the above procedure.

INTI2Cx interrupt

if MST = 0

Then go to the slave-mode processing.

if TRX = 0

Then go to the receiver-mode processing.

if LRB = 0

Then go to processing for generating the stop condition.

I2CxCR1	←	X	X	X	X	0	X	X	X	Specifies the number of bits to be transmitted and specify whether ACK is required.
I2CxDBR	←	X	X	X	X	X	X	X	X	Writes the transmit data.

End of interrupt processing.

Note: X; Don't care

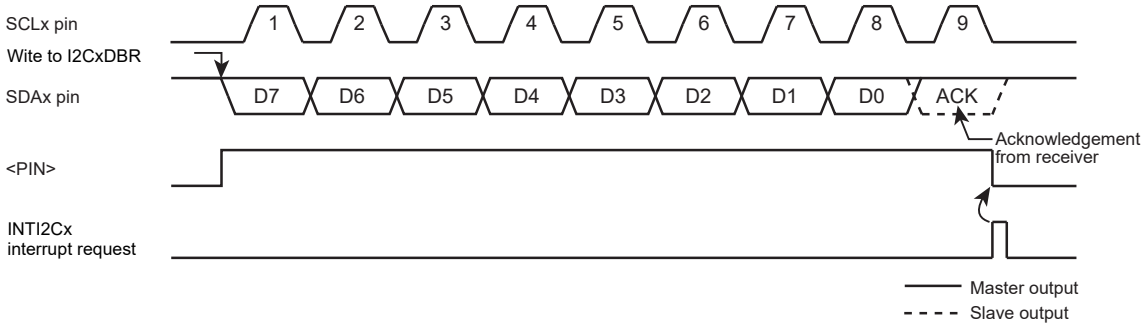


Figure 13-11 <BC[2:0]>= "000",<ACK>= "1" (Transmitter Mode)

(2) Receiver mode (<TRX> = "0")

If the next data to be transmitted has eight bits, the transmit data is written into I2CxDBR.

If the data has different length, <BC[2:0]> and <ACK> are programmed and the received data is read from I2CxDBR to release the SCL line. (The data read immediately after transmission of a slave address is undefined.)On reading the data, <PIN> is set to "1", and the serial clock is output to the SCL pin to transfer the next data word.In the last bit, when the acknowledgment signal becomes the "Low" level, "0" is output to the SDA pin.

After that, the INTI2Cx interrupt request is generated, and <PIN> is cleared to "0", pulling the SCL pin to the "Low" level.Each time the received data is read from I2CxDBR, one-word transfer clock and an acknowledgement signal are output.

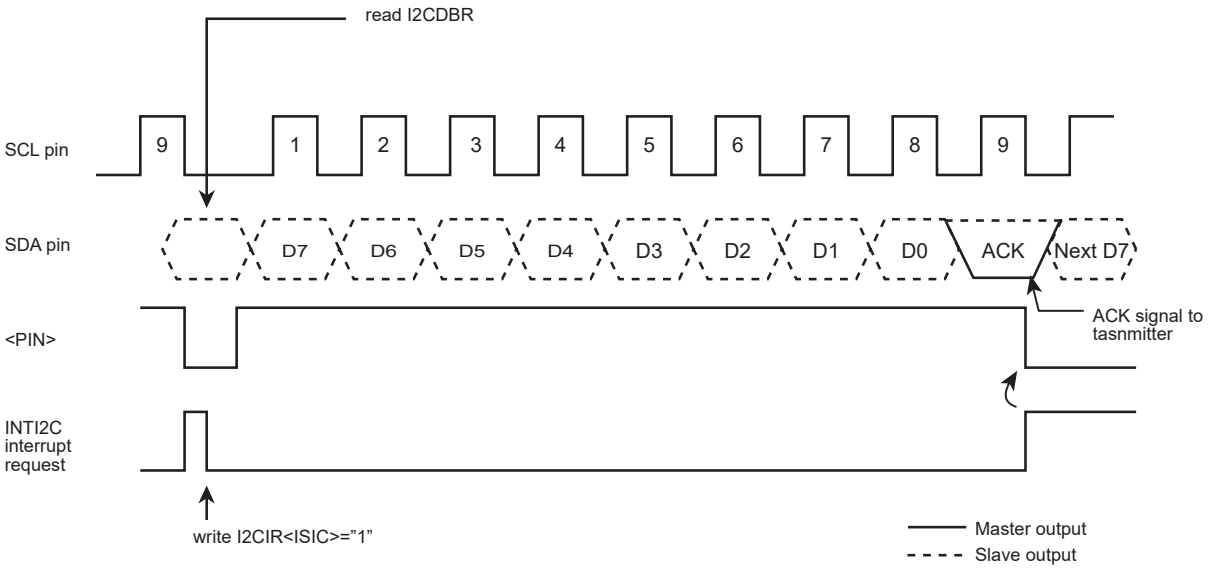


Figure 13-12 <BC[2:0]>= "000",<ACK>= "1" (Receiver Mode)

To terminate the data transmission from the transmitter, <ACK> must be cleared to "0" immediately before reading the data word second to last.

This disables generation of an acknowledgment clock for the last data word.

When the transfer is completed, an interrupt request is generated. After the interrupt processing, <BC[2:0]> must be set to "001" and the data must be read so that a clock is generated for 1-bit transfer.

At this time, the master receiver holds the SDA bus line at the "High" level, which signals the end of transfer to the transmitter as an acknowledgment signal.

In the interrupt processing for terminating the reception of 1-bit data, the stop condition is generated to terminate the data transfer.

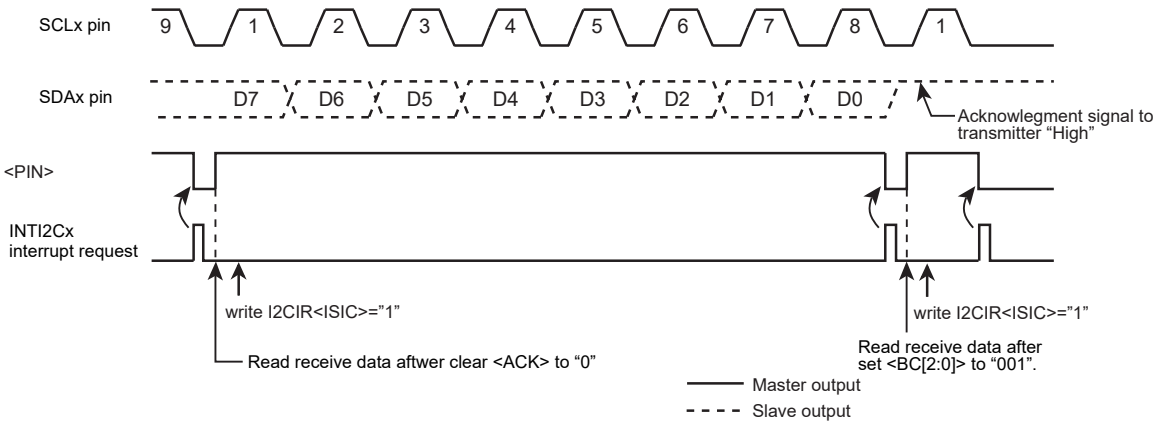


Figure 13-13 Terminating Data Transmission in the Master Receiver Mode

Example: When receiving N data word

INTI2Cx interrupt (after data transmission)

		7	6	5	4	3	2	1	0	
I2CxCR1	←	X	X	X	X	0	X	X	X	Sets the number of bits of data to be received and specify whether ACK is required.
Reg.	←	I2CxDBR								Reads dummy data.
End of interrupt										

INTI2Cx interrupt (first to (N-2)th data reception)

		7	6	5	4	3	2	1	0	
Reg.	←	I2CxDBR								Reads the first to (N-2)th data words.
End of interrupt										

INTI2Cx interrupt ((N-1)th data reception)

		7	6	5	4	3	2	1	0	
I2CxCR1	←	X	X	X	0	0	X	X	X	Disables generation of acknowledgement clock.
Reg.	←	I2CxDBR								Reads the (N-1)th data word.
End of interrupt										

INTI2Cx interrupt (Nth data reception)

		7	6	5	4	3	2	1	0	
I2CxCR1	←	0	0	1	0	0	X	X	X	Disables generation of acknowledgement clock.
Reg.	←	I2CxDBR								Reads the Nth data word.
End of interrupt										

INTI2Cx interrupt (after completing data reception)

Processing to generate the stop condition.										Terminates the data transmission.
End of interrupt										

Note: X; Don’t care

13.5.3.2 Slave mode (<MST> = "0")

In the slave mode, the I2C generates the INTI2Cx interrupt request on following status:

1) when I2CxCR1<NOACK> is "0", the I2C has received a general-call address.

2) when I2CxCR1<NOACK> is "0", the received slave address matches I2CxAR<SA> address.

3) when a data transfer has been completed in response to a received slave address matching or a general-call address.

Also, if the I2C detects Arbitration Lost in the master mode, it switches to the slave mode.

Upon the completion of data word transfer in which Arbitration Lost is detected, the INTI2Cx interrupt request is generated, <PIN> is cleared to "0", and the SCL pin is pulled to the "Low" level.

When data is written to or read from I2CxDBR or when <PIN> is set to "1", the SCLx pin is released after a period of t_{LOW}.

In the slave mode, the normal slave mode processing or the processing as a result of Arbitration Lost is carried out.

I2CxSR<AL>, <TRX>, <AAS> and <AD0> are tested to determine the processing required.

"Table 13-3 Processing in Slave Mode"shows the slave mode states and required processing.

Example: When the received slave address matches the I2C's own address and the direction bit is "1" in the slave receiver mode.

INTI2Cx interrupt

```
if TRX = 0
Then go to other processing.
if AL = 0
Then go to other processing.
if AAS = 0
Then go to other processing.
I2CxCR1    ←  X  X  X  1  0  X  X  X    Sets the number of bits to be transmitted.
I2CxDBR    ←  X  X  X  X  X  X  X  X    Sets the transmit data.
```

Note: X; Don't care

Table 13-3 Processing in Slave Mode

<TRX>	<AL>	<AAS>	<AD0>	State	Processing
1	1	1	0	Arbitration Lost is detected while the slave address was being transmitted and the I2C received a slave address with the direction bit "1" transmitted by another master.	Set the number of bits in a data word to <BC[2:0]> and write the transmit data into I2CxDBR.
	0	1	0	In the slave receiver mode, the I2C received a slave address with the direction bit "1" transmitted by the master.	
		0	0	In the slave transmitter mode, the I2C has completed a transmission of one data word.	Test LRB. If it has been set to "1", that means the receiver does not require further data. Set <PIN> to 1 and reset <TRX> to 0 to release the bus. If <LRB> has been reset to "0", that means the receiver requires further data. Set the number of bits in the data word to <BC[2:0]> and write the transmit data to the I2CxDBR.
0	1	1	1/0	Arbitration Lost is detected while a slave address is being transmitted, and the I2C receives either a slave address with the direction bit "0" or a general-call address transmitted by another master.	Read the I2CxDBR (a dummy read) to set <PIN> to 1, or write "1" to <PIN>.
		0	0	Arbitration Lost is detected while a slave address or a data word is being transmitted, and the transfer is terminated.	
	0	1	1/0	In the slave receiver mode, the I2C received either a slave address with the direction bit "0" or a general-call address transmitted by the master.	
		0	1/0	In the slave receiver mode, the I2C has completed a reception of a data word.	Set the number of bits in the data word to <BC[2:0]> and read the received data from I2CxDBR.

Note: In salve mode, if I2CxAR<SA[6:0]> is set to "0x00" a START byte (0x01) of the I2C bus standard is received, a slave address match is detected and I2CxSR<TRX> is set to "1". Not set I2CxAR<SA[6:0]> to "0x00".

13.5.4 Generating the Stop Condition

When I2CxSR<BB> is "1", writing "1" to I2CxCR2<MST, TRX, PIN> and "0" to <BB> causes the I2C to start a sequence for generating the stop condition on the bus.

Do not alter the contents of <MST, TRX, BB, PIN> until the stop condition appears on the bus.

If another device is holding down the SCL bus line, the I2C waits until the SCL line is released.

After that, the SDA pin goes "High", and then causing the stop condition to be generated for a "t_{HIGH}".

		7	6	5	4	3	2	1	0	
I2CxCR2	←	1	1	0	1	1	0	0	0	Generates the stop condition.

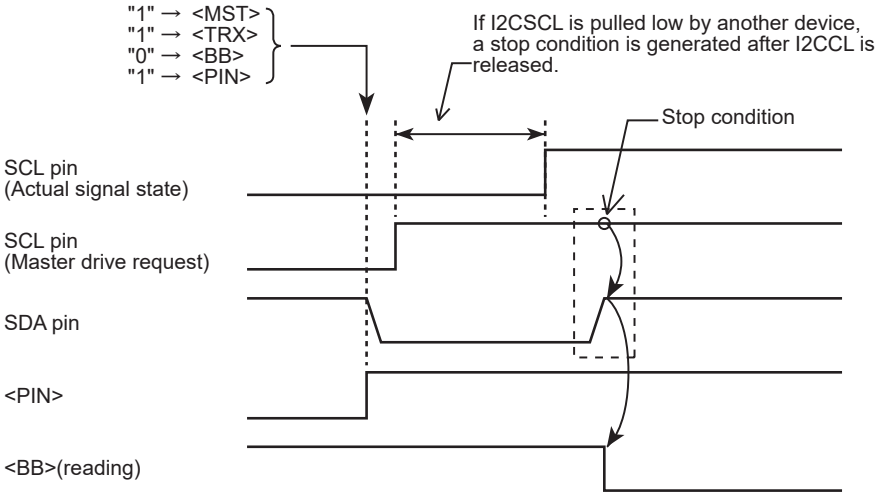


Figure 13-14 Generating the Stop Condition

13.5.5 Restart Procedure

Restart is used when a master device changes the data transfer direction without terminating the transfer to a slave device. The procedure of generating a restart in the master mode is described below.

First, write I2CxCR2<MST, TRX, BB> to "0" and write "1" to <PIN> to release the bus. At this time, the SDAx pin is held at the "High" level and the SCLx pin is released. Because no stop condition is generated on the bus, other devices recognize that the bus is busy.

Then, test I2CxSR<BB> and wait until it becomes "0" to ensure that the SCLx pin is released.

Next, test <LRB> and wait until it becomes "1" to ensure that no other device is pulling the SCLx bus line to the "Low" level.

Once the bus is determined to be free by following the above procedures, follow the procedures described in "13.5.2 Generating the Start Condition and a Slave Address" to generate the start condition.

To satisfy the setup time of restart, at least 4.7μs wait period (in the standard mode) must be created by the software after the bus is determined to be free.

Note 1: Do not write <MST> to "0" when it is "0". (Restart cannot be initiated.)

Note 2: When the master device is acting as a receiver, data transmission from the slave device which serves as a transmitter must be completed before generating a restart. To complete data transfer, slave device must receive a "High" level acknowledge signal. For this reason, <LBR> before generating a restart becomes "1", the rising edge of the SCL line is not detected even <LBR>= "1" is confirmed by following the restart procedure. To check the status of the SCL line, read the port.

		7	6	5	4	3	2	1	0	
I2CxCR2	←	0	0	0	1	1	0	0	0	Releases the bus.
if I2CxSR<BB> ≠ 0										Checks that the SCL pin is released.
Then										
if I2CxSR<LRB> ≠ 1										Checks that no other device is pulling the SCL pin to the "Low".
Then										
4.7 μs Wait										
I2CxCR1	←	X	X	X	1	0	X	X	X	Selects the acknowledgment mode.
I2CxDBR	←	X	X	X	X	X	X	X	X	Sets the desired slave address and direction.
I2CxCR2	←	1	1	1	1	1	0	0	0	Generates the start condition.

Note: X; Don't care

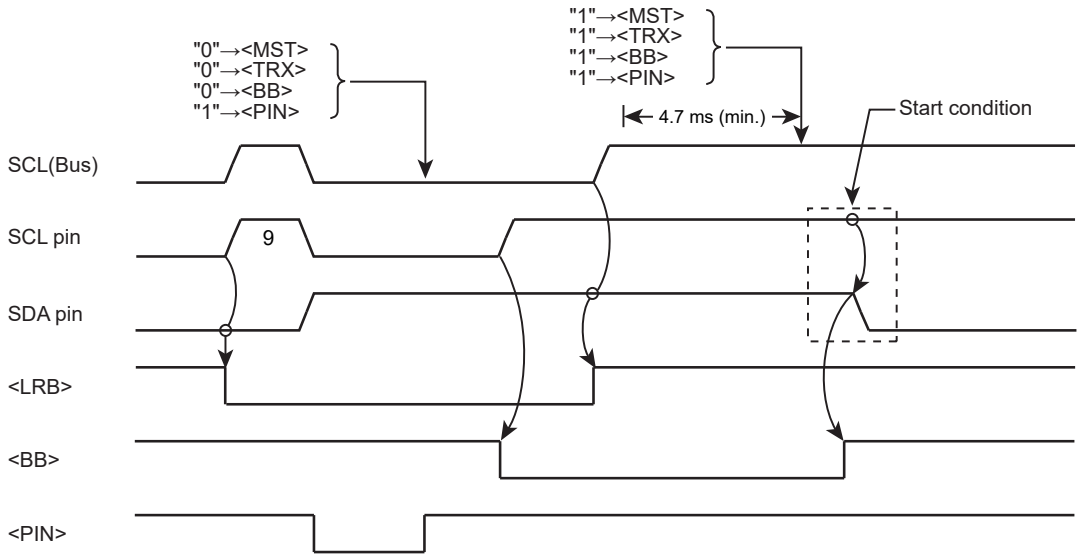


Figure 13-15 Timing Chart of Generating a Restart

13.6 Notice on usage

13.6.1 Register values after a Software Reset

A software reset initializes the I2C register other than I2CxCR2<I2CM> and internal circuit and releases the SCL and SDA lines. (refer to section "13.4.1.2 Clock Synchronization".)

However, depending on read timing after a software reset, reading I2CxSR<LRB> may return a value other than the initial value as "0".

