TOSHIBA

32-bit TX System RISC
TX03 Series

TMPM330FDFG
TMPM330FYFG
TMPM330FWFG

Tentative

TOSHIBA CORPORATION

Semiconductor Company

Revision History

Date	Revision	
2009/6/25	Tentative 1	First Release
2009/9/8	Tentative 2	Contents Revised

Table of Contents

Table of Contents ..... **i**

1. Overview and Features **2**

1.1 Features 2

1.2 Block Diagram 5

2 Pin Layout and Pin Functions **7**

2.1 Pin Layout (Top view) 7

2.2 Pin names and Functions 10

2.2.1 Sorted by Pin 10

2.2.2 Sorted by Port 15

2.3 Pin Names and Power Supply Pins 20

2.4 Pin Numbers and Power Supply Pins 20

3 Processor Core and Memory Map **21**

3.1 Processor Core 21

4 Debug Interface **23**

4.1 Specification Overview 23

4.2 Features of SWJ-DP 23

4.3 Features of ETM 23

4.4 Pin Functions 23

4.5 Connection with a Debug Tool 24

5 Memory Map **25**

5.1 Memory Map of the TMPM330FDFG 26

5.2 Memory Map of the TMPM330FYFG 27

5.3 Memory Map of the TMPM330FWFG 28

6 Reset **29**

6.1 Cold Reset 29

6.2 Warm Reset 30

6.2.1 Reset Period 30

6.2.2 After Reset 30

7 Clock/Mode Control **31**

7.1 Features 31

7.2 Registers 32
7.2.1 Register List 32
7.2.2 Detailed Description of Registers 33
7.3 Clock Control 38
7.3.1 Clock System Block Diagram 38
7.3.2 Initial Values after Reset 38
7.3.3 Clock Multiplication Circuit (PLL) 40
7.3.4 Warm-up Function 40
7.3.5 System Clock 42
7.3.6 Prescaler Clock Control 42
7.3.7 System Clock Pin Output Function 43
7.4 Modes and Mode Transitions 44
7.4.1 Mode Transitions 44
7.5 Operation Modes 45
7.5.1 NORMAL Mode 45
7.5.2 SLOW Mode 45
7.6 Low Power Consumption Modes 45
7.6.1 IDLE Mode 46
7.6.2 SLEEP Mode 46
7.6.3 STOP Mode 46
7.6.4 Low power Consumption Mode Setting 48
7.6.5 Operational State in Each Mode 48
7.6.6 Releasing the Low Power Consumption Mode 49
7.6.7 Warm-up 50
7.6.8 Clock Operations in Mode Transition 51

8 Exceptions 53

8.1 Overview 53
8.1.1 Exception Types 53
8.1.2 Handling Flowchart 54
8.2 Reset Exceptions 60
8.3 Non-Maskable Interrupts (NMIs) 60
8.4 SysTick 61
8.5 Interrupts 62
8.5.1 Interrupt Sources 62
8.5.2 Interrupt Handling 66
8.6 Exception/Interrupt-Related Registers 73

8.6.1 Register List 73
8.6.2 NVIC Registers 74
8.6.3 NVIC Registers 96

9 Input/Output Ports 103

9.1 Port registers 103
9.2 Port Functions 104
9.2.1 Port States in STOP Mode 104
9.2.2 Precaution for Mode Transition 104
9.2.3 Port A (PA0~PA7) 106
9.2.4 Port B (PB0~PB7) 108
9.2.5 Port C (PC0~PC3) 110
9.2.6 Port D (PD0~PD7) 111
9.2.7 Port E (PE0~PE6) 112
9.2.8 Port F (PF0~PF7) 114
9.2.9 Port G (PG0~PG7) 116
9.2.10 Port H (PH0~PH7) 118
9.2.11 Port I (PI0~PI7) 120
9.2.12 Port J (PJ0~PJ7) 122
9.2.13 Port K (PK0~PK2) 124
9.3 Block Diagrams of Ports 126
9.3.1 Port Types 126
9.3.2 Type T1 127
9.3.3 Type T2 128
9.3.4 Type T3 129
9.3.5 Type T4 130
9.3.6 Type T5 131
9.3.7 Type T6 132
9.3.8 Type T7 133
9.3.9 Type T8 134
9.3.10 Type T9 135
9.3.11 Type T10 136
9.3.12 Type T11 137
9.3.13 Type T12 138
9.3.14 Type T13 139
9.3.15 Type T14 140
9.3.16 Type T15 141

9.3.17 Type T16 142
9.3.18 Type T17 143
9.3.19 Type T18 144

10 16-bit Timer/Event Counters (TMRBs) 145

10.1 Outline 145
10.2 Differences in the Specifications 146
10.3 Configuration 148
10.4 Registers 150
10.4.1 TMRB registers 150
10.5 Description of Operations for Each Circuit 163
10.5.1 Prescaler 163
10.5.2 Up-counter (UC0) 168
10.5.3 Timer registers (TB0RG0, TB0RG1) 168
10.5.4 Capture 170
10.5.5 Capture Registers (TB0CP0H/L, TB0CP1H/L) 170
10.5.6 Up-counter capture register (TB0UCH/L) 170
10.5.7 Comparators (CP0, CP1) 170
10.5.8 Timer Flip-flop (TB0FF0) 170
10.5.9 Capture interrupt (INTCAP00, INTCAP01) 170
10.6 Description of Operations for Each Mode 171
10.6.1 16-bit Interval Timer Mode 171
10.6.2 16-bit Event Counter Mode 171
10.6.3 16-bit Programmable Square Wave Output Mode (PPG) 172
10.7 Timer synchronous mode 174
10.8 Applications using the Capture Function 175

11 Serial Channel (SIO) 179

11.1 Features 179
11.2 Block Diagram (Channel 0) 182
11.3 Operation of Each Circuit (Channel 0) 183
11.3.1 Prescaler 183
11.3.2 Baud Rate Generator 188
11.3.3 Serial Clock Generation Circuit 192
11.3.4 Receive Counter 192
11.3.5 Receive Control Unit 192
11.3.6 Receive Buffer 192

11.3.7 Receive FIFO Buffer 194
11.3.8 Receive FIFO Operation 194
11.3.9 Transmit Counter 196
11.3.10 Transmit Control Unit 196
11.3.11 Transmit Buffer 198
11.3.12 Transmit FIFO Buffer 199
11.3.13 Transmit FIFO Operation 199
11.3.14 Parity Control Circuit 201
11.3.15 Error Flag 201
11.3.16 Direction of Data Transfer 202
11.3.17 Stop Bit Length 202
11.3.18 Status Flag 202
11.3.19 Configurations of Transmit/Receive Buffer 203
11.3.20 Software reset 203
11.3.21 Signal Generation Timing 204
11.4 Register Description (Only for Channel 0) 205
11.4.1 Enable register 205
11.4.2 Buffer register 205
11.4.3 Control register 206
11.4.4 Mode control register 0 207
11.4.5 Mode control register 1 208
11.4.6 Mode control register 2 209
11.4.7 Baud rate generator control register(SC0BRCR) Baud rate generator control register 2(SC0BRADD) 210
11.4.8 FIFO configuration register 212
11.4.9 RX FIFO configuration register 213
11.4.10 TX FIFO configuration register 214
11.4.11 RX FIFO status register 215
11.4.12 TX FIFO status register 215
11.5 Operation in Each Mode 216
11.5.1 Mode 0 (I/O interface mode) 216
11.5.2 Mode 1 (7-bit UART Mode) 226
11.5.3 Mode 2 (8-bit UART Mode) 227
11.5.4 Mode 3 (9-bit UART) 228
12 Serial Bus Interface (SBI) 231
12.1 Configuration 232

12.2 Control 233
12.3 I2C Bus Mode Data Formats 234
12.4 Control Registers in the I2C Bus Mode 235
12.5 Control in the I2C Bus Mode 244
12.5.1 Setting the Acknowledgement Mode 244
12.5.2 Setting the Number of Bits per Transfer 244
12.5.3 Serial Clock 244
12.5.4 Slave Addressing and Address Recognition Mode 245
12.5.5 Configuring the SBI as a Master or a Slave 245
12.5.6 Configuring the SBI as a Transmitter or a Receiver 246
12.5.7 Generating Start and Stop Conditions 247
12.5.8 Interrupt Service Request and Release 248
12.5.9 Serial Bus Interface Operating Modes 248
12.5.10 Lost-arbitration Detection Monitor 248
12.5.11 Slave Address Match Detection Monitor 249
12.5.12 General-call Detection Monitor 249
12.5.13 Last Received Bit Monitor 249
12.5.14 Software Reset 250
12.5.15 Serial Bus Interface Data Buffer Register (SBIxDBR) 250
12.5.16 I2C Bus Address Register (SBIxI2CAR) 250
12.5.17 IDLE Setting Register (SBIxBR0) 250
12.6 Data Transfer Procedure in the I2C Bus Mode 251
12.6.1 Device Initialization 251
12.6.2 Generating the Start Condition and a Slave Address 252
12.6.3 Transferring a Data Word 254
12.6.4 Generating the Stop Condition 259
12.6.5 Restart Procedure 260
12.7 Control in the Clock-synchronous 8-bit SIO Mode 261
12.7.1 Serial Clock 265
12.7.2 Transfer Modes 267

13 Consumer Electronics Control (CEC) 273

13.1 Outline 273
13.1.1 Reception 273
13.1.2 Transmission 273
13.2 Registers 274
13.2.1 Control Registers and Addresses 274

13.2.2 CEC Enable Register [CECEN] 275
13.2.3 Logical Address Register [CECADD] 275
13.2.4 Software Reset Register [CECRESET] 276
13.2.5 Receive Enable Register [CECREN] 276
13.2.6 Receive Buffer Register [CECRBUF] 277
13.2.7 Receive Control Register 1 [CECRCR1] 278
13.2.8 Receive Control Register 2 [CECRCR2] 280
13.2.9 Receive Control Register 3 [CECRCR3] 282
13.2.10 Transmit Enable Register [CECTEN] 284
13.2.11 Transmit Buffer Register [CECTBUF] 285
13.2.12 Transmit Control Register [CECTCR] 286
13.2.13 Receive Interrupt Status Register [CECRSTAT] 288
13.2.14 Transmit Interrupt Status Register [CECTSTAT] 289
13.3 Operations 290
13.3.1 Reception 290
13.3.2 Transmission 301
13.3.3 Software Reset 307

14 Remote control signal preprocessor (RMC) 309

14.1 Basic operation 309
14.1.1 Reception of Remote Control Signal 309
14.2 Registers 310
14.2.1 Register Map 310
14.2.2 Remote Control Enable Register [RMCEN] 311
14.2.3 Remote Control Receive Enable Register [RMCREN] 311
14.2.4 Remote Control Receive Data Buffer Register 1 [RMCRBUF1] 312
14.2.5 Remote Control Receive Data Buffer Register 2 [RMCRBUF2] 313
14.2.6 Remote Control Receive Data Buffer Register 3 [RMCRBUF3] 313
14.2.7 Remote Control Receive Control Register 1 [RMCRCR1] 314
14.2.8 Remote Control Receive Control Register 2 [RMCRCR2] 316
14.2.9 Remote Control Receive Control Register 3 [RMCRCR3] 317
14.2.10 Remote Control Receive Control Register 4 [RMCRCR4] 318
14.2.11 Remote Control Receive Status Register [RMCRSTAT] 319
14.3 Operation Description 320
14.3.1 Reception of Remote Control Signal 320

15 Analog/Digital Converter 333

15.1 Registers 334
15.2 Registers Description 335
15.3 Conversion Clock 347
15.4 Description of Operations 348
15.4.1 Analog Reference Voltage 348
15.4.2 Selecting the Analog Input Channel 348
15.4.3 Starting A/D Conversion 349
15.4.4 A/D Conversion Modes and A/D Conversion Completion Interrupts 351
15.5 High-priority Conversion Mode 354
15.6 A/D Monitor Function 354
15.7 Storing and Reading A/D Conversion Results 354
15.8 Data Polling 355

16 Watchdog Timer (Runaway Detection Timer) 357

16.1 Configuration 357
16.2 Watchdog Timer Interrupt 358
16.3 Control Registers 359
16.3.1 Watchdog Timer Mode Register (WDMOD) 359
16.3.2 Watchdog Timer Control Register (WDCR) 360
16.4 Control Register 362

17 Real Time Clock (RTC) 363

17.1 Functions 363
17.2 Block Diagram 363
17.3 Control Registers 364
17.4 Detailed Description of Control Register 365
17.5 Operational Description 373
17.5.1 Reading clock data 373
17.5.2 Writing clock data 374
17.5.3 Entering the Low Power Consumption Mode 375
17.6 Alarm Function 376

18 Flash Memory Operation 379

18.1 Flash Memory 379
18.1.1 Features 379
18.1.2 Block Diagram of the Flash Memory Section 380
18.2 Operation Mode 381

18.2.1 Reset Operation 382
18.2.2 User Boot Mode (Single chip mode) 383
18.2.3 Single Boot Mode 390
18.3 On-board Programming of Flash Memory (Rewrite/Erase) 420
18.3.1 Flash Memory 420

19 ROM protection 439

19.1 Outline 439
19.2 Features 439
19.2.1 Write/ erase-protection function 439
19.2.2 Security function 439
19.3 Register 440
19.4 Writing and erasing 442
19.4.1 Protection bits 442
19.4.2 Security bit 442

20 Special Function Registers 443

20.1 Addresses 444

21 Electrical Characteristics 459

21.1 Absolute Maximum Ratings 459
21.2 DC Electrical Characteristics (1/2) 461
21.3 DC Electrical Characteristics (2/2) 462
21.4 10-bit ADC Electrical Characteristics 463
21.5 AC Electrical Characteristics 464
21.5.1 Serial Channel Timing (SIO) 464
21.5.2 SBI（I2C） 467
21.5.3 Event Counter 471
21.5.4 Capture 471
21.5.5 General Interrupts 471
21.5.6 NMI, STOP Release Interrupts 472
21.5.7 SCOUT Pin AC Characteristic 472
21.5.8 Debug Communication 473
21.5.9 TRACE Output 474
21.6 Recommended Oscillation Circuit 475
21.7 Handling Precaution 476

22 Port Section Equivalent Circuit Schematic 477

23 Package **483**

RESTRICTIONS ON PRODUCT USE **485**

****
ARM, ARM Powered, AMBA, ADK, ARM9TDMI, TDMI, PrimeCell, RealView, Thumb, Cortex, Coresight, ARM9, ARM926EJ-S, Embedded Trace Macrocell, ETM, AHB, APB, and KEIL are registered trademarks or trademarks of ARM Limited in the EU and other countries.
****

32-bit RISC Microcontroller – TX03 Series

TMPM330FDFG
TMPM330FYFG
TMPM330FWFG

1. Overview and Features

The TX03 series is a 32-bit RISC microcontroller series with an ARM Cortex™-M3 microcontroller core.

Features of the TMPM330FDFG/FYFG/FWFG are as follows:

1.1 Features

(1) ARM Cortex-M3 microcontroller core

1) Improved code efficiency has been realized through the use of Thumb®-2 instruction

- New 16-bit Thumb instructions for improved program flow
- New 32-bit Thumb instructions for improved performance
- Auto-switching between 32-bit instruction and 16-bit instruction is executed by compiler.

2) Both high performance and low power consumption have been achieved.

-High performance

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

-Low power consumption

- Optimized design using a low power consumption library
- Standby function that stops the operation of the microcontroller core

3) High-speed interrupt response suitable for real-time control

- An interruptible long instruction.
- Stack push automatically handled by hardware.

(2) On Chip program memory and data memory

Product name	On chip Flash ROM	On chip RAM
TMPM330FDFG	512Kbyte	32Kbyte
TMPM330FYFG	256Kbyte	16Kbyte
TMPM330FWFG	128Kbyte	8Kbyte

(3) 16-bit timer : 10 channels
- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit PPG output
- Input capture function

(4) Real time clock (RTC) : 1 channel
- Clock (hour, minute and second)
- Calendar (Month, week, date and leap year)
- Time correction ＋ or － 30 seconds (by software)
- Alarm (Alarm output)
- Alarm interrupt

(5) Watchdog timer : 1 channel
- 26 cycles of binary counter
- Watchdog timer out

(6) General-purpose serial interface : 3 channels
- Either UART mode or synchronous mode can be selected (4byte FIFO equipped)

(7) Serial bus interface : 3 channels
- Either I^2C bus mode or synchronous mode can be selected.

(8) CEC : 1 channel
- Transmission and reception per １ byte.

(9) Remote control signal preprocessor : 2 channels
- Can receive up to 72bit data at a time

(10) 10-bit A/D converter : 12 channels
- Start by an internal or external timer trigger
- Fixed channel/scan mode
- Single/repeat mode
- AD monitoring 2ch
- Conversion speed 1.15usec(@fsys = 40MHz)

(11) Interrupt source
- Internal: 42 factors…The order of precedence can be set over 7 levels (except the watchdog timer interrupt).
- External: 8 factors…The order of precedence can be set over 7 levels.

(12) Input/ output ports
- 79 pins

(13) Standby mode

- Standby modes :IDLE, SLOW, SLEEP, STOP
- Sub clock operation(32.768kHz) :SLOW, SLEEP

(14) Clock generator

- On-chip PLL (quadrupled)
- Clock gear function: The high-speed clock can be divided into 1/1, 1/2, 1/4 or 1/8.

(15) Endian

- Little endian

(16) Maximum operating frequency

- 40MHz

(17) Operating voltage range

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

(18) Temperature range

- -20~85 degrees (except during Flash writing/ erasing)
- 0~70 degrees (during Flash writing/ erasing)

(19) Package

- LQFP100-P-1414-0.5H (14mm × 14mm, 0.5mm pitch)

1.2 Block Diagram

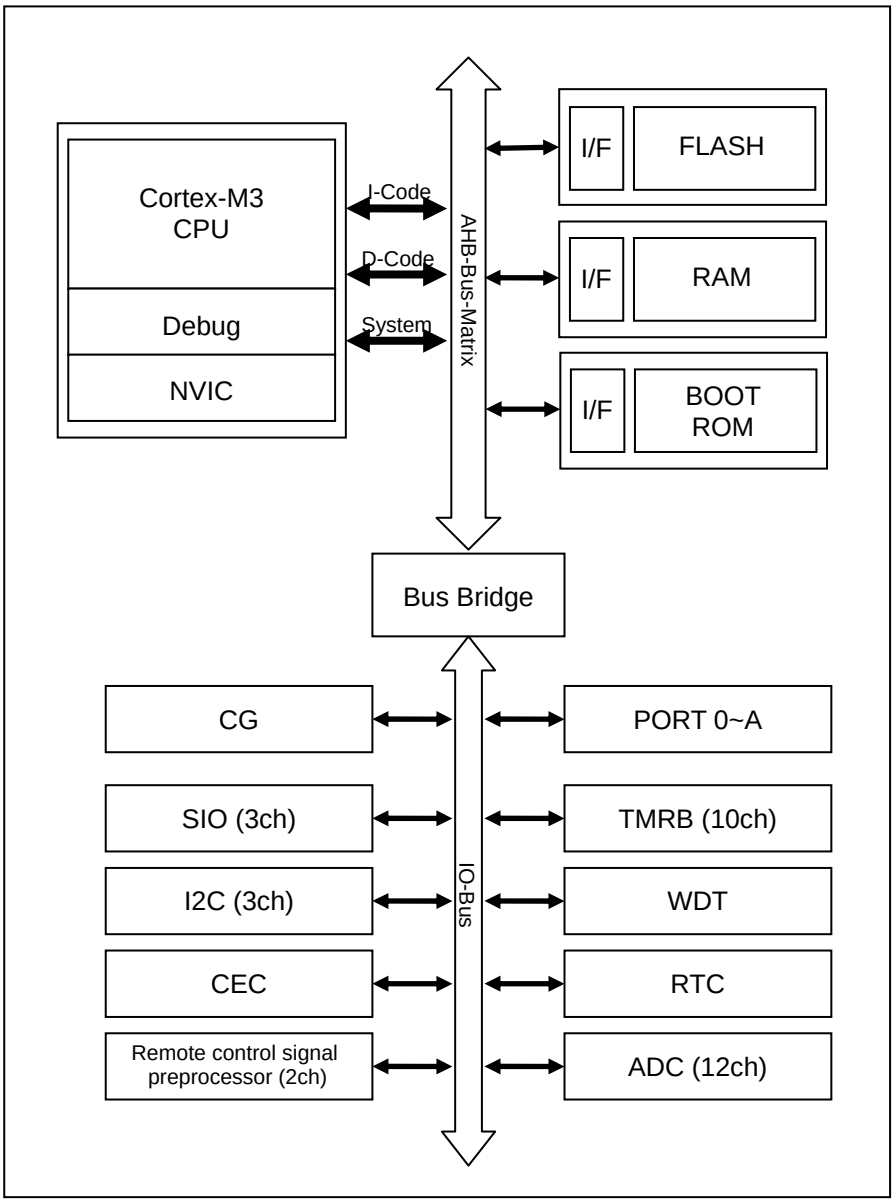


Fig. 1-1 TMPM330 FDFG/FYFG/FWFG Block Diagram

2 Pin Layout and Pin Functions

This chapter describes the pin layout, pin names and pin functions of TMPM330FDFG/ TMPM330FYFG/ TMPM330FWFG.

2.1 Pin Layout (Top view)

Fig.2-1 shows the pin layout of TMPM330FDFG/TMPM330FYFG/TMPM330FWF.

