

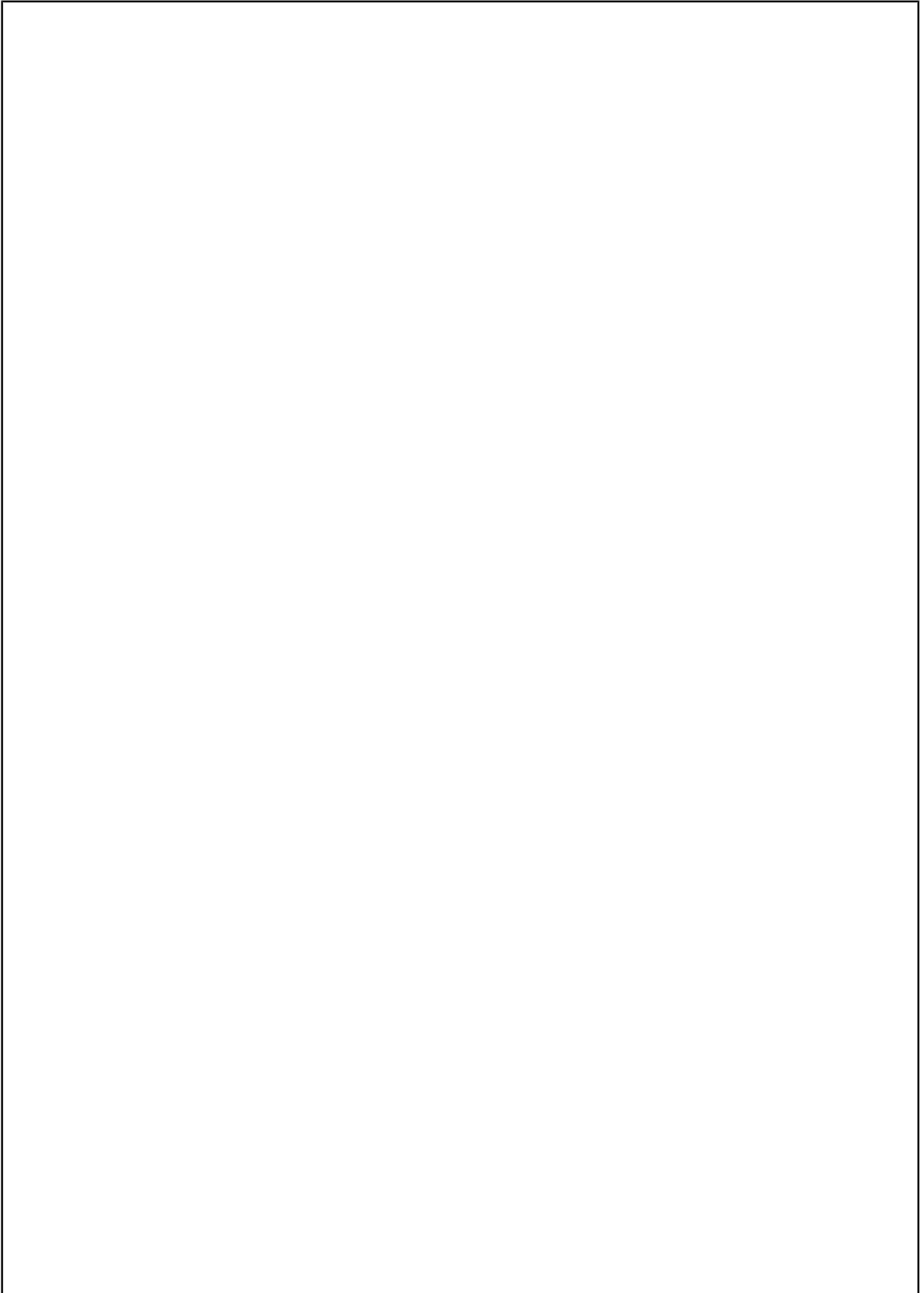
TOSHIBA

32 Bit RISC Microcontroller
TX03 Series

TMPM311CHDUG

TOSHIBA CORPORATION

Semiconductor & Storage Products Company





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



General precautions on the use of Toshiba MCUs

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

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

1. The MCUs' operation at power-on

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

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

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

2. Unused pins

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

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

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

3. Clock oscillation stability

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

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

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

The SFR addresses 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 descriptopn

Registers are described as shown below.

- Register name <Bit Symbol>
Exmample: SAMCR<MODE>="000" or SAMCR<MODE[2:0]>="000"
<MODE[2:0]> indicates bit 2 to bit 0 in bit symbol mode (3bit width).
- Register name [Bit]
Example: SAMCR[9:7]="000"
It indicates bit 9 to bit 7 of the register SAMCR (32 bit width).

Revision History

Date	Revision	Comment
2014/08/21	Tentative 1	First Release
2015/08/20	1	First Release
2023/07/31	2	Contents Revised

Table of Contents

General precautions on the use of Toshiba MCUs

TMPM311CHDUG

1.1	Functional Outline	1
1.2	A List of Peripheral Functions	2
1.3	Block Diagram	3
1.4	Pin Layout	4
1.5	Pin Information	5
1.5.1	Pin names and Functions.....	5
1.5.1.1	Peripheral functions	
1.5.1.2	Control function	
1.5.1.3	Clock	
1.5.1.4	Power supply	
1.5.2	Precautions on Power Supply.....	6
1.5.3	Pin List.....	7
1.5.3.1	The detail for pin names and function list	
1.5.3.2	PORT / Peripheral functions	
1.5.3.3	Dedicated pins	

2. Product Information (Product Info.)

2.1	μDMA Controller (μDMAC)	12
2.2	16-bit Timer/Event Counter (TMRB)	12
2.3	16-bit Timer A (TMR16A)	13
2.4	Serial Channel (SIO/UART)	13
2.5	Synchronous Serial Interface (SSP)	14
2.6	24-bit $\Delta\Sigma$ Analog / Digital Converter (DSADC)	14
2.7	Watchdog Timer (WDT)	15

3. Processor Core

3.1	Information on the processor core	17
3.2	Configurable Options	17
3.3	Exceptions/ Interruptions	18
3.3.1	Number of Interrupt Inputs.....	18
3.3.2	Number of Priority Level Interrupt Bits.....	18
3.3.3	SysTick.....	18
3.3.4	SYSRESETREQ.....	18
3.3.5	LOCKUP.....	18
3.3.6	Auxiliary Fault Status register.....	18
3.4	Events	19
3.5	Power Management	19
3.6	Exclusive access	19

<hr/>	
<hr/>	
4. Memory Map	
4.1 Bus Configuration.....	21
4.1.1 Single Chip Mode.....	21
4.2 Memory Map.....	22
4.3 Details of Memory Map.....	23
4.3.1 Code Area/SRAM Area.....	23
4.3.2 Peripheral Area.....	23
4.4 A List of Base Address for Peripheral Functions.....	24
<hr/>	
<hr/>	
5. Startup Sequence	
5.1 Without the use of the RESET pin (Reset by the Power-on-reset Circuit).....	25
5.2 Using the RESET pin.....	26
<hr/>	
<hr/>	
6. Boot Program(BOOTROM)	
6.1 Outline.....	27
6.2 Precautions.....	27
6.2.1 Setting of the Vector Table.....	27
6.2.2 Reset during the Operation.....	27
6.3 System Configuration.....	28
6.3.1 Used Pins.....	28
6.3.2 Memory Map.....	28
6.4 Operational Description.....	29
6.4.1 Whole Flowchart.....	29
6.4.2 Reception Flowchart for RAM Transfer Size Setting.....	31
6.4.3 Reception Flowchart for RAM Transfer Data.....	33
6.4.4 Data Transfer Format.....	34
6.4.5 Cecksum Calculation.....	35
<hr/>	
<hr/>	
7. Clock Control	
7.1 Outline.....	37
7.2 Schematic Diagram of Clocks.....	37
7.3 Registers.....	38
7.3.1 Register List.....	38
7.3.2 Details of the Registers.....	39
7.3.2.1 CGxPROTECT (Protect Register)	
7.3.2.2 CGxOSCSEL (High-speed Oscillation Select Register)	
7.3.2.3 CGxOSCSTF (High-speed Oscillation Status Register)	
7.3.2.4 CGxCLKCR (Clock Control Register)	
7.3.2.5 CGxOSCEN (Oscillation Enable Register)	
7.3.2.6 CGxWUHCR (High-Speed Oscillator Warm-up Control Register)	
7.4 Enabling/Disabling the Oscillators.....	45
7.4.1 Internal High-speed Oscillator.....	45
7.4.2 External High-speed Oscillator.....	45
7.5 Warm-up Timer Function.....	45
7.5.1 Source Clock Selection of the Warm-up Timer.....	45
7.5.2 Starting the Warm-up Timer.....	45
7.5.3 Calculating a Comparison Value for the Counter of the Warm-up timer.....	46
7.5.4 Confirmation of the Completion of the Warm-up Timer Operation.....	46
7.5.5 Example of the Setting of the Warm-up Timer.....	46

7.6	System Clock	47
7.7	Prescaler Clock	47
7.8	System Clock Setting after Reset	48

8. Exceptions

8.1	Overview	49
8.1.1	Exception Types.....	49
8.1.2	Handling Flowchart.....	50
8.1.2.1	Exception Request and Detection	
8.1.2.2	Exception Handling and Branch to the Interrupt Service Routine (Pre-emption)	
8.1.2.3	Executing an ISR	
8.1.2.4	Exception exit	
8.2	Reset	56
8.2.1	Factors.....	56
8.2.2	Reset Factors and Their Valid Ranges.....	56
8.2.3	Checking Reset Factors.....	56
8.3	Non-maskable Interrupt (NMI)	56
8.4	SysTick	57
8.5	Interrupts	58
8.5.1	Interrupt Request.....	58
8.5.1.1	Interrupt Path	
8.5.1.2	Generating Interrupt Requests	
8.5.2	A List of Interrupt Factors.....	60
8.5.3	Details of the Process.....	62
8.5.3.1	Process Flow	
8.5.3.2	Preparation	
8.5.3.3	Detection by Interrupt Controller	
8.5.3.4	Detection by CPU	
8.5.3.5	CPU processing	
8.5.3.6	Interrupt Service Routine (ISR)	
8.6	Exception/Interrupt-Related Registers	67
8.6.1	Register List.....	67
8.6.1.1	NVIC Register List	
8.6.1.2	INTC Register List	
8.6.2	NVIC Registers.....	68
8.6.2.1	SysTick Control and Status Register	
8.6.2.2	SysTick Reload Value Register	
8.6.2.3	SysTick Current Value Register	
8.6.2.4	SysTick Calibration Value Register	
8.6.2.5	Interrupt control registers	
8.6.2.6	Interrupt Priority Register	
8.6.2.7	Vector Table Offset Register	
8.6.2.8	Application Interrupt and Reset Control Register	
8.6.2.9	System Handler Priority Register	
8.6.2.10	System Handler Control and State Register	
8.6.3	Clock generator registers.....	82
8.6.3.1	INTxCRn (Interrupt Control Register n)	
8.6.3.2	INTxCLR (Interrupt Request Clear Register)	
8.6.3.3	INTxRESETF (Reset Flag Register)	

9. μ DMA Controller (μ DMAC)

9.1	Overview	85
9.1.1	Function List.....	85
9.2	Block Diagram	86
9.3	Registers	87
9.3.1	Register List.....	87
9.3.2	DMAXStatus (DMAC Status Register).....	88
9.3.3	DMAXCfg (DMAC Configuration Register).....	89
9.3.4	DMAXCtrlBasePtr (Channel Control Data Base-pointer Register).....	90
9.3.5	DMAXAltCtrlBasePtr (Channel Alternate Control Data Base-pointer Register).....	90

9.3.6	DMAxChnlSwRequest (Channel Software Request Register).....	91
9.3.7	DMAxChnlUseburstSet (Channel useburst Set Register).....	92
9.3.8	DMAxChnlUseburstClr (Channel useburst Clear Register).....	93
9.3.9	DMAxChnlReqMaskSet (Channel Request Mask Set Register).....	94
9.3.10	DMAxChnlReqMaskClr (Channel Request Mask Clear Register).....	95
9.3.11	DMAxChnlEnableSet (Channel Enable Set Register).....	96
9.3.12	DMAxChnlEnableClr (Channel Enable Clear Register).....	97
9.3.13	DMAxChnlPriAltSet (Channel Primary-alternate Set Register).....	98
9.3.14	DMAxChnlPriAltClr (Channel Primary-alternate Clear Register).....	99
9.3.15	DMAxChnlPrioritySet (Channel Priority Set Register).....	100
9.3.16	DMAxChnlPriorityClr (Channel Priority Clear Register).....	101
9.3.17	DMAxErrClr (Bus Error Clear Register).....	102
9.4	Operation.....	103
9.4.1	Channel Control Data Memory Map.....	103
9.4.2	Channel Control Data Structure.....	104
9.4.2.1	Final Address of the Transfer Source Data	
9.4.2.2	Final Address of the Transfer Destination Address	
9.4.2.3	Control Data Setting	
9.4.3	Operation Modes.....	106
9.4.3.1	Invalid Setting	
9.4.3.2	Basic Mode	
9.4.3.3	Automatic Request Mode	
9.4.3.4	Ping-pong Mode	
9.4.3.5	Memory Scatter/Gather Mode	
9.4.3.6	Peripheral Scatter/Gather Mode	
9.5	Precautions.....	113
9.5.1	When the SSP is Used.....	113
9.5.2	SIO/UART is Used.....	114

10. Input / Output port

10.1	Registers.....	115
10.1.1	Register list.....	116
10.1.2	Port function and setting list.....	117
10.1.2.1	PORT A	
10.1.2.2	PORT B	
10.1.2.3	PORT C	
10.1.2.4	PORT D	
10.2	Block Diagrams of Ports.....	122
10.2.1	Type FT1.....	122
10.2.2	Type FT2.....	123
10.2.3	Type FT4.....	124

11. 16-bit Timer / Event Counters (TMRB)

11.1	Outline.....	125
11.2	Block Diagram.....	126
11.3	Registers.....	127
11.3.1	Register list.....	127
11.3.2	TBxEN (Enable register).....	128
11.3.3	TBxRUN (RUN register).....	129
11.3.4	TBxCR (Control register).....	130
11.3.5	TBxMOD (Mode register).....	131
11.3.6	TBxFFCR (Flip-flop control register).....	132
11.3.7	TBxST (Status register).....	133
11.3.8	TBxIM (Interrupt mask register).....	134
11.3.9	TBxUC (Up counter capture register).....	135
11.3.10	TBxRG0 (Timer register 0).....	136
11.3.11	TBxRG1 (Timer register 1).....	136
11.3.12	TBxCP0 (Capture register 0).....	137
11.3.13	TBxCP1 (Capture register 1).....	137
11.4	Description of Operation.....	138

11.4.1	Prescaler.....	138
11.4.2	Up-counter (UC).....	138
11.4.2.1	Source clock	
11.4.2.2	Counter start / stop	
11.4.2.3	Timing to clear UC	
11.4.2.4	UC overflow	
11.4.3	Timer registers (TBxRG0, TBxRG1).....	139
11.4.4	Capture control.....	139
11.4.5	Capture registers (TBxCP0, TBxCP1).....	139
11.4.6	Up-counter capture register (TBxUC).....	140
11.4.7	Comparators (CP0, CP1).....	140
11.4.8	Timer flip-flop (TBxFF0).....	140
11.4.9	Capture interrupt (INTCAPx0, INTCAPx1).....	140
11.5	Description of Operation for each mode.....	141
11.5.1	Interval timer mode.....	141
11.5.2	Event counter mode.....	141
11.5.3	Programmable pulse generation (PPG) output mode.....	142
11.5.4	Programmable pulse generation (PPG) external trigger output mode.....	144
11.6	Applications using the capture function.....	146
11.6.1	Frequency measurement.....	146
11.6.2	Pulse width measurement.....	148

12. 16-Bit Timer A (TMR16A Ver.B)

12.1	Outline.....	151
12.2	Block Diagram.....	151
12.3	Registers.....	152
12.3.1	Register List.....	152
12.3.1.1	T16AxEN (Enable Register)	
12.3.1.2	T16AxRUN (RUN Register)	
12.3.1.3	T16AxCR (Control Register)	
12.3.1.4	T16AxRG (Timer Register)	
12.3.1.5	T16AxCP (Capture Register)	
12.4	Operation Description.....	156
12.4.1	Timer Operation.....	156
12.4.2	T16AxOUT Control.....	156
12.4.3	Read Capture.....	156
12.4.4	Automatic Stop.....	156

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

13.1	Overview.....	159
13.2	Configuration.....	160
13.3	Registers Description.....	161
13.3.1	Registers List.....	161
13.3.2	SCxEN (Enable Register).....	162
13.3.3	SCxBUF (Buffer Register).....	163
13.3.4	SCxCR (Control Register).....	164
13.3.5	SCxMOD0 (Mode Control Register 0).....	166
13.3.6	SCxMOD1 (Mode Control Register 1).....	167
13.3.7	SCxMOD2 (Mode Control Register 2).....	168
13.3.8	SCxBRCR (Baud Rate Generator Control Register).....	170
13.3.9	SCxBRADD (Baud Rate Generator Control Register 2).....	171
13.3.10	SCxFCNF (FIFO Configuration Register).....	172
13.3.11	SCxRFC (Receive FIFO Configuration Register).....	174
13.3.12	SCxTFC (Transmit FIFO Configuration Register).....	175
13.3.13	SCxRST (Receive FIFO Status Register).....	176
13.3.14	SCxTST (Transmit FIFO Status Register).....	177
13.4	Operation in Each Mode.....	178
13.5	Data Format.....	179

13.5.1	Data Format List.....	179
13.5.2	Parity Control.....	180
13.5.2.1	Transmission	
13.5.2.2	Reception	
13.5.3	STOP Bit Length.....	180
13.6	Clock Control.....	181
13.6.1	Prescaler.....	181
13.6.2	Serial Clock Generation Circuit.....	181
13.6.2.1	Baud Rate Generator	
13.6.2.2	Clock Selection Circuit	
13.7	Transmit/Receive Buffer and FIFO.....	185
13.7.1	Configuration.....	185
13.7.2	Transmit/Receive Buffer.....	185
13.7.3	Initialize Transmit Buffer.....	186
13.7.4	FIFO.....	186
13.8	Status Flag.....	187
13.9	Error Flag.....	187
13.9.1	OERR Flag.....	187
13.9.2	PERR Flag.....	188
13.9.3	FERR Flag.....	188
13.10	Receive.....	189
13.10.1	Receive Counter.....	189
13.10.2	Receive Control Unit.....	189
13.10.2.1	I/O interface mode	
13.10.2.2	UART Mode	
13.10.3	Receive Operation.....	189
13.10.3.1	Receive Buffer	
13.10.3.2	Receive FIFO Operation	
13.10.3.3	I/O interface mode with clock output mode	
13.10.3.4	Read Received Data	
13.10.3.5	Wake-up Function	
13.10.3.6	Overrun Error	
13.11	Transmit.....	193
13.11.1	Transmit Counter.....	193
13.11.2	Transmit Control.....	193
13.11.2.1	In I/O Interface Mode	
13.11.2.2	In UART Mode	
13.11.3	Transmit Operation.....	194
13.11.3.1	Operation of Transmit Buffer	
13.11.3.2	Transmit FIFO Operation	
13.11.3.3	Transmit in I/O interface Mode with Clock Output Mode	
13.11.3.4	Level of SCxTXD pin after the last bit is output in I/O interface mode	
13.11.3.5	Under-run error	
13.11.3.6	Data Hold Time In the I/O interface mode with clock input mode	
13.12	Handshake function.....	198
13.13	Interrupt/Error Generation Timing.....	199
13.13.1	Receive Interrupts.....	199
13.13.1.1	Single Buffer / Double Buffer	
13.13.1.2	FIFO	
13.13.2	Transmit interrupts.....	200
13.13.2.1	Singe Buffer / Double Buffer	
13.13.2.2	FIFO	
13.13.3	Error Generation.....	201
13.13.3.1	UART Mode	
13.13.3.2	I/O Interface Mode	
13.14	DMA Request.....	202
13.15	Software Reset.....	203
13.16	Operation in Each Mode.....	204
13.16.1	Mode 0 (I/O interface mode).....	204
13.16.1.1	Transmit	
13.16.1.2	Receive	
13.16.1.3	Transmit and Receive (Full-duplex)	
13.16.2	Mode 1 (7-bit UART mode).....	215
13.16.3	Mode 2 (8-bit UART mode).....	215
13.16.4	Mode 3 (9-bit UART mode).....	216
13.16.4.1	Wakeup function	
13.16.4.2	Protocol	

14. Synchronous Serial Port (SSP)

14.1 Overview	219
14.2 Block Diagram	220
14.3 Register	221
14.3.1 Register List.....	221
14.3.2 SSPxCR0(Control register 0).....	222
14.3.3 SSPxCR1(Control register1).....	223
14.3.4 SSPxDR(Data register).....	224
14.3.5 SSPxSR(Status register).....	225
14.3.6 SSPxCPSR (Clock prescale register).....	226
14.3.7 SSPxIMSC (Interrupt enable/disable register).....	227
14.3.8 SSPxRIS (Pre-enable interrupt status register).....	228
14.3.9 SSPxMIS (Post-enable interrupt status register).....	229
14.3.10 SSPxICR (Interrupt clear register).....	230
14.3.11 SSPxDMACR (DMA control register).....	230
14.4 Overview of SSP	231
14.4.1 Clock prescaler.....	231
14.4.2 Transmit FIFO.....	231
14.4.3 Receive FIFO.....	231
14.4.4 Interrupt generation logic.....	232
14.4.5 DMA Interface.....	233
14.4.5.1 Burst Transfer	
14.4.5.2 Single Transfer	
14.5 SSP operation	235
14.5.1 Initial setting for SSP.....	235
14.5.2 Enabling SSP.....	235
14.5.3 Clock ratios.....	235
14.6 Frame Format	236
14.6.1 SSI frame format.....	237
14.6.2 SPI frame format.....	238
14.6.3 Microwire frame format.....	242

15. 24-bit $\Delta\Sigma$ Analog/Digital Converter (DSADC)

15.1 Features	245
15.1.1 Pin Treatment.....	246
15.2 Block Diagram	247
15.3 Registers	248
15.3.1 Register List.....	248
15.3.2 Details of Registers.....	249
15.3.2.1 DSADxCLK (Clock Setting Register)	
15.3.2.2 DSADxCR0 (Control Register 0)	
15.3.2.3 DSADxCR1 (Control Register 1)	
15.3.2.4 DSADxCR2 (Control Register 2)	
15.3.2.5 DSADxCR3 (Control Register 3)	
15.3.2.6 DSADxCR4 (Control Register 4)	
15.3.2.7 DSADxCR5 (Control Register 5)	
15.3.2.8 DSADxADJ (Correction Register)	
15.3.2.9 DSADxST (Conversion Status Register)	
15.3.2.10 DSADxRES (Conversion Result Stored Register)	
15.4 Operation Description	257
15.4.1 Startup and Stop Procedures.....	257
15.4.1.1 Startup	
15.4.1.2 Stop	
15.4.2 Conversion Clock (ADCLK).....	259
15.4.2.1 Conversion Time	
15.4.3 Conversion Mode.....	259
15.4.4 Starting Conversion.....	259
15.4.5 Conversion Status.....	259

15.4.6	Switching the Conversion Object.....	260
15.4.7	Stopping Conversion.....	260
15.4.8	Conversion End.....	260
15.4.9	Conversion Result.....	260
15.5	Synchronous Start Function.....	261
15.5.1	Startup.....	261
15.5.2	Stop.....	262
15.6	Conversion Start Correction Function.....	262

16. Temperature Sensor (TEMP)

16.1	Outline.....	263
16.2	Block diagram.....	263
16.3	Registers.....	264
16.3.1	Register List.....	264
16.3.2	Details of Register.....	264
16.3.2.1	TEMPEN (Enable register)	
16.3.2.2	TEMPCR (Control Register)	
16.4	Operation Description.....	266

17. Power-on-Reset Circuit (POR)

17.1	Configuration.....	267
17.2	Function.....	267
17.2.1	Operation at Power-on.....	267
17.2.2	Operation at Power-down.....	267
17.2.3	Operation at re-power-on.....	268

18. Watchdog Timer(WDT)

18.1	Configuration.....	269
18.2	Register.....	270
18.2.1	Register List.....	270
18.2.2	WDxMOD(Watchdog Timer Mode Register)	270
18.2.3	WDxCR (Watchdog Timer Control Register).....	271
18.2.4	WDxFLG (Watchdog Flag Register).....	271
18.3	Description of Operation.....	272
18.3.1	Basci Operation.....	272
18.3.2	Operation Status.....	272
18.3.3	Operation when malfunction(runway) is detected.....	273
18.3.3.1	INTWDTx interrupt generation	
18.3.3.2	Internal Resetgeneration	
18.4	Control of the watchdog timer.....	274
18.4.1	Register access.....	274
18.4.2	Disable control.....	274
18.4.3	Enable control.....	274
18.4.4	Watchdog timer clearing control.....	274
18.4.5	Detection time of watchdog timer.....	274

19. Port Section Equivalent Circuit Schematic

19.1	PORT pin.....	275
19.2	Analog pin.....	275
19.3	Control pin.....	276

19.4	Clock pin	276
-------------	------------------------	------------

20. Electrical Characteristics

20.1	Absolute Maximum Ratings	277
20.2	DC Electrical Characteristics (1/2)	278
20.3	DC Electrical Characteristics (2/2)	279
20.4	24-bit $\Delta\Sigma$ADC Electrical Characteristics	280
20.5	Temperature Sensor Characteristics	280
20.6	AC Electrical Characteristics	281
20.6.1	Serial Channel (SIO/UART).....	281
20.6.1.1	AC measurement condition	
20.6.1.2	I/O Interface mode	
20.6.2	Synchronous serial Interface (SSP).....	283
20.6.2.1	AC measurement conditions	
20.6.2.2	AC Electrical Characteristics	
20.6.3	16-bit Timer / Event counter (TMRB).....	287
20.6.3.1	Event Counter	
20.6.3.2	Capture	
20.6.4	External Interrupt.....	288
20.6.4.1	AC measurement conditions	
20.6.4.2	AC Electrical Characteristics	
20.6.5	24-bit $\Delta\Sigma$ ADC Trigger Input pin AC Characteristics.....	288
20.6.5.1	AC measurement conditions	
20.6.5.2	AC Electrical Characteristics	
20.6.6	On chip oscillator.....	289
20.6.7	External Oscillator.....	289
20.6.8	External Clock Input.....	289
20.6.9	Noise Filter Characteristic.....	289
20.7	Recommended Oscillation Circuit	290
20.7.1	Ceramic Oscillator.....	290
20.7.2	Crystal Oscillator.....	290
20.7.3	Precautions for designing printed circuit board.....	290

21. Package Dimensions



CMOS 32-Bit Microcontroller

TMPM311CHDUG

TMPM311CHDUG is a 32-bit RISC microprocessor containing ARM Cortex®-M3.

The functional outline and features are as follows:

1.1 Functional Outline

1. ARM Cortex-M3 core
 - Improves code efficiency using Thumb®-2 instructions
 - Achieves high-performance and low-power consumption
 - High-speed interrupt response suitable for real-time control
2. Internal program memory/data memory
 - Program memory (RAM): 16KB
 - Data memory (RAM): 5KB
3. External interrupt function: 2 (External interrupt pins)
 - 7-level priority is settable.
4. Power-on-Reset Circuit (POR)
5. Endianness: Little-endian
6. Maximum operating frequency: 24MHz
7. Operating voltage range: 2.7 to 3.6V
8. Temperature range: -40°C to 85°C
9. Package: LQFP48 (7mm × 7mm, 0.5mm pitch)

1.2 A List of Peripheral Functions

Peripheral function	Description	Unit A	Unit
Clock control (CG)	<ul style="list-style-type: none"> Internal high-speed oscillator (10MHz) External high-speed oscillator: <ul style="list-style-type: none"> -Oscillator connection (8 to 20MHz) -Clock input (8 to 24MHz) Clock gear function: <ul style="list-style-type: none"> Divides the high-speed clock to 1/1, 1/2, 1/4, 1/8 or 1/16 	1	Channel
μ DMA controller (μ DMA)	<ul style="list-style-type: none"> Supports 5 transfer modes. Maximum transfers: 1024 	32	Channel
Input/output port (PORT)	<ul style="list-style-type: none"> Input/output port Function setting, pull-up/down selection 	23	pin
16-bit timer (TMRB)	<ul style="list-style-type: none"> 16-bit interval timer mode 16-bit event counter mode 16-bit PPG output (Synchronous output is possible on multiple channels) Event capture function 	4	Channel
16-bit timer (TMR16A)	<ul style="list-style-type: none"> 16-bit compare interrupt Read capture function 	1	Channel
Watchdog timer (WDT)	<ul style="list-style-type: none"> A reset or non-maskable interrupt (NMI) occurs. 	1	Channel
Serial channel (SIO/UART)	<ul style="list-style-type: none"> Selectable from the UART or synchronous communication mode. Built-in 4-byte FIFO for transmission/reception. 	1	Channel
Synchronous serial bus interface (SSP)	<ul style="list-style-type: none"> Supports SPI, SSI, and Microwire formats Communication rate <ul style="list-style-type: none"> -Master mode: $f_{sys}/2$ to $f_{sys}/65024$ -Slave mode: $f_{sys}/12$ to $f_{sys}/65024$ 	1	Channel
24-bit $\Delta\Sigma$ AD converter (DSADC)	<ul style="list-style-type: none"> Minimum conversion time: 112μs Input voltage range: -0.375 to +1V Conversion mode: Single or repeat Synchronous start of multiple units 	4	Unit
Temperature sensor (TEMP)	<ul style="list-style-type: none"> By measuring at several temperature conditions, it can be measured in relative temperature. 	1	Unit

1.3 Block Diagram

Figure 1-1 shows the block diagram of TMPM311CHDUG.

AHB bus has a multi-layer structure. For details, refer to the chapter on "Memory Map."

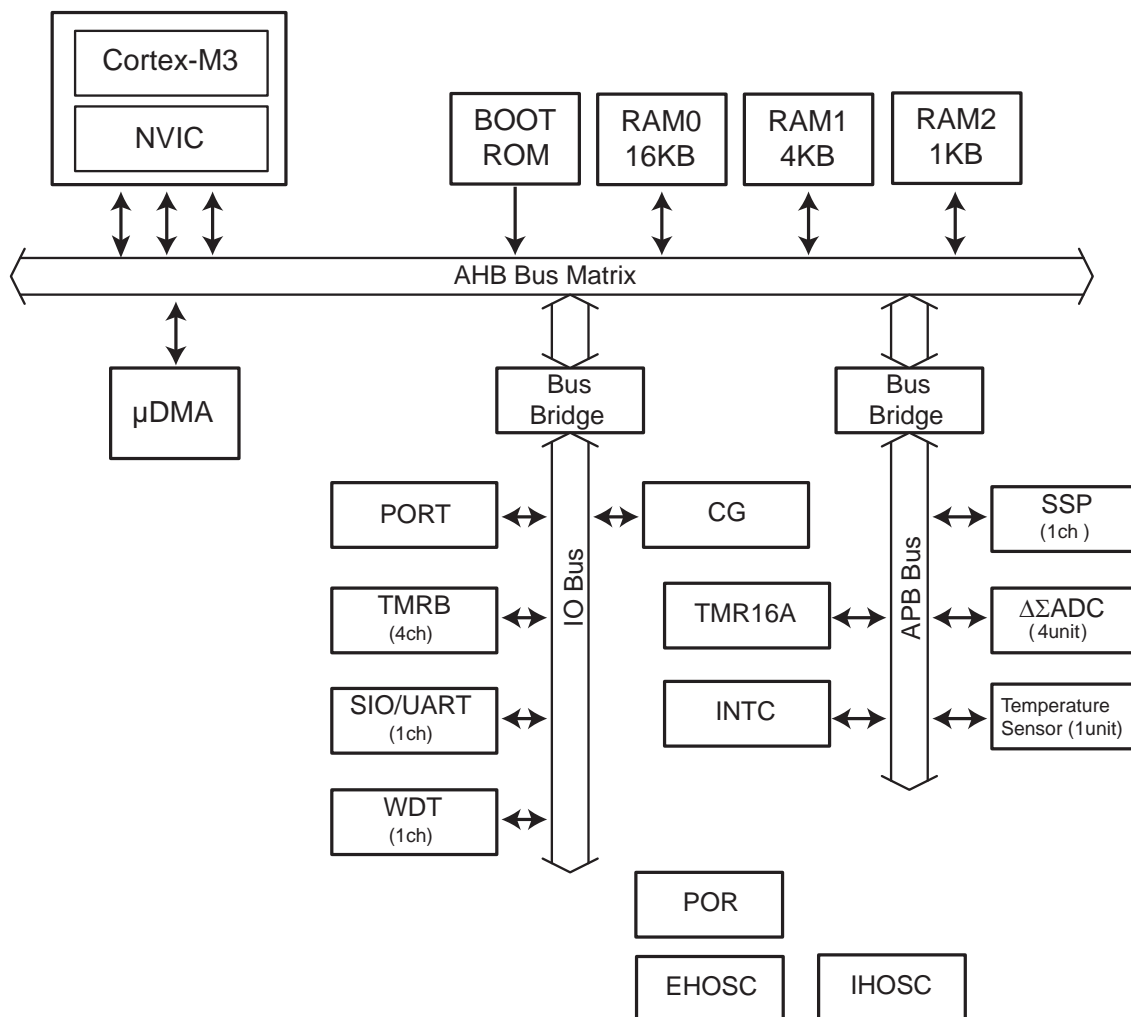


Figure 1-1 Block diagram

1.4 Pin Layout

Figure 1-2 shows the pin layout of TMPM311CHDUG.

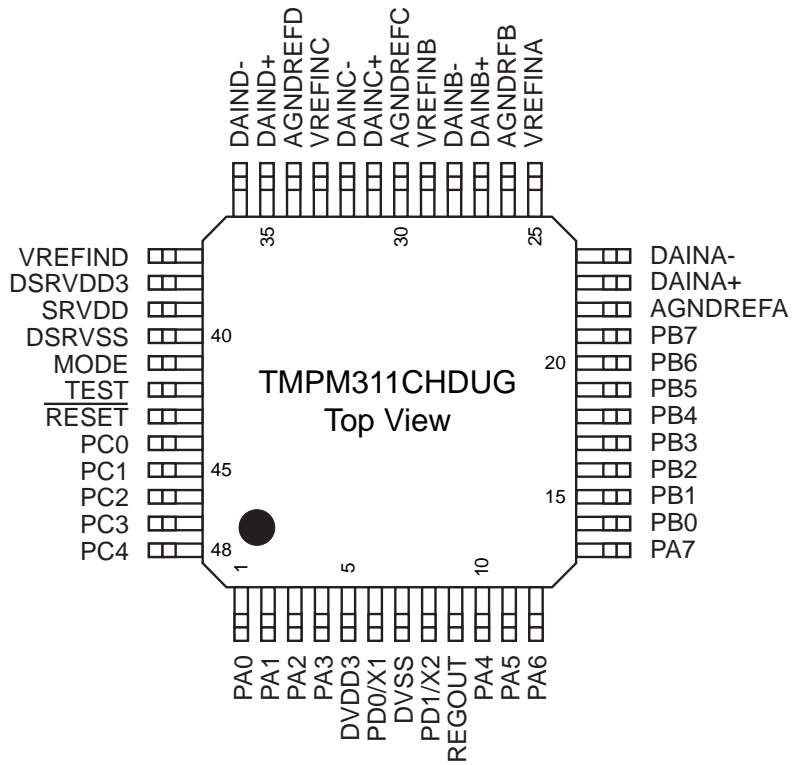


Figure 1-2 Pin Layout

1.5 Pin Information

1.5.1 Pin names and Functions

1.5.1.1 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.).
TMRB	TBxIN	Input	Input capture input pin
	TBxOUT	Output	PPG output pin
SIO/UART	SCxTXD	Output	Data output pin
	SCxRXD	Input	Data input pin
	SCxSCLK	I/O	Clock input / output pin
SSP	SPxDO	Output	Data output pin
	SPxDI	Input	Data input pin
	SPxCLK	I/O	Clock input / output pin
	SPxFSS	I/O	Frame / slave select input / output pin
DSADC	DAINx+/-	Input	Analog input pin
	DSADEOC	Output	Conversion end
	DSADTRG	Input	External trigger input pin

1.5.1.2 Control function

Table 1-2 Pin name and functions

Pin name	Input or Output	Function
RESET	Input	Reset signal input pin
MODE	Input	MODE pin This pin must be fixed to Low level.
TEST	Input	Test pin This pin must be fixed to Low level.

1.5.1.3 Clock

Table 1-3 Pin name and functions

Pin name	Input or Output	Function
X1	Input	High frequency resonator connection pin
X2	Output	High frequency resonator connection pin

1.5.1.4 Power supply

Table 1-4 Pin name and functions

Power supply pin name	Function
REGOUT	Pin connected with the capacitor (1.0 μ F 20%) for the regulator
DVDD3	Power supply pin for the digital circuit DVDD3 supplies the following pins. PA 0 to 7, PB0 to 7, PC0 to 4, PD0 to 1, MODE, TEST, $\overline{\text{RESET}}$
DVSS	GND pin for the digital circuit
SRVDD	Supplying the voltage reference circuit with a power supply.
DSRVSS	Voltage reference circuit: GND pin
DSRVDD3	Supplying the amplifier circuit for the $\Delta\Sigma$ ADC with a power supply.
VREFINx	Supplying the 24bit $\Delta\Sigma$ ADC with a power supply.
AGNDREFx	24bit $\Delta\Sigma$ ADC: GND pin.

1.5.2 Precautions on Power Supply

Power should be supplied to the TMPM311CHDUG based on any one of the following connection method. Capacitors for regulators should be connected as shown below:

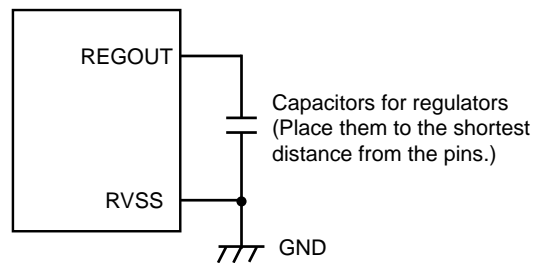


Figure 1-3 Capacitor for a regulator connection circuit

1.5.3 Pin List

1.5.3.1 The detail for pin names and function list

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

- SMT/CMOS : Type of input gate
 - SMT : Schmitt input
 - CMOS : CMOS input
- 5V_T : 5V tolerant support
 - Yes : supported
 - N/A : Not supported
- OD : Programmable open drain output support
 - Yes : supported
 - N/A : Not supported
- PU/PD : Programmable Pull-Up / Pull-Down
 - PU : Programmable Pull-Up supported
 - PD : Programmable Pull-Down supported

1.5.3.2 PORT / Peripheral functions

Table 1-5 Pin list <Sorted by Port>

Pin No.	PORT	Function A	Function B					Port Specification			
			1	2	3	4	5	PU/PD	OD	5V_T	SMT/CMOS
PORTA											
1	PA0		SP0CLK					PU	N/A	N/A	SMT
2	PA1		SP0DO					PU	N/A	N/A	SMT
3	PA2		SP0DI					PU	N/A	N/A	SMT
4	PA3		SP0FSS					PU	N/A	N/A	SMT
10	PA4		SC0SCLK					PU	N/A	N/A	SMT
11	PA5		SC0RXD					PU	N/A	N/A	SMT
12	PA6	INT1	SC0TXD					PU	N/A	N/A	SMT
13	PA7							PU	N/A	N/A	SMT
PORTB											
14	PB0		TB0IN					PU	N/A	N/A	SMT
15	PB1		TB0OUT					PU	N/A	N/A	SMT
16	PB2		TB1IN					PU	N/A	N/A	SMT
17	PB3		TB1OUT					PU	N/A	N/A	SMT
18	PB4		DSADEOC					PU	N/A	N/A	SMT
19	PB5		DSADTRG					PU	N/A	N/A	SMT
20	PB6							PU	N/A	N/A	SMT
21	PB7							PU	N/A	N/A	SMT
PORTC											
44	PC0	INT0						PU	N/A	N/A	SMT
45	PC1							PU	N/A	N/A	SMT
46	PC2							PU	N/A	N/A	SMT
47	PC3							PU/PD	N/A	N/A	SMT
48	PC4							PU	N/A	N/A	SMT
PORTD											
6	PD0	X1						PU	N/A	N/A	SMT
8	PD1	X2						PU	N/A	N/A	SMT

1.5.3.3 Dedicated pins

(1) Peripheral function pins

Table 1-6 Pin number and Pin name

Pin No.	Pin name
23	DAINA+
24	DAINA-
27	DAINB+
28	DAINB-
31	DAINC+
32	DAINC-
35	DAIND+
36	DAIND-

(2) Control pins

Table 1-7 Pin number and Pin name

Pin No.	Pin name
41	MODE
42	TEST
43	RESET

(3) Power supply pins

Table 1-8 Pin number and Pin name

Pin No.	Pin name
5	DVDD3
7	DVSS
9	REGOUT
39	SRVDD
40	DSRVSS
38	DSRVDD3
25	VREFINA
22	AGNDREFA
29	VREFINB
26	AGNDREFB
33	VREFINC
30	AGNDREFC
37	VREFIND
34	AGNDREFD

2. Product Information (Product Info.)

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 μ DMA Controller (μ DMAC)"
- "2.2 16-bit Timer/Event Counter (TMRB)"
- "2.3 16-bit Timer A (TMR16A)"
- "2.4 Serial Channel (SIO/UART)"
- "2.5 Synchronous Serial Interface (SSP)"
- "2.6 24-bit $\Delta\Sigma$ Analog / Digital Converter (DSADC)"
- "2.7 Watchdog Timer (WDT)"

2.1 μ DMA Controller (μ DMAC)

TMPM311CHDUG contains 1 unit of built-in μ DMA controller. (Unit A)

Table 2-1 μ DMA Request Table

Channel	Burst	Single
0	SSP0 reception	SSP0 reception
1	SSP0 transmission	SSP0 transmission
2	SIO/UART0 reception	-
3	SIO/UART0 transmission	-
Other than those above	-	-

2.2 16-bit Timer/Event Counter (TMRB)

TMPM311CHDUG contains 4 channels of TMRB.

Differences in each channel are as follows.

Table 2-2 Pin specifications (x: Channel number)

Channel	TBxOUT	TBxIN
TMRB0	PB1	PB0
TMRB1	PB3	PB2
TMRB2	-	-
TMRB3	-	-

Table 2-3 Synchronous start specification

Master channel	Slave channel
TMRB0	TMRB1, TMRB2, TMRB3

Table 2-4 Capture trigger specification

Trigger input channel	Trigger output
TMRB1 TMRB2 TMRB3	TB0OUT

In the TMPM311CHDUG, the following function of the TMRB has no meaning. Write "0" to the related register.

Table 2-5 Non-Usable Function (x: Channel number)

Function	Register
Low-power consumption mode operation function	TBxCR<I2TB>

2.3 16-bit Timer A (TMR16A)

TMPM311CHDUG contains 1 channel of TMR16A and does not provide output signal to the rectangular wave pin.

Table 2-6 Pin specification (x: Channel number)

Channel	T16AxOUT
TMR16A0	-

In the TMPM311CHDUG, the following function of the TMR16A has no meaning. Write "0" to the related register.

Table 2-7 Non-Usable Function

Function	Register
Low-power consumption mode operation function	T16A0EN<I2T16A>

2.4 Serial Channel (SIO/UART)

TMPM311CHDUG contains 1 channel of SIO/UART.

Table 2-8 Pin specifications (x: Channel number)

Channel	SCxTXD	SCxRXD	SCxSCLK	SCxCTS
SC0	PA6	PA5	PA4	-

In the TMPM311CHDUG, the following functions of the SIO/UART have no meaning. Write "0" to the related registers.

Table 2-9 Non-Usable Functions

Function	Register
Wake-up function using serial link	SC0MOD0<WU>
Handshake function using the CTS pin	SC0MOD0<CTSE>
Low-power consumption mode operation function	SC0MOD1<I2SC>

In the TMPM311CHDUG, a TMRB output can not be selected as transfer clock. Therefore, specifying "00" to SCxMOD0<SC> is prohibited.

2.5 Synchronous Serial Interface (SSP)

TMPM311CHDUG contains 1 channel of SSP.

Table 2-10 Pin specifications (x: Channel number)

Channel	SPxDO	SPxDI	SPxCLK	SPxFSS
SSP0	PA1	PA2	PA0	PA3

2.6 24-bit $\Delta\Sigma$ Analog / Digital Converter (DSADC)

TMPM311CHDUG contains 4 units of DSADC.

In the synchronous start function of DSADC, the following table is an assignment of a master unit and slave unit.

Table 2-11 Master/slave assignment

Master	Slave
Unit A	Unit B Unit C Unit D

Table 2-12 Hardware Trigger assignment

Unit	Hardware Trigger	
	External Trigger	Internal Trigger
Unit A	$\overline{\text{DSADTRG}}$	-
Unit B		
Unit C		
Unit D		

Table 2-13 Analog Input assignment

Unit	Analog Input	
	External Analog Input	Internal Analog Input
Unit A	DAINA+ / DAINA-	-
Unit B	DAINB+ / DAINB-	-
Unit C	DAINC+ / DAINC-	-
Unit D	DAIND+ / DAIND-	Temperature Sensor

2.7 Watchdog Timer (WDT)

In the TMPM311CHDUG, the following function of the WDT has no meaning. Write "0" to the related register.

Table 2-14 Non-Usable Function

Function	Register
Low-power consumption mode operation function	WD0MOD<I2WDT>

3. Processor Core

The TX03 series has a high-performance 32-bit processor core (the ARM Cortex-M3 processor core). For information on the operations of this processor core, please refer to the documentation set issued by ARM Limited. This chapter describes the functions unique to the TX03 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 TMPM311CHDUG.

Refer to the detailed information about the CPU core and architecture, refer to the ARM documentation set for "the Cortex-M3 series processors" in the following URL:

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

Product Name	Core Revision
TMPM311CHDUG	r2p1

3.2 Configurable Options

The Cortex-M3 core has optional blocks. The following table shows the configurable options in the TMPM311CHDUG.

Feature	Configure option
FPB	Two literal comparators Six instruction comparators
DWT	Four comparators
ITM	Present
MPU	Absent
ETM	Present
AHB-AP	Present
AHB Trace Macrocell Interface	Absent
TPIU	Present
WIC	Absent
Debug Port	Absent
Bit Band	Present
constant AHB control	Absent

3.3 Exceptions/ Interruptions

Exceptions and interruptions are described in the following section.

3.3.1 Number of Interrupt Inputs

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

TMPM311CHDUG has 23 interrupt inputs. The number of interrupt inputs is reflected in <INTLINESNUM[4:0]> bit of NVIC register. In this product, if read <INTLINESNUM[4:0]> bit, 0x0 is read out.

3.3.2 Number of Priority Level Interrupt Bits

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

TMPM311CHDUG has 3 priority level interrupt bits. The number of priority level interrupt bits is used for assigning a priority level in the interrupt priority registers and system handler priority registers.

3.3.3 SysTick

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

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

3.3.4 SYSRESETREQ

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

TMPM311CHDUG provides the same operation when SYSRESETREQ signal are output.