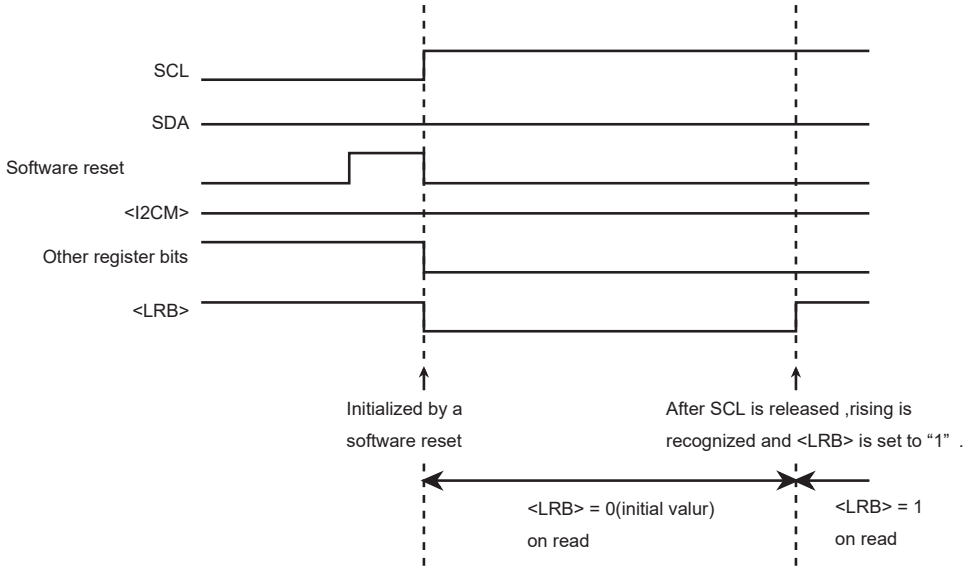


Figure 13-16 Software reset release to the SCL0 "0" to "1" while SDA="1"

14. 10-bit Analog/Digital Converter (ADC)

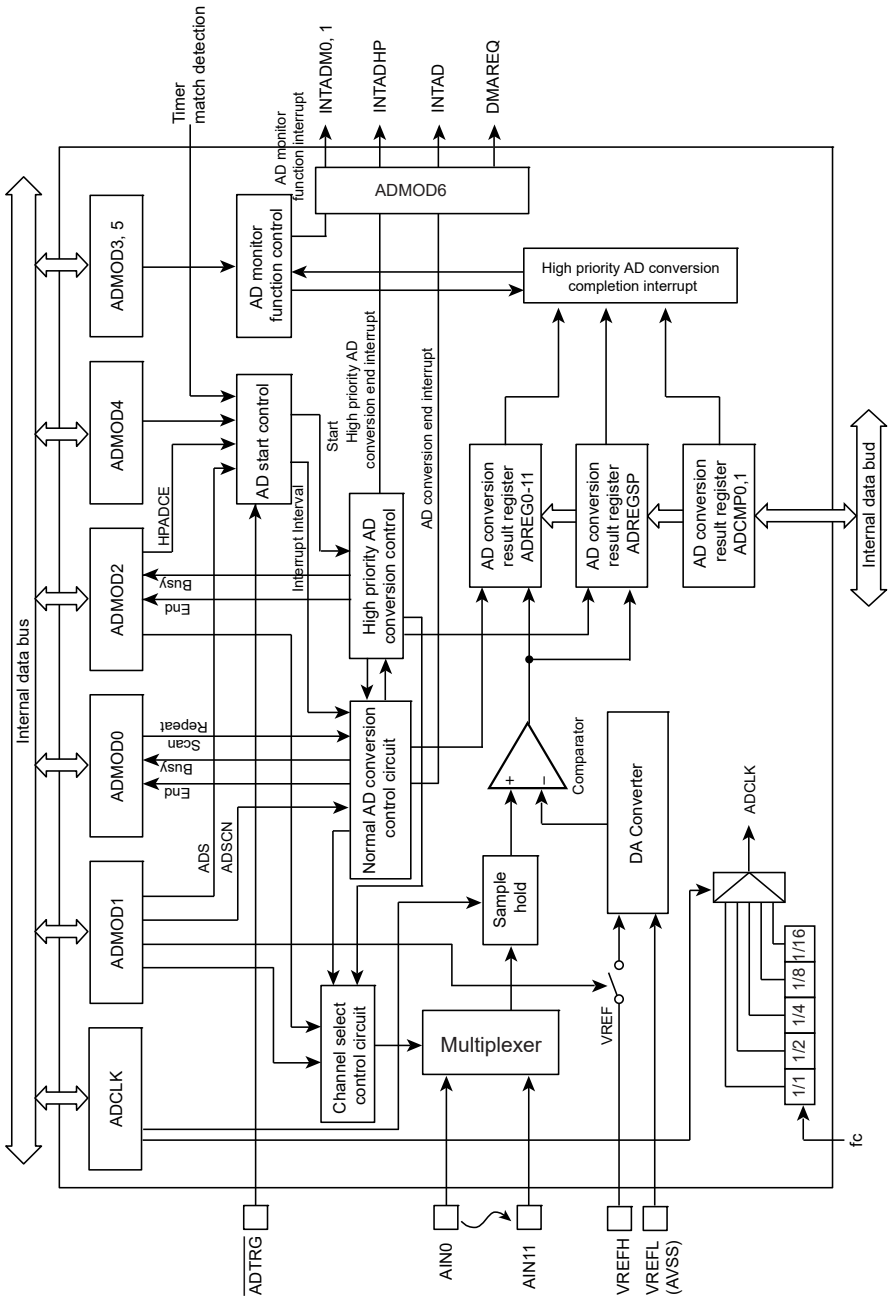
14.1 Outline

A 10-bit, sequential-conversion analog/digital converter (AD converter) is built into the TMPM037FWUG.

For details, refer to "Product information" to confirm usable channels and settings.

14.2 Configuration

Figure 14-1 shows the block diagram of this AD converter.



Note: VREFH and AVDD3 are shared. VREFL and AVSS are shared.

Figure 14-1 10-bit AD Converter Block Diagram

14.3 Registers

14.3.1 Register list

The control registers and addresses of the AD converter are as follows.

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

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
Conversion Result Register 0	ADREG0	0x0030
Conversion Result Register 1	ADREG1	0x0034
Conversion Result Register 2	ADREG2	0x0038
Conversion Result Register 3	ADREG3	0x003C
Conversion Result Register 4	ADREG4	0x0040
Conversion Result Register 5	ADREG5	0x0044
Conversion Result Register 6	ADREG6	0x0048
Conversion Result Register 7	ADREG7	0x004C
Conversion Result Register 8	ADREG8	0x0050
Conversion Result Register 9	ADREG9	0x0054
Conversion Result Register 10	ADREG10	0x0058
Conversion Result Register 11	ADREG11	0x005C
Conversion Result Register SP	ADREGSP	0x0060
Conversion Result Comparison Register 0	ADCMP0	0x0064
Conversion Result Comparison Register 1	ADCMP1	0x0068

14.3.2 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	ADCC		-	-	ADCLK			
After reset	0	1	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7-4	ADCC[1:0]	R/W	Select the AD conversion clock count 00: 35.5 conversion clock 01: 42 conversion clock 10: 68 conversion clock 11: 81 conversion clock
5-4	−	R	Read as 0.
3-0	ADCLK[3:0]	R/W	Select the AD conversion clock (Note1) (Note2) 0000: fc 1000:fc/6 0001: fc/2 1001:fc/12 0010: fc/4 1010:fc/24 0011: fc/8 1011:fc/48 0100: fc/16 1100:fc/96 0101 - 0111: Reserved 0101--0111 & 1101 - 1111:Reserved

Note 1: Do not change the setting of the AD conversion clock during AD conversion.

Note 2: AD conversion clock setting by ADCLK is less than or equal to fsys (system clock).(ADCLK ≤ fsys)

A clock count is required to satisfy the condition that described below.

VREFH AVDD	Conversion time
2.7 to 3.6V	16.2 μs or longer
2.3 to 3.6V	32.4 μs or longer

14.3.3 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	EOCFN	ADBFN	-	ITM		REPEAT	SCAN	ADS
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	EOCFN	R	Normal AD conversion completion flag (note1) 0: Before or during conversion 1: Completion This bit is "0" cleared when it is read.
6	ADBFN	R	Normal AD conversion BUSY flag 0: Conversion stop 1: During conversion
5	−	R	Read as 0.
4-3	ITM[1:0]	R/W	Interrupt in fixed channel repeat conversion mode 00: Generate in interrupt once every single conversion 01: Generate interrupt once every 4 conversions 10: Generate interrupt once every 8 conversions 11: Setting prohibited It is valid only when it's specified in the fixed channel repeat mode (<REPEAT> ="1", <SCAN> = "0").
2	REPEAT	R/W	Specify repeat mode 0: Single conversion mode 1: Repeat conversion mode
1	SCAN	R/W	Specify scan mode 0: Fixed channel mode 1: Channel scan mode If channel scan mode is selected, set the channel number to ADMOD1<ADSCAN>.
0	ADS	W	Start AD conversion start 0: Don't care 1: Start conversion Conversion must be started after setting the mode. "0" is always read.

14.3.4 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	ADSCN		ADCH			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	VREFON	R/W	VREF application control (Note) 0: OFF 1: ON
6	I2AD	R/W	Specify operation mode in IDLE mode 0: Stop 1: Operation
5-4	ADSCN[1:0]	R/W	Specify operation mode in channel scan mode 00: 4-channel scan 01: 8-channel scan 10: 12-channel scan 11: Reserved Specify operation mode when channel scan mode is selected by ADMOD0<SCAN>. The conversion channel is selected by setting of <ADCH>. Refer to the below table.
3-0	ADCH[3:0]	R/W	Select analog input channel (Refer to the below table.)

Note: **Before starting AD conversion, write "1" to the <VREFON> bit, wait for 3μs during which time the internal reference voltage should stabilize, and then write "1" to the ADMOD0<ADS>.**

Selection of analog input channel

		<ADCH[3:0]>							
		0000	0001	0010	0011	0100	0101	0110	0111
ADMOD0 <SCAN>=0	Fixed channel	AIN0	AIN1	AIN2	AIN3	AIN4	AIN5	AIN6	AIN7
ADMOD0 <SCAN>=1	<ADSCN>=00 4-channel scan	AIN0	AIN0~ AIN1	AIN0~ AIN2	AIN0~ AIN3	AIN4	AIN4~ AIN5	AIN4~ AIN6	AIN4~ AIN7
	<ADSCN>=01 8-channel scan	AIN0	AIN0~ AIN1	AIN0~ AIN2	AIN0~ AIN3	AIN0~ AIN4	AIN0~ AIN5	AIN0~ AIN6	AIN0~ AIN7
	<ADSCN>=10 12-channel scan	AIN0	AIN0~ AIN1	AIN0~ AIN2	AIN0~ AIN3	AIN0~ AIN4	AIN0~ AIN5	AIN0~ AIN6	AIN0~ AIN7

		<ADCH[3:0]>							
		1000	1001	1010	1011	1100	1101	1110	1111
ADMOD0 <SCAN>=0	Fixed channel	AIN8	AIN9	AIN10	AIN11	AIN12	AIN13	AIN14	AIN15
ADMOD0 <SCAN>=1	<ADSCN>=00 4-channel scan	AIN0	AIN0~ AIN1	AIN0~ AIN2	AIN0~ AIN3	AIN4	AIN4~ AIN5	AIN4~ AIN6	AIN4~ AIN7
	<ADSCN>=01 8-channel scan	AIN0	AIN0~ AIN1	AIN0~ AIN2	AIN0~ AIN3	AIN0~ AIN4	AIN0~ AIN5	AIN0~ AIN6	AIN0~ AIN7
	<ADSCN>=10 12-channel scan	AIN0~ AIN8	AIN0~ AIN9	AIN0~ AIN10	AIN0~ AIN11	-	-	-	-

14.3.5 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	EOCFHP	ADBFHP	HPADCE	-	HPADCH			
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	EOCFHP	R	Top-priority AD conversion completion flag (Note) 0: Before or during conversion 1: Completion
6	ADBFHP	R	Top-priority AD conversion BUSY flag 0: During conversion halts 1: During conversion
5	HPADCE	R/W	Activate top-priority conversion 0: Don't care 1: Start conversion "0" is always read.
4	−	R/W	Write "0".
3-0	HPADCH[3:0]	R/W	Select analog input channel when activating top-priority conversion. (See the table below)

Note: This bit is "0" cleared when it is read.

Selection of analog input channel

HPADCH[3:0]	0000	0001	0010	0011	0100	0101	0110	0111
Conversion channel	AIN0	AIN1	AIN2	AIN3	AIN4	AIN5	AIN6	AIN7

HPADCH[3:0]	1000	1001	1010	1011	1100	1101	1110	1111
Conversion channel	AIN8	AIN9	AIN10	AIN11	AIN12	AIN13	AIN14	AIN15

14.3.6 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	-	-	ADOBIC0	ADREGS0				ADOBSV0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	−	R/W	Write "0".
6	−	R	Read as 0.
5	ADOBIC0	R/W	Set the AD monitor function interrupt 0 0: If the value of the conversion result is smaller than the comparison register 0, an interrupt is generated. 1: If the value of the conversion result is bigger than the comparison register 0, an interrupt is generated.
4-1	ADREGS0[3:0]	R/W	Select a target conversion result register when using the AD monitor function 0 (See the below table).
0	ADOBSV0	R/W	AD monitor function 0 0: Disable 1: Enable

<ADREGS0[3:0]>	Conversion result register to be compared	<ADREGS0[3:0]>	Conversion result register to be compared
0000	ADREG0	0100	ADREG4
0001	ADREG1	0101	ADREG5
0010	ADREG2	0110	ADREG6
0011	ADREG3	0111	ADREG7
-	-	1xxx	ADREGSP

14.3.7 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	HADHS	HADHTG	ADHS	ADHTG	-	-	ADRST	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-8	−	R	Read as 0.
7	HADHS	R/W	H/W source for activating top-priority AD conversion 0: External trigger 1: Match with timer register (Note 1)
6	HADHTG	R/W	H/W for activating top-priority AD conversion 0: Disable 1: Enable
5	ADHS	R/W	H/W source for activating normal AD conversion (note2) 0: External trigger 1: Match with timer register (Note 1)
4	ADHTG	R/W	HW for activating normal AD conversion 0: Disable 1: Enable
3-2	−	R	Read as 0.
1-0	ADRST[1:0]	W	Overwriting 10 with 01 allows ADC to be software reset.(note 3)

Note 1: For details, refer to "Product information" to confirm H/W source.

Note 2: The external trigger cannot be used for H/W activation of AD conversion when it is used for H/W activation of top priority AD conversion.

Note 3: A software reset initializes all the registers except for ADCLK<ADCLK>.

14.3.8 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	-	-	ADOBIC1	ADREGS1				ADOBSV1
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-6	–	R	Read as 0.
5	ADOBIC1	R/W	Set the AD monitor function interrupt 1. 0: If the value of the conversion result is smaller than the comparison register 1, an interrupt is generated. 1: If the value of the conversion result is bigger than the comparison register 1, an interrupt is generated.
4-1	ADREGS1[3:0]	R/W	Select a target conversion result register when using the AD monitor function 1 (See the below table).
0	ADOBSV1	R/W	AD monitor function 1 0: Disable 1: Enable

<ADREGS1[3:0]>	Conversion result register to be compared	<ADREGS1[3:0]>	Conversion result register to be compared
0000	ADREG0	0100	ADREG4
0001	ADREG1	0101	ADREG5
0010	ADREG2	0110	ADREG6
0011	ADREG3	0111	ADREG7
–	–	1xxx	ADREGSP

14.3.9 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	-	-	-	-	ADM1DMA	ADM0DMA	ADHPDMA	ADDMA
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as 0.
3	ADM1DMA	R/W	Specify AD monitor function 1 DMA activation factor.(Triggered by INTADM1) 0: Disable 1: Enable
2	ADM0DMA	R/W	Specify AD monitor function 0 DMA activation factor.(Triggered by INTADM0) 0: Disable 1: Enable
1	ADHPDMA	R/W	Specify top-priority AD conversion DMA activation factor.(Triggered by INTADHP) 0: Disable 1: Enable
0	ADDMA	R/W	Specify normal AD conversion DMA activation factor.(Triggered by INTAD) 0: Disable 1: Enable

14.3.10 ADREGn (Conversion Result Register n: n = 0 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	ADRn							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	ADRn		-	-	-	-	OVRn	ADRnRF
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-6	ADRn[9:0]	R	AD conversion result Conversion result is stored. For information about the correlation between the conversion channel and the conversion result register, refer to the Table 14-2 and Table 14-3, "14.4.5.7 Interrupt generation timings and AD conversion result storage register".
5-2	−	R	Read as 0.
1	OVRn	R	Overrun flag 0: Not generated 1: Generated If the conversion result is overwritten before reading <ADR0>, "1" is set. This bit is "0" cleared when it is read.
0	ADRnRF	R	AD conversion result storage flag 0: Conversion result is not stored 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

14.3.11 ADREGSP (AD Conversion Result Register SP)

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

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-6	ADRSP[9:0]	R	AD conversion result. Top-priority AD conversion result is stored
5-2	−	R	Read as 0
1	OVRSP	R	Overrun flag 0: Not generated 1: Generated If a conversion result is overwritten before reading <ADRSP>, "1" is set. This bit is "0" cleared when it is read.
0	ADRSPRF	R	AD conversion result storage flag 0: Conversion result is not stored. 1: Conversion result is stored. If a conversion result is stored, "1" is set. This bit is "0" cleared when the conversion result is read.

Note: Access to this register must be a half word or a word access.

14.3.12 ADCMP0 (AD Conversion Result Comparison Register 0)

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	ADCOM0							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	ADCOM0		-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-6	ADCOM0[9:0]	R/W	When AD monitor function 0 is enabled, it sets a value to be compared with the value of the conversion result register specified by ADMOD3<ADREGS0>.
5-0	−	R	Read as 0.

Note: **To write values into this register, the AD monitor function 0 must be disabled (ADMOD3<ADBSV0> ="0").**

14.3.13 ADCMP1 (AD Conversion Result Comparison Register 1)

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit symbol	ADCOM1							
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	ADCOM1		-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-16	−	R	Read as 0.
15-6	ADCOM1[9:0]	R/W	When AD monitor function 1 is enabled, it sets a value to be compared with the value of the conversion result register specified by ADMODt<ADREGS1>.
5-0	−	R	Read as 0.

Note: **To write values into this register, the AD monitor function 1 must be disabled (ADMOD5<ADBSV1>="0").**

14.4 Description of Operations

14.4.1 Analog Reference Voltage

The "High" level of the analog reference voltage shall be applied to the VRFEH pin, and the "Low" shall be applied to the VREFL pin.

To start AD conversion, make sure that you first write "1" to the <VREFON> bit, wait for 3 μs during which time the internal reference voltage should stabilize, and then write "1" to the ADMOD0<ADS> bit.

If you do not use ADC function, write "0" to the ADMOD1<DACON>. The consumption current of the analog circuit is reduced.

Note: In TMPM037FWUG VREFH and AVDD3 are shared.Also VREFL and AVSS are shared.

14.4.2 AD Conversion Mode

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

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

14.4.2.1 Normal AD conversion

For normal AD conversion, the following four operation modes are supported and the operation mode is selected with the ADMOD0 <REPEAT> <SCAN>.

- Fixed channel single conversion mode
- Channel scan single conversion mode
- Fixed channel repeat conversion mode
- Channel scan repeat conversion mode

For channel scan mode, the following three modes are supported and the operation mode is selected with the ADMOD1<ADSCN>.

- 4-channel scan mode
- 8-channel scan mode
- 12-channel scan mode

(1) Fixed channel single conversion mode

If ADMOD0<REPEAT, SCAN> is set to "00", "AD conversion is performed in the fixed channel single conversion mode.

In this mode, AD conversion is performed once for one channel selected. After AD conversion is completed, ADMOD0<EOCFN> is set to "1", ADMOD0<ADBFN> is cleared to "0", and the AD conversion completion interrupt request (INTAD) is generated. <EOCFN> is cleared to "0" upon read.

(2) Channel scan single conversion mode

If ADMOD0 <REPEAT, SCAN> is set to "01," AD conversion is performed in the channel scan single conversion mode.

In this mode, AD conversion is performed once for each scan channel selected. After AD scan conversion is completed, ADMOD0<EOCFN> is set to "1", ADMOD0<ADBFN> is cleared to "0", and the conversion completion interrupt request (INTAD) is generated. <EOCFN> is cleared to "0" upon read.

(3) Fixed channel repeat conversion mode

If ADMOD0<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. After AD conversion is completed, ADMOD0<EOCFN> is set to "1". ADMOD0<ADBFN> is not cleared to "0". It remains at "1". The timing with which the conversion completion interrupt request (INTAD) is generated can be selected by setting ADMOD0<ITM> to an appropriate setting. <EOCFN> is set with the same timing as this interrupt INTAD is generated.

<EOCFN> is cleared to "0" upon read.

(4) Channel scan repeat conversion mode

If ADMOD0<REPEAT, SCAN> is set to "11", AD conversion is performed in the channel scan repeat conversion mode.

In this mode, AD conversion is performed repeatedly for a scan channel selected. Each time one AD scan conversion is completed, ADMOD0<EOCFN> is set to "1", and the conversion completion interrupt request (INTAD) is generated. ADMOD0<ADBFN> is not cleared to "0". It remains at "1". <EOCFN> is cleared to "0" upon read.

14.4.2.2 Top-priority AD conversion

By interrupting ongoing normal AD conversion, top-priority AD conversion can be performed.