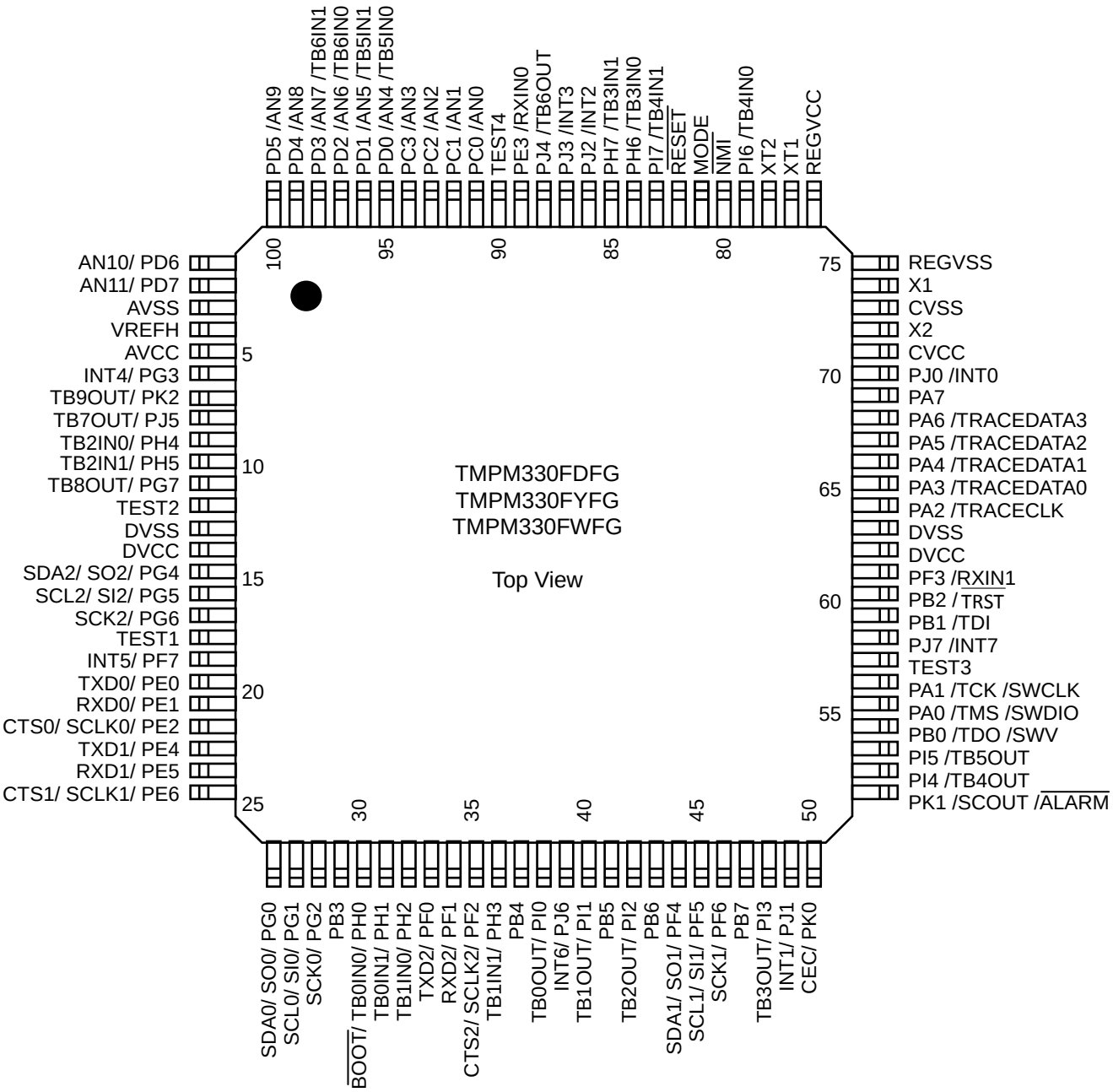


Fig.2-1 Pin Layout (LQFP100)

Table2-1 Pin Numbers and Names (1/2)

Pin No.	Pin name	Pin No.	Pin name
1	PD6, AN10	26	PG0, SO0, SDA0
2	PD7, AN11	27	PG1, SI0, SCL0
3	AVSS	28	PG2, SCK0
4	VREFH	29	PB3
5	AVCC	30	PH0, TB0IN0, $\overline{\text{BOOT}}$
6	PG3, INT4	31	PH1, TB0IN1
7	PK2, TB9OUT	32	PH2, TB1IN0
8	PJ5, TB7OUT	33	PF0, TXD2
9	PH4, TB2IN0	34	PF1, RXD2
10	PH5, TB2IN1	35	PF2, SCLK2, CTS2
11	PG7, TB8OUT	36	PH3, TB1IN1
12	TEST2	37	PB4
13	DVSS	38	PI0, TB0OUT
14	DVCC	39	PJ6, INT6
15	PG4, SO2, SDA2	40	PI1, TB1OUT
16	PG5, SI2, SCL2	41	PB5
17	PG6, SCK2	42	PI2, TB2OUT
18	TEST1	43	PB6
19	PF7, INT5	44	PF4, SO1, SDA1
20	PE0, TXD0	45	PF5, SI1, SCL1
21	PE1, RXD0	46	PF6, SCK1
22	PE2, SCLK0, CTS0	47	PB7
23	PE4, TXD1	48	PI3, TB3OUT
24	PE5, RXD1	49	PJ1, INT1
25	PE6, SCLK1, CTS1	50	PK0, CEC

Table2-1 Pin Numbers and Names (2/2)

Pin No.	Pin name	Pin No.	Pin name
51	PK1, SCOUT, $\overline{\text{ALARM}}$	76	REGVCC
52	PI4, TB4OUT	77	XT1
53	PI5, TB5OUT	78	XT2
54	PB0, TDO, SWV	79	PI6, TB4IN0
55	PA0, TMS, SWDIO	80	$\overline{\text{NMI}}$
56	PA1, TCK, SWCLK	81	MODE
57	TEST3	82	$\overline{\text{RESET}}$
58	PJ7, INT7	83	PI7, TB4IN1
59	PB1, TDI	84	PH6, TB3IN0
60	PB2, $\overline{\text{TRST}}$	85	PH7, TB3IN1
61	PF3, RXIN1	86	PJ2, INT2
62	DVCC	87	PJ3, INT3
63	DVSS	88	PJ4, TB6OUT
64	PA2, TRACECLK	89	PE3, RXIN0
65	PA3, TRACEDATA0	90	TEST4
66	PA4, TRACEDATA1	91	PC0, AN0
67	PA5, TRACEDATA2	92	PC1, AN1
68	PA6, TRACEDATA3	93	PC2, AN2
69	PA7	94	PC3, AN3
70	PJ0, INT0	95	PD0, AN4, TB5IN0
71	CVCC	96	PD1, AN5, TB5IN1
72	X2	97	PD2, AN6, TB6IN0
73	CVSS	98	PD3, AN7, TB6IN1
74	X1	99	PD4, AN8
75	REGVSS	100	PD5, AN9

2.2 Pin names and Functions

Table2-2 and Table2-3 sort the input and output pins of the TMPM330FDFG/ TMPM330FYFG/ TMPM330FWFG by pin or port. Each table includes alternate pin names and functions for multi-function pins.

2.2.1 Sorted by Pin

Table2-2 Pin Names and Functions Sorted by Pin (1/5)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	1	PD6 AN10	I I	Input port Analog input	Pull up	-	-
	2	PD7 AN11	I I	Input port Analog input	Pull up	-	-
PS	3	AVSS	I	A/D converter: GND pin (0V) Tie AVSS to power supply even if the A/D converter is not used.	-	-	-
	4	VREFH	I	Supplying the A/D converter with a reference power supply. Tie VREFH to power supply even if the A/D converter is not used.	-	-	-
	5	AVCC	I	Supplying the A/D converter with a power supply. Tie AVCC to power supply even if the A/D converter is not used.	-	-	-
Function	6	PG3 INT4	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○
	7	PK2 TB9OUT	I/O O	I/O port Timer B output	Pull up	-	-
	8	PJ5 TB7OUT	I/O I	I/O port Timer B output	Pull up	-	-
	9	PH4 TB2IN0	I/O I	I/O port Inputting the timer B capture trigger	Pull up	○	-
	10	PH5 TB2IN1	I/O I	I/O port Inputting the timer B capture trigger	Pull up	○	-
	11	PG7 TB8OUT	I/O O	I/O port Timer B output	Pull up	-	○
Test	12	TEST2	-	TEST pin: Not connected.	-	-	-
PS	13	DVSS	-	GND pin	-	-	-
	14	DVCC	-	Power supply pin	-	-	-
Function	15	PG4 SDA2/ SO2	I/O I/O O	I/O port If the serial bus interface operates -in the I2C mode : data pin -in the SIO mode: data pin	Pull up	○	○
	16	PG5 SCL2/ SI2	I/O I/O I	I/O port If the serial bus interface operates -in the I2C mode : clock pin -in the SIO mode: data pin	Pull up	○	○
	17	PG6 SCK2	I/O I/O	I/O port Inputting and outputting a clock if the serial bus interface operates in the SIO mode.	Pull up	○	○
Test	18	TEST1	-	TEST pin: Not connected.	-	-	-
Function	19	PF7 INT5	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○

Table2-2 Pin Names and Functions Sorted by Pin (2/5)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	20	PE0 TXD0	I/O O	I/O port Sending serial data	Pull up	-	○
	21	PE1 RXD0	I/O I	I/O port Receiving serial data	Pull up	○	○
	22	PE2 SCLK0 CTS0	I/O I I	I/O port Serial clock input/ output Handshake input pin	Pull up	○	○
	23	PE4 TXD1	I/O O	I/O port Sending serial data	Pull up	-	○
	24	PE5 RXD1	I/O I	I/O port Receiving serial data	Pull up	○	○
	25	PE6 SCLK1 CTS1	I/O I I	I/O port Serial clock input/ output Handshake input pin	Pull up	○	○
	26	PG0 SDA0/ SO0	I/O I/O O	I/O port If the serial bus interface operates -in the I2C mode : data pin -in the SIO mode: data pin	Pull up	○	○
	27	PG1 SCL0/ SI0	I/O I/O I	I/O port If the serial bus interface operates -in the I2C mode : clock pin -in the SIO mode: data pin	Pull up	○	○
	28	PG2 SCK0	I/O I/O	I/O port Inputting and outputting a clock if the serial bus interface operates in the SIO mode.	Pull up	○	○
	29	PB3	I/O	I/O port	Pull up	-	-
Function/ Control	30	PH0 TB0IN0 $\overline{\text{BOOT}}$	I/O I I	I/O port Inputting the timer B capture trigger Setting a single boot mode: This pin goes into single boot mode by sampling "L" at the rise of a reset signal.	Pull up	○	-
Function	31	PH1 TB0IN1	I/O I	I/O port Inputting the timer B capture trigger	Pull up	○	-
	32	PH2 TB1IN0	I/O I	I/O port Inputting the timer B capture trigger	Pull up	○	-
	33	PF0 TXD2	I/O O	I/O port Sending serial data	Pull up	-	○
	34	PF1 RXD2	I/O I	I/O port Receiving serial data	Pull up	○	○
	35	PF2 SCLK2 CTS2	I/O I I	I/O port Serial clock input/ output Handshake input pin	Pull up	○	○
	36	PH3 TB1IN1	I/O I	I/O port Inputting the timer B capture trigger	Pull up	○	-
	37	PB4	I/O	I/O port	Pull up	-	-
	38	PI0 TB0OUT	I/O O	I/O port Timer B output	Pull up	-	-
	39	PJ6 INT6	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○

Table2-2 Pin Names and Functions Sorted by Pin (3/5)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	40	PI1 TB1OUT	I/O O	I/O port Timer B output	Pull up	-	-
	41	PB5	I/O	I/O port	Pull up	-	-
	42	PI2 TB2OUT	I/O O	I/O port Timer B output	Pull up	-	-
	43	PB6	I/O	I/O port	Pull up	-	-
	44	PF4 SDA1/ SO1	I/O I/O O	I/O port If the serial bus interface operates -in the I2C mode : data pin -in the SIO mode: data pin	Pull up	○	○
	45	PF5 SCL1/ SI1	I/O I/O I	I/O port If the serial bus interface operates -in the I2C mode : clock pin -in the SIO mode: data pin	Pull up	○	○
	46	PF6 SCK1	I/O I/O	I/O port Inputting and outputting a clock if the serial bus interface operates in the SIO mode.	Pull up	○	○
	47	PB7	I/O	I/O port	Pull up	-	-
	48	PI3 TB3OUT	I/O O	I/O port Timer B output	Pull up	-	-
	49	PJ1 INT1	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○
	50	PK0 CEC	I/O I/O	I/O port CEC pin	-	○	● (Note 4)
	51	PK1 SCOUT $\overline{\text{ALARM}}$	I/O O O	I/O port System clock output Alarm output	Pull up	-	-
	52	PI4 TB4OUT	I/O O	I/O port Timer B output	Pull up	-	-
	53	PI5 TB5OUT	I/O O	I/O port Timer B output	Pull up	-	-
Function/ Debug	54	PB0 TDO/SWV	I/O O	I/O port Debug pin	Pull up	-	-
	55	PA0 TMS/SWDIO	I/O I/O	I/O port Debug pin	Pull up	○	-
	56	PA1 TCK/ SWCLK	I/O I	I/O port Debug pin	Pull up	-	-
Test	57	TEST3	-	TEST pin: Not connected.	-	-	-
Function	58	PJ7 INT7	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○
Function/ Debug	59	PB1 TDI	I/O I	I/O port Debug pin	Pull up	-	-
	60	PB2 $\overline{\text{TRST}}$	I/O I	I/O port Debug pin	Pull up	○	-
Function	61	PF3 RXIN1	I/O I	I/O port Inputting signal to remote controller	Pull up	○	○
PS	62	DVCC	-	Power supply pin	-	-	-
	63	DVSS	-	GND pin	-	-	-

Table2-2 Pin Names and Functions Sorted by Pin (4/5)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function/ Debug	64	PA2 TRACECLK	I/O O	I/O port Debug pin	Pull up	-	-
	65	PA3 TRACEDATA0	I/O O	I/O port Debug pin	Pull up	-	-
	66	PA4 TRACEDATA1	I/O O	I/O port Debug pin	Pull up	-	-
	67	PA5 TRACEDATA2	I/O O	I/O port Debug pin	Pull up	-	-
	68	PA6 TRACEDATA3	I/O O	I/O port Debug pin	Pull up	-	-
Function	69	PA7	I/O	I/O port	Pull up		
	70	PJ0 INT0	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○
PS	71	CVCC	-	Power supply pin	-	-	-
Clock	72	X2	O	Connected to a high-speed oscillator.	-	-	-
PS	73	CVSS	-	GND pin	-	-	-
Clock	74	X1	I	Connected to a high-speed oscillator.	-	○	-
PS	75	REGVSS	-	GND pin	-	-	-
	76	REGVCC	-	Power supply pin	-	-	-
Clock	77	XT1	I	Connected to a low-speed oscillator.	-	○	-
	78	XT2	O	Connected to a low-speed oscillator.	-	-	-
Function	79	PI6 TB4IN0	I/O I	I/O port Inputting the timer B capture trigger	Pull up	○	-
	80	$\overline{\text{NMI}}$	I	Non-maskable interrupt	-	○ w/ noise filter	-
Control	81	MODE	I	Mode pin: Tied to GND pin	-	○	-
Function	82	$\overline{\text{RESET}}$	I	Reset input pin	Tied to Pull up	○ w/ noise filter	-
	83	PI7 TB4IN0	I/O I	I/O port Inputting the timer B capture trigger	Pull up	○	-
	84	PH6 TB3IN0	I/O I	I/O port Inputting the timer B capture trigger	Pull up	○	-
	85	PH7 TB3IN1	I/O I	I/O port Inputting the timer B capture trigger	Pull up	○	-
	86	PJ2 INT2	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○
	87	PJ3 INT3	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○
	88	PJ4 TB6OUT	I/O O	I/O port Timer B output	Pull up	-	-
	89	PE3 RXIN0	I/O I	I/O port Inputting signal to remote controller	Pull up	○	○
Test	90	TEST4	-	TEST pin: Not connected.	-	-	-

Table2-2 Pin Names and Functions Sorted by Pin (5/5)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	91	PC0 AN0	I I	Input port Analog input	Pull up	-	-
	92	PC1 AN1	I I	Input port Analog input	Pull up	-	-
	93	PC2 AN2	I I	Input port Analog input	Pull up	-	-
	94	PC3 AN3	I I	Input port Analog input	Pull up	-	-
	95	PD0 AN4 TB5IN0	I I I	Input port Analog input Inputting the timer B capture trigger	Pull up	-	-
	96	PD1 AN5 TB5IN1	I I I	Input port Analog input Inputting the timer B capture trigger	Pull up	-	-
	97	PD2 AN6 TB6IN0	I I I	Input port Analog input Inputting the timer B capture trigger	Pull up	-	-
	98	PD3 AN7 TB6IN1	I I I	Input port Analog input Inputting the timer B capture trigger of	Pull up	-	-
	99	PD4 AN8	I I	Input port Analog input	Pull up	-	-
	100	PD5 AN9	I I	Input port Analog input	Pull up	-	-

(Note 1) TEST1 through 4 must be left unconnected.
(Note 2) Be sure to tie MODE to GND.
(Note 3) Tie VREFH/ AVCC to power supply and AVSS to GND even if the A/D converter is not used.
(Note 4) Nch open drain port.
(Note 5) The noise elimination width of the noise filter is approximately 30 ns under typical conditions.

2.2.2 Sorted by Port

Table2-3 Pin Names and Functions Sorted by Port (1/5)

PORT	Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
A	Function / Debug	55	PA0 TMS/SWDIO	I/O I/O	I/O port Debug pin	P-up	○	-
		56	PA1 TCK/SWCLK	I/O I	I/O port Debug pin	P-up	-	-
		64	PA2 TRACECLK	I/O O	I/O port Debug pin	P-up	-	-
		65	PA3 TRACEDATA0	I/O O	I/O port Debug pin	P-up	-	-
		66	PA4 TRACEDATA1	I/O O	I/O port Debug pin	P-up	-	-
		67	PA5 TRACEDATA2	I/O O	I/O port Debug pin	P-up	-	-
		68	PA6 TRACEDATA3	I/O O	I/O port Debug pin	P-up	-	-
	Function	69	PA7	I/O	I/O port	P-up		
B	Function / Debug	54	PB0 TDO/SWV	I/O O	I/O port Debug pin	P-up	-	-
		59	PB1 TDI	I/O I	I/O port Debug pin	P-up	-	-
		60	PB2 $\overline{\text{TRST}}$	I/O I	I/O port Debug pin	P-up	○	-
	Function	29	PB3	I/O	I/O port	P-up	-	-
		37	PB4	I/O	I/O port	P-up	-	-
		41	PB5	I/O	I/O port	P-up	-	-
		43	PB6	I/O	I/O port	P-up	-	-
		47	PB7	I/O	I/O port	P-up	-	-
C	Function	91	PC0 AN0	I I	Input port Analog input	P-up	-	-
		92	PC1 AN1	I I	Input port Analog input	P-up	-	-
		93	PC2 AN2	I I	Input port Analog input	P-up	-	-
		94	PC3 AN3	I I	Input port Analog input	P-up	-	-
D	Function	95	PD0 AN4 TB5IN0	I I I	Input port Analog input Inputting the timer B capture trigger	P-up	-	-
		96	PD1 AN5 TB5IN1	I I I	Input port Analog input Inputting the timer B capture trigger	P-up	-	-
		97	PD2 AN6 TB6IN0	I I I	Input port Analog input Inputting the timer B capture trigger	P-up	-	-

Table2-3 Pin Names and Functions Sorted by Port (2/5)

PORT	Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
D	Function	98	PD3 AN7 TB6IN1	I I I	Input port Analog input Inputting the timer B capture trigger	P-up	-	-
		99	PD4 AN8	I I	Input port Analog input	P-up	-	-
		100	PD5 AN9	I I	Input port Analog input	P-up	-	-
		1	PD6 AN10	I I	Input port Analog input	P-up	-	-
		2	PD7 AN11	I I	Input port Analog input	P-up	-	-
E	Function	20	PE0 TXD0	I/O O	I/O port Sending serial data	P-up	-	○
		21	PE1 RXD0	I/O I	I/O port Receiving serial data	P-up	○	○
		22	PE2 SCLK0 CTS0	I/O I I	I/O port Serial clock input/ output Handshake input pin	P-up	○	○
		89	PE3 RXIN0	I/O I	I/O port Inputting signal to remote controller	P-up	○	○
		23	PE4 TXD1	I/O O	I/O port Sending serial data	P-up	-	○
		24	PE5 RXD1	I/O I	I/O port Receiving serial data	P-up	○	○
		25	PE6 SCLK1 CTS1	I/O I I	I/O port Serial clock input/ output Handshake input pin	P-up	○	○
F	Function	33	PF0 TXD2	I/O O	I/O port Sending serial data	P-up	-	○
		34	PF1 RXD2	I/O I	I/O port Receiving serial data	P-up	○	○
		35	PF2 SCLK2 CTS2	I/O I I	I/O port Serial clock input/ output Handshake input pin	P-up	○	○
		61	PF3 RXIN1	I/O I	I/O port Inputting signal to remote controller	P-up	○	○
		44	PF4 SDA1/ SO1	I/O I/O O	I/O port If the serial bus interface operates -in the I2C mode : data pin -in the SIO mode: data pin	P-up	○	○
		45	PF5 SCL1/ SI1	I/O I/O I	I/O port If the serial bus interface operates -in the I2C mode : clock pin -in the SIO mode: data pin	P-up	○	○
		46	PF6 SCK1	I/O I/O	I/O port Inputting and outputting a clock if the serial bus interface operates in the SIO mode.	P-up	○	○
		19	PF7 INT5	I/O I	I/O port Interrupt request pin	P-up	○ w/ noise filter	○

Table2-3 Pin Names and Functions Sorted by Port (3/5)

PORT	Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
G	Function	26	PG0 SDA0/ SO0	I/O I/O O	I/O port If the serial bus interface operates -in the I2C mode : data pin -in the SIO mode: data pin	P-up	○	○
		27	PG1 SCL0/ SI0	I/O I/O I	I/O port If the serial bus interface operates -in the I2C mode : clock pin -in the SIO mode: data pin	P-up	○	○
		28	PG2 SCK0	I/O I/O	I/O port Inputting and outputting a clock if the serial bus interface operates in the SIO mode.	P-up	○	○
		6	PG3 INT4	I/O I	I/O port Interrupt request pin	P-up	○ w/ noise filter	○
		15	PG4 SDA2/ SO2	I/O I/O O	I/O port If the serial bus interface operates -in the I2C mode : data pin -in the SIO mode: data pin	P-up	○	○
		16	PG5 SCL2/ SI2	I/O I/O I	I/O port If the serial bus interface operates -in the I2C mode : clock pin -in the SIO mode: data pin	P-up	○	○
		17	PG6 SCK2	I/O I/O	I/O port Inputting and outputting a clock if the serial bus interface operates in the SIO mode.	P-up	○	○
		11	PG7 TB8OUT	I/O O	I/O port Timer B output	P-up	-	○
H	Function/ Control	30	PH0 TB0IN0 $\overline{\text{BOOT}}$	I/O I I	I/O port Inputting the timer B capture trigger Setting a single boot mode: This pin goes into single boot mode by sampling "L" at the rise of a reset signal.	P-up	○	-
	Function	31	PH1 TB0IN1	I/O I	I/O port Inputting the timer B capture trigger	P-up	○	-
		32	PH2 TB1IN0	I/O I	I/O port Inputting the timer B capture trigger	P-up	○	-
		36	PH3 TB1IN1	I/O I	I/O port Inputting the timer B capture trigger	P-up	○	-
		9	PH4 TB2IN0	I/O I	I/O port Inputting the timer B capture trigger	P-up	○	-
		10	PH5 TB2IN1	I/O I	I/O port Inputting the timer B capture trigger	P-up	○	-
		84	PH6 TB3IN0	I/O I	I/O port Inputting the timer B capture trigger	P-up	○	-
		85	PH7 TB3IN1	I/O I	I/O port Inputting the timer B capture trigger	P-up	○	-

Table2-3 Pin Names and Functions Sorted by Port Number (4/5)