3.3.5 LOCKUP

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

TMPM311CHDUG does not use this signal. To return from LOCKUP status, it is necessary to use non-maskable interrupt (NMI) or reset.

3.3.6 Auxiliary Fault Status register

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

However, TMPM311CHDUG is not defined this function. If auxiliary fault status register is read, always "0x0000_0000" is read out.

3.4 Events

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

TMPM311CHDUG 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-M3 core provides power management system which uses SLEEPING signal and SLEEPDEEP signal. SLEEPDEEP signals are output when <SLEEPDEEP> bit of System Control Register is set.

These signals are output in the following circumstances:

-Wait-For-Interrupt (WFI) instruction execution

-Wait-For-Event (WFE) instruction execution

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

TMPM311CHDUG does not support low-power consumption mode so that please do not use WFI instruction and WFE instruction.

3.6 Exclusive access

In Cortex-M3 core, the DCode bus system supports exclusive access. However TMPM311CHDUG does not use this function.

4. Memory Map

This chapter describes the bus configuration and memory map.

4.1 Bus Configuration

TMPM311CHDUG incorporates two types of bus masters such as the CPU core and μ DMA controller.

The bus masters connect to slave ports (S0 to S3) of the bus matrix. In the bus matrix, the slave ports connect to the master ports (M0 to M3 and M4 to M7) via nodes described as symbols (o, •). The master ports connect to the peripheral functions. • means the connection to the mirror area.

When multiple slaves connect on the same bus master line in the bus matrix, if multiple slaves are accessed simultaneously, a smaller slave number of the bus master is given a higher priority.

4.1.1 Single Chip Mode

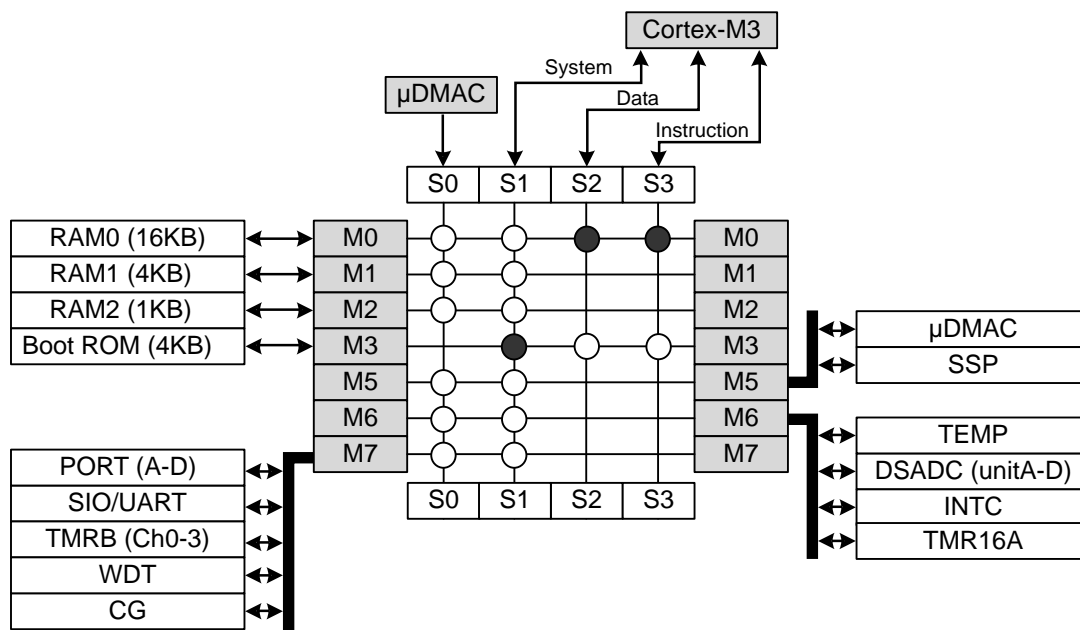


Figure 4-1 Bus configuration

4.2 Memory Map

The memory maps of TMPM311CHDUG are based on the memory map of ARM Cortex-M3 core. The Special Function Register (SFR) is the control register of the input/output ports and peripheral functions.

The CPU register areas mean the processor core's internal register areas.

For details of each area, refer to the documentation set issued by ARM Limited.

Note that accessing the areas indicated as "Fault" causes a memory fault when memory faults are enabled; when memory faults are disabled, it causes a hard fault. Also, do not access the vendor-specific area.

Figure 4-2 shows the memory map of TMPM311CHDUG.

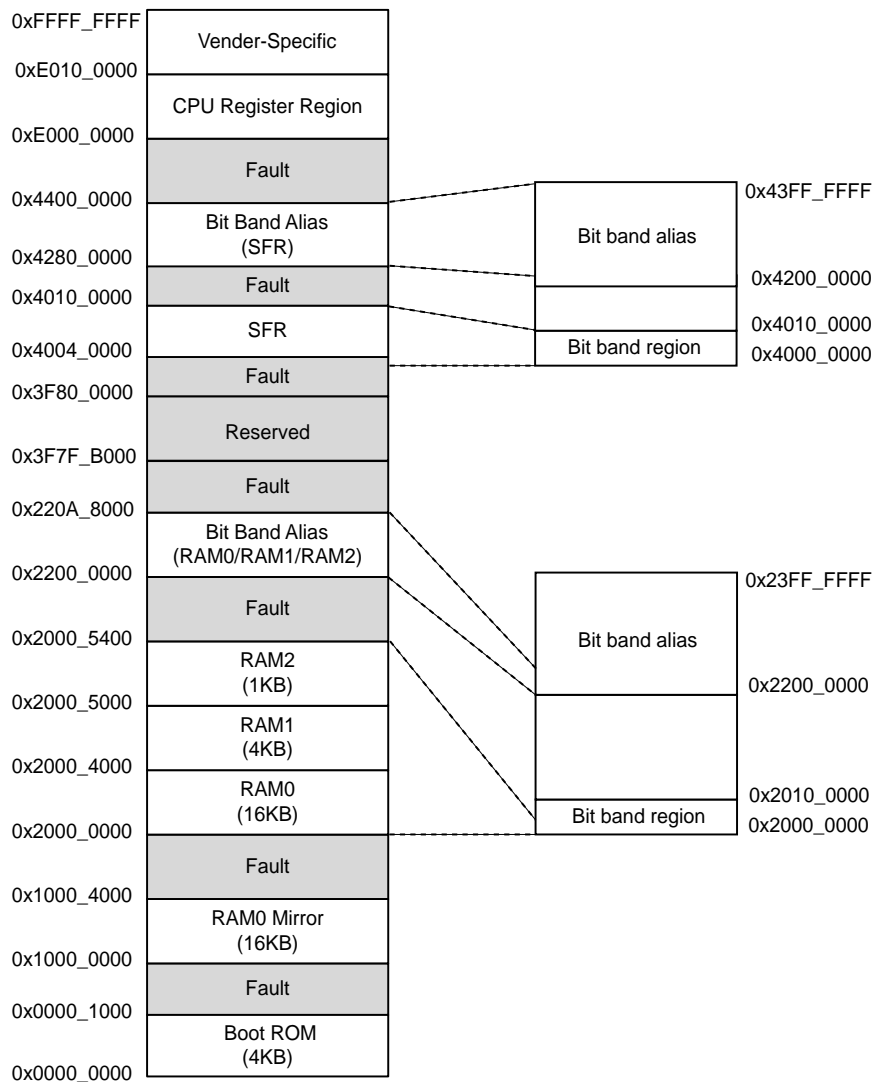


Figure 4-2 Memory map

4.3 Details of Memory Map

4.3.1 Code Area/SRAM Area

Start Address	Master		μDMAC	Core S-Bus	Core D-Bus	Core I-Bus
	Slave					
			S0	S1	S2	S3
0x0000_0000	Boot ROM	M3	Fault	Fault	o	o
0x0000_1000	Fault	-	Fault	Fault	Fault	Fault
0x1000_0000	RAM0 (mirror)	M0	Fault	Fault	o	o
0x1000_4000	Fault	-	Fault	Fault	Fault	Fault
0x2000_0000	RAM0	M0	o	o	Fault	Fault
0x2000_4000	RAM1	M1	o	o	Fault	Fault
0x2000_5000	RAM2	M2	o	o	Fault	Fault
0x2000_5400	Fault	-	Fault	Fault	Fault	Fault
0x2200_0000	Bit band alias	-	Fault	o	Fault	Fault
0x220A_8000	Fault	-	Fault	Fault	Fault	Fault
0x3F7F_F000	Reserved	-	Fault	Reserved	Fault	Fault
0x3F7F_B000	Fault	-	Fault	Fault	Fault	Fault

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

4.3.2 Peripheral Area

Start Address	Master		μDMAC	Core S-Bus	Core D-Bus	Core I-Bus
	Slave					
			S0	S1	S2	S3
0x4000_0000	Fault	-	Fault	Fault	Fault	Fault
0x4004_0000	SSP	M5	o	o	Fault	Fault
0x4004_C000	μDMAC(SFR)		o	o	Fault	Fault
0x4005_D000	TEMP	M6	o	o	Fault	Fault
0x4006_7000	DSADC		o	o	Fault	Fault
0x4008_D000	TMR16A		o	o	Fault	Fault
0x400B_8000	INTC	M7	o	o	Fault	Fault
0x400C_0000	PORT		-	o	Fault	Fault
0x400C_4000	TMRB		-	o	Fault	Fault
0x400E_1000	SIO/UART		-	o	Fault	Fault
0x400F_2000	WDT		-	o	Fault	Fault
0x400F_3000	CG	-	o	Fault	Fault	
0x4010_0000	Fault	-	Fault	Fault	Fault	Fault
0x4200_0000	Bit band alias	-	Fault	o	Fault	Fault
0x4400_0000	Fault	-	Fault	Fault	Fault	Fault

4.4 A List of Base Address for Peripheral Functions

In the peripheral area, do not access the addresses except those of the control registers. For details of the control registers, refer to each Chapter of Peripheral Functions.

Peripheral functions		Base address
Synchronous serial interface (SSP)	ch0	0x4004_0000
μ DMA controller (μ DMAC)	Unit A	0x4004_C000
Temperature sensor (TEMP)		0x4005_D000
24-bit $\Delta\Sigma$ Analg/Digital Converter (DSADC)	Unit A	0x4006_7000
	Unit B	0x4006_7400
	Unit C	0x4006_7800
	Unit D	0x4006_7C00
16-bit timer A (TMR16A)	ch0	0x4008_D000
Interrupt controller (INTC)	ch0	0x400B_8000
Input/output port	Port A	0x400C_0000
	Port B	0x400C_0100
	Port C	0x400C_0200
	Port D	0x400C_0300
16-bit timer/event counter (TMRB)	ch0	0x400C_4000
	ch1	0x400C_4100
	ch2	0x400C_4200
	ch3	0x400C_4300
Serial channel (SIO/UART)	ch0	0x400E_1000
Watchdog timer (WDT)	ch0	0x400F_2000
Clock control (CG)	ch0	0x400F_3000

5. Startup Sequence

When turning-on power, it is necessary to take a stable time of built-in regulator, built-in Flash memory and internal high-speed oscillator into consideration. TMPM311CHDUG has a function to insert a stable time automatically.

5.1 Without the use of the $\overline{\text{RESET}}$ pin (Reset by the Power-on-reset Circuit)

Once power voltage is over the release voltage of the power-on-reset, the power counter starts operation, and then an internal reset signal is released after approximately 0.5ms has elapsed.

For details of the power-on-reset circuit operation, refer to Section "Power-on-reset circuit (POR)".

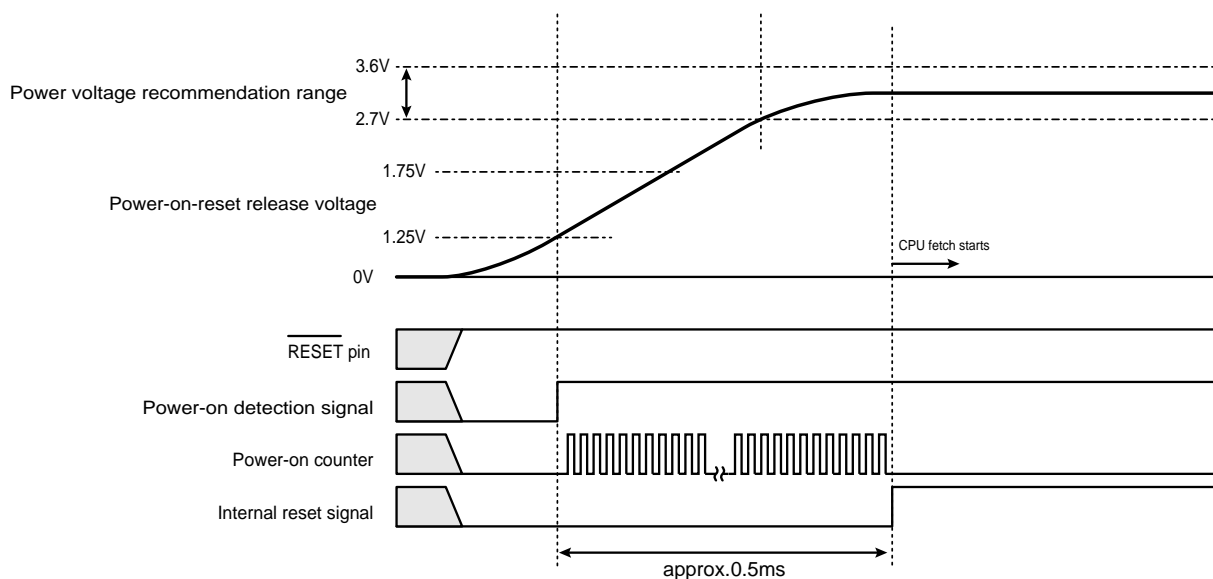


Figure 5-1 Without the use of the $\overline{\text{RESET}}$ pin

5.2 Using the $\overline{\text{RESET}}$ pin

When the $\overline{\text{RESET}}$ pin is used, if power supply voltage exceeds a detection voltage range of the power-on reset circuit, the power-on counter starts operation. After the power-on counter has started, if the $\overline{\text{RESET}}$ pin changes to "High" level, the internal reset signal is released.

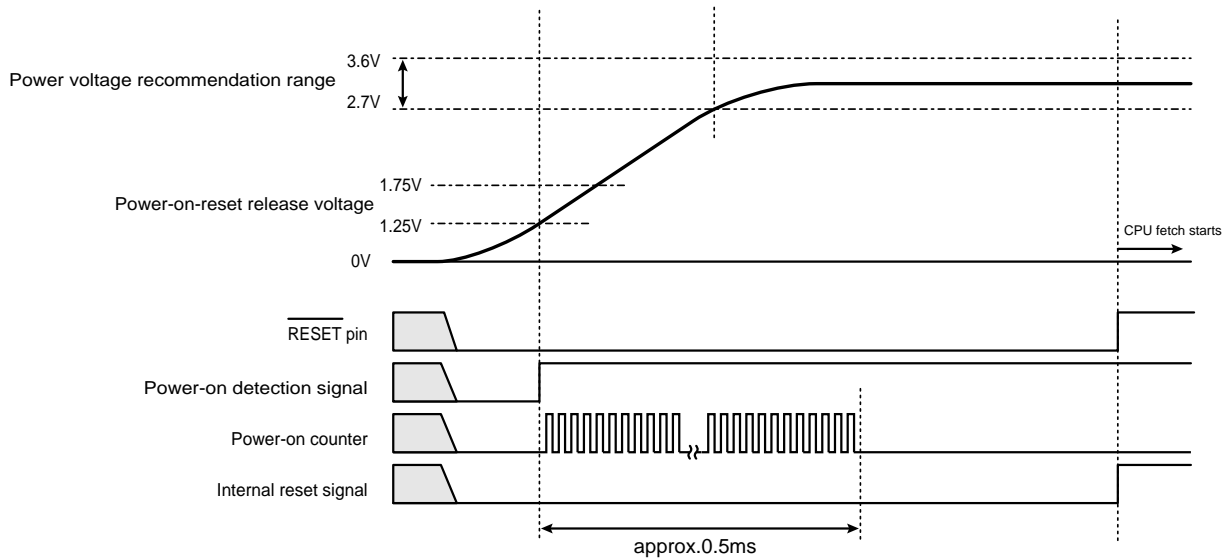


Figure 5-2 Using the $\overline{\text{RESET}}$ pin

6. Boot Program(BOOTROM)

6.1 Outline

This product starts up at the internal boot ROM after reset. The boot ROM contains a boot program that loads a user program to the internal RAM via synchronous serial interface (SSP: Synchronous Serial Port) and executes the program at the internal RAM.

The boot program has the following major functions:

1. SSP communication specifications: SPI master mode, 1.6 Mbps (internal high-speed oscillation is divided by 6), and 8-bit data length
2. RAM loader function to the internal RAM (checksum function included)

6.2 Precautions

6.2.1 Setting of the Vector Table

The boot ROM is located in 0x0000_0000 of the vector table as default. Thus, the vector table is required to be moved to user program area. The start address of the vector table can be relocated with the vector table offset register of the NVIC of the CPU.

For details, refer to "Exception/Interrupt Related Registers" in the chapter on "Exceptions."

6.2.2 Reset during the Operation

If a reset occurs during the operation, internal RAM data is not guaranteed. Therefore, data transfer is required by the boot program.

For details of the reset factor, refer to "Reset" in the chapter on "Exceptions."

6.3 System Configuration

6.3.1 Used Pins

The figure below shows the pins used by the boot program.

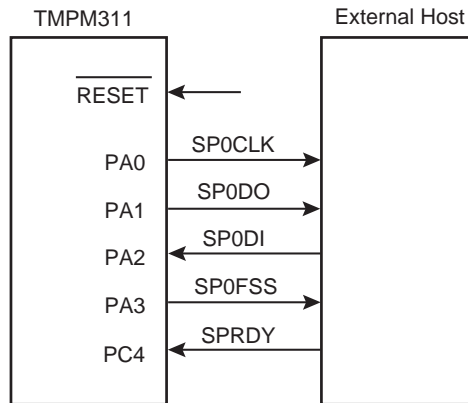


Figure 6-1 Pins used by the boot program

Table 6-1 Pins used by the boot program

Pin Name	Description
$\overline{\text{RESET}}$	This pin is used to acquire a factor released by the reset pin.
PA0	This pin outputs a clock (SPOCLK) to communicate data between the master and slaves.
PA1	This pin is used as the signal line (SP0DO) to transmit data from the master to slaves.
PA2	This pin is used as the signal line (SP0DI) to receive data from slaves to the master.
PA3	This pin is used as the signal line (SP0FSS) to specify a slave by the master.
PC4	This pin is used as the signal line (SPRDY) for handshake. When an external host is ready to receive SPOCLK from this product, set this pin to "Low" (Ready). In other pin status, set this pin to "High" (Not Ready). External hosts should be controlled to avoid missing data.

6.3.2 Memory Map

The table below shows the memories used by the boot program.

Table 6-2 Memory map used by the boot program

BOOTROM	0x0000_0000 - 0x0000_0FFF
RAM0 Mirror Program RAM area	0x1000_0000 - 0x1000_3FFF
RAM0 Data RAM area	0x2000_0000 - 0x2000_3FFF
RMA1 Data RAM area	0x2000_4000 - 0x2000_4FFF

The boot ROM uses 4 KBytes memory and is mapped from 0x0000_0000. For program execution, 16-KBytes areas are available starting from address 0x1000_0000; and as the RAM0 mirror area, 16 KBytes are available starting from 0x2000_0000.

6.4 Operational Description

6.4.1 Whole Flowchart

This section shows a boot program whole flowchart.

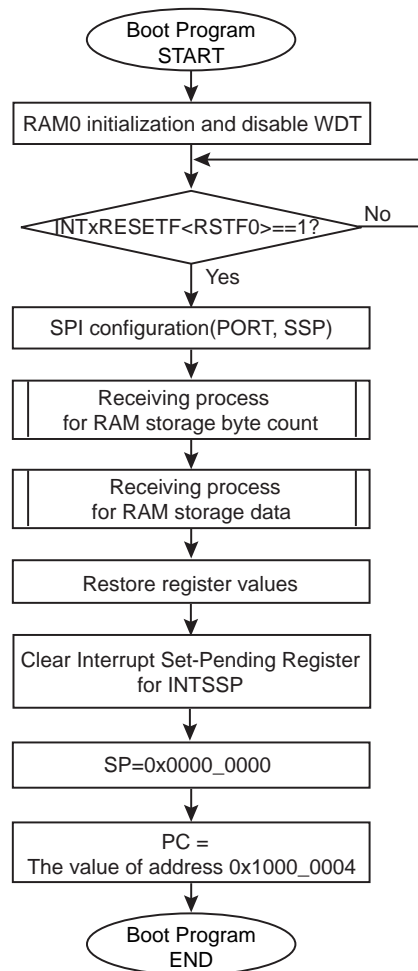


Figure 6-2 Whole Flowchart

The following explains the whole flowchart:

(1) Initial setting

Set as follows and then go to "(2) Determination of reset release using the external reset pin."

- Stop the watchdog timer (WDT).
- Initialize the destination RAM area.

(2) Determination of reset release using the external reset pin.

Poll data until INTxRESETF<RSTF0> is "1". When INTxRESETF<RSTF0> is "1", go to "(3) Initialization of the SPI communication."

Note) If an external reset is released during the power-on reset, INTxRESETF<RSTF0> is not set to "1". The external host should implement time out process, or external reset should be released after power-on reset is released.

(3) Initialization of SPI communication

Set as follows, and then execute a receive process for RAM transfer size and RAM transfer data. Go to "(4) Starting operation."

- Set a general-purpose pin to the SPI communication pin.
- Initialize SPI.

Note) The boot program sets the SSP communication rate to the internal high-speed oscillation (10 MHz) divided by 6. The SSP communication rate of the external host must be set faster (approximately 1.6 MHz) than the above-mentioned.

(4) Starting operation

- The boot program converts the changed value of the register into the previous value.
- Clear the bit of the SSP interrupt pending-set (clear) register.
- Set the value of 0x1000_0004 (reset address) to the PC. Note that 0x0000_0000 is set to the SP before the PC setting. Therefore, initialize the SP before the user program is executed.

Note) The boot program converts the changed value of the WDT register into the previous value. Poll data until WDxFLG<FLG> is "0" (register write-enable). Thus, if "1" (register write-disable) is continued, the PC is not set. The external host is required to execute timeout process.

6.4.2 Reception Flowchart for RAM Transfer Size Setting

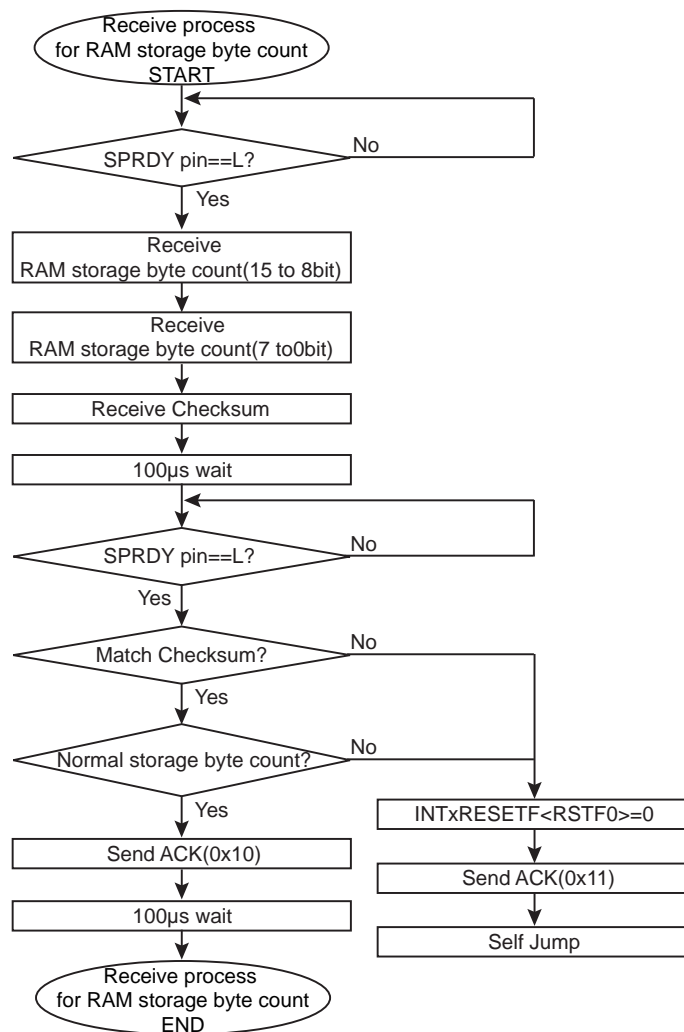


Figure 6-3 Reception Flowchart for RAM Transfer Size Setting

(1) Checking the SPRDY pin

- The boot program polls data until the SPRDY pin is "Low" (Ready). When the SPRDY pin is "Low", go to "(2) Determination of the RAM transfer size."
 - An external host should prepare to transmit the RAM transfer size and the checksum for RAM transfer size. Set the SPRDY pin to "Low".

(2) Determination of the RAM transfer size

- The boot program receives the RAM transfer size and checksum for the RAM transfer size from the external host. Then the boot program waits for 100 μ s, and polls data until the SPRDY pin is "Low".
 - The external host prepares for the next response. Set the SPRDY pin to "Low".
- The boot program compares the received checksum with the checksum for the RAM transfer size.
 - If checksums are matched and the RAM transfer size is within 16 KBytes (0x4000), a normal response (0x10) is sent. Then, the boot program waits for 100 μ s and completes the RAM transfer size receive process.
 - If an abnormal condition occurs, clear INTxRESETF<RSTF0> to "0" and send an abnormal response (0x11). Then, the boot program executes self-jump.
If an ACK response is abnormal (0x11), the external host should take countermeasures including an external reset against this product.

Note) In reception, transmit dummy data is 0xFF.

6.4.3 Reception Flowchart for RAM Transfer Data

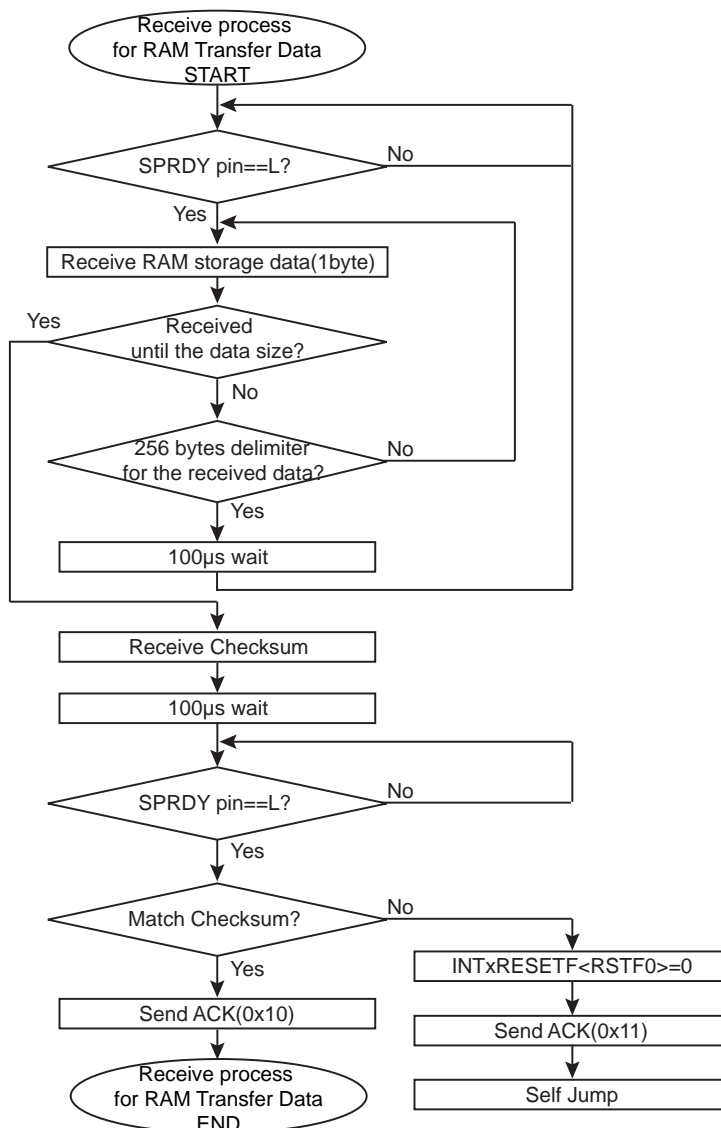


Figure 6-4 Reception Flowchart for RAM Transfer Data

- (1) Checking the SPRDY pin
- The boot program polls data until the SPRDY pin is "Low" (Ready). When the SPRDY pin is "Low", go to "(2) Determination of the RAM transfer data."
 - An external host should prepare to transmit RAM transfer data and checksum for RAM transfer data. Set the SPRDY pin to "Low".

(2) Determination of RAM transfer data

- The boot program receives RAM transfer data from an external host. Received data is written to the start address (0x2000_0000) of RAM0.
 - If the total amount of received data does not reach the RAM transfer size, the next data is received. Note that the boot program put a wait of 100 μ s each 256 bytes. It polls data until the SPRDY pin is "Low". If the external host stops transmitting RAM loaded data, set the SPRDY pin to "High" and execute recovery process including a reset.
 - If the total amount of received data is the same as RAM transfer size, a checksum for RAM transfer data is received. Then the boot program waits for 100 μ s, and polls data until the SPRDY pin is "Low". The external host prepares the next response. Set the SPRDY pin to "Low".
- The boot program compares the received checksum with the checksum for the total amount of data written in RAM0 area.
 - If the checksums are matched, a normal response (0x10) is send. Then RAM transfer process is complete.
 - If the checksums are not matched, clear INTxRESETF<RSTF0> to "0" and send an abnormal response (0x11).Then, the boot program executes self-jump.
 - If an ACK response is abnormal (0x11), the external host should take countermeasures including an external reset against this product.

Note) In reception, transmit dummy data is 0xFF.

6.4.4 Data Transfer Format

Table 6-3 Data Transfer Format

Byte	Data transfered from the external host to TMPM311	Data transfered from TMPM311 to the external host
1 byte	RAM storage byte count 15 to 8 bit	-
2 byte	RAM storage byte count 7 to 0 bit	-
3 byte	Check sum value for 1 - 2 byte	-
4 byte	-	ACK for the RAM storage byte count Normal acknowledge: 0x10 Negative acknowledge: 0x11
5 byte to m byte	RAM storage data	-
m+1 byte	Check sum value for 4 to m byte	-
m+2 byte	-	ACK for the RAM storage data Normal acknowledge: 0x10 Negative acknowledge: 0x11

6.4.5 Cecksum Calculation

The checksum byte for a series of bytes of data is calculated by adding the bytes together, dropping the carries, and taking the two's complement of the total sum. The external host must perform the same checksum operation in transmitting checksum bytes.

Example) Calculation example for the Checksum

To calculate the checksum for a series of 0xE5 and 0xF6:

Add the bytes together

$$0xE5 + 0xF6 = 0x1DB$$

Take the two's complement of the sum, and that is the checksum byte.

$$0 - 0xDB = 0x25$$

7. Clock Control

7.1 Outline

The clock control circuit controls the internal/external oscillator, warm-up, clock gear, and prescaler.

7.2 Schematic Diagram of Clocks

This section shows the schematic diagram of clocks.

After reset, the clock indicated by the arrow is selected among input clocks supplied to the selector.

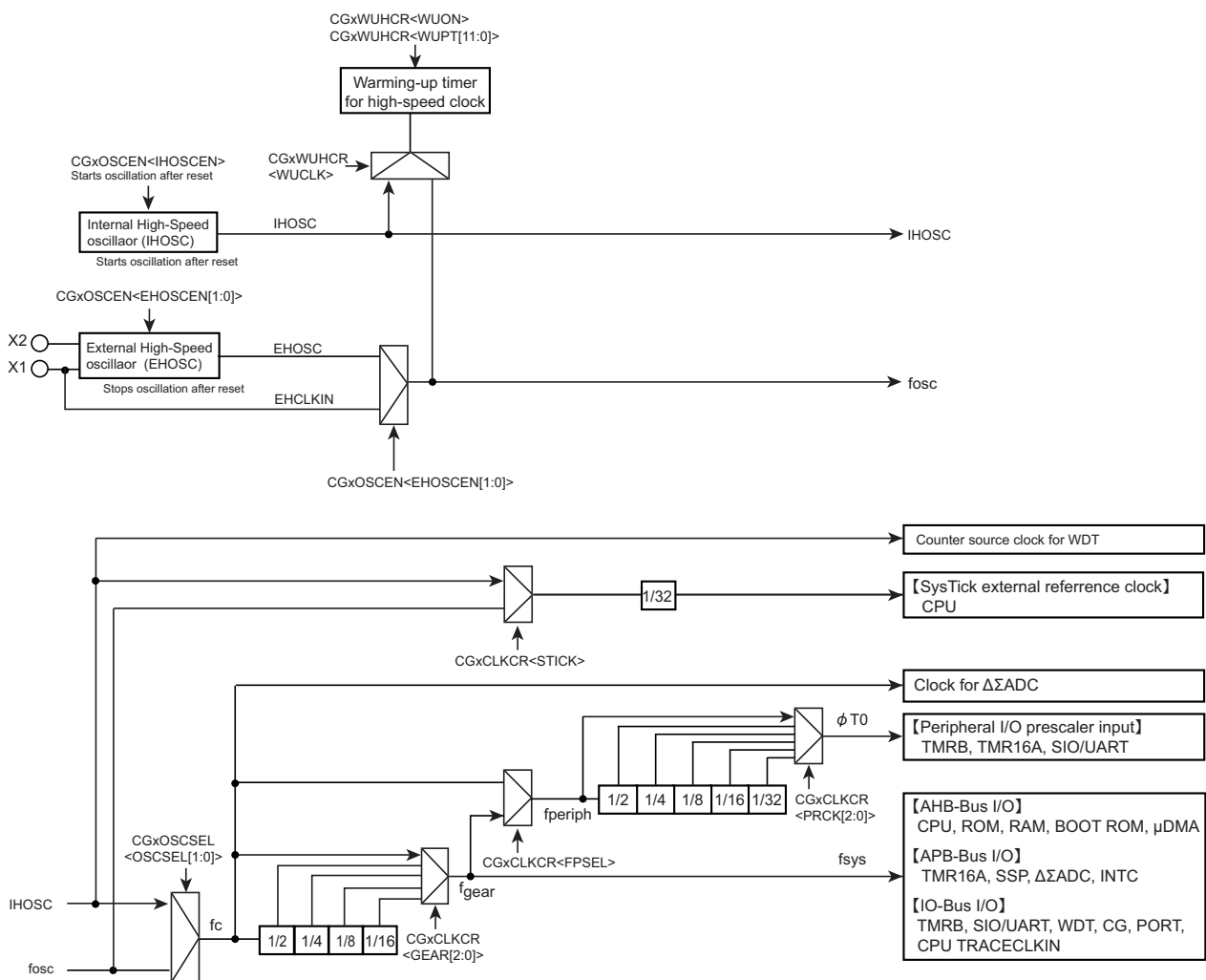


Figure 7-1 Schematic diagram of the clocks

7.3 Registers

7.3.1 Register List

The tables below show the registers and their addresses related to the clock control.

For the base addresses, refer to the chapter on "A List of the Peripheral Function Base addresses" of "Memory Map."

Peripheral function :CG(ch0)

Register name		Address (Base+)
Protect register	CGxPROTECT	0x0000
High-speed oscillation select register	CGxOSCSEL	0x0004
High-speed oscillation status register	CGxOSCSTF	0x0008
Clock control register	CGxCLKCR	0x000C
Oscillation enable register	CGxOSCEN	0x0018
High-speed oscillation warm-up register	CGxWUHCR	0x0024

7.3.2 Details of the Registers

This section explains the registers in the clock control circuit.

7.3.2.1 CGxPROTECT (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 [7:0]	R/W	Register write control 0xC1: Enabled Other than 0xC1: Disabled After reset, this bit is "0xC1" and is write enable. If this bit is set to other than "0xC1", CGxOSCSEL, CGxCLKCR, CGxOSCEN and CGxWUHCRCR cannot be written to.

7.3.2.2 CGxOSCSEL (High-speed Oscillation Select Register)

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

Bit	Bit Symbol	Type	Function
31-2	-	R	Read as "0".
1-0	OSCSEL[1:0]	R/W	<p>Source clock selection for high-speed clock (fc)</p> <p>00: Internal high-speed oscillator (IHOSC)</p> <p>01: External high-speed oscillator (EHOSC) /external clock input (EHCLKIN)</p> <p>10: Reserved</p> <p>11: Reserved</p> <p>This bit is used to select the source clock for the high-speed clock (fc). When the contents of the register is updated, confirm whether destination clock oscillates stably.</p> <p>Also, if the source clock is changed, check whether the changed value is reflected to CGxOSCSTF<OSCF [1:0]>.</p>

7.3.2.3 CGxOSCSTF (High-speed Oscillation 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	-	-	-	-	-	-	OSCF	
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	-	R	Read as "0".
1-0	OSCF[1:0]	R	Source clock for high-speed clock (fc) selection status 00: Internal high-speed oscillator (IHOSC) 01: External high-speed oscillator (EHOSC) /external clock input (EHCLKIN) 10: Reserved 11: Reserved

7.3.2.4 CGxCLKCR (Clock Control Register)

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

Bit	Bit Symbol	Type	Function
31-8	-	R	Read as "0".
7	STICK	R/W	Source clock selection for the SysTick reference clock 0: IHOSC 1: fosc Selects a source clock for the SysTick reference clock. Before changing the reference clock, disable the SysTick timer operation.
6	FPSEL	R/W	Source clock selection for the fperiph 0: fgear 1: fc Selects a source clock for fperiph. When fc is selected as the source clock, a frequency of fperiph can be fixed regardless of the clock gear setting.
5-3	PRCK[2:0]	R/W	Division ratio selection for the prescaler clock ($\phi T0$) 000: fperiph 100: fperiph/16 001: fperiph/2 101: fperiph/32 010: fperiph/4 110: Reserved 011: fperiph/8 111: Reserved Selects a division ratio for the prescaler clock ($\phi T0$) supplied to the peripheral functions. Do not change a division ratio for the prescaler clock while the peripheral functions are operating. Note that when a division ratio of the prescaler clock is changed, a frequency of $\phi T0$ must be lower than the frequency of the fsys.
2-0	GEAR[2:0]	R/W	Division ratio selection of the clock gear (fgear) 000: fc 100: fc/16 001: fc/2 101: Reserved 010: fc/4 110: Reserved 011: fc/8 111: Reserved Selects a division ratio of the clock gear. Do not change a division ratio of the clock gear while the peripheral functions are operating. Note that when the clock gear function is used, a frequency of $\phi T0$ must be lower than the frequency of the fsys.

7.3.2.5 CGxOSCEN (Oscillation 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	-	-	-	-	-	EHOSCEN		IHOSCEN
After reset	0	0	0	0	0	0	0	1

Bit	Bit Symbol	Type	Function
31-3	-	R	Read as "0".
2-1	EHOSCEN[1:0]	R/W	<p>Selection for the external high-speed oscillator operation and source clock for fosc.</p> <p>00: Disuse of the external high-speed oscillator</p> <p>01: Selects EHOSC as the source clock for the fosc and enables the high-speed oscillator.</p> <p>10: Selects EHCLKIN as the source clock for the fosc.</p> <p>11: Reserved</p> <p>Selects the external high-speed oscillator operation and source clock of the fosc. When "01" is set, an oscillation stable time is required using the warm-up counter. For details, refer to "7.5 Warm-up Timer Function".</p> <p>When the external high-speed oscillator is selected, all of the port D control registers (PDCR/PDPUP/PDIE) are set to disable.</p>
0	IHOSCEN	R/W	<p>Internal high-speed oscillator (IHOSC) control</p> <p>0: Disabled</p> <p>1: Enabled</p> <p>Controls the internal high-speed oscillator (IHOSC).</p> <p>When this bit is enabled, an oscillation stable time is required using the warm-up counter. For details, refer to "7.5 Warm-up Timer Function".</p>

7.3.2.6 CGxWUHCR (High-Speed Oscillator Warm-up Control Register)

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

Bit	Bit Symbol	Type	Function
31-28	-	R	Read as "0".
27-16	WUPT[11:0]	R/W	Counter compare value setting of the warm-up timer for the high-speed oscillator Sets a compare value for the counter of the warm-up timer. Sets the upper 12 bits values of the 16 bits calculation values of the warm-up time.
15-9	-	R	Read as "0".
8	WUCLK	R/W	Selects a source clock of the warm-up timer for the high-speed oscillator. 0: IHOSC 1: fosc
7-2	-	R	Read as "0".
1	WUEF	R	The warm-up timer operation status for the high-speed oscillator 0: The operation is complete. 1: The operation is in process. Checks the operation status of the warm-up timer.
0	WUON	W	The warm-up timer control for the high-speed oscillator 1: The operation starts. Check whether <WUEF> is "0" before the warm-up timer operation starts (set "1"). Writing "0" to <WUON> has no meaning and "0" is read.

7.4 Enabling/Disabling the Oscillators

7.4.1 Internal High-speed Oscillator

To enable/disable the internal high-speed oscillator, use CGxOSCEN<IHOSCEN>.

After reset, the oscillator is enabled (CGxOSCEN<IHOSCEN> = "1").

7.4.2 External High-speed Oscillator

To enable/disable the external high-speed oscillator, use CGxOSCEN<EHOSCEN[1:0]>.

After reset, the oscillator is disabled (CGxOSCEN<EHOSCEN[1:0]> = "00").

7.5 Warm-up Timer Function

The warm-up timer function secures a stable time at start of the external oscillator.

Note: The warm-up timer operates on an unstable clock. Therefore, when the counter of the warm-up timer is set, set a necessary warm-up time with enough margin.

7.5.1 Source Clock Selection of the Warm-up Timer

IHOSC or EHOSC is selected as the source clock of the warm-up timer for the high-speed oscillator with CGxWUHCR<WUCLK>.

7.5.2 Starting the Warm-up Timer

To start the warm-up timer, set "1" to CGxWUHCR<WUON>.

7.5.3 Calculating a Comparison Value for the Counter of the Warm-up timer

To secure a stable time for warm-up, set CGxWUHCR<WUPT[11:0]>.

The upper 12 bits of the 16-bit counter of the warm-up timer for the high-speed oscillator are compared with CGxWUHCR<WUPT[11:0]>.

Therefore, the lower 4 bits of the calculated value of the following equation are rounded down. The upper 12 bits are set to CGxWUHCR<WUPT[11:0]>.

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

For example, where an 8MHz external oscillator is used and warm-up time is 5ms, the equation is:

$$\frac{\text{Warm-up time}}{\text{Warm-up clock cycle}} = \frac{5\text{ms}}{1/8\text{MHz}} = 40000 \text{ cycles} = 0x9C40$$



Round down the lower 4 bits. Set 0x9C4 to CGxWUHCR<WUPT[11:0]>.

7.5.4 Confirmation of the Completion of the Warm-up Timer Operation

To check the completion of the warm-up timer operation, read CGxWUHCR<WUEF> and CGxWULCR<WULEF>.

7.5.5 Example of the Setting of the Warm-up Timer

Table 7-1 Example of the setting of the warm-up timer (for the external high-speed oscillator)

	Read CGxWUHCR<WUEF>	: Confirms whether the warm-up counter operation is complete. Repeat the operation until "0" is read.
	CGxWUHCR<WUPT[11:0]> = "0x9C4"	: Sets a warm-up time.
	CGxOSCEN<EHOSCEN[1:0]> = "01"	: Enables the external high-speed oscillation. Selects EHOSC as the source clock for fosc.
	CGxWUHCR<WUON> = "1"	: Starts the warm-up timer.
	Read CGxWUHCR<WUEF>	: Confirms whether the warm-up counter operation is complete. Repeat the operation until "0" is read.

7.6 System Clock

The system clock (f_{sys}) is used to operate the CPU core and the peripheral functions.

A gear clock (fgear) which divides the high-speed clock by the clock gear can be selected as the source clock for the f_{sys}.

A division ratio is selected with CGCLKCR<GEAR[2:0]>.

A value in CGxCLKCR<GEAR[2:0]> can be changed even if the clock gear is operating. However, it needs a certain time period until the fgear is changed after the value in CGxCLKCR<GEAR[2:0]> is changed.

As a source clock for f_c, a clock input from the internal oscillator (IHOSC), clock input from the external oscillator (EHOSC) or external clock input (EHCLKIN) can be selected.

The table below shows the number of division ratios of the clock gear and setting ranges of the high-speed clock.

Table 7-2 The number of division ratios of the clock gear and setting ranges of the high-speed clock

External oscillator (EHOSC)/ External clock input (EHCLKIN) frequency (MHz)	High-speed clock(f _c) frequency (MHz)	fgear frequency (MHz)				
		Division ratio of the clock gear				
		1/1	1/2	1/4	1/8	1/16
8	8	8	4	2	1	-
10	10	10	5	2.5	1.25	-
12	12	12	6	3	1.5	-
16	16	16	8	4	2	1
18	18	18	9	4.5	2.25	1.125
20	20	20	10	5	2.5	1.25
24	24	24	12	6	3	1.5

↑ Initial state after reset.

Note: Do not use the setting of 1/16 as the division ratio of the clock gear when SysTick is used.

7.7 Prescaler Clock

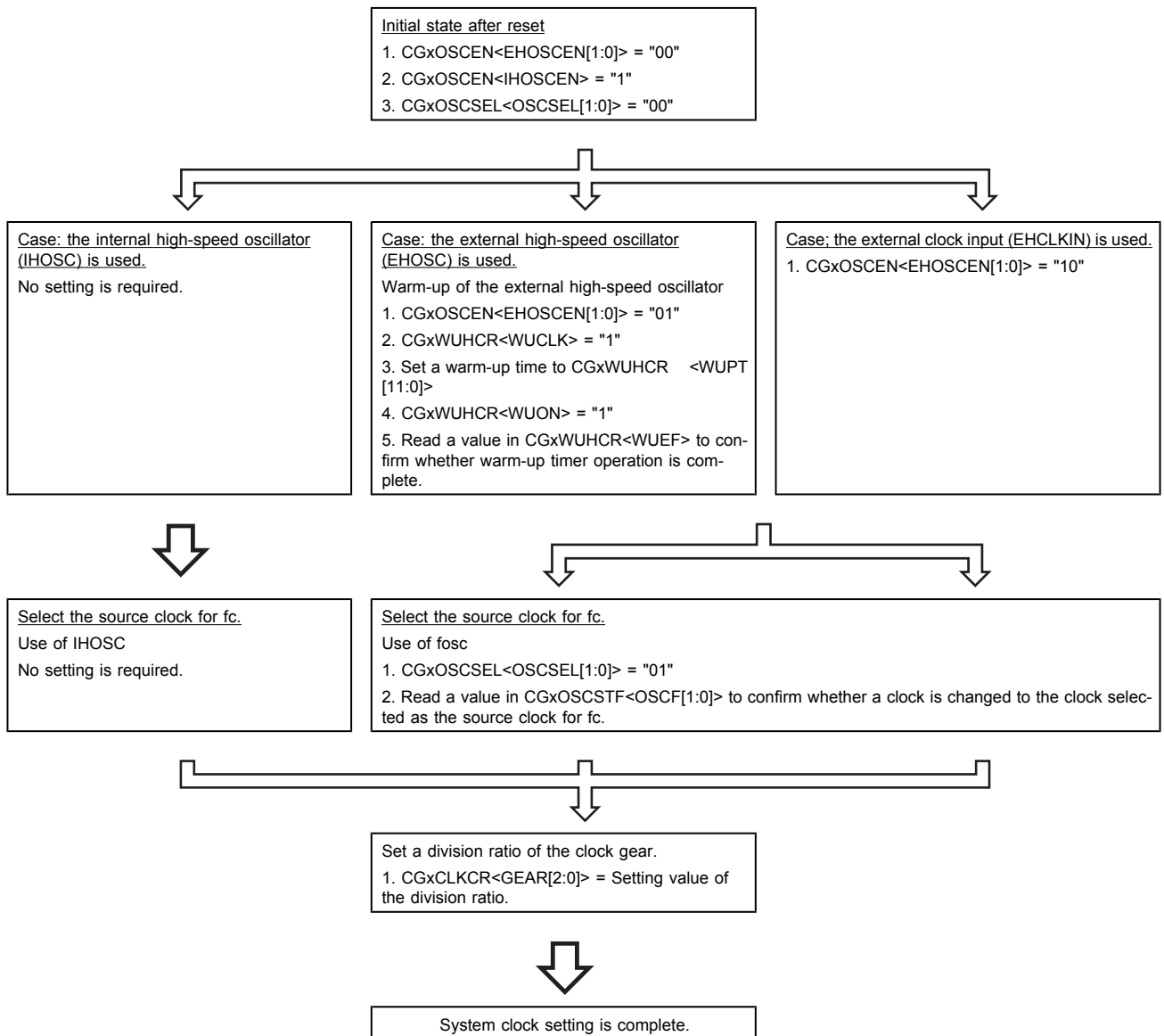
A prescaler clock (φT0) is input to the prescaler contained in a peripheral function.