The fixed-channel single conversion is automatically selected, irrespective of the ADMOD0 <REPEAT, SCAN> setting. When conditions to start operation are met, a conversion is performed just once for a channel designated by ADMOD2<HPADCH>. When conversion is completed, the top-priority AD conversion completion interrupt (INTADHP) is generated, and ADMOD2<EOCFHP> showing the completion of AD conversion is set to "1". <ADBFHP> returns to "0". EOCFHP flag is cleared to "0" upon read.

Top-priority AD conversion activated while top-priority AD conversion is under way is ignored.

14.4.3 AD Monitor Function

There are two channels of AD monitor function.

If ADMOD3<ADOBSV0> and ADMOD5<ADOBSV1> are set to "1", the AD monitor function is enabled. If the value of the conversion result register specified by ADMOD3 <ADREGS0> and ADMOD5 <ADREGS1> becomes larger or smaller ("Larger" or "Smaller" to be designated by ADMOD3 <ADOBIC0> and ADMOD5 <ADBIC1>) than the value of a comparison register, the AD monitor function interrupt (INTADM0, INTADM1) is generated. This comparison operation is performed each time a result is stored in a corresponding conversion result register.

If the conversion result register assigned to perform the AD monitor function is continuously used without reading the conversion result, the conversion result is overwritten. The conversion result storage flag <ADRxRF> and the overrun flag <OVRx> remain being set.

14.4.4 Selecting the Input Channel

How the input channel is selected is different depending on AD converter operation mode to be used.

1. Normal AD conversion mode
   - If the analog input channel is used in a fixed state (ADMOD0<SCAN> = "0")

     One channel is selected from analog input pins by setting ADMOD1<ADCH> to an appropriate setting.
   - If the analog input channel is used in a scan state (ADMOD0<SCAN> = "1")

     One scan mode is selected from the scan modes by setting ADMOD1 <ADCH> and <ADSCN> to an appropriate setting.

2. Top-priority AD conversion mode

   One channel is selected from analog input pins by setting ADMOD2<HPADCH> to an appropriate setting.

14.4.5 AD Conversion Details

14.4.5.1 Starting AD Conversion

Normal AD conversion is activated by setting ADMOD0<ADS> to "1". Top-priority AD conversion is activated by setting ADMOD2<HPADCE> to "1".

Four operation modes are made available to normal AD conversion. In performing normal AD conversion, one of these operation modes must be selected by setting ADMOD0<REPEAT,SCAN> to an appropriate setting. For top-priority AD conversion, only one operation mode can be used: fixed channel single conversion mode.

Normal AD conversion can be activated using the H/W activation source selected by ADMOD4 <ADHS>, and top-priority AD conversion can be activated using the HW activation source selected by ADMOD4 <HADHS>. If bits of <ADHS> and <HADHS> are "0", normal and top-priority AD conversions are activated in response to the input of a falling edge through the $\overline{\text{ADTRG}}$ pin. If these bits are "1", conversion is activated in response to match detection of timer.

To permit H/W activation, set ADMOD4 <ADHTG> to "1" for normal AD conversion and set ADMOD4 <HADHTG> to "1" for top-priority AD conversion.

Software activation is still valid even after H/W activation has been permitted.

Note 1: Some products don't provide the $\overline{\text{ADTRG}}$ pin.

Note 2: When an external trigger is used for the HW activation source of a top-priority AD conversion, an external trigger cannot be set for activating normal AD conversion H/W.

Note 3: For details, refer to "Product information" to confirm usable match detection of timer.

14.4.5.2 AD Conversion

When normal AD conversion starts, the AD conversion Busy flag (ADMOD0<ADBFN>) showing that AD conversion is under way is set to "1".

When top-priority AD conversion starts, the top-priority AD conversion Busy flag (ADMOD2 <ADBFHP>) showing that AD conversion is underway is set to "1".

At that time, the value of the Busy flag ADMOD0<ADBFN> for normal AD conversion before the start of top-priority AD conversions are retained. The value of the conversion completion flag ADMOD0<EOCFN> for normal AD conversion before the start of top-priority AD conversion can also be retained.

Note: Normal AD conversion must not be activated when top-priority AD conversion is under way.

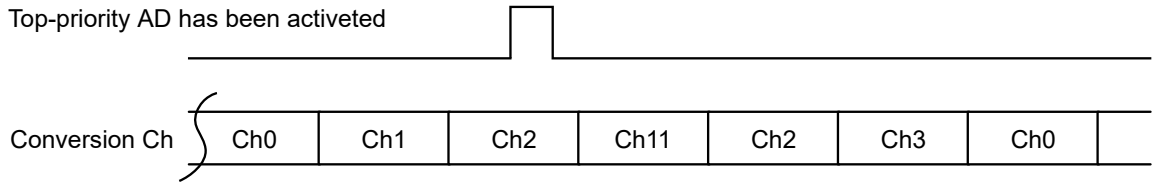
14.4.5.3 Top-priority AD conversion during normal AD conversion

If top-priority AD conversion has been activated during normal AD conversion, ongoing normal AD conversion is suspended, and restarts normal AD conversion after top-priority AD conversion is completed.

If ADMOD2<HPADCE> is set to "1" during normal AD conversion, ongoing normal AD conversion is suspended, and the top-priority AD conversion starts; specifically, AD conversion (fixed-channel single conversion) is executed for a channel designated by ADMOD2<HPADCH>. After the result of this top-priority AD conversion is stored in the storage register ADREGSP, normal AD conversion is resumed.

If H/W activation of top-priority AD conversion is authorized during normal AD conversion, ongoing AD conversion is discontinued when requirements for activation using a H/W activation resource are met, and top-priority AD conversion (fixed-channel single conversion) starts for a channel designated by ADMOD2<HPADCH>. After the result of this top-priority AD conversion is stored in the storage register ADREGSP, normal AD conversion is resumed.

For example, if channel repeat conversion is activated for channels AIN0 through AIN3 and if <HPADCE> is set to "1" during AIN2 conversion, AIN2 conversion is suspended, and conversion is performed for a channel designated by <HPADCH> (AIN11 in the case shown below). After the result of conversion is stored in ADREGSP, channel repeat conversion is resumed, starting from AIN2.



14.4.5.4 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 ADMOD0<REPEAT>. When ongoing AD conversion is completed, the repeat conversion mode terminates, and ADMOD0<ADBFN> is set to "0".

14.4.5.5 Reactivating normal AD conversion

To reactivate normal AD conversion while the conversion is underway, a software reset (ADMOD3 <ADRST>) must be performed before starting AD conversion. The H/W activation method must not be used to reactivate normal AD conversion.

14.4.5.6 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 ADMOD0 <EOCFN> which indicates the completion of AD conversion and the register ADMOD0 <ADBFN>.

For details, refer to Table 14-2 and Table 14-3 to confirm storage register corresponding to the conversion mode.

Interrupt requests, flag changes are as shown below.

- Fixed-channel single conversion mode

  After AD conversion completed, ADMOD0<EOCFN> is set to "1", ADMOD0<ADBFN> is cleared to "0", and the interrupt request is generated.

- Channel scan single conversion mode

  After the channel scan conversion is completed, ADMOD0<EOCFN> is set to "1", ADMOD0<ADBFN> is set to "0", and the interrupt request INTAD is generated.

- Fixed-channel repeat conversion mode

  ADMOD0<ADBFN> is not cleared to "0". It remains at "1". The timing with which the interrupt request INTAD is generated can be selected by setting ADMOD0<ITM> to an appropriate setting. ADMOD0<EOCFN> is set with the same timing as this interrupt INTAD is generated.

- Channel scan repeat conversion mode

  Each time one AD scan conversion is completed, ADMOD0 <EOCF> is set to "1" and interrupt request INTAD is generated. ADMOD0<ADBFN> is not cleared to "0". It remains at "1".

(2) Top-priority AD conversion completion

After the AD conversion is completed, the top-priority AD conversion completion interrupt (INTADHP) is generated, and ADMOD2<EOCFHP> which indicates the completion of top-priority AD conversion is set to "1".

AD conversion results are stored in the AD conversion result register SP.

(3) Data polling

To confirm the completion of AD conversion without using interrupts, data polling can be used. When AD conversion is completed, ADMOD0<EOCFN> is set to "1". To confirm the completion of AD conversion and to obtain the results, poll this bit.

AD conversion result storage register must be read by half word or word access. If <OVRx> = “0” and <ADRxRF> = "1", a correct conversion result has been obtained.

14.4.5.7 Interrupt generation timings and AD conversion result storage register

Table 14-1 shows a relation in the following three items: AD conversion modes, interrupt generation timings and flag operations. Table 14-2 and Table 14-3 shows a relation between analog channel inputs and AD conversion result registers.

Table 14-1 Relations in conversion modes, interrupt generation timings and flag operations

Conversion mode		Scan/repeat mode setting			Interrupt generation timing	ADMOD0<EOCFN>/ ADMOD2<EOCFHP> set timing (Note)	ADMOD0	ADMOD2
		ADMOD0 <REPEAT>	ADMOD0 <SCAIN>	ADMOD0 <ITM[1:0]>			<ADBFN> (After the interrupt is generated)	<ADBFHP>
Normal conversion	Fixed-channel single conversion	0	0	–	After generation is completed.	After conversion is completed.	0	–
	Fixed-channel repeat conversion	1	0	00	Each time one conversion is completed.	After one conversion is completed.	1	–
				01	Each time four conversion is completed.	After four conversions are completed.	1	–
				10	Each time eight conversion is completed.	After eight conversions are completed.	1	–
	Channel scan single conversion	0	1	–	After scan conversion is completed.	After scan conversion is completed.	0	–
	Channel scan repeat conversion	1	1	–	After one scan conversion is completed.	After one scan conversion is completed.	1	–
Top-priority conversion		–	–	–	After completion is completed.	Conversion completion	–	0

Note: ADMOD0<EOCFN> and ADMOD2<EOCFHP> are cleared upon read.

Table 14-2 Result registers (Fixed-channel repeat conversion mode)

<ITM[1:0]>	Result register
00 Generate in interrupt once every single conversion	ADREG0
01 Generate interrupt once every 4 conversions	ADREG0 to ADREG3
10 Generate interrupt once every 8 conversions	ADREG0 to ADREG7

Table 14-3 Result registers (Except for fixed-channel repot conversion mode)

ADMOD1 <ADCH[3:0]>	ADMOD0 <SCAN>=0		ADMOD0 <SCAN>=1					
	Fixed channel		<ADSCN>=00 4-channel scan		<ADSCN>=00 8-channel scan		<ADSCN>=00 12-channel scan	
	Conversion channel	Result register	Conversion channel	Result register	Conversion channel	Result register	Conversion channel	Result register
0000	AIN0	ADREG0	AIN0	ADREG0	AIN0	ADREG0	AIN0	ADREG0
0001	AIN1	ADREG1	AIN0 to AIN1	ADREG0 to ADREG1	AIN0 to AIN1	ADREG0 to ADREG1	AIN0 to AIN1	ADREG0 to ADREG1
0010	AIN2	ADREG2	AIN0 to AIN2	ADREG0 to ADREG2	AIN0 to AIN2	ADREG0 to ADREG2	AIN0 to AIN2	ADREG0 to ADREG2
0011	AIN3	ADREG3	AIN0 to AIN3	ADREG0 to ADREG3	AIN0 to AIN3	ADREG0 to ADREG3	AIN0 to AIN3	ADREG0 to ADREG3
0100	AIN4	ADREG4	AIN4	ADREG4	AIN0 to AIN4	ADREG0 to ADREG4	AIN0 to AIN4	ADREG0 to ADREG4
0101	AIN5	ADREG5	AIN4 to AIN5	ADREG4 to ADREG5	AIN0 to AIN5	ADREG0 to ADREG5	AIN0 to AIN5	ADREG0 to ADREG5
0110	AIN6	ADREG6	AIN4 to AIN6	ADREG4 to ADREG6	AIN0 to AIN6	ADREG0 to ADREG6	AIN0 to AIN6	ADREG0 to ADREG6
0111	AIN7	ADREG7	AIN4 to AIN7	ADREG4 to ADREG7	AIN0 to AIN7	ADREG0 to ADREG7	AIN0 to AIN7	ADREG0 to ADREG7
1000	AIN8	ADREG0	AIN8	ADREG0	AIN8	ADREG0	AIN0 to AIN8	ADREG0 to ADREG8
1001	AIN9	ADREG1	AIN8 to AIN9	ADREG0 to ADREG1	AIN8 to AIN9	ADREG0 to ADREG1	AIN0 to AIN9	ADREG0 to ADREG9
1010	AIN10	ADREG2	AIN8 to AIN10	ADREG0 to ADREG2	AIN8 to AIN10	ADREG0 to ADREG2	AIN0 to AIN10	ADREG0 to ADREG10
1011	AIN11	ADREG3	AIN8 to AIN11	ADREG0 to ADREG3	AIN8 to AIN11	ADREG0 to ADREG3	AIN0 to AIN11	ADREG0 to ADREG11
1100	AIN12	ADREG4	AIN12	ADREG4	AIN8 to AIN12	ADREG0 to ADREG4	-	-
1101	AIN13	ADREG5	AIN12 to AIN13	ADREG4 to ADREG5	AIN8 to AIN13	ADREG0 to ADREG5	-	-
1110	AIN14	ADREG6	AIN12 to AIN14	ADREG4 to ADREG6	AIN8 to AIN14	ADREG0 to ADREG6	-	-
1111	AIN15	ADREG7	AIN12 to AIN15	ADREG4 to ADREG7	AIN8 to AIN15	ADREG0 to ADREG7	-	-

14.4.5.8 DMA Request

A DMA request is issued to the DMAC at the timing of AD conversion completion interrupts or AD monitor function interrupts generation.

When DMA transfer is performed, set DMA request to be enabled by the corresponding bit of ADMOD6 register

14.4.5.9 Cautions

Cautions
The result value of AD conversion may vary depending on the fluctuation of the supply voltage, or may be affected by noise.
When using analog input pins and ports alternately, do not read and write ports during conversion because the conversion accuracy may be reduced. Also the conversion accuracy may be reduced if the output ports current fluctuate during AD conversion.
Please take counteractive measures with the program such as averaging the AD conversion results.

15. Low Voltage detection circuit (LVD)

The low voltage detection circuit generates reset or an interrupt (INTLVD) by detecting a decreasing voltage.

Note: INTLVD is a factor of non-maskable interrupts (NMI).

15.1 Configuration

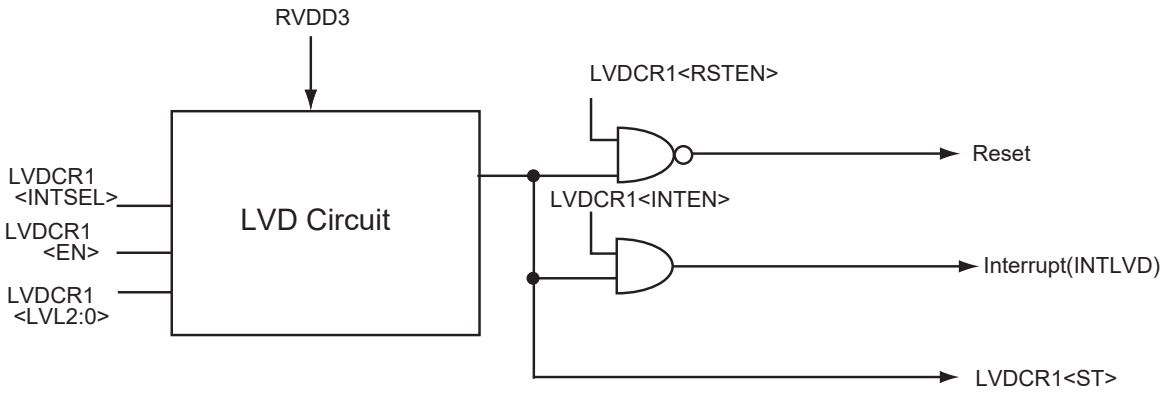


Figure 15-1 LVD Block Diagram

15.2 Registers

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

15.2.1 Register list

Register name		Address (Base+)
Reserved	-	0x0000
LVD detection control 1	LVDCR1	0x0004

15.2.2 LVDCR1 (LVD detection control register 1)

	31	30	29	28	27	26	25	24
bit symbol	−	−	−	−	−	−	−	−
after reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	−	−	−	−	−	−	−	−
after reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit symbol	−	−	−	−	−	−	-	−
after reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit symbol	ST	RSTEN	INTEN	INTSEL	LVL			EN
after reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31 - 8	−	R	Read as "0"
7	ST	R	LVL voltage detection status 0: Power-supply voltage is the same as detection voltage or higher. 1: Power-supply voltage is the same as detection voltage or lower.
6	RSTEN	R/W	Controls RESET output 0: Disable 1: Enable
5	INTEN	R/W	Controls INTLVD output 0: Disable 1: Enable
4	INTSEL	R	Interrupt generation condition. 0: Only lower than the setting voltage when voltage decreasing. 1: Both lower and upper than the setting voltage when voltage decreasing. this bit can be used with <RSTEN>="0" and <INTEN>="1".
3 - 1	LVL[2:0]	R/W	3V Power supply detection voltage 1 000: Reserved 001: Reserved 010: Reserved 011: 2.5 ± 0.2V 100: 2.6 ± 0.2V 101: 2.7 ± 0.2V 110: 2.8 ± 0.2V 111: 2.9 ± 0.2V
0	EN	R/W	Voltage detection operation 0: Disable 1: Enable

Note 1: LVDCRn is initialized by power-on reset and a terminal reset..

Note 2: There is not hysteresis between the detection of LVD and release voltage. Chattering may occur during detection, depending on the slope of power supply.

15.3 Operation

15.3.1 Selecting detection voltage and enabling voltage detection operation

Voltage detection is enabled when voltage to be detected is selected by setting the register LVDCRn<LVL [2:0]> and "1" is set to the LVDCRn<EN>.

15.3.2 Reset by Detecting a supply voltage

Reset occurs when "1" is set to the LVDCRx<RSTEN> and the supply voltage falls under the set detection voltage.

It needs approximately 100 μs to detect voltage reduction and generate reset. If the period that the supply voltage falls under the detected voltage is short, reset may not occur.

15.3.3 Interrupt by Detecting a supply voltage

A interrupt (INTLVD) occurs When "0" is set to LVDCEn<RSTEN> and "1" is set to LVDCRn<INTEN> if the supply voltage falls under or over the set detection voltage level.

The Interrupt condition can be set by LVDCRn<INTSEL>.

An interrupt occurs when "0" is set to the LVDCRx<INTSEL> and the supply voltage falls under the set detection voltage.

An interrupt occurs when "1" is set to the LVDCRx<INTSEL> and the supply voltage falls under or over the set detection voltage.

It needs approximately 100 μs to detect voltage reduction and generate a interrupt. If the period that the supply voltage falls under the detected voltage is short, an interrupt may not occur.

15.3.4 Detecting Status

Reading the LVDCRn<SR> can be confirmed the Detecting status of low voltage detection.

When the LVDCRn<ST> is "0", the voltage is the same as detection voltage or higher.

When the LVDCRn<ST> is "1", the voltage is the same as detection voltage or lower.

In the Interrupt service routine (ISR) of low voltage detection (LVD), to read the status of LVDCRn<ST> after detection is known stably the shift of the voltage

16. Watchdog Timer (WDT)

The watchdog timer (WDT) is for detecting malfunctions (runaway) 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: TMPM037FWUG does not have the watchdog timer out pin ($\overline{\text{WDTOUT}}$).

16.1 Configuration

Figure 16-1 shows the block diagram of the watchdog timer.