PORT	Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
I	Function	38	PI0 TB0OUT	I/O O	I/O port Timer B output	P-up	-	-
		40	PI1 TB1OUT	I/O O	I/O port Timer B output	P-up	-	-
		42	PI2 TB2OUT	I/O O	I/O port Timer B output	P-up	-	-
		48	PI3 TB3OUT	I/O O	I/O port Timer B output	P-up	-	-
		52	PI4 TB4OUT	I/O O	I/O port Timer B output	P-up	-	-
		53	PI5 TB5OUT	I/O O	I/O port Timer B output	P-up	-	-
		79	PI6 TB4IN0	I/O I	I/O port Inputting the timer B capture trigger	P-up	○	-
		83	PI7 TB4IN0	I/O I	I/O port Inputting the timer B capture trigger	P-up	○	-
J	Function	70	PJ0 INT0	I/O I	I/O port Interrupt request pin	P-up	○ w/ noise filter	○
		49	PJ1 INT1	I/O I	I/O port Interrupt request pin	P-up	○ w/ noise filter	○
		86	PJ2 INT2	I/O I	I/O port Interrupt request pin	P-up	○ w/ noise filter	○
		87	PJ3 INT3	I/O I	I/O port Interrupt request pin	P-up	○ w/ noise filter	○
		88	PJ4 TB6OUT	I/O O	I/O port Timer B output	P-up	-	-
		8	PJ5 TB7OUT	I/O I	I/O port Timer B output	P-up	-	-
		39	PJ6 INT6	I/O I	I/O port Interrupt request pin	P-up	○ w/ noise filter	○
		58	PJ7 INT7	I/O I	I/O port Interrupt request pin	P-up	○ w/ noise filter	○
K	Function	50	PK0 CEC	I/O I/O	I/O port CEC pin	-	○	● (Note 4)
		51	PK1 SCOUT $\overline{\text{ALARM}}$	I/O O O	I/O port System clock output Alarm output	P-up	-	-
		7	PK2 TB9OUT	I/O O	I/O port Timer B output	P-up	-	-

Table2-3 Pin Names and Functions Sorted by Port (5/5)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	82	$\overline{\text{RESET}}$	I	Reset input pin	Tied to Pull up	○ w/ noise filter	-
	80	$\overline{\text{NMI}}$	I	Non-maskable interrupt	-	○ w/ noise filter	-
Control	81	MODE	I	Mode pin: Tied to GND pin	-	○	-
Clock	72	X2	O	Connected to a high-speed oscillator.	-	-	-
	74	X1	I	Connected to a high-speed oscillator.	-	○	-
	77	XT1	I	Connected to a low-speed oscillator.	-	○	-
	78	XT2	O	Connected to a low-speed oscillator.	-	-	-
Test	12	TEST2	-	TEST pin: Not coneected.	-	-	-
	18	TEST1	-	TEST pin: Not connected.	-	-	-
	57	TEST3	-	TEST pin: Not connected.	-	-	-
	90	TEST4	-	TEST pin: Not connected.	-	-	-
PS	3	AVSS	I	A/D converter: GND pin (0V) Tie AVSS to power supply even if the A/D converter is not used.	-	-	-
	4	VREFH	I	Supplying the A/D converter with a reference power supply. Tie VREFH to power supply even if the A/D converter is not used.	-	-	-
	5	AVCC	I	Supplying the A/D converter with a power supply. Tie AVCC to power supply even if the A/D converter is not used.	-	-	-
	13	DVSS	-	GND pin	-	-	-
	14	DVCC	-	Power supply pin	-	-	-
	62	DVCC	-	Power supply pin	-	-	-
	63	DVSS	-	GND pin	-	-	-
	71	CVCC	-	Power supply pin	-	-	-
	73	CVSS	-	GND pin	-	-	-
	75	REGVSS	-	GND pin	-	-	-
	76	REGVCC	-	Power supply pin	-	-	-

(Note 1) TEST1 through 4 must be left unconnected.
(Note 2) Be sure to tie MODE to GND.
(Note 3) Tie VREFH/ AVCC to power supply and AVSS to GND even if the A/D converter is not used.
(Note 4) Nch open drain port.
(Note 5) The noise elimination width of the noise filter is approximately 30 ns under typical conditions.

2.3 Pin Names and Power Supply Pins

Table2-4 Pin Names and Power Supplies

Pin name	Power supply
PA	DVCC
PB	DVCC
PC	AVCC
PD	AVCC
PE	DVCC
PF	DVCC
PG	DVCC
PH	DVCC
PI	DVCC
PJ	DVCC
PK	DVCC
X1, X2	CVCC
XT1, XT2	DVCC
$\overline{\text{RESET}}$	DVCC
$\overline{\text{NMI}}$	DVCC
MODE	DVCC

2.4 Pin Numbers and Power Supply Pins

Table2-5 Pin Numbers and Power Supplies

Power supply	Pin number	Voltage range
DVCC	14, 62	2.7V~3.6V
AVCC	5	
REGVCC	76	
CVCC	71	

3 Processor Core and Memory Map

3.1 Processor Core

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

The following table shows the revision of the processor core in the TMPM330FDFG/ TMPM330FYFG/ TMPM330FWFG. For further information on each revision, see the documents issued by ARM Limited.

Product Name	Core Revision
TMPM330FDFG TMPM330FYFG	r1p1-00rel0
TMPM330FWFG	r1p1-01rel0

The Cortex-M3 core has the optional blocks. The optional blocks of the revision r1p1 are ETM™ and MPU. Not MPU but ETM is contained in the TMPM330FDFG/ TMPM330FYFG/ TMPM330FWFG.

4 Debug Interface

4.1 Specification Overview

The TMPM330 contains the Serial Wire JTAG Debug Port (SWJ-DP) unit for interfacing with the In-Circuit Emulator (ICE) and the Embedded Trace Macrocell™ (ETM) unit for instruction trace output. Trace data is output to the dedicated pins (TRACEDATA[0]-[3], SWV) via the on-chip Trace Port Interface Unit (TPIU).

For details about SWJ-DP, ETM and TPIU, refer to “Cortex-M3 Technical Reference Manual” .

4.2 Features of SWJ-DP

SWJ-DP supports the two-pin Serial Wire Debug Port (SWDCK, SWDIO) and the JTAG Debug Port (TDI, TDO, TMS, TCK, TRST).

4.3 Features of ETM

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

4.4 Pin Functions

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

The PA0 and PA1 pins are shared between the JTAG debug port function and the serial wire debug port function. The PB0 pin is shared between the JTAG debug port function and the SWV trace output function.

After reset, the PA0, PA1, PB0, PB1 and PB2 pins are configured as debug port function pins. The functions of other debug interface pins need to be programmed as required.

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

(Note 1)	**If PA0 and PB0 are configured as TMS/SWDIO and TDO/SWV, output continues to be enabled even in STOP mode regardless of the setting of the STBYCR2<DRVE> bit .**
(Note 2)	**If PA1 is configured as a debug function pin, it prevents a low power consumption mode from being fully effective. Configure PA1 to function as a general-purpose port if the debug function is not used.**

The table below summarizes the debug interface pin functions and related port settings after reset.

Pin Number	Port (Bit Name)	Debug Function	Port Settings after Reset				
			Function (PxFR)	Input (PxIE)	Output (PxCR)	Pull-up (PxPUP)	Pull-down (PxPDN)
55	PA0	TMS/SWDIO	○	○	○	○	-
56	PA1	TCK/SWCLK	○	○	×	-	○
54	PB0	TDO/SWV	○	×	○	×	-
59	PB1	TDI	○	○	×	○	-
60	PB2	TRST	○	○	×	○	-
64	PA2	TRCECLK	×	×	×	×	-
65	PA3	TRACEDATA0	×	×	×	×	-
66	PA4	TRACEDATA1	×	×	×	×	-
67	PA5	TRACEDATA2	×	×	×	×	-
68	PA6	TRACEDATA3	×	×	×	×	-

○: Enabled ×: Disabled

4.5 Connection with a Debug Tool

For how to connect a debug tool, refer to the method recommended by each manufacturer.

5 Memory Map

The memory maps for the TMPM330FDFG/ TMPM330FYFG/ TMPM330FWFG are based on the ARM Cortex-M3 processor core memory map. The internal ROM, internal RAM and internal I/O of the TMPM330FDFG/ TMPM330FYFG/ TMPM330FWFG are mapped to the code, SRAM and peripheral regions of the Cortex-M3 respectively. The SRAM and internal I/O regions are all included in the bit-band region.

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

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

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

See Chapter 20 "Special Function Registers" for details on the internal I/O region.

5.1 Memory Map of the TMPM330FDFG

Fig 5-1 shows the memory map of the TMPM330FDFG.

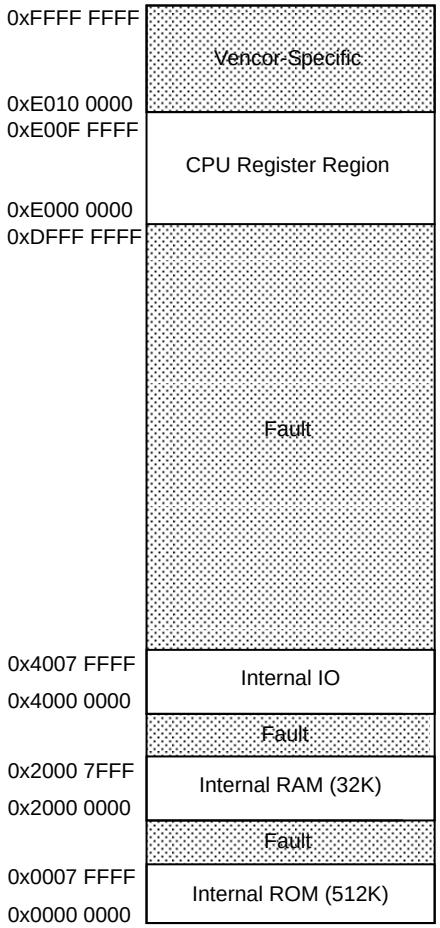


Fig 5-1 Memory Map (TMPM330FDFG)

5.2 Memory Map of the TMPM330FYFG

Fig. 5-2 shows the memory map of the TMPM330FYFG.

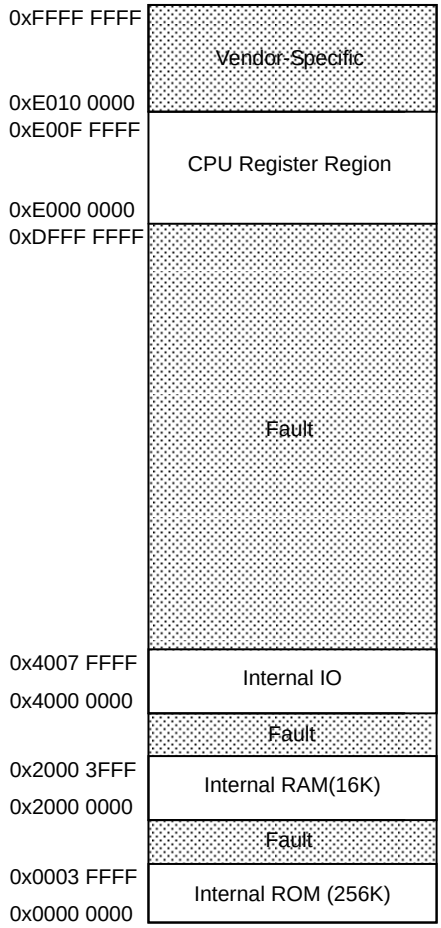


Fig. 5-2 Memory Map (TMPM330FYFG)

(Note) In addition to 256KB flash area, the TMPM330FYFG provides 128-word data/ password area (1 page) for Show Product Information command in the address range 0x0007_FF00 - 0x0007_FF80. See Chapter 16 "Flash Memory Operation" for details on the single boot mode.
Access to the range from 0x0008_0000 through the password area does not cause a hard fault. Writing to the range is ignored. Reading the range yields undefined values.

5.3 Memory Map of the TMPM330FWFG

Fig. 5-3 shows the memory map of the TMPM330FWFG.

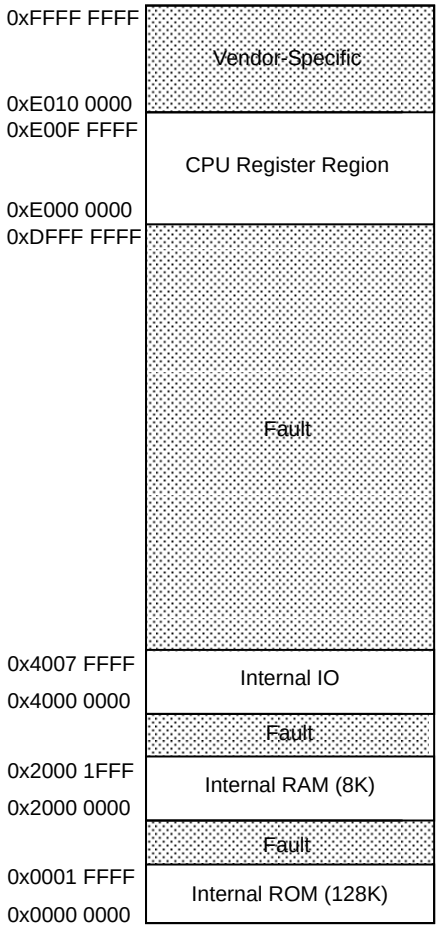


Fig. 5-3 Memory Map (TMPM330FWFG)

6 Reset

The TMPM330 has three reset sources: an external reset pin, WDT and SYSRESETREQ. For reset from the WDT, refer to the chapter on the WDT. For reset from SYSRESETREQ, refer to “Cortex-M3 Technical Reference Manual”.

6 Cold Reset

The power-on sequence must include the time for the internal regulator and oscillator to be stable. In the TMPM330, the internal regulator requires at least 700 μs to be stable. The time required to achieve stable oscillation varies with system. At cold reset, the external reset pin must be kept low for a duration of time sufficiently long enough for the internal regulator and oscillator to be stable.

Fig. 6-1 shows the power-on sequence.

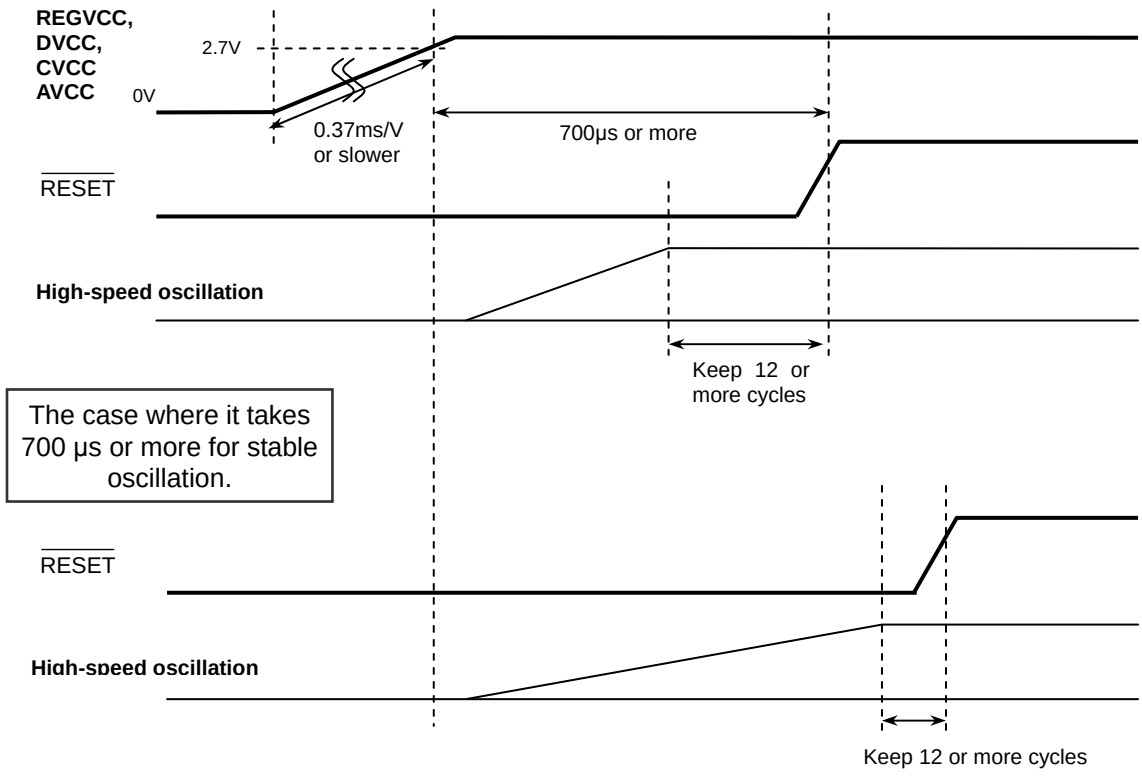


Fig. 6-1 Cold Reset Sequence

(Note 1)	**The power supply must be raised (from 0V to 2.7V) at a speed of 0.37ms/V or slower.**
(Note 2)	**Turn on the power while the $\overline{\text{RESET}}$ pin is fixed to “L”. Release the $\overline{\text{RESET}}$ pin while all the power supplies are stabilized within operating voltage.**

6.1 Warm Reset

6.2.1 Reset Period

As a precondition, ensure that the power supply voltage is within the operating range and the internal high-frequency oscillator is providing stable oscillation. To reset the TMPM330, assert the $\overline{RESET}$ signal (active low) for a minimum duration of 12 system clocks.

6.2.2 After Reset

A warm reset initializes the majority of the Cortex-M3 processor core's system control registers and internal I/O registers. Registers that are only initialized by a cold reset are registers related to the processor core's system debug components (FPB, DWT, ITM), the clock generator's reset flag and the Flash security bit.

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

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

(Note 1) The reset operation may alter the internal RAM state.

7 Clock/Mode Control

7.1 Features

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

The low power consumption mode can reduce power consumption.

This chapter describes how to control clocks, clock operating modes and mode transitions.

The clock/mode control block has the following functions:

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

In addition to NORMAL mode, the TMPM330 can operate in three types of low power mode to reduce power consumption according to its usage conditions.

7.2 Registers

7.2.1 Register List

Table 7-1 shows registers and addresses of the clock generator.

Table 7-1 Registers of Clock Generator

Register name		Address
System control register 0	SYSCR0	0x4004_0200
System control register 1	SYSCR1	0x4004_0201
System control register 2	SYSCR2	0x4004_0202
Oscillation control register 0	OSCCR0	0x4004_0204
Oscillation control register 1	OSCCR1	0x4004_0205
Standby control register 0	STBYCR0	0x4004_0208
Standby control register 1	STBYCR1	0x4004_0209
Standby control register 2	STBYCR2	0x4004_020A
PLL selection register	PLLSEL	0x4004_020C
System clock selection register	CKSEL	0x4004_0210

7.2.2 Detailed Description of Registers

7.2.2.1 System Control Register

		7	6	5	4	3	2	1	0
SYSCR0	Bit symbol	-	-	-	-	-	GEAR2	GEAR1	GEAR0
	Read/Write	R					R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	"0" is read.					High-speed clock (fc) gear 000: fc 100: fc/2 001: reserved 101: fc/4 010: reserved 110: fc/8 011: reserved 111: reserved		
		15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SYSCR1	Bit symbol	-	-	-	FPSEL	-	PRCK2	PRCK1	PRCK0
	Read/Write	R			R/W	R	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	"0" is read.			fperiph 0:fgear 1:fc	"0" is read.	Prescaler clock 000: fperiph 100: fperiph/16 001: fperiph/2 101: fperiph/32 010: fperiph/4 110: Reserved 011: fperiph/8 111: Reserved		
		23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SYSCR2	Bit symbol	-	-	-	-	-	--	SCOSEL1	SCOSEL0
	Read/Write	R						R/W	R/W
	After reset	0	0	0	0	0	0	0	1
	Function	"0" is read.						SCOUT output 00: fs 01: fsys/2 10: fsys 11: φT0	
		31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
	Bit symbol	-	-	-	-	-	-	-	-
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	"0" is read.							

<Bit 2:0><GEAR 2:0> : Specifies the high-speed clock (fc) gear.

<Bit 10:8><PRCK 2:0> : Specifies the prescaler clock to peripheral I/O.

<Bit 12><FPSEL> : Specifies the source clock to fperiph.

<Bit 17:16><SCOSEL1:0> : Enables to output the specified clock from SCOUT pin.

7.2.2.2 Oscillation Control Register

OSCCR0

	7	6	5	4	3	2	1	0
Bit symbol	-	WUPT2	WUPT1	WUPT0	WUPSEL	PLLON	WUEF	WUEON
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R	W
After reset	0	0	0	1	0	0	0	0
Function	"0" is read.	Selection of warm-up time X1 000: No warm-up 001:2^{10}/input freq. 010:2^{11}/ input freq. 011:2^{12}/ input freq. 100:2^{13}/ input freq. 101:2^{14}/ input freq. 110:2^{15}/ input freq. 111:2^{16}/ input freq.	XT1 000:No warm-up 001:2^{6}/ input freq. 010:2^{7}/ input freq. 011:2^{8}/ input freq. 100:2^{15}/ input freq. 101:2^{16}/ input freq. 110:2^{17}/ input freq. 111:2^{18}/ input freq.		Warm-up counter 0: X1 1: XT1	PLL operation 0: Stop 1: Oscillation	Status of Warm-up timer (WUP) 0: warm-up completed 1: Warm-up in operation	Operation of warm-up timer (WUP) 0: don't care 1: starting warm-up

OSCCR1

	15	14	13	12	11	10	9	8
Bit symbol	-	-	-	-	-	-	XTEN	XEN
Read/Write	R		R/W		R		R/W	R/W
After reset	0	0	0	0	0	0	1	1
Function	"0" is read.		Write "0".		"0" is read.		Low-speed oscillator 0: Stop 1: Oscillation	High-speed oscillator 0: Stop 1: Oscillation

	23	22	21	20	19	18	17	16
Bit symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

<Bit 0><WUEON> : Enables to start the warm-up timer.

<Bit 1><WUEF> : Enables to monitor the status of the warm-up timer.

<Bit 2><PLLON> : Specifies operation of the PLL.
It stops after reset. Setting the bit is required.

<Bit 3><WUPSEL> : Specifies the oscillator to warm-up. A clock generated by the specified oscillator is used for the warm-up timer count.

<Bit 6:4><WUPT2:0> : Specifies time of the warm-up timer.

<Bit 8><XEN> : Specifies operation of the high-speed oscillator.

<Bit 9><XTEN> : Specifies operation of the low-speed oscillator.

7.2.2.3 Standby Control Register

		7	6	5	4	3	2	1	0
STBYCR0	Bit symbol	-	-	-	-	-	STBY2	STBY1	STBY0
	Read/Write	R					R/W	R/W	R/W
	After reset	0	0	0	0	0	0	1	1
	Function	"0" is read.					Low power consumption mode 000: Reserved 001: STOP 010: SLEEP 011: IDLE 100: Reserved 101: Reserved 110: Reserved 111: Reserved		
		15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
STBYCR1	Bit symbol	-	-	-	-	-	-	RXTEN	RXEN
	Read/Write	R						R/W	R/W
	After reset	0	0	0	0	0	0	0	1
	Function	"0" is read.						Low-speed oscillator after releasing STOP mode 0: Stop 1: Oscillation	High-speed oscillator after releasing STOP mode 0: Stop 1: Oscillation
		23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
STBYCR2	Bit symbol	-	-	-	-	-	-	-	DRVE
	Read/Write	R						R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	"0" is read.						Write "0".	Pin status in STOP mode 0: Active 1: Inactive
		31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
	Bit symbol	-	-	-	-	-	-	-	-
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	"0" is read.							

<Bit 2:0><STBY2:0> : Specifies the low power consumption mode.

<Bit 8><RXEN> : Specifies the high-speed oscillator operation after releasing the STOP mode.