The source clock for φT0 is a clock that divides an f_{periph}.

The source clock for f_{periph} can be selected from f_c or fgear with CGxCLKCR<FPSEL>.

A division ratio is selected with CGxCLKCR<PRCK[2:0]>.

7.8 System Clock Setting after Reset



8. Exceptions

This chapter describes features, types and handling of exceptions.

Exceptions have close relation to the CPU core. Refer to the documentation set issued by ARM Limited if needed.

8.1 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.

8.1.1 Exception Types

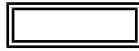
The following types of exceptions exist in the Cortex-M3.

For detailed descriptions on each exception, refer to the documentation set issued by ARM Limited.

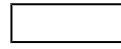
- Reset
- Non-Maskable Interrupt (NMI)
- Hard Fault
- Memory Management
- Bus Fault
- Usage Fault
- SVCall (Supervisor Call)
- Debug Monitor
- PendSV
- SysTick
- External Interrupt

8.1.2 Handling Flowchart

The following shows how an exception/interrupt is handled. In the following descriptions,



indicates hardware handling.



Indicates software handling.

Each step is described later in this chapter.

Processing	Description	See	
<div style="border: 3px double black; padding: 5px; width: fit-content; margin: 0 auto;">Detection by CG/CPU</div>	The CG/CPU detects the exception request.	Section 8.1.2.1	
<div style="border: 3px double black; padding: 5px; width: fit-content; margin: 0 auto;">Handling by CPU</div>	The CPU handles the exception request.	Section 8.1.2.2	
<div style="border: 3px double black; padding: 5px; width: fit-content; margin: 0 auto;">Branch to ISR</div>	The CPU branches to the corresponding interrupt service routine (ISR).		
<div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 0 auto;">Execution of ISR</div>	Necessary processing is executed.	Section 8.1.2.3	
<div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 0 auto;">Return from exception</div>	The CPU branches to another ISR or returns to the previous program.	Section 8.1.2.4	

Note: ISR : Interrupt Service Routine

8.1.2.1 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 "8.5 Interrupts".

(2) Exception detection

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

Table 8-1 shows the priority of exceptions. "Configurable" means that you can assign a priority level to that exception. Memory Management, Bus Fault and Usage Fault exceptions can be enabled or disabled. If a disabled exception occurs, it is handled as Hard Fault.

Table 8-1 Exception Types and Priority

No.	Exception type	Priority	Description
1	Reset	-3 (highest)	Reset pin, WDT, POR or SYSRETRQ
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
4	Memory Management	Configurable	Exception from the Memory Protection Unit (MPU) (Note 1) Instruction fetch from the Execute Never (XN) region
5	Bus Fault	Configurable	Access violation to the Hard Fault region of the memory map
6	Usage Fault	Configurable	Undefined instruction execution or other faults related to instruction execution
7 to 10	Reserved	-	
11	SVCcall	Configurable	System service call with SVC instruction
12	Debug Monitor	Configurable	Debug monitor when the CPU is not faulting
13	Reserved	-	
14	PendSV	Configurable	Pendable system service request
15	SysTick	Configurable	Notification from system timer
up to 16	External Interrupt	Configurable	External interrupt pin or peripheral function (Note 2)

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

Note 2: **External interrupts have different sources and numbers in each product. For details, see "8.5.2 A List of Interrupt Factors".**

(3) Priority setting

- Priority levels

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

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

TMPM311CHDUG has a 3-bit configuration.

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

- Priority grouping

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

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

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

Table 8-2 Priority grouping setting

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

Note: If the configuration of <PRI_n> is less than 8 bits, the lower bit is "0".

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

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

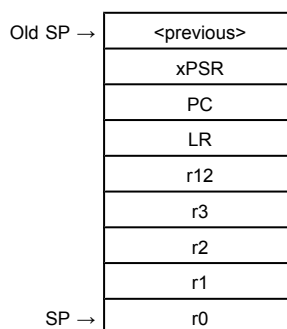
When an exception occurs, the CPU suspends the currently executing process and branches to the interrupt service routine (ISR). This is called "pre-emption".

(1) Stacking

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

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

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



(2) Fetching an ISR

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

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

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

(3) Late-arriving

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

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

(4) Vector table

The vector table is configured as shown below.

You must always set the first four words (stack top address, reset ISR address, NMI ISR address, and Hard Fault ISR address).

Set ISR addresses for other exceptions if necessary.

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

8.1.2.3 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 "8.5 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.

8.1.2.4 Exception exit

(1) Execution after returning from an ISR

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

- Tail-chaining

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

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

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

(2) Exception exit sequence

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

- Pop eight registers
 - Pops the eight registers (xPSR, PC, KR, r0 to r3 and r12) from the stack and adjust the SP.
- Load current active interrupt number
 - Loads the current active interrupt number from the stacked xPSR. The CPU uses this to track which interrupt to return to.
- Select SP
 - If returning to an exception (Handler Mode), SP is SP_main. If returning to Thread Mode, SP can be SP_main or SP_process.

8.2 Reset

A reset initializes the control registers of the core and control registers (SFR) of the peripheral functions.

There are several factors to generate a reset, and the target to be initialized varies on factors.

8.2.1 Factors

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

When the $\overline{\text{RESET}}$ pin is set to "High" from "Low", reset exception occurs.

To reset TMPM311CHDUG while the CPU is in process, set the $\overline{\text{RESET}}$ pin to be "Low" at least for 12 clocks of the internal oscillation.

- Power-On Reset (POR)

The POR has the function to generate reset exception. For details, refer to the chapter on "Power-on Reset Circuit."

- Watchdog Timer (WDT)

The WDT has the function to generate reset exception. For details, refer to the chapter on "Watchdog Timer."

- SYSRESETREQ

To generate reset exception, set <SYSRESETREQ> of the Application Interrupt and Reset Control Register of the NVIC.

8.2.2 Reset Factors and Their Valid Ranges

The table below shows the reset factors and their valid ranges:

Table 8-3 Reset factors and their valid ranges

Block	Power-on-reset	$\overline{\text{RESET}}$ pin	WDT reset	SYSRESETREQ
System debug components (FPB, DWT, ITM)	o	-	-	-
Others	o	o	o	o

o : Registers to be Initialized. The contents of the RAM are not maintained.

- : Registers not to be initialized. The contents of the RAM are maintained.

8.2.3 Checking Reset Factors

To check reset factors, read INTxRESETF of the interrupt controller.

8.3 Non-maskable Interrupt (NMI)

The WDT can generate non-maskable interrupts. For details, refer to the chapter on "Watchdog Timer."

8.4 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 CGxOSCCR <OSCSEL> <HOSCON> by 32 is used as external reference clock.

8.5 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.

8.5.1 Interrupt Request

8.5.1.1 Interrupt Path

Figure 8-1 shows the interrupt paths.

[1] indicates interrupts which are directly input to the CPU from the peripheral circuit. These interrupts are used only at the NVIC setting.

[3] indicates interrupts which are directly input from the external interrupts input from the interrupt pins (INTx) via ports. The interrupt detection circuit detects interrupts corresponding to active levels specified in the INTxCRn register.

The detected interrupt [3] is input to the CPU. If the interrupt enable set register of the NVIC enables interrupts, an interrupt occurs.

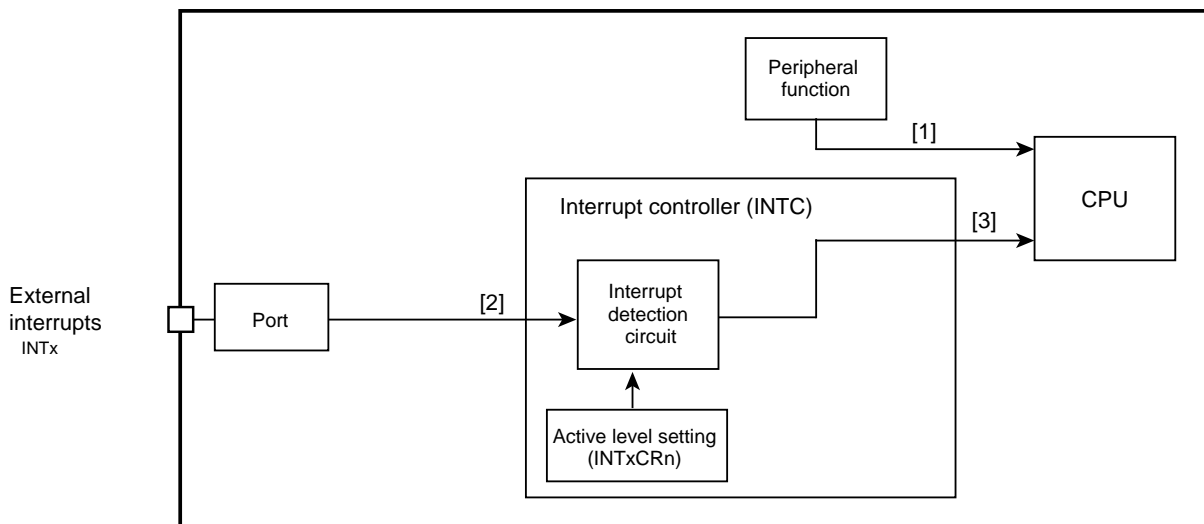


Figure 8-1 Interrupt request paths

8.5.1.2 Generating Interrupt Requests

Interrupt requests are generated by the settings of the external interrupt pins allocated to interrupt factors, peripheral functions, or Interrupt Set-Pending Register.

- Interrupts by the external interrupt pins
- Interrupts by the peripheral functions
 - When the interrupts by the peripheral functions are used, interrupt generation must be enabled in the peripheral function to be used.
 - For details of the setting, refer to each chapter of the peripheral functions.
- Forcible generation of the interrupts
 - To forcibly generate interrupt requests, set the corresponding bit of the interrupt hold setting register in the NVIC.

The CPU recognizes "High" level of the interrupt request signal as interrupts.

8.5.2 A List of Interrupt Factors

Table 8-4 lists the interrupt names in each peripheral function.

For the interrupts via the INTC, their active levels are set with INTxCRn. Settable active levels are marked with "o." Note that, the interrupts used to release STOP1 mode are required to set INTxEN as well.

The INTC interrupt numbers correspond to "n" of INTxCRn and n-th digit of binary numbers of INTxEN.

Table 8-4 Peripheral functions and interrupt names

Peripheral function	Interrupt name	Function
External interrupt	INTx	External interrupt input pin x
μDMA controller (μDMAC)	INTDMAxTCn	Channel n of Unit x is transferred completely.
	INTDMAxERR	Transfer error of Unit x interrupt
16-bit timer / event counter (TMRB)	INTTBx	Timer register 0 compare match interrupt Timer register 1 compare match interrupt Overflow interrupt
	INTCAPx0	Input capture 0 interrupt
	INTCAPx1	Input capture 1 interrupt
16-bit timer A (TMR16A)	INTT16Ax	Match interrupt
Serial Chanel (SIO/UART)	INTRXx	Receive interrupt
	INTTXx	Transmit interrupt
Synchronous serial communication circuit (SSP)	INTSSPx	Transmit interrupt Receive interrupt Receive overrun interrupt Time-out interrupt
24-bit ΔΣ analog to digital converter (DSADC)	INTDSADx	Unit x conversion completion interrupt

Table 8-5 A list of interrupt factors

No.	Factor	INTC interrupt no.	Active level				
			"Low" level	"High" level	Rising edge	Falling edge	Rising and falling edge
0	INTDSADA						
1	INTDSADB						
2	INTDSADC						
3	INTDSADD						
4	INT0	0	o	o	o	o	o
5	INT1	1	o	o	o	o	o
6	INTSSP0						
7	INTRX0						
8	INTTX0						
9	INTTB0						
10	INTTB1						
11	INTTB2						
12	INTTB3						
13	INTCAP00						


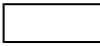
Table 8-5 A list of interrupt factors

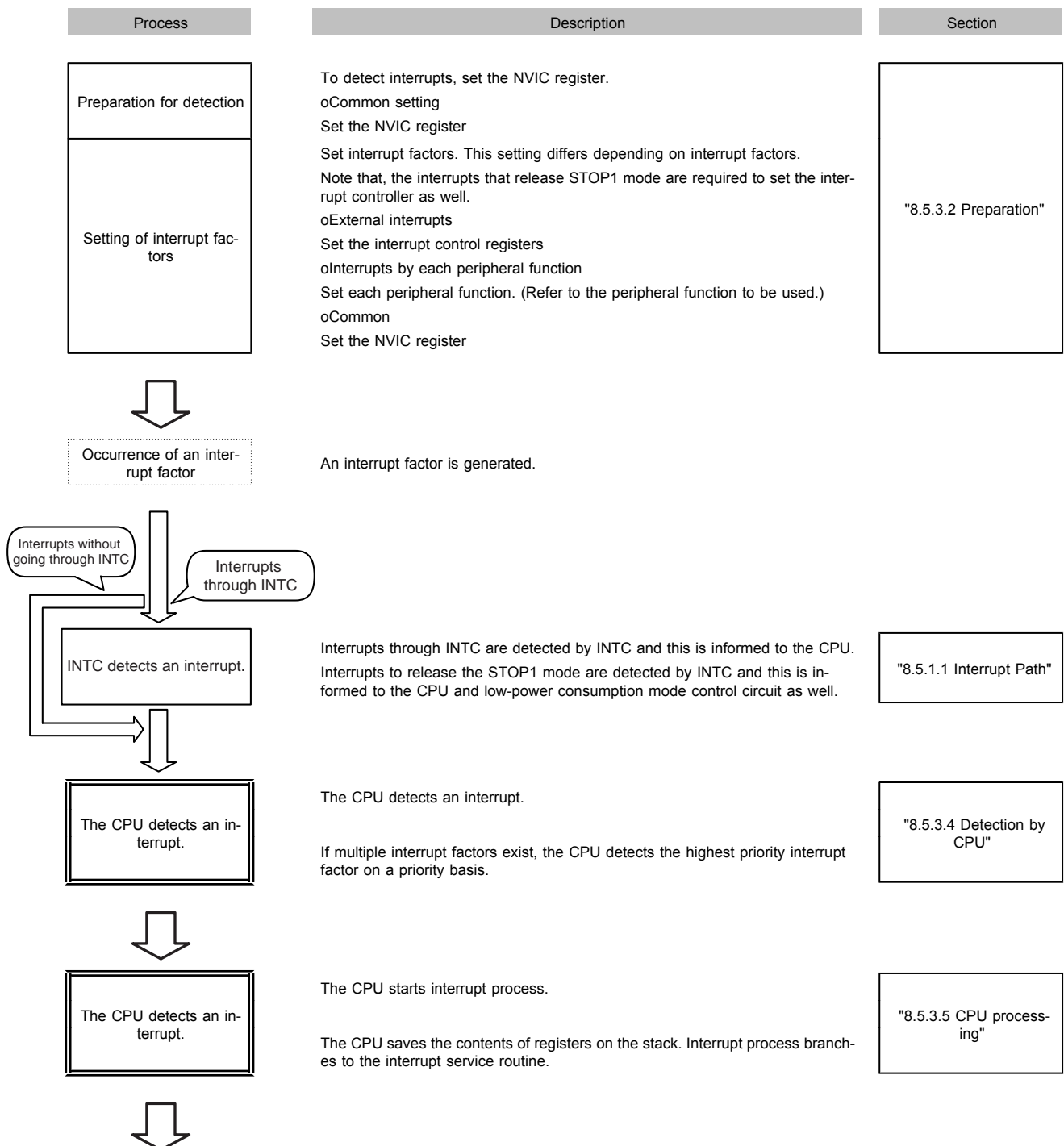
No.	Factor	INTC interrupt no.	Active level				
			"Low" level	"High" level	Rising edge	Falling edge	Rising and falling edge
14	INTCAP01						
15	INTCAP10						
16	INTCAP11						
17	INTT16A0						
18	INTDMAATC0						
19	INTDMAATC1						
20	INTDMAATC2						
21	INTDMAATC3						
22	INTDMAAERR						


8.5.3 Details of the Process

8.5.3.1 Process Flow

This section explains the flow of interrupt process.

In the following explanations,  indicates hardware process;  indicates software process.



Process	Description	Section
The interrupt service routine is executed.	Program the necessary process. Withdraw interrupt requests if required.	"8.5.3.6 Interrupt Service Routine (ISR)"
<div style="text-align: center;">  </div> Return to the original program.	The CPU returns to the normal process program from the interrupt service routine.	

8.5.3.2 Preparation

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

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

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

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

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

(1) Disabling interrupt by CPU

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

Use "MSR" instruction to set this register.

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

Note 1: PRIMASK register cannot be modified by the user access level.

Note 2: If a fault causes when "1" is set to the PRIMASK register, it is treated as a hard fault.

(2) CPU registers setting

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

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

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

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

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

This product uses three bits for assigning a priority level.

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

Set "1" to PxIE of the corresponding pin of external interrupts to enable input.

Port register		
PxIE<PxmlE>	←	"1"

Note:m: corresponding bit

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

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

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

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

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

Note:m: corresponding bit

(6) Setting of the interrupt controller

Active levels of the interrupts through the interrupt controller are set with INTxCRn. INTxCRn is a register that is required to be set according to factors. For each selectable active levels, refer to "Table 8-5 A list of interrupt factors".

To avoid the occurrence of unnecessary remaining interrupts, write the corresponding value to a factor to clear interrupt requests using INTxCLR before interrupts are enabled.

Interrupt control register		
INTxCRn<ACT>	←	Active level
INTxCLR	←	Values corresponding to the target factor.

Note: "n" indicates a register number; "m" indicates a INTC interrupt number.

(7) Enabling interrupt by CPU

Enable the interrupt by the CPU as shown below.

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

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

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

At the end, PRIMASK register is zero cleared.

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

Note 1: m : corresponding bit

Note 2: PRIMASK register cannot be modified by the user access level.

8.5.3.3 Detection by Interrupt Controller

An interrupt request through INTC is detected according to the active level specified in the interrupt controller, and is notified to the CPU.

An edge-triggered interrupt request, once detected, is held in the INTC. 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 INTC detects an interrupt request, it keeps sending the interrupt signal in "High" level to the CPU until the interrupt request is cleared in the INTxCLR register. If the interrupt is not cleared, the same interrupt will be detected again when normal operation is resumed. Be sure to clear each interrupt request in the ISR.

8.5.3.4 Detection by CPU

The CPU detects an interrupt request with the highest priority.

8.5.3.5 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.

8.5.3.6 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-M3 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

Each interrupt request through INTC must be cleared with the INTxCLR register.

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

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

8.6 Exception/Interrupt-Related Registers

8.6.1 Register List

The following table lists the control registers and their addresses:

8.6.1.1 NVIC Register List

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 to 0x0417
Vector Table Offset Register	0x0D08
Application Interrupt and Reset Control Register	0x0D0C
System Handler Priority Register	0x0D18, 0x0D1C, 0x0D20
System Handler Control and State Register	0x0D24

8.6.1.2 INTC Register List

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

peripheral funcion : INTC(ch0)

Register name	Register Name	Address
Interrupt Control Register 0	INTxCR0	0x0000
Interrupt Control Register 1	INTxCR1	0x0004
Interrupt Request Clear Register	INTxCLR	0x0044
Reset Flag Register	INTxRESETF	0x0048

8.6.2 NVIC Registers

8.6.2.1 SysTick Control and Status Register

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

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

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

8.6.2.2 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".

8.6.2.3 SysTick Current Value Register

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

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

8.6.2.4 SysTick Calibration Value Register

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

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

Note: This product does not prepare the calibration value.

8.6.2.5 Interrupt control registers

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

(1) Interrupt Set-Enable Register

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

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

Writing to "0" is no meaning.

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

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

Table 8-6 Interrupt Set-Enable Register

Bit symbol	Type	Function
SETENA	R/W	Interrupt No. [22:0] [Write] 0 : Don't care 1 : Enable interrupt [Read] 0 : The interrupt state is enabled 1 : The interrupt state is disabled

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	SETENA (Interrupt 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

(2) Interrupt Clear-Enable Register

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

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

Writing to "0" is no meaning.

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

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

Table 8-7 Interrupt Clear-Enable Register

Bit symbol	Type	Function
CLRENA	R/W	Interrupt No. [22:0] [Write] 0 : Don't care 1 : Disable interrupt [Read] 0 : The interrupt state is enabled 1 : The interrupt state is disabled

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit symbol	-	CLRENA (Interrupt 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

(3) Interrupt Set-Pending Register

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

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

Writing to "0" is no meaning.

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

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

Table 8-8 Interrupt Set-Pending Register

Bit symbol	Type	Function
SETPEND	R/W	Interrupt No. [22:0] [Write] 0 : Don't care 1 : Pending interrupt [Read] 0 : No pending 1 : Pending

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	SETPEND (Interrupt 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

(4) Interrupt Clear-Pending Register

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

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

Writing to "0" is no meaning.

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

Table 8-9 Interrupt Clear-Pending Register

Bit symbol	Type	Function
CLRPEND	R/W	Interrupt No. [22:0] [Write] 0 : Don't care 1 : Clear Pended interrupt [Read] 0 : No pending 1 : Pending

	31	30	29	28	27	26	25	24
bit symbol	-	-	-	-	-	-	-	-
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	23	22	21	20	19	18	17	16
bit symbol	-	CLRPEND (Interrupt 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

8.6.2.6 Interrupt Priority Register

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

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

	31	24 23	16 15	8 7	0
0xE000_E400	PRI_3	PRI_2	PRI_1	PRI_0	
0xE000_E404	PRI_7	PRI_6	PRI_5	PRI_4	
0xE000_E408	PRI_11	PRI_10	PRI_9	PRI_8	
0xE000_E40C	PRI_15	PRI_14	PRI_13	PRI_12	
0xE000_E410	PRI_19	PRI_18	PRI_17	PRI_16	
0xE000_E414	-	PRI_22	PRI_21	PRI_20	

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

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

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

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

8.6.2.7 Vector Table Offset Register

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

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

8.6.2.8 Application Interrupt and Reset Control Register

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

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

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

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

8.6.2.9 System Handler Priority Register

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

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

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

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

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

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

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

8.6.2.10 System Handler Control and State Register

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

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

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

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

8.6.3 Clock generator registers

8.6.3.1 INTxCRn (Interrupt Control Register n)

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

Bit	Bit Symbol	Type	Function
31-6	-	R	Read as 0.
5-4	ST[1:0]	R	<p>Active level detection status</p> <p>00: No active level is detected.</p> <p>01: Rising edge of a signal as an active level is detected.</p> <p>10: Falling edge of a signal as an active level is detected.</p> <p>11: Rising-falling edge of a signal as an active level is detected.</p> <p>When rising-falling edge is set as the active level with ACT[2:0], if both rising edge and falling edge are detected until the CPU reads this register, "11" is set to this bit.</p> <p>If the interrupt request is cleared by the Interrupt request clear register, this register is cleared.</p> <p>When rising-falling edge is not set as the active level, this register is read as 00.</p>
3	-	R	Read as 0.
2-0	ACT[2:0]	R/W	<p>Active level selection</p> <p>000: Detects "Low" level of the signal.</p> <p>001: Detects "High" level of the signal.</p> <p>010: Detects falling edge of the signal.</p> <p>011: Detects rising edge of the signal.</p> <p>100: Detects rising-falling edge of the signal.</p> <p>101: Reserved</p> <p>110: Reserved</p> <p>111: Reserved</p>

8.6.3.2 INTxCLR (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	-	-	-	-	-	-	-	CLR
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	-	R	Read as 0.
0	CLR	W	Clear interrupt requests. Set the interrupt number to clear the request. Read as 0. 0: Interrupt number 0 1: Interrupt number 1

8.6.3.3 INTxRESETF (Reset 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	-	-	-	RSTF4	RSTF3	-	-	RSTF0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-5	-	R	Read as 0.
4	RSTF4	R/W	Reset flag 4 [Write] 0 : Clear reset flag 1 : Don't care [Read] 0: - 1: Reset from SYSRESETREQ
3	RSTF3	R/W	Reset flag 3 [Write] 0 : Clear reset flag 1 : Don't care [Read] 0: - 1: Reset flag by WDT
2-1	-	R	Read as 0.
0	RSTF0	R/W	Reset flag 0 [Write] 0 : Clear reset flag 1 : don't care [Read] 0: - 1: Reset from $\overline{\text{RESET}}$ pin.

9. μ DMA Controller (μ DMAC)

9.1 Overview

9.1.1 Function List

The main functions per unit are shown as below:

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

Table 9-1 μ DMA outline (Per unit)

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

9.2 Block Diagram

The μ DMA controller contains the following blocks:

- APB block
This block controls the access to the control register.
- AHB block
This block controls the bus cycle of the DMA transfer.
- DMA control block
This block controls the whole operation of the DMA.

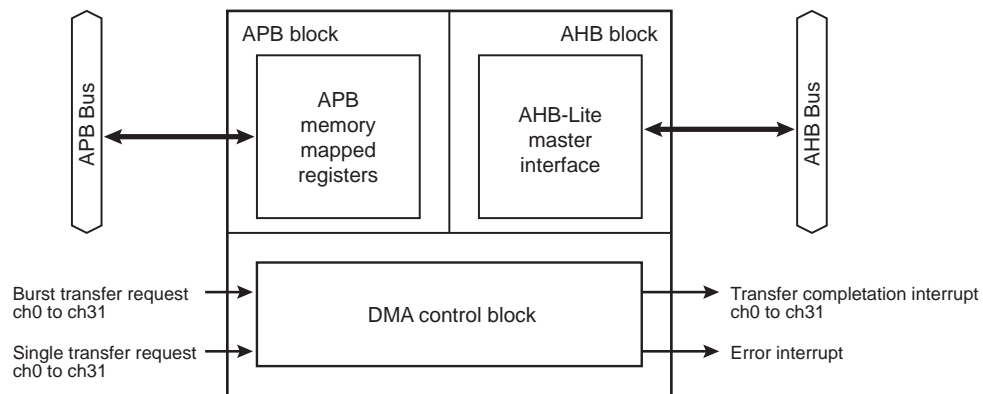


Figure 9-1 μ DMA block diagram

9.3 Registers

9.3.1 Register List

The following table shows control registers and addresses:

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

Then name of peripheral: DMA

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

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

9.3.2 DMAxStatus (DMAC Status Register)

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

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

9.3.3 DMAxCfg (DMAC Configuration Register)

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

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

Note: After DMAxCfg = 0x00000001, DMAxChnlReqMaskSet = 0xFFFFFFFF and DMAxChnlEnableSet = 0xFFFFFFFF are set, set "1" to the corresponding bit of DMAxChanlReqMaskClr to release masking of the channel to be used.

9.3.4 DMAxCtrlBasePtr (Channel Control Data Base-pointer Register)

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

Bit	Bit symbol	Type	Function
31-10	ctrl_base_ptr	R/W	Primary data base-pointer Specifies the base address of the primary data.
9-0	-	R	Read as "0".

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

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

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

9.3.6 DMAxChnlSwRequest (Channel Software Request Register)

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

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

9.3.7 DMAxChnlUseburstSet (Channel useburst Set Register)

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

Bit	Bit symbol	Type	Function
31-0	chnl_useburst_set	R/W	<p>Single-transfer is disabled [Write] 1: Single-transfer is disabled.</p> <p>[Read] 0: Single-transfer is enabled. 1: Single-transfer is disabled.</p> <p>Each bit corresponds to the channels in the specified number.</p> <p>Writing "1" disables the single-transfer to the corresponding channel, and only burst transfer request becomes valid. Writing "0" has no meaning. Set the DMAxChnlUseburstClr register in order to cancel the disabled the single-transfer.</p> <p>By reading the bit, the channel state of the corresponding bit can be checked whether it is enabled or disabled.</p> <p>Bits are automatically set in the following conditions:</p> <ul style="list-style-type: none"> This bit is cleared to "0", if the number of remaining transfer is less than 2^R times at the end of second 2^R time transfer from the end ("R" is specified by the channel_cfg<R_power> of the control data). If the channel_cfg<next_useburst> of the control data is set to "1" in the peripheral scatter/gather mode, this bit is set to "1" when the DMA transfer of the alternative data ends.

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

9.3.8 DMAxChnlUseburstClr (Channel useburst Clear Register)

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

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

9.3.9 DMAxChnlReqMaskSet (Channel Request Mask Set Register)

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

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

9.3.10 DMAxChnlReqMaskClr (Channel Request Mask Clear Register)

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

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

9.3.11 DMAxChnlEnableSet (Channel Enable Set Register)

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

Bit	Bit symbol	Type	Function
31-0	chnl_enable_set	R/W	<p>DMA operation</p> <p>[Write]</p> <p>1: Enable the corresponding channel.</p> <p>[Read]</p> <p>0: The corresponding bit is invalid.</p> <p>1: The corresponding bit is valid.</p> <p>Each bit corresponds to the channels in the specified number.</p> <p>Writing "1" enables the corresponding channels. Writing "0" has no meaning. To disable the setting, configure the DMAxChnlEnableClr register.</p> <p>By reading the bit, the corresponding channel can be checked whether it is enabled or disabled.</p> <p>In the following conditions, the function automatically becomes invalid.</p> <ul style="list-style-type: none"> • DMA cycle ends. • If the channel_cfg<cycle_ctrl> reads the control data of "000". • A bus error occurs.

9.3.12 DMAxChnlEnableClr (Channel Enable Clear Register)

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

Bit	Bit symbol	Type	Function
31-0	chnl_enable_clr	W	<p>DMA disabled</p> <p>1: Disables the corresponding channel.</p> <p>Each bit corresponds to the channels in the specified number.</p> <p>Writing "1" disables the corresponding channel. Writing "0" has no meaning.</p> <p>Configure the DMAxChnlEnableSet register in order to enable and confirm the setting.</p> <p>In the following conditions, the function automatically becomes invalid.</p> <ul style="list-style-type: none"> • DMA cycle ends. • The channel_cfg<cycle_ctrl> reads the control data of "000". • A bus error occurs.

9.3.13 DMAxChnlPriAltSet (Channel Primary-alternate Set Register)

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

Bit	Bit symbol	Type	Function
31-0	chnl_pri_alt_set	R/W	<p>Selects primary data or alternative data</p> <p>[Write]</p> <p>1: Uses alternative data</p> <p>[Read]</p> <p>0: Primary data</p> <p>1: Alternative data</p> <p>Each bit corresponds to the channels in the specified number.</p> <p>Writing "1" specifies the data that is firstly used in the corresponding channel as "alternative data". Writing "0" has no meaning. To disable this bit, use the DMAxChnlEnableClr register.</p> <p>Only in basic mode, automatic request mode, and ping-pong mode the first data can be specified as alternative data.</p> <p>When this bit is read, data of the corresponding channel can be checked whether data is primary data or alternative data.</p> <p>In the following conditions, the settings are automatically changed.</p> <ul style="list-style-type: none"> The primary data transfer is completed in ping-pong mode, memory scatter / gather mode or peripheral scatter / gather mode. Data transfer of the alternative data is completed in the ping-pong mode, memory scatter / gather mode or peripheral scatter / gather mode.

9.3.14 DMAxChnlPriAltClr (Channel Primary-alternate Clear Register)

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

Bit	Bit symbol	Type	Function
31-0	chnl_pri_alt_clr	W	<p>Clears the alternative data setting.</p> <p>1: Uses the primary data</p> <p>Each bit corresponds to the channels in the specified number.</p> <p>Writing "1" sets the data of the corresponding channel to the primary data. Setting "0" is invalid. Configure the DMAxChnlPriAltSet register to set the primary data or to confirm the setting.</p> <p>In the following conditions, the setting is automatically changed.</p> <ul style="list-style-type: none"> • The primary data transfer in memory scatter/gather mode or peripheral scatter/gather mode is complete. • The primary data transfer in ping-pong mode is complete. • The alternative transfer in ping-pong mode, memory scatter/gather mode, or peripheral scatter/gather mode is complete.

9.3.15 DMAxChnlPrioritySet (Channel Priority Set Register)

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

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

9.3.16 DMAxChnlPriorityClr (Channel Priority Clear Register)

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

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

9.3.17 DMAxErrClr (Bus Error Clear Register)

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

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

9.4 Operation

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

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

9.4.1 Channel Control Data Memory Map

Figure 9-2 shows the example of memory map of the channel control data.

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

Alternate Ch31	0x3F0	Primary Ch31	0x1F0
Alternate Ch30	0x3E0	Primary Ch30	0x1E0
Alternate Ch29	0x3D0	Primary Ch29	0x1D0
Alternate Ch28	0x3C0	Primary Ch28	0x1C0
Alternate Ch27	0x3B0	Primary Ch27	0x1B0
Alternate Ch26	0x3A0	Primary Ch26	0x1A0
Alternate Ch25	0x390	Primary Ch25	0x190
Alternate Ch24	0x380	Primary Ch24	0x180
Alternate Ch23	0x370	Primary Ch23	0x170
Alternate Ch22	0x360	Primary Ch22	0x160
Alternate Ch21	0x350	Primary Ch21	0x150
Alternate Ch20	0x340	Primary Ch20	0x140
Alternate Ch19	0x330	Primary Ch19	0x130
Alternate Ch18	0x320	Primary Ch18	0x120
Alternate Ch17	0x310	Primary Ch17	0x110
Alternate Ch16	0x300	Primary Ch16	0x100
Alternate Ch15	0x2F0	Primary Ch15	0x0F0
Alternate Ch14	0x2E0	Primary Ch14	0x0E0
Alternate Ch13	0x2D0	Primary Ch13	0x0D0
Alternate Ch12	0x2C0	Primary Ch12	0x0C0
Alternate Ch11	0x2B0	Primary Ch11	0x0B0
Alternate Ch10	0x2A0	Primary Ch10	0x0A0
Alternate Ch9	0x290	Primary Ch9	0x090
Alternate Ch8	0x280	Primary Ch8	0x080
Alternate Ch7	0x270	Primary Ch7	0x070
Alternate Ch6	0x260	Primary Ch6	0x060
Alternate Ch5	0x250	Primary Ch5	0x050
Alternate Ch4	0x240	Primary Ch4	0x040
Alternate Ch3	0x230	Primary Ch3	0x030
Alternate Ch2	0x220	Primary Ch2	0x020
Alternate Ch1	0x210	Primary Ch1	0x010
Alternate Ch0	0x200	Primary Ch0	0x000

Reserved	0x00C
Control	0x008
Destination End Pointer	0x004
Source End Pointer	0x000

Figure 9-2 Memory map of the control data

Figure 9-2 shows the memory map of which all 32 channels can be used. Necessary areas are determined by the number of usable channels. Table 9-2 shows the relationship between the number of channels and addresses.

Table 9-2 Address bit setting of channel control

Channel	Address						[3:0]	Settable base address
	[9]	[8]	[7]	[6]	[5]	[4]		
0	-	-	-	-	-	A	Channel control data setting	0xFFFF_X000, 0xFFFF_X020, 0xFFFF_X040, 0xFFFF_X060, 0xFFFF_X080, 0xFFFF_X0A0, 0xFFFF_X0C0, 0xFFFF_X0E0
0 to 1	-	-	-	-	A	C[0]		0xFFFF_X000, 0xFFFF_X040, 0xFFFF_X080, 0xFFFF_X0C0
0 to 3	-	-	-	A	C[1:0]			0xFFFF_X000, 0xFFFF_X080
0 to 7	-	-	A	C[2:0]				0xFFFF_X000, 0xFFFF_X100, 0xFFFF_X200, 0xFFFF_X300, 0xFFFF_X400, 0xFFFF_X500, 0xFFFF_X600, 0xFFFF_X700, 0xFFFF_X800, 0xFFFF_X900, 0xFFFF_XA00, 0xFFFF_XB00, 0xFFFF_XC00, 0xFFFF_XD00, 0xFFFF_XE00, 0xFFFF_XF00
0 to 15	-	A	C[3:0]					0xFFFF_X000, 0xFFFF_X200, 0xFFFF_X400, 0xFFFF_X600, 0xFFFF_X800, 0xFFFF_XA00, 0xFFFF_XC00, 0xFFFF_XE00
0 to 31	A	C[4:0]						0xFFFF_X000, 0xFFFF_X400, 0xFFFF_X800, 0xFFFF_XC00

A: Primary/alternative setting (0:primary, 1:alternative)

C[x:0]: Channel number setting

9.4.2 Channel Control Data Structure

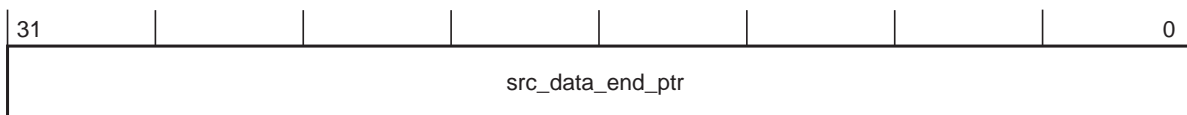
The channel control data contains the three kinds of data shown below:

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

Each data is described in the following sections:

9.4.2.1 Final Address of the Transfer Source Data

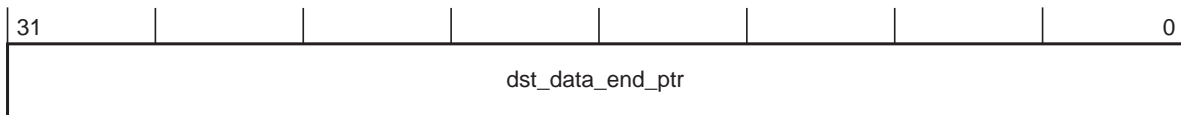
Specify the final address of the data to be transferred. The alignment of an address should be adjusted to a transfer data size. The DMA calculates the start address of the source address using this data.



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

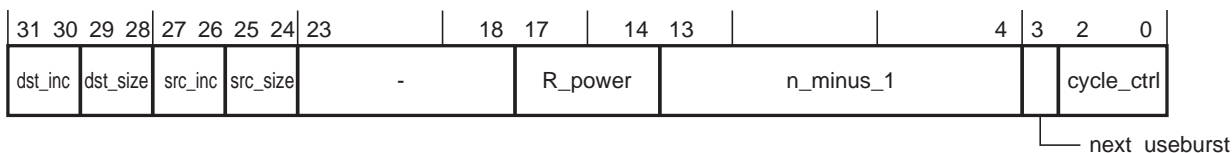
9.4.2.2 Final Address of the Transfer Destination Address

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



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

9.4.2.3 Control Data Setting



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

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

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

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

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

9.4.3 Operation Modes

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

9.4.3.1 Invalid Setting

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

9.4.3.2 Basic Mode

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

A transfer is started by receiving a transfer request.

An arbitration is performed for every transfer configured by $\langle R_power \rangle$. If a higher-priority request exists, the DMA switches a channel. If a transfer request for the operating channel is received, the transfer is continued.

After performing transfers for the number of times specified by $\langle n_minus_1 \rangle$, a transfer completion interrupt occurs.

9.4.3.3 Automatic Request Mode

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

The DMA transfer is started by a transfer request.

In each transfer configured by $\langle R_power \rangle$, a channel is switched if a higher-priority request is received. If not, the transfer is continued.

After performing transfers for the number of times specified by $\langle n_minus_1 \rangle$, a transfer completion interrupt occurs.

9.4.3.4 Ping-pong Mode

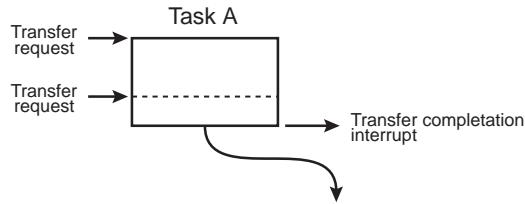
In ping-pong mode, a continuous DMA transfer that uses primary data and alternative data alternately is performed. If $\langle cycle_ctrl \rangle$ reads data specified to be invalid ("000"), or the channel is specified to be invalid, the transfer is stopped. Every time a DMA transfer (task) that uses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

Preparation:

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

Task A: Primary data

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



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

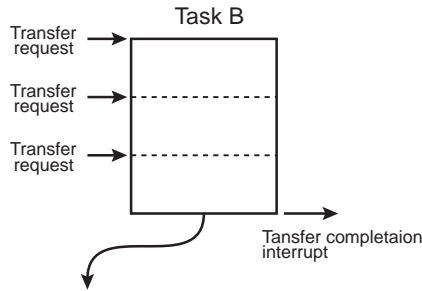
If there is no other high-priority requests, the DMA performs remaining transfers twice toward a request for a transfer to the corresponding channels.

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

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

Task B: Alternative data

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



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

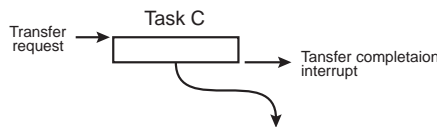
If there is no other high-priority requests, The DMA performs transfers twice toward a request for a transfer to the corresponding channels.

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

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

Task C: Primary data

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



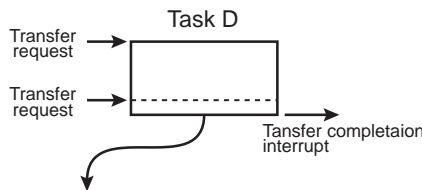
Receiving a transfer request, the DMA performs a transfer twice and performs arbitration.

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

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

Task D: Alternative data

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



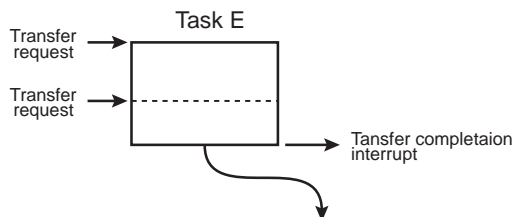
Receiving a transfer request, the DMA performs a transfer four times and performs arbitration.

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

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

Task E: Primary data

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



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

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

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

Final: Alternative data

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



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

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

9.4.3.5 Memory Scatter/Gather Mode

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

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

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

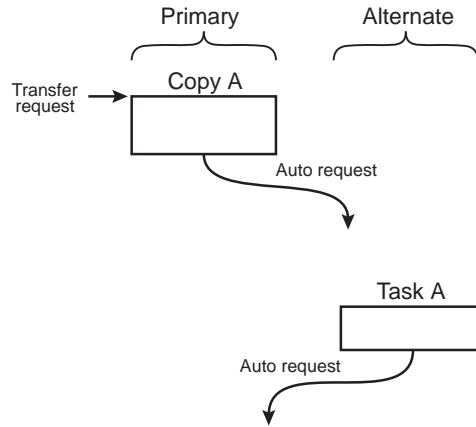
Table 9-3 Setting values of Memory scatter/gather mode (Primary data)

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

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

Preparation: Prepare primary data. Set "100" to <cycle_ctrl> and set four task data $4 \times 4 = 16$ as the number of transfers <n_minus_1>. Set alternative data for Task A,B,C and D to the memory location which is set to the <src_data_end_ptr>. Set "1" to bits of channels corresponding to DMAxCfg <master_enable> and DMAxChnlSet.

Copy A: Primary data
 <cycle_ctrl[2:0]> = "100"
 (Memory scatter / gather mode)
 <R_power[3:0]> = "0010"
 (4 times)
 <n_minus_1[9:0]> = "0x00F" (16 times)

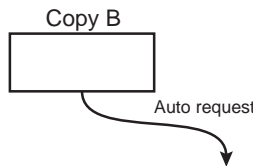


Receiving a transfer request, the DMA performs a transfer for alternative data of Task A for four times. After completing the transfer, a transfer request is automatically generated and arbitration starts.

Task A: Alternative data
 <cycle_ctrl[2:0]> = "100"
 <R_power[3:0]> = "0010"
 (4 times)
 <n_minus_1[9:0]> = "0x002" (3 times)

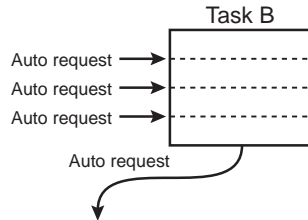
The DMA performs Task A. After completing the transfer, a transfer request is automatically generated and arbitration starts.

Copy B: Primary data



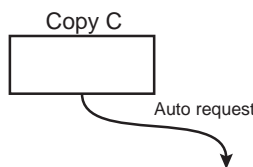
The DMA performs transfers for alternative data of Task B for four times. After completing the transfer, a transfer request is automatically generated and arbitration starts.

Task B: Alternative data
 <cycle_ctrl[2:0]> = "100"
 <R_power[3:0]> = "0001"
 (2 times)
 <n_minus_1[9:0]> = "0x007" (8 times)



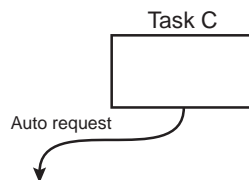
The DMA performs Task B. After completing the transfer, a transfer request is automatically generated and arbitration starts.

Copy C: Primary data



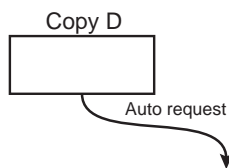
The DMA performs transfers for alternative data of Task C for four times. After completing the transfer, a transfer request is automatically generated and arbitration starts.

Task C: Alternative data
 <cycle_ctrl[2:0]> = "100"
 <R_power[3:0]> = "0011"
 (8 times)
 <n_minus_1[9:0]> = "0x004" (5 times)



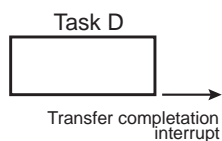
The DMA performs Task C. After completing the transfer, a transfer request is automatically generated and arbitration starts.

Copy D: Primary data



The DMA performs transfers for alternative data of Task D for four times. The DMA also sets "000" to <cycle_ctrl> of the primary data in order to set the next primary data invalid. A transfer request is automatically generated and arbitration starts.