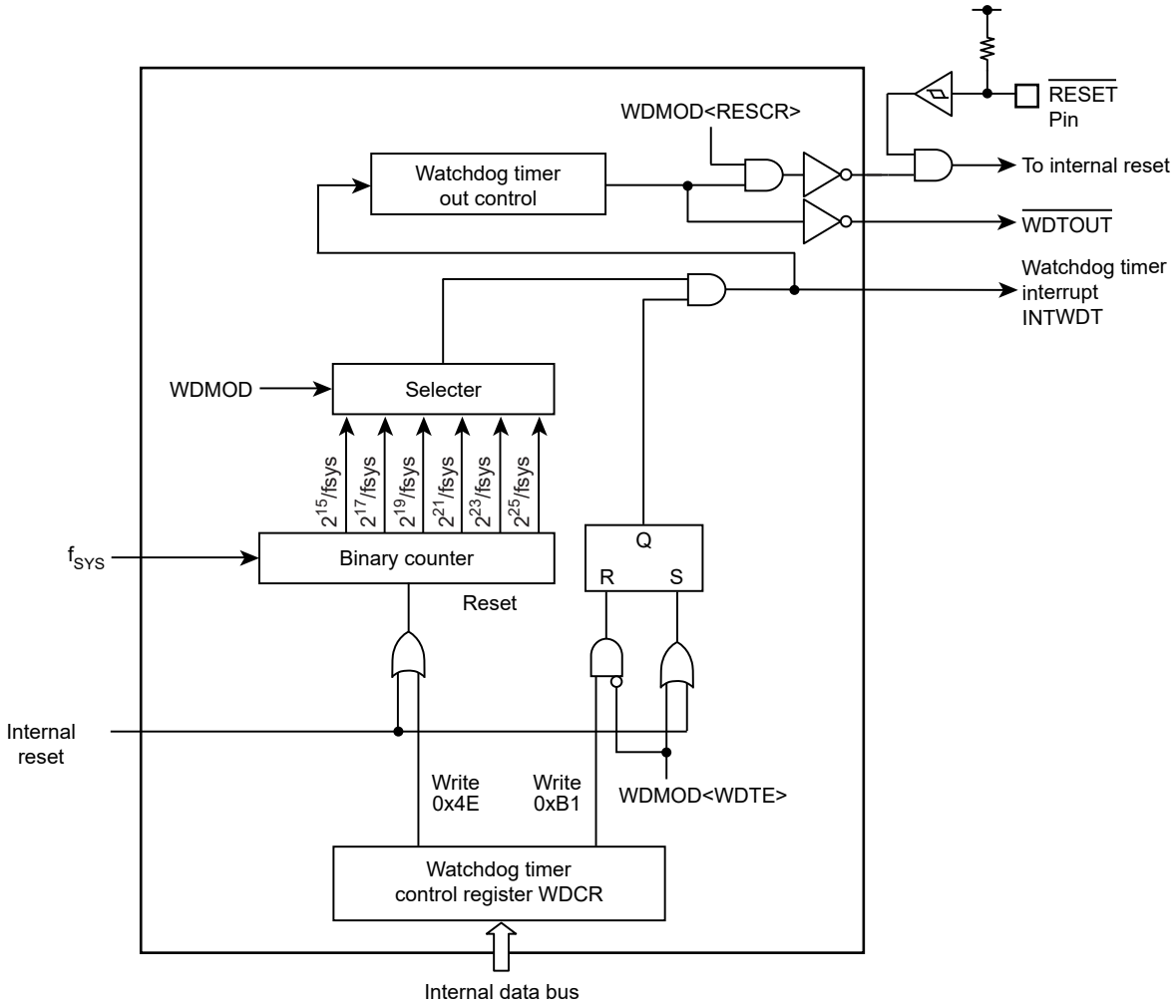


Figure 16-1 Block Diagram of the watchdog Timer

16.2 Register

16.2.1 Register List

The table below shows control registers and their addresses.

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

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

16.2.2 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 To disable the watchdog timer to protect from the error writing by the malfunction, first <WDTE> is set to "0", and then the disable code (0xB1) must be written to WDCR. To change the status of the watchdog timer from "disable" to "enable", set <WDTE> to "1".
6-4	WDTP[2:0]	R/W	Selects WDT detection time 000: $2^{15}/f_{SYS}$ 100: $2^{23}/f_{SYS}$ 001: $2^{17}/f_{SYS}$ 101: $2^{25}/f_{SYS}$ 010: $2^{19}/f_{SYS}$ 110: Reserved 011: $2^{21}/f_{SYS}$ 111: Reserved
3	−	R	Read as "0"
2	I2WDT	R/W	Operation in IDLE mode 0: Stop 1: Operate
1	RESCR	R/W	Operation after detecting malfunction 0: INTWDT interrupt request is generated. Note) 1: Reset
0	−	R/W	Write "0".

Note: INTWDT interrupt is a factor of the non-mask interrupt.

16.2.3 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

16.3 Description of Operation

16.3.1 Basic Operation

The watchdog timer is consists of the binary counter that works 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) is generated, and the watchdog timer out pin ($\overline{\text{WDTOUT}}$) outputs "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 is generated. If the binary counter is not cleared, the non-maskable interrupt is generated by INTWDT. Thus CPU detect malfunction (runaway), 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: TMPM037FWUG does not have a watchdog timer out pin ($\overline{\text{WDTOUT}}$).

16.3.2 Operation Mode and Status

The watchdog timer begins operation immediately after a reset is released. If not using the watchdog timer, it should be disabled.

The watchdog timer can not be used at the high-speed frequency clock is stopped. Before transition to below operation modes, the watchdog timer should be disabled.

In IDLE mode, its operation depend on WDMOD<I2WDT> setting.

- STOP1 mode

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

16.3.3 Operation when malfunction (runaway) is detected.

16.3.3.1 INTWDT interrupt generation

Figure 16-2 shows the case that INTWDT interrupt is generated (WDMOD<RESCR>="0").

When an overflow of the binary counter occurs, INTWDT interrupt is generated. It is a factor of non-maskable interrupt (NMI). Thus CPU detects non-maskable interrupt and perform 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, CGNMIFLG<NMIFLG0> is set.

When INTWDT interrupt is generated, 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.

Note: TMPM037FWUG does not have a watchdog timer out pin ($\overline{\text{WDTOUT}}$).

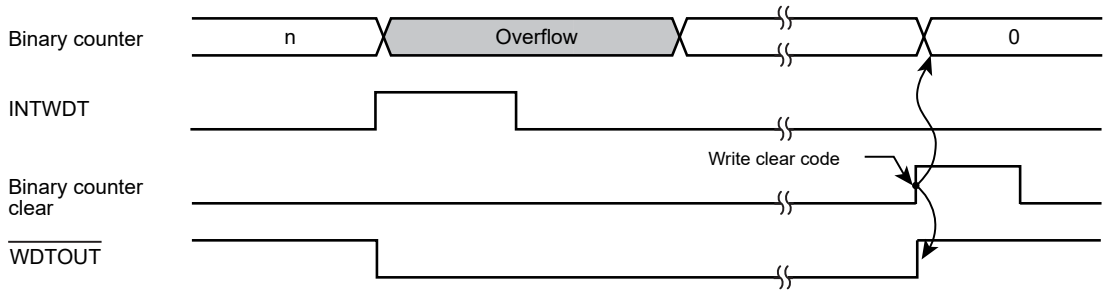


Figure 16-2 INTWDT interrupt generation

16.3.3.2 Internal Reset Generation

Figure 16-3 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.

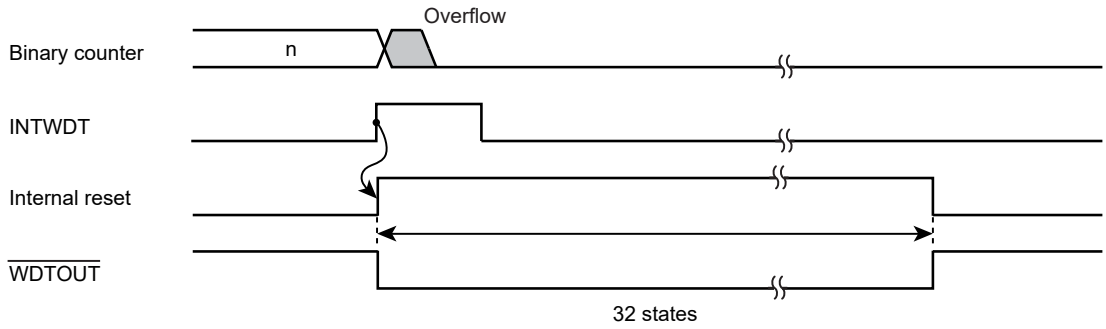


Figure 16-3 Internal reset generation

16.4 Control of the watchdog timer

16.4.1 Disable control

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

16.4.2 Enable control

Set WDMOD<WDTE> to "1".

16.4.3 Watchdog timer clearing control

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

16.4.4 Detection time of watchdog timer

Set WDMOD<WDTP[2:0]> depend on the detection time.

For example, in the case that $2^{21}/f_{SYS}$ is used, set "011" to WDMOD<WDTP[2:0]>.

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 TMPM037FWUG.

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
128 KB	−	−	4	−	32	1024	1.25ms	1.28 sec	0.1sec	0.2 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.

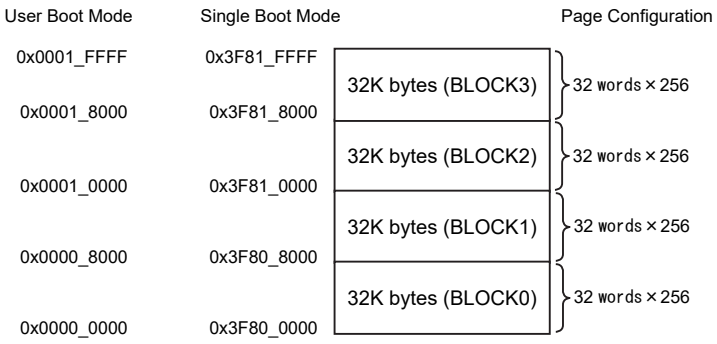


Figure 17-1 Block configuration

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

- Page

  One page is 32 words. Same address [31:7] is used in a page. First address of the group is [6:0] = 0 and the last address of the group is [6:0] = 0x7F.

- Block

  One block is 32KB and flash memory is consists of four blocks.

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.

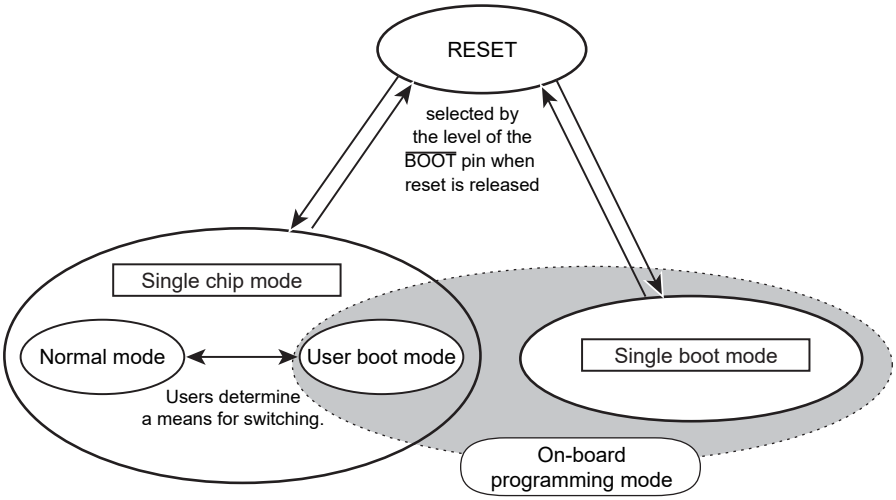


Figure 17-2 Mode transition

(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-programmed 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

Note: This device may start up in single-boot mode, when the $\overline{\text{BOOT}}$ pin is "Low" level at power-on. Therefore when MCU starting in single mode, The $\overline{\text{BOOT}}$ pin must be at "High" level at power-on until reset release operation is completed.

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
128 KB	16 KB	0x0000_0000 to 0x0001_FFFF(single chip mode) 0x3F80_0000 to 0x3F81_FFFF(single boot mode)	0x2000_0000 to 0x2000_3FFF

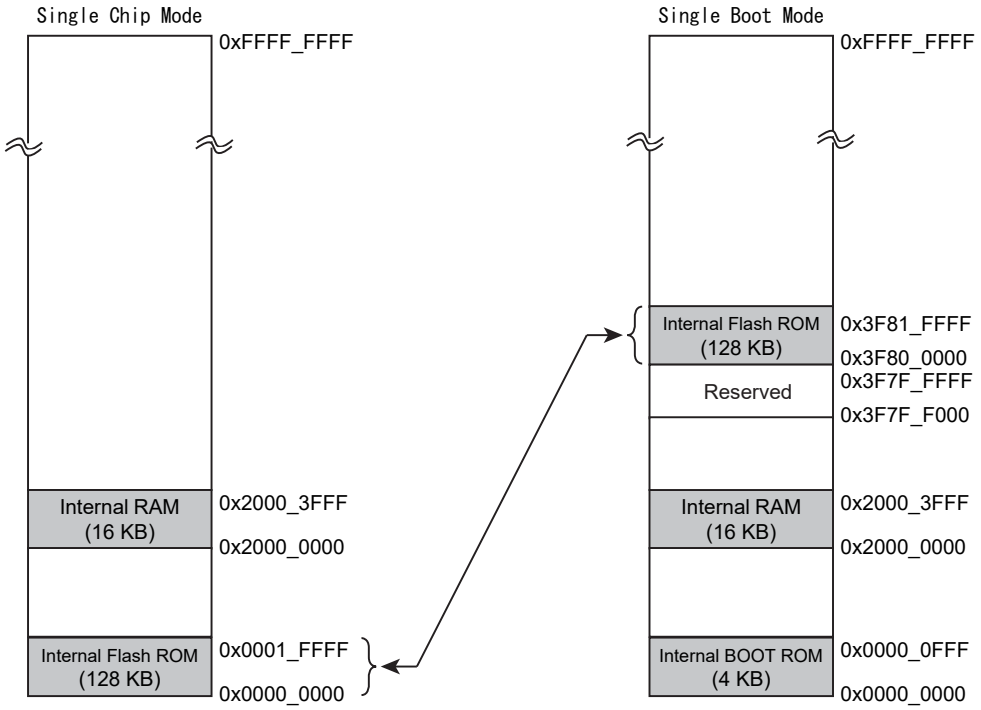


Figure 17-3 Comparison of memory map

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 cancelled. The protect bit can be monitored with FCPSRA<BLK[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 (FCPSRA<BLK>) are set to "1".

FCSECBIT<SECBIT> is set to "1" by the PowerOnReset. 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

The following table shows control registers and addresses.

For details of base address, refer to "Address lists of peripheral functions" of "Memory Map" Chapter.

Register name		Address(Base+)
Security bit register	FCSECBIT	0x0010
Flash status register	FCSR	0x0020
Flash protect status register A	FCPSRA	0x0030

17.1.6.2 FCSR (Flash 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	-	-	-	-	-	-	-	RDY_BSY
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-1	−	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.

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 PowerOnReset.

17.1.6.4 FCPSRA (Flash protect 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	-	-	-	-	BLK3	BLK2	BLK1	BLK0
After reset	0	0	0	0	(Note 1)	(Note 1)	(Note 1)	(Note 1)

Bit	Bit Symbol	Type	Function
31-4	−	R	Read as "0".
3-0	BLK3- BLK0	R	Protection status of Block3 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.

Note 1: A value will correspond to the protection status.

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.

Note: Check the FCSR<RDY_BSY> to make sure each command sequence end such as Flash writing, Flash Erase, Protection bit program, Protection bit Erase. and then hold for 200 μs or more before reading data from Flash memory or starting instruction fetch.

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 PowerOnReset or warm reset to use returning to the read mode when the automatic programming/erase operation is forcibly cancelled, 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 FCSR<RDY_BSY> is set to "0". When the automatic operation normally ends, FCSR<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 (FCSR<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 FCSR<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 one-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 one-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.

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.

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 FCPSRA<BLK>.

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 FCPSRA<BLK>.

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, FCSR<RDY_BSY> becomes "0" during the automatic protect bit erase command operation. After the operation is complete, FCSR<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	0xX55X	0xXAAX	0xX55X	–	–	–	–
	0xAA	0x55	0xF0	–	–	–	–
ID-Read	0xX55X	0xXAAX	0xX55X	IA	0xXX	–	–
	0xAA	0x55	0x90	0x00	ID	–	–
Automatic page program	0xX55X	0xXAAX	0xX55X	PA	PA	PA	PA
	0xAA	0x55	0xA0	PD0	PD1	PD2	PD3
Automatic chip erase	0xX55X	0xXAAX	0xX55X	0xX55X	0xXAAX	0xX55X	–
	0xAA	0x55	0x80	0xAA	0x55	0x10	–
Automatic block erase	0xX55X	0xXAAX	0xX55X	0xX55X	0xXAAX	BA	–
	0xAA	0x55	0x80	0xAA	0x55	0x30	–
Automatic protect bit program	0xX55X	0xXAAX	0xX55X	0xX55X	0xXAAX	0xX55X	PBA
	0xAA	0x55	0x9A	0xAA	0x55	0x9A	0x9A
Automatic protect bit erase	0xX55X	0xXAAX	0xX55X	0xX55X	0xXAAX	0xX55X	0xXX
	0xAA	0x55	0x6A	0xAA	0x55	0x6A	0x6A

Supplementary explanation

- IA: ID Address
- 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)

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 [31:15]	Addr [14]	Addr [13:12]	Addr [11:9]	Addr [8:7]	Addr [6:4]	Addr [3:0]

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 recom-mended.	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 pro-gram	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	Fix to "0"		Protect bit selection (Table 17-8)	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)
2	0x0001_8000 to 0x0001_FFFF	0x3F81_8000 to 0x3F81_FFFF	32
3	0x0001_0000 to 0x0001_7FFF	0x3F81_0000 to 0x3F81_7FFF	32
1	0x0000_8000 to 0x0000_FFFF	0x3F80_8000 to 0x3F80_FFFF	32
0	0x0000_0000 to 0x0000_7FFF	0x3F80_0000 to 0x3F80_7FFF	32

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 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 use boot 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 [14:9]	Address [8]	Address [7]	
Block0	<BLK[0]>	Fix to "0"	0	0	0x0000_0000 0x3F80_0000
Block1	<BLK[1]>		0	1	0x0000_0080 0x3F80_0080
Block2	<BLK[2]>		1	0	0x0000_0100 0x3F80_0100
Block3	<BLK[3]>		1	1	0x0000_0180 0x3F80_0180

17.2.6.5 ID-Read Code (IA, ID)

Table 17-9 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 use boot mode and the lower side is used in the single boot mode

Table 17-9 ID-Read Command codes and contents

Code	ID[7:0]	IA[13:12]	Address Example [31:0]
Manufacture code	0x98	0b00	0x0000_0000 0x3F80_0000
Device code	0x5A	0b01	0x0000_1000 0x3F80_1000
-	Reserved	0b10	-
Macro code	0x33	0b11	0x0000_3000 0x3F80_3000