<Bit 9><RXTEN> : Specifies the low-speed oscillator operation after releasing the STOP mode.

<Bit 16><DRVE> : Specifies the pin status in the STOP mode.

7.2.2.4 PLL Selection Register

PLLSEL

	7	6	5	4	3	2	1	0
Bit symbol	-	-	-	-	-	-	-	PLLSEL
Read/Write	R							R/W
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							Use of PLL 0: Disuse. X1 selected 1: Use

	15	14	13	12	11	10	9	8
Bit symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

	23	22	21	20	19	18	17	16
Bit symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

<Bit 0><PLLSEL> : Specifies use or disuse of the clock multiplied by the PLL.
"X1" is automatically set after reset. Resetting is required when using the PLL.

7.2.2.5 System Clock Selection Register

CKSEL

	7	6	5	4	3	2	1	0
Bit symbol	-	-	-	-	-	-	SYSCK	SYSCK FLG
Read/Write	R						R/W	R
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.						System clock 0: High-speed (fc) 1: Low-speed (fs)	System clock status 0: High-speed (fc) 1: Low-speed (fs) Stable oscillation identical with <SYSCK> value.

	15	14	13	12	11	10	9	8
Bit symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

	23	22	21	20	19	18	17	16
Bit symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

	31	30	29	28	27	26	25	24
Bit symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

<Bit 0><SYSCKFLG> : Shows the status of the system clock.
Switching the oscillator with <SYSCK> generates time lag to complete. If the output of the oscillator specified in <SYSCK> is read out by <SYSCLKFLG>, the switching has been completed.

<Bit 1><SYSCK> : Enables to specify the system clock. Setting OSCCR1<XEN> and <XTEN> to "1" in advance is required.

7.3 Clock Control

7.3.1 Clock System Block Diagram

Fig.7-1 shows the clock system diagram. Each clock is defined as follows.

fosc : Clock input from the X1 and X2 pins
fs : Clock input from the XT1 and XT2 (low-speed clock)
fpll : Clock quadrupled by PLL
fc : Clock specified by PLLSEL<PLLSEL> (high-speed clock)
fgear : Clock specified by SYSCR1<GEAR2:0>
fsys : Clock specified by CKSEL<SYSCK> (system clock)
fperiph : Clock specified by SYSCR1<FPSEL2:0>
ΦT0 : Clock specified by SYSCR1<PRCK2:0> (prescaler clock)

The high-speed clock fc and the prescaler clock ΦT0 are dividable.

- High-speed clock: fc, fc/2, fc/4, fc/8
- Prescaler clock: fperiph, fperiph/2, fperiph/4, fperiph/8, fperiph/16,fperiph/32

7.3.2 Initial Values after Reset

Reset initializes the clock configuration as follows.

High-speed oscillator	: ON (oscillating)
Low-speed oscillator	: ON (oscillating)
PLL (phase locked loop circuit)	: OFF (stop)
High-speed clock gear	: fc (no frequency dividing)

Reset causes all the clock configurations excluding the low-speed clock (fs) to be the same as fosc. For example, reset configures fsys as 10MHz when a 10MHz oscillator is connected to the X1 or X2 pin.

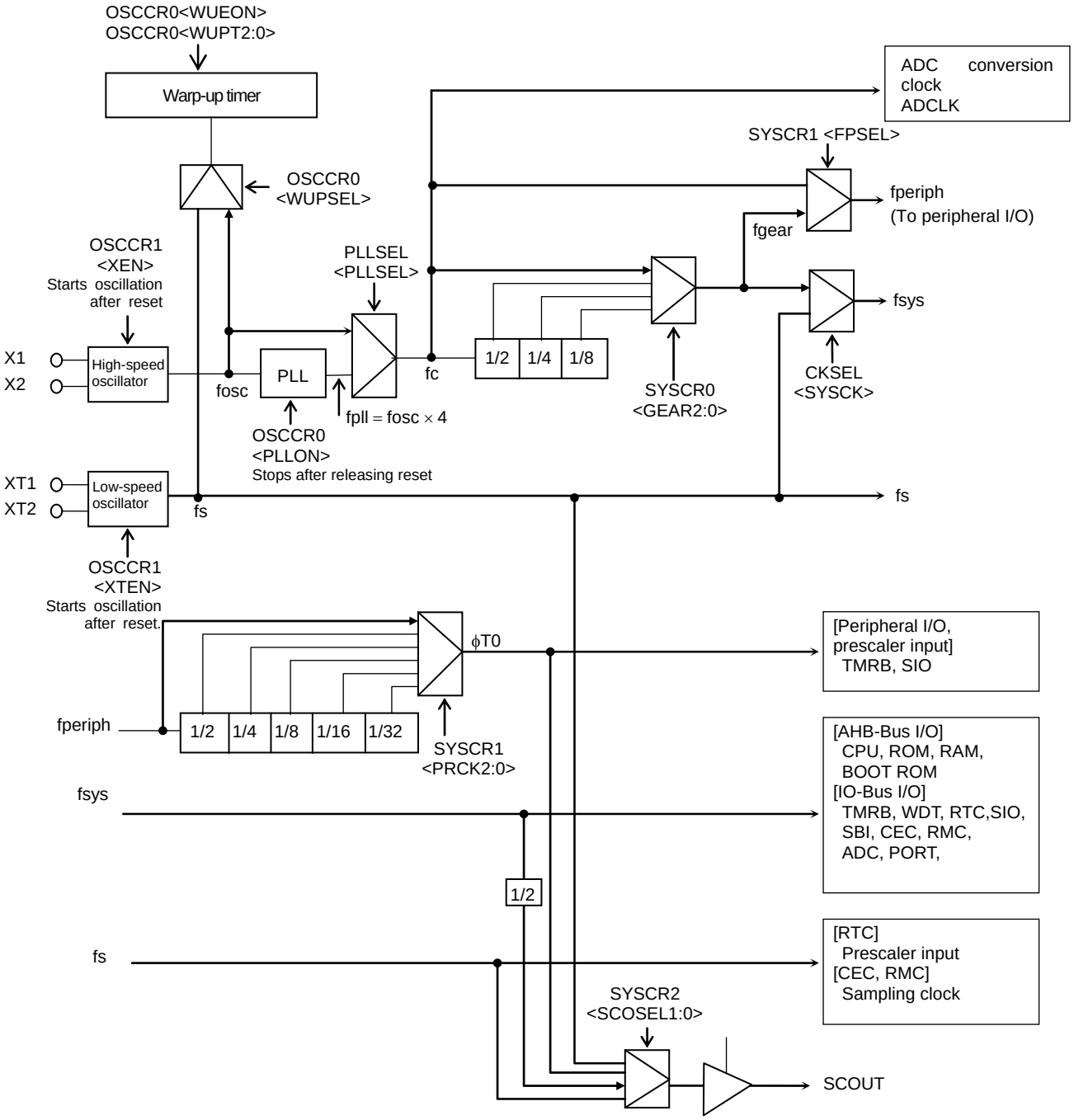


Fig.7-1 Clock Block Diagram

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

7.3.3 Clock Multiplication Circuit (PLL)

This circuit outputs the fpll clock that is quadruple of the high-speed oscillator output clock, fosc. This lowers the oscillator input frequency while increasing the internal clock speed.

The PLL is disabled after reset is released. To enable the PLL, set "1" to the OSCCR0<WUEF> bit. The PLL requires a certain amount of time to be stabilized, which should be secured using the warm-up function.

(Note) It takes approx. 200μs for the PLL to be stabilized

7.3.4 Warm-up Function

The warm-up function secures the stability time for the oscillator and the PLL with the warm-up timer.

The warm-up function is used when returning from STOP/SLEEP mode. In this case, an interrupt for returning from the low power consumption mode triggers the automatic timer count. After the specified time is reached, the system clock is output and the CPU starts operation.

In STOP/ SLEEP modes, the PLL is disabled. When returning from these modes, configure the warm-up time in consideration of the stability time of the PLL and the internal oscillator.

How to configure the warm-up function

Specify the count up clock for the warm-up counter in the OSCCR0<WUPSEL> bit.

The warm-up time can be selected by setting the OSCCR0<WUPT2:0>. The OSCCR0<WUEON><WUEF> is used to confirm the start and completion of warm-up through software (instruction). After the completion of warm-up is confirmed, switch the system clock by setting the CKSEL<SYSCK>.

When clock switching occurs, the current system clock can be checked by monitoring the CKSEL<SYSCKFLG>.

Table 7-2 shows the warm-up time.

Table 7-2 Warm-up Time (fosc=10 MHz, fs=32.768 kHz)

Warm-up time options OSCCR0<WUPT 2:0>	High-speed clock (fosc) OSCCR0<WUPSEL>="0"		Low-speed clock (fs) OSCCR0<WUPSEL>="1"	
000	–	Without warm-up	–	Without warm-up
001	2^{10}/input frequency	102.4 (μs)	2^{6}/ input frequency	1.953 (ms)
010	2^{11}/ input frequency	204.8 (μs)	2^{7}/ input frequency	3.906 (ms)
011	2^{12}/ input frequency	409.6 (μs)	2^{8}/ input frequency	7.813 (ms)
100	2^{13}/ input frequency	819.2 (μs)	2^{15}/ input frequency	1.0 (s)
101	2^{14}/ input frequency	1.638 (ms)	2^{16}/ input frequency	2.0 (s)
110	2^{15}/ input frequency	3.277 (ms)	2^{17}/ input frequency	4.0 (s)
111	2^{16}/ input frequency	6.554 (ms)	2^{18}/ input frequency	8.0 (s)

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

The following are the examples of the warm-up function configuration.

<Example 1 Securing the stability time fro the PLL>
OSCCR0<WUPSEL>="0" : Specify the warm-up counter
OSCCR0<WUPT2:0>="010" : Specify the warm-up time (204.8μs)
OSCCR0<WUEON>="1" : Start the warm-up timer (WUP)
OSCCR0<WUEF> Read : Wait until the state becomes "0" (warm-up is finished)

<Example 2> Transition from the NORMAL mode to the SLOW mode
OSCCR0<WUPSEL>="1" : Specify the warm-up counter
OSCCR0<WUPT2:0>="xxx" : Specify the warm-up time
OSCCR1<XTEN>="1" : Enable the low-speed oscillation (fs)
OSCCR0<WUEON>="1" : Start the warm-up timer.
OSCCR0<WUEF>Read : Wait until the state becomes "0" (warm-up is finished)
CKSEL<SYSCK>="1" : Switch the system clock to low speed (fs)
CKSEL<SYSCKFLG> Read: Confirm that the current state is "1" (the current system clock is fs)
OSCCR1<XEN>="0" : Disable the high-speed oscillation (fosc)

<Example 3> Transition from the SLOW mode to the NORMAL mode
OSCCR0<WUPSEL>="0" : Specify the warm-up counter
OSCCR0<WUPT2:0>="xxx" : Specify the warm-up time
OSCCR1<XEN>="1" : Enable the high-speed oscillation (fosc)
OSCCR0<WUEF>="1" : Start the warm-up timer.
OSCCR0<WUEF> Read : Wait until the state becomes "0" (warm-up is finished).
CKSEL<SYSCK>="0" : Switch the system clock to high speed (fgear)
CKSEL<SYSCKFLG> Read: Confirm that the current state is "0" (the current system clock is fgear)
OSCCR1<XTEN>="0" : Disable the low-speed oscillation (fs)

(Note) When switching the system clock, ensure that the switching has been completed by reading the SYSCR1<SYSCKFLG>.

7.3.5 System Clock

The TMPM330 offers two selectable system clocks: low-speed or high-speed. The high-speed clock is dividable.

- Input frequency from X1 and X2: 8MHz~10MH
- Allows for oscillator connection or external clock input.
- Clock gear:1/1, 1/2, 1/4, 1/8 (after reset: 1/1)

Table 7-3Range of High-frequency

Input freq. from X1 and X2	Min. operating freq.	Max. operating freq.	After reset (PLL=OFF, CG=1/1)	Clock gear (CG) @PLL=ON				Clock gear (CG) @PLL=OFF			
				1/1	1/2	1/4	1/8	1/1	1/2	1/4	1/8
8MHz	1MHz	40MHz	8	32	16	8	4	8	4	2	1
10MHz			10	40	20	10	5	10	5	2.5	1.25

* PLL=ON/OFF setting: available in OSCCR0<PLLON>
Clock gear setting: available in SYSCR0<GEAR2:0>

- Input frequency from XT1 and XT2

Table 7-4Range of Low Frequency

Input Frequency Range	Maximum Operating Frequency	Minimum Operating Frequency
30 ~ 34(kHz)	34 kHz	30 kHz

(Note 1) Switching of clock gear is executed when a value is written to the SYSCR0<GEAR2:0> register. The actual switching takes place after a slight delay.

(Note 2) The CEC function uses the low-speed clock as a sampling clock. The allowable margin of error when the CEC function is used is approximately ±4% at 32.768 kHz.

7.3.6 Prescaler Clock Control

Each peripheral function (TMRB0-9 and SIO0-2) has a prescaler for dividing a clock. As the clock φT0 to be input to each prescaler, the "fperiph" clock specified in the SYSCR1<FPSEL> can be divided according to the setting in the SYSCR1<PRCK2:0>. After the controller is reset, fperiph/1 is selected as φT0.

(Note) To use the clock gear, ensure that you make the time setting such that prescaler output φTn from each peripheral function is slower than fsys (φTn < fsys). Do not switch the clock gear while the timer counter or other peripheral function is operating.

7.3.7 System Clock Pin Output Function

The TMPM330 enables to output the system clock from a pin. The PK1/SCOUT pin can output the low speed clock fs, the system clock fsys and fsys/2, and the prescaler input clock for peripheral I/O φT0. By setting the port K registers, the PKCR<PK1C> and PKFR1<PK1F1> to “1”, the PK1/SCOUT pin (pin number 51) becomes the SCOUT output pin. The output clock is selected by setting the SYSCR3<SCOSEL1:0>.

Table 7-5 shows the pin states in each mode when the SCOUT pin is set to the SCOUT output.

Table 7-5 Scout Output State in Each Mode

Mode / SCOUT selection SYSCR2	NORMAL	SLOW	Low power consumption mode		
			IDLE	SLEEP	STOP
<SCOSEL1:0> = “00”	Output the fs clock.				
<SCOSEL1:0> = “01”	Output the fsys/2 clock.				
<SCOSEL1:0> = “10”	Output the fsys clock.				
<SCOSEL1:0> = “11”	Output the ΦT0 clock.		Fixed to “0” or “1”.		

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

7.4 Modes and Mode Transitions

7.4.1 Mode Transitions

The NORMAL mode and the SLOW mode use the high-speed and low-speed clocks for system clock respectively.

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

When the low-speed clock is not used, the SLOW and SLEEP modes cannot be used.

Fig.7-2 shows a mode transition diagram

For a description of sleep-on-exit, refer to “Cortex-M3 Technical Reference Manual”.

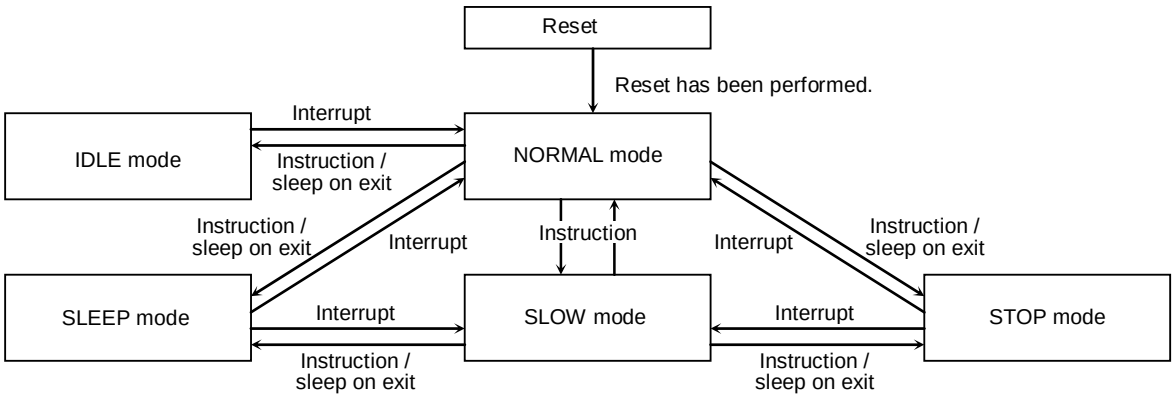


Fig.7-2 Mode Transition Diagram

7.5 Operation Modes

Two operation modes, NORMAL and SLOW, are available. The features of each mode are described below.

7.5.1 NORMAL Mode

This mode is to operate the CPU core and the peripheral hardware by using the high-speed clock. It is shifted to the NORMAL mode after reset. The low-speed clock can also be used.

7.5.2 SLOW Mode

This mode is to operate the CPU core and the peripheral hardware by using the low-speed clock with high-speed clock stopped.
The SLOW mode reduces power consumption compared to the NORMAL mode.
This mode allows only the following peripheral functions to operate: I/O ports, real-time clock (RTC), CEC and remote control signal preprocessor (RMC).

(Note) Be sure to stop peripheral functions except for the CPU, RTC, I/O ports, CEC and RMC before switching to the SLOW mode.

7.6 Low Power Consumption Modes

The TMPM330 has three low power consumption modes: IDLE, SLEEP and STOP. To shift to the low power consumption mode, specify the mode in the system control register STBYCR0<STBY2:0> and execute the WFI (Wait For Interrupt) instruction. In this case, execute reset or generate the interrupt to release the mode. Releasing by the interrupt requires settings in advance. See chapter 6 for details.

(Note 1) Transition to the low power consumption mode by executing the WFE (Wait For Event) instruction is prohibited as the TMPM330 does not offer any event for releasing the low power consumption mode.

(Note 2) The TMPM330 does not support the low power consumption mode configured with the SLEEPDEEP bit in the Cortex-M3 core. Setting the SLEEPDEEP bit of the system control register is prohibited.

The features of each mode are described as follows.

7.6.1 IDLE Mode

Only the CPU is stopped in this mode.

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

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

- 16-bit timer/event counter (TMRB)
- Serial channel (SIO)
- Serial bus interface (SBI)
- CEC
- Remote control signal preprocessor (RMC)
- AD converter (ADC)
- Watchdog timer (WDT)

7.6.2 SLEEP Mode

The internal low-speed oscillator, real time clock, CEC and RMC operate. By releasing the SLEEP mode, the device returns to the preceding mode of the SLEEP mode and starts operation.

(Note) When PA1 (pin number 56) is configured as a debug function pin, it prevents the low power consumption mode from being fully effective. Configure PA1 to function as a general-purpose port if the debug function is not used.

7.6.3 STOP Mode

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

By releasing the STOP mode, the device returns to the preceding mode of the STOP mode and starts operation.

The STOP mode enables to select the pin status by setting the STBYCR2<DRVE>.Table 7-6 shows the pin status in the STOP mode.

(Note) When PA1 (pin number 56) is configured as a debug function pin, it prevents the low power consumption mode from being fully effective. Configure PA1 to function as a general-purpose port when the debug function is not used.

Table 7-6 Pin States in STOP Mode

	Pin Name	I/O	<DRVE>=0	<DRVE>=1
Not Ports	X1, XT1	Input only	×	×
	X2, XT2	Output only	“H” level output	“H” level output
	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE	Input only	○	○
Ports	PA0, PB0 [When used as a debug pin (PxFR<n>=1) and output is enabled (PxCR<n>=1)]	Input	×	Depends on PxIE<n>.
		Output	Enabled when data is valid. Disabled when data is invalid.	Enabled when data is valid. Disabled when data is invalid.
	PF7, PG3, PJ0-3, PJ6, PJ7 [When used as an interrupt pin (PxFR<n>=1) and input is enabled (PxIE<n>=1)]	Input	○	○
		Output	×	Depends on PxCR<n>.
	Other port pins	Input	×	Depends on PxIE<n>.
		Output	×	Depends on PxCR<n>.

○ : Input or output enabled

× : Input or output disabled.

7.6.4 Low power Consumption Mode Setting

The low power consumption mode is specified by the setting of the standby control register STBYCR<STBY2:0>.

Table 7-7 shows the mode setting in the <STBY2:0>.

Table 7-7 Low power consumption mode setting

Mode	STBYCR0 <STBY2:0>
STOP	001
SLEEP	010
IDLE	011

(Note) Do not set any value other than those shown above in <STBY2:0>.

7.6.5 Operational State in Each Mode

Table 7-8 show the operational state in each mode.

For I/O port, "○" and "×" indicate that input/output is enabled and disabled respectively. For other functions, "○" and "×" indicate that clock is supplied and is not supplied respectively.

Table 7-8 Operational State in Each Mode

Block	NORMAL	SLOW	IDLE	SLEEP	STOP
Processor core	○	○	×	×	×
I/O port	○	○	○	○	* (Note 3)
ADC	○	× (Note 1)	ON/OFF selectable for each module	×	×
SIO	○	× (Note 1)		×	×
SBI	○	× (Note 1)		×	×
TMRB	○	× (Note 1)		×	×
WDT	○	× (Note 1)		×	×
CEC	○	○	○	○	×
RMC	○	○	○	○	×
RTC	○	○	○	○	×
CG	○	○	○	○	×
PLL	○	×	○	×	×
High-speed oscillator (fc)	○	*(Note 2)	○	×	×
Low-speed oscillator (fs)	○	○	○	○	×

○: Operating, ×: Stopped

(Note 1) In the SLOW mode, the ADC, SIO, SBI, TMRB and WDT cannot be used and must be stopped.

(Note 2) The high-speed oscillator does not stop automatically and must be stopped by setting the OSCCR1<XEN> bit.

(Note 3) The state depends on the SYSCR2<DRVE> bit.

7.6.6 Releasing the Low Power Consumption Mode

The low power consumption mode can be released by an interrupt request, NMI or reset. The release source that can be used is determined by the low power consumption mode selected. Details are shown in Table 7-9.

Table 7-9 Release Source in Each Mode

Low power consumption mode			IDLE	SLEEP	STOP
Release source	Interrupt	INT0~7 (Note 1)	○	○	○
		INTRTC	○	○	×
		INTTB0~9	○	×	×
		INTCAP00~60, 01~61	○	×	×
		INTRX0~2, INTTX0~2	○	×	×
		INTSBI0~2	○	×	×
		INTCECRX, INTCECTX	○	○	×
		INTRMCRX0,1	○	○	×
		INTAD/INTADHP/ INTADM	○	×	×
	NMI (INTWDT)		○	×	×
	NMI (INT pin)		○	○	○
	RESET (RESET pin)		○	○	○

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

×: Unavailable

(Note 1) **To release the low power consumption mode by using the level mode interrupt, keep the level until the interrupt handling is started. Changing the level before then will prevent the interrupt handling from starting properly.**

(Note 2) **For shifting to the low power consumption mode, set the CPU to prohibit all the interrupts other than the release source. If not, releasing may be executed by an unspecified interrupt.**

- Release by interrupt request

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

- Release by NMI

There are two kinds of NMI sources: WDT interrupt (INTWDT) and NMI pin. INTWDT can only be used in the IDLE mode. The NMI pin can be used to release all the lower power consumption modes.