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



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

9.4.3.6 Peripheral Scatter/Gather Mode

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

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

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

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

Table 9-4 Fixed values in peripheral scatter / gather mode (Primary data)

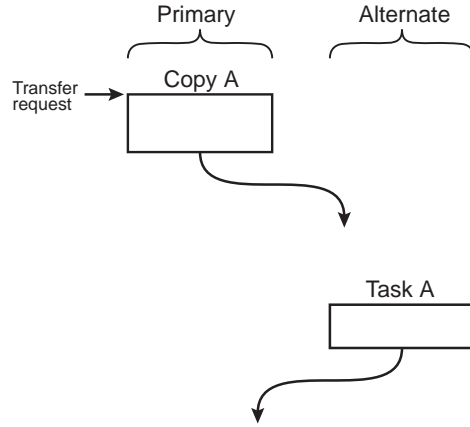
Bit	Bit symbol	Setting value	Description
[31:30]	dst_inc	10	A 4-byte increment is specified for transfer destination address.
[29:28]	dst_size	10	4 bytes are specified as transfer destination address.
[27:26]	src_inc	10	A 4-byte increment is specified for transfer source address.
[25:24]	src_size	10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

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

Preparation:

Prepare primary data. Set "110" to <cycle_ctrl> and $4 \times 4 = 16$ for four tasks to the number of transfers <n_minus_1>. Set alternative data for Task A,B,C and D to the memory location which is set to the <src_data_end_ptr>. Set "1" to bits of channels corresponding to DMAxCfg <master_enable> and DMAxChnlEnableSet.

Copy A: Primary data
 <cycle_ctrl[2:0]> = "110"
 (Peripheral scatter / gather)
 <R_power[3:0]> = "0010"
 (4 times)
 <n_minus_1[9:0]> = "0x00F" (16 times)

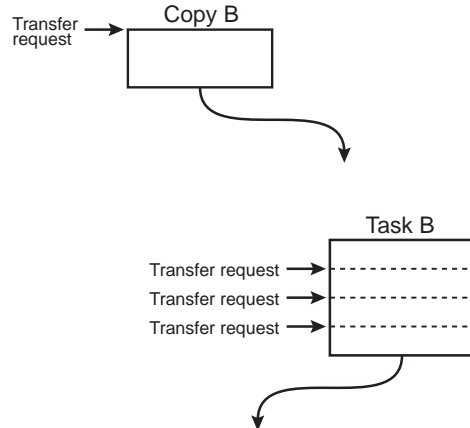


Receiving a transfer request, the DMA performs transfers for alternative data of Task A for four times. After completing the transfer, operation automatically moves onto Task A.

Task A: Alternative data
 <cycle_ctrl[2:0]> = "111"
 <R_power[3:0]> = "0010"
 (4 times)
 <n_minus_1[9:0]> = "0x002" (3 times)

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

Copy B: Primary data

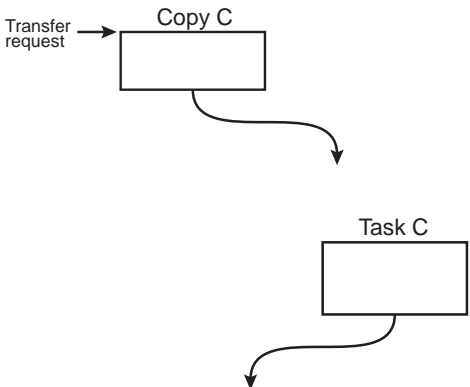


The DMA performs transfers for alternative data of Task B for four times. After completing the transfer, processing of Task B automatically starts.

Task B: Alternative data
 <cycle_ctrl[2:0]> = "111"
 <R_power[3:0]> = "0001"
 (2 times)
 <n_minus_1[9:0]> = "0x007" (8 times)

The DMA performs Task B. Since an arbitration occurs every 2^R times of transfers, three times of transfer is required at least to complete Task B. After completing the transfer, if a transfer request is sent from peripheral function and if it is high-priority request, the next operation starts

Copy C: Primary data

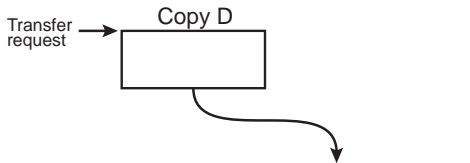


The DMA performs transfers for alternative data of Task C for four times. After completing the transfer, operation automatically moves onto Task C.

Task C: Alternative data
 <cycle_ctrl[2:0]> = "111"
 <R_power[3:0]> = "0011"
 (8 times)
 <n_minus_1[9:0]> = "0x004" (5 times)

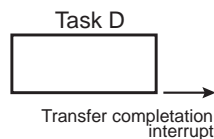
The DMA performs Task C. After completing the transfer, if a transfer request is sent from peripheral function and if it is high-priority request, the next operation starts.

Copy D: Primary data



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

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



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

9.5 Precautions

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

- Synchronous serial interface (SSP)
- Serial channel with 4-byte FIFO (SIO/UART)

9.5.1 When the SSP is Used

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

- Transmission
 - It is recommended to use a basic mode as a transfer mode.
 - Disable the single-transfer.
 - The following two methods are used according to the number of transfers:
 - a. When a DMA transfer rate is set to "after 1 transfer" as an arbitration rate.
 - This method can be used in any cases.
 - Specify "0000" as the arbitration rate setting <R_power> for the control data.
 - b. When the number of transfers is a multiple of the watermark level of the FIFO.
 - This method can be used when the number of transfers is a multiple of the watermark level of the FIFO and the watermark level and the number of arbitration rate are the same number.
 - Set the number of arbitration rates <R_power> for the control data to a multiple of the watermark level of the FIFO.
- Reception
 - Use the SSP and UART according to the number of transfers as follows:
 - a. Less than the watermark level
 - Only a single-transfer request occurs.
 - It is recommended to use a basic mode as a transfer mode.
 - Specify "0000" as the arbitration rate <R_power> for the control data.
 - b. A multiple of the watermark level
 - Disable the single-transfer.
 - It is recommended to use a basic mode as a transfer mode.
 - Set the number of arbitration rates <R_power> for the control data to a multiple of the watermark level of the FIFO.
 - c. Other than the above

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

Prepare two tasks.

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

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

9.5.2 SIO/UART is Used

The following points should be considered:

- It is recommended to use the basic mode as a transfer mode.
- Set "after 1 transfer" as a DMA transfer rate.
 - Specify "0000" as the arbitration rate <R_power> for the control data.
- Do not use the FIFO of the SIO/UART.

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

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

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

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

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

10. Input / Output port

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

10.1 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.
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.

10.1.1 Register list

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

When the bits that does not exist in the function are read as "0". Writing to them does not influence.

Register name	Address (Base+)	PORT A	PORT B	PORT C	PORT D
Data register	0x0000	PADATA	PBDATA	PCDATA	PDDATA
Output control register	0x0004	PACR	PBCR	PCCR	PDCR
Function register 1	0x0008	PAFR1	PBFR1	PCFR1	PDFR1
Pull-up control register	0x002C	PAPUP	PBPUP	PCPUP	PDPUP
Pull-down control register	0x0030	-	-	PCPDN	-
Input control register	0x0038	PAIE	PBIE	PCIE	PDIE

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

10.1.2 Port function and setting list

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

- "Table 10-1 PORT A Setting List"
- "Table 10-2 PORT B Setting List"
- "Table 10-3 PORT C Setting List"
- "Table 10-4 PORT D Setting List"

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

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

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

10.1.2.1 PORT A

Table 10-1 PORT A Setting List

PO RT	Reset status	Input/ Output	PORT Type	Control registers					
				PADATA	PACR	PAFRn	PAPUP	PAPDN	PAIE
PA0	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	SP0CLK	Input	FT2	0/1	0	PAFR1	0/1		1
Output		0/1		1	PAFR1	0/1		0	
PA1	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	SP0DO	Output	FT2	0/1	1	PAFR1	0/1		0
PA2	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	SP0DI	Input	FT2	0/1	0	PAFR1	0/1		1
PA3	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	SP0FSS	Input	FT2	0/1	0	PAFR1	0/1		1
Output		0/1		1	PAFR1	0/1		0	
PA4	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	SC0SCLK	Input	FT1	0/1	0	PAFR1	0/1		1
Output		0/1		1	PAFR1	0/1		0	
PA5	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	SC0RXD	Input	FT1	0/1	0	PAFR1	0/1		1
PA6	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	SC0TXD	Output	FT1	0/1	1	PAFR1	0/1		0
PA7	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	INT1	Input	FT4	0/1	0	0	0/1		1

10.1.2.2 PORT B

Table 10-2 PORT B Setting List

PO RT	Reset status	Input/ Output	PORT Type	Control registers					
				PBDATA	PBCR	PBFRn	PBPUP	PBPDN	PBIE
PB0	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	TB0IN0	Input	FT1	0/1	0	PBFR1	0/1		1
PB1	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	TB0OUT	Output	FT1	0/1	1	PBFR1	0/1		0
PB2	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	TB1IN0	Input	FT1	0/1	0	PBFR1	0/1		1
PB3	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	TB1OUT	Output	FT1	0/1	1	PBFR1	0/1		0
PB4	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	DSADEOC	Output	FT1	0/1	1	PBFR1	0/1		0
PB5	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	DSADTRG	Input	FT1	0/1	0	PBFR1	0/1		1
PB6	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
PB7	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0

10.1.2.3 PORT C

Table 10-3 PORT C Setting List

PO RT	Reset status	Input/ Output	PORT Type	Control registers					
				PCDATA	PCCR	PCFRn	PCPUP	PCPDN	PCIE
PC0	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
	INT0	Input	FT4	0/1	0	0	0/1		1
PC1	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
PC2	After reset (Note)			0	1	PCFR1	1		1
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0
PC3	After reset (Note)			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1	0/1	1
	Output Port	Output		0/1	1	0	0/1	0/1	0
PC4	After reset			0	0	0	0		0
	Input Port	Input		0/1	0	0	0/1		1
	Output Port	Output		0/1	1	0	0/1		0

Note: After reset is released, change the value of settings if needed.

10.1.2.4 PORT D

Table 10-4 PORT D Setting List

PO RT	Reset status	Input/ Output	PORT Type	Control registers					
				PDDATA	PDCR	PDFRn	PDPUP	PDPDN	PDIE
PD0	After reset			0	0		0		0
	Input Port	Input		0/1	0		0/1		1
	Output Port	Output		0/1	1		0/1		0
	X1: EHOSC	Input		0/1	0		0		0
	X1: EHCLKIN	Input		0/1	0		0		0
PD1	After reset			0	0		0		0
	Input Port	Input		0/1	0		0/1		1
	Output Port	Output		0/1	1		0/1		0
	X2	Input		0/1	0		0		0

Note: When the external high-speed oscillator pins (X1 and X2) are used, the all registers for Port D set to "0" (disable). The clock inputs from external high-speed oscillator are used, the CGxOSCEN<EHOSCEN> must be set. For the value of setting, refer to a chapter of the "Clock control".

10.2 Block Diagrams of Ports

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

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

The operation of "Direct reset" shown in the circuit diagram is enabled when turn on the power.

10.2.1 Type FT1

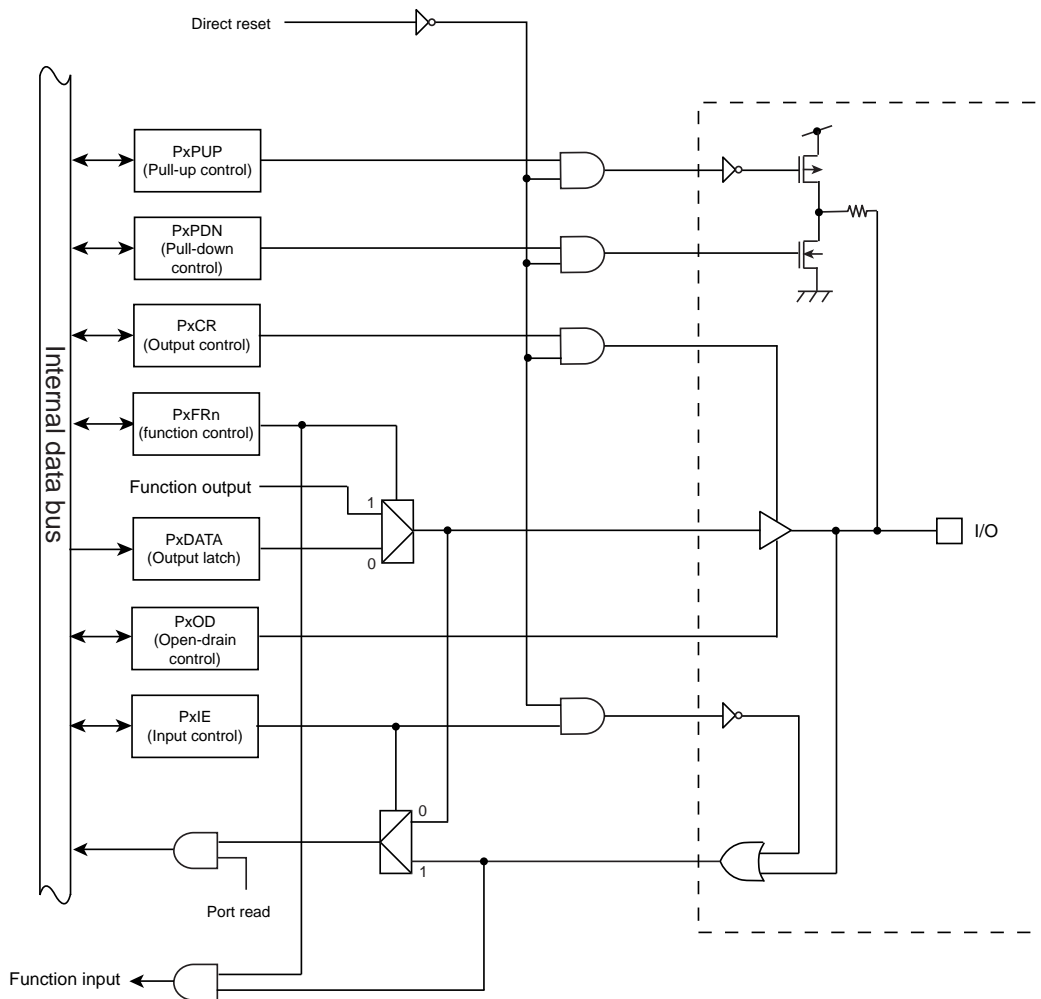


Figure 10-1 Port Type FT1

10.2.2 Type FT2

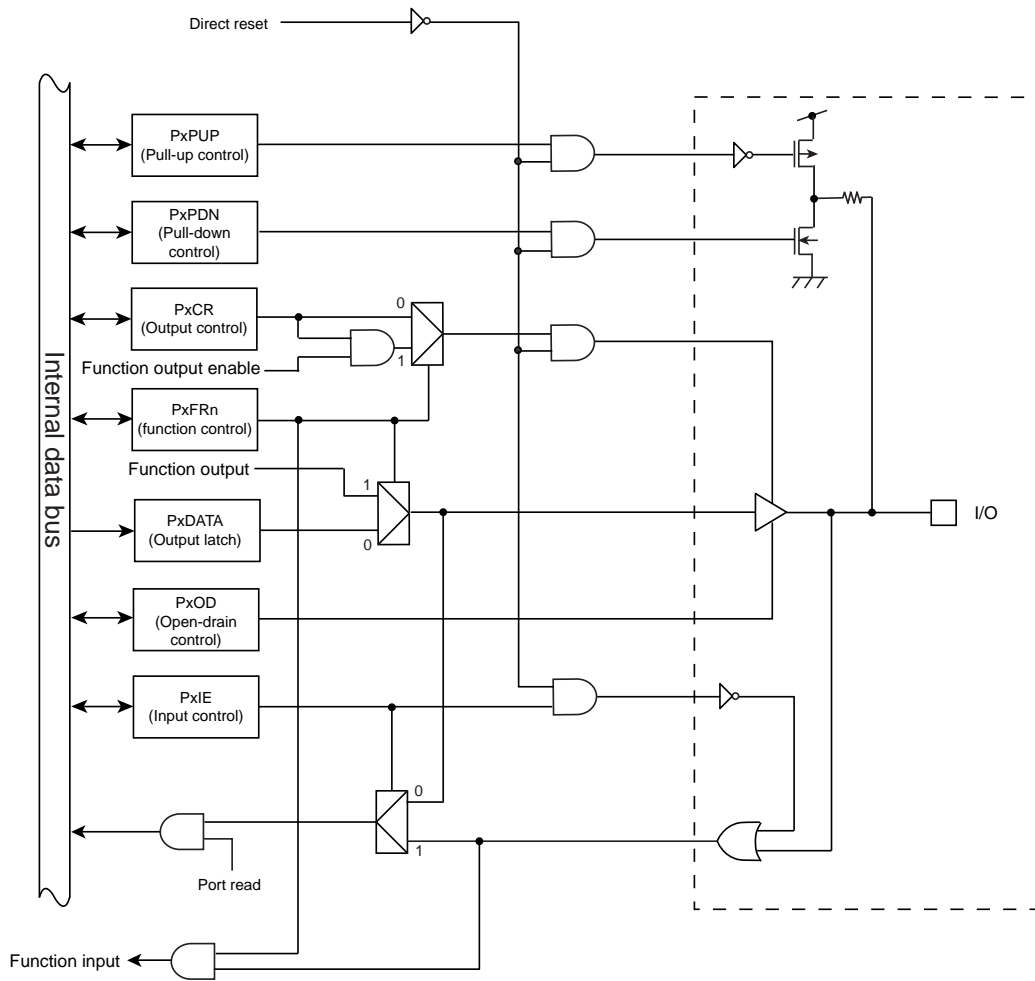


Figure 10-2 Port Type FT2

10.2.3 Type FT4

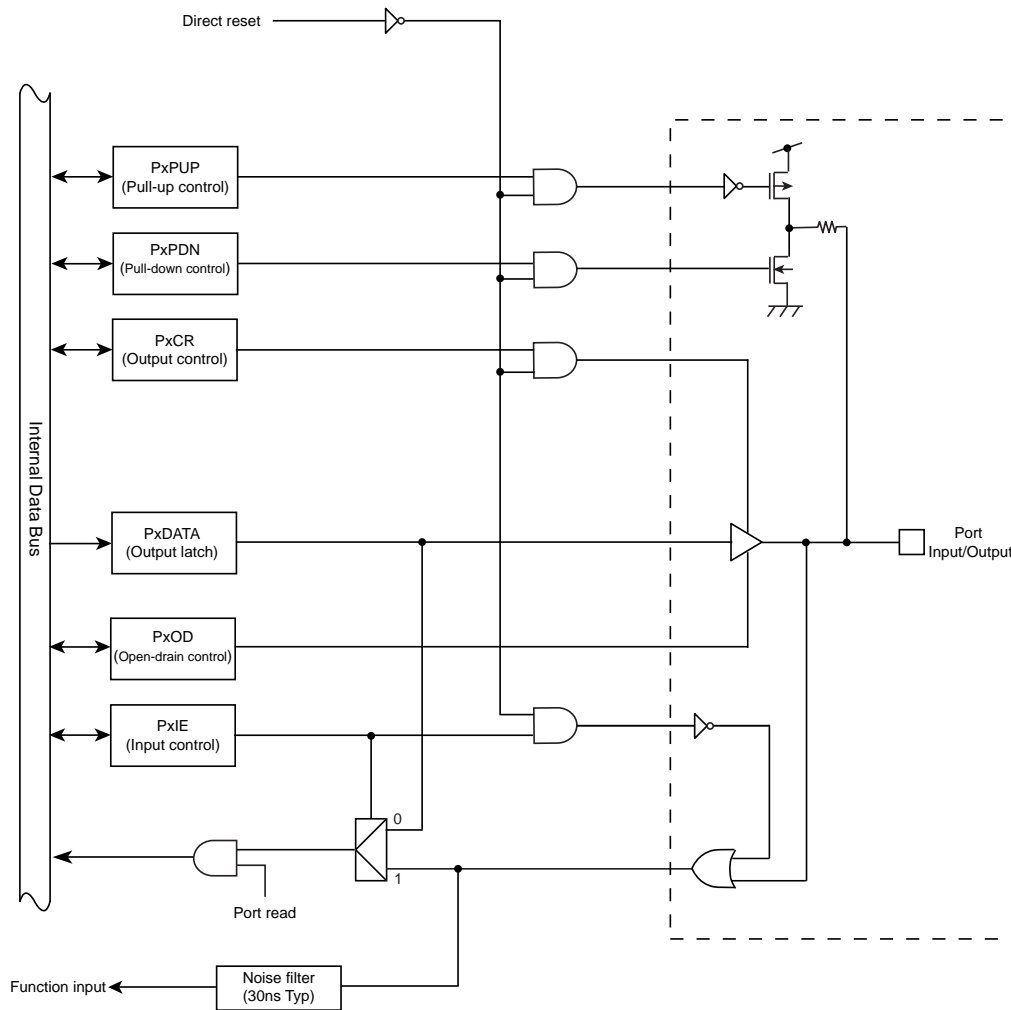


Figure 10-3 Port Type FT4

11. 16-bit Timer / Event Counters (TMRB)

11.1 Outline

TMRB has the operation modes shown as 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 TMRB to perform the following measurements.

- Frequency measurement
- Pulse width measurement

In the following explanation, "x" indicates a channel number.

11.2 Block Diagram

TMRB 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 a register.

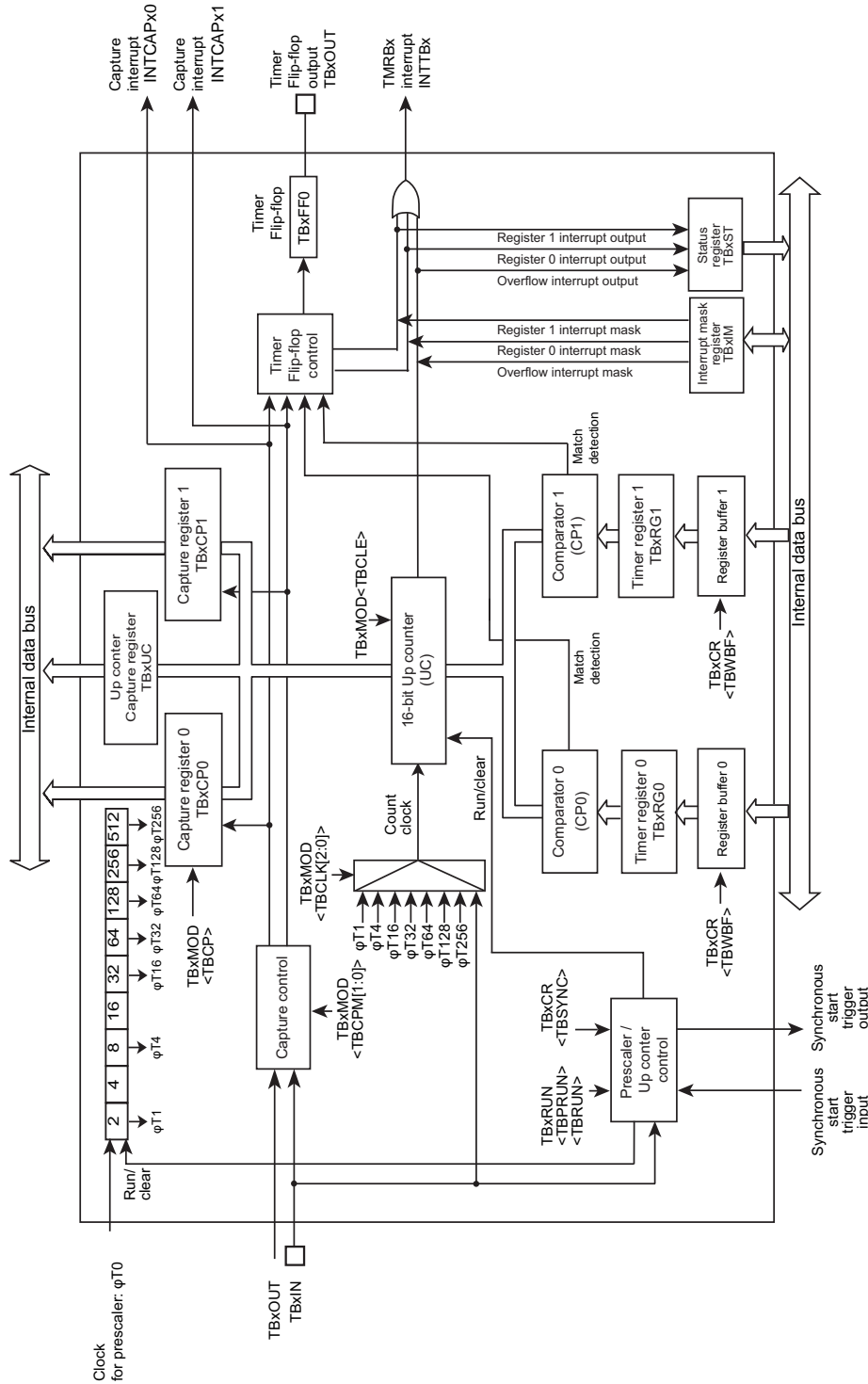


Figure 11-1 TMRBx Block Diagram

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	TBxEN	0x0000
RUN register	TBxRUN	0x0004
Control register	TBxCR	0x0008
Mode register	TBxMOD	0x000C
Flip-flop control register	TBxFFCR	0x0010
Status register	TBxST	0x0014
Interrupt mask register	TBxIM	0x0018
Up counter capture register	TBxUC	0x001C
Timer register 0	TBxRG0	0x0020
Timer register 1	TBxRG1	0x0024
Capture register 0	TBxCP0	0x0028
Capture register 1	TBxCP1	0x002C

Note: During timer operation, timer control register, timer mode register and timer flip-flop control register should not be modified. After stopping timer operation, they should be modified.

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

11.3.3 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

11.3.4 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	CSEL
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 external triggers. 0: rising 1: falling Controls the edge selection (of signal to TBxIN pin) when the external triggers is selected.
0	CSEL	R/W	Selects the count start 0: starts by software 1: starts by external trigger

11.3.5 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) takes count value. Read as "1".
5-4	TBCPM[1:0]	R/W	Capture timing 00: Disable Capture 01: Reserved 10: TBxIN \uparrow , TBxIN \downarrow Takes count values into capture register 0 (TBxCP0) upon rising of TBxIN pin input. Takes count values into capture register 1 (TBxCP1) upon falling of TBxIN pin input. 11: TBxFF0 \uparrow , TBxFF0 \downarrow Takes count values into capture register 0 (TBxCP0) upon rising of TBxFF0 and into capture register 1 (TBxCP1) upon falling of TBxFF0.
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 register1 (TBxRG1).
2-0	TBCLK[2:0]	R/W	Selects the TMRBx source clock. 000: TBxIN pin input 001: ϕ T1 010: ϕ T4 011: ϕ T16 100: ϕ T32 101: ϕ T64 110: ϕ T128 111: ϕ T256

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

11.3.6 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 (reverse by using software). 01: Set Sets TBxFF0 to "1". 10: Clear Clears TBxFF0 to "0". 11: Don't care This is always read as "11".

11.3.7 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, TBxST register should be read.

11.3.8 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.

11.3.9 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.

11.3.10 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.

11.3.11 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.

11.3.12 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.

11.3.13 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.

11.4 Description of Operation

11.4.1 Prescaler

There is prescaler to generate the source clock for up-counter.

The prescaler input $\phi T0$ is $f_{periph}/1$, $f_{periph}/2$, $f_{periph}/4$, $f_{periph}/8$, $f_{periph}/16$ or $f_{periph}/32$ selected CGSYSCR<PRCK[2:0] in the CG circuit. The peripheral clock is either f_{gear} , a clock selected by CGSYSCR<FPSEL> in the CG circuit, or f_c , which is a clock before it is divided by the clock gear.

The operation or the stoppage of a prescaler is set with TBxRUN<TBPRUN> where writing "1" status counting and writing "0" clears and stops counting.

11.4.2 Up-counter (UC)

UC is a 16-bit binary counter.

11.4.2.1 Source clock

UC's source clock is specified by TBxMOD<TBCLK[2:0]>.

It can be selected from the prescaler output clock - $\phi T1$, $\phi T4$, $\phi T16$, $\phi T32$, $\phi T64$, $\phi T128$ and $\phi T256$ - or the external clock of the TBxIN pin.

11.4.2.2 Counter start / stop

To start the counter, there are a software start, external trigger start and synchronous start.

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 a 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.

11.4.2.3 Timing to clear UC

1. When a match with TBxRG1 is detected

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

2. When UC stops

UC stops and is cleared if TBxRUN<TBRUN> = "0".

11.4.2.4 UC overflow

If UC overflow occurs, the INTTBx overflow interrupt is generated.

11.4.3 Timer registers (TBxRG0, TBxRG1)

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

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

Controlling double buffering disable or enable is specified by TBxCR<TBWBF>. If <TBWBF> = 0, the double buffering becomes disable, If <TBWBF> = "1", it becomes enable.

When the double buffering is enabled, data transfers from the register buffer to the timer register (TBxRG0/1) in the case that UC is matched with TBxRG1.

When UC 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.

11.4.4 Capture control

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

Software can also capture the value of UC to capture registers. The value of UC are taken into the TBxCP0 each time "0" is written to TBxMOD<TBCP>.

11.4.5 Capture registers (TBxCP0, TBxCP1)

This register captures the value of UC.

11.4.6 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.

11.4.7 Comparators (CP0, CP1)

This circuit compares with UC and the value set to TBxRG0/1 and detects match. If a match is detected, INTTBx is occurred.

11.4.8 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. It can be enabled or disabled to reverse 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, the corresponding port settings should be programmed beforehand.

11.4.9 Capture interrupt (INTCAPx0, INTCAPx1)

INTCAPx0 and INTCAPx1 can be generated at the timing of latching value from UC into the TBxCP0 and TBxCP1.

11.5 Description of Operation for each mode

11.5.1 Interval timer mode

In the case of generating constant period interrupt, set the interval time to the timer register (TBxRG1) to generate the INTTBx interrupt.

	7	6	5	4	3	2	1	0	
TBxEN	← 1	X	X	X	X	X	X	X	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 corresponding bit to "1".
TBxFFCR	← X	X	0	0	0	0	1	1	disable to TBÇ0FF0 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

11.5.2 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 UC counts up on the rising edge of TBxIN pin input. The value of UC can be captured by soft capture. It is possible to read the count value by reading it.

	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 a 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

11.5.3 Programmable pulse generation (PPG) output mode

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

TBxFF0 is reversed when UC 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 satisfy the following requirement.

Set value of TBxRG0 < Set value of TBxRG1

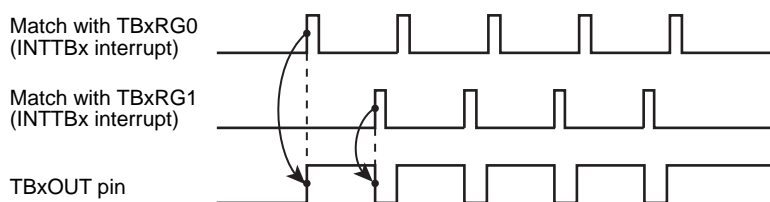


Figure 11-2 Example of Programmable pulse generation output

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

It is possible to modify frequency and duty without timing of modifying TBxRG0 and TBxRG1.

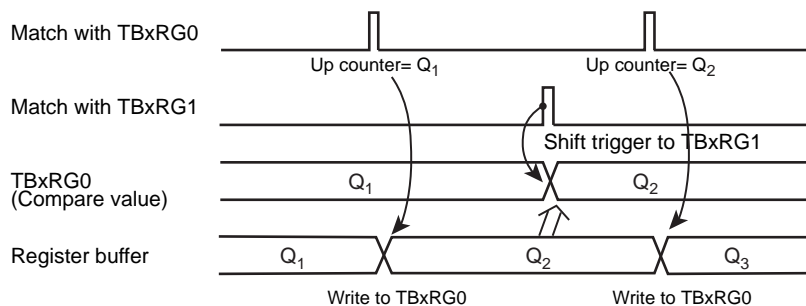


Figure 11-3 Register Buffer Operation

The block diagram of this mode is shown below.

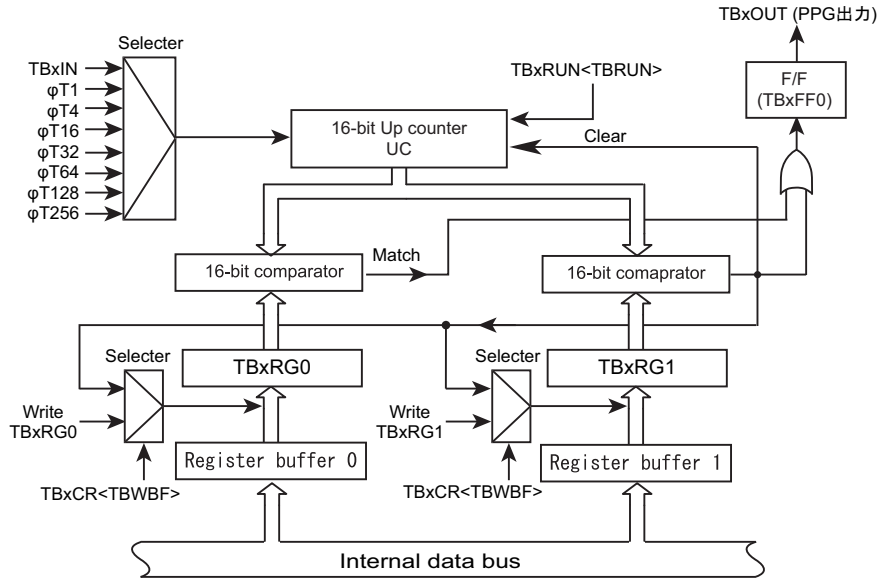


Figure 11-4 Block diagram of 16-bit PPG mode

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	Specifies to trigger TBxFF0 to reverse when a match with TBxRG0 or TBxRG1 is detected. And sets the initial value of TBxFF0 to "0".
TBxMOD	←	X	1	0	0	0	*	*	*	Designates the prescaler output clock as the input clock, and disable the capture function.
							(***) = 001 to 111)			
Assigns a 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

11.5.4 Programmable pulse generation (PPG) external trigger output mode

A PPG wave with a short delay time can be output by using external trigger count start mode.

The example of an one-shot pulse output by external trigger count start mode is shown below.

To start count up by the rising edge of TBxIN, set TBxCR<CSSEL> to "1" and clear TBxCR<TRGSEL> to "0" in stopping 16-bit up counter.

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

To reverse TBxFF0 when UC matches TBxRG0 and TBxRG1, TBxFFCR<TBE1T1> and TBxFFCR<TBE1T1> are set to "1".

UC is readied to start UC by setting TBxRUN<TBPRUN> and TBxRUN<TBRUN> to "1".

UC starts by the rising edge of external trigger.

TBxFF0 is reversed when UC counts up to (d) and UC matches TBxRG0. TBxFF0 is "High" level.

TBxFF0 is reversed when UC counts up to (d)+(p) and UC matches TBxRG1. TBxFF0 is "Low" level.

To fix the level of TBxFF0, clear TBxFFCR<TBE1T1> and TBxFFCR<TBE0T1> to "0" or stops UC by TBxRUN<TBPRUN><TBRUN> in INTTBx which is generated when UC matches TBxRG1.

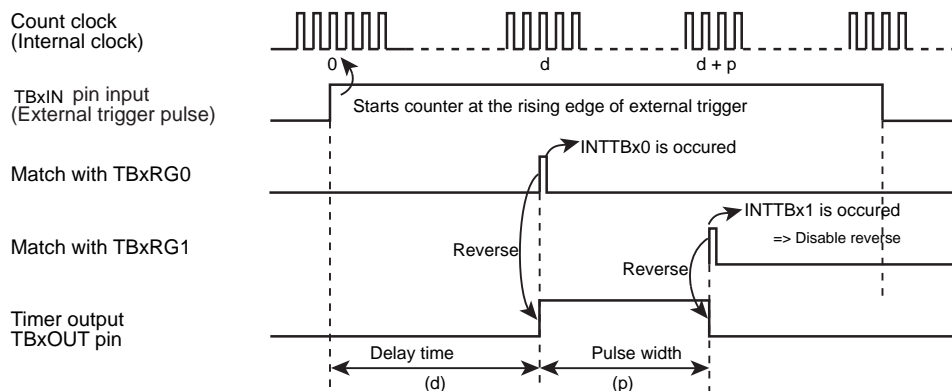


Figure 11-5 One-shot pulse output with delay by external trigger start

The followings shows the setting in the case that 2 ms width one-shot pulse is output after 3 ms by triggering TBxIN input at the rising edge. In this example, the source clock is $\phi T1$.

	7	6	5	4	3	2	1	0	
[Main processing]									
Assigns a corresponding port to TBxIN.									
TBxEN	←	1	X	X	X	X	X	X	Enables TMRBx operation.
TBxRUN	←	X	X	X	X	X	0	X	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	Reverses TBxFF0 if UC matches TBxRG0 and TBxRG1. Clear TBxFF0 to "0".
TBxMOD	←	X	1	0	0	0	0	1	Starts UC as free-running. Selects φT1 for the source clock. Disable capture UC.
Assigns a corresponding port to TBxOUT.									
TBxIM	←	X	X	X	X	X	1	0	Masks except TBxRG1 interrupt.
Interrupt set-enable register	←	*	*	*	*	*	*	*	Permits to generate interrupt specified by INTTBx interrupt corresponding bit setting to "1".
TBxRUN	←	X	X	X	X	X	1	X	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	Stops prescaler and counter.

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

11.6 Applications using the capture function

The capture function can be used many applications.

The applications are shown below.

1. Frequency measurement
2. Pulse width measurement

11.6.1 Frequency measurement

The frequency of an external clock can be measured.

To measure frequency, TMRBm is used as 16-bit interval timer mode and TMRBn is used as 16-bit event counter mode.

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

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

To capture UC to TBnCP0 at rising edge of TBmFF0 and UC to TBmCP1 at falling edge of TBmFF0, set TBxMOD<TBCPM> to "11".

Set TBmRG0 and TBmRG1 to time when UC counts an external clock and start TMRBm.

Rises-up TBmFF0 when UC of TMRBm matches TBmRG0 and captures the value of TMRBn's UC to TBnCP0. Falls-down TBmFF0 when UC of TMRBm matches TBmRG1 and captures the value of TMRBn's UC to TBnCP1.

A frequency is measured from $(TBnCP1 - TBnCP0) \div (TBmRG1 - TBmRG0)$ in 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 ($100 \div 0.5 \text{ s} = 200\text{Hz}$)

TBnCP1 - TBnCP0 may be less than zero depend on the changing timing of TBmFF0. Please correct the value if TBnCP1 - TBnCP0 is less than zero.

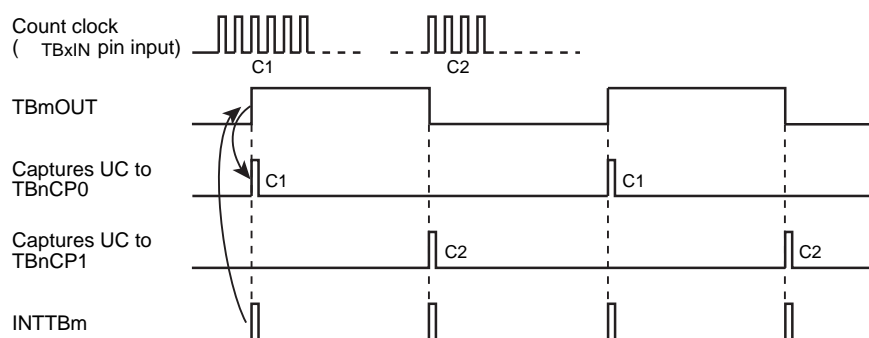


Figure 11-6 Frequency measurement

The following shows in the case that the measured pulse is input to TBxIN. In this example, the source clock is $\phi T1$.

	7	6	5	4	3	2	1	0	
Assigns a corresponding port to TBxIN.									
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	← *	*	*	*	*	*	*	*	Set the external clock measured time 1.
	← *	*	*	*	*	*	*	*	
TBmRG1	← *	*	*	*	*	*	*	*	Set the external clock measured time 2.
	← *	*	*	*	*	*	*	*	
TBmFFCR	← X	X	0	0	1	1	1	0	Reverses TBxFF0 if UC matches TBxRG0 and TBxRG1. Clear TBxFF0 to "0".
TBnMOD	← 0	1	1	1	0	0	0	0	Captures at the rising / falling edge. Clears and disables UC. Input clock is TBxIN.
TBmIM	← X	X	X	X	X	1	0	1	Masks except TBxRG1 interrupt.
Interrupt set-enable register	← *	*	*	*	*	*	*	*	Permits to generate interrupt specified by INTTBm interrupt corresponding bit setting to "1".
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	← *	*	*	*	*	*	*	*	Prohibits interrupt specified by INTTBm corresponding bit by setting to "1".
TBnCP0 and TBnCP1 are read out and the frequency is calculated.									

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

11.6.2 Pulse width measurement

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

To capture UC to TBxCP0 at rising edge of TBxIN and UC to TBxCP1 at falling edge of TBxIN, set TBxMOD<TBCPM> to "10".

Enables INTCAPx1 interrupt.

Enables TMRBx operation.

Captures the value of UC to TBxCP0 when the rising edge of the external pulse into TBxIN. Captures the value of UC to TBxCP1 when the falling edge of the external pulse into TBxIN and INTCAPx1 interrupt is occurred.

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 μ s, the pulse width is $100 \times 0.5 \mu\text{s} = 50 \mu\text{s}$.

When the pulse width which is more than maximum count time of UC is measured, please correct the measured value.

The "Low" level width of an external pulse can also be measured.

In this case, enables INTCAPx0 interrupt. In twice process of INTCAPx0 interrupt, the difference between C2 generated the first time and C1 generated the second time in "Figure 11-7 Pulse width measurement" is multiplied by the cycle of the prescaler output clock.

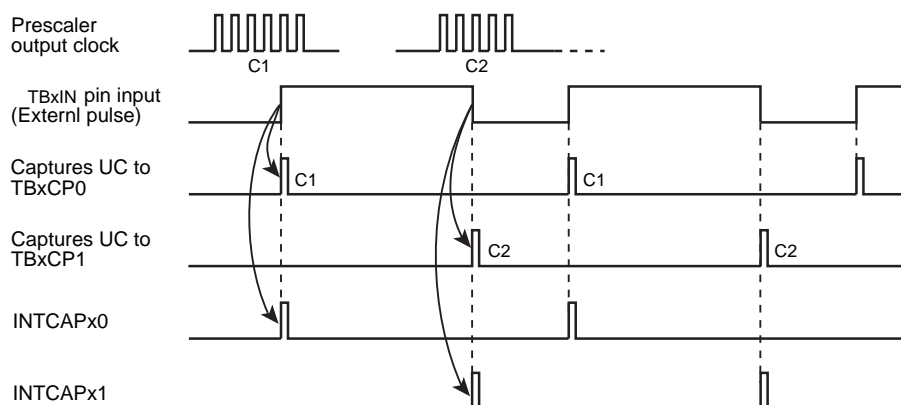


Figure 11-7 Pulse width measurement

The following is shown that the "High" level width of the external pulse into TBxIN is measured. In this example, the source clock is $\phi T1$.

	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	Clears TBxFF0 reverse trigger and TBxFF0.
TBxMOD	← X	1	1	0	0	0	0	1	Starts UC as free-running. Selects φT1 for the source clock. UC is captured to TBxCP0 at the rising edge of TBxIN. UC is captured to TBxCP1 at the falling edge of TBxIN.
Interrupt set-enable register	← *	*	*	*	*	*	*	*	Permits to generate interrupt specified by INTCAPx1 interrupt corresponding bit setting to "1".
TBxRUN	← X	X	X	X	X	1	X	1	Starts prescaler and counter.
[Processing INTCAPx1 interrupt service routine] Calculate the width of "High" level.									
Interrupt enable clear register	← *	*	*	*	*	*	*	*	Prohibits interrupt specified by INTCAPx1 interrupt corresponding bit setting to "1".
Calculated the width of "High" level by reading TBxRG0 and TBxRG1.									

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

12. 16-Bit Timer A (TMR16A Ver.B)

12.1 Outline

TMR16A contains the following functions:

- Match interrupt
- Square waveform output
- Read capture

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

12.2 Block Diagram

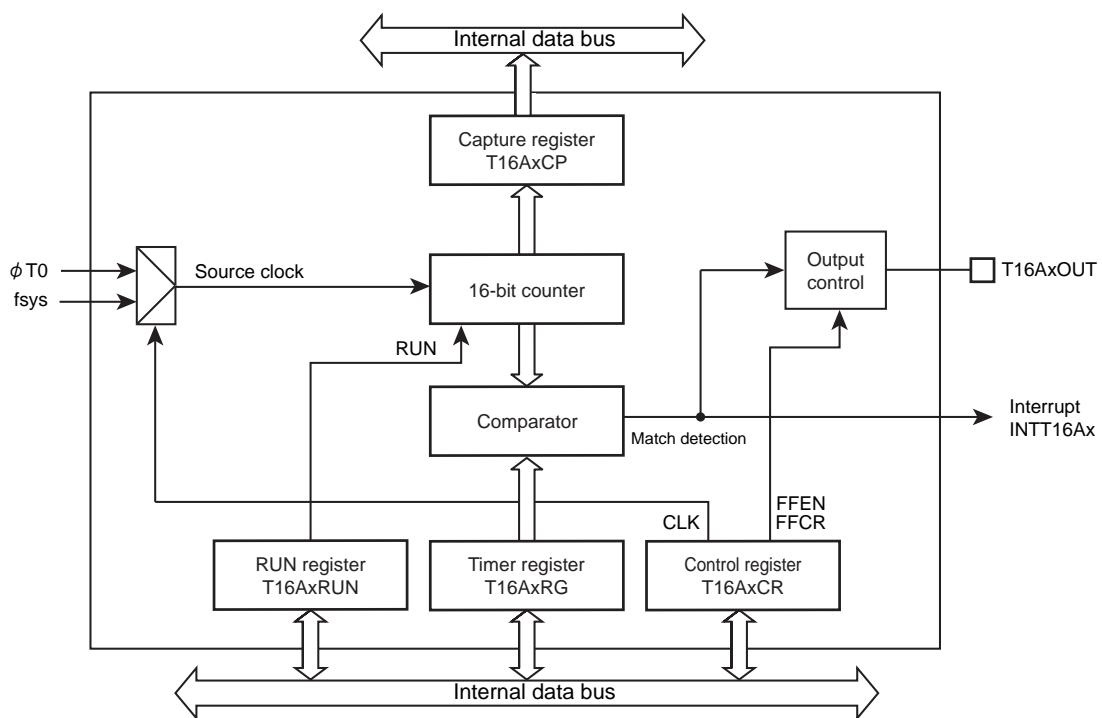


Figure 12-1 Block diagram of TMR16A

12.3 Registers

12.3.1 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.

12.3.1.1 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.

12.3.1.2 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

12.3.1.3 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: $\Phi T0$ Specifies a source clock.

12.3.1.4 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".

12.3.1.5 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.

12.4 Operation Description

12.4.1 Timer Operation

1. Preparation

Select a source clock with T16AxCR<CLK>. Write "0" to set fsys or write "1" to set $\Phi T0$. 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").

12.4.2 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.

12.4.3 Read Capture

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

12.4.4 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.