17.2.6.6 Example of Command Sequence

(1) use boot mode

Command		Bus cycle						
		1	2	3	4	5	6	7
Read	Address	0x0000_0000	–	–	–	–	–	–
	Data	0x0000_00F0	–	–	–	–	–	–
Read/reset	Address	0x0000_0550	0x0000_0AA0	0x0000_0550	–	–	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_00F0	–	–	–	–
ID-Read	Address	0x0000_0550	0x0000_0AA0	0x0000_0550	IA	0x0000_0000	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0090	0x0000_0000	ID	–	–
Automatic page program	Address	0x0000_0550	0x0000_0AA0	0x0000_0550	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_0550	0x0000_0AA0	0x0000_0550	0x0000_0550	0x0000_0AA0	0x0000_0550	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0010	–
Automatic block erase	Address	0x0000_0550	0x0000_0AA0	0x0000_0550	0x0000_0550	0x0000_0AA0	BA	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0030	–
Automatic protect bit program	Address	0x0000_0550	0x0000_0AA0	0x0000_0550	0x0000_0550	0x0000_0AA0	0x0000_0550	PBA
	Data	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_009A
Automatic protect bit erase	Address	0x0000_0550	0x0000_0AA0	0x0000_0550	0x0000_0550	0x0000_0AA0	0x0000_0550	0x0000_0550
	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_0550	0x3F80_0AA0	0x3F80_0550	–	–	–	–
	Data	0x0000_00AA	0x3F80_0055	0x3F80_00F0	–	–	–	–
ID-Read	Address	0x3F80_0550	0x3F80_0AA0	0x3F80_0550	IA	0x0000_0000	–	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0090	0x0000_0000	ID	–	–
Automatic page program	Address	0x3F80_0550	0x3F80_0AA0	0x3F80_0550	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_0550	0x3F80_0AA0	0x3F80_0550	0x3F80_0550	0x3F80_0AA0	0x3F80_0550	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0010	–
Automatic block erase	Address	0x3F80_0550	0x3F80_0AA0	0x3F80_0550	0x3F80_0550	0x3F80_0AA0	BA	–
	Data	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0030	–
Automatic protect bit program	Address	0x3F80_0550	0x3F80_0AA0	0x3F80_0550	0x3F80_0550	0x3F80_0AA0	0x3F80_0550	PBA
	Data	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_009A
Automatic protect bit erase	Address	0x3F80_0550	0x3F80_0AA0	0x3F80_0550	0x3F80_0550	0x3F80_0AA0	0x3F80_0550	0x3F80_0550
	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

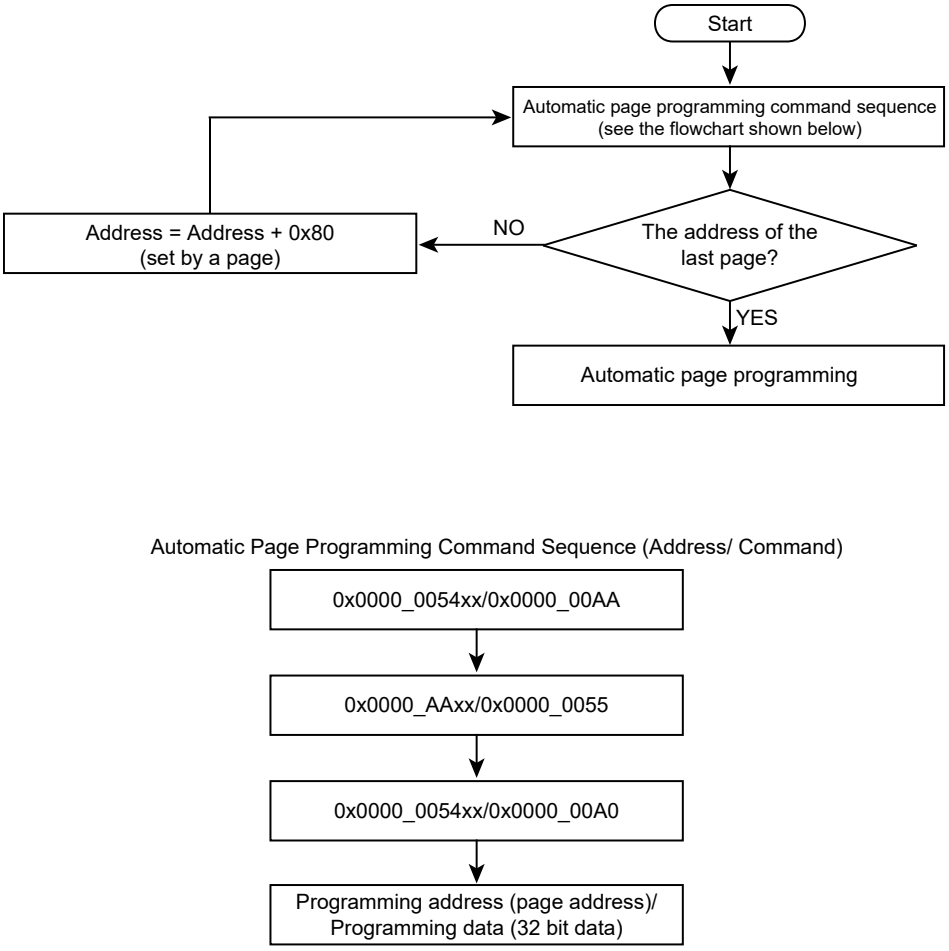


Figure 17-4 Flowchart of automatic program

17.2.7.2 Automatic Erase

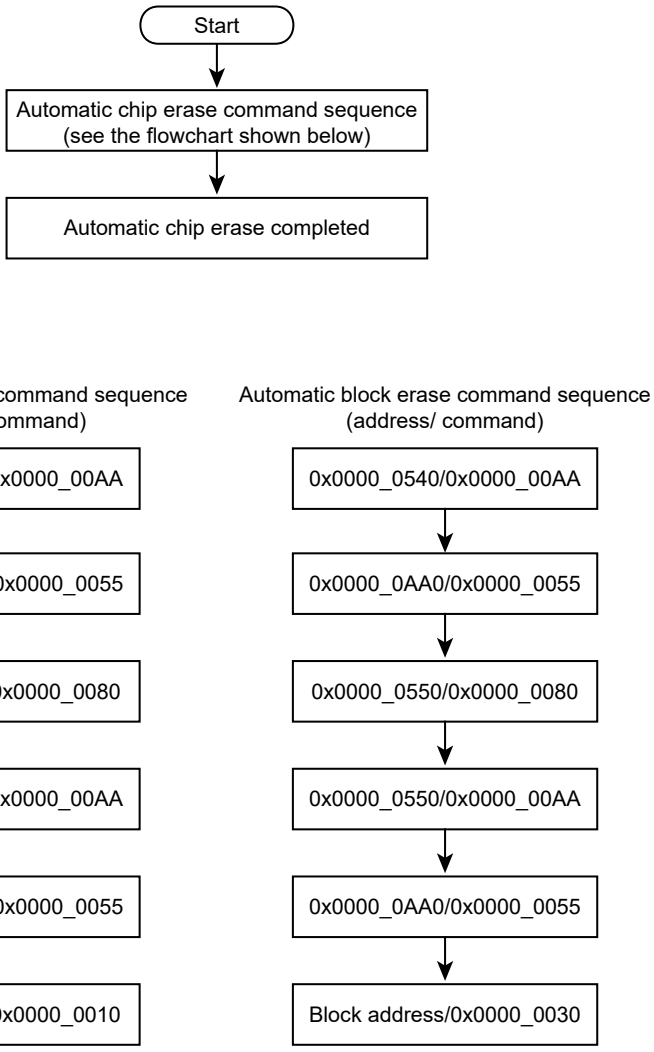


Figure 17-5 Flowchart of automatic erase

17.3 How to Reprogram Flash using Single Boot Mode

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

In the 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.

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

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

Note: This device may start up in single-boot mode, when the $\overline{BOOT}$ pin is "Low" level at power-on. Therefore when MCU starting in single mode, The $\overline{BOOT}$ pin must be at "High" level at power-on until reset release operation is completed.

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 4

  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 4

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

  Synchronous signal (SCLK4): Input mode, rising edge setting

  Handshaking signal: PB4 (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 17-10 shows the pins used in the boot program. Other than these pins are not used by the boot program.

Table 17-10 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	TXD4 (PB2)	o	o
	RXD4 (PB1)	o	o
	SCLK4 (PB3)	×	o (Input mode)
	PB4	×	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 17-11.

Table 17-11 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. The start address of the program must be even address.
Internal flash memory	The following addresses are assigned for storing software ID information and passowerds. Storing program in below addresses is not recommendable. 0x3F81_FFF0 to 0x3F81_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 17-12 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. The start address must be even 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.

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.

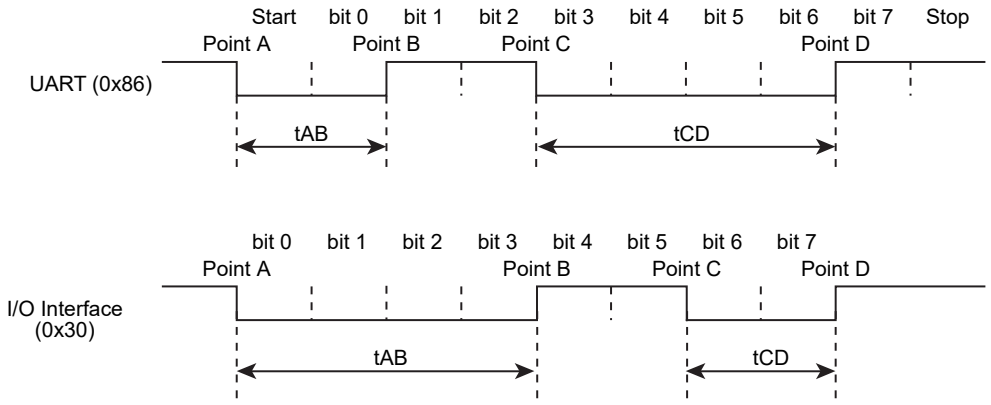


Figure 17-6 Serial operation mode determination data

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 mar-

gin 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, 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.

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).

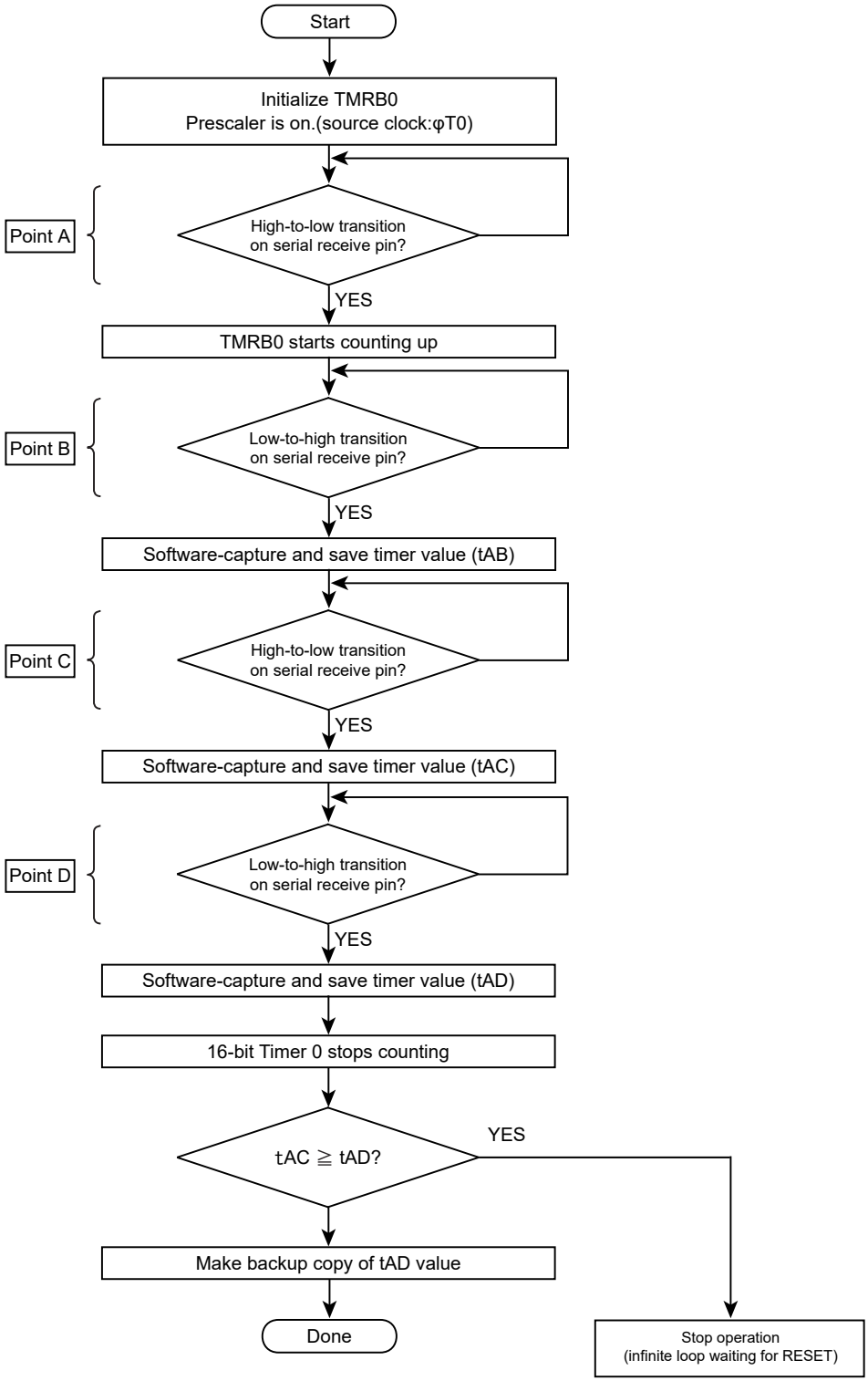


Figure 17-7 Serial operation mode receive flowchart

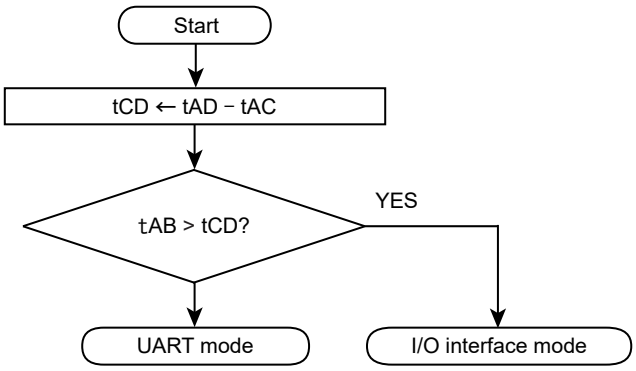


Figure 17-8 Serial operation mode determination flowchart

17.3.5.2 Acknowledge Response Data

The boot program represents processing states in specific codes and sends them to the controller. Table 17-13 to Table 17-16 show the values of acknowledge responses to each receive data.

In Table 17-14 to Table 17-16, 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 17-13 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 17-14 ACK response to the operation command data

Transmit data	Description
0xN8 (Note)	A receive error occurs in the operation command data
0xN1 (Note)	An undefined operation command data is received normally.
0x10	Determined as a RAM transfer command
0x40	Determined as a flash memory chip erase command

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

Table 17-15 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 17-16 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.

Note: Even when an erase command is performed normally, a Negative acknowledge may be returned by ACK response. Check the FCSR<RDY_BSY> to make sure the command sequence end, and then hold for 200 μ s or more, after that reconfirm the erase status.

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	0x3F81_FFF0 (1byte)
Password area	0x3F81_FFF4 to 0x3F81_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.

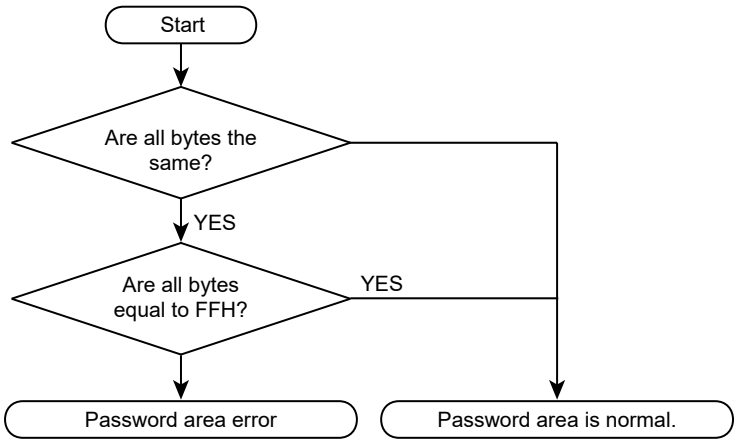


Figure 17-9 Password area check flowchart

(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.

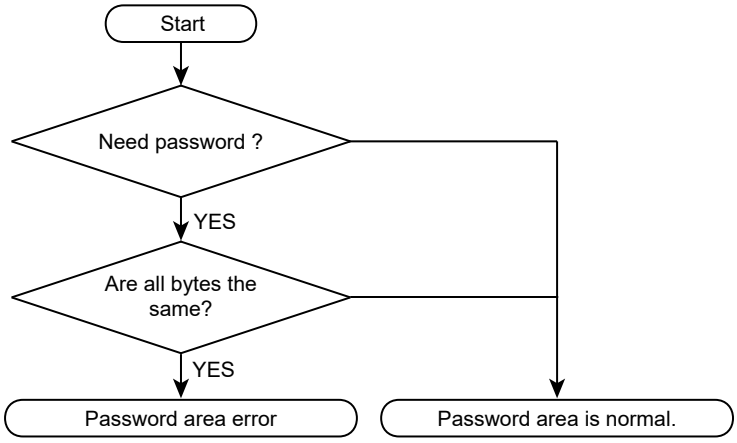


Figure 17-10 Password area check flowchart

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 TMPM037FWUG

Transfer direction (C←T): TMPM037FWUG 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 17-12, 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 17-12, 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) 0x3F81_FFF4 to 0x3F81_FFFF	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 0x3F81_FFF0 to 0x3F81_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. The address must be even address.
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 TMPM037FWUG

Transfer direction (C←T): TMPM037FWUG 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 17-12, receive data is echoed back. If the data does not correspond to the command in Table 17-12, 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) 0x3F81_FFF4 to 0x3F81_FFFF	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 0x3F81_FFF0 to 0x3F81_FFFF of data of Flash memory in order. If the data does not match, a password error flag is set.
17	C→T	5th to 16th byte CHECK SUM value	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".

Number of transfer bytes	Transfer direction	Transfer data	Description
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 (note1) 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.

Note 1: Even when an erase command is performed normally, a Negative acknowledge may be returned by ACK response. Check the FCSR<RDY_BSY> to make sure the command sequence end, and then hold for 200 μ s or more, after that reconfirm the erase status.

17.3.8 Boot Program Whole Flowchart

This section shows a boot program whole flowchart.