- Release by reset

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

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

Refer to “Interrupts" for details.

7.6.7 Warm-up

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

In the mode transition from STOP to NORMAL/ SLOW or from SLEEP to NORMAL, the warm-up counter is activated automatically. And then the system clock output is started after the elapse of configured warm-up time. It is necessary to select the oscillator to be used for warm-up in the OCCR0<WUPSEL> and to set the warm-up time in the OSCCR0<WUPT2:0> before executing the instruction to enter the STOP/ SLEEP mode.

(Note) In STOP/ SLEEP modes, the PLL is disabled. When returning from these modes, configure the warm-up time in consideration of the stability time of the PLL and the internal oscillator. It takes approx. 200μs for the PLL to be stabilized.

In the transition from NORMAL to SLOW/ SLEEP, the warm-up is required so that the internal oscillator to stabilize if the low-speed clock is disabled. Enable the low-speed clock and then activate the warm-up by software.

In the transition from SLOW to NORMAL when the high-speed clock is disabled, enable the high-speed clock and then activate the warm-up.

Table 7-10 shows whether the warm-up setting of each mode transition is required or not.

Table 7-10 Warm-up setting in mode transition

Mode transition	Warm-up setting
NORMAL→IDLE	Not required
NORMAL→SLEEP	(Note 1)
NORMAL→SLOW	(Note 1)
NORMAL→STOP	Not required
IDLE→NORMAL	Not required
SLEEP→NORMAL	Auto-warm-up
SLEEP→SLOW	Not required
SLOW→NORMAL	(Note 2)
SLOW→SLEEP	Not required
SLOW→STOP	Not required
STOP→NORMAL	Auto-warm-up (Note 3)
STOP→SLOW	Auto-warm-up

(Note 1) If the low-speed clock is disabled, enable the low-speed clock and then activate the warm-up by software.
(Note 2) If the high-speed clock is disabled, enable the high-speed clock and then activate the warm-up by software.
(Note 3) Returning to NORMAL mode by reset does not induce the automatic warm-up. Keep the reset signal valid until the oscillator operation becomes stable.

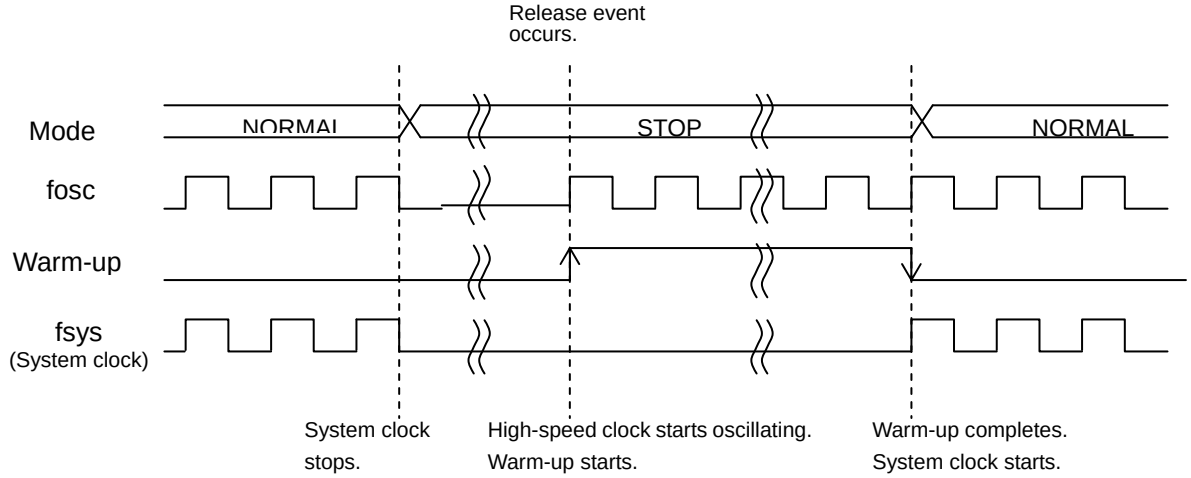
7.6.8 Clock Operations in Mode Transition

The clock operations in mode transition are described in the following sections 7.6.8.1 to 7.6.8.4.

7.6.8.1 Transition of operation modes: NORMAL→STOP→NORMAL

When returning to NORMAL mode from STOP mode, the warm-up is activated automatically. It is necessary to set the warm-up time before entering the STOP mode.

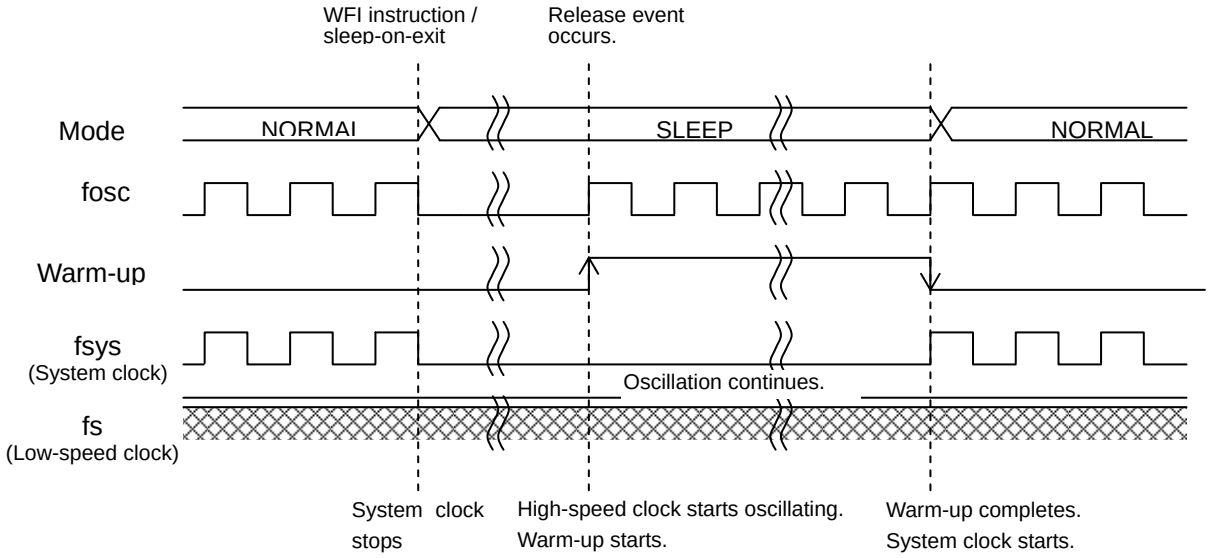
Returning to NORMAL mode by reset does not induce the automatic warm-up. Keep the reset signal asserted until the oscillator operation becomes stable.



7.6.8.2 Transition of operation modes: NORMAL→SLEEP→NORMAL

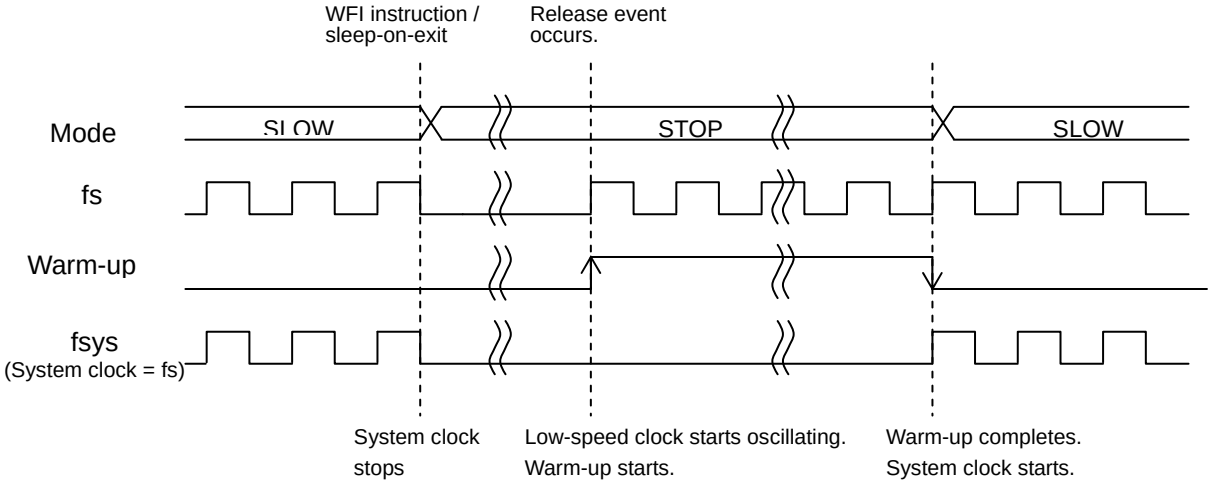
When returning to NORMAL mode from SLEEP mode, the warm-up is activated automatically. It is necessary to set the warm-up time before entering the SLEEP mode.

Returning to NORMAL mode by reset does not induce the automatic warm-up. Keep the reset signal asserted until the oscillator operation becomes stable.



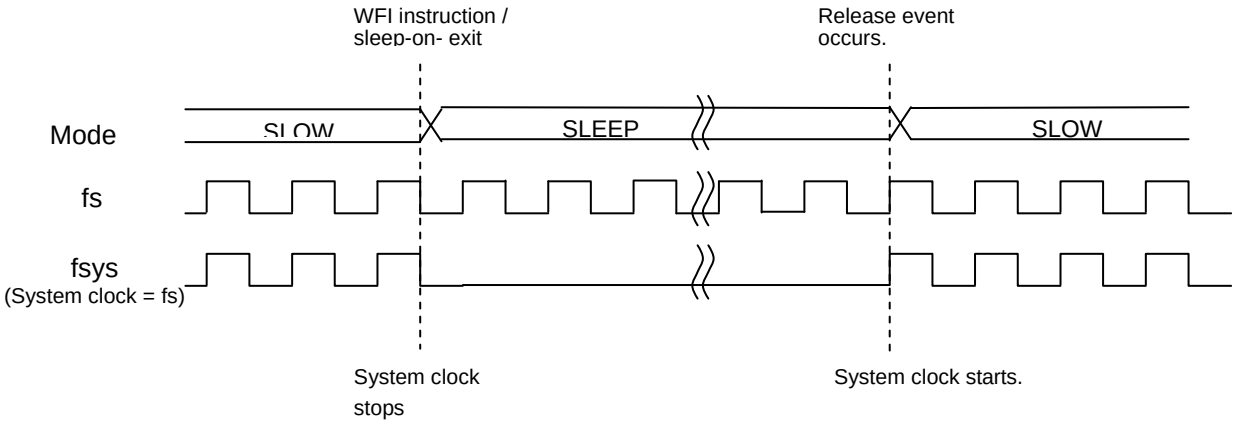
7.6.8.3 Transition of operation modes: SLOW→STOP→SLOW

The warm-up is activated automatically. It is necessary to set the warm-up time before entering the STOP mode.



7.6.8.4 Transition of operation modes: SLOW→SLEEP→SLOW

The low-speed clock continues oscillation in the SLEEP mode. There is no need to make a warm-up setting.



8 Exceptions

This chapter describes features, types and handling of exceptions.

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

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 “Cortex-M3 Technical Reference Manual”.

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

8.1.2 Handling Flowchart

The following shows how an exception/interrupt is handled.

indicates hardware handling. Indicates software handling.

Each step is described later in this chapter.

Processing	Description	See
Detection by CG/CPU	The CG/CPU detects the exception request.	Section 8.1.2.1
Handling by CPU	The CPU handles the exception request.	Section 8.1.2.2
Branch to ISR	The CPU branches to the corresponding interrupt service routine (ISR).	
Execution of ISR	Necessary processing is executed.	Section 8.1.2.3
Return from exception	The CPU branches to another ISR or returns to the previous program.	Section 8.1.2.4

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 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 or SYSRETREQ
2	Non-Maskable Interrupt	-2	NMI pin or WDT
3	Hard Fault	-1	Fault that cannot activate because a higher-priority fault is being handled or it is disabled
4	Memory Management	Configurable	Exception from the Memory Protection Unit (MPU) (Note 1) Instruction fetch from the Execute Never (XN) region
5	Bus Fault	Configurable	Access violation to the Hard Fault region of the memory map
6	Usage Fault	Configurable	Undefined instruction execution or other faults related to instruction execution
7-10	Reserved		
11	SVCall	Configurable	System service call with SVC instruction
12	Debug Monitor	Configurable	Debug monitor when the core is not halting
13	Reserved		
14	PendSV	Configurable	Pendable system service request
15	SysTick	Configurable	Notification from system timer
16-	External Interrupt	Configurable	External interrupt pin or peripheral function (Note 2)

(Note 1)	**This product does not contain the MPU.**
(Note 2)	**External interrupts have different sources and numbers in each product. For details, see “List of Interrupt Sources”.**

(3) Priority setting

Use the Interrupt Priority Registers to assign a priority to each of the external interrupts. The priority of other exceptions can be set in the System Handler Priority Registers.

The priority registers are configurable, allowing the number of bits for setting priority levels to vary between three to eight bits. Therefore, the range of priority levels that can be assigned vary with each product.

You can assign a priority level from 0 to 255 when using eight bits. Priority level 0 is the highest priority level.

If you assign the same priority level to multiple exceptions, the lowest-numbered exception has the highest priority.

(Note)	**In this product, three bits are used for assigning a priority level in the Interrupt Priority Registers and System Handler Priority Registers.**

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. This is called “pre-emption”.

(1) Stacking

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

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

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

Old SP →	<previous>
	xPSR
	PC
	LR
	r12
	r3
	r2
	r1
SP→	r0

(2) Fetching an ISR

At the same time as pushing the register contents to the stack, the CPU executes an instruction to fetch an ISR.

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 space. 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 contens 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	Bus Fault	ISR address	Optional
0x14	Usage Fault	ISR address	Optional
0x18 - 0x24	Reserved		
0x28	SVCall	ISR address	Optional
0x2C	Debug Monitor	ISR address	Optional
0x30	Reserved		
0x34	PendSV	ISR address	Optional
0x38	SysTick	ISR address	Optional
0x3C	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 (PC, xPSR, r0 to r3, r12 and LR) from the stack and adjust the SP.

- Load current active interrupt number

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

- Select SP

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

8.2 Reset Exceptions

Reset exceptions are generated from the following three sources.

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

・External reset pin

A reset exception occurs when an external reset pin changes from “L” to “H”.

・Reset exception by WDT

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

・Reset exception by SYSRESETREQ

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

8.3 Non-Maskable Interrupts (NMIs)

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

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

・External NMI pin

A non-maskable interrupt is generated when an external NMI pin changes from “H” to “L”.

・Non-maskable interrupt by WDT

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

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 pend exceptions and use a flag to know when the timer reaches "0".

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

(Note)	**In this product, the system timer counts based on a clock obtained by dividing the clock input from the X1 pin by 32.** **The SysTick Calibration Value Register is set to 0x9C4, which provides 10 ms timing when the clock input from X1 is 8 MHz.**

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 Sources

8.5.1.1 Interrupt Route

Fig. 8-1 shows an interrupt request route.

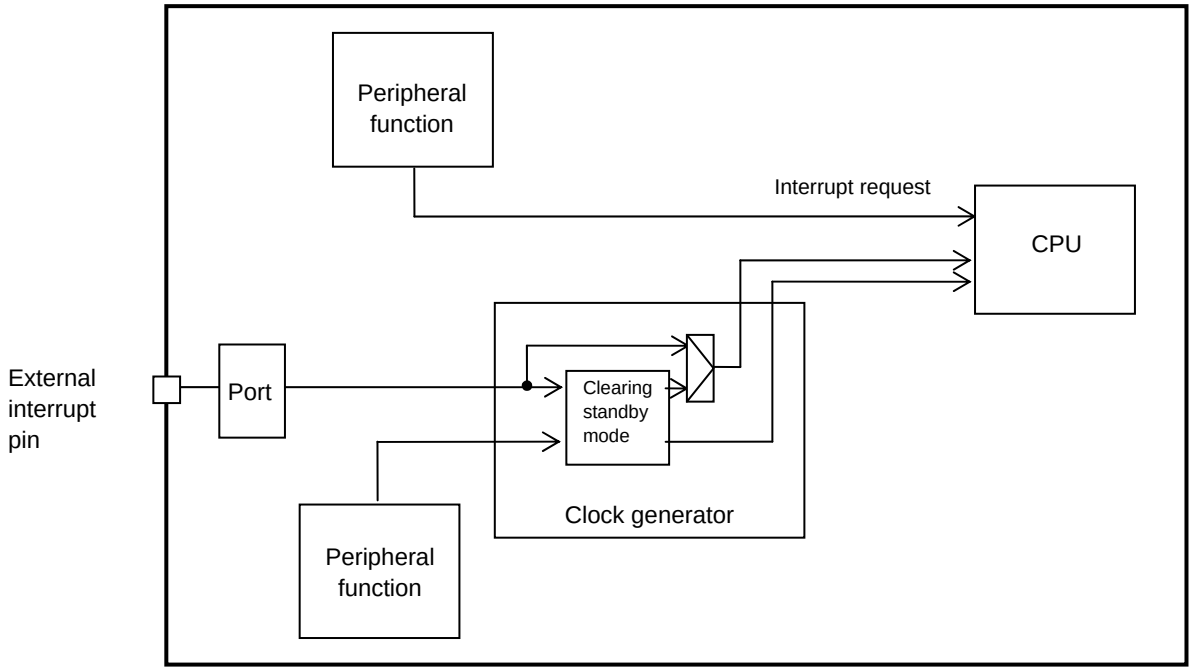


Fig. 8-1 Interrupt Route

8.5.1.2 Generation

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

・From external pin

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

・From peripheral function

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

See the chapter of each peripheral function for details.

・By setting Interrupt Set-Pending Register (forced pending)

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

8.5.1.3 Transmission

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

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

8.5.1.4 List of Interrupt Sources

Table 8-2 shows the list of interrupt sources.

Table 8-2 List of Hardware Interrupt Sources (1/2)

No.	Interrupt Source		Active state (Clearing standby)	CG interrupt mode control register
0	INT0	Interrupt pin (PJ0/70pin)	Selectable	IMCGA
1	INT1	Interrupt pin (PJ1/49pin)		
2	INT2	Interrupt pin (PJ2/86pin)		
3	INT3	Interrupt pin (PJ3/87pin)		
4	INT4	Interrupt pin (PG3/6pin)		IMCGB
5	INT5	Interrupt pin (PF7/19pin)		
6	INTRX0	Serial reception (channel.0)		
7	INTTX0	Serial transmission (channel.0)		
8	INTRX1	Serial reception (channel.1)		
9	INTTX1	Serial transmission (channel.1)		
10	INTSBI0	Serial bus interface 0		
11	INTSBI1	Serial bus interface 1		
12	INTCECRX	CEC reception	Rising edge	IMCGB
13	INTCECTX	CEC transmission		IMCGD
14	INTRMCRX0	Remote control signal reception (channel.0)		IMCGB
15	INTADHP	Highest priority AD conversion complete interrupt		
16	INTADM0	AD conversion monitoring function interrupt 0		
17	INTADM1	AD conversion monitoring function interrupt 1		
18	INTTB0	16bit TMRB match detection 0		
19	INTTB1	16bit TMRB match detection 1		
20	INTTB2	16bit TMRB match detection 2		
21	INTTB3	16bit TMRB match detection 3		
22	INTTB4	16bit TMRB match detection 4		
23	INTTB5	16bit TMRB match detection 5		
24	INTTB6	16bit TMRB match detection 6		
25	INTRTC	Real time clock timer	Falling edge	IMCGC
26	INTCAP00	16bit TMRB input capture 00		
27	INTCAP01	16bit TMRB input capture 01		
28	INTCAP10	16bit TMRB input capture 10		
29	INTCAP11	16bit TMRB input capture 11		
30	INTCAP50	16bit TMRB input capture 50		
31	INTCAP51	16bit TMRB input capture 51		
32	INTCAP60	16bit TMRB input capture 60		
33	INTCAP61	16bit TMRB Input capture 61		
34	INT6	Interrupt pin (PJ6/39pin)	Selectable	IMCGC
35	INT7	Interrupt pin (PJ7/58pin)		
36	INTRX2	Serial reception (channel.2)		
37	INTTX2	Serial transmission (channel.2)		
38	INTSBI2	Serial bus interface 2		
39	INTRMCRX1	Remote control signal reception (channel.1)	Rising edge	IMCGC
40	INTTB7	16bit TMRB match detection 7		
41	INTTB8	16bit TMRB match detection 8		
42	INTTB9	16bit TMRB match detection 9		

Table 8-2 List of Hardware Interrupt Sources (2/2)

No.	Interrupt Sources		Active state (Clearing standby)	Clock Generator
43	INTCAP20	16bit TMRB input capture 20		
44	INTCAP21	16bit TMRB input capture 21		
45	INTCAP30	16bit TMRB input capture 30		
46	INTCAP31	16bit TMRB input capture 31		
47	INTCAP40	16bit TMRB input capture 40		
48	INTCAP41	16bit TMRB input capture 41		
49	INTAD	A/D conversion completion		

8.5.1.5 Active State

The active state indicates which change in signal of an interrupt source triggers an interrupt. The CPU detects an interrupt request when an interrupt signal changes from “L” to “H”. Interrupt signals directly sent from peripheral functions to the CPU are configured to output “H” to indicate an interrupt request.

Interrupt requests from peripheral functions are set as rising-edge or falling-edge triggered. Interrupt requests from interrupt pins can be set as level-sensitive (“H” or “L”) or edge-triggered (rising or falling).

If an interrupt source is used for clearing a standby mode, setting the relevant clock generator register is also required. Enable the IMCGx<INTxEN> bit and specify the active state in the IMCGx<EMCG2:0> bits. You must set the active state for interrupt requests from each peripheral function as shown in Table 8-2.

An interrupt request detected by the clock generator is notified to the CPU with a signal in “H” level.

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

(Note) For the CEC reception/transmission, remote control signal reception and real time clock timer interrupts, set the <INTxEN>bit to “1” and specify the active state as shown in Table 8-2, even when they are not used for clearing a standby mode.

8.5.2 Interrupt Handling

8.5.2.1 Flowchart

The following shows how an interrupt is handled.

indicates hardware handling. indicates software handling.

Processing	Details	See
Settings for detection	Set the relevant NVIC registers for detecting interrupts. Set the clock generator as well if each interrupt source is used to clear a standby mode. ○ Common setting NVIC registers ○ Setting to clear standby mode Clock generator	8.5.2.2 Preparation
Settings for sending interrupt signal	Execute an appropriate setting to send the interrupt signal depending on the interrupt type. ○ Setting for interrupt from external pin Port ○ Setting for interrupt from peripheral function Peripheral function (See the chapter of each peripheral function for details.)	
Interrupt generation	An interrupt request is generated.	
CG detects interrupt (clearing standby mode)	Interrupt lines used for clearing a standby mode are connected to the CPU via the clock generator.	8.5.2.3 Detection by Clock Generator

Not clearing standby mode

Clearing standby mode

Processing	Details	See
CPU detects interrupt	The CPU detects the interrupt. If multiple interrupt requests occur simultaneously, the interrupt request with the highest priority is detected according to the priority order.	8.5.2.4 Detection by CPU
CPU handles interrupt	The CPU handles the interrupt. The CPU pushes register contents to the stack before entering the ISR.	8.5.2.5 CPU processing
ISR execution	Program for the ISR. Clear the interrupt source if needed.	8.5.2.6 Interrupt Service Routine (ISR)
Return to preceding program	Configure to return to the preceding program of the ISR.	