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

13.1 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 ($\phi T0$) 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{SCxCTS} pin
 - Noise cancel for SCxRXD pin

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

13.2 Configuration

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

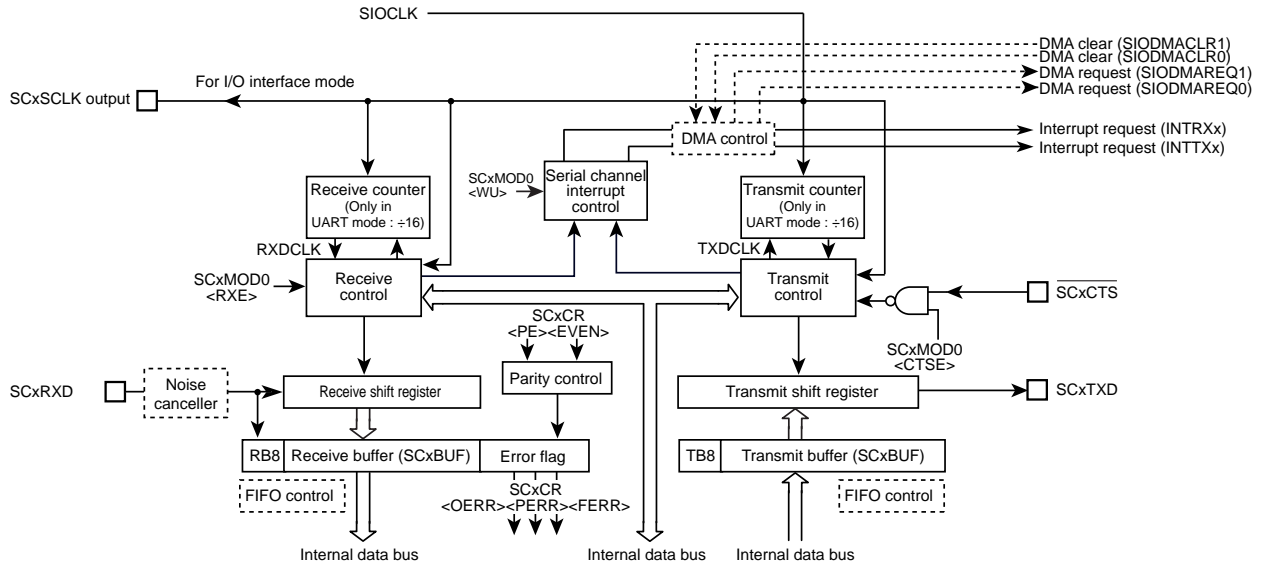


Figure 13-1 Serial Channel Block Diagram

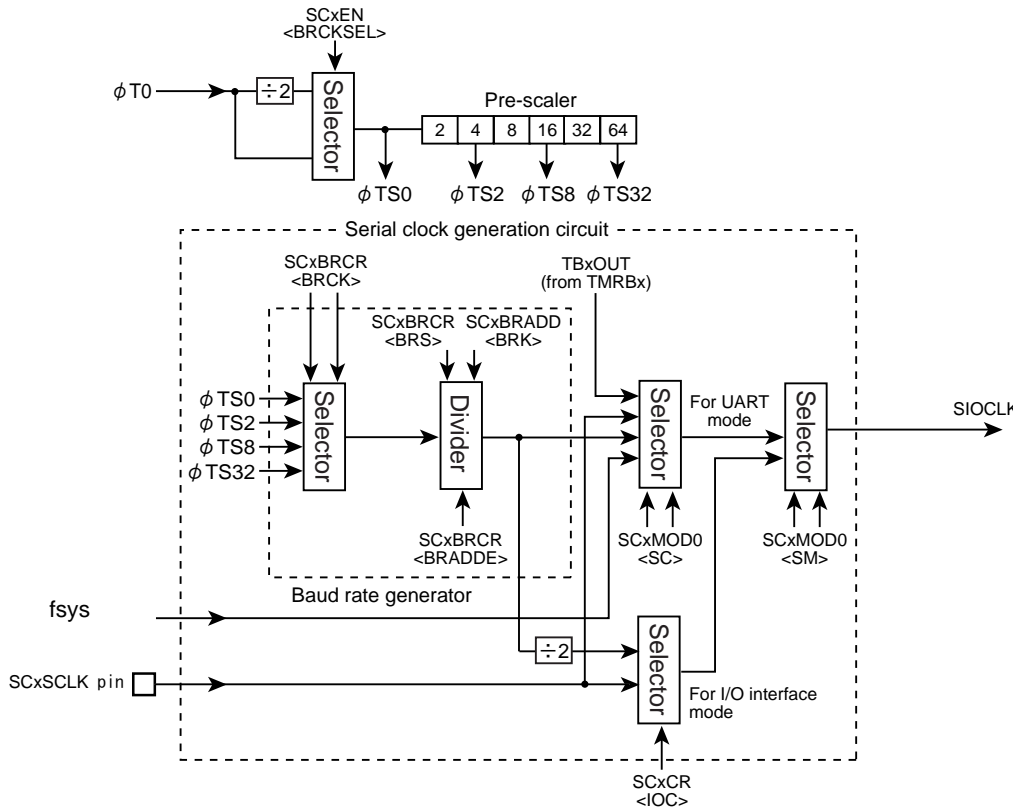


Figure 13-2 Serial clock generation circuit block diagram

13.3 Registers Description

13.3.1 Registers List

The table below shows control registers and their addresses.

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

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

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

13.3.2 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: $\phi T0/2$ 1: $\phi T0$
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.

13.3.3 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

13.3.4 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.

13.3.5 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 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.

13.3.6 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.

13.3.7 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	RBFL	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	<p>Transmit buffer empty flag</p> <p>0: Full 1: Empty</p> <p>If double buffering is disabled, this flag is insignificant.</p> <p>This flag shows that the transmit double buffers are empty.</p> <p>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".</p> <p>Writing data again to the double buffers sets this bit to "0".</p>											
6	RBFL	R	<p>Receive buffer full flag</p> <p>0: Empty 1: Full</p> <p>If double buffering is disabled, this flag is insignificant.</p> <p>This is a flag to show that the receive double buffers are full.</p> <p>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".</p>											
5	TXRUN	R	<p>In transmission flag</p> <p>0: Stop 1: Operate</p> <p>This is a status flag to show that data transmission is in progress.</p> <p><TXRUN> and <TBEMP> bits indicate the following status.</p> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th><TXRUN></th><th><TBEMP></th><th>Status</th></tr> </thead> <tbody> <tr> <td>1</td><td>-</td><td>Transmission in progress</td></tr> <tr> <td rowspan="2">0</td><td>1</td><td>Transmission is completed.</td></tr> <tr> <td>0</td><td>Wait state with data in transmit buffer</td></tr> </tbody> </table>	<TXRUN>	<TBEMP>	Status	1	-	Transmission in progress	0	1	Transmission is completed.	0	Wait state with data in transmit buffer
<TXRUN>	<TBEMP>	Status												
1	-	Transmission in progress												
0	1	Transmission is completed.												
	0	Wait state with data in transmit buffer												
4	SBLEN	R/W	<p>STOP bit length (for UART mode)</p> <p>0: 1-bit 1: 2-bit</p> <p>This specifies the length of transmission stop bit in the UART mode.</p> <p>On the receive side, the decision is made using only a single bit regardless of the <SBLEN>.</p>											
3	DRCHG	R/W	<p>Setting transfer direction</p> <p>0: LSB first 1: MSB first</p> <p>Specifies the direction of data transfer.</p> <p>In the UART mode, set this bit to LSB first.</p>											
2	WBUF	R/W	<p>Enable double-buffer</p> <p>0: Disabled 1: Enabled</p> <p>This parameter enables or disables the transmit/receive double buffers to transmit (in both SCLK output/input modes) and receive (in SCLK output mode) data in the I/O interface mode and to transmit in the UART mode.</p> <p>When receiving data in the I/O interface mode (in clock input mode) and UART mode, double buffering is enabled regardless of the <WBUF>.</p>											

Bit	Bit Symbol	Type	Function										
1-0	SWRST[1:0]	R/W	<p>Software reset</p> <p>Overwriting "01" in place of "10" generates a software reset.</p> <p>When a software reset is executed, the following bits are initialized and the transmit/receive circuit and FIFO become initial state (Note1)(Note2).</p> <table border="1"> <thead> <tr> <th>Register</th> <th>Bit</th> </tr> </thead> <tbody> <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> </tbody> </table>	Register	Bit	SCxMOD0	<RXE>	SCxMOD1	<TXE>	SCxMOD2	<TBEMP>, <RBFLL>, <TXRUN>	SCxCR	<OERR>, <PERR>, <FERR>
Register	Bit												
SCxMOD0	<RXE>												
SCxMOD1	<TXE>												
SCxMOD2	<TBEMP>, <RBFLL>, <TXRUN>												
SCxCR	<OERR>, <PERR>, <FERR>												

Note 1: While data transmission is in progress, any software reset operation must be executed twice in succession.

Note 2: A software reset requires 2 clocks-duration at the time between the end of recognition and the start of execution of software reset instruction.

13.3.8 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.

13.3.9 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 13-1 lists the settings of baud rate generator division ratio.

Table 13-1 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.

13.3.10 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. <table border="1" data-bbox="529 1550 1383 1742"> <tr> <td>Half duplex Receive</td><td>When the receive shift register, receive buffers and receive FIFO are filled up to the specified number of valid bytes, SCxMOD0<RXE> is automatically set to "0" to inhibit further reception.</td></tr> <tr> <td>Half duplex Transmit</td><td>When the transmit shift register, transmit buffers and the transmit FIFO are empty, SCxMOD1<TXE> is automatically set to "0" to inhibit further transmission.</td></tr> <tr> <td>Full duplex</td><td>When either of the above two conditions is satisfied, <TXE> and <RXE> are automatically set to "0" to inhibit further transmission and reception.</td></tr> </table>	Half duplex Receive	When the receive shift register, receive buffers 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 transmit 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.
Half duplex Receive	When the receive shift register, receive buffers 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 transmit 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: <table border="1" data-bbox="529 1921 1383 2076"> <tr> <td>Half duplex Receive</td><td>Receive FIFO 4bytes</td></tr> <tr> <td>Half duplex Transmit</td><td>Transmit FIFO 4bytes</td></tr> <tr> <td>Full duplex</td><td>Receive FIFO 2bytes and Transmit FIFO 2bytes</td></tr> </table>	Half duplex Receive	Receive FIFO 4bytes	Half duplex Transmit	Transmit FIFO 4bytes	Full duplex	Receive FIFO 2bytes and Transmit FIFO 2bytes
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.

13.3.11 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 "13.13.1.2 FIFO"															
5-2	-	R	Read as "0".															
1-0	RIL[1:0]	R/W	FIFO fill level to generate receive interrupts. <table border="1"> <thead> <tr> <th></th><th>Half duplex</th><th>Full duplex</th></tr> </thead> <tbody> <tr> <td>00</td><td>4 bytes</td><td>2 bytes</td></tr> <tr> <td>01</td><td>1 byte</td><td>1 byte</td></tr> <tr> <td>10</td><td>2 bytes</td><td>2 bytes</td></tr> <tr> <td>11</td><td>3 bytes</td><td>1 byte</td></tr> </tbody> </table>		Half duplex	Full duplex	00	4 bytes	2 bytes	01	1 byte	1 byte	10	2 bytes	2 bytes	11	3 bytes	1 byte
	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")

13.3.12 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 "13.13.2.2 FIFO"															
5-2	-	R	Read as "0".															
1-0	TIL[1:0]	R/W	Fill level which transmit interrupt is occurred. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th></th> <th>Half duplex</th> <th>Full duplex</th> </tr> </thead> <tbody> <tr> <td>00</td> <td>Empty</td> <td>Empty</td> </tr> <tr> <td>01</td> <td>1 byte</td> <td>1 byte</td> </tr> <tr> <td>10</td> <td>2 bytes</td> <td>Empty</td> </tr> <tr> <td>11</td> <td>3 bytes</td> <td>1 byte</td> </tr> </tbody> </table>		Half duplex	Full duplex	00	Empty	Empty	01	1 byte	1 byte	10	2 bytes	Empty	11	3 bytes	1 byte
	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.

13.3.13 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.

13.3.14 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.

13.4 Operation in Each Mode

Table 13-2 shows the modes.

Table 13-2 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	0	1 bit or 2 bits
Mode 2		8 bits		0	
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.

13.5 Data Format

13.5.1 Data Format List

Figure 13-3 shows data format.

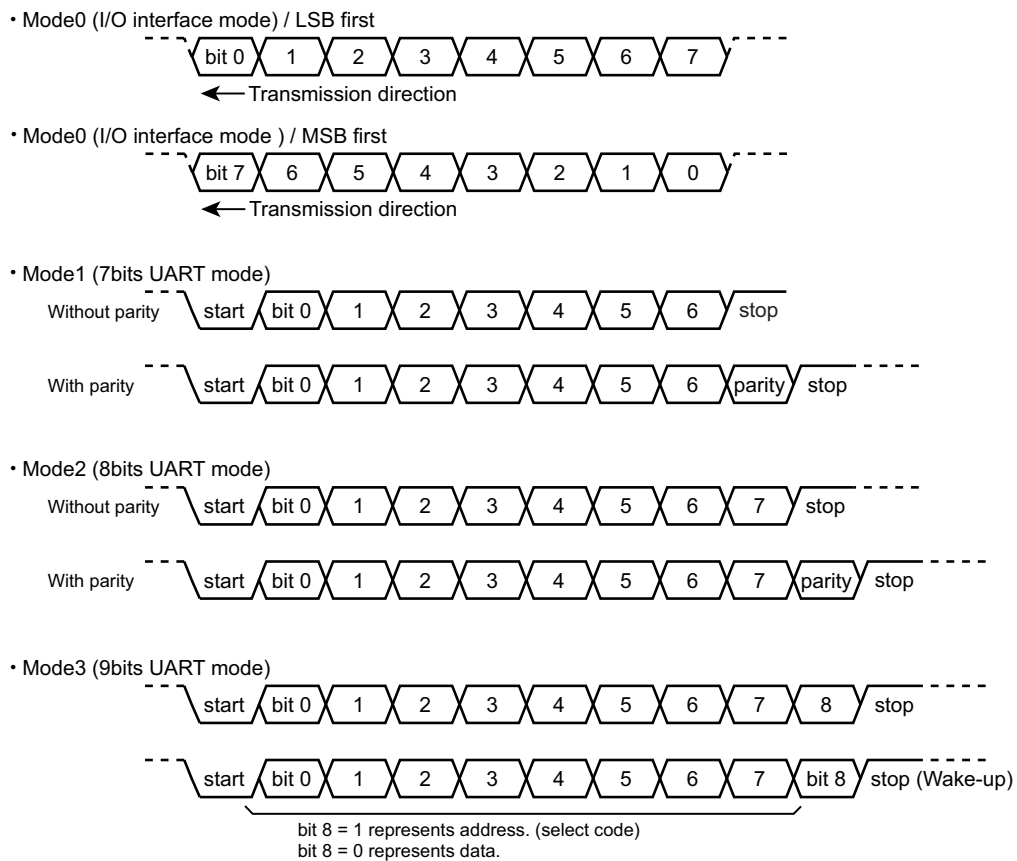


Figure 13-3 Data Format

13.5.2 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.

13.5.2.1 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.

13.5.2.2 Reception

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.

13.5.3 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<SBLN>. The length of the STOP bit data is determined as one-bit when it is received regardless of the setting of this bit.

13.6 Clock Control

13.6.1 Prescaler

There is a 7-bit prescaler to divide a prescaler input clock $\phi T0$ 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".

13.6.2 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.

13.6.2.1 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 ϕT_x
0	00	$\phi T0/2$
0	01	$\phi T0/8$
0	10	$\phi T0/32$
0	11	$\phi T0/128$
1	00	$\phi T0$
1	01	$\phi T0/4$
1	10	$\phi T0/16$
1	11	$\phi T0/64$

(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

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

13.6.2.2 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 13-3 shows clock selection in I/O interface mode.

Table 13-3 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 13-4 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 13-4 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 $\Phi T1$ (2division ratio) is selected.
 └ One clock cycle is a period that the timer flip-flop is inverted twice.

13.7 Transmit/Receive Buffer and FIFO

13.7.1 Configuration

Figure 13-4 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.

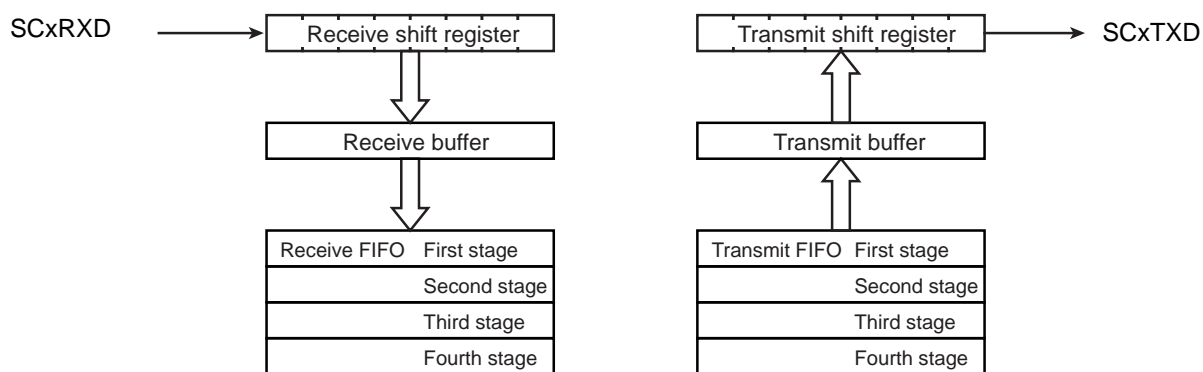


Figure 13-4 The Configuration of Buffer and FIFO

13.7.2 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 13-5 shows correlation between modes and buffers.

Table 13-5 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

13.7.3 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 it 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.

13.7.4 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 13-6 shows correction between modes and FIFO.

Table 13-6 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

13.8 Status Flag

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

<RBFL> 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".

13.9 Error Flag

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

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

Mode	Flag		
	<OERR>	<PERR>	<FERR>
UART mode	Over-run error	Parity error	Framing error
I/O Interface mode (Clock input mode)	Over-run error	Under-run error (When a double buffer and FIFO are used)	Fixed to 0
		Fixed to 0 (When a double buffer and FIFO are not used)	
I/O Interface mode (Clock output mode)	Undefined	Undefined	Fixed to 0

13.9.1 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.

13.9.2 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.

13.9.3 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.

13.10 Receive

13.10.1 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.

13.10.2 Receive Control Unit

13.10.2.1 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>.

13.10.2.2 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.

13.10.3 Receive Operation

13.10.3.1 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<RBFL>) 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.

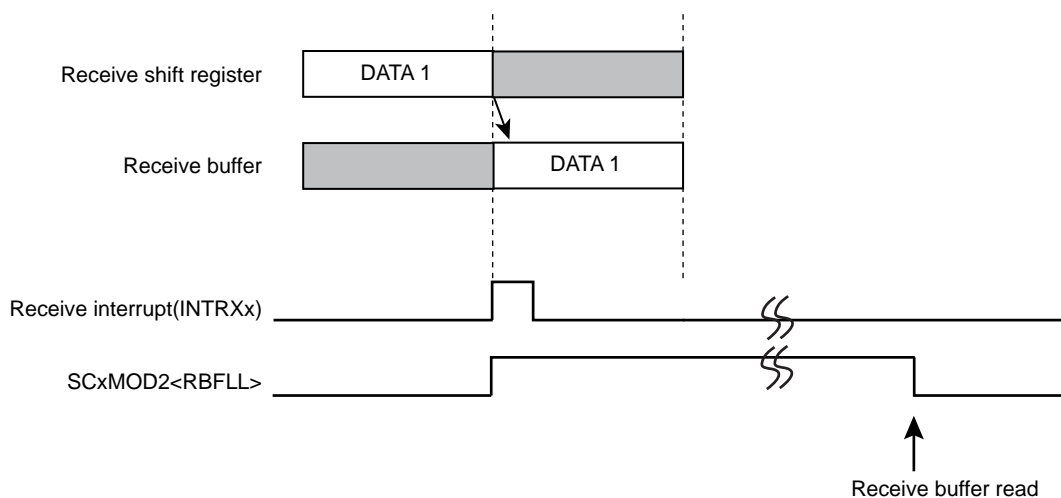


Figure 13-5 Receive Buffer Operation

13.10.3.2 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.

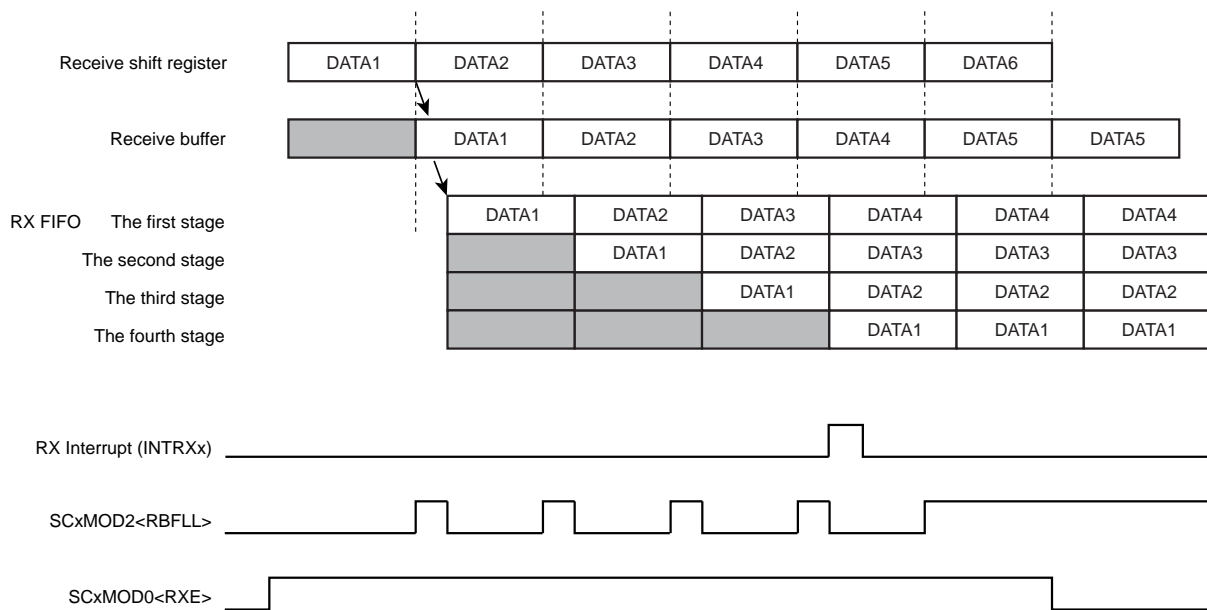


Figure 13-6 Receive FIFO Operation

13.10.3.3 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>.

13.10.3.4 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<RBFL> 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>.

13.10.3.5 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".

13.10.3.6 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.

13.11 Transmit

13.11.1 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.

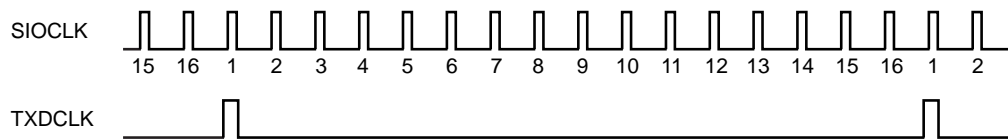


Figure 13-7 Generation of Transmission Clock in UART mode

13.11.2 Transmit Control

13.11.2.1 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 $SCxCR<SCLKS>$.

13.11.2.2 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.

13.11.3 Transmit Operation

13.11.3.1 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".

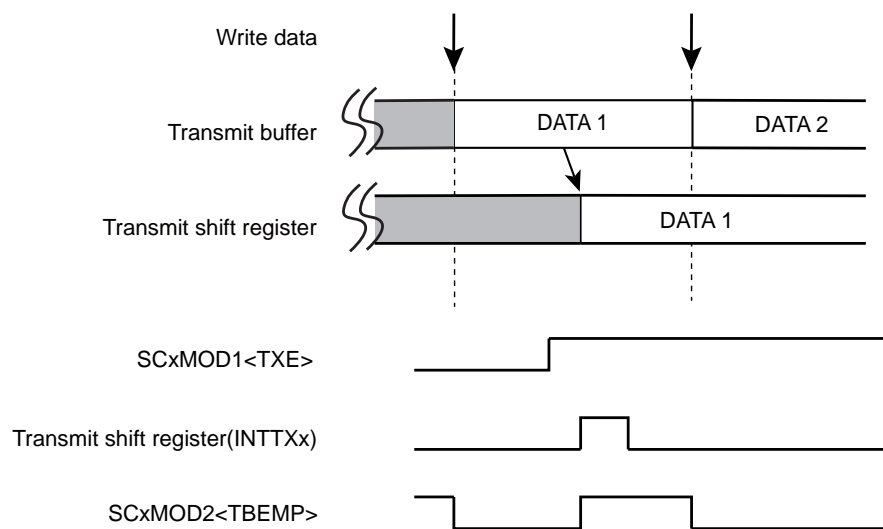


Figure 13-8 Operation of Transmit Buffer (Double-buffer is enabled)

13.11.3.2 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".

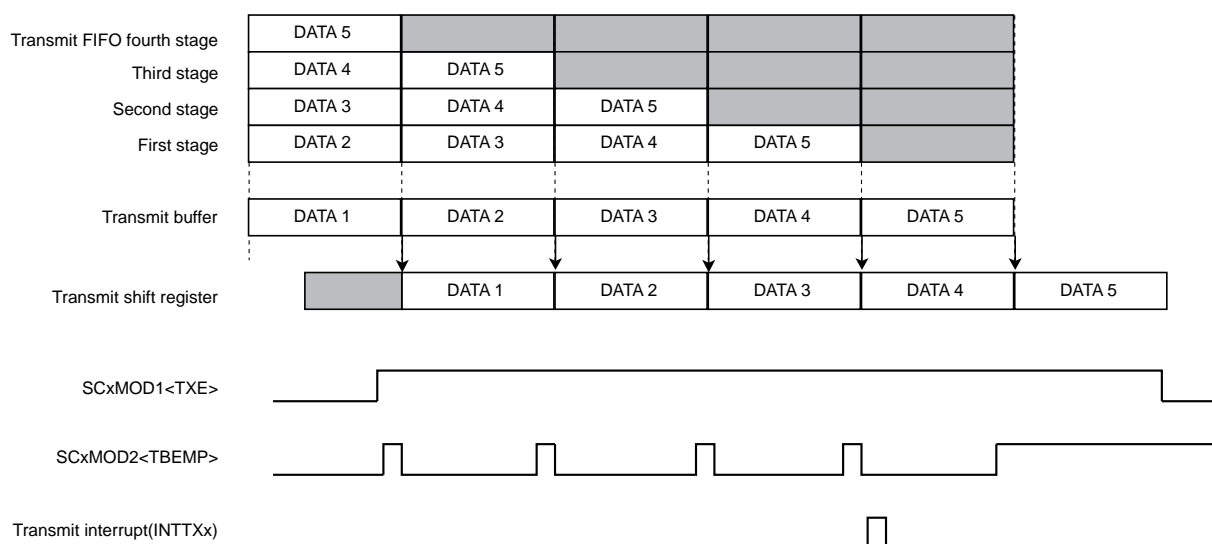
Note: To use Transmit FIFO buffer, Transmit FIFO must be cleared after setting the SIO transfer mode (half duplex/ full duplex) and enabling FIFO (SCxFCNF<CNFG>="1").

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.



13.11.3.3 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 $SC_xFCNF\langle RXTXCNT \rangle$ is configured, $SC_xMOD0\langle TXE \rangle$ bit is cleared at the same time as clock stops and the transmission stops.

13.11.3.4 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 $SC_xCR\langle TIDLE \rangle$.

When $SC_xCR\langle TIDLE \rangle$ is "00", the level of SCxTXD pin is output "Low" level. When $SC_xCR\langle TIDLE \rangle$ is "01", the level of SCxTXD pin is output "High" level. When $SC_xCR\langle TIDLE \rangle$ is "10", the level of SCxTXD pin is output the level of the last bit.

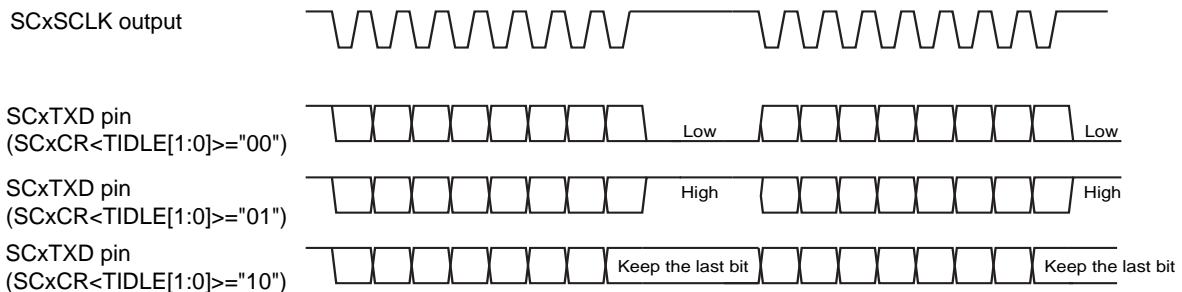


Figure 13-9 Level of SCxTXD pin After the last bit is output

13.11.3.5 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.

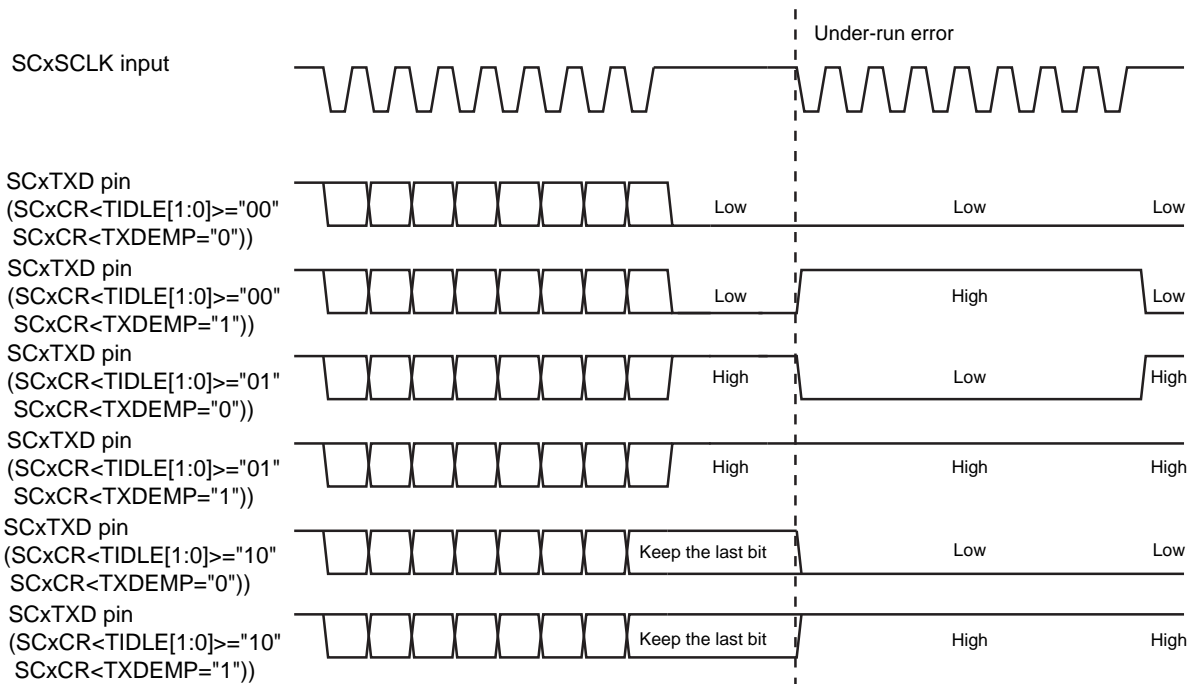


Figure 13-10 Level of SCxTXD pin when Under-run Error is Occurred

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.

13.11.3.6 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.

$$\text{The data hold time of the last bit} \leq \text{The period of SCLK} / 2$$

13.12 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 $\text{SCxMOD0}<\text{CTSE}>$.

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.

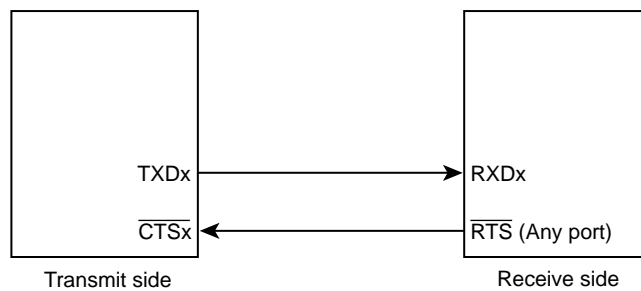


Figure 13-11 Handshake Function

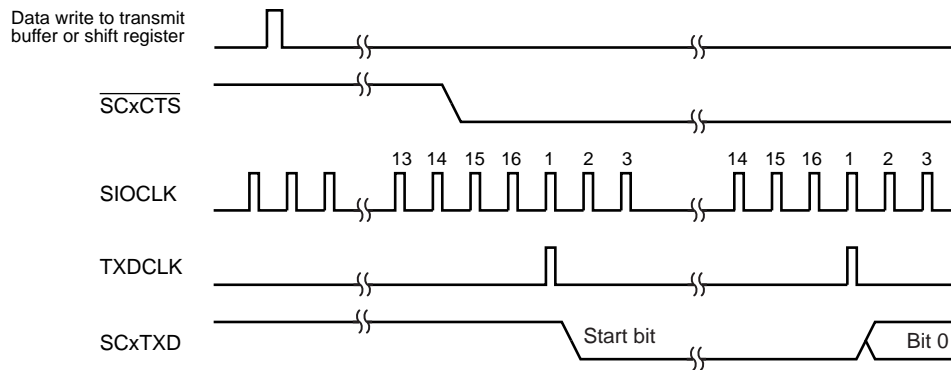


Figure 13-12 $\overline{\text{SCxCTS}}$ Signal timing

13.13 Interrupt/Error Generation Timing

13.13.1 Receive Interrupts

Figure 13-13 shows the data flow of receive operation and the route of read.

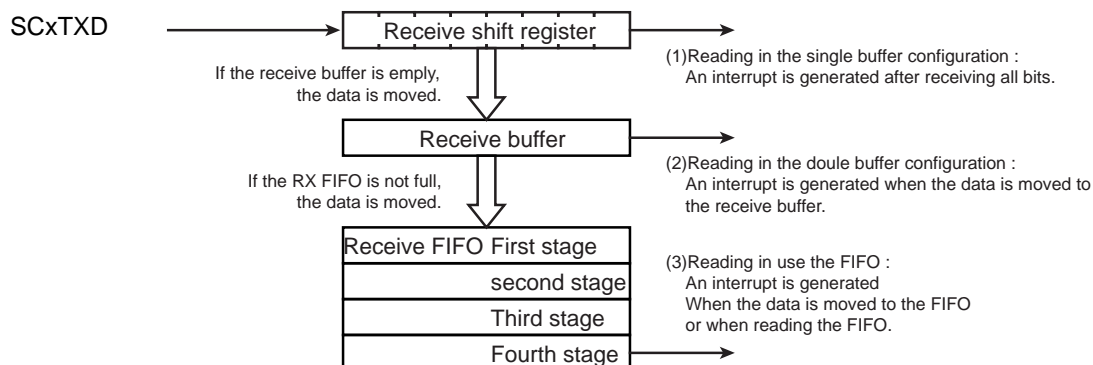


Figure 13-13 Receive Buffer/FIFO Configuration Diagram

13.13.1.1 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 13-7 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 buffer. Specific timings are : <ul style="list-style-type: none"> • 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: <ul style="list-style-type: none"> • If data does not exist in 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.

13.13.1.2 FIFO

When the FIFO is used, a receive interrupt occurs on depending on the timing described in Table 13-8 and the condition specified with SCxRFC<RFIS>.

Table 13-8 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

13.13.2 Transmit interrupts

Figure 13-14 shows the data flow of transmit operation and the route of read.

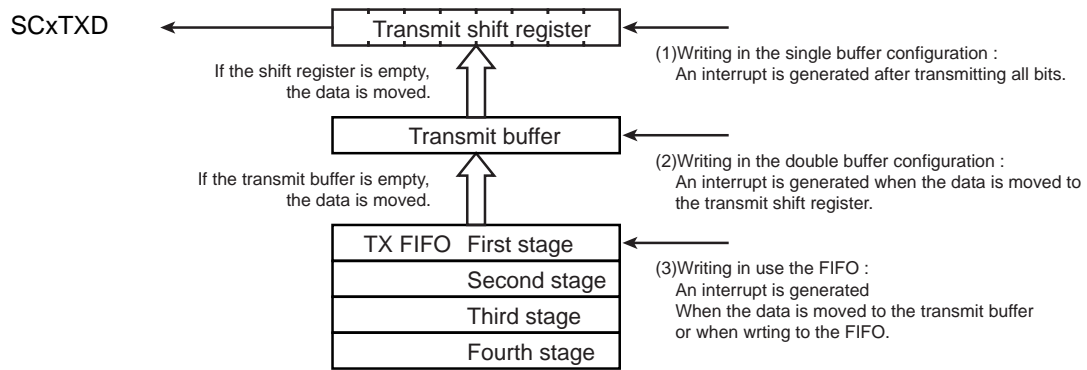


Figure 13-14 Transmit Buffer / FIFO Configuration Diagram

13.13.2.1 Single 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 13-9 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.	

13.13.2.2 FIFO

When the FIFO is used, a transmit interrupt occurs depending on the timing described in Table 13-10 and the condition specified with SCxTFC<TFIS>.

Table 13-10 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

13.13.3 Error Generation

13.13.3.1 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

13.13.3.2 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.

13.14 DMA Request

DMA transfer can be started at the timing of interrupt request.

Please refer to the chapter of "product information" for the channel which can be used for a DMA request with this product.

Note 1: In case using DMA transfer by transmit or receive interrupt request, enabled DMA and set transmit and receive registers after generating software reset by SCxMOD<SWRST>.

Note 2: When the DMA transfer is used, the FIFO cannot be used.

13.15 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><RBFL><TXRUN>, SCxCR <OERR><PERR><FERR> are initialized. And the receive circuit and the transmit circuit become initial state.

Other states are maintained.

13.16 Operation in Each Mode

13.16.1 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.

13.16.1.1 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.

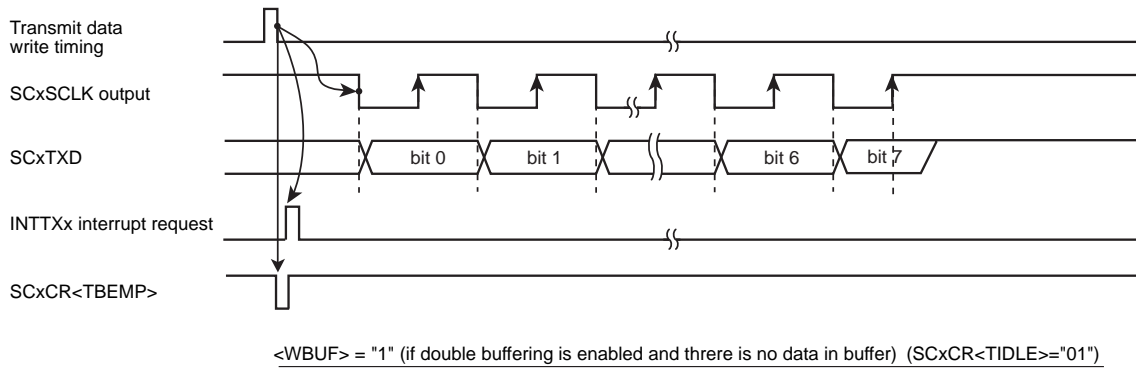
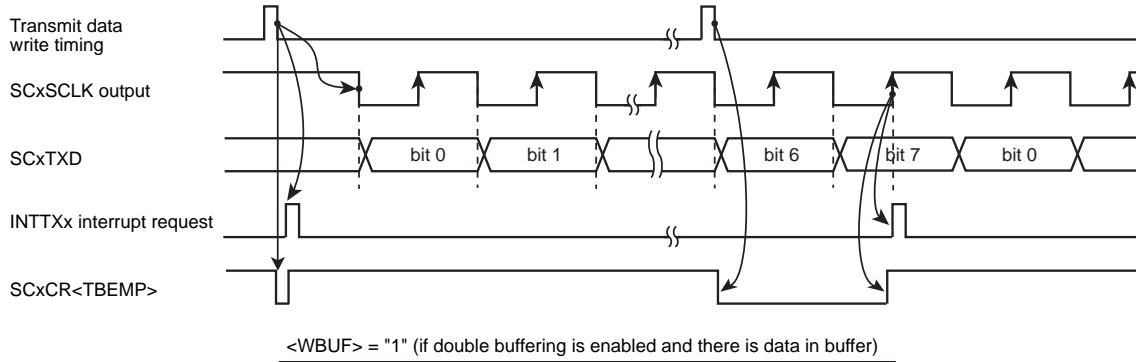
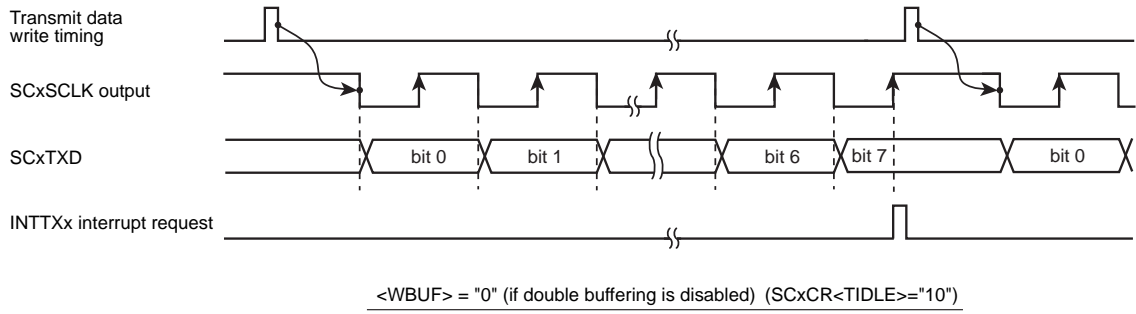


Figure 13-15 Transmit Operation in the I/O Interface Mode (Clock Output Mode)

(2) Clock Input Mode

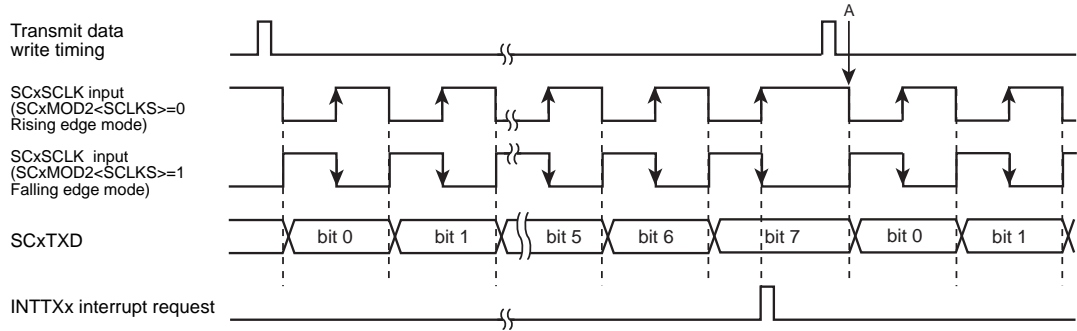
- If double buffering is disabled ($SCxMOD2\langle WBUF \rangle = "0"$)

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 13-16.

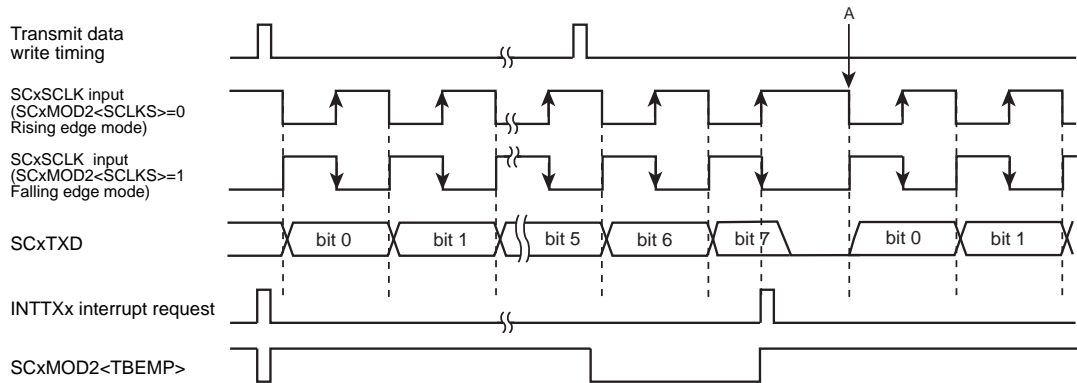
- If double buffer is enabled ($SCxMOD2\langle WBUF \rangle = "1"$)

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, $SCxMOD2\langle TBEMP \rangle$ is set to "1", and the INTTXx interrupt is generated.

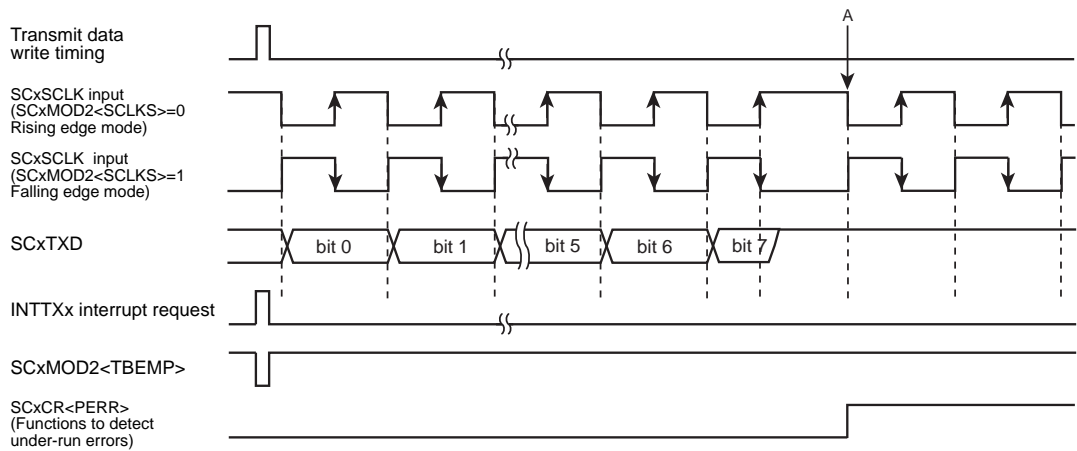
If the clock input becomes active while no data is in the transmit buffer, although the internal bit counter is started, an under-run error occurs and the level which is specified by $SCxCR\langle TXDEMP \rangle$ is output to SCxTXD pin.



<WBUF> = "0" (if double buffering is disabled) (SCxCR<TILDE>="10")



<WBUF> = "1" (if double buffering is enabled and there is data in buffer2) (SCxCR<TILDE>="00")



<WBUF> = "1" (if double buffering is enabled and there is no data in buffer2) (SCxCR<TXDEMP><TILDE>="100")

Figure 13-16 Transmit Operation in the I/O Interface Mode (Clock Input Mode)

13.16.1.2 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<RBFL> 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.