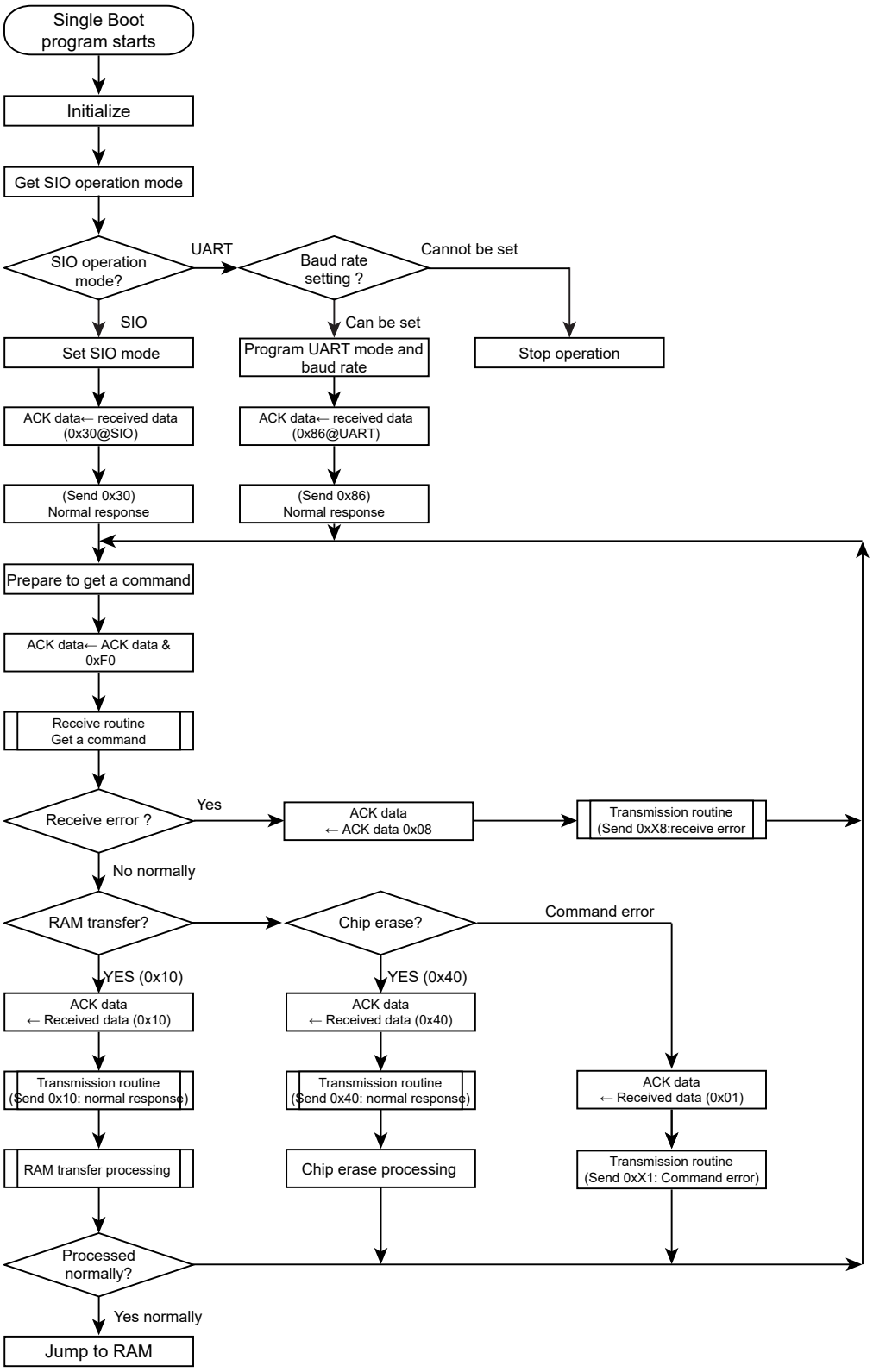


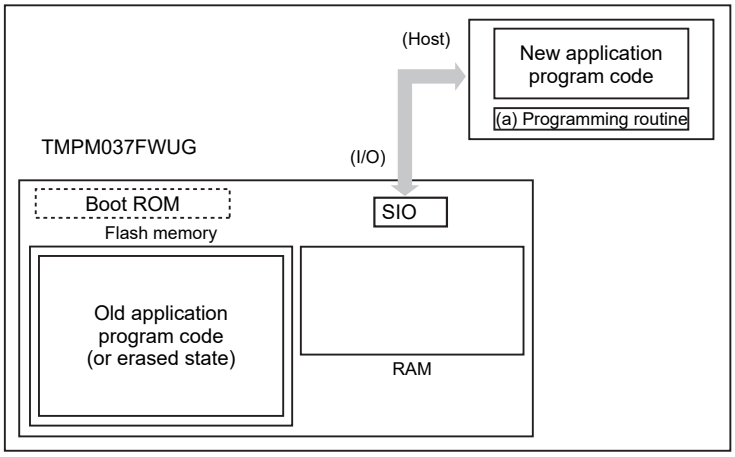
Figure 17-11 Boot program whole flowchart

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.

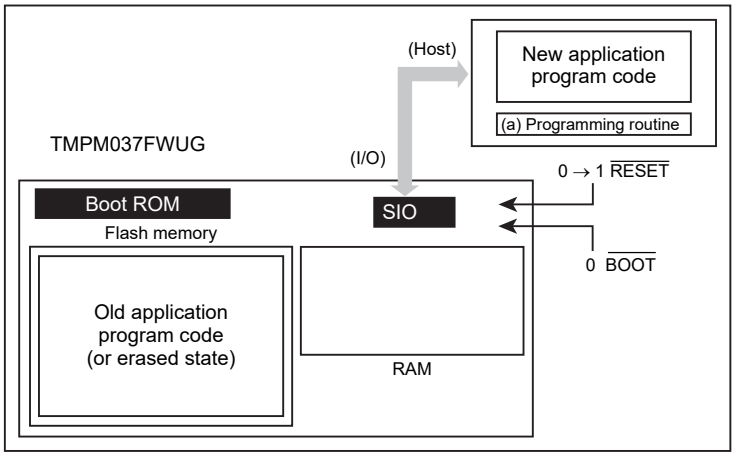
17.3.9.1 Step-1

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 (SIO4), the SIO4 must be connected to an external host. A programming routine (a) is prepared on the host.



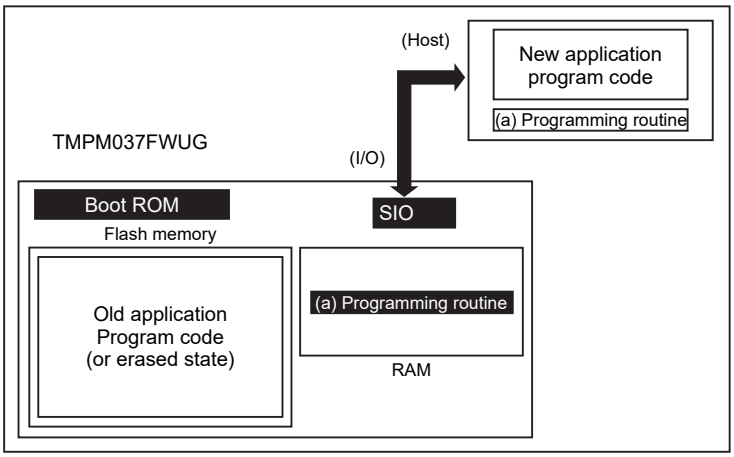
17.3.9.2 Step-2

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 SIO4 from the source (host). A password verification with the password in the user application program is performed. (If Flash memory is erased, an erase data (0xFF) is dealt with a password.)



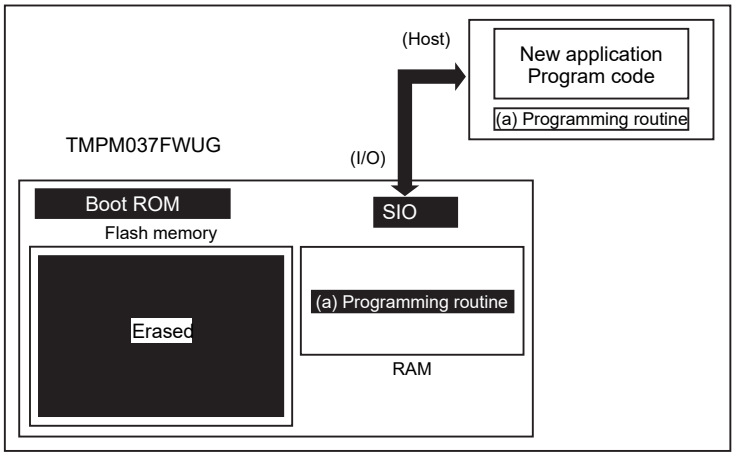
17.3.9.3 Step-3

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

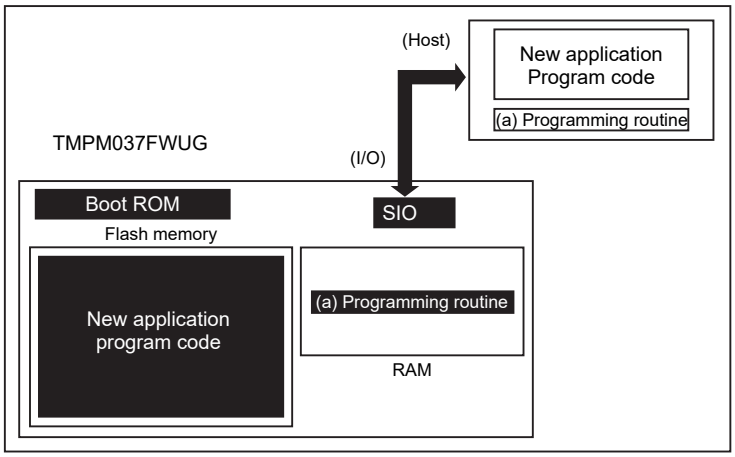
The boot program jumps to the programming routine (a) in the on-chip RAM to erase the flash block containing old application program codes. The Block Erase or Chip Erase command is be used.



17.3.9.5 Step-5

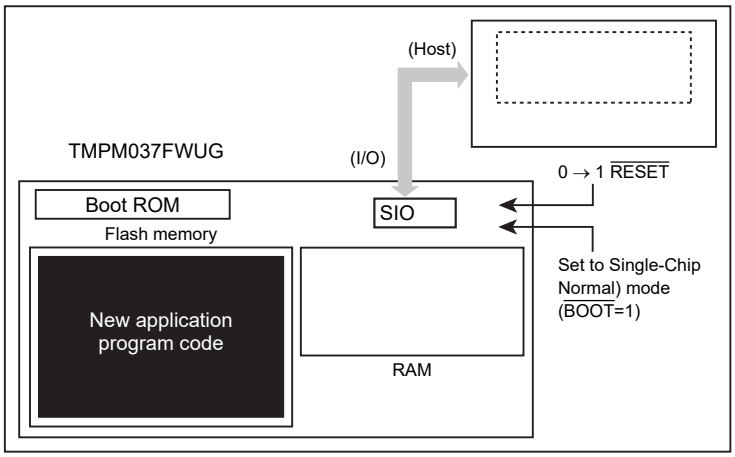
The boot program executes the programming routine (a) to download new application program codes from the host and programs it into the erased flash area. When the programming is complete, the writing or erase protection of that flash area in the user's program must be set.

In the example below, new program code comes from the same host via the same SIO4 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 use boot 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.

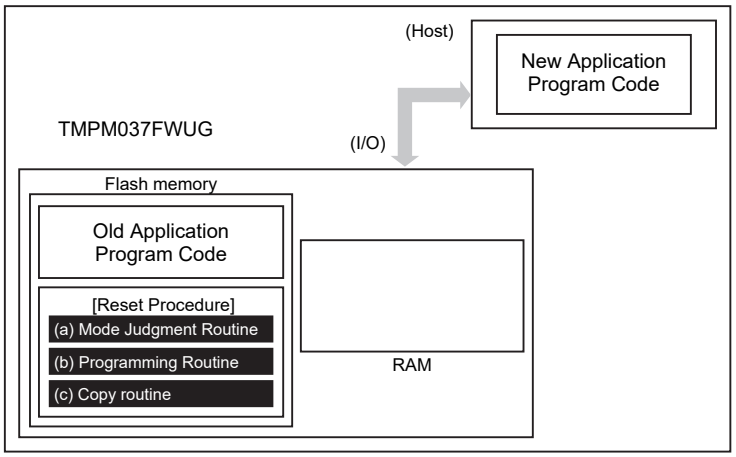
Taking examples from two cases such as the method that reprogramming routine stored in Flash memory (1-A) and transferred from the external device (1-B), the following section explains the procedure. For a detail of the program/erase to Flash memory, refer to "17.2 Detail of Flash Memory".

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

17.4.1.1 Step-1

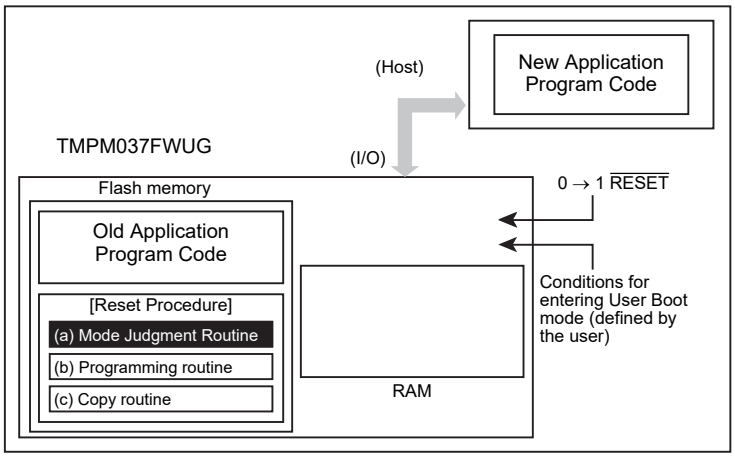
A user determines the conditions (e.g., pin status) to enter the user boot mode and the I/O bus to be used to transfer data. Then suitable circuit design and program are created. Before installing the device on a printed circuit board, write the following three program routines into an arbitrary flash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

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



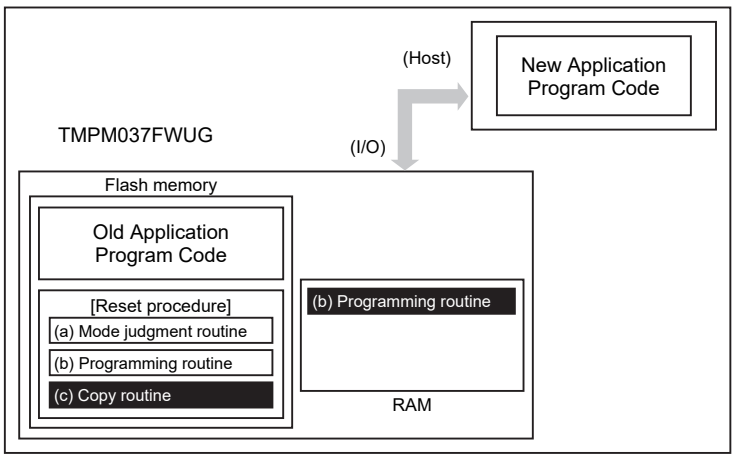
17.4.1.2 Step-2

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.



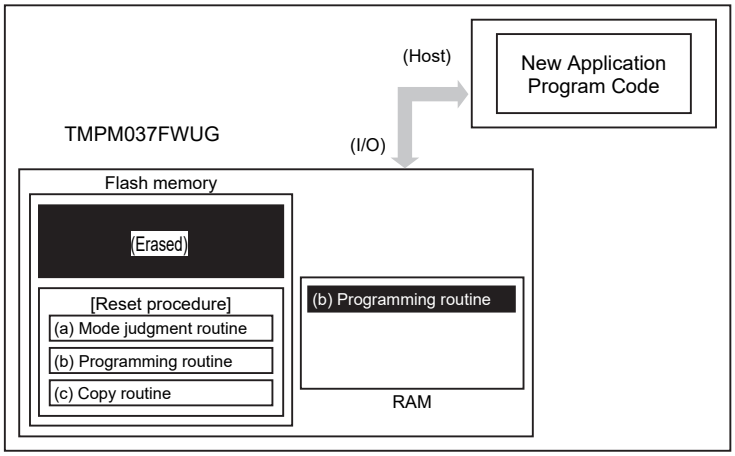
17.4.1.3 Step-3

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.



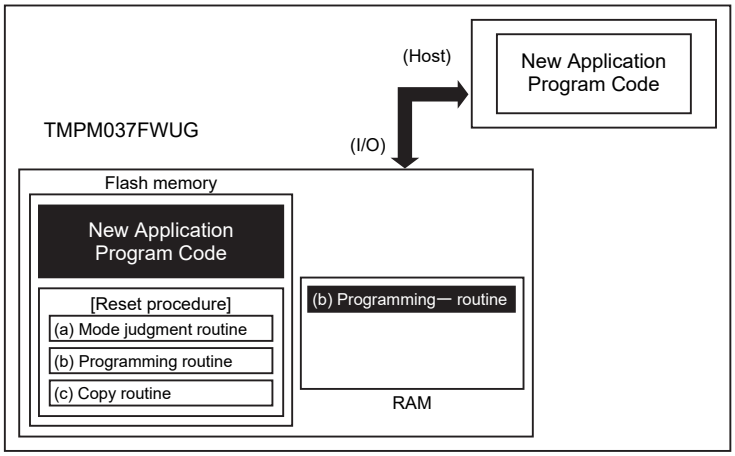
17.4.1.4 Step-4

Jump to the 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.



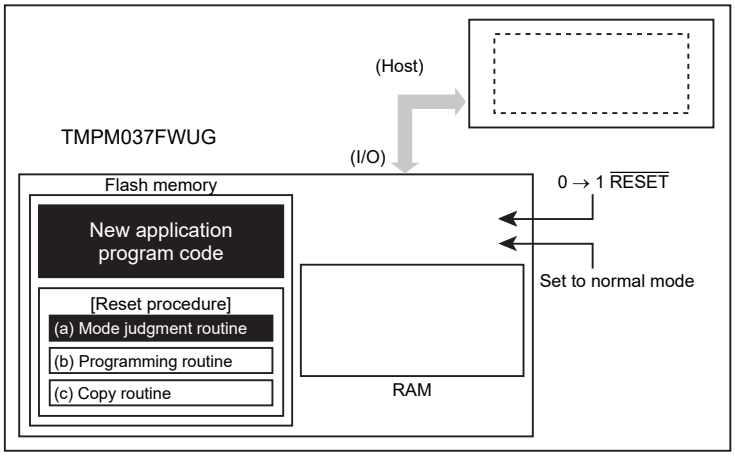
17.4.1.5 Step-5

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

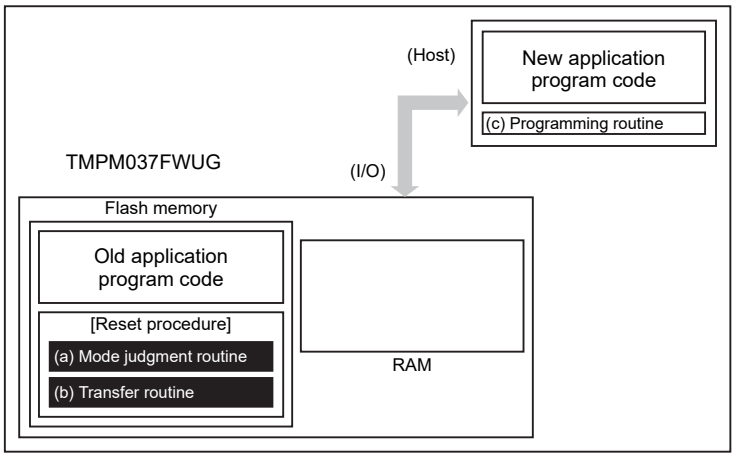
17.4.2.1 Step-1

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

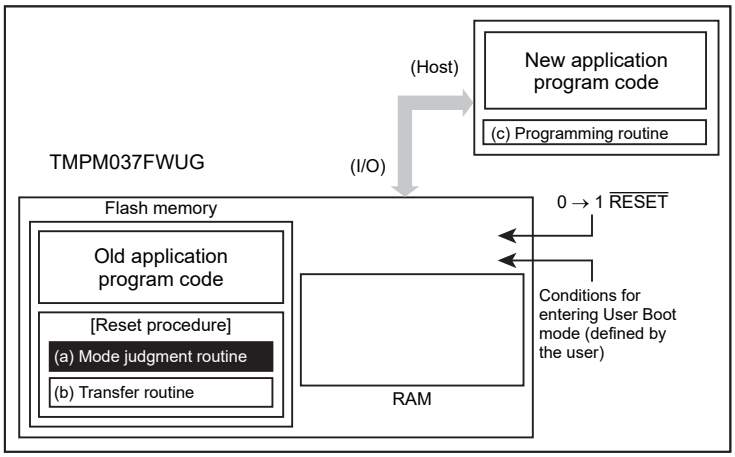
In addition, prepare a reprogramming routine shown below must be stored on the host controller.

(c) Reprogramming routine:	A program to reprogram data



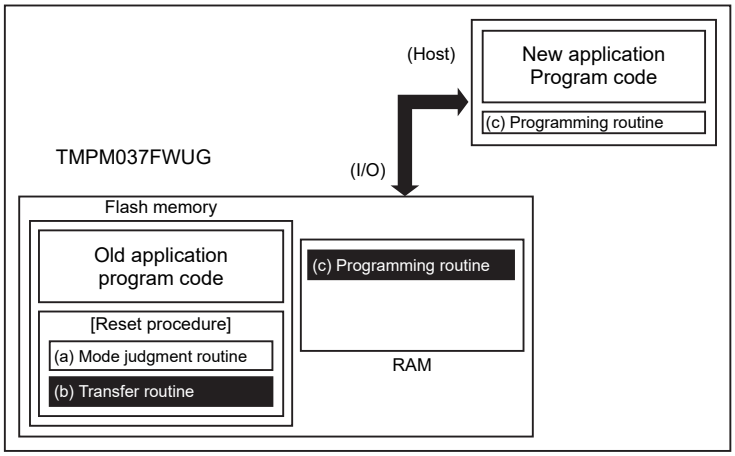
17.4.2.2 Step-2

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.