8.5.2.2 Preparation

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

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

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

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

(1) Disabling interrupt by CPU
(2) CPU registers setting
(3) Preconfiguration 1 (Interrupt from external pin)
(4) Preconfiguration 2 (interrupt from peripheral function)
(5) Preconfiguration 3 (Interrupt Set-Pending Register)
(5) Configuring the clock generator
(6) 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 Interrupt Clear-Enable Register. Each bit of the register, of which default setting is disabled, is assigned to a single interrupt source.

● NVIC register		
Interrupt Clear-Enable<m>	←	“1” (interrupt disabled)

(Note) m: corresponding bit

(2) CPU registers setting

You can assign a priority level to each interrupt source in the corresponding Interrupt Priority Register of the NVIC.

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.

●NVIC register		
Interrupt Priority<m>	←	“priority”

(Note) m: corresponding bit

This product uses three bits for assigning a priority level.

(3) Preconfiguration 1 (Interrupt from external pin)

Set “1” to the port function register of the corresponding pin. Setting PnFRx[m] allows the pin to be used as the function pin. Setting PnIE[m] allows the pin to be used as the input port.

● Port register		
PnFRx<PnmFRx>	←	“1”
PnIE<PnmIE>	←	“1”

(Note) n: port number
m: corresponding bit
x: function register number

In modes other than STOP mode, setting PnIE to enable input enables the corresponding interrupt input regardless of the PnFR setting. Be careful not to enable interrupts that are not used.

(4) Preconfiguration 2 (interrupt from peripheral function)

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

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

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

● NVIC register		
Interrupt Set-Pending<m>	←	“1”

(Note) m: corresponding bit

(6) Configuring the clock generator

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

Before enabling an interrupt, clear the corresponding interrupt request already held. This can avoid unexpected interrupt. To clear corresponding interrupt request, write a value corresponding to the interrupt to be used to the ICRCG register. See 8.6.3.5 CG Interrupt Request Clear Register for each value.

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

● Clock generator register		
IMCGn<EMCGm>	←	Active state
ICRCG<ICRCG>	←	Value corresponding to the interrupt to be used
IMCGn<INTmEN>	←	“1” (interrupt enabled)

(Note) n: register number
m: number assigned to interrupt source

(7) Enabling interrupt by CPU

Enable the interrupt by the CPU as shown below.

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

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

If the Interrupt Set-Pending Register is used for generating an interrupt, setting of the Clear-Pending Register is not needed as this operation will cause an interrupt request to be cleared.

●NVIC register		
Interrupt Clear-Pending<m>	←	“1”
Interrupt Set-Enable<m>	←	“1”

(Note) m: corresponding bit

8.5.2.3 Detection by Clock Generator

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

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

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

8.5.2.4 Detection by CPU

The CPU detects an interrupt request with the highest priority.

8.5.2.5 CPU processing

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

8.5.2.6 Interrupt Service Routine (ISR)

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

(1) Pushing during ISR

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

Push the contents of other registers if needed.

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

(2) Clearing an interrupt source

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

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

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

8.6 Exception/Interrupt-Related Registers

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

8.6.1 Register List

●NVIC registers	
SysTick Control and Status Register	0xE000_E010
SysTick Reload Value Register	0xE000_E014
SysTick Current Value Register	0xE000_E018
SysTick Calibration Value Register	0xE000_E01C
Interrupt Set-Enable Register 1	0xE000_E100
Interrupt Set-Enable Register 2	0xE000_E104
Interrupt Clear-Enable Register 1	0xE000_E180
Interrupt Clear-Enable Register 2	0xE000_E184
Interrupt Set-Pending Register 1	0xE000_E200
Interrupt Set-Pending Register 2	0xE000_E204
Interrupt Clear-Pending Register 1	0xE000_E280
Interrupt Clear-Pending Register 2	0xE000_E284
Interrupt Priority Register	0xE000_E400-0xE000_E430
Vector Table Offset Register	0xE000_ED08
System Handler Priority Register	0xE000_ED18,0xE000_ED1C,0xE000_ED20
System Handler Control and State Register	0xE000_ED24

● Clock generator registers		
ICRCG	CG Interrupt Request Clear Register	0x4004_0214
NMIFLG	NMI Flag Register	0x4004_0218
RSTFLG	Reset Flag Register	0x4004_021C
IMCGA	CG Interrupt Mode Control Register A	0x4004_0220
IMCGB	CG Interrupt Mode Control Register B	0x4004_0224
IMCGC	CG Interrupt Mode Control Register C	0x4004_0228
IMCGD	CG Interrupt Mode Control Register D	0x4004_022C

8.6.2 NVIC Registers

8.6.2.1 SysTick Control and Status Register

	7	6	5	4	3	2	1	0
bit Symbol						CLK SOURCE	TICKINT	ENABLE
Read/Write	R					R/W	R/W	R/W
After reset	0					0	0	0
Function	“0” is read.					0: External reference clock 1: Core clock	0: Do not pend SysTick 1: Pend SysTick	0: Disable 1: Enable

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

	23	22	21	20	19	18	17	16
bit Symbol								COUNTFLAG
Read/Write	R							R/W
After reset	0							0
Function	“0” is read.							0: Timer not counted to 0 1: Timer counted to 0

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

<bit0> <ENABLE> 1 = The counter loads with the Reload value and then begins counting down.
0 = The timer is disabled.

<bit1> <TICKINT> 1 = SysTick exceptions are pended.
0 = SysTick exceptions are not pended.

<bit2> <CLKSOURCE> 0 = External reference clock
1 = Core clock

<bit16> <COUNTFLAG> 1 =Indicates that the timer counted to 0 since last time this was read.
Clears on read of any part of the SysTick Control and Status Register.

8.6.2.2 SysTick Reload Value Register

	7	6	5	4	3	2	1	0
bit Symbol	RELOAD							
Read/Write	R/W							
After reset	Undefined							
Function	Reload value							
	15	14	13	12	11	10	9	8
bit Symbol	RELOAD							
Read/Write	R/W							
After reset	Undefined							
Function	Reload value							
	23	22	21	20	19	18	17	16
bit Symbol	RELOAD							
Read/Write	R/W							
After reset	Undefined							
Function	Reload value							
	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

<bit23:0> <RELOAD> Set the value to load into the SysTick Current Value Register when the timer reaches “0”.

(Note) In this product, the system timer counts based on a clock obtained by dividing the clock input from the X1 pin by 32.

8.6.2.3 SysTick Current Value Register

	7	6	5	4	3	2	1	0
bit Symbol	CURRENT							
Read/Write	R/W							
After reset	Undefined							
Function	[Read] Current SysTick timer value [Write] Clear							
	15	14	13	12	11	10	9	8
bit Symbol	CURRENT							
Read/Write	R/W							
After reset	Undefined							
Function	[Read] Current SysTick timer value [Write] Clear							
	23	22	21	20	19	18	17	16
bit Symbol	CURRENT							
Read/Write	R/W							
After reset	Undefined							
Function	[Read] Current SysTick timer value [Write] Clear							
	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

<bit23:0> <CURRENT> [Read] Current SysTick timer value.

[Write] 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

	7	6	5	4	3	2	1	0
bit Symbol	TENMS							
Read/Write	R							
After reset	1	1	0	0	0	1	0	0
Function	Calibration value (Note)							
	15	14	13	12	11	10	9	8
bit Symbol	TENMS							
Read/Write	R							
After reset	0	0	0	0	1	0	0	1
Function	Calibration value (Note)							
	23	22	21	20	19	18	17	16
bit Symbol	TENMS							
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Calibration value (Note)							
	31	30	29	28	27	26	25	24
bit Symbol	NOREF	SKEW						
Read/Write	RR	R0	R					
After reset	0	0	0					
Function	0: Reference clock provided 1: No reference clock	0: Calibration value is 10 ms. 1: Calibration value is not 10 ms.	“0” is read.					

<bit23:0> <TENMS> Reload value to use for 10 ms timing (0x9C4). (Note)

<bit30> <SKEW> 1 = The calibration value is not exactly 10 ms.

<bit31> <NOREF> 1 = The reference clock is not provided.

(Note) In this product, the system timer counts based on a clock obtained by dividing the clock input from the X1 pin by 32.
The SysTick Calibration Value Register is set to a value that provides 10 ms timing when the cock input from X1 is 8 MHz.

8.6.2.5 Interrupt Set-Enable Register 1

	7	6	5	4	3	2	1	0
bit Symbol	SETENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 7 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 6 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 5 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 4 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 3 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 2 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 1 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 0 [Write] 1: Enable [Read] 0: Disabled 1: Enabled
	15	14	13	12	11	10	9	8
bit Symbol	SETENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 15 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 14 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 13 [Write] 1: Enable [Read] 0 Disabled 1: Enabled	Interrupt number 12 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 11 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 10 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 9 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 8 [Write] 1: Enable [Read] 0: Disabled 1: Enabled
	23	22	21	20	19	18	17	16
bit Symbol	SETENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 23 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 22 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 21 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 20 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 19 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 18 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 17 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number16 [Write] 1: Enable [Read] 0: Disabled 1: Enabled
	31	30	29	28	27	26	25	24
bit Symbol	SETENA							
Read/Write	R/W							
After reset	00	00	00	0	0	0	0	0
Function	Interrupt number 31 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 30 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 29 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 28 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 27 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 26 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 25 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 24 [Write] 1: Enable [Read] 0: Disabled 1: Enabled

<bit31:0> <SETENA> Use these bits to enable interrupts or determine which interrupts are currently enabled.

Writing “1” to a bit in this register enables the corresponding interrupt. Writing “0” has no effect.

Reading a bit in this register returns the current state of the corresponding interrupt as shown below.

0 = Disabled
1 = Enabled

(Note) For descriptions of interrupts and interrupt numbers, see Section 8.5.1.4 List of Interrupt Sources.

8.6.2.6 Interrupt Set-Enable Register 2

	7	6	5	4	3	2	1	0
bit Symbol	SETENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 39 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 38 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 37 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 36 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 35 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 34 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 33 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 32 [Write] 1: Enable [Read] 0: Disabled 1: Enabled
	15	14	13	12	11	10	9	8
bit Symbol	SETENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 47 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 46 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 45 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 44 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 43 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 42 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 41 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 40 [Write] 1: Enable [Read] 0: Disabled 1: Enabled
	23	22	21	20	19	18	17	16
bit Symbol	SETENA						SETENA	
Read/Write	R						R/W	
After reset	0						0	0
Function	“0” is read.						Interrupt number 49 [Write] 1: Enable [Read] 0: Disabled 1: Enabled	Interrupt number 48 [Write] 1: Enable [Read] 0: Disabled 1: Enabled
	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

<bit17:0> <SETENA> Use these bits to enable interrupts or determine which interrupts are currently enabled.

Writing “1” to a bit in this register enables the corresponding interrupt. Writing “0” has no effect.

Reading a bit in this register returns the current state of the corresponding interrupt as shown below.

0 = Disabled
1 = Enabled

(Note) For descriptions of interrupts and interrupt numbers, see Section 8.5.1.4 List of Interrupt Sources.

8.6.2.7 Interrupt Clear-Enable Register

	7	6	5	4	3	2	1	0
bit Symbol	CLRENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 7 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 6 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 5 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 4 [Write] 1: Disable [read] 0: Disabled 1: Enabled	Interrupt number 3 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 2 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 1 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 0 [Write] 1: Disable [Read] 0: Disable 1: Enable
	15	14	13	12	11	10	9	8
bit Symbol	CLRENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 15 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 14 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 13 [Write] 1: Disable [Read] 0: Disabled 1 Enabled	Interrupt number 12 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 11 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 10 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 9 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 8 [Write] 1: Disable [Read] 0: Disabled 1: Enabled
	23	22	21	20	19	18	17	16
bit Symbol	CLRENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 23 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 22 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 21 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 20 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 19 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 18 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 17 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 16 [Write] 1: Disable [Read] 0: Disabled 1: Enabled
	31	30	29	28	27	26	25	24
bit Symbol	CLRENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 31 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 30 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 29 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 28 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 27 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 26 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 25 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 24 [Write] 1: Disable [Read] 0: Disabled 1: Enabled

<bit31:0> <CLRENA> Use these bits to disable or determine which interrupts are currently disabled.

Writing “1” to a bit in this register disables the corresponding interrupt. Writing “0” has no effect.

Reading a bit in this register returns the current state of the corresponding interrupt as shown below.

0 = Disabled
1 = Enabled

(Note) For descriptions of interrupts and interrupt numbers, see Section 8.5.1.4 List of Interrupt Sources.

8.6.2.8 Interrupt Clear-Enable Register 2

	7	6	5	4	3	2	1	0
bit Symbol	CLRENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 39 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 38 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 37 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 36 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 35 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 34 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 33 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 32 [Write] 1: Disable [Read] 0: Disabled 1: Enabled
	15	14	13	12	11	10	9	8
bit Symbol	CLRENA							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Interrupt number 47 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 46 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 45 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 44 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 43 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 42 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 41 [Write] 1: Disable [Read] 0: Disabled 1: Enabled	Interrupt number 40 [Write] 1: Disable [Read] 0: Disabled 1: Enabled
	23	22	21	20	19	18	17	16
bit Symbol							CLRENA	
Read/Write	R						R/W	
After reset	0						0	0
Function	“0” is read.						Interrupt number 49 [Write] 1: Disable [Read] 0: Disabled 1:Enabled	Interrupt number 48 [Write] 1: Disable [Read] 0: Disabled 1: Enabled
	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

<bit17:0> <CLRENA> Use these bits to disable or determine which interrupts are currently disabled.

Writing “1” to a bit in this register disables the corresponding interrupt. Writing “0” has no effect.

Reading a bit in this register returns the current state of the corresponding interrupt as shown below.

0 = Disabled
1 = Enabled

(Note) For descriptions of interrupts and interrupt numbers, see Section 8.5.1.4 List of Interrupt Sources.

8.6.2.9 Interrupt Set-Pending Register 1

	7	6	5	4	3	2	1	0
bit Symbol	SETPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 7 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 6 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 5 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 4 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 3 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 2 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 1 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 0 [Write] 1: Pend [Read] 0: Not pending 1: Pending
	15	14	13	12	11	10	9	8
bit Symbol	SETPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 15 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 14 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 13 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 12 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 11 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number10 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 9 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 8 [Write] 1: Pend [Read] 0: Not pending 1: Pending
	23	22	21	20	19	18	17	16
bit Symbol	SETPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 23 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 22 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 21 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 20 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 19 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 18 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 17 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 16 [Write] 1: Pend [Read] 0: Not pending 1: Pending
	31	30	29	28	27	26	25	24
bit Symbol	SETPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 31 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 30 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 29 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 28 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 27 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 26 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 25 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 24 [Write] 1: Pend [Read] 0: Not pending 1: Pending

<bit31:0>	<SETPEND>	Use these bits to force interrupts into the pending state or determine which interrupts are currently pending. Writing “1” to a bit in this register pends the corresponding interrupt. However, writing “1” has no effect on an interrupt that is already pending or is disabled. Writing “0” has no effect. Reading a bit in this register returns the current state of the corresponding interrupt as shown below. 0 = Not pending 1 = Pending Each bit in this register can be cleared by writing “1” to the corresponding bit in the Interrupt Clear-Pending Register.

(Note) For descriptions of interrupts and interrupt numbers, see Section 8.5.1.4 List of Interrupt Sources.

8.6.2.10 Interrupt Set-Pending Register 2

	7	6	5	4	3	2	1	0
bit Symbol	SETPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 39 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 38 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 37 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 36 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 35 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 34 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 33 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 32 [Write] 1: Pend [Read] 0: Not pending 1: Pending

	15	14	13	12	11	10	9	8
bit Symbol	SETPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 47 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 46 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 45 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 44 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 43 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 42 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 41 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 40 [Write] 1: Pend [Read] 0: Not pending 1: Pending

	23	22	21	20	19	18	17	16
bit Symbol							SETPEND	
Read/Write	R						R/W	
After reset	0						Undefined	
Function	“0” is read.						Interrupt number 49 [Write] 1: Pend [Read] 0: Not pending 1: Pending	Interrupt number 48 [Write] 1: Pend [Read] 0: Not pending 1: Pending

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

<bit17:0>	<SETPEND>	Use these bits to force interrupts into the pending state or determine which interrupts are currently pending. Writing “1” to a bit in this register pends the corresponding interrupt. However, writing “1” has no effect on an interrupt that is already pending or is disabled. Writing “0” has no effect. Reading a bit in this register returns the current state of the corresponding interrupt as shown below. 0 = Not pending 1 = Pending Each bit in this register can be cleared by writing “1” to the corresponding bit in the Interrupt Clear-Pending Register.

(Note) For descriptions of interrupts and interrupt numbers, see Section 8.5.1.4 List of Interrupt Sources.

8.6.2.11 Interrupt Clear-Pending Register 1

	7	6	5	4	3	2	1	0
bit Symbol	CLRPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 7 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 6 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 5 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 4 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 3 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 2 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 1 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 0 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending

	15	14	13	12	11	10	9	8
bit Symbol	CLRPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 15 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 14 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 13 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 12 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 11 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 10 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 9 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 8 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending

	23	22	21	20	19	18	17	16
bit Symbol	CLRPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 23 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 22 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 21 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 20 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 19 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 18 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 17 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 16 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending

	31	30	29	28	27	26	25	24
bit Symbol	CLRPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 31 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 30 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 29 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 28 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 27 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 26 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 25 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 24 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending

<bit31:0>	<CLRPEND>	Use these bits to clear pending interrupts or determine which interrupts are currently pending. Writing “1” to a bit in this register clears the corresponding pending interrupt. However, writing “1” has no effect on an interrupt that is already being serviced. Writing “0” has no effect. Reading a bit in this register returns the current state of the corresponding interrupt as shown below. 0 = Not pending 1 = Pending

(Note) For descriptions of interrupts and interrupt numbers, see Section 8.5.1.4 List of Interrupt Sources.

8.6.2.12 Interrupt Clear-Pending Register 2

	7	6	5	4	3	2	1	0
bit Symbol	CLRPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 39 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 38 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 37 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 36 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 35 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 34 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 33 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 32 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending

	15	14	13	12	11	10	9	8
bit Symbol	CLRPEND							
Read/Write	R/W							
After reset	Undefined							
Function	Interrupt number 47 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 46 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 45 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 44 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 43 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 42 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 41 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 40 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending

	23	22	21	20	19	18	17	16
bit Symbol							CLRPEND	
Read/Write	R						R/W	
After reset	0						Undefined	
Function	“0” is read.						Interrupt number 49 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending	Interrupt number 48 [Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

<bit17:0>	<CLRPEND>	Use these bits to clear pending interrupts or determine which interrupts are currently pending. Writing “1” to a bit in this register clears the corresponding pending interrupt. However, writing “1” has no effect on an interrupt that is already being serviced. Writing “0” has no effect. Reading a bit in this register returns the current state of the corresponding interrupt as shown below. 0 = Not pending 1 = Pending

(Note) For descriptions of interrupts and interrupt numbers, see Section 8.5.1.4 List of Interrupt Sources.

8.6.2.13 Interrupt Priority Registers

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

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

	31 24	23 16	15 8	7 0
0xE000_E400	PRI_3	PRI_2	PRI_1	PRI_0
0xE000_E404	PRI_7	PRI_6	PRI_5	PRI_4
0xE000_E408	PRI_11	PRI_10	PRI_9	PRI_8
0xE000_E40C	PRI_15	PRI_14	PRI_13	PRI_12
0xE000_E410	PRI_19	PRI_18	PRI_17	PRI_16
0xE000_E414	PRI_23	PRI_22	PRI_21	PRI_20
0xE000_E418	PRI_27	PRI_26	PRI_25	PRI_24
0xE000_E41C	PRI_31	PRI_30	PRI_29	PRI_28
0xE000_E420	PRI_35	PRI_34	PRI_33	PRI_32
0xE000_E424	PRI_39	PRI_38	PRI_37	PRI_36
0xE000_E428	PRI_43	PRI_42	PRI_41	PRI_40
0xE000_E42C	PRI_47	PRI_46	PRI_45	PRI_44
0xE000_E430	-	-	PRI_49	PRI_48

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.

	7	6	5	4	3	2	1	0
bit Symbol	PRI_0							
Read/Write	R/W			R				
After reset	0			0				
Function	Priority of interrupt number 0			“0” is read.				
	15	14	13	12	11	10	9	8
bit Symbol	PRI_1							
Read/Write	R/W			R				
After reset	0			0				
Function	Priority of interrupt number 1			“0” is read.				
	23	22	21	20	19	18	17	16
bit Symbol	PRI_2							
Read/Write	R/W			R				
After reset	0			0				
Function	Priority of interrupt number 2			“0” is read.				
	31	30	29	28	27	26	25	24
bit Symbol	PRI_3							
Read/Write	R/W			R				
After reset	0			0				
Function	Priority of interrupt number 3			“0” is read.				

<bit7:5>	<PRI_0>	Priority of interrupt number 0
<bit15:13>	<PRI_1>	Priority of interrupt number 1
<bit23:21>	<PRI_2>	Priority of interrupt number 2
<bit31:29>	<PRI_3>	Priority of interrupt number 3

8.6.2.14 Vector Table Offset Register

	7	6	5	4	3	2	1	0
bit Symbol	TBLOFF							
Read/Write	R/W	R						
After reset	0	0						
Function	Offset value	“0” is read.						
	15	14	13	12	11	10	9	8
bit Symbol	TBLOFF							
Read/Write	R/W							
After reset	0							
Function	Offset value							
	23	22	21	20	19	18	17	16
bit Symbol	TBLOFF							
Read/Write	R/W							
After reset	0							
Function	Offset value							
	31	30	29	28	27	26	25	24
bit Symbol			TBLBASE	TBLOFF				
Read/Write	R		R/W	R/W				
After reset	0		0	0				
Function	“0” is read.		Table base	Offset value				

<bit28:7> <TBLOFF> 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.

<bit29> <TBLBASE> The vector table is in:
0 = Code space
1 = SRAM space

8.6.2.15 System Handler Priority Registers

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. The System Handler Priority Registers for all other exceptions have the identical fields. Unused bits return “0” when read, and writing to unused bits has no effect.

	7	6	5	4	3	2	1	0
bit Symbol	PRI_4							
Read/Write	R/W			R				
After reset	0			0				
Function	Priority of Memory Management			“0” is read.				
	15	14	13	12	11	10	9	8
bit Symbol	PRI_5							
Read/Write	R/W			R				
After reset	0			0				
Function	Priority of Bus Fault			“0” is read.				
	23	22	21	20	19	18	17	16
bit Symbol	PRI_6							
Read/Write	R/W			R				
After reset	0			0				
Function	Priority of Usage Fault			“0” is read.				
	31	30	29	28	27	26	25	24
bit Symbol	PRI_7							
Read/Write	R/W			R				
After reset	0			0				
Function	Reserved			“0” is read.				