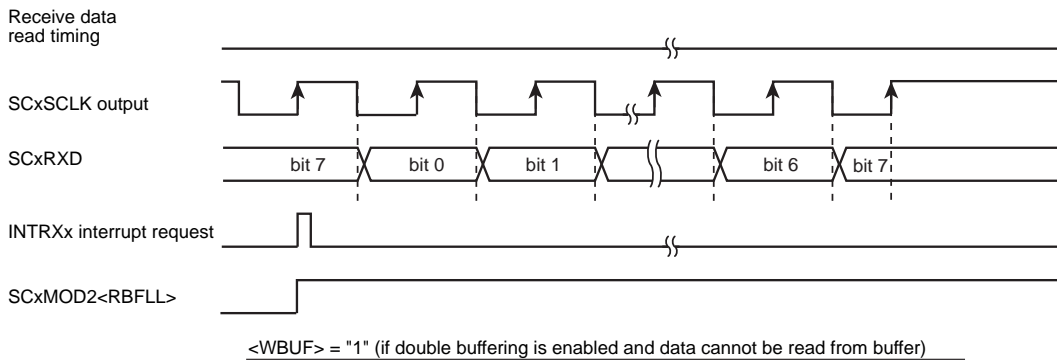
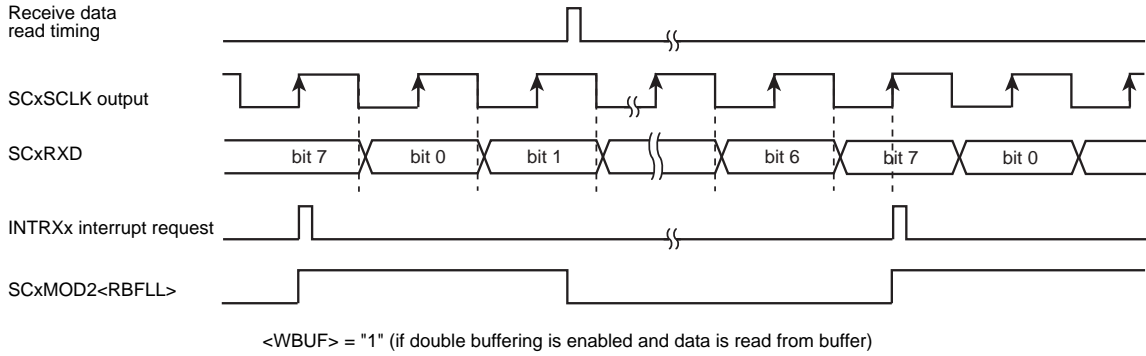
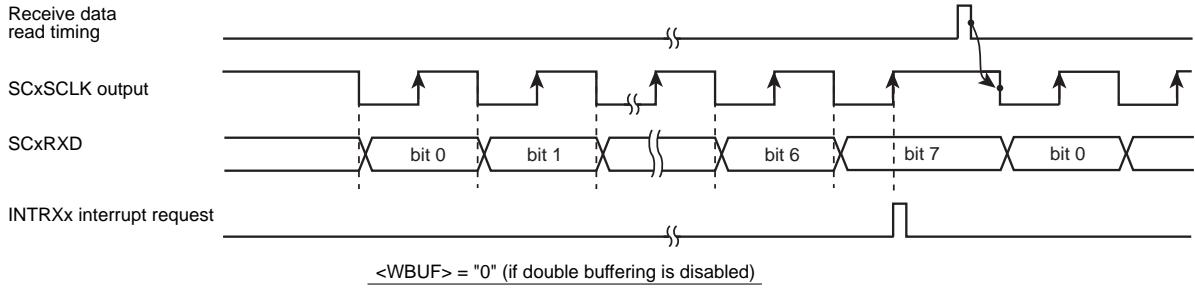


Figure 13-17 Receive Operation in the I/O Interface Mode (Clock Output Mode)

(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.

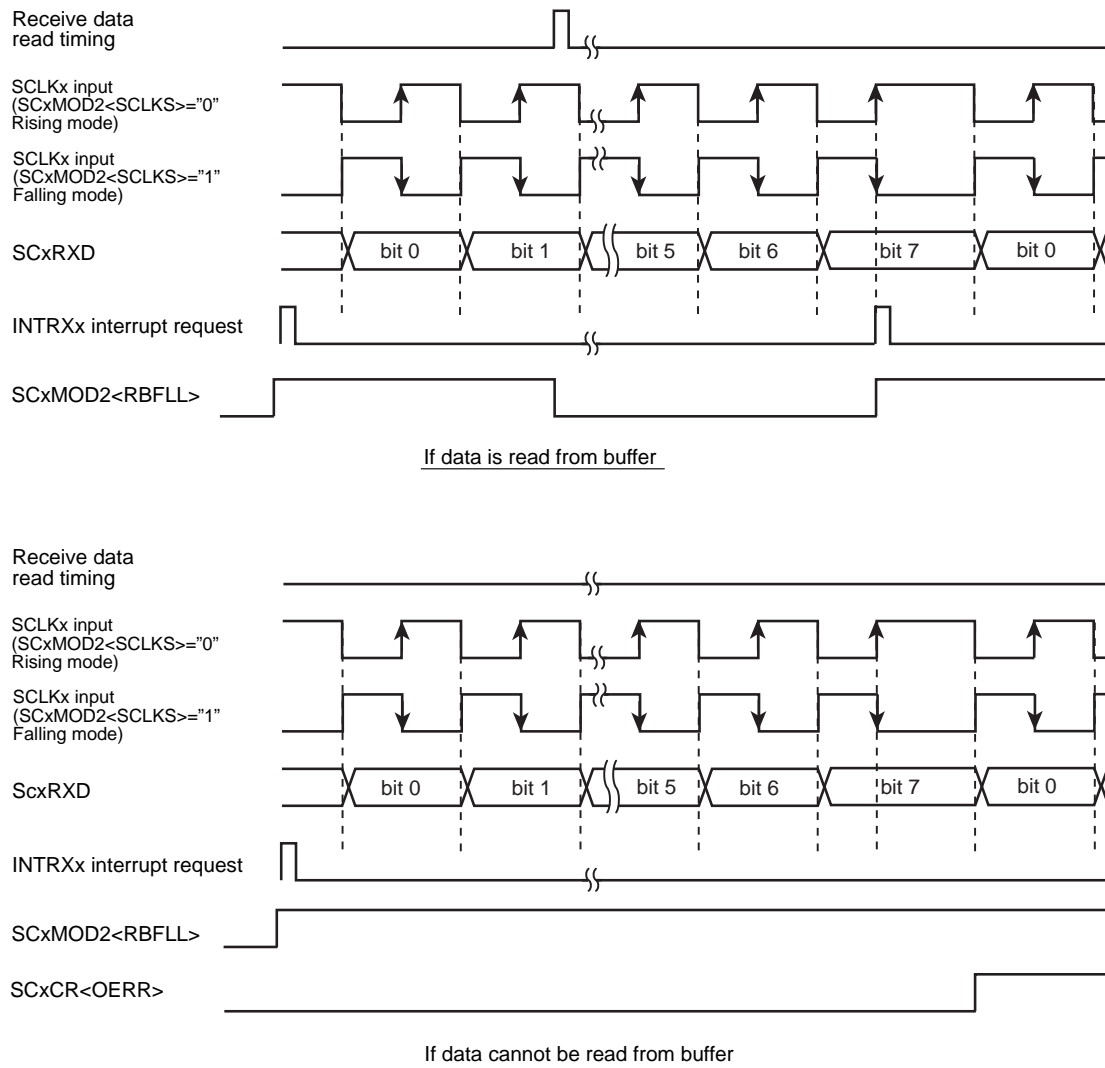


Figure 13-18 Receive Operation in the I/O Interface Mode (Clock Input Mode)

13.16.1.3 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<RBFL> = "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.

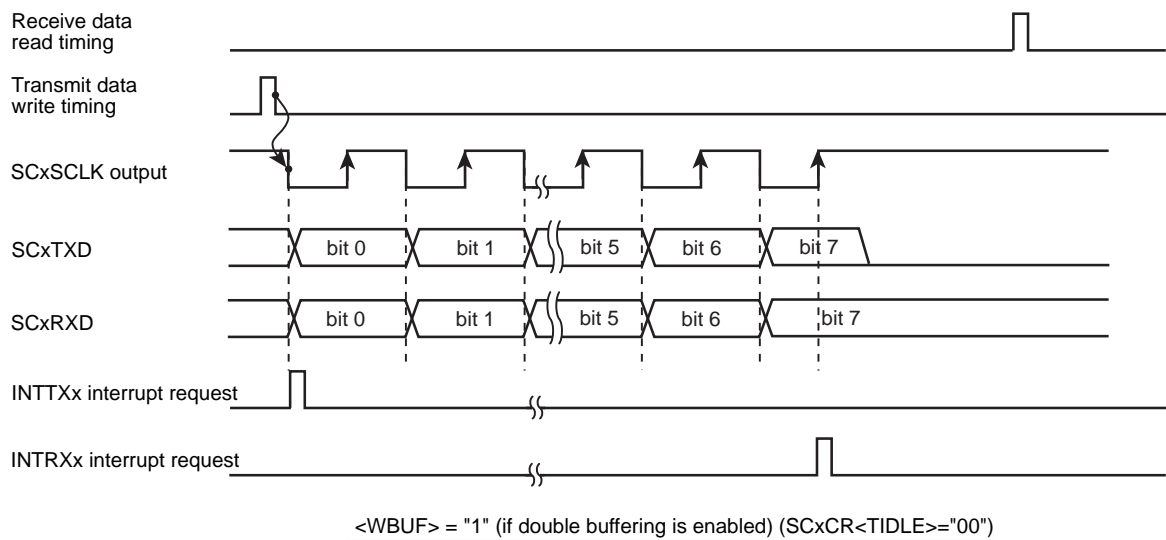
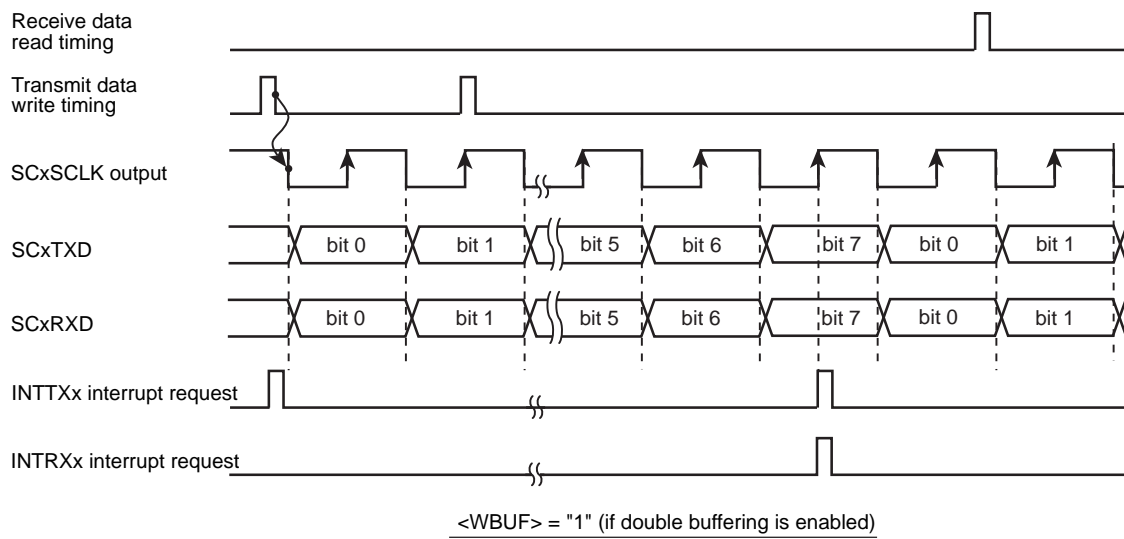
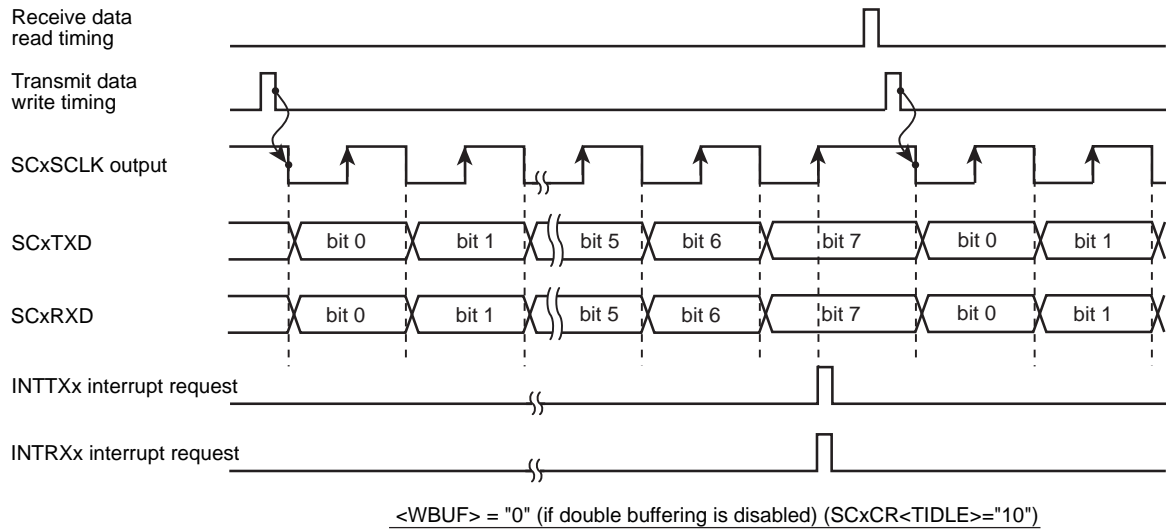


Figure 13-19 Transmit/Receive Operation in the I/O Interface Mode (SCLK Output Mode)

(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 13-20). 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 13-20). 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 under-run error occurs. The level which is specified by SCxCR<TXDEMP> is output to SCxTXD pin.

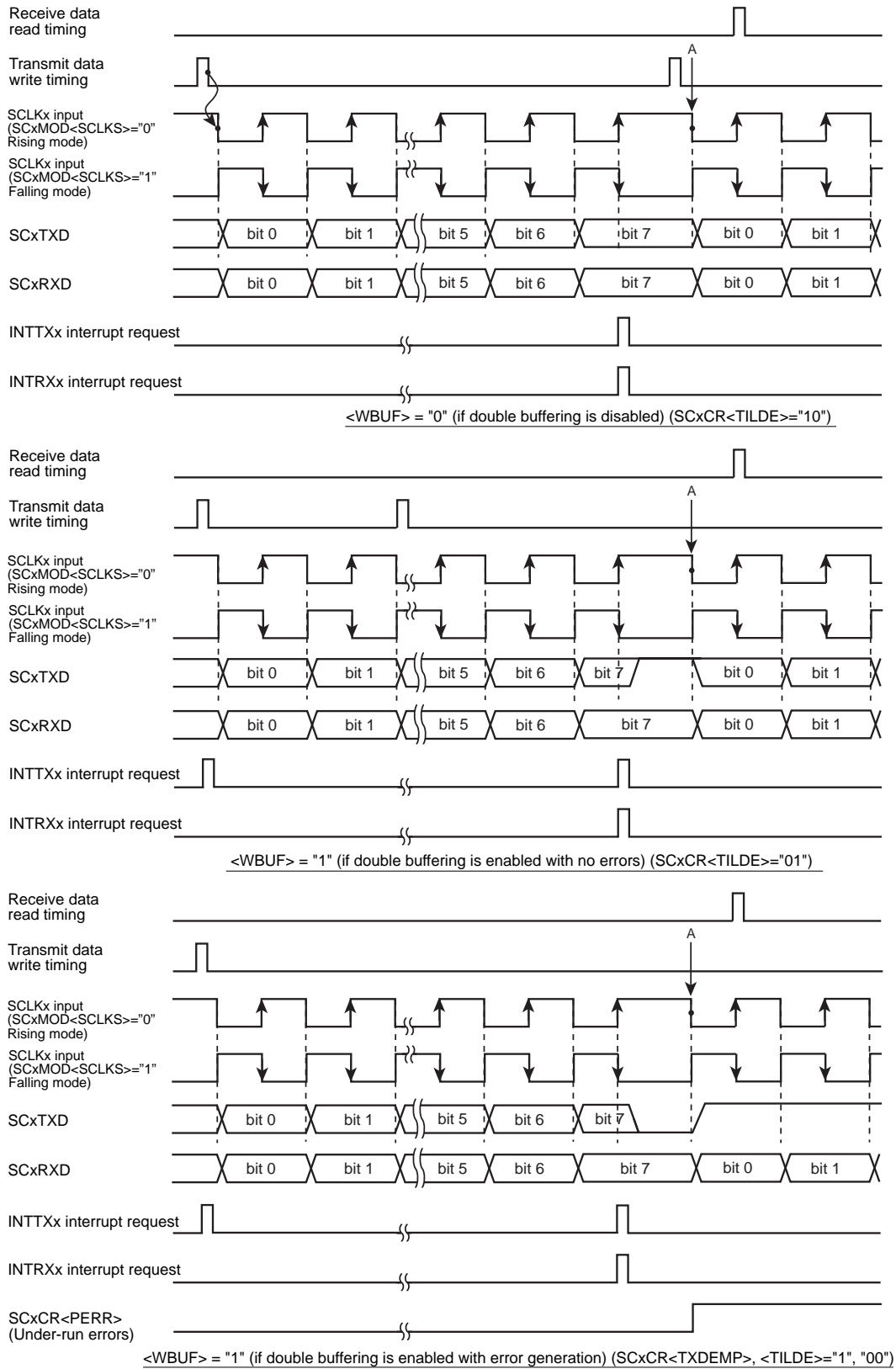


Figure 13-20 Transmit/Receive Operation in the I/O Interface Mode (Clock Input Mode)

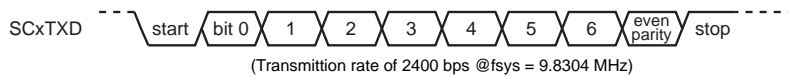
13.16.2 Mode 1 (7-bit UART mode)

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.



Clocking conditions	system clock:	High-speed (fc)
	High-speed clock gear:	x 1 (fc)
	Prescaler clock:	fperiph/2 (fperiph = fsys)

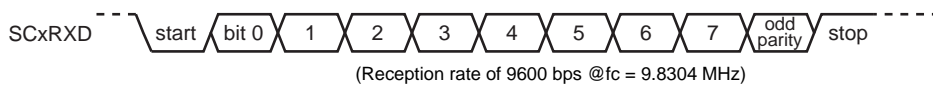
		7	6	5	4	3	2	1	0	
SCxMOD0	←	x	0	-	0	0	1	0	1	Set 7-bit UART mode
SCxCR	←	x	1	1	x	x	x	0	0	Even parity enabled
SCxBRCR	←	0	0	1	0	0	1	0	0	Set 2400bps
SCxBUF	←	*	*	*	*	*	*	*	*	Set transmit data

x: don't care - : no change

13.16.3 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:



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

13.16.4 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>.

13.16.4.1 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.

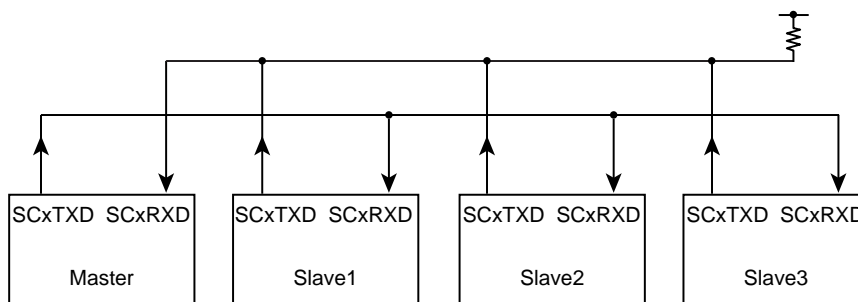


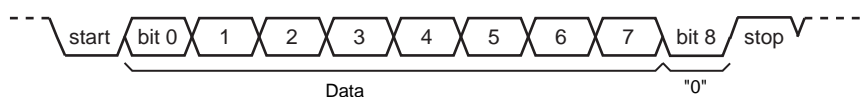
Figure 13-21 Serial Links to Use Wake-up Function

13.16.4.2 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.

14. Synchronous Serial Port (SSP)

14.1 Overview

This LSI contains the SSP (Synchronous Serial Port) with channels. These channels have the following features.

Communication protocol	Three types of synchronous serial ports including the SPI <ul style="list-style-type: none"> • Motorola SPI (SPI) frame format • TI synchronous (SSI) frame format • National Microwire (Microwire) frame format 	
Operation mode	Master/slave mode	
Transmit FIFO	16bits wide / 8 tiers deep	
Receive FIFO	16bits wide / 8 tiers deep	
Transmitted/received data size	4 to 16 bits	
Interrupt type	Transmit interrupt Receive interrupt Receive overrun interrupt Time-out interrupt	
Communication speed (note)	In master mode	$f_{sys} / 2$ to $f_{sys}/65024$
	In slave mode	$f_{sys} / 12$ to $f_{sys}/65024$
DMA	Supported	
Internal test function	The internal loopback test mode is available.	

14.2 Block Diagram

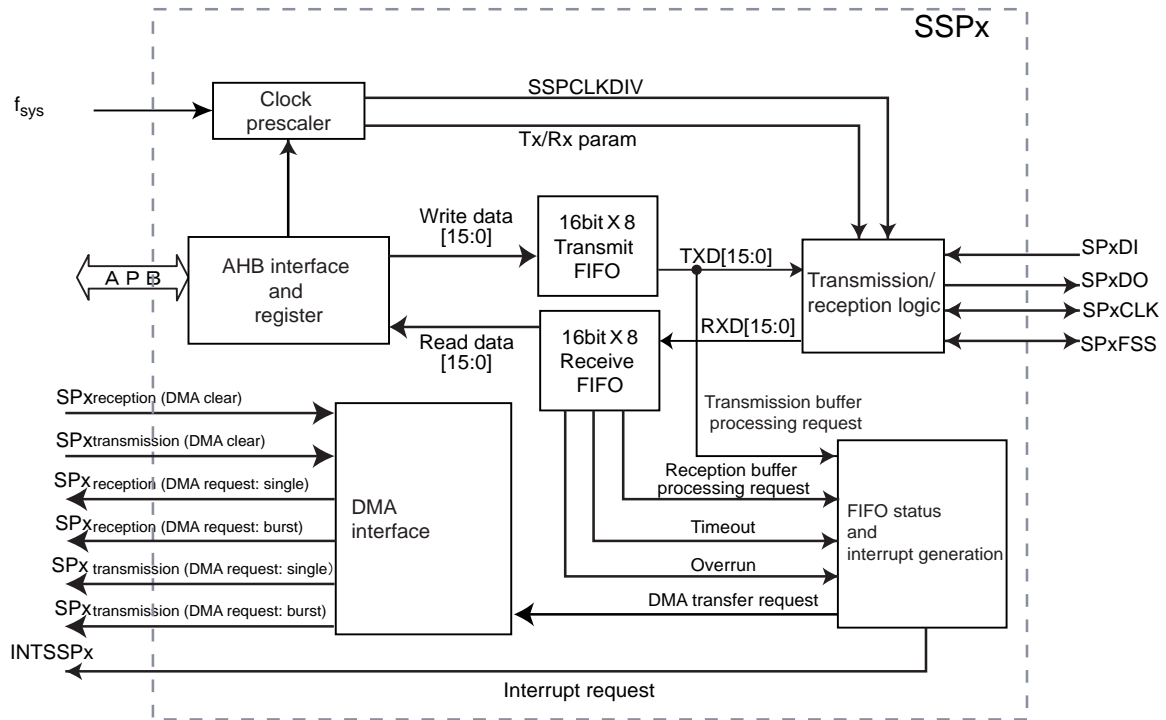


Figure 14-1 SSP block diagram

14.3 Register

14.3.1 Register List

The table shows control registers and their addresses.

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

Register Name		Address (Base+)
Control register 0	SSPxCR0	0x0000
Control register 1	SSPxCR1	0x0004
Receive FIFO (read) and transmit FIFO (write) data register	SSPxDR	0x0008
Status register	SSPxSR	0x000C
Clock prescale register	SSPxCPSR	0x0010
Interrupt enable/disable register	SSPxIMSC	0x0014
Pre-enable interrupt status register	SSPxRIS	0x0018
Post-enable interrupt status register	SSPxMIS	0x001C
Interrupt clear register	SSPxICR	0x0020
DMA control register	SSPxDMACR	0x0024

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

14.3.2 SSPxCR0(Control register 0)

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

Bit	Bit Symbol	Type	Function																																
31-16	-	W	Write as "0".																																
15-8	SCR[7:0]	R/W	<p>For serial clock rate setting. Parameter : 0x00 to 0xFF.</p> <p>Bits to generate the SSP transmit bit rate and receive bit rate. This bit rate can be obtained by the following equation. Bit rate = $f_{sys} / (<CPSDVSR> \times (1 + <SCR>))$ <CPSDVSR> is an even number between 2 to 254, which is programmed by the SSPxCPSR register, and <SCR> takes a value between 0 to 255.</p>																																
7	SPH	R/W	<p>SPxCLK phase: 0 : Captures data at the 1st clock edge. 1 : Captures data at the 2nd clock edge. This is applicable to Motorola SPI frame format only. Refer to Section "Motorola SPI frame format"</p>																																
6	SPO	R/W	<p>SPxCLK polarity: 0:SPxCLK is in Low state. 1:SPxCLK is in High state. This is applicable to Motorola SPI frame format only. Refer to Section "Motorola SPI frame format"</p>																																
5-4	FRF[1:0]	R/W	<p>Frame format: 00: SPI frame format 01: SSI serial frame format 10: Microwire frame format 11: Reserved, undefined operation</p>																																
3-0	DSS[3:0]	R/W	<p>Data size select:</p> <table border="1"> <tr> <td>0000:</td> <td>Reserved, undefined operation</td> <td>1000:</td> <td>9 bits data</td> </tr> <tr> <td>0001:</td> <td>Reserved, undefined operation</td> <td>1001:</td> <td>10 bits data</td> </tr> <tr> <td>0010:</td> <td>Reserved, undefined operation</td> <td>1010:</td> <td>11 bits data</td> </tr> <tr> <td>0011:</td> <td>4 bits data</td> <td>1011:</td> <td>12 bits data</td> </tr> <tr> <td>0100:</td> <td>5 bits data</td> <td>1100:</td> <td>13 bits data</td> </tr> <tr> <td>0101:</td> <td>6 bits data</td> <td>1101:</td> <td>14 bits data</td> </tr> <tr> <td>0110:</td> <td>7 bits data</td> <td>1110:</td> <td>15 bits data</td> </tr> <tr> <td>0111:</td> <td>8 bits data</td> <td>1111:</td> <td>16 bits data</td> </tr> </table>	0000:	Reserved, undefined operation	1000:	9 bits data	0001:	Reserved, undefined operation	1001:	10 bits data	0010:	Reserved, undefined operation	1010:	11 bits data	0011:	4 bits data	1011:	12 bits data	0100:	5 bits data	1100:	13 bits data	0101:	6 bits data	1101:	14 bits data	0110:	7 bits data	1110:	15 bits data	0111:	8 bits data	1111:	16 bits data
0000:	Reserved, undefined operation	1000:	9 bits data																																
0001:	Reserved, undefined operation	1001:	10 bits data																																
0010:	Reserved, undefined operation	1010:	11 bits data																																
0011:	4 bits data	1011:	12 bits data																																
0100:	5 bits data	1100:	13 bits data																																
0101:	6 bits data	1101:	14 bits data																																
0110:	7 bits data	1110:	15 bits data																																
0111:	8 bits data	1111:	16 bits data																																

14.3.3 SSPxCR1(Control register1)

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

Bit	Bit Symbol	Type	Function
31-4	-	W	Write as "0".
3	SOD	R/W	Slave mode SPxDO output control: 0: Enable 1: Disable Slave mode output disable. This bit is relevant only in the slave mode (<MS>="1").
2	MS	R/W	Master/slave mode select: (Note) 0: Device configured as a master. 1: Device configured as a slave.
1	SSE	R/W	SSP enable/disable 0: Disable 1: Enable
0	LBM	R/W	Loop back mode 0: Normal serial port operation enabled. 1: Output of transmit serial shifter is connected to input of receive serial shifter internally.

Note: This bit is for switching between master and slave. Be sure to configure in the following steps in slave mode and in transmission.

- 1) Set to slave mode :<MS>=1
- 2) Set transmit data in FIFO :<DATA>=0x****
- 3) Set SSP to Enable. :<SSE>=1

14.3.4 SSPxDR(Data register)

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

Bit	Bit Symbol	Type	Function
31-16	-	W	Write as "0".
15-0	DATA[15:0]	R/W	Transmit/receive FIFO data: 0x0000 to 0xFFFF Read: Receive FIFO Write: Transmit FIFO If the data size used in the program is less than 16bits, write the data to fit LSB.The transmit control circuit ignores unused bits of MSB side. The receive control circuit receives the data to fit LSB automatically.

14.3.5 SSPxSR(Status register)

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

Bit	Bit Symbol	Type	Function
31-5	-	W	Write as "0".
4	BSY	R	Busy flag: 0: Idle 1: Busy <BSY>="1" indicates that the SSP is currently transmitting and/or receiving a frame or the transmit FIFO is not empty.
3	RFF	R	Receive FIFO full flag: 0: Receive FIFO is not full. 1: Receive FIFO is full.
2	RNE	R	Receive FIFO empty flag: 0: Receive FIFO is empty. 1: Receive FIFO is not empty.
1	TNF	R	Transmit FIFO full flag: 0: Transmit FIFO is full. 1: Transmit FIFO is not full.
0	TFE	R	Transmit FIFO empty flag: 0: Transmit FIFO is not empty. 1: Transmit FIFO is empty.

14.3.6 SSPxCPSR (Clock prescale register)

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

Bit	Bit Symbol	Type	Function
31-8	-	W	Write as "0".
7-0	CPSDVSR[7:0]	R/W	<p>Clock prescale divisor: Set an even number from 2 to 254.</p> <p>Clock prescale divisor: Must be an even number from 2 to 254, depending on the frequency of fsys. The least significant bit always returns zero when read.</p>

14.3.7 SSPxIMSC (Interrupt enable/disable register)

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

Bit	Bit Symbol	Type	Function
31-4	-	W	Write as "0".
3	TXIM	R/W	Transmit FIFO interrupt enable: 0: Disable 1: Enable Enable or disable a conditional interrupt to occur if the transmit FIFO is half empty or less.
2	RXIM	R/W	Receive FIFO interrupt enable: 0: Disable 1: Enable Enable or disable a conditional interrupt to occur if the receive FIFO is half full or less.
1	RTIM	R/W	Receive time-out interrupt enable: 0: Disable 1: Enable Enable or disable a conditional interrupt to indicate that data exists in the receive FIFO to the time-out period and data is not read.
0	RORIM	R/W	Receive overrun interrupt enable: 0: Disable 1: Enable Enable or disable a conditional interrupt to indicate that data was written when the receive FIFO was in the full condition.

14.3.8 SSPxRIS (Pre-enable interrupt status register)

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

Bit	Bit Symbol	Type	Function
31-4	-	W	Write as "0".
3	TXRIS	R	Pre-enable transmit interrupt flag: 0: Interrupt not present 1: Interrupt present
2	RXRIS	R	Pre-enable receive interrupt flag: 0: Interrupt not present 1: Interrupt present
1	RTRIS	R	Pre-enable timeout interrupt flag: 0: Interrupt not present 1: Interrupt present
0	RORRIS	R	Pre-enable overrun interrupt flag: 0: Interrupt not present 1: Interrupt present

14.3.9 SSPxMIS (Post-enable interrupt status register)

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

Bit	Bit Symbol	Type	Function
31-4	-	W	Write as "0".
3	TXMIS	R	Post-enable transmit interrupt flag: 0: Interrupt not present 1: Interrupt present
2	RXMIS	R	Post-enable receive interrupt flag: 0: Interrupt not present 1: Interrupt present
1	RTMIS	R	Post-enable time-out interrupt flag: 0: Interrupt not present 1: Interrupt present
0	RORMIS	R	Post-enable overrun interrupt flag: 0: Interrupt not present 1: Interrupt present

14.3.10 SSPxICR (Interrupt clear register)

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

Bit	Bit Symbol	Type	Function
31-2	-	W	Write as "0".
1	RTIC	W	Clear the time-out interrupt flag: 0: Invalid 1: Clear
0	RORIC	W	Clear the overrun interrupt flag: 0: Invalid 1: Clear

14.3.11 SSPxDMA CR (DMA control register)

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

Bit	Bit Symbol	Type	Function
31-2	-	W	Write as "0".
1	TXDMAE	R/W	Transmit FIFO DMA control: 0:Disable 1:Enable
0	RXDMAE	R/W	Transmit FIFO DMA control: 0:Disable 1:Enable

14.4 Overview of SSP

The SSP is an interface that enables serial communications with the peripheral devices with three types of synchronous serial interface functions.

The SSP performs serial-parallel conversion of the data received from a peripheral device.

The transmit buffers data in the independent 16-bit wide and 8-layered transmit FIFO in the transmit mode, and the receive buffers data in the 16-bit wide and 8-layered receive FIFO in receive mode. Serial data is transmitted via SPxDO and received via SPxDI.

The SSP contains a programmable prescaler to generate the serial output clock SPxCLK from the input clock f_{sys}. The operation mode, frame format, and data size of the SSP are programmed in the control registers SSPxCR0 and SSPxCR1.

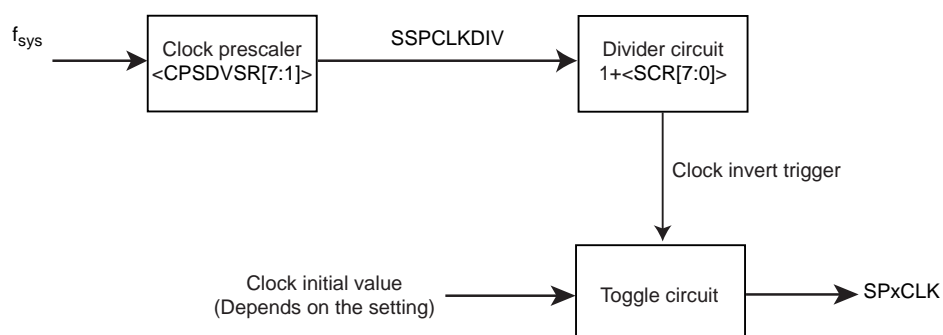
14.4.1 Clock prescaler

When configured as a master, a clock prescaler comprising two free-running serially linked counters is used to provide the serial output clock SPxCLK.

You can program the clock prescaler through the SSPxCPSR register, to divide f_{sys} by a factor of 2 to 254 in steps of two. Because the least significant bit of the SSPxCPSR register is not used, division by an odd number is not possible.

The output of the prescaler is further divided by a factor of 1 to 256, which is obtained by adding 1 to the value programmed in the SSPxCR0 register, to give the master output clock SPxCLK.

$$\text{Bitrate} = f_{\text{sys}} / (<\text{CPSDVSR}> \times (1 + <\text{SCR}>))$$



14.4.2 Transmit FIFO

This is a 16-bit wide, 8-layered transmit FIFO buffer, which is shared in master and slave modes.

14.4.3 Receive FIFO

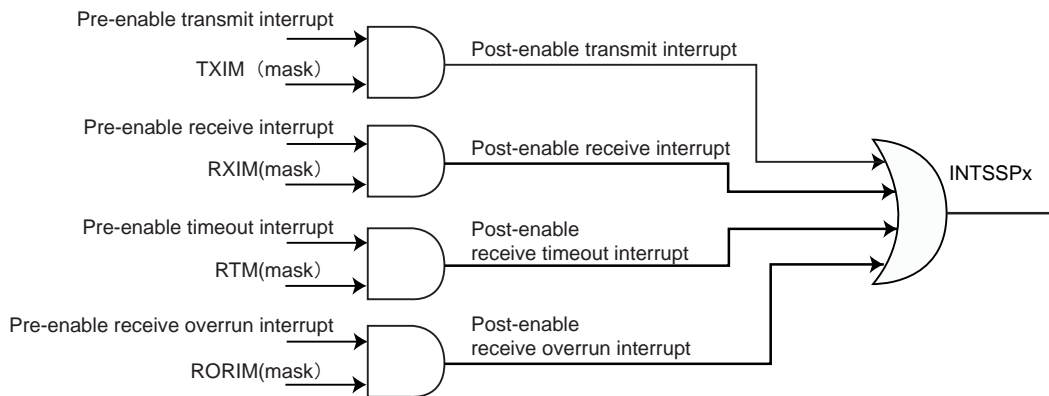
This is a 16-bit wide 8-layered receive FIFO buffer, which is shared in master and slave modes.

14.4.4 Interrupt generation logic

The interrupts, each of which can be masked separately, are generated.

Transmit interrupt	A conditional interrupt to occur when the transmit FIFO has free space more than (including half) of the entire capacity. (Number of valid data items in the transmit FIFO ≤ 4)
Receive interrupt	A conditional interrupt to occur when the receive FIFO has valid data more than half (including half) the entire capacity. (Number of valid data items in the receive FIFO ≥ 4)
Time-out interrupt	A conditional interrupt to indicate that the data exists in the receive FIFO to the time-out period.
Overrun interrupt	Conditional interrupts indicating that data is written to receive FIFO when it is full.

Also, The individual masked sources are combined into a single interrupt. When any of the above interrupts is asserted, the combined interrupt INTSSPx is asserted.



a. Transmit interrupt

The transmit interrupt is asserted when there are four or fewer valid entries in the transmit FIFO. The transmit interrupt is also generated when the SSP operation is disabled ($SSPxCR1 \langle SSE \rangle = "0"$).

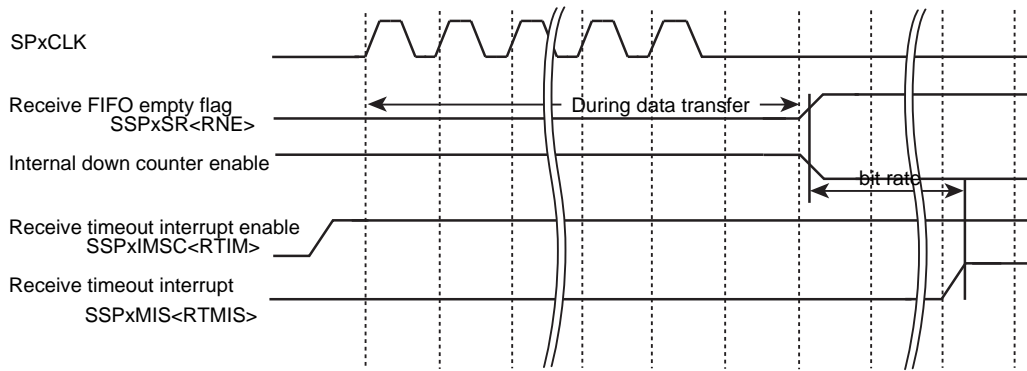
The first transmitted data can be written in the FIFO by using this interrupt.

b. Receive interrupt

The receive interrupt is asserted when there are four or more valid entries in the receive FIFO.

c. Time-out interrupt

The time-out interrupt is asserted when the receive FIFO is not empty and the SSP has remained idle for a fixed 32-bit period (bit rate). This mechanism ensures that the user is aware that data is still present in the receive FIFO and requires servicing. This operation occurs in both master and slave modes. When the time-out interrupt is generated, read all data from the receive FIFO. Even if all the data is not read, data can be transmitted / received if the receive FIFO has a free space and the number of data to be transmitted does not exceed the free space of the receive FIFO. When transfer starts, the timeout interrupt will be cleared. If data is transmitted / received when the receive FIFO has no free space, the time-out interrupt will not be cleared and an overrun interrupt will be generated.



d. Overrun interrupt

When the next data (9th data item) is received when the receive FIFO is already full, an overrun interrupt is generated immediately after transfer. The data received after the overrun interrupt is generated (including the 9th data item) will become invalid and be discarded. However, if data is read from the receive FIFO while the 9th data item is being received (before the interrupt is generated), the 9th received data will be written in the receive FIFO as valid data. To perform transfer properly when the overrun interrupt has been generated, write "1" to SSPxICR<RORIC> register, and then read all data from the receive FIFO. Even if all the data is not read, data can be transmitted / received if the receive FIFO has free space and the number of data to be transmitted does not exceed the free space of the receive FIFO. Note that if the receive FIFO is not read (provided that the receive FIFO is not empty) within a certain 32-bit period (bit rate) after the overrun interrupt is cleared, a time-out interrupt will be generated.

14.4.5 DMA Interface

SSP supports DMA burst and single transfers. When DMA transfer is enabled by SSPxDMACR, both burst and single transfers are enabled. All transfer requests are cleared if SSP operation or DMA transfers is disabled.

14.4.5.1 Burst Transfer

When data is increased over the watermark level (half of the FIFO) in the receive FIFO, a receive burst DMA transfer request is asserted.

When data is reduced to less than the watermark level (half of the FIFO) in the transmit FIFO, a transmit burst DMA transfer request is asserted.

Set the DMA burst length to 4 words.

The following table shows the trigger points for DMABREQ, for both transmit and receive FIFOs.

Watermark level	Burst length	
	Transmit (Number of empty locations)	Receive (Number of filled locations)
1/2	4	4

14.4.5.2 Single Transfer

If at least one data is stored in the receive FIFO, a receive single DMA transfer request is asserted.

If at least one empty space exists in the transmit FIFO, a transmit single DMA transfer request is asserted.

Both burst and single transfers can be used simultaneously. In the case of reception, when data is increased over the watermark level, both burst and single transfer requests are asserted. When data is reduced to less than the watermark level, only single transfer request is asserted. In the case of transmission, when data is reduced to less than the watermark level, both burst and single transfer requests are asserted. When data is increased over the watermark level, only single transfer request is asserted.

For example, when 19-word is received, 4-word burst transfer is performed 4 times. After this transfer, DMA asserts a transfer completion signal and the burst transfer is finished. A single transfer request is asserted for left 3 words. Through three times single transfers all data can be transferred.

14.5 SSP operation

14.5.1 Initial setting for SSP

Settings for the SSP communication protocol must be made with the SSP disabled.

Control registers SSPxCR0 and SSPxCR1 need to configure this SSP as a master or slave operating under one of the following protocols. In addition, make the settings related to the communication speed in the clock prescale registers SSPxCPSR and SSPxCR0 <SCR>.

This SSP supports the following protocols:

- SPI
- SSI
- Microwire

14.5.2 Enabling SSP

The transfer operation starts when the operation is enabled with the transmitted data written in the transmit FIFO, or when transmitted data is written in the transmit FIFO with the operation enabled.

However, if the transmit FIFO contains only four or fewer entries when the operation is enabled, a transmit interrupt will be generated. This interrupt can be used to write the initial data.

Note: When the SSP is in the SPI slave mode and the SPxFSS pin is not used, be sure to transmit data of one byte or more in the FIFO before enabling the operation. If the operation is enabled with the transmit FIFO empty, the transfer data will not be output correctly.

14.5.3 Clock ratios

When setting a frequency for f_{sys} , the following conditions must be met.

If there are further conditions with a product, refer to the "product information" chapter.

- In master mode
 - $f_{SPxCLK} \text{ (maximum)} \rightarrow f_{sys} / 2$
 - $f_{SPxCLK} \text{ (minimum)} \rightarrow f_{sys} / (254 \times 256)$
- In slave mode
 - $f_{SPxCLK} \text{ (maximum)} \rightarrow f_{sys} / 12$
 - $f_{SPxCLK} \text{ (minimum)} \rightarrow f_{sys} / (254 \times 256)$

14.6 Frame Format

Each frame format is between 4 and 16 bits wide depending on the size of data programmed, and is transmitted starting from the MSB.

- Serial clock (SPxCLK)

Signals remain "Low" in the SSI and Microwire formats and as inactive in the SPI format while the SSP is in the idle state. In addition, data is output at the set bit rate only during data transmission.

- Serial frame (SPxFSS)

In the SPI and Microwire frame formats, signals are set to "Low" active and always asserted to "Low" during frame transmission.

In the SSI frame format, signals are asserted only during 1 bit rate before each frame transmission. In this frame format, output data is transmitted at the rising edge of SPxCLK and the input data is received at its falling edge.

Refer to Section "14.6.1" to "14.6.3" for details of each frame format.

14.6.1 SSI frame format

In this mode, the SSP is in idle state, SPxCLK and SPxFSS are forcedly set to "Low", and the transmit data line SPxDO becomes Hi-Z. When data is written in the transmit FIFO, the master outputs "High" pulses of 1 SPxCLK to the SPxFSS line. The transmitted data will be transferred from the transmit FIFO to the transmit serial shift register. Data of 4 to 16 bits will be output from the SPxDO pin at the next rising edge of SPxCLK.

Likewise, the received data will be input starting from the MSB to the SPxDI pin at the falling edge of SPxCLK. The received data will be transferred from the serial shift register into the receive FIFO at the rising edge of SPxCLK after its LSB data is latched.

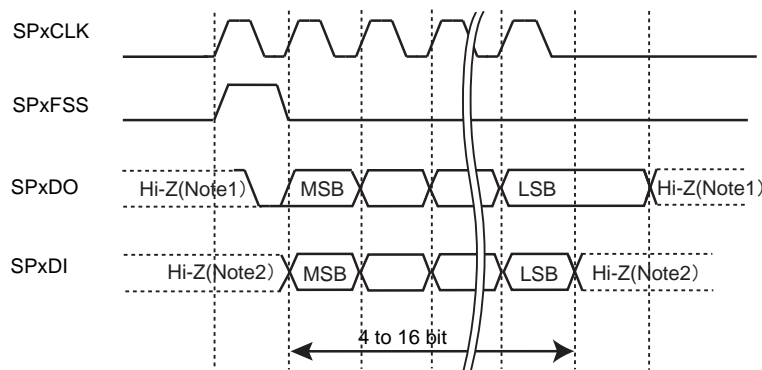


Figure 14-2 SSI frame format (transmission/reception during single transfer)

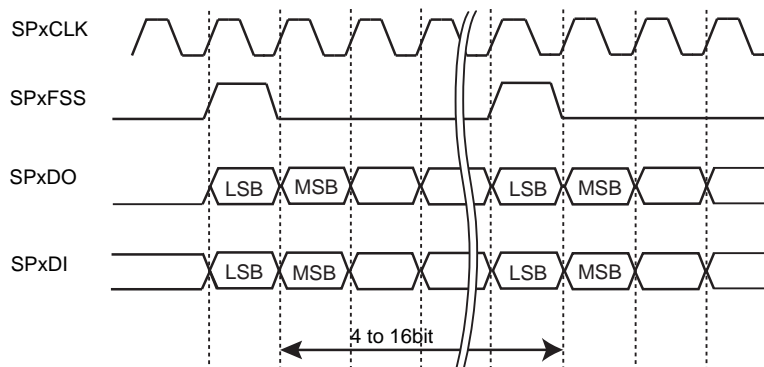


Figure 14-3 SSI frame format (transmission/reception during continuous transfer)

Note 1: When transmission is disable, SPxDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note 2: SPxDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

14.6.2 SPI frame format

The SPI interface has 4 lines. SPx \overline{FSS} is used for slave selection. One of the main features of the SPI format is that the <SPO> and <SPH> bits in the SSPxCR0 register can be used to set the SPxCLK operation timing.