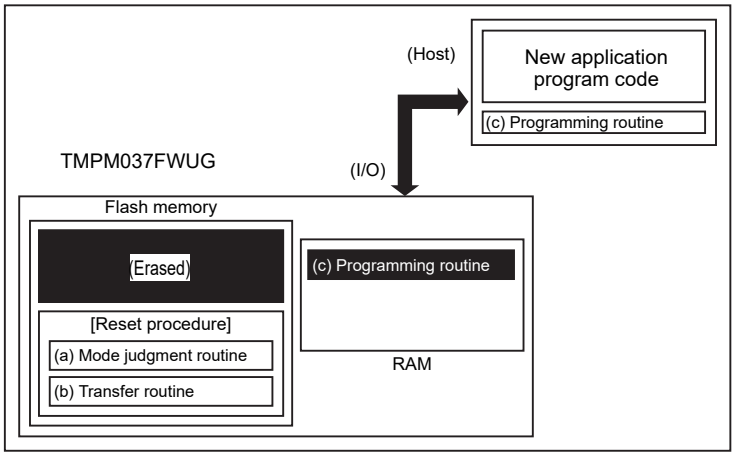
17.4.2.3 Step-3

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.



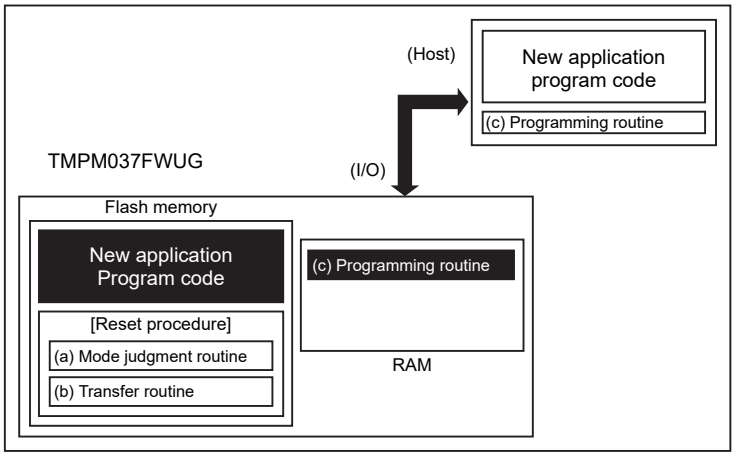
17.4.2.4 Step-4

Jump to the 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.



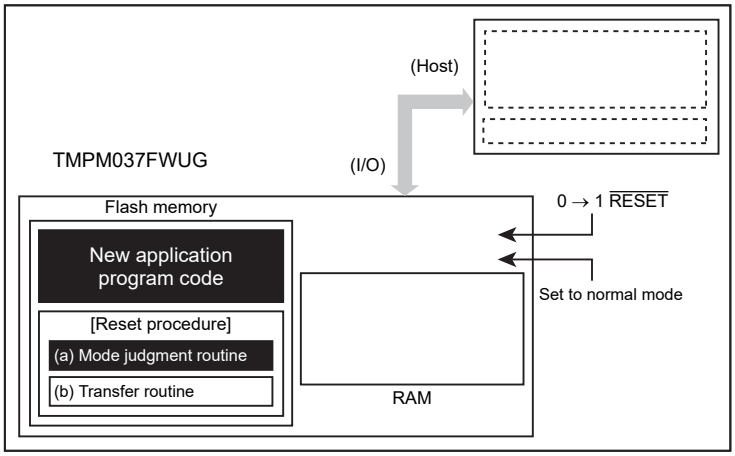
17.4.2.5 Step-5

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. Debug Interface

18.1 Specification Overview

TMPM037FWUG contains the Serial Wire JTAG Debug Port (SWJ-DP) unit for interfacing with the debugging tools .

For details about SWJ-DP, refer to "ARM documentations set for the Cortex-M0".

18.2 SWJ-DP

SWJ-DP supports the Serial Wire Debug Port (SWCLK, SWDIO) .

Pin name	Function	Description	I/O
SWDIO	SW	Serial Wire Data Input/Output	I/O
SWCLK	SW	Serial Wire Clock	Input

18.3 Peripheral Functions in Halt Mode

When the Cortex-M0 core enters in the halt mode, the watchdog-timer (WDT) automatically stops. It is selectable that 16-bit Timer(TMRB and TMR16A) continue or stop counting.Other peripheral functions continue to operate.

18.4 Connection with a Debug Tool

18.4.1 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 prohibited.

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

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

After releasing reset, the particular pins of the debug interface pins are initialized as the debug interface pins. The other debug interface pins should be changed to the debug interface pins if needed.

If the debug interface pins are used as the general I/O port, please prepare the way to change the general I/O port to the debug interface pins beforehand.

Table 18-1 Example Table of using debug interface pins

	Debug interface pins	
	SWCLK	SWDIO
SW	o	o

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

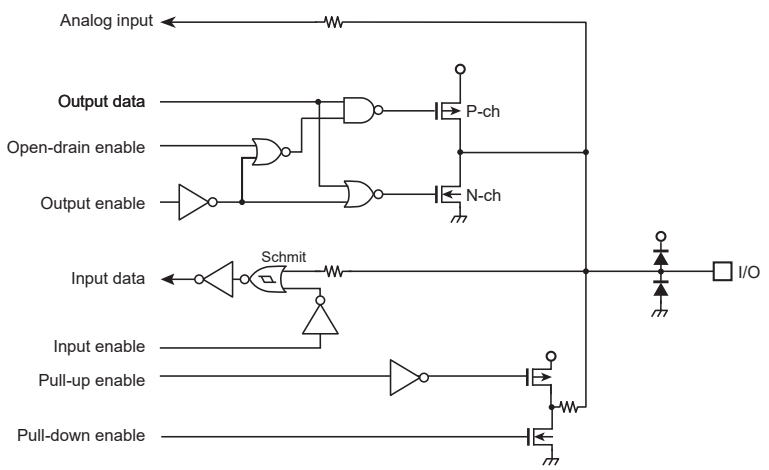
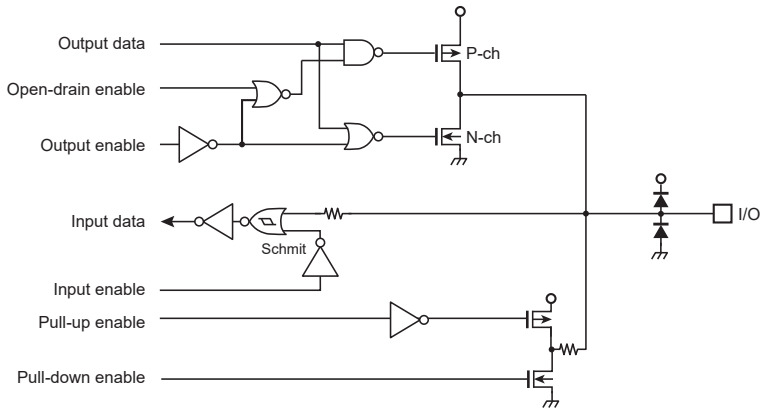
19. Port Section Equivalent Circuit Schematic

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

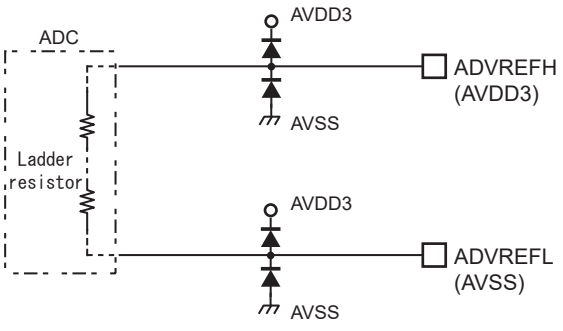
The input protection resistance ranges from several tens of Ω to several hundreds of Ω . Damping resistors X2 are shown with a typical value.

Note: Resistors without values in the figure show input protection resistors.

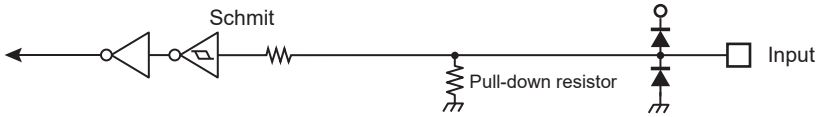
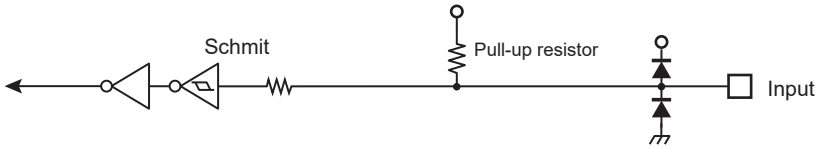
19.1 PORT pin

PA0 to PA7	
PB0 to PB7, PC0 to PC5, PD0 to PD5, PE0 to PE7, PF0 to PF7, PG0 to PG7	

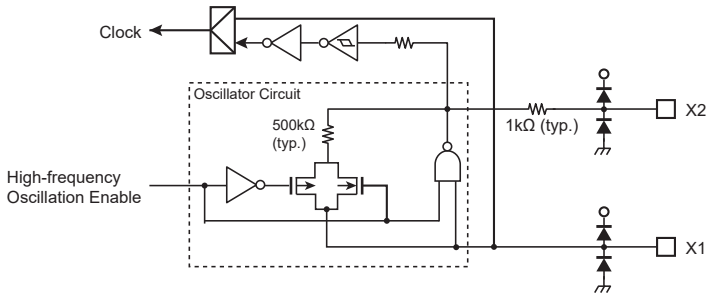
19.2 Analog pin

VREFH VREFL	

19.3 Control pin

MODE	 Mode pin must be connected to GND.
$\overline{\text{RESET}}$	

19.4 Clock pin

X1, X2	

19.5 Test pin

FTEST3	 Test pin must be open

20. Electrical Characteristics

20.1 Absolute Maximum Ratings

Parameter		Symbol	Rating	Unit
Supply voltage		DVDD3	−0.3 to 3.9	V
		RVDD3	−0.3 to 3.9	
		AVDD3	−0.3 to 3.9	
input voltage	Digital input pins	V_{IN1}	−0.3 to DVDD3 + 0.3	V
	Analog input pins	V_{IN2}	−0.3 to AVDD3 + 0.3	
Low-level Output current	Per pin(except the following)	I_{OL1}	5	mA
	PC2,PC3,PG6,PG7	I_{OL2}	20	
	Total	ΣI_{OL}	75	
High-level output current	Per pin(except the following)	I_{OH1}	−5	
	PC2,PC3,PG6,PG7	I_{OH2}	−20	
	Total	ΣI_{OH}	-75	
Power consumption (Ta = 85 °C)		PD	600	mW
Soldering temperature(10 s)		T_{SOLDER}	260	°C
Storage temperature		T_{STG}	−55 to 125	°C
Operating temperature		T_{OPR}	−40 to 85	°C

Note: **Absolute maximum ratings are limiting values of operating and environmental conditions which should not be exceeded under the worst possible conditions. The equipment manufacturer should design so that no Absolute maximum rating value is exceeded with respect to current, voltage, power consumption, temperature, etc. Exposure to conditions beyond those listed above may cause permanent damage to the device or affect device reliability, which could increase potential risks of personal injury due to IC blow up and/or burning.**

20.2 DC Electrical Characteristics (1/2)

DVSS = RVSS = AVSS = 0V
Ta = −40 to 85 °C

Parameter		Symbol	Condition	Min	Typ.	Max	Unit
Supply voltage	DVDD3 RVDD3 AVDD3	DVDD3 RVDD3 AVDD3	f_{OSC} = 8 to 16 MHz fsys = 1 to 20 MHz	2.3	−	3.6	V
Low-level Input voltage	PB0 to 7, PC0 to 5, PD0 to 5, PE0 to 7, PF0 to PF7, PG0 to 7	V_{IL1}	Schmitt	−0.3	−	0.2 DVDD3	V
	PA0 to 7	V_{IL2}				0.2 AVDD3	
	X1, MODE, $\overline{RESET}$	V_{IL3}				0.2 DVDD3	
High-level Input voltage	PB0 to 7, PC0 to 5, PD0 to 5, PE0 to 7, PF0 to PF7, PG0 to 7	V_{IH1}	Schmitt	0.8 DVDD3	−	DVDD3+0.3	V
	PA0 to 7	V_{IH2}		0.8 AVDD3		AVDD3+0.3	
	X1, MODE, $\overline{RESET}$	V_{IH3}		0.8 DVDD3		DVDD3+0.3	
Low-level output voltage	PAx,PBx,PC0to1,PC4to5,PDx, PDx,PEx,PFx,PG0to5	V_{OL1}	I_{OL1} = 2 mA 2.3 ≤ DVDD3 ≤ 3.6V	−	−	0.4	V
	PC2,PC3,PG6,PG7	V_{OL2}	I_{OL2} = 10 mA 2.3 ≤ DVDD3 ≤ 3.6V			0.4	
High-level output voltage	PAx,PBx,PC0-1,PC4to5,PDx, PDx,PEx,PFx,PG0to5	V_{OH1}	I_{OH1} = −2 mA 2.3 ≤ DVDD3 ≤ 3.6V	DVDD3-0.4	−	DVDD3	V
	PC2,PC3,PG6,PG7	V_{OH2}	I_{OH2} = −10 mA 2.3 ≤ DVDD3 ≤ 2.7V	DVDD3-0.4	−	DVDD3	

DVSS = RVSS = AVSS = 0V

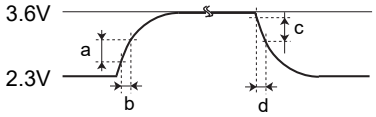
Ta = −40 to 85 °C

Parameter		Symbol	Condition	Min	Typ. (Note1)	Max	Unit
Input leakage current	−	I_{LI1}	0.0 ≤ V_{IN} ≤ DVDD3 0.0 ≤ V_{IN} ≤ AVDD3	−	0.02	±5	μA
	PC0, PC1	I_{LI2}	0.0 ≤ V_{IN} ≤ DVDD3 0 ≤ DVDD3 ≤ 0.2	−	−	±1.8	
Output leakage current		I_{LO}	0.2 ≤ V_{IN} ≤ DVDD3 -0.2 0.2 ≤ V_{IN} ≤ AVDD3 -0.2	−	0.05	±10	
Schmitt trigger input width		VTH1	2.7 V ≤ DVDD3 ≤ 3.6 V	0.1 DVDD3	−	−	V
		VTH2	2.3 V ≤ DVDD3 ≤ 2.7 V	0.1 DVDD3	−		
Pull-up resistor at Reset		RRST	2.7 V ≤ DVDD3 ≤ 3.6 V	25	50	75	kΩ
Programmable pull-up/pull-down resistor		PKH	2.7 V ≤ DVDD3 ≤ 3.6 V	25	50	75	kΩ
Power supply variation rate in operation range		VRS	RVDD3 = DVDD3	−	−	10	mV/μs
		VFS		−	−	-1.44	
Pin capacitance (Except power supply pins)		C_{IO}	fc = 1 MHz	−	−	10	pF
Low-level output current		I_{OL1}	Per pin: PAx,PBx,PC0to1,PC4to5, PDx,PEx,PF,PG0to5 2.7 V ≤ DVDD3 ≤ 3.6 V	−	−	2	mA
		I_{OL2}	Per pin: PC2,PC3,PG6,PG7 2.7 V ≤ DVDD3 ≤ 3.6 V	−	−	10	mA
		ΣI_{OL1}	Per port:PA	−	−	10	mA
		ΣI_{OL2}	Per area: PBx,PC0to1,PEx,PGx	−	−	32	mA
		ΣI_{OL3}	Per area:PC2to5,PDx,PFx	−	−	32	mA
High-level output current		I_{OH1}	Per pin: PAx,PBx,PC0to1,PC4to5, PDx,PEx,PF,PG0to5 2.7 V ≤ DVDD3 ≤ 3.6 V	−	−	-2	mA
		I_{OH2}	Per pin: PC2,PC3,PG6,PG7 2.7 V ≤ DVDD3 ≤ 3.6 V	−	−	-10	mA
		ΣI_{OH1}	Per port :PA	−	−	-10	mA
		ΣI_{OH2}	Per area: porPBx,PC0to1,PEx,PGx	−	−	-32	mA
		ΣI_{OH3}	Per area:PC2to5,PDx,PFx	−	−	-32	mA
output current		ΣI_O	total, all ports	−	−	± 60	mA

Note 1: Ta = 25 °C, DVDD3 = RVDD3 = AVDD3 = 3.3 V, unless otherwise noted.

Note 2: The same voltage must be supplied to DVDD3, RVDD3, AVDD3.

Note 3: VRS(Rising), VFS(Falling) should be measured at a strict level against a characteristics.



VRS = a / b, VFS = c / d

20.3 DC Electrical Characteristics (2/2)

Ta = −40 to 85 °C

Parameter	Symbol	condition			Min	Typ. (Note)	Max	Unit
		Operation Voltage	High-speed oscillation	Operation condition				
NORMAL	I_{DD}	DVDD3 = RVDD3 = AVDD3 = 3.6V	Enabled	All peripheral function included with CPU	−	−	13.1	mA
		DVDD3 = RVDD3 = AVDD3 = 3.3V	Refer toTable 20-1 regrading to the operation condition		−	9	12	
IDLE					−	4.5	6.5	mA
STOP1			Disabled	Refer toTable 20-1 re-grading to the operation condition	−	50	650	mA

Note: Ta = 25 °C, DVDD3 = RVDD3 = AVDD3 = 3.3 V, unless otherwise noted.

Table 20-1 I_{DD}Measurement Condition (Pin condition, Oscillator)

		NORMAL	IDLE	STOP1
Pin condition	DVDD3 = RVDD3 = AVDD3	3.3V		
	X1, X2 pin	Connected to the high-speed oscillator (10MHz)		
	Input pin	Fixed		
	Output pin	Open		
Operation condition (Oscillator)	System clock (fsys)	20MHz		Disabled
	External high-speed oscillator (EHOSC)	Enabled		Disabled
	Internal high-speed oscillator (IHOSC)	Disabled		
	PLL for fsys	Enabled (by 2)		Disabled
	peripheral function	All Enabled	Disabled	Disabled

20.4 10-bit AD Converter electrical Characteristics

AVSS = DVSS = 0V, Ta = -40 to 85 °C

Parameter		Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage (+)		AVDD	−	2.3	2.7	3.6	V
Analog input voltage		VAIN	−	AVSS	−	AVDD	V
Power supply current of analog reference voltage	AD conversion	IREF	DVSS = AVSS	−	0.45	0.7	mA
	Non-AD conversion			−	15	50	μA
INL error		−	AIN resistance ≤ 300 Ω AIN load capacitance ≥ 0.1 μF Conversion time ≥ 16.2 μs AVDD=2.7 to 3.6V	−	4.0	6.0	LSB
DNL error				−	4.0	6.0	
Zero-scale error				−	4.0	6.0	
Full-scale error				−	4.0	6.0	
Total error				−	4.0	6.0	
INL error		−	AIN resistance ≤ 300 Ω AIN load capacitance ≥ 0.1 μF Conversion time ≥ 32.4 μs AVDD=2.3 to 3.6V	−	4.0	6.0	
DNL error				−	4.0	6.0	
Zero-scale error				−	4.0	6.0	
Full-scale error				−	4.0	6.0	
Total error				−	4.0	6.0	

Note 1: 1LSB = (AVDD − AVSS)/1024 [V]

Note 2: This characteristics is shown in operating only ADC.

20.5 AC Electrical Characteristics

20.5.1 Serial channel (SIO/UART)

20.5.1.1 AC measurement Condition

The AC characteristics data of this chapter is measured under the following conditions.