8.6.2.16 System Handler Control and State Register

	7	6	5	4	3	2	1	0
bit Symbol	SVCALL ACT				USGFAULT ACT		BUSFAULT ACT	MEMFAULT ACT
Read/Write	R/W	R			R/W	R	R/W	R/W
After reset	0	0			0	0	0	0
Function	SVCall 0: Inactive 1: Active	"0" is read.			Usage fault 0: Inactive 1: Active	"0" is read.	Bus fault 0: Inactive 1: Active	Memory Management 0: Inactive 1: Active

	15	14	13	12	11	10	9	8
bit Symbol	SVCALL PENDED	BUSFAULT PENDED	MEMFAULT PENDED	USGFAULT PENDED	SYSTICK ACT	PENDSV ACT		MONITOR ACT
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
After reset	0	0	0	0	0	0	0	0
Function	SVCall 0: Not pended 1: Pended	Bus Fault 0: Not pended 1: Pended	Memory Management 0: Not pended 1: Pended	Usage Fault 0: Not pended 1: Pended	SysTick 0: Inactive 1: Active	PendSV 0: Inactive 1: Active	"0" is read.	Debug Monitor 0: Inactive 1: Active

	23	22	21	20	19	18	17	16
bit Symbol						USGFAULT ENA	BUSFAULT ENA	MEMFAULT ENA
Read/Write	R					R/W	R/W	R/W
After reset	0					0	0	0
Function	"0" is read.					Usage Fault 0: Disable 1: Enable	Bus Fault 0: Disable 1: Enable	Memory Management 0: Disable 1: Enable

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	"0" is read.							

<bit0>	<MEMFAULTACT>	Reads as “1” if Memory Management is active.
<bit1>	<BUSFAULTACT>	Reads as “1” if Bus Fault is active.
<bit3>	<USGFALTACT>	Reads as “1” if Usage Fault is active.
<bit7>	<SVCALLACT>	Reads as “1” if SVCall is active.
<bit8>	<MONITORACT>	Reads as “1” if Debug Monitor is active.
<bit10>	<PENDSVACT>	Reads as “1” if PendSV is active.
<bit11>	<SYSTICKACT>	Reads as “1” if SysTick is active.
<bit12>	<USGFAULTPENDED>	Reads as “1” if Usage Fault is pended.
<bit13>	<MEMFAULTPENDED>	Reads as “1” if Memory Management is pended.
<bit14>	<BUSFAULTPENDED>	Reads as “1” if Bus Fault is pended.
<bit15>	<SVCALLPENDED>	Reads as “1” if SVCall is pended.
<bit16>	<MEMFAULTENA>	Set to “0” to disable or “1” to enable Memory Management.
<bit17>	<BUSFAULTENA>	Set to “0” to disable or “1” to enable Bus Fault.
<bit18>	<USGFAULTENA>	Set to “0” to disable or “1” to enable Usage Fault.

(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 NVIC Registers

8.6.3.1 CG Interrupt Mode Control Register A

IMCGA

	7	6	5	4	3	2	1	0
bit Symbol		EMCG02	EMCG01	EMCG00	EMST01	EMST00		INT0EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INT0 standby clear request. (101~111: setting prohibited) 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active state of INT0 standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INT0 clear input 0: Disable 1: Enable
	15	14	13	12	11	10	9	8
bit Symbol		EMCG12	EMCG11	EMCG10	EMST11	EMST10		INT1EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INT1 standby clear request. (101~111: setting prohibited) 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active state of INT1 standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INT1 clear input 0: Disable 1: Enable
	23	22	21	20	19	18	17	16
bit Symbol		EMCG22	EMCG21	EMCG20	EMST21	EMST20		INT2EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INT2 standby clear request. (101~111: setting prohibited) 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active state of INT2 standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INT2 clear input 0: Disable 1: Enable
	31	30	29	28	27	26	25	24
bit Symbol		EMCG32	EMCG31	EMCG30	EMST31	EMST30		INT3EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INT3 standby clear request. (101~111: setting prohibited) 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active state of INT3 standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INT3 clear input 0: Disable 1: Enable

(Note 1) **Refer to EMSTxx bit to know the active condition which is used for clearing standby.**

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

8.6.3.2 CG Interrupt Mode Control Register B

IMCGB

	7	6	5	4	3	2	1	0
bit Symbol		EMCG42	EMCG41	EMCG40	EMST41	EMST40		INT4EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INT4 standby clear request (101~111: setting prohibited) 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active state of INT4 standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INT4 clear input 0: Disable 1: Enable
	15	14	13	12	11	10	9	8
bit Symbol		EMCG52	EMCG51	EMCG50	EMST51	EMST50		INT5EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INT5 standby clear request. (101~111: setting prohibited) 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active state of INT5 standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INT5 clear input 0: Disable 1: Enable
	23	22	21	20	19	18	17	16
bit Symbol		EMCG62	EMCG61	EMCG60	EMST61	EMST60		INT6EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INTCECRX standby clear request. **Set it as shown below.** 011: Rising edge			Active state of INTCECRX standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INTCEC RX clear input 0: Disable 1: Enable
	31	30	29	28	27	26	25	24
bit Symbol		EMCG72	EMCG71	EMCG70	EMST71	EMST70		INT7EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INTRMCRX0 standby clear request. **Set it as shown below.** 011: Rising edge			Active state of INTRMCRX0 standby clear request 00: – 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INTRMC RX0 clear input 0: Disable 1: Enable

(Note 1) Refer to EMSTxx bit to know the active condition which is used for clearing standby.

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

8.6.3.3 CG Interrupt Mode Control Register C

IMCGC

	7	6	5	4	3	2	1	0
bit Symbol		EMCG82	EMCG81	EMCG80	EMST81	EMST80		INT8EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INTRTC standby clear request. **Set it as shown below.** 010: Falling edge			Active state of INTRTC standby clear request. 00: - 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INTRTC clear input 0: Disable 1: Enable
	15	14	13	12	11	10	9	8
bit Symbol		EMCG92	EMCG91	EMCG90	EMST91	EMST90		INT9EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INT6 standby clear request. (101~111: setting prohibited) 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active state of INT6 standby clear request. 00: - 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INT6 clear input 0: Disable 1: Enable
	23	22	21	20	19	18	17	16
bit Symbol		EMCGA2	EMCGA1	EMCGA0	EMSTA1	EMSTA0		INTAEN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INT7 standby clear request. (101~111: setting prohibited) 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active state of INT7 standby clear request. 00: - 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INT7clear input 0: Disable 1: Enable
	31	30	29	28	27	26	25	24
bit Symbol		EMCGB2	EMCGB1	EMCGB0	EMSTB1	EMSTB0		INTBEN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INTRMCRX1 standby clear request. **Set it as shown below.** 011: Rising edge			Active state of INTRMCRX1 standby clear request. 00: - 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INTRMC RX1 clear input 0: Disable 1: Enable

(Note 1) Refer to EMSTxx bit to know the active condition which is used for clearing standby.

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

8.6.3.4 CG Interrupt Mode Control Register D

IMCGD

	7	6	5	4	3	2	1	0
bit Symbol		EMCGC2	EMCGC1	EMCGC0	EMSTC1	EMSTC0		INTCEN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Active state setting of INTCECTX standby clear request. **Set it as shown below.** 010: Rising edge			Active state of INTCECTX standby clear request. 00: - 01: Rising edge 10: Falling edge 11: Both edges		"0" is read.	INTCECTX Clear input 0: Disable 1: Enable

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R	R/W			R			R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Write any value.			"0" is read.			Write "0".

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R/W			R			R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Write any value.			"0" is read.			Write "0".

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R/W			R			R/W
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Write any value.			"0" is read.			Write "0".

(Note 1) Refer to EMSTxx bit to know the active condition which is used for clearing standby.

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

8.6.3.5 CG Interrupt Request Clear Register

ICRCG

	7	6	5	4	3	2	1	0
bit Symbol				ICRCG4	ICRCG3	ICRCG2	ICRCG1	ICRCG0
Read/Write	R			W				
After reset	0			0	0	0	0	0
Function	“0” is read.			Clear interrupt requests. 0_0000: INT0 0_0001: INT1 0_0010: INT2 0_0011: INT3 0_0100: INT4 0_0101: INT5 0_0110: INTCECRX 0_0111: IINTRMCRX0 0_1000: INTRTC 0_1001: INT6 0_1010: INT7 0_1011: INTRMCRX1 0_1100: INTCECTX * 0_1101~1_1111: setting prohibited * ”0” is read.				

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write								
After reset								
Function	“0” is read.							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	“0” is read.							

8.6.3.6 NMI Flag Register

NMIFLG

	7	6	5	4	3	2	1	0
bit Symbol							NMIFLG1	NMIFLG0
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.						NMI source generation flag 0: not applicable 1: generated from NMI pin	NMI source generation flag 0: not applicable 1: generated from WDT

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

(Note) <NMIFLG1:0> are cleared to "0" when they are read.

8.6.3.7 Reset Flag Register

RSTFLG

	7	6	5	4	3	2	1	0
bit Symbol				SYSRSTF		WDTRSTF	PINRSTF	PONRSTF
Read/Write	R			R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	1	1
Function	"0" is read.			Debug reset flag 0: "0" is written 1: Reset from SYSRSTRQ (Note 2)	Write "0".	WDT reset flag 0: "0" is written 1: Reset from WDT	RESET pin flag 0: "0" is written 1: Reset from RESET pin	Power-on reset flag 0: "0" is written 1: Reset from power-on reset

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.							

(Note 1) The TMPM330 has power-on reset circuit and this register is initialized only by power-on reset. Therefore, "1" is set to the <PONRSTF> bit in initial reset state right after power-on. Note that this bit is not set by the second and subsequent resets and this register is not cleared automatically. Write "0" to clear the register.

(Note 2) This flag indicates a reset generated by the SYSRESETREQ bit of the Application Interrupt and Reset Control Register of the CPU's NVIC.

9 Input/Output Ports

9.1 Port registers

Px **: Port register**
To read/ write port data.

PxCR **: Control register**
To control input/output
* Need to enable the input with PxIE register even when input is set.

PxFRn **: Function register**
To set functions. An assigned function can be activated by setting “1”.

PxOD **: Open drain control register**
To switch the input of a register that can be set as programmable open drain.

PxPUP **: Pull up control register**
To control program pull ups.

PxPDN **: Pull down control register**
To control programmable pull downs.

PxIE **: Input control enable register**
To control inputs. “0” is set as default to avoid through current. This setting prohibits inputs.

9.2 Port Functions

9.2.1 Port States in STOP Mode

Input and output in STOP mode are enabled/disabled by the STBYCR2<DRVE> bit in the Standby Control Register.

If PxIE or PxCR is enabled with <DRVE>=1, input or output is enabled respectively in STOP m

If <DRVE>=0, both input and output are disabled in STOP mode except for some ports even if PxIE and PxCR are enabled.

The differences are summarized in the table shown below.

Port	I/O	<DRVE>=0	<DRVE>=1
PA0, PB0 [When used for debug (PxFR<n>=1) and output is enabled (PxCR<n>=1)]	Input	×	Depends on PxIE<n>.
	Output	Enabled when data is valid. Disabled when data is invalid.	Enabled when data is valid. Disabled when data is invalid.
PF7, PG3, PJ0-3, PJ6, PJ7 [When used for interrupt (PxFR<n>=1) and input is enabled (PxIE<n>=1)]	Input	○	○
	Output	×	Depends on PxCR<n>.
Other ports	Input	×	Depends on PxIE<n>.
	Output	×	Depends on PxCR<n>.

○: Input or output enabled

×: Input or output disabled

9.2.2 Precaution for Mode Transition

If PA1 is configured as a debug function pin, it prevents the low power consumption mode from being fully effective. Configure PA1 to function as a general-purpose port if the debug function is not used.

9.2.3 Port A (PA0~PA7)

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

Reset configures PA0 and PA1 as debug communication pins. PA0 is initialized as the TMS/SWDIO pin with input, output and pull-up enabled. PA1 is initialized as the TCK/SWCLK pin with input and pull-down enabled.

Other bits of the port A are initialized as general-purpose ports with input, output and pull-up disabled.

(Note 1)	**PA0 and PA1 are initialized as debug communication pins with input, output, pull-up and pull-down enabled.**
(Note 2)	**If PA0 is configured as the TMS/SWDIO pin, output is enabled even in STOP mode regardless of the STBYCR2<DRVE> bit setting.**
(Note 3)	**If PA1 is configured as a debug communication pin, it prevents the low power consumption mode from being fully effective. Configure PA1 to function as a general-purpose port if the debug function is not used.**

Port A Circuit Type

	7	6	5	4	3	2	1	0
Type	T1	T9	T9	T9	T9	T9	T6	T12

Port A register

PA (0x4000_0000)

	7	6	5	4	3	2	1	0
Bit Symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Read/Write	R/W							
After reset	"0"							

Port A control register

PACR (0x4000_0004)

	7	6	5	4	3	2	1	0
Bit Symbol	PA7C	PA6C	PA5C	PA4C	PA3C	PA2C	PA1C	PA0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	1
Function	0: Output disabled 1: Output enabled							

Port A function register 1

PAFR1 (0x4000_0008)

	7	6	5	4	3	2	1	0
Bit Symbol	—	PA6F1	PA5F1	PA4F1	PA3F1	PA2F1	PA1F1	PA0F1
Read/Write	R	R/W						
After reset	0	0	0	0	0	0	1	1
Function	"0" is read.	0:PORT 1: TRACE DATA3	0:PORT 1: TRACE DATA2	0:PORT 1: TRACE DATA1	0:PORT 1TRACE DATA0	0:PORT 1: TRACE CLK	0:PORT 1: TCK/ SWCLK	0:PORT 1: TMS/ SWDIO

Port A pull-up control register

PAPUP (0x4000_002C)

	7	6	5	4	3	2	1	0
Bit Symbol	PA7UP	PA6UP	PA5UP	PA4UP	PA3UP	PA2UP	—	PA0UP
Read/Write	R/W						R	R/W
After reset	0	0	0	0	0	0	0	1
Function	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	"0" is read.	Pull-up 0:off 1:on

Port A pull-down control register

PAPDN (0x4000_0030)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	—	PA1DN	—
Read/Write	R						R/W	R
After reset	0	0	0	0	0	0	1	0
Function	"0" is read.						Pull-up 0:off 1:on	"0" is read.

Port A input enable control register

PAIE (0x4000_0038)

	7	6	5	4	3	2	1	0
Bit Symbol	PA7IE	PA6IE	PA5IE	PA4IE	PA3IE	PA2IE	PA1IE	PA0IE
Read/Write	R/W							
After reset	0	0	0	0	0	0	1	1
Function	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.4 Port B (PB0~PB7)

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

Reset configures PB0, PB1 and PB2 as debug communication pins. PB0 is initialized as the TDO/SWV pin with output enabled. PB1 is initialized as the TDI pin with input and pull-up enabled. PB2 is initialized as the TRST pin with input and pull-up enabled.

Other bits of the port B are initialized as general-purpose ports with input, output and pull-up disabled.

(Note 1)	**PB0, PB1 and PB2 are initialized as debug communication pins with input, output and pull-up enabled.**
(Note 2)	**If PB0 is configured as the TDO/SWV pin, output is enabled even in STOP mode regardless of the STBYCR2<DRVE> bit setting.**

Port B Circuit Type

	7	6	5	4	3	2	1	0
Type	T1	T1	T1	T1	T1	T2	T2	T11

Port B register

PB (0x4000_0040)

	7	6	5	4	3	2	1	0
Bit Symbol	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
Read/Write	R/W							
After reset	"0"							

Port B control register

PBCR (0x4000_0044)

	7	6	5	4	3	2	1	0
Bit Symbol	PB7C	PB6C	PB5C	PB4C	PB3C	PB2C	PB1C	PB0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	1
Function	0: Output disabled 1: Output enabled							

Port B function register 1

PBFR1 (0x4000_0048)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	PB2F1	PB1F1	PB0F1
Read/Write	R					R/W		
After reset	0					1	1	1
Function	"0" is read.					0:PORT 1: TRST	0:PORT 1: TDI	0:PORT 1: TDO/SWV

Port B pull-up control register

PBPUP (0x4000_006C)

	7	6	5	4	3	2	1	0
Bit Symbol	PB7UP	PB6UP	PB5UP	PB4UP	PB3UP	PB2UP	PB1UP	PB0UP
Read/Write	R/W							
After reset	0	0	0	0	0	1	1	0
Function	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on

Port B input enable control register

PBIE (0x4000_0078)

	7	6	5	4	3	2	1	0
Bit Symbol	PB7IE	PB6IE	PB5IE	PB4IE	PB3IE	PB2IE	PB1IE	PB0IE
Read/Write	R/W							
After reset	0	0	0	0	0	1	1	0
Function	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.5 Port C (PC0~PC3)

The port C is a 4-bit input port. Besides the general-purpose input function, the port C functions as analog input pins of the A/D converter.

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

Set the input enable control register when you use the port C as input ports. The setting is not required when you use it as analog input pins of the A/D converter.

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

Port C Circuit Type

	7	6	5	4	3	2	1	0
Type	–	–	–	–	T17	T17	T17	T17

Port C register

		7	6	5	4	3	2	1	0
PC (0x4000_0080)	Bit Symbol	–	–	–	–	PC3	PC2	PC1	PC0
	Read/Write	R				R			
	After reset	"0" is read.				"1"			

Port C pull-up control register

		7	6	5	4	3	2	1	0
PCPUP (0x4000_00AC)	Bit Symbol	–	–	–	–	PC3UP	PC2UP	PC1UP	PC0UP
	Read/Write	R				R/W			
	After reset	0				0	0	0	0
	Function	"0" is read.				Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on

Port C input enable control register

		7	6	5	4	3	2	1	0
PCIE (0x4000_00B8)	Bit Symbol	–	–	–	–	PC3IE	PC2IE	PC1IE	PC0IE
	Read/Write	R				R/W			
	After reset	0				0	0	0	0
	Function	"0" is read.				Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.6 Port D (PD0~PD7)

The port D is an 8-bit input port. Besides the general-purpose input function, the port D receives an analog input of the A/D converter and a 16-bit timer input.

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

Set the input enable control register when you use the port D as input ports. Set the function register 1 and input enable control register when you use the port D as input pins of the 16-bit timer. No register setting is required when you use it as analog input pins of the A/D converter.

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

Port D Circuit Type

	7	6	5	4	3	2	1	0
Type	T17	T17	T17	T17	T18	T18	T18	T18

Port D register

PD (0x4000_00C0)

	7	6	5	4	3	2	1	0
Bit Symbol	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
Read/Write	R							
After reset	“1”							

Port D function register 1

PDFR1 (0x4000_00C8)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	PD3F1	PD2F1	PD1F1	PD0F1
Read/Write	R				R/W			
After reset	0				0	0	0	0
Function	“0” is read.				0:PORT 1: TB6IN1	0:PORT 1: TB6IN0	0:PORT 1: TB5IN1	0:PORT 1: TB5IN0

Port D pull-up control register

PDPUP (0x4000_00EC)

	7	6	5	4	3	2	1	0
Bit Symbol	PD7UP	PD6UP	PD5UP	PD4UP	PD3UP	PD2UP	PD1UP	PD0UP
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on

Port D input enable control register

PDIE (0x4000_00F8)

	7	6	5	4	3	2	1	0
Bit Symbol	PD7IE	PD6IE	PD5IE	PD4IE	PD3IE	PD2IE	PD1IE	PD0IE
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.7 Port E (PE0~PE6)

The port E is a general-purpose, 7-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose port function, the port E performs the serial interface function and the remote control signal preprocessor input function.

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

Port E Circuit Type

	7	6	5	4	3	2	1	0
Type	—	T16	T4	T10	T4	T16	T4	T10

Port E register

		7	6	5	4	3	2	1	0
PE	Bit Symbol	—	PE6	PE5	PE4	PE3	PE2	PE1	PE0
(0x4000_0100)	Read/Write	R	R/W						
	After reset	“0”	“0”						

Port E control register

PECR (0x4000_0104)

	7	6	5	4	3	2	1	0
Bit Symbol	—	PE6C	PE5C	PE4C	PE3C	PE2C	PE1C	PE0C
Read/Write	R	R/W						
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.	0: Output disabled 1: Output enabled						

Port E function register 1

PEFR1 (0x4000_0108)

	7	6	5	4	3	2	1	0
Bit Symbol	—	PE6F1	PE5F1	PE4F1	PE3F1	PE2F1	PE1F1	PE0F1
Read/Write	R	R/W						
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.	0:PORT 1:SCLK1	0:PORT 1:RXD1	0:PORT 1: TXD1	0:PORT 1:RXIN0	0:PORT 1:SCLK0	0:PORT 1:RXD0	0:PORT 1: TXD0

Port E function register 2

PEFR2 (0x4000_010C)

	7	6	5	4	3	2	1	0
Bit Symbol	—	PE6F2	—	—	—	PE2F2	—	—
Read/Write	R	R/W	R			R/W	R	
After reset	0	0	0			0	0	
Function	"0" is read.	0:PORT 1:CTS1	"0" is read.			0:PORT 1:CTS0	"0" is read.	

Port E open drain control register

PEOD (0x4000_ 0128)

	7	6	5	4	3	2	1	0
Bit Symbol	—	PE6OD	PE5OD	PE4OD	PE3OD	PE2OD	PE1OD	PE0OD
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain

Port E pull-up control register

PEPUP (0x4000_012C)

	7	6	5	4	3	2	1	0
Bit Symbol	—	PE6UP	PE5UP	PE4UP	PE3UP	PE2UP	PE1UP	PE0UP
Read/Write	R	R/W						
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on

Port E input enable control register

PEIE (0x4000_0138)

	7	6	5	4	3	2	1	0
Bit Symbol	—	PE6IE	PE5IE	PE4IE	PE3IE	PE2IE	PE1IE	PE0IE
Read/Write	R	R/W						
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.8 Port F (PF0~PF7)

The port F is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose port function, the port F performs the functions of the serial interface, the remote control signal preprocessor, the serial bus interface and the external interrupt input.

To use the external interrupt input for releasing STOP mode, select this function in the PFFR register and enable input in the PFIE register. These settings enable the interrupt input even if the STBYCR2<DRVE> bit in the clock/mode control block is set to stop driving of pins during STOP mode.

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

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

Port F Circuit Type

	7	6	5	4	3	2	1	0
Type	T8	T13	T13	T13	T4	T16	T4	T10

Port F register

		7	6	5	4	3	2	1	0
PF	Bit Symbol	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
(0x4000_0140)	Read/Write	R/W							
	After reset	“0”							

Port F control register

PFCR (0x4000_0144)

	7	6	5	4	3	2	1	0
Bit Symbol	PF7C	PF6C	PF5C	PF4C	PF3C	PF2C	PF1C	PF0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port F function register 1

PFFR1 (0x4000_0148)

	7	6	5	4	3	2	1	0
Bit Symbol	PF7F1	PF6F1	PF5F1	PF4F1	PF3F1	PF2F1	PF1F1	PF0F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1: INT5	0:PORT 1: SCK1	0:PORT 1: SI1/ SCL1	0:PORT 1: SO1/ SDA1	0:PORT 1: RXIN1	0:PORT 1: SCLK2	0:PORT 1: RXD2	0:PORT 1: TXD2

Port F function register 2

PFFR2 (0x4000_014C)

	7	6	5	4	3	2	1	0
Bit Symbol	–	–	–	–	–	PF2F2	–	–
Read/Write	R					R/W	R	
After reset	0					0	0	
Function	“0” is read.					0:PORT 1:CTS2	“0” is read.	