SSPxCR0 <SPO> is used to set the level at which SPxCLK in idle state is held.

SSPxCR0 <SPH> is used to select the clock edge at which data is latched.

	SSPxCR0<SPO>	SSPxCR0<SPH>
0	"Low" state	Capture data at the 1st clock edge.
1	"High" state	Capture data at the 2nd clock edge.

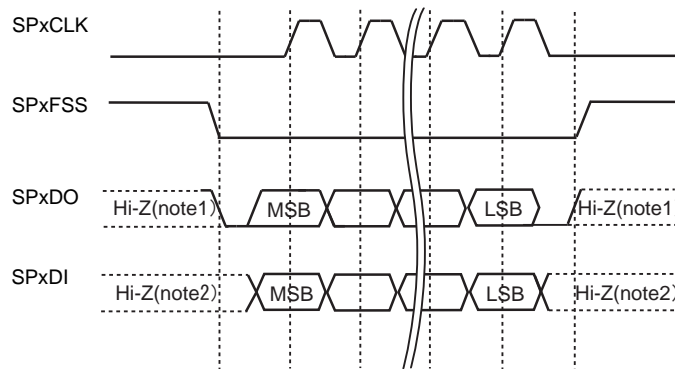


Figure 14-4 SPI frame format (single transfer, <SPO>="0" & <SPH>="0")

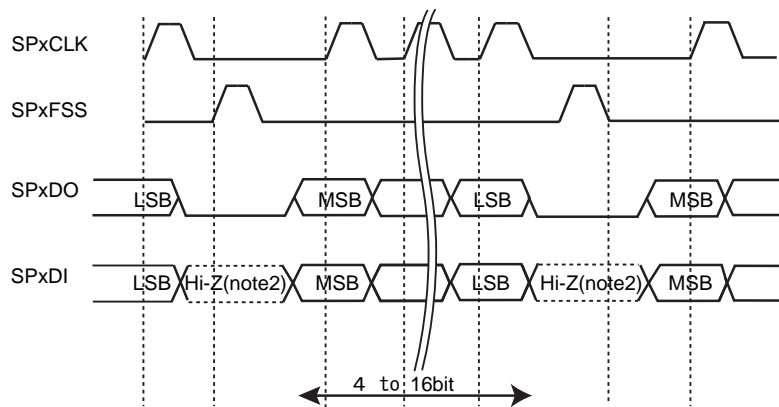


Figure 14-5 SPI frame format (continuous transfer, <SPO>="0" & <SPH>="0")

Note 1: When transmission is disable, SPxDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note 2: SPxDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

With this setting $\langle SPO \rangle = "0"$, during the idle period:

- The SPxCLK signal is set to "Low".
- SPxFSS is set to "High".
- The transmit data line SPxDO is set to "Low".

If the SSP is enabled and valid data exists in the transmit FIFO, the SPxFSS master signal driven by "Low" notifies of the start of transmission. This enables the slave data in the SPxDI input line of the master.

When a half of the SPxCLK period has passed, valid master data is transferred to the SPxDO pin. Both the master data and slave data are now set. When another half of SPxCLK has passed, the SPxCLK master clock pin becomes "High". After that, the data is captured at the rising edge of the SPxCLK signal and transmitted at its falling edge.

In the single transfer, the SPxFSS line will return to the idle "High" state when all the bits of that data word have been transferred, and then one cycle of SPxCLK has passed after the last bit was captured.

However, for continuous transfer, the SPxFSS signal must be pulsed at HIGH between individual data word transfers. This is because change is not enabled when the slave selection pin freezes data in its peripheral register and the $\langle SPH \rangle$ bit is logical 0.

Therefore, to enable writing of serial peripheral data, the master device must drive the SPxFSS pin of the slave device between individual data transfers. When the continuous transfer is completed, the SPxFSS pin will return to the idle state when one cycle of SPxCLK has passed after the last bit is captured.

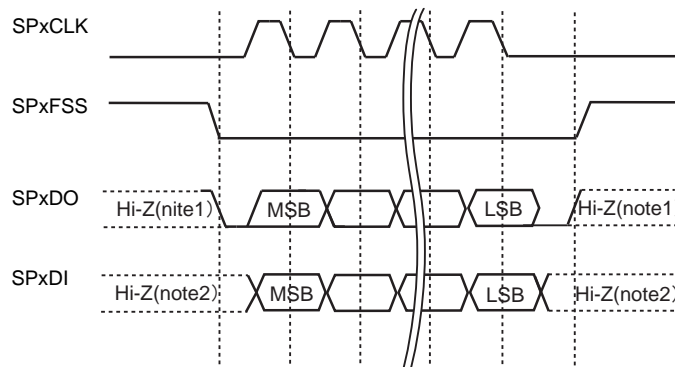


Figure 14-6 SPI frame format (Single & continuous transfer, $\langle SPO \rangle = "0"$ & $\langle SPH \rangle = "1"$)

Figure 14-6 show the SPI frame format with $\langle SPO \rangle = 0$ & $\langle SPH \rangle = 1$, which is covers both single and continuous transfer.

- Note 1: When transmission is disable, SPxDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.
- Note 2: SPxDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

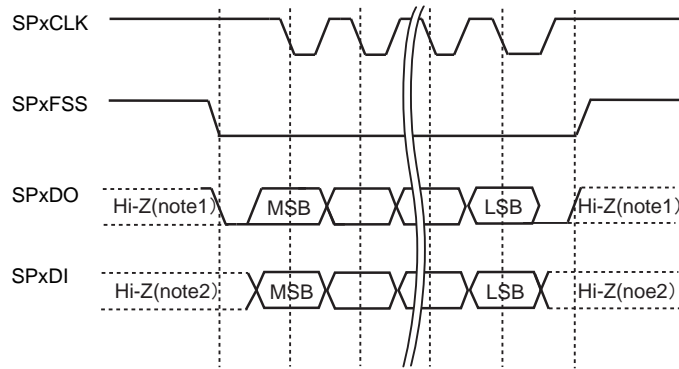


Figure 14-7 SPI frame format (single transfer, $\langle SPO \rangle = "1"$ & $\langle SPH \rangle = "0"$)

Figure 14-7 shows the SPI frame format with $\langle SPO \rangle = 1$ & $\langle SPH \rangle = 0$, which is for single transmission signal sequence.

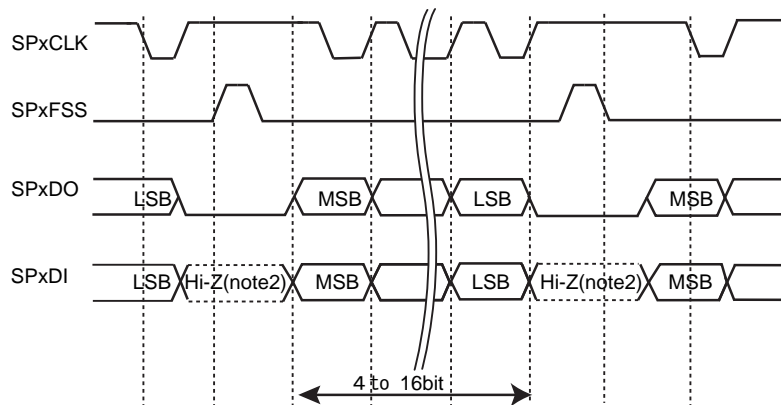


Figure 14-8 SPI frame format (continuous transfer, $\langle SPO \rangle = "1"$ & $\langle SPH \rangle = "0"$)

Figure 14-8 shows the SPI frame format with $\langle SPO \rangle = 1$ & $\langle SPH \rangle = 0$, which is for continuous transmission signal sequence.

Note 1: When transmission is disable, SPxDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note 2: SPxDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

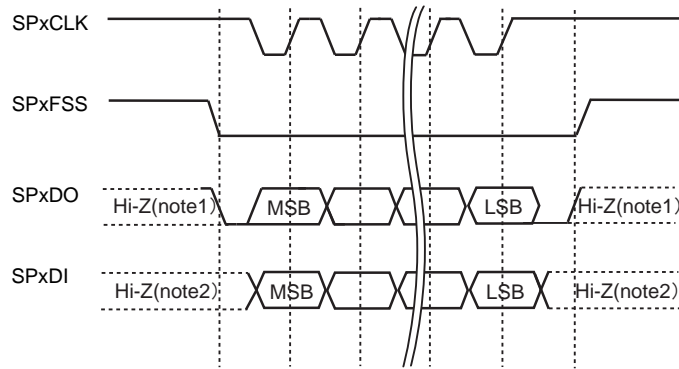


Figure 14-9 SPI frame format (Single & continuous transfer, <SPO>="1" & <SPH>="1")

Figure 14-9 show the SPI frame format with <SPO>=1 & <SPH>=1, which is covers both single and continuous transfer.

- Note 1: When transmission is disable, SPxDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.
- Note 2: SPxDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

14.6.3 Microwire frame format

The Microwire format uses a special master/slave messaging method, which operates in half-duplex mode. In this mode, when a frame begins, an 8-bit control message is transmitted to the slave. During this transmission, no incoming data is received by the SSP. After the message has been transmitted, the slave decodes it, and after waiting one serial clock after the last bit of the 8-bit control message has been sent, it responds with the requested data. The returned data can be 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits.

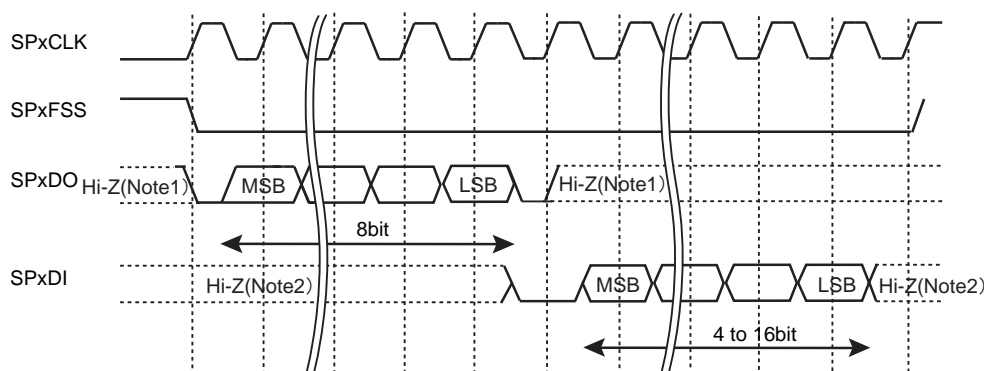


Figure 14-10 Microwire frame format (single transfer)

Note 1: When transmission is disabled, SPxDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to fix the voltage level.

Note 2: SPxDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to fix the voltage level.

Though the Microwire format is similar to the SPI format, it uses the master/slave message transmission method for half-duplex communications. Each serial transmission is started by an 8-bit control word, which is sent to the off-chip slave device. During this transmission, the SSP does not receive input data. After the message has been transmitted, the off-chip slave decodes it, and after waiting one serial clock after the last bit of the 8-bit control message has been sent, responds with the requested data. The returned data can be 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits. With this configuration, during the idle period:

- The SPxCLK signal is set to "Low".
- SPxFSS is set to "High".
- The transmit data line SPxDO is set to "Low".

A transmission is triggered by writing a control byte to the transmit FIFO. The falling edge of SPxFSS causes the value stored in the bottom entry of the transmit FIFO to be transferred to the serial shift register for the transmit logic, and the MSB of the 8-bit control frame to be shifted out onto the SPxDO pin.

SPxFSS remains "Low" and the SPxDI pin remains tristate during this transmission. The off-chip serial slave device latches each control bit into its serial shifter on the rising edge of each SPxCLK.

After the last bit is latched by the slave device, the control byte is decoded during a one clock wait-state, and the slave responds by transmitting data back to the SSP. Each bit is driven onto SPxDI line on the falling edge of SPxCLK.

The SSP in turn latches each bit on the rising edge of SPxCLK. At the end of the frame, for single transfers, the SPxFSS signal is pulled "High" one clock period after the last bit has been latched in the receive serial shifter, which causes the data to be transferred to the receive FIFO.

Note: The off-chip slave device can tristate the receive line either on the falling edge of SPxCLK after the LSB has been latched by the receive shifter, or when the SPxFSS pin goes "High".

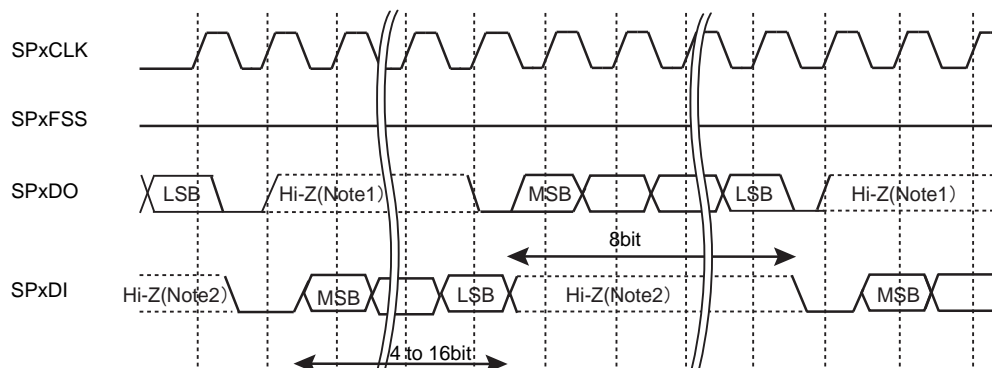


Figure 14-11 Microwire frame format (continuous transfer)

Note 1: When transmission is disabled, SPxDO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to fix the voltage level.

Note 2: SPxDI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to fix the voltage level.

For continuous transfers, data transmission begins and ends in the same manner as a single transfer. However, the SPxFSS line is continuously asserted (held Low) and transmission of data occurs back to back.

The control byte of the next frame follows directly after the LSB of the received data from the current frame. Each of the received values is transferred from the receive shifter on the falling edge of SPxCLK, after the LSB of the frame has been latched into the SSP.

Note:[Example of connection] The SSP does not support dynamic switching between the master and slave in the system. Each sample SSP is configured and connected as either a master or slave.

15. 24-bit $\Delta\Sigma$ Analog/Digital Converter (DSADC)

The reference voltage circuit (BGR) used in the DSADC is shared with a temperature sensor. The BGR needs to set the control register (TEMPEN) of the temperature sensor.

15.1 Features

DSADC has the following features:

- Conversion start
 - Conversion can be started by software.
 - Conversion can be started by hardware triggers.
- Conversion modes
 - Single conversion
 - Repeat conversion
- Status flags
 - Conversion result store flag
 - Overrun flag
 - Conversion end flag
 - Conversion flag
- A conversion clock can be divided by ratios below:
 - fc/1, fc/2, fc/4 and fc/8
- Conversion end interrupt output
- Conversion start correct function
- Synchronous start function for multiple units
- Conversion completion signal output

15.1.1 Pin Treatment

The reference voltage of the DSADC can be supplied from the following methods: the voltage supplied from the built-in reference voltage circuit or the voltage supplied from VREFINx.

Each supply method needs the pin treatment below:

- Using the voltage generated by the built-in reference voltage circuit
 - Do not apply the reference voltage to VREFINx.
 - Set AGNDREFx to the same level as DVSS.
 - Place a 1 μ F capacitor between VREFINx and AGNDREFx.
- Using the voltage from VREFINx
 - Apply the reference voltage to VREFINx.
 - Set AGNDREFx to the same level as DVSS.
 - Place a 1 μ F capacitor between VREFINx and AGNDREFx.

Note: When the voltage generated by the built-in reference voltage circuit is used, BGR and AMP must be enabled.

When, however, the voltage generated from VREFINx is used as the reference voltage, do not enable AMP.

BGR is shared with the temperature sensor; therefore, the control of BGR and AMP is specified with TMPEN <EN0><EN1> of the temperature sensor register.

When DSADC is not used, below settings are required.

- Set AGNDREFx to the same level as DVSS.

When a temperature sensor is also not used, a reference voltage circuit requires below settings.

- Connect DSRVDD3 and SRVDD to DVDD3.
- Connect DSRVSS to DVSS.

15.2 Block Diagram

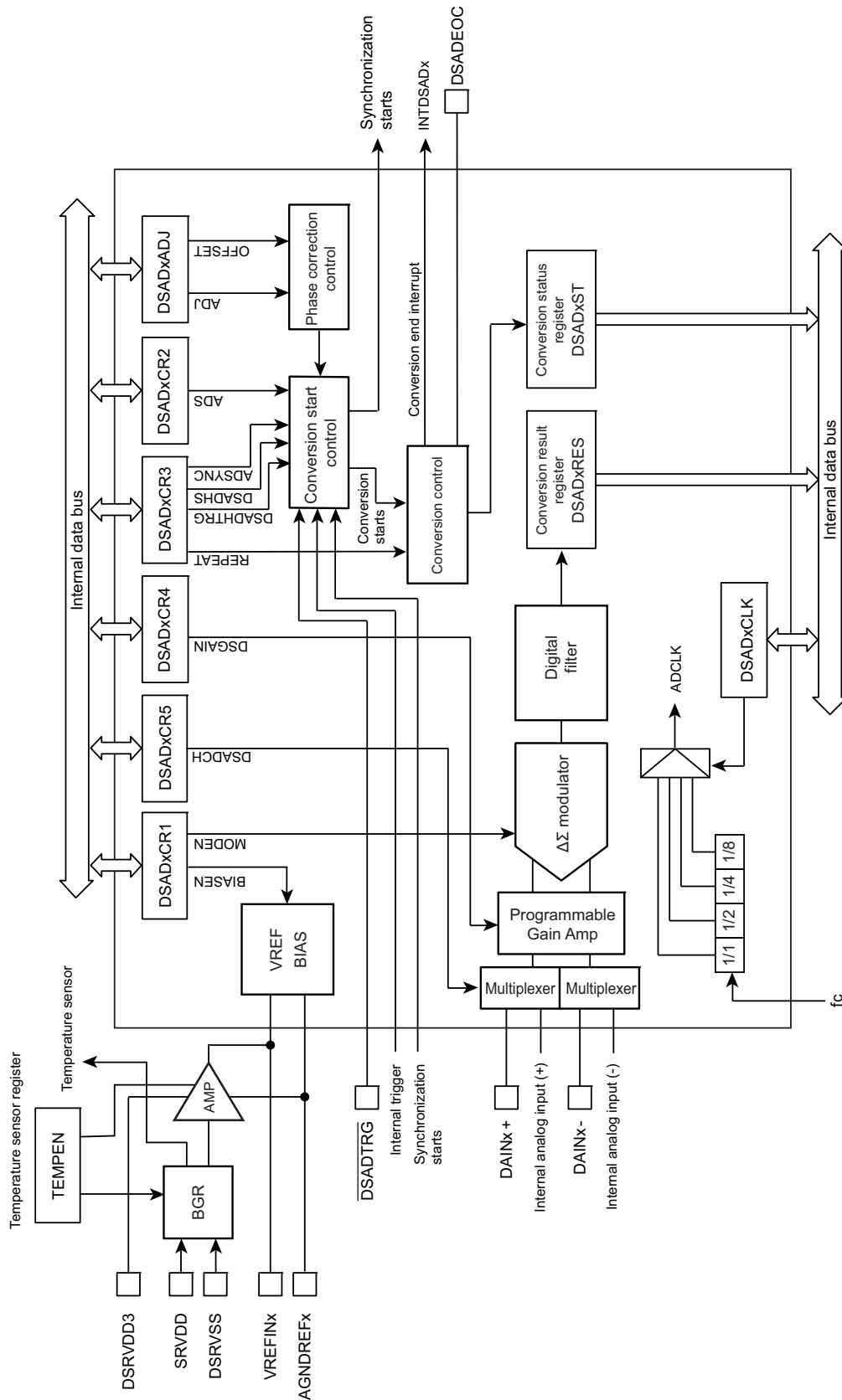


Figure 15-1 Block diagram of 24-bit $\Delta\Sigma$ AD converter

15.3 Registers

15.3.1 Register List

The table below shows control registers and their addresses.

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

Peripheral function : DSAD

Register name		Address(Base+)
Clock Setting Register	DSADxCLK	0x0000
Control Register 0	DSADxCR0	0x0004
Control Register 1	DSADxCR1	0x0008
Control Register 2	DSADxCR2	0x000C
Control Register 3	DSADxCR3	0x0010
Control Register 4	DSADxCR4	0x0014
Control Register 5	DSADxCR5	0x0018
Correction Register	DSADxADJ	0x0030
Conversion Status Register	DSADxST	0x0040
Conversion Result Stored Register	DSADxRES	0x0044

Notes on access to the registers

Only the following registers can be accessed during the conversion:

- DSADxCR0<ADRST[1:0]>
- DSADxCR3<REPEAT>
- DSADxCR5<DSADCH>

15.3.2 Details of Registers

15.3.2.1 DSADxCLK (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	-	-	-	-	-	ADCLK		
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-3	-	R	Read as "0".
2-0	ADCLK[2:0]	R/W	Selects the AD conversion clock. 000: fc/1 001: fc/2 010: fc/4 011: fc/8 100-111: Prohibited

Note 1: The <ADCLK[2:0]> register is modified under the circumstances where DSADxCR1<BIASEN> and <MODEN> are "0" and the AD conversion stops.

Note 2: When the synchronous start function is used, select the same conversion clock for all units.

15.3.2.2 DSADxCR0 (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	-	-	-	-	-	-	ADRST	
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-2	-	-	Read as "0".
1-0	ADRST	W	Software reset (Note) To generate a software reset, write "10", and then write "01". The internal circuits and registers except DSADxCLK are initialized.

Note: Valid only when DSADxCR1<BIASEN>="1".

15.3.2.3 DSADxCR1 (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	-	-	-	-	-	-	BIASEN	MODEN
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	-	-	Read as "0".
1	BIASEN	R/W	Bias control 0: Stop 1: Operation
0	MODEN	R/W	Modulator control 0: Stop 1: Control

15.3.2.4 DSADxCR2 (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	-	-	-	-	-	-	-	ADS
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-1	-	-	Read as "0".
0	ADS	W	Starts conversion. 1: Starts conversion. Set "1" to start conversion. Writing "0" has no meaning. Read as "0".

15.3.2.5 DSADxCR3 (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	-	-	DSADHS	DSADHTG	-	-	-	ADSYNC
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
Bit symbol	-	-	-	-	-	-	-	REPEAT
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-14	-	-	Read as "0".
13	DSADHS	R/W	Hardware startup factor (Note 1) 0: External trigger 1: Internal trigger
12	DSADHTG	R/W	Hardware startup 0: Disabled 1: Enabled
11-9	-	R	Read as "0".
8	ADSYNC	R/W	Synchronous mode 0: Asynchronous operation 1: Synchronous operation Set "1" to start conversion in multiple units simultaneously. (Note 2)
7-1	-	R	Read as "0".
0	REPEAT	R/W	Conversion mode 0: Single conversion 1: Repeat conversion

Note 1: For the hardware startup factors of this product, refer to the chapter on "Product Information."

Note 2: Set "1" to only the unit used as a slave. Set "0" to the unit used as the master. For assigning to the master or slave, refer to the chapter on "Product Information."

15.3.2.6 DSADxCR4 (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	-	-	-	-	-	DSGAIN		
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-3	-	-	Read as "0".
2-0	DSGAIN[2:0]	R/W	Sets the gain amplifier. 000: ×1 001: ×2 010: ×4 011: ×8 100: ×16 101-111: Reserved

15.3.2.7 DSADxCR5 (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	-	-	-	-	-	-	-	DSADCH
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-1	-	-	Read as "0".
0	DSADCH	R/W	Sets the analog inputs. 0: DAIN x (+/-) 1: Internal analog input (+/-) Specifies the analog input signals.

Note: For the internal analog inputs of this product, refer to the chapter on "Product Information."

15.3.2.8 DSADxADJ (Correction Register)

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

Bit	Bit symbol	Type	Function
31-16	OFFSET	R/W	Conversion start correction time Set the time period from setting "1" to DSADxCR2<ADS> to starting conversion. The delay time of <OFFSET> \times 1/fsys is added.
15-1	-	R	Read as "0".
0	ADJ	R/W	Corrects the start of conversion. 0: No correction 1: Correction If this bit is "1", conversion is started after the delay time (set with <OFFSET>) has elapsed from start of conversion.

15.3.2.9 DSADxST (Conversion 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	-	-	-	-	-	-	EOCF	ADBF
After reset	0	0	0	0	0	0	0	0

Bit	Bit symbol	Type	Function
31-2	-	R	Read as "0".
1	EOCF	R	Conversion end flag (note) 0: In conversion. 1: Conversion is complete.
0	ADBF	R	Conversion flag 0: Not converting. 1: In conversion.

Note: This bit is cleared by reading the DSADxST register.

15.3.2.10 DSADxRES (Conversion Result Stored Register)

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

Bit	Bit symbol	Type	Function
31-8	ADR[23:0]	R	Conversion result Conversion result is stored in two's complement format. If this bit is read during conversion, the previous conversion result is read.
7-2	-	R	Read as "0".
1	ADOVR	R	Overrun flag (note) 0: No generation 1: Generated If conversion result is overwritten before reading <ADR>, "1" is set.
0	ADRF	R	Conversion result store flag (note) 0: No result is stored. 1: A result is stored. If a conversion result is stored in <ADR>, "1" is set.

Note: This bit is cleared by reading the DSADxRES register.

15.4 Operation Description

15.4.1 Startup and Stop Procedures

This section explains the procedure how to start DSADC and how to stop DSADC. The table below shows registers required to set.

Register	Bit	Description
TEMPEN	EN0, EN1	Reference voltage circuit (note)
DSADxCLK	ADCLK	Conversion clock division setting
DSADxCR4	DSGAIN	Gain setting
DSADxCR1	BIASEN, MODEN	Setting of the bias circuit and modulator circuit operation

Note: The reference voltage circuit is shared with a temperature sensor.

15.4.1.1 Startup

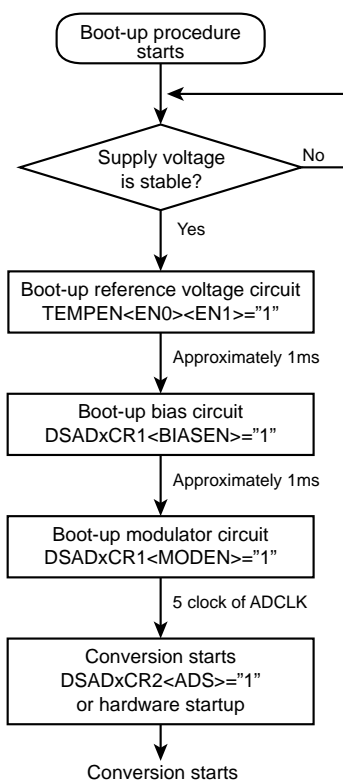


Figure 15-2 Boot-up procedure

Perform the following procedure under the circumstance where supply voltage is stable.

1. Start up the reference voltage circuit

Set "1" to TEMPEN<EN0><EN1> and wait 1 ms or more to secure the stabilization time.

TEMPEN<EN1> must be enabled when TEMPEN<EN0> is enabled. Setting both <EN0> and <EN1> at the same time is capable.

2. Start up the bias circuit

Set "1" to DSADxCR1<BIASEN> and wait 1ms or more to secure the stabilization time.

Supply a conversion clock before DSADxCR1<BIASEN> is set. For details of the conversion clock, refer to "1.4.2 Conversion Clock (ADCLK)".

3. Start up the modulator circuit

Set "1" to DSADxCR1<MODEN>.

After 5 clocks of ADCLK has elapsed, conversion can be enabled.

Before starting conversion, specify the conversion mode (DSADxCR3<REPEAT>) and gain setting (DSADxCR4<DSGAIN>).

4. Start the conversion

Start the conversion by software or hardware.

15.4.1.2 Stop

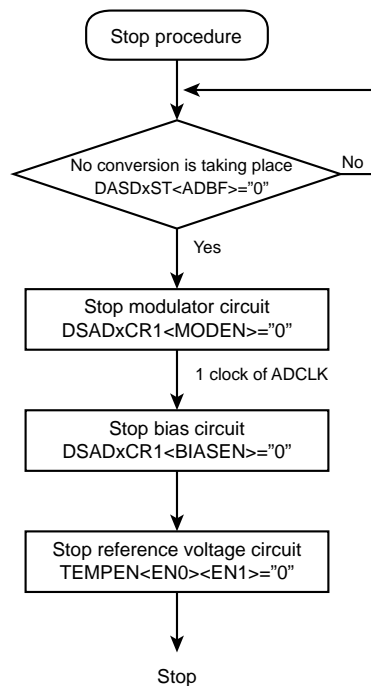


Figure 15-3 Stop procedure

Perform the following procedure under the circumstance where conversion is complete. (DSADxST<ADBF>="0")

1. Stop the modulator circuit

Set "0" to DSADxCR1<MODEN>

2. Stop the bias circuit

Set "0" to DSADxCR1<BIASEN> after one or more ADCLK have elapsed after the modulator circuit is stopped.

3. Stop the reference voltage generation circuit

Set "0" to TEMPEN<EN0><EN1>

Note: The reference voltage circuit is shared with the temperature sensor. While the temperature sensor is operating, do not stop TEMPEN<EN0> (setting "0" means stop).

15.4.2 Conversion Clock (ADCLK)

The conversion clock supplied to the DSADC can be divided by the clock specified with DSADxCLK.

When the conversion clock is changed or stopped, stop the DSADC. (Set "0" to both DSADxCR1<BIASEN> and <MODEN>.)

When the synchronous start function is used, select the same conversion clock for all units.

15.4.2.1 Conversion Time

A conversion time can be calculated by the formula below where the frequency of ADCLK is f_{ADCLK} .

$$\text{Conversion time} = 1 / f_{ADCLK} \times 2640 + \text{Fixed delay time [s]}$$

The fixed delay time is 673 to 675 clocks at the first conversion in repeat conversion and single conversion modes. In repeat conversion, the fixed delay time is 0 clock at the 2nd conversion or later.

For example, the conversion time is 165 μ s at the 2nd conversion or later where $f_c/1$ is selected at $f_c=16$ MHz.

15.4.3 Conversion Mode

The DSADC provides two conversion modes: single mode and repeat mode. In single mode, conversion is performed once; in repeat mode conversion is sequentially performed. The mode is set with DSADxCR3<REPEAT>.

15.4.4 Starting Conversion

Conversion is started by software or hardware triggers.

When a software trigger is used, set "1" to DSADxCR2<ADS> to start conversion.

When a hardware trigger is used, first select either an external trigger or internal trigger with DSADxCR3<DSADHS>, and then enable a hardware trigger with <DSADHTG>. When a hardware trigger occurs, conversion is started.

Note 1: After conversion has started, restart is ignored.

Note 2: Even when hardware startup is enabled, conversion can be started by software.

15.4.5 Conversion Status

Conversion status can be checked with DSADxST.

During the conversion, DSADxST<ADBF> is "1". After conversion, DSADxST<EOCF> is "1". <EOCF> is cleared by reading DSADxST.

In repeat conversion mode, DSADxST<ADBF> holds "1" during operation. When repeat conversion is complete, DSADxST<ADBF> is cleared to "0".

15.4.6 Switching the Conversion Object

During repeat conversion, external inputs and internal inputs can be switched with DSADxCR5<DSADCH>. However, do not switch the conversion object in the following time periods; 16 ADCLK clocks before conversion completion, and 220 ACLK clocks after conversion completion. Note that there is a time difference between actual conversion completion and conversion completion interrupt generation or the DSAEOC output. Therefore, provide enough margin for the above time period.

Conversion results are invalid until the 2nd conversion result. Use the 3rd or later results.

15.4.7 Stopping Conversion

In single mode, when conversion is complete, the DSADC is automatically stopped.

In repeat mode, in order to stop conversion, set "0" to DSADxCR3<REPEAT>. The DSADC suspends the current conversion, and then stops. At this time, a conversion end interrupt does not occur.

Note: If repeat conversion is completed by setting "0" to <REPEAT>, do not modify other bits of DSADxCR3.

15.4.8 Conversion End

When conversion ends, a conversion end interrupt occurs. This conversion result is stored in DSADxRES<ADR>, and DSADxRES<ADRF> is set to "1". The DSADEOC pin outputs "High" pulses during 10 ADCLK clocks.

If a conversion end interrupt is not used, poll DSADxST<EOCF>. If <EOCF> is "1", conversion is complete.

If <ADR> is written to the next result before reading the current value, <ADOVR> is set to "1". <ADRF> and <ADOVR> are cleared by reading DSADxRES.

15.4.9 Conversion Result

The table below shows the conversion results where differential input voltage is ± 1 V.

Note: VREFINx = 2.75 V

AINxP - AINxN	Conversion result
+1 V	0x6A5900
0 V	0x000000
-1 V	0x95A700

15.5 Synchronous Start Function

The master unit and slave unit can start conversion simultaneously. For details of the assignment of the master and slaves in this product, refer to "Product Information."

15.5.1 Startup

The following flowchart shows the setting procedure when the synchronous start function is used.

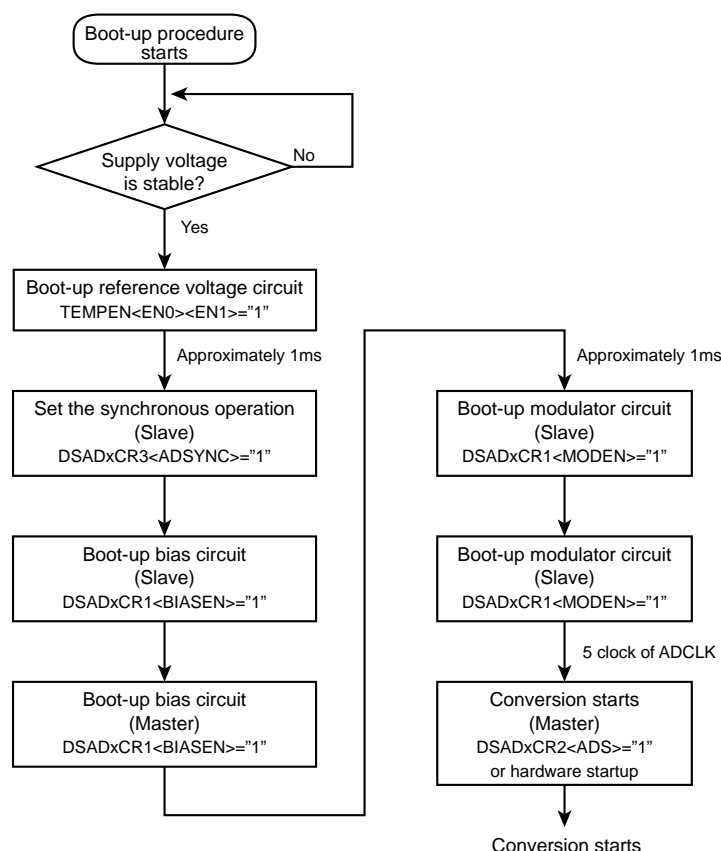


Figure 15-4 Synchronous start function procedure

1. Start up the reference voltage circuit
 - Set "1" to TEMPEN<EN0><EN1> and wait 1 ms or more to secure the stabilization time.
 - TEMPEN<EN1> must be enabled when TEMPEN<EN0> is enabled. Setting both <EN0> and <EN1> at the same time is capable.
2. Set the synchronous operation
 - When the synchronous start function is used, set the slave unit DSADxCR3<ADSYNC> to "1" and set the master unit <ADSYNC> to "0".
3. Start up the bias circuit
 - Set "1" to DSADxCR1<BIASEN> and wait 1 ms or more to secure the stabilization time. When <BIASEN> is set, specify slaves first, and then specify the master.
 - Supply a conversion clock to each module before DSADxCR<BIASEN> is set. For details of a conversion clock, refer to "1.4.2 Conversion Clock (ADCLK)".

4. Start up the modulator circuit

Set "1" to DSADxCR1<MODEN>. When <MODEN> is set, specify slaves first, and then specify the master. After 5 ADCLK clocks have elapsed, conversion can be enabled.

5. Start the conversion

Start conversion by software or hardware. Conversion should be started at the master channel. Do not enable software startup or hardware startup at slave channels.

Conversion mode (DSADxCR3<REPEAT> and gain setting (DSADxCR4<DSGAIN>) can be set in each unit. Set them in each unit before conversion starts.

15.5.2 Stop

In single conversion mode, DSADC stops when conversion is complete in each unit.

In repeat conversion mode, stop DSADC as follows:

- Stop only slave units

Change the conversion mode of a slave unit (set DSADxCR3<REPEAT> to "0") or perform a software reset.

At this time, the master unit continues conversion.

- Stop only master side

Cancel the synchronous operation by setting DSADxCR3<ADSYNC> of slave units to "0", then stop conversion in the master unit by changing the conversion mode or by performing a software reset.

At this time, slave units continue conversion.

- Stop both the master unit and slave units

Stop conversion in the slave units, then stop conversion in the master unit by changing the conversion mode or by performing a software reset.

15.6 Conversion Start Correction Function

By using the conversion start correction function, the conversion start time can be delayed from the time at which DSADxCR2<ADS> is set to "1".

This function is enabled by setting "1" to DSADxADJ<ADJ>. The delay time is set with DSADxADJ<OFFSET>. After the delay time of <OFFSET> \times 1/fsys has elapsed, conversion starts synchronously with ADCLK.

In the synchronous operation, the desired delay time can be set to <OFFSET> in each unit. Conversion starts after the time defined in <OFFSET> has elapsed from the time at which <ADS> of the master unit is set to "1".

Duration between setting "1" to <ADS> and starting conversion, do not modify the value of <OFFSET>.

16. Temperature Sensor (TEMP)

16.1 Outline

The MCU measures a relative temperature using a temperature sensor.

A temperature sensor outputs a voltage based on the reference voltage circuit (BGR) according to temperatures. The output voltage is input to Unit D in the $\Delta\Sigma$ analog/digital converter (DSADC). With AD conversion, a corresponding digital value to temperatures is obtained.

Note: The reference voltage circuit (BGR) is shared with a $\Delta\Sigma$ type analog/digital converter (DSADC).

A difference among the temperature sensor output voltages is linearity related to temperature changes. To obtain a relative temperature, collect data under several conditions.

If the temperature sensor or DSADC is not used, the reference voltage circuit requires below settings.

- Connect DSRVDD3 and SRVDD to DVDD3
- Connect DSRVSS to DGND

16.2 Block diagram

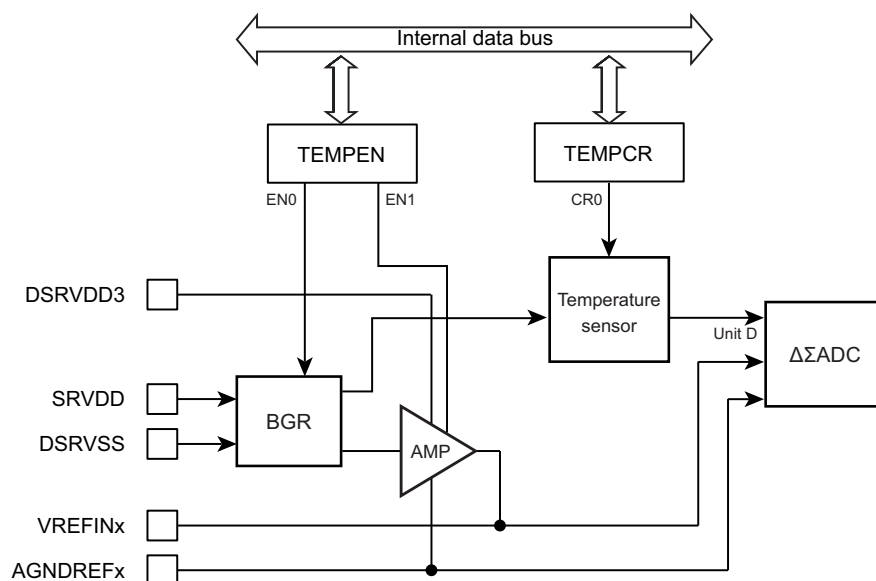


Figure 16-1 Block diagram of Temperature sensor

16.3 Registers

16.3.1 Register List

Then 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	TEMPEN	0x0000
Control register	TEMPCR	0x0004

16.3.2 Details of Register

16.3.2.1 TEMPEN (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	-	-	-	-	-	-	EN1	EN0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-2	-	R	Read as "0".
1	EN1	R/W	AMP operation(note1)(note2) 0: Disabled 1: Enabled Sets AMP for $\Delta\Sigma$ ADC to enable/disable.
0	EN0	R/W	BGR operation 0: Disabled 1: Enabled Sets the reference voltage circuit to enable/disable.

Note 1: TEMPEN<EN1> must be enabled when TEMPEN<EN0> is enabled. Setting both <EN0> and <EN1> at the same time is possible.

Note 2: Do not enable AMP when the reference voltage is applied to VREFINx using the $\Delta\Sigma$ ADC.

16.3.2.2 TEMPCR (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	-	-	-	-	-	-	-	CR0
After reset	0	0	0	0	0	0	0	0

Bit	Bit Symbol	Type	Function
31-1	-	-	Read as "0".
0	CR0	R/W	Temperature sensor operation 0: Disabled 1: Enabled Sets the temperature sensor to enable/disable.

16.4 Operation Description

1. Boot-up

Perform the following procedure while power supply voltage is stable.

1. Boot-up the reference voltage circuit

Set "1" to TEMPEN<EN0> and wait 1 ms or more to be stable.

2. Boot-up the temperature sensor

Set "1" to TEMPPCR<CR0>

Approximately after 10 μ s, an output voltage is enabled.

2. Stop

Perform the following procedure.

1. Stop the temperature sensor

Set "0" to TEMPPCR<CR0>

2. Stop the reference voltage circuit

Set "0" to TEMPEN<EN0>

Note: The reference voltage circuit is shared with $\Delta\Sigma$ ADC. Do not set TEMPEN<EN0> to "0" (stop) during the $\Delta\Sigma$ ADC operation.

17. Power-on-Reset Circuit (POR)

The power-on-reset circuit (POR) generates a power-on reset signal when power-on or power-down.

Note: Due to the fluctuation of supply voltage, the power-on reset circuit may not operate properly. Users should give due consideration based on the electrical characteristic in the device designing.

17.1 Configuration

The POR consists of the reference voltage generation circuit, comparator and the power-on counter.

This circuit compares a voltage divided by the ladder resistor with a reference voltage generated in the reference voltage generation circuit using the comparator.

Power supply voltage means DVDD3.

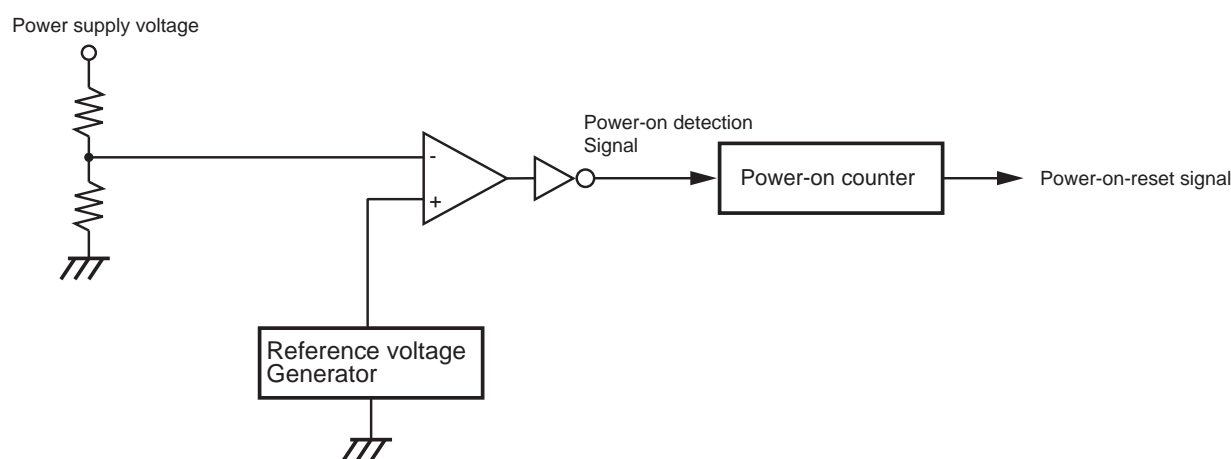


Figure 17-1 Power-on-reset circuit

17.2 Function

17.2.1 Operation at Power-on

At power-on, a power-on detection signals generate as long as the power supply voltage is lower than the power-on-reset release voltage (1.25V to 1.75V).

After waiting time (approximately 0.5ms) has elapsed, the power-on-counter overflows and the power-on reset state is released.

Power supply voltage must be increased to the operating voltage range before release of the power-on reset. At power-on, 10 μ s or more is necessary to increase the voltage to the operating voltage range from 0V.

While the power-on reset signals are generated, the CPU and the peripheral functions are in reset state.

17.2.2 Operation at Power-down

At the power-down, when the power the power supply voltage is lower than the power-on-reset detection voltage (1.20V to 1.70V), a power-on detection signal is generated and the CPU and the peripheral function are in the reset state.

While the power-on reset signals are generated, the CPU and the peripheral functions are in reset state.

17.2.3 Operation at re-power-on

At the power-down, if the power supply voltage is lower than the power-on-reset detection voltage, the power supply voltage must be less than 0V.

Then, supply the power to the MCU under several constraints of the power-on.

If the power supply voltage is higher than 0V or the constraints at power-on are not observed, TMPM311CHDUG will not work correctly.

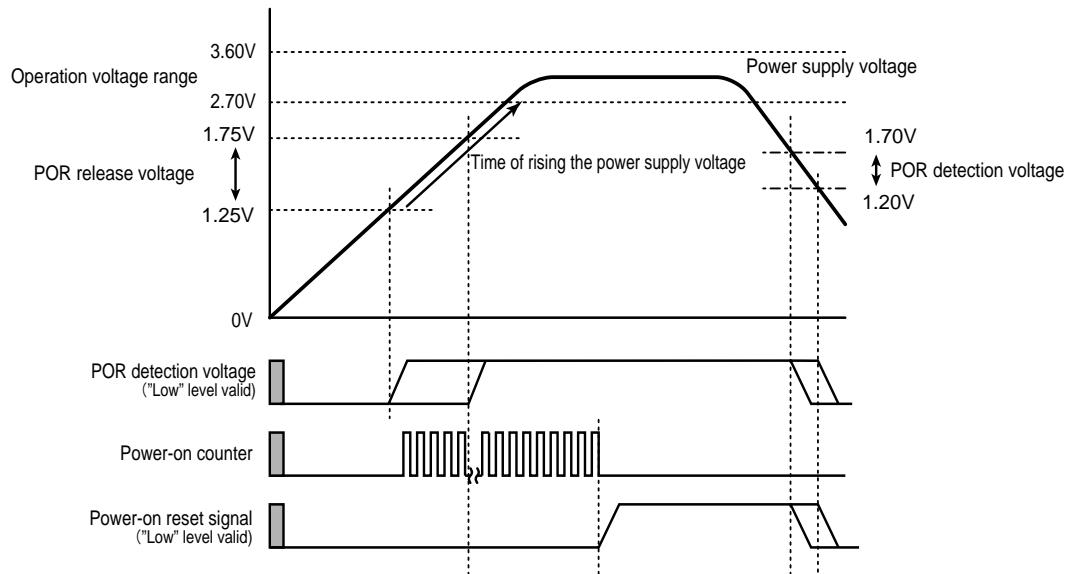


Figure 17-2 Power-on-reset operation timing

18. Watchdog Timer(WDT)

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

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

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

18.1 Configuration

Figure 18-1 shows the block diagram of the watchdog timer.