- Output levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Input levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Load capacity: CL = 30pF

20.5.1.2 AC Electrical Characteristics (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.

(1) SCLK input mode

[Input]

DVDD3=2.3 to3.6V

Parameter	Symbol	Equation		fsys = 20 MHz		Unit
		Min	Max	Min	Max	
SCLK Clock High width (input)	t_{SCH}	4x	−	200	−	ns
SCLK Clocked Low width (input)	t_{SCL}	4x	−	200	−	
SCLK cycle	t_{SCY}	t_{SCH} + t_{SCL}	−	400	−	
Valid Data input ← SCLK rise/ fall (Note1)	t_{SRD}	30	−	30	−	
SCLK rise / fall (Note1) → Input Data hold	t_{HSR}	x + 30	−	80	−	

[Output]

DVDD3=2.3 to3.6V

Parameter	Sym-bol	Equation		fsys = 20 MHz		Unit
		Min	Max	Min	Max	
SCLK Clock High width (input)	t_{SCH}	4x	−	200 (Note3)	−	ns
SCLK Clocked Low width (input)	t_{SCL}	4x	−	200 (Note3)	−	
SCLK cycle	t_{SCY}	t_{SCH} + t_{SCL}	−	400	−	
Output Data ← SCLK rise or fall (Note1)	t_{OSS}	t_{SCY}/2 − 3x− 45	−	5 (Note2)	−	
SCLK rise or fall → Output Data hold (Note1)	t_{OHS}	t_{SCY}/2	−	105	−	

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 mode.

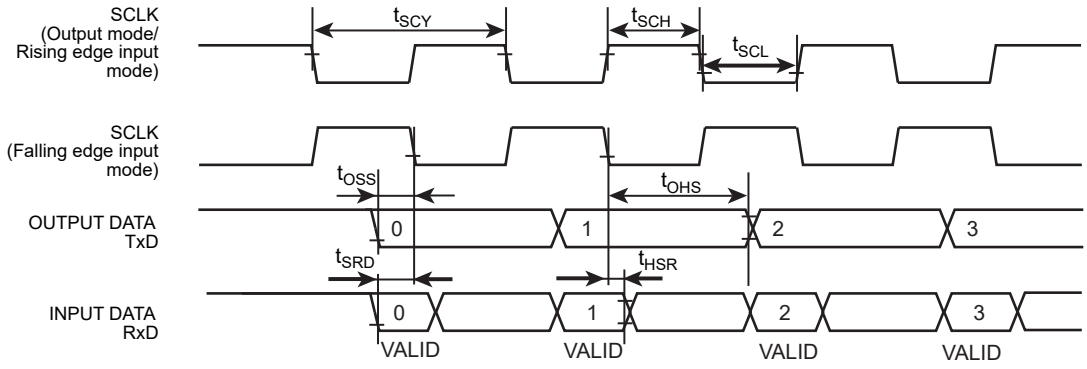
(2) SCLK Output mode

DVDD3=2.7 to3.6V

Parameter	Symbol	Equation		fsys = 20 MHz		Unit
		Min	Max	Min	Max	
SCLK cycle (programmable)	t_{SCY}	2x	−	100	−	ns
Output Data ← SCLK rise	t_{OSS}	t_{SCY}/2 - 30	−	20	−	
SCLK rise → Output Data hold	t_{OHS}	t_{SCY}/2 - 30	−	20	−	
Valid Data Input ← SCLK rise	t_{SRD}	45	−	45	−	
SCLK rise → Input Data hold	t_{HSR}	0	−	0	−	

DVDD3=2.3 to2.7V

Parameter	Symbol	Equation		fsys = 20 MHz		Unit
		Min	Max	Min	Max	
SCLK cycle (programmable)	t_{SCY}	2x	−	100	−	ns
Output Data ← SCLK rise	t_{OSS}	t_{SCY}/2 - 30	−	20	−	
SCLK rise → Output Data hold	t_{OHS}	t_{SCY}/2 - 50	−	0	−	
Valid Data Input ← SCLK rise	t_{SRD}	70	−	70	−	
SCLK rise → Input Data hold	t_{HSR}	0	−	0	−	



20.5.2 I2C interface (I2C)

20.5.2.1 AC measurement Condition

The AC characteristics data of this chapter is measured under the following conditions unless otherwise noted.

- Output levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Input levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Load capacity: CL = 30pF

20.5.2.2 AC Electrical Characteristics

In the table below, the letter x represents the I2C operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.It varies depending on the programming of the clock gear function.

n denotes the value of n programmed into the SCK (SCL output frequency select) field in the I2CxCR1.

p denotes the value of p programmed into the PRSCK(dividing clock select)field in the I2CxPRS.

Parameter	Symbol	Equation		Standard Mode		Fast Mode		Unit
		Min	Max	Min	Max	Min	Max	
SCL Clock frequency	t_{SCL}	0	−	0	100	0	400	kHz
Hold timer for START condition	$t_{HD;\ STA}$	−	−	4.0	−	0.6	−	μs
SCL Low width (Input)(Note1)	t_{LOW}	−	−	4.7	−	1.3	−	
SCL High width (Input)(Note2)	t_{HIGH}	−	−	4.0	−	0.6	−	
Setup time for a repeated START condition	$t_{SU;\ STA}$	(Note5)	−	4.7	−	0.6	−	
Data hold time (Input) (Note 3, 4)	$t_{HD;\ DAT}$	−	−	0.0	−	0.0	−	
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 an start condition	t_{BUF}	(Note5)	−	4.7	−	1.3	−	

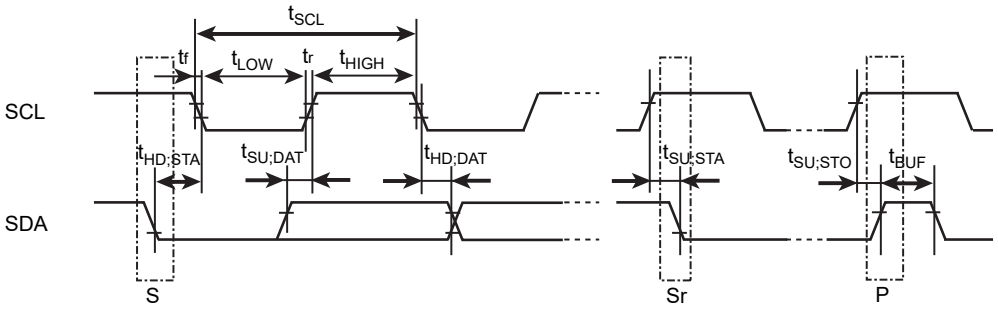
Note 1: SCL clock Low width (output): $P \times (2^{n-1} + 6)/X$

Note 2: SCL clock High width (output): $P \times (2^{n-1} + 6)/X$ On I2C-bus specification, maximum Speed of Standard Mode/fast mode is 100kHz/400khz. Internal SCL Frequency setting should comply with fsys and Note1 & Note2 shown above. (P:depend on I2CxPROS<'RSCK> ,n:depend on I2CxCR1<SCK>)

Note 3: The output data hold time is equal to 4 cycle of Prescaler clock(Tprsck) from the edge of internal SCL.

Note 4: The Philips I2C-bus specification states that a device must internally provide a hold time of at least 300 ns for the SDA signal to bridge the undefined region of the falling edge of SCL. However, this I2C does not satisfy this requirement. Also, the output buffer for SCL does not incorporate slope control of the falling edges; therefore, the equipment manufacturer should design so that the input data hold time shown in the table is satisfied, including tr/tf of the SCL and SDA lines.

Note 5: Software -dependent



S: Start condition
Sr: Re-start condition
P: Stop condition

20.5.3 16-bit Timer / Event counter (TMRB)

20.5.3.1 Event Counter

(1) AC Measurement Condition

The AC characteristics data of this chapter is measured under the following conditions.

- Input levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Load capacity: CL = 30pF

(2) AC Electrical Characteristics

In the table below, the letter x represents the TMRB operation clock cycle time. TMRB use the clock cycle as same as one of fsys. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		fsys = 20 MHz		Unit
		Min	Max	Min	Max	
Clock low pulse width	t_{VCKL}	2x + 100	−	200	−	ns
Clock high pulse width	t_{VCKH}	2x + 100	−	200	−	

20.5.3.2 Capture

(1) AC Measurement Condition

The AC characteristics data of this chapter is measured under the following conditions.

- Input levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Load capacity: CL = 30pF

(2) AC Electrical Characteristics´

In the table below, the letter x represents the TMRB operation clock cycle time.TMRB use the clock cycle as same as one of fsys. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		fsys = 20 MHz		Unit
		Min	Max	Min	Max	
Low pulse width	t_{CPL}	2x + 100	−	200	−	ns
High pulse width	t_{CPH}	2x + 100	−	200	−	

20.5.4 External Interrupt

20.5.4.1 AC Measurement Condition

The AC characteristics data of this chapter is measured under the following conditions.

- Input levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Load capacity: CL = 30pF

20.5.4.2 AC Electrical Characteristics

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

1. Except STOP1 release interrupt

Parameter	Symbol	Equation		fsys = 20 MHz		Unit
		Min	Max	Min	Max	
low-level pulse width	t_{INTAL}	x + 100	−	150	−	ns
High-level pulse width	t_{INTAH}	x + 100	−	150	−	

2. STOP1 release interrupt

Parameter	Symbol	Equation		fsys = 20 MHz		Unit
		Min	Max	Min	Max	
low-level pulse width	t_{INTBL}	500	−	500	−	ns
High-level pulse width	t_{INTBH}	500	−	500	−	

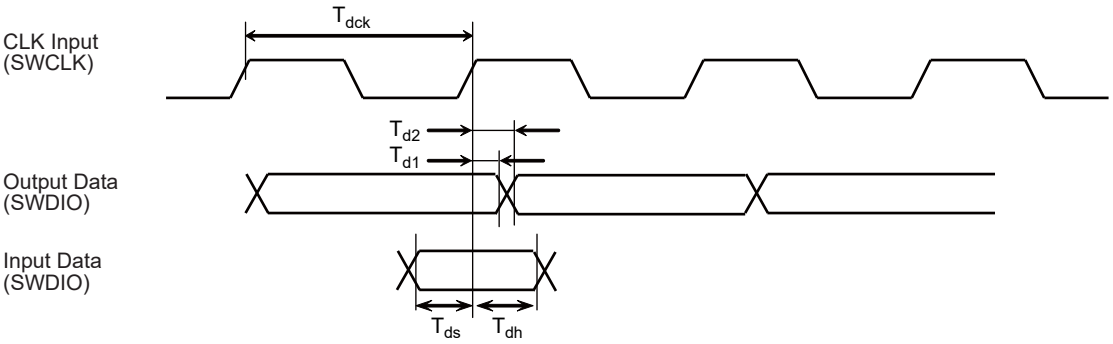
20.5.5 Debug Communication

20.5.5.1 AC Measurement Condition

- Output levels: High = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Input levels: Low = 0.8 × DVDD3, Low = 0.2 × DVDD3
- Load capacity: CL = 30pF (SWDIO)

20.5.5.2 SWD interface

Parameter	Symbol	Min	Max	Unit
CLK cycle	T_{dck}	100	−	ns
CLK rise → Output data hold	T_{d1}	4	−	
CLK rise → to output data valid	T_{d2}	−	30	
Input data valid → CLK rise	T_{ds}	20	−	
CLK rise → Input data hold	T_{dh}	15	−	



20.5.6 On-chip Oscillator Characteristic

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Oscillation frequency	IHOSC	Ta = -40 to 85°C	9.0	10	11	MHz

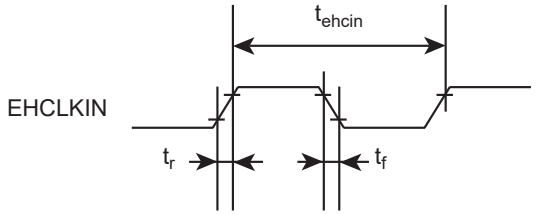
Note: Do not use an on-chip oscillator as a system clock (fsys) when high-accuracy oscillation frequency is required.

20.5.7 External Oscillator

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
High-frequency oscillation	EHOSC	Ta = -40 to 85°C	8	−	20	MHz

20.5.8 External Clock Input

Parameter	Symbol	Min	Typ.	Max	Unit
External clock frequency	t_{ehcin}	8	–	20	MHz
External clock duty	–	45	–	55	%
External clock input rise time	t_r	–	–	10	ns
External clock input fall time	t_f	–	–	10	ns



20.5.9 Flash Characteristic

Parameter	Condition	Min	Typ.	Max	Unit
Guaranteed number of Flash memory programming	DVDD3 = RVDD3 = AVDD3 = 2.7 V to 3.6 V Ta = 0 to 70°C	–	–	100	times

20.5.10 Noise Filter Characteristic

Parameter	Condition	Min	Typ.	Max	Unit
The width of noise filter	–	15	30	60	ns

20.6 Recommended Oscillation Circuit

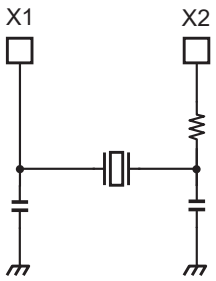


Figure 20-1 High-frequency oscillation connection

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 substrate patterns, please evaluate oscillation stability using the substrate you use.

The TMPM037FWUG has been evaluated by the oscillator vendor below. Please refer this information when selecting external parts.

20.6.1 Ceramic Oscillator

The TMPM037FWUG recommends the high-frequency oscillator by Murata Manufacturing Co., Ltd.

Please refer to the Murata Website for details.

20.6.2 Precautions for designing printed circuit board

Be sure to design printed circuit board patterns that connect a crystal unit with other oscillation elements so that the length of such patterns become shortest possible to prevent deterioration of characteristics due to stray capacitances and wiring inductance. For multi-layer circuit boards, it is important not to wire the ground and other signal patterns right beneath the oscillation circuit.For more information, please refer to the URL of the oscillator vendor.

20.7 Handling Precaution

20.7.1 Voltage level of power supply at power-on

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

TMPM037FWUG has somepower supply pin. They must be supplied the power at same time.

Power supply pin =DVDD3, AVDD3, RVDD3
Ta = -40 to 85 °C

Parameter	condition	Min	Typ.	Max	Umit
The rising of power supply in power-on	0V→2.3V to 3.6V	-	-	10	mV/μs

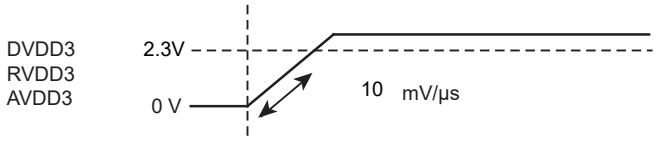


Figure 20-2 Voltage level of power supply at power-on

20.7.2 Voltage droup during operations

When the supply voltage droup occurs during operations, if the supply voltage is below the operation voltage (Brownout) , turn on power again.

20.7.3 Operating Voltage on Startup

The Operating lower Voltage of TMPM037FWUG is 2.3V, however, the detection voltage of LVD is set to 2.5V ±0.2 on first Startup, Over 2.7V Operating Voltage(DVDD3(=RVDD3)) is required on that time.

After startup, TMPM037FWUG can operate from 3.6V to the 2.3V lower Voltage when the LVD has been disabled by a program

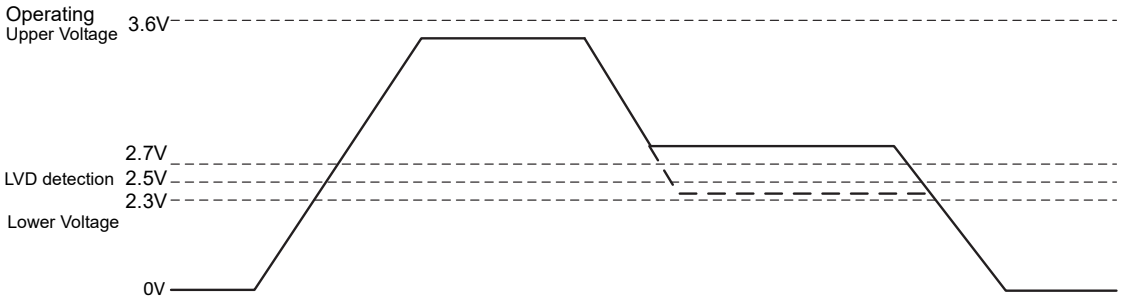


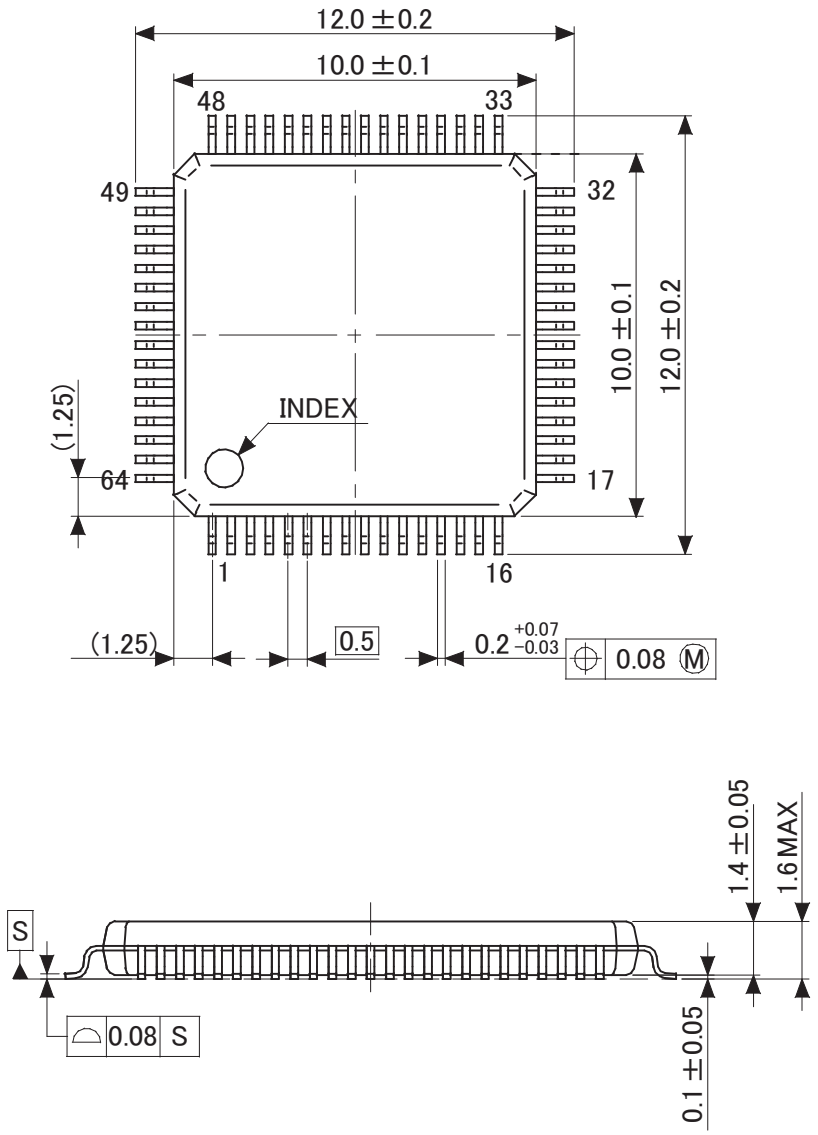
Figure 20-3 Operating Voltage on Startup

21. Package Dimensions

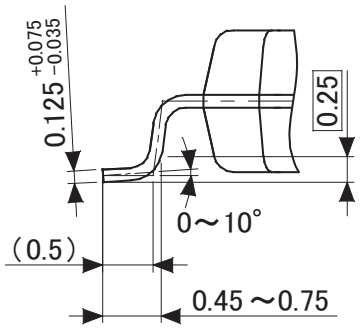
Type:LQFP64-P-1010-0.50E

Unit: mm

Dimensions



Pin detail



• RESTRICTIONS ON PRODUCT USE

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

- TOSHIBA reserves the right to make changes to the information in this document and related Product without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating 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.**