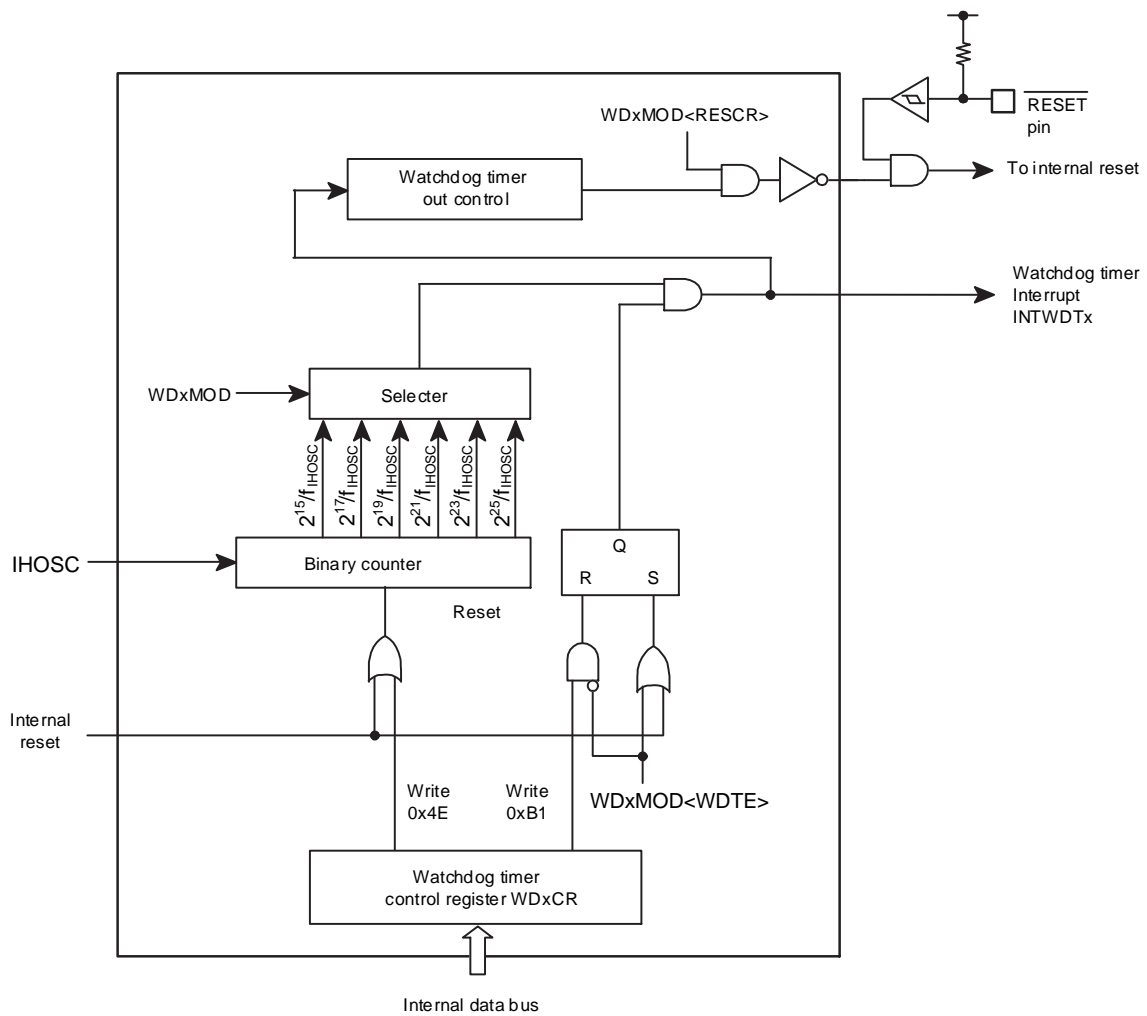


Figure 18-1 Block Diagram of the Watchdog Timer

18.2 Register

18.2.1 Register List

The table below shows control registers and their addresses.

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

Peripheral function: WDT

Register name		Address(Base+)
Watchdog Timer Mode Register	WDxMOD	0x0000
Watchdog Timer Control Register	WDxCR	0x0004
Watchdog Flag register	WDxFLG	0x0008

18.2.2 WDxMOD(Watchdog Timer Mode Register)

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

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

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

18.2.3 WDxCR (Watchdog Timer Control Register)

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

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

18.2.4 WDxFLG (Watchdog Flag Register)

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

Bit	Bit Symbol	Type	Function
31-2	-	R	Read as 0.
1	-	R	Reads as undefined.
0	FLG	R	Flag for writing to registers 0: Enable 1: Disable When writing to WDxMOD or WDxCR, confirm "0" of this bit.

18.3 Description of Operation

18.3.1 Basic Operation

The Watchdog timer is consists of the binary counters that work using the count clock (IHOSC) as an input. Detecting time can be selected between 2^{15} , 2^{17} , 2^{19} , 2^{21} , 2^{23} and 2^{25} by the $WDxMOD<WDTP[2:0]>$.

The detecting time as specified is elapsed, the watchdog timer interrupt (INTWDTx) generates.

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

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

18.3.2 Operation Status

The watchdog timer begins operation immediately after a reset is released. If not using the watchdog timer, it should be disabled.

18.3.3 Operation when malfunction(runway) is detected.

18.3.3.1 INTWDTx interrupt generation

In the Figure 18-2 shows the case that INTWDTx interrupt generates ($WDxMOD<RESCR>="0"$).

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

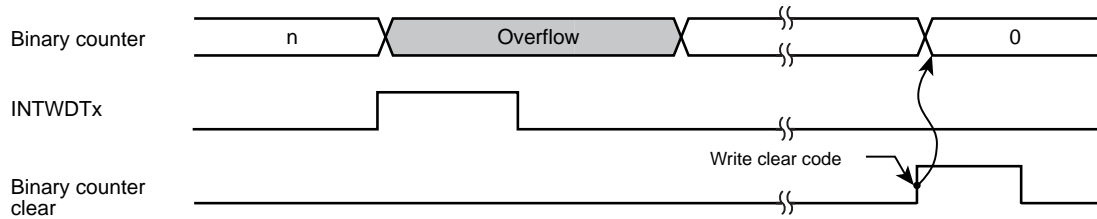


Figure 18-2 INTWDTx interrupt generaiton

18.3.3.2 Internal Resetgeneration

Figure 18-3 shows the internal reset generation ($WDxMOD<RESCR>="1"$).

MCU is reset by the overflow of the binary counter. In this case, reset status continues for 32 states.

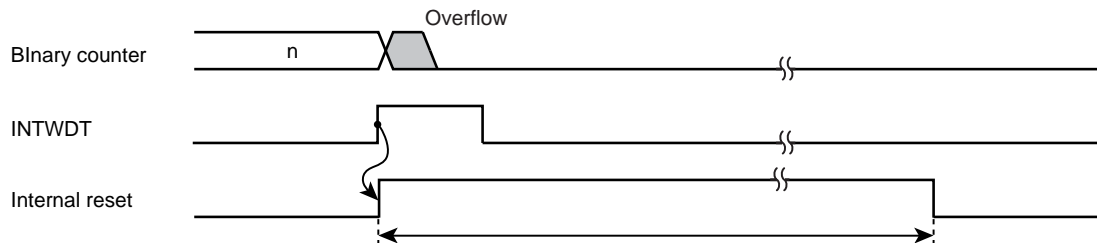


Figure 18-3 Internal reset generation

18.4 Control of the watchdog timer

18.4.1 Register access

When writing to WDxMOD and WDxCR, confirm whether WDxFLG<FLG> is "0".

However, the consecutive writing to WDxMOD and WDxCR is available for the case of the disable control. Confirming WDxFLG<FLG> is required only before writing to WDxMOD.

18.4.2 Disable control

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

18.4.3 Enable control

Set WDxMOD<WDTE> to "1".

18.4.4 Watchdog timer clearing control

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

18.4.5 Detection time of watchdog timer

Set WDxMOD<WDTP[2:0]> depend on the detection time.

For example, in the case that $2^{21}/f_{\text{HOSC}}$ is used, set "011" to WDxMOD<WDTP[2:0]>.

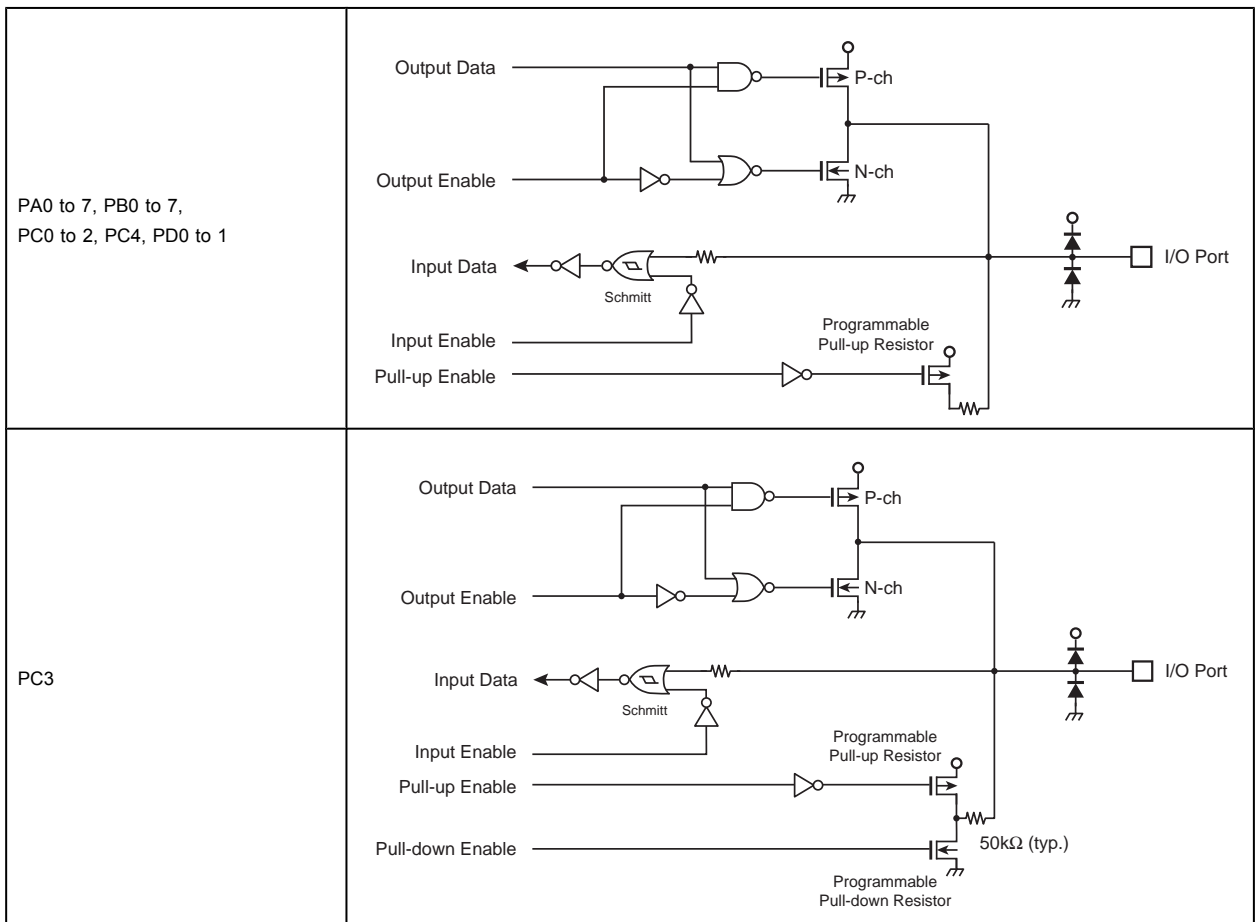
19. Port Section Equivalent Circuit Schematic

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

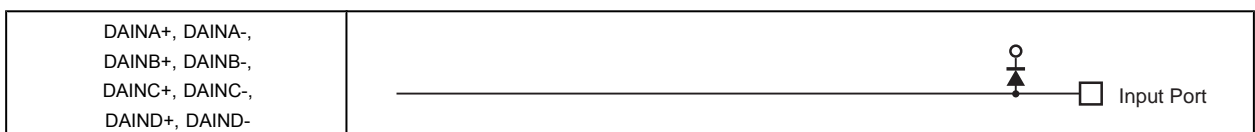
The input protection resistance ranges from several tens of Ω to several hundreds of Ω .

Note: Resistors without values in the figure show input protection resistors.

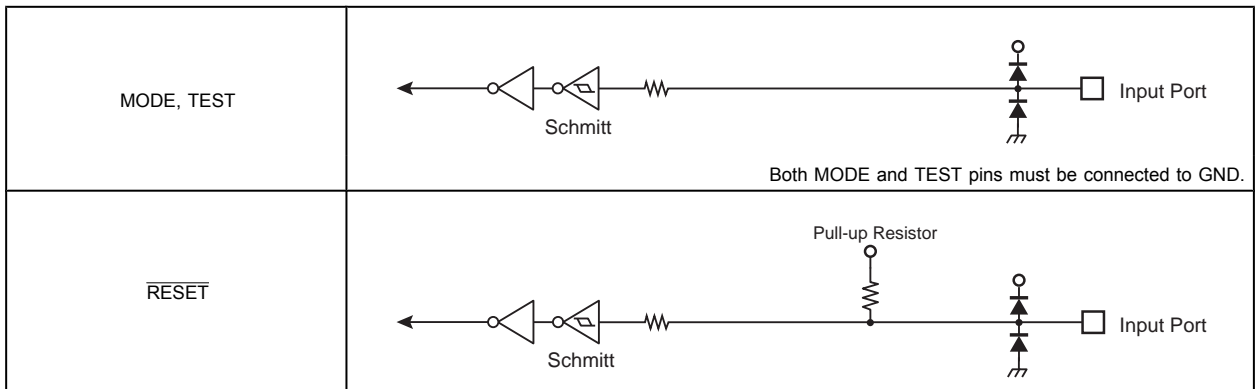
19.1 PORT pin



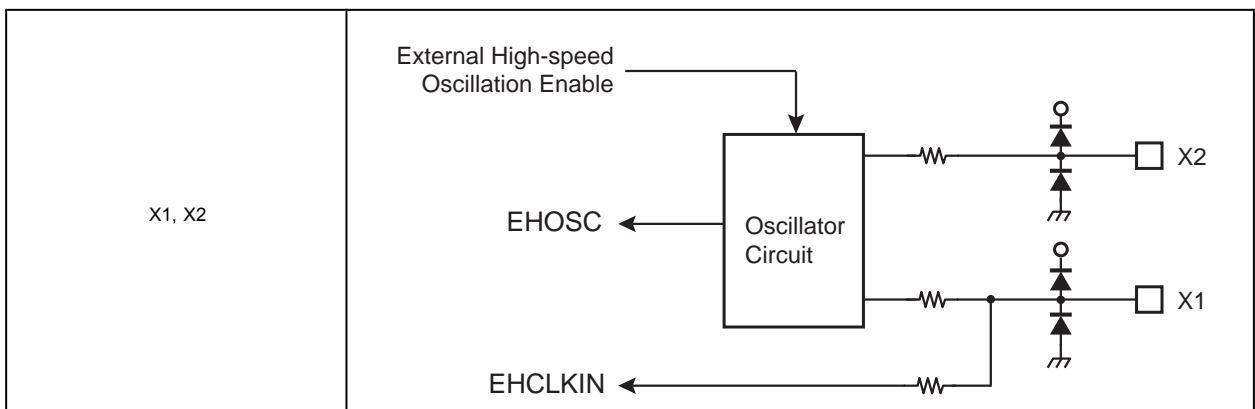
19.2 Analog pin



19.3 Control pin



19.4 Clock pin



20. Electrical Characteristics

20.1 Absolute Maximum Ratings

Parameter		Symbol	Rating	Unit
Supply voltage		DVDD3	-0.3 to 3.9	V
		DSRVDD3	-0.3 to 3.9	
		SRVDD3	-0.3 to 3.9	
Capacitor voltage		REGOUT	-0.3 to 3.0	V
Input voltage	Digital input pins	V_{IN1}	-0.3 to DVDD3 + 0.3	V
	DAIN0+, DAIN0-, DAIN1+, DAIN1-, DAIN2+, DAIN2-, DAIN3+, DAIN3-	V_{IN2}	-0.375 to DSRVDD3 + 0.3	
Low-level output current	Per pin	I_{OL}	5	mA
	Total	ΣI_{OL}	50	
High-level output current	Per pin	I_{OH}	-5	
	Total	ΣI_{OH}	50	
Power consumption (Ta = 85 °C)		PD	600	mW
Soldering temperature(10 s)		T_{SOLDER}	260	°C
Storage temperature		T_{STG}	-40 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 blowup and/or burning.

20.2 DC Electrical Characteristics (1/2)

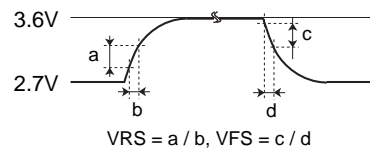
DVDD3 = DSRVDD3 = SRVDD = 2.7 V to 3.6 V
 DVSS = DSRVSS = 0V
 Ta = -40 to 85 °C

Parameter		Symbol	Condition	Min.	Typ. (Note 1)	Max.	Unit
Supply voltage	DVDD3 DSRVDD3 RSVDD	VDD DSRVDD3 RSVDD	$f_{OSC} = 8 \text{ to } 24 \text{ MHz}$ $f_{SYS} = 1 \text{ to } 24 \text{ MHz}$	2.7	-	3.6	V
Low-level input voltage	PA0 to 7, PB0 to 7, PC0 to 4, PD0 to 1, TEST, MODE, $\overline{\text{RESET}}$	V_{IL}	-	-0.3	-	0.25 DVDD3	V
High-level input voltage	PA0 to 7, PB0 to 7, PC0 to 4, PD0 to 1, TEST, MODE, $\overline{\text{RESET}}$	V_{IH}	-	0.75 DVDD3	-	DVDD3 + 0.3	V
Low-level output voltage		V_{OL}	$I_{OL} = 2 \text{ mA}$	-	-	0.4	V
High-level output voltage		V_{OH}	$I_{OH} = -2 \text{ mA}$	2.4	-	DVDD3	V
Input leakage current		I_{LI}	$0.0 \leq V_{IN} \leq \text{DVDD3}$	-	0.02	± 5	μA
Output leakage current		I_{LO}	$0.2 \leq V_{IN} \leq \text{DVDD3} - 0.2$	-	0.05	± 10	
Schmitt trigger input width		V_{TH}	$2.7 \text{ V} \leq \text{DVDD3} \leq 3.6 \text{ V}$	0.3	0.6	-	V
Pull-up resistor at Reset		RRST	DVDD3 = 2.7 V to 3.6 V	-	50	150	k Ω
Programmable pull-up/pull-down resistor		PKH	DVDD3 = 2.7 V to 3.6 V	-	50	150	k Ω
Power supply variation rate in operation range		VRS	DSRVDD3 = DVDD3	-	-	10	mV/ μs
		VFS		-	-	-10	
Pin capacitance (Except power supply pins)		C_{IO}	$f_c = 1 \text{ MHz}$	-	-	10	pF
Low-level output current		I_{OL}	Per pin	-	-	2	mA
		ΣI_{OL}	Total, all Port	-	-	35	mA
High-level output current		I_{OH}	Per pin	-	-	-2	mA
		ΣI_{OH}	Total, all Port	-	-	-35	mA

Note 1: Ta = 25 °C, DVDD3 = DSRVDD3 = SRVDD = 3.3 V, unless otherwise noted.

Note 2: The same voltage must be supplied to DVDD3, DSRVDD3 and SRVDD.

Note 3: VRS(Rising), VFS(Falling) should be measured at a strict level against a characteristics.



20.3 DC Electrical Characteristics (2/2)

Ta = -40 to 85 °C

Parameter	Symbol	Condition		Min.	Typ. (Note)	Max.	Unit
		System clock (fsys)	Operating conditions				
Current consumption	IDD	20MHz	Refer to Table 20-1 Table 20-2, regarding to the operation condition	-	6.4	7.2	mA
		24MHz		-	7.5	8.5	

Note: Ta = 25 °C, DVDD3 = DSRVDD3 = SRVDD = 3.3 V, unless otherwise noted.

Table 20-1 IDD Measurement Condition (Pin condition, Oscillator)

Pin condition	DVDD3 = DSRVDD3 = SRVDD	3.3 V
	Input pin	Fixed
	Output pin	Opened
Operating conditions (Oscillator)	External high-speed oscillator (EHOSC)	Enabled
	Internal high-speed oscillator (IHOSC)	Enabled

Table 20-2 IDD Measurement Condition (CPU, Peripheral circuit)

Circuit	The number of equipped circuits	
CPU	1	Enabled (Drystone Ver. 2.1)
μDMAC	1	Transfer source:SIO transmit Data transfer:RAM to SIO
DSADC	4	Unit A to D:Enabled
TMRB	4	Ch0 to 3:Enabled
WDT	1	Enabled
SIO/UART	1	SIO, transmit (10Mbps)
SSP	1	SPI, transmit (5Mbps)
I/O port	4	Disabled
TMR16A	1	Enabled
TEMP	1	Enabled

20.4 24-bit $\Delta\Sigma$ ADC Electrical Characteristics

DVDD3 = DSRVDD3 = SRVDD = 2.9V to 3.6V

DVSS = DSRVSS = 0V

Ta = -40 to 85 °C

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Analog reference voltage(+)	dsVREFIN	Internal BGR	-	2.75	-	V
		Applied externally	2.7	-	2.8	
Consumption current of AMP	dsAMP _{Icc}	-	-	0.5	1.0	mA
Consumption current of conversion (per 1unit)	dsAD _{Icc}	-	-	1.4	2.5	mA
Input range	AINP AINN	-	-0.375	-	1	V
Differential input voltage	V _{pp}	-	-1	-	1	V
SNDR	dsSNDR	GAIN = ×1	-	90	-	dB
Input impedance	dsinp	Conversion time ≥ 330 μs Input amplitude = ±500mV Offset = 500mV	-	49.5 (note 2)	-	kΩ
Conversion time	T _{convds}	-	112	-	-	μs

Note 1: Peripheral functions are disable.

Note 2: A value when AINN=0V.

20.5 Temperature Sensor Characteristics

DVDD3 = DSRVDD3 = SRVDD = 2.7V to 3.6V

DVSS = DSRVSS = 0V

Ta = -40 to 85 °C

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Consumption current of reference voltage circuit (Note 1)	BGR _{Icc}	-	-	0.2	0.5	mA
Consumption current of temperature sensor	TEMP _{Icc}	-	-	0.7	1.0	mA
Relative error (Note 2)	-	Ta = -20 to 85 °C	-	-	±3	°C
		Ta = -40 to 85 °C	-	-	±5	

Note 1: The reference voltage circuit is shared with a $\Delta\Sigma$ analog/digital converter.

Note 2: These are design assurance values of single temperature sensor, which were obtained from a straight-line approximation based on the measured values at 30 °C and 60 °C.

20.6 AC Electrical Characteristics

20.6.1 Serial Channel (SIO/UART)

20.6.1.1 AC measurement condition

- Output levels: High = $0.8 \times DVDD3$, $0.2 \times DVDD3$
- Input levels: High = $0.75 \times DVDD3$, $0.25 \times DVDD3$
- Load capacity: $CL = 30pF$

20.6.1.2 I/O Interface mode

In the table below, the letter x represents the SIO operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

(1) SCLK input mode

[Input]

Parameter	Symbol	Equation		fsys = 24 MHz		Unit
		Min.	Max	Min.	Max	
SCLK Clock High width (input)	t _{SCH}	4x	-	167	-	ns
SCLK Clock Low width (input)	t _{SCL}	4x	-	167	-	
SCLK cycle	t _{SCY}	t _{SCH} + t _{SCL}	-	334	-	
Valid Data input ← SCLK rise or fall (Note 1)	t _{SRD}	30	-	30	-	
SCLK rise → Input Data hold or fall (Note 1)	t _{HSR}	x + 30	-	72	-	

[Output]

Parameter	Symbol	Equation		fsys = 24 MHz		Unit
		Min.	Max	Min.	Max	
SCLK Clock High width (input)	t _{SCH}	4x	-	170 (note 3)	-	ns
SCLK Clock Low width (input)	t _{SCL}	4x	-	170 (note 3)	-	
SCLK cycle	t _{SCY}	t _{SCH} + t _{SCL}	-	340	-	
Output Data ← SCLK rise or fall (Note 1)	t _{OSS}	t _{SCY} /2 - 3x - 45	-	0 (note 2)	-	
SCLK rise → Output Data hold or fall (Note 1)	t _{OHS}	t _{SCY} /2	-	170	-	

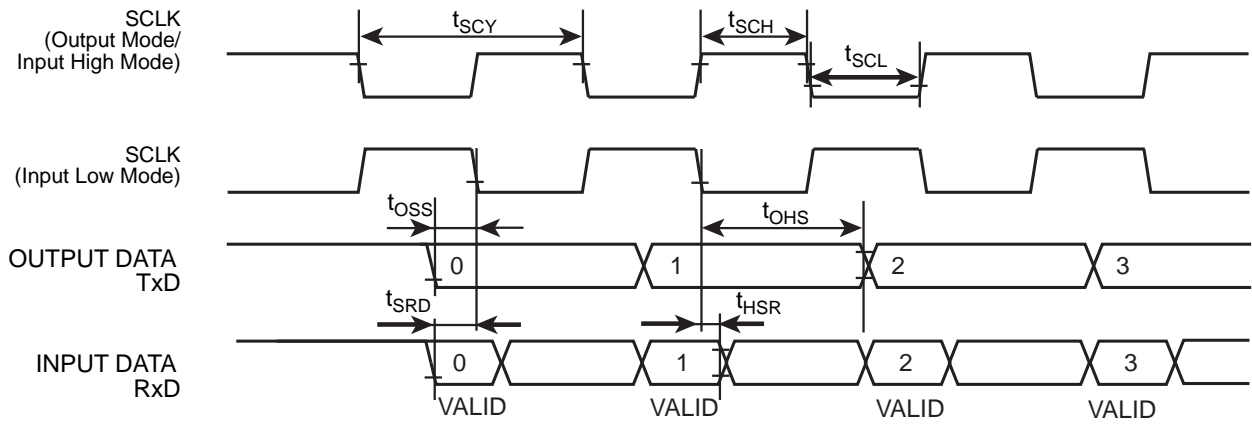
Note 1: SCLK rise/fall : SCLK rise mode uses the rise timing of SCLK. SCLK fall mode uses the fall timing of SCLK.

Note 2: Use the frequency of SCLK in a range where the calculation value keeps positive.

Note 3: The value indicates a minimum value that enables t_{oss} to be zero or more

(2) SCLK output mode

Parameter	Symbol	Equation		f _{sys} = 24 MHz		Unit
		Min.	Max	Min.	Max	
SCLK cycle (programmable)	t _{SCY}	2x	-	83	-	ns
Output Data ← SCLK rise	t _{OSS}	t _{SCY} /2 - 20	-	22	-	
SCLK rise → Output Data hold	t _{OHS}	t _{SCY} /2 - 20	-	22	-	
Valid Data Input ← SCLK rise	t _{SRD}	45	-	45	-	
SCLK rise → Input Data hold	t _{HSR}	0	-	0	-	



20.6.2 Synchronous serial Interface (SSP)

20.6.2.1 AC measurement conditions

The letter "T" used in the equations in the table represents the period of internal bus frequency (fsys).

- Output levels: High = $0.8 \times DVDD3$, Low = $0.2 \times DVDD3$
- Input levels: High = $0.75 \times DVDD3$, Low = $0.25 \times DVDD3$
- Load capacity: CL = 30pF

20.6.2.2 AC Electrical Characteristics

Baud rate clock is set below condition.

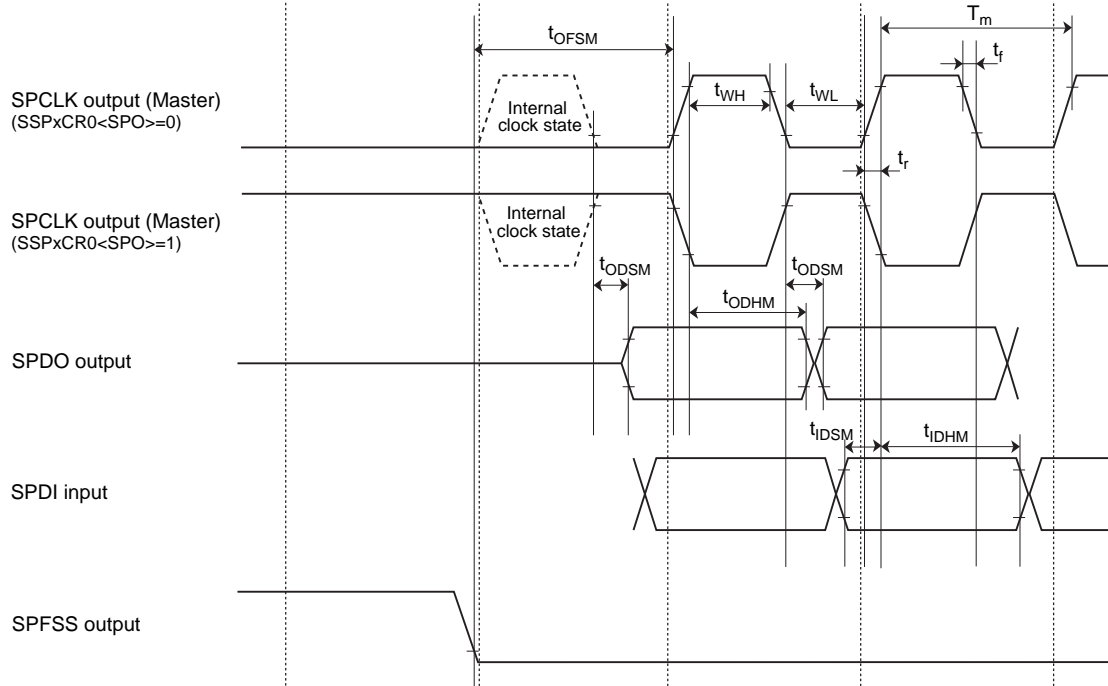
- Master Mode
 $m = (\langle CPSDVR \rangle \times (1 + \langle SCR \rangle)) = f_{sys}/SPCLK$
 $\langle CPSDVR \rangle$ is set only even number. $65024 \geq m \geq 2$
- Slave mode
 $n = f_{sys}/SPCLK$ ($65024 \geq n \geq 12$)

Parameter	Symbol	Equation		f _{sys} = 24MHz (m=2, n=12)		Unit
		Min.	Max.	Min.	Max.	
SPCLK Period (Master)	T _m	(m)T However more than 83ns	-	83 (12MHz)	-	ns
SPCLK Period (Slave)	T _s	(n)T However more than 500ns	-	500 (2MHz)	-	
SPCLK rise up time	t _r	-	15	-	15	
SPCLK fall down time	t _f	-	15	-	15	
Master mode: SPCLK low level pulse width	t _{WLM}	(m)T/2 - 15	-	24	-	
Master mode: SPCLK high level pulse width	t _{WHM}	(m)T/2 - 15	-	24	-	
Slave mode: SPCLK low level pulse width	t _{WLS}	(n)T/2 - 15	-	235	-	
Slave mode: SPCLK high level pulse width	t _{WHS}	(n)T/2 - 15	-	235	-	
Master mode: SPCLK rise/fall → output data valid	t _{ODSM}	-	15	-	15	
Master mode: SPCLK rise/fall → output data hold	t _{ODHM}	(m)T/2 - 15	-	24	-	
Master mode: SPCLK rise/fall → input data valid delay time	t _{IDSM}	20	-	20	-	
Master mode: SPCLK rise/fall → input data hold	t _{IDHM}	0	-	0	-	
Master mode: SPFSS valid → SPCLK rise/fall	t _{OFSM}	(m)T - 15	(m)T + 15	68	98	
Slave mode: SPCLK rise/fall → output data valid delay time	t _{ODSS}	-	(3T) + 40	-	155	
Slave mode: SPCLK rise/fall → output data hold	t _{ODHS} (Note1)	(n)T/2 + (2T)	-	333	-	
Slave mode: SPCLK rise/fall → input data valid delay time	t _{IDSS}	10	-	10	-	
Slave mode: SPCLK rise/fall → input data hold	t _{IDHS}	(3T) + 15	-	140	-	
Slave mode: SPFSS valid → SPCLK rise/fall	t _{OFSS}	(n)T + 10	-	510	-	

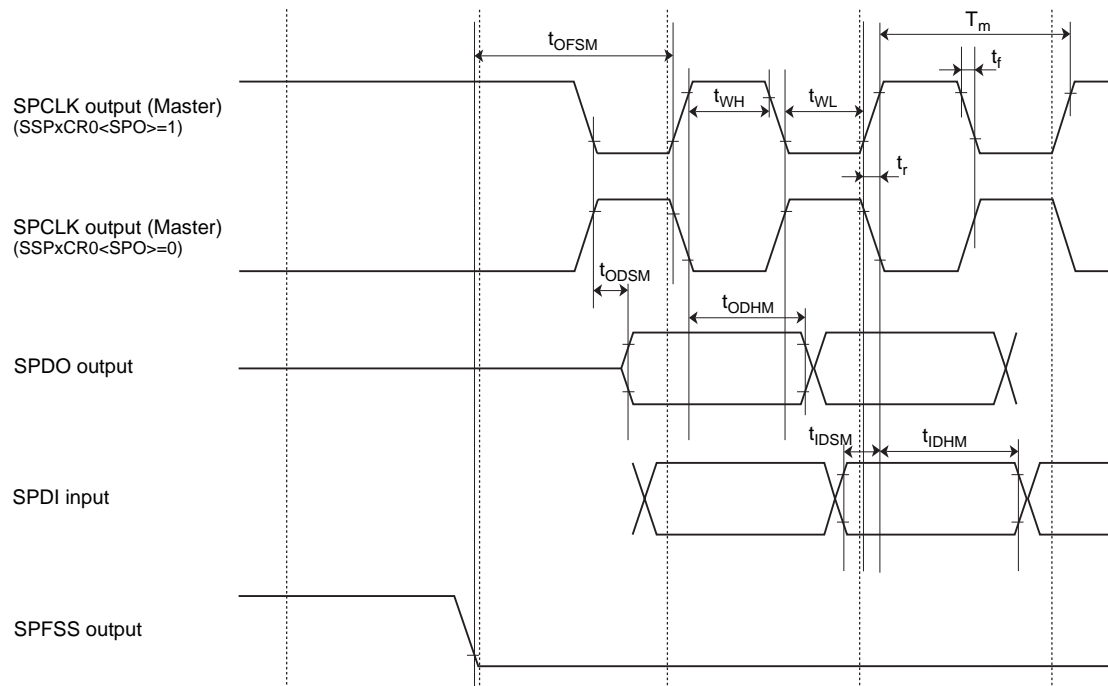
(1) SSP SPI mode (Master)

- $f_{sys} \geq 2 \times f_{SPCLK}$ (Max)
- $f_{sys} \geq 65024 \times f_{SPCLK}$ (Min)

(1) Master SSPCR0<SPH>="0"(Data is latched on the first edge.)



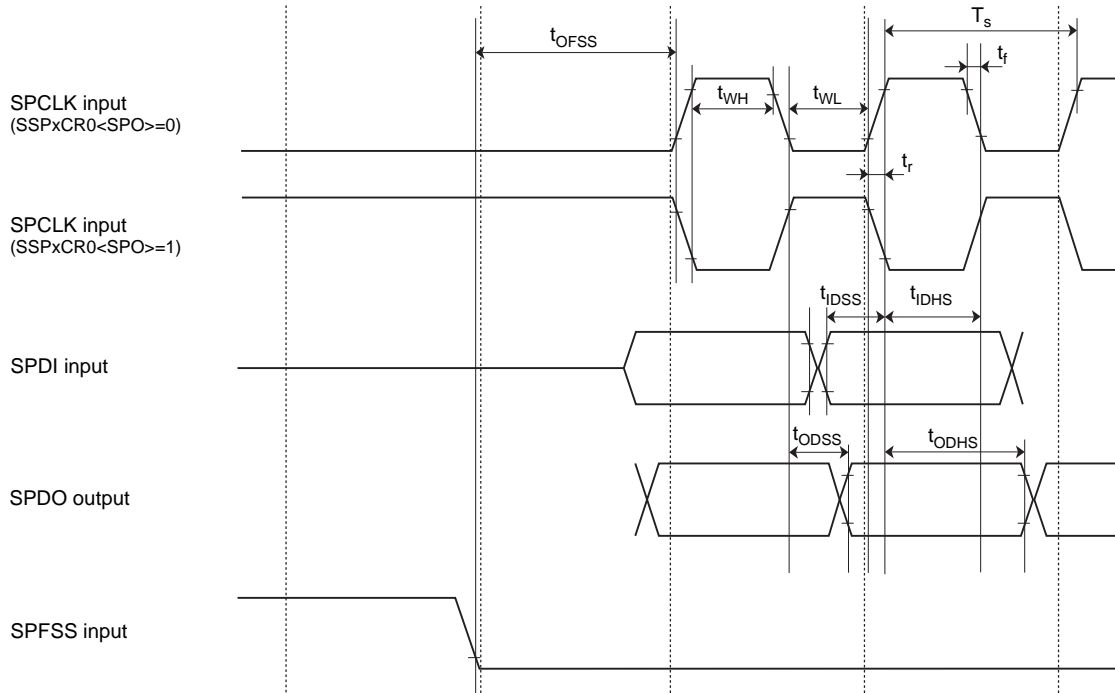
(2) Master SSPCR0<SPH>="1" (2nd Data is latched on the second edge.)



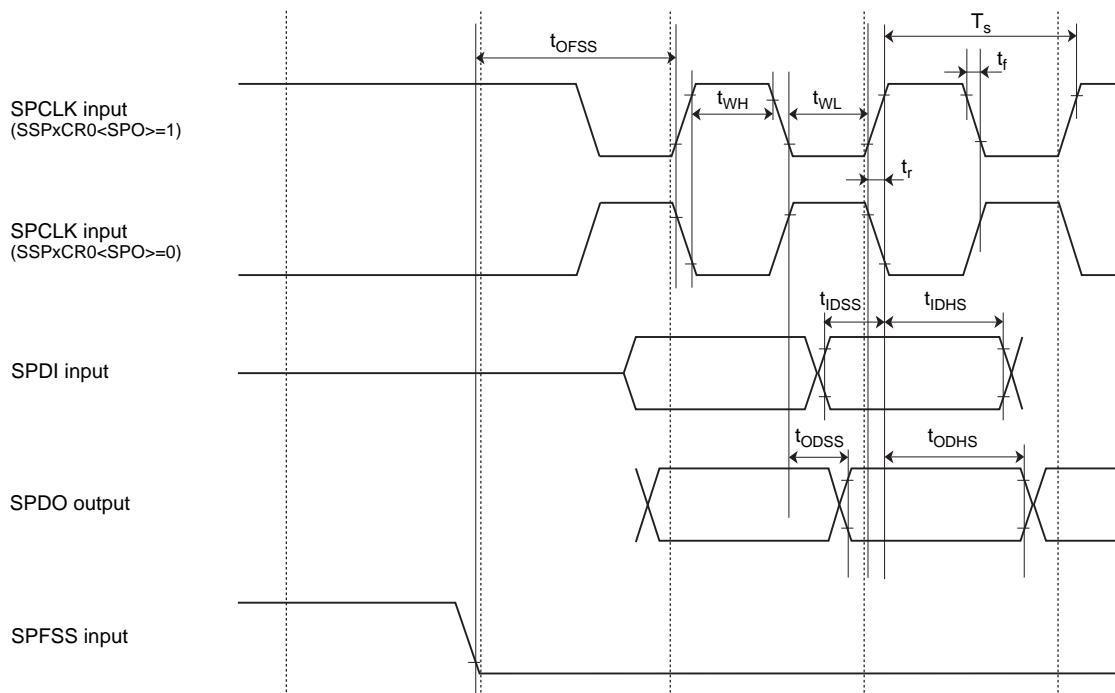
(2) SSP SPI mode(Slave)

- $f_{sys} \geq 12 \times f_{SPCLK}$ (Max)
- $f_{sys} \geq 65024 \times f_{SPCLK}$ (Min)

(1) Slave SSPCR0<SPH>="0"(Data is latched on the first edge.)



(2) Slave SSPCR0<SPH> = "1" (Data is latched on the second edge.)



20.6.3 16-bit Timer / Event counter (TMRB)

20.6.3.1 Event Counter

(1) AC measurement conditions

- Input levels: High = $0.75 \times DVDD3$, Low = $0.25 \times DVDD3$
- Load capacity: CL = 30pF

(2) AC Electrical Characteristics

In the table below, the letter x represents the TMRB operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		fsys = 24 MHz		Unit
		Min.	Max	Min.	Max	
Clock Low pulse width	t _{VCKL}	2x + 100	-	183	-	ns
Clock High pulse width	t _{VCKH}	2x + 100	-	183	-	ns

20.6.3.2 Capture

(1) AC measurement conditions

- Input levels: High = $0.75 \times DVDD3$, Low = $0.25 \times DVDD3$
- Load capacity: CL = 30pF

(2) AC Electrical Characteristics

In the table below, the letter x represents the TMRB operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		fsys = 24 MHz		Unit
		Min.	Max	Min.	Max	
Low pulse width	t _{CPL}	2x + 100	-	183	-	ns
High pulse width	t _{CPH}	2x + 100	-	183	-	ns

20.6.4 External Interrupt

20.6.4.1 AC measurement conditions

- Input levels: High = $0.75 \times DVDD3$, Low = $0.25 \times DVDD3$
- Load capacity: CL = 30pF

20.6.4.2 AC Electrical Characteristics

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

Parameter	Symbol	Equation		fsys = 24 MHz		Unit
		Min.	Max	Min.	Max	
Low level pulse width	t _{INTAL}	x + 100	-	142	-	ns
High level pulse width	t _{INTAH}	x + 100	-	142	-	ns

20.6.5 24-bit ΔΣADC Trigger Input pin AC Characteristics

20.6.5.1 AC measurement conditions

- Input levels: High = $0.75 \times DVDD3$, Low = $0.25 \times DVDD3$
- Load capacity: CL = 30pF

20.6.5.2 AC Electrical Characteristics

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

Parameter	Symbol	Equation		fsys = 24 MHz		Unit
		Min.	Max	Min.	Max	
Low level pulse width	T _{DSADL}	2x + 20	-	104	-	ns
High level pulse interval	T _{DSADH}	2x + 20	-	104	-	ns

20.6.6 On chip oscillator

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

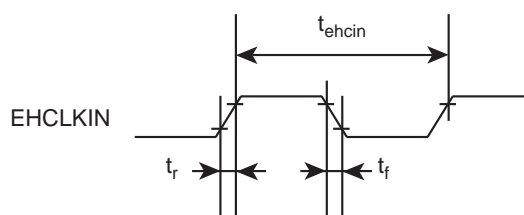
Note: Do not use an on-chip oscillator as a system clock (fsys) when high-accuracy oscillation frequency is required.

20.6.7 External Oscillator

Parameter	Symbol	Condition	Min.	Typ.	Max	Unit
High-frequency oscillation	EHOSC	Ta = -40 to 85°C	8	-	20	MHz

20.6.8 External Clock Input

Parameter	Symbol	Min.	Typ.	Max	Unit
External clock frequency	t_{ehcin}	8	-	24	MHz
External clock duty	-	45	-	55	%
External clock input rise time	t_r	-	-	10	ns
External clock input fall time	t_f	-	-	10	ns



20.6.9 Noise Filter Characteristic

Parameter	Condition	Min.	Typ.	Max	Unit
High-frequency oscillation	-	15	30	60	ns

20.7 Recommended Oscillation Circuit

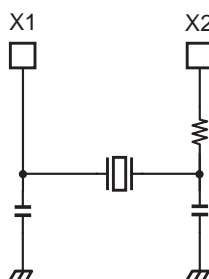


Figure 20-1 High-frequency oscillation connection

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 TMPM311CHDUG has been evaluated by the oscillator vendor below. Please refer this information when selecting external parts

20.7.1 Ceramic Oscillator

The TMPM311CHDUG recommends the high-frequency oscillator by Murata Manufacturing Co., Ltd.

Please refer to the Murata Website for details.

20.7.2 Crystal Oscillator

The TMPM311CHDUG recommends the low-frequency oscillator by KYOCERA Corporation.

Please refer to the KYOCERA Website for details.

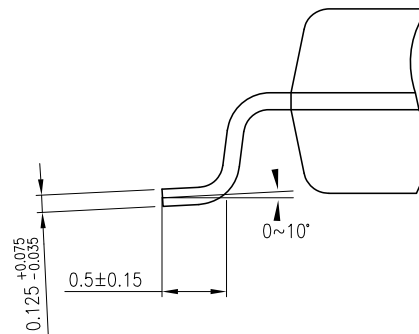
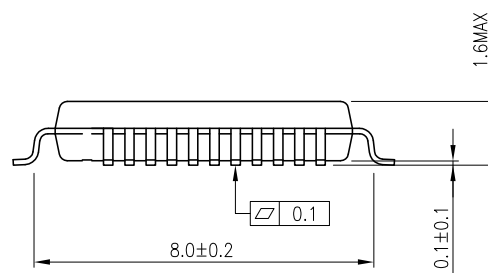
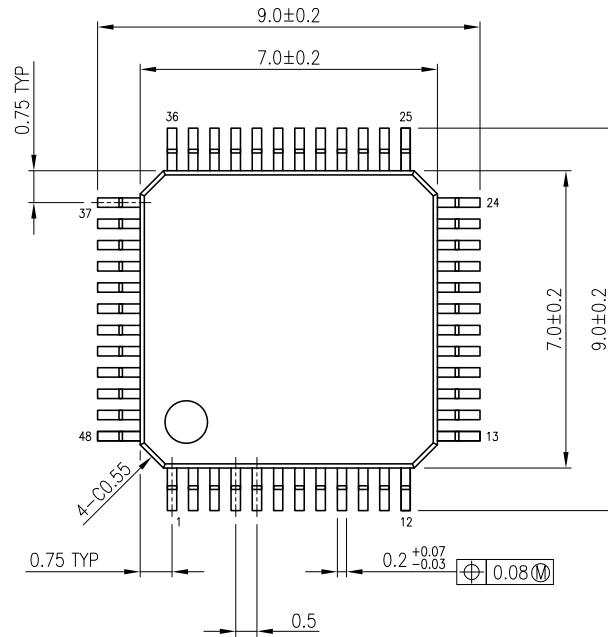
20.7.3 Precautions for designing printed circuit board

Be sure to design printed circuit board patterns that connect a crystal unit with other oscillation elements so that the length of such patterns become shortest possible to prevent deterioration of characteristics due to stray capacitances and wiring inductance. For multi-layer circuit boards, it is important not to wire the ground and other signal patterns right beneath the oscillation circuit. For more information, please refer to the URL of the oscillator vendor.

21. Package Dimensions

Type: LQFP48-P-0707-0.50C

"Unit:mm"



• RESTRICTIONS ON PRODUCT USE

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

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