Port F open drain control register

PFOD (0x4000_0168)

	7	6	5	4	3	2	1	0
Bit Symbol	PF7OD	PF6OD	PF5OD	PF4OD	PF3OD	PF2OD	PF1OD	PF0OD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain

Port F pull-up control register

PFPUP (0x4000_016C)

	7	6	5	4	3	2	1	0
Bit Symbol	PF7UP	PF6UP	PF5UP	PF4UP	PF3UP	PF2UP	PF1UP	PF0UP
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on

Port F input enable control register

PFIE (0x4000_0178)

	7	6	5	4	3	2	1	0
Bit Symbol	PF7IE	PF6IE	PF5IE	PF4IE	PF3IE	PF2IE	PF1IE	PF0IE
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.9 Port G (PG0~PG7)

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

To use the external interrupt input for releasing STOP mode, select this function in the PGFR register and enable input in the PGIE register. These settings enable the interrupt input even if the STBYCR2<DRVE> bit in the clock/mode control block is set to stop driving of pins during STOP mode.

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

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

Port G Circuit Type

	7	6	5	4	3	2	1	0
Type	T10	T13	T13	T13	T8	T13	T13	T13

Port G register

PG (0x4000_0180)

	7	6	5	4	3	2	1	0
Bit Symbol	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
Read/Write	R/W							
After reset	"0"							

Port G control register

PGCR (0x4000_0184)

	7	6	5	4	3	2	1	0
Bit Symbol	PG7C	PG6C	PG5C	PG4C	PG3C	PG2C	PG1C	PG0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port G function register 1

PGFR1 (0x4000_0188)

	7	6	5	4	3	2	1	0
Bit Symbol	PG7F1	PG6F1	PG5F1	PG4F1	PG3F1	PG2F1	PG1F1	PG0F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1: TB8OUT	0:PORT 1: SCK2	0:PORT 1: SI2/ SCL2	0:PORT 1: SO2/ SDA2	0:PORT 1: INT4	0:PORT 1: SCK0	0:PORT 1: SI0/ SCL0	0:PORT 1: SO0/ SDA0

Port G open drain control register

PGOD (0x4000_01A8)

	7	6	5	4	3	2	1	0
Bit Symbol	PG7OD	PG6OD	PG5OD	PG4OD	PG3OD	PG2OD	PG1OD	PG0OD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain

Port G pull-up control register

PGPUP (0x4000_01AC)

	7	6	5	4	3	2	1	0
Bit Symbol	PG7UP	PG6UP	PG5UP	PG4UP	PG3UP	PG2UP	PG1UP	PG0UP
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on

Port G input enable control register

PGIE (0x4000_01B8)

	7	6	5	4	3	2	1	0
Bit Symbol	PG7IE	PG6IE	PG5IE	PG4IE	PG3IE	PG2IE	PG1IE	PG0IE
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.10 Port H (PH0~PH7)

The port H is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose port function, the port H performs the functions of the 16-bit timer input and the operation mode setting.

While a reset signal is in “0”state, the PH0 input and pull-up are enabled. At the rising edge of the reset signal, if PH0 is “1”, the device enters single mode and boots from the on-chip flash memory. If PH0 is “0”, the device enters single boot mode and boots from the internal boot program. For details of single boot mode, refer to “Chapter 17 Flash Memory Operation”.

Reset initializes all bits of the port H as general-purpose ports with input and output disabled. Pull-up is enabled for PH0 and disabled for PH1 through PH7.

Port H Circuit Type

	7	6	5	4	3	2	1	0
Type	T3	T3	T3	T3	T3	T3	T3	T5

Port H register

PH (0x4000_01C0)

	7	6	5	4	3	2	1	0
Bit Symbol	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
Read/Write	R/W							
After reset	"0"							

Port H control register

PHCR (0x4000_01C4)

	7	6	5	4	3	2	1	0
Bit Symbol	PH7C	PH6C	PH5C	PH4C	PH3C	PH2C	PH1C	PH0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port H function register 1

PHFR1 (0x4000_01C8)

	7	6	5	4	3	2	1	0
Bit Symbol	PH7F1	PH6F1	PH5F1	PH4F1	PH3F1	PH2F1	PH1F1	PH0F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1: TB3IN1	0:PORT 1: TB3IN0	0:PORT 1: TB2IN1	0:PORT 1: TB2IN0	0:PORT 1: TB1IN1	0:PORT 1: TB1IN0	0:PORT 1: TB0IN1	0:PORT 1: TB0IN0

Port H pull-up control register

PHPUP (0x4000_01EC)

	7	6	5	4	3	2	1	0
Bit Symbol	PH7UP	PH6UP	PH5UP	PH4UP	PH3UP	PH2UP	PH1UP	PH0UP
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	1
Function	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on

Port H input enable control register

PHIE (0x4000_01F8)

	7	6	5	4	3	2	1	0
Bit Symbol	PH7IE	PH6IE	PH5IE	PH4IE	PH3IE	PH2IE	PH1IE	PH0IE
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.11 Port I (PI0~PI7)

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

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

Port I Circuit Type

	7	6	5	4	3	2	1	0
Type	T3	T3	T9	T9	T9	T9	T9	T9

Port I register

PI (0x4000_0200)

	7	6	5	4	3	2	1	0
Bit Symbol	PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0
Read/Write	R/W							
After reset	"0"							

Port I control register

PICR (0x4000_0204)

	7	6	5	4	3	2	1	0
Bit Symbol	PI7C	PI6C	PI5C	PI4C	PI3C	PI2C	PI1C	PI0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port I function register 1

PIFR1 (0x4000_0208)

	7	6	5	4	3	2	1	0
Bit Symbol	PI7F1	PI6F1	PI5F1	PI4F1	PI3F1	PI2F1	PI1F1	PI0F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1: TB4IN1	0:PORT 1: TB4IN0	0:PORT 1: TB5OUT	0:PORT 1: TB4OUT	0:PORT 1: TB3OUT	0:PORT 1: TB2OUT	0:PORT 1: TB1OUT	0:PORT 1: TB0OUT

Port I pull-up control register

PIPUP (0x4000_022C)

	7	6	5	4	3	2	1	0
Bit Symbol	PI7UP	PI6UP	PI5UP	PI4UP	PI3UP	PI2UP	PI1UP	PI0UP
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on

Port I input enable control register

PIIE (0x4000_0238)

	7	6	5	4	3	2	1	0
Bit Symbol	PI7IE	PI6IE	PI5IE	PI4IE	PI3IE	PI2IE	PI1IE	PI0IE
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.12 Port J (PJ0~PJ7)

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

To use the external interrupt input for releasing STOP mode, select this function in the PJFR register and enable input in the PJIE register. These settings enable the interrupt input even if the STBYCR2<DRVE> bit in the clock/mode control block is set to stop driving of pins during STOP mode.

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

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

Port J Circuit Type

	7	6	5	4	3	2	1	0
Type	T7	T7	T9	T9	T7	T7	T7	T7

Port J register

PJ (0x4000_0240)

	7	6	5	4	3	2	1	0
Bit Symbol	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
Read/Write	R/W							
After reset	"0"							

Port J control register

PJCR (0x4000_0244)

	7	6	5	4	3	2	1	0
Bit Symbol	PJ7C	PJ6C	PJ5C	PJ4C	PJ3C	PJ2C	PJ1C	PJ0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port J function register 1

PJFR1 (0x4000_0248)

	7	6	5	4	3	2	1	0
Bit Symbol	PJ7F1	PJ6F1	PJ5F1	PJ4F1	PJ3F1	PJ2F1	PJ1F1	PJ0F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1: INT7	0:PORT 1: INT6	0:PORT 1: TB7OUT	0:PORT 1: TB6OUT	0:PORT 1: INT3	0:PORT 1: INT2	0:PORT 1: INT1	0:PORT 1: INT0

Port J pull-up control register

PJPUP (0x4000_026C)

	7	6	5	4	3	2	1	0
Bit Symbol	PJ7UP	PJ6UP	PJ5UP	PJ4UP	PJ3UP	PJ2UP	PJ1UP	PJ0UP
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on	Pull-up 0:off 1:on

Port J input enable control register

PJIE (0x4000_0278)

	7	6	5	4	3	2	1	0
Bit Symbol	PJ7IE	PJ6IE	PJ5IE	PJ4IE	PJ3IE	PJ2IE	PJ1IE	PJ0IE
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.2.13 Port K (PK0~PK2)

The port K is a general-purpose, 3-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Besides the general-purpose port function, the port K performs the functions of the 16-bit timer output, the CEC input, the clock output and the alarm output.

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

(Note) PK0 is an Nch open drain port.

Port K Circuit Type

	7	6	5	4	3	2	1	0
Type	—	—	—	—	—	T9	T15	T14

Port K register

PK (0x4000_0280)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	PK2	PK1	PK0
Read/Write	R					R/W		
After reset	"0" is read.					"0"		

Port K control register

PKCR (0x4000_0284)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	PK2C	PK1C	PK0C
Read/Write	R					R/W		
After reset	0					0	0	0
Function	"0" is read.					0: Output disabled 1: Output enabled		

Port K function register 1

PKFR1 (0x4000_0288)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	PK2F1	PK1F1	PK0F1
Read/Write	R					R/W		
After reset	0					0	0	0
Function	"0" is read.					0:PORT 1: TB9OUT	0:PORT 1: SCOUT	0:PORT 1: CEC

Port K function register 2

PKFR2 (0x4000_028C)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	—	PK2F1	—
Read/Write	R						R/W	R
After reset	0						0	0
Function	"0" is read.						0:PORT 1: ALARM	"0" is read.

Port K pull-up control register

PKPUP (0x4000_02AC)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	PJ2UP	PJ1UP	—
Read/Write	R					R/W		R
After reset	0					0	0	0
Function	"0" is read.					Pull-up 0:off 1:on	Pull-up 0:off 1:on	"0" is read.

Port K input enable control register

PKIE (0x4000_02B8)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	PK2IE	PK1IE	PK0IE
Read/Write	R					R/W		
After reset	0					0	0	0
Function	"0" is read.					Input 0:disabled 1:enabled	Input 0:disabled 1:enabled	Input 0:disabled 1:enabled

9.3 Block Diagrams of Ports

9.3.1 Port Types

The ports are classified into 18 types shown below. Please refer to the following pages for the block diagrams of each port type.

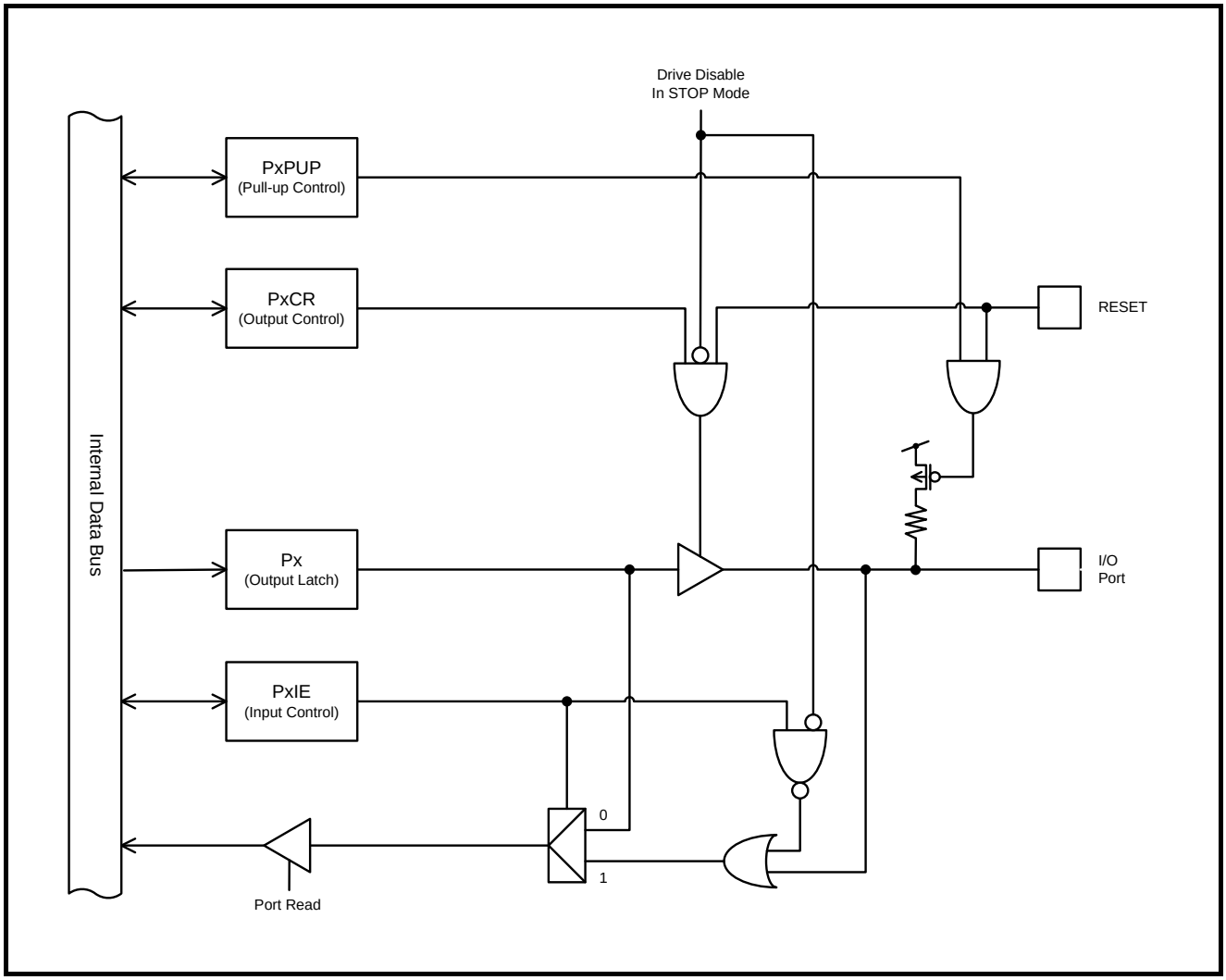
Type	GP port	Function 1	Function 2	Analog	Pull up	Pull down	OD	Note
T1	i/o	-	-	-	R	-	-	
T2	i/o	i	-	-	NoR	-	-	
T3	i/o	i	-	-	R	-	-	
T4	i/o	i	-	-	R	-	○	
T5	i/o	i	-	-	NoR	-	-	BOOT input enabled during reset
T6	i/o	i	-	-	-	NoR	-	
T7	i/o	i(int)	-	-	R	-	-	
T8	i/o	i(int)	-	-	R	-	○	
T9	i/o	o	-	-	R	-	-	
T10	i/o	o	-	-	R	-	○	
T11	i/o	o	-	-	R	-	-	Function output triggered by enable signal
T12	i/o	i/o	-	-	NoR	-	-	Function output triggered by enable signal
T13	i/o	i/o	-	-	R	-	○	
T14	i/o	i/o	-	-	-	-	●	Nch open drain port
T15	i/o	o	o	-	R	-	-	
T16	i/o	i/o	i	-	R	-	○	
T17	i	-	-	○	R	-	-	
T18	i	i	-	○	R	-	-	

R : Forced disable during reset.
NoR : Unaffected by reset.

9.3.2 Type T1

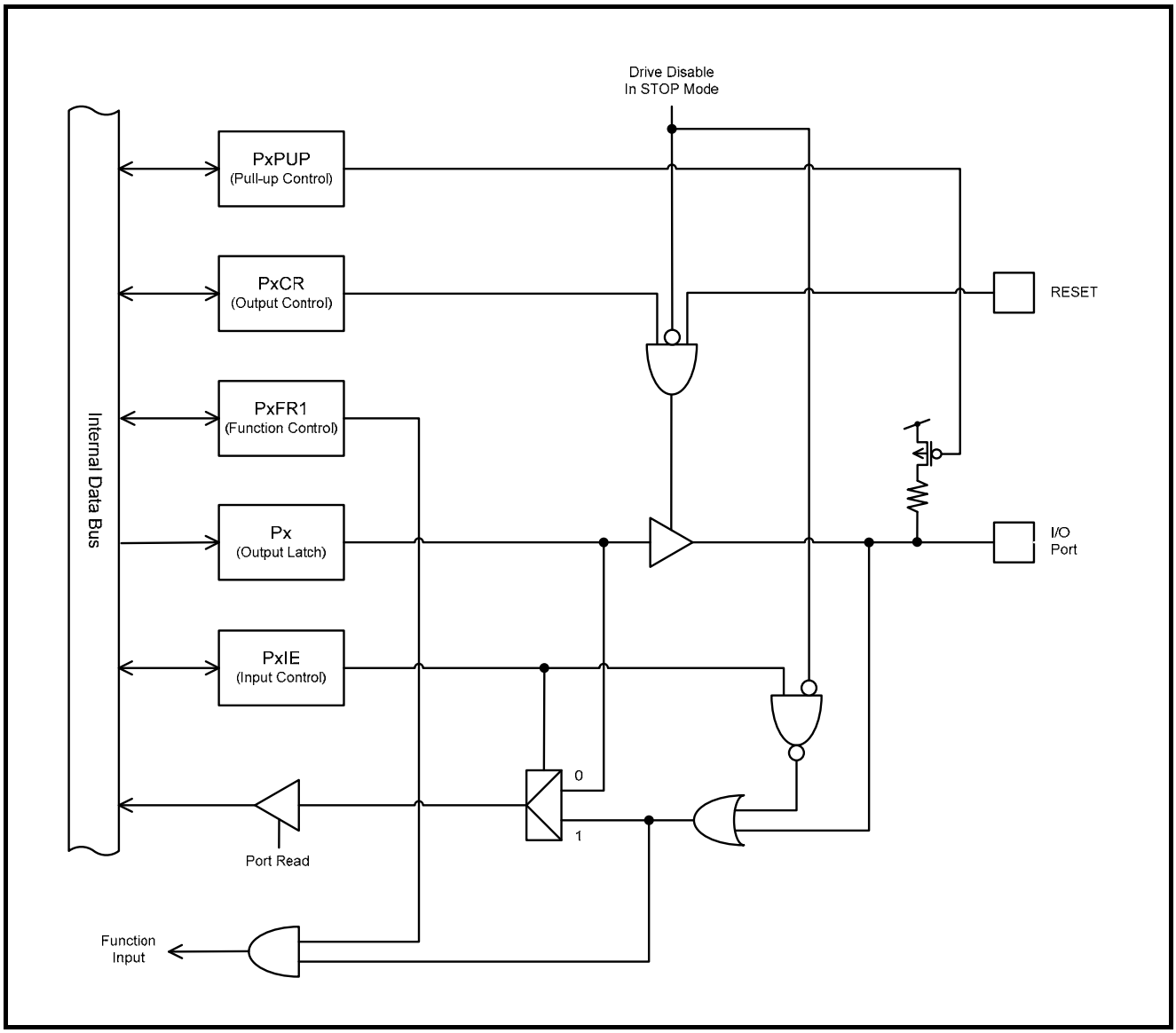
Type T1 is a general-purpose input/ output port with pull-up.

Pull-up and output are disabled during reset.



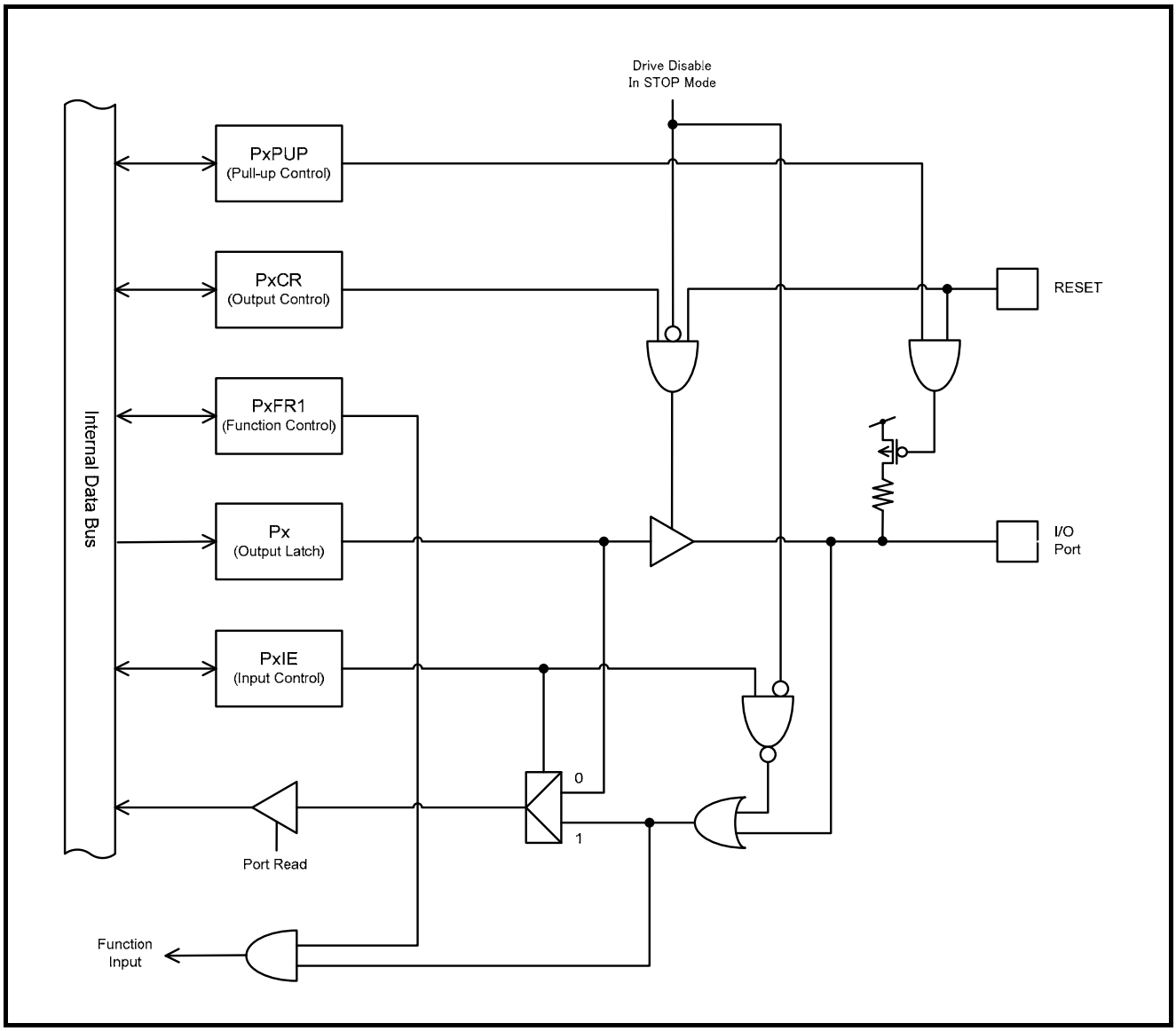
9.3.3 Type T2

Type T2 is a general-purpose input/ output port with pull-up. It is used to input function data as well. Output is disabled during reset.



9.3.4 Type T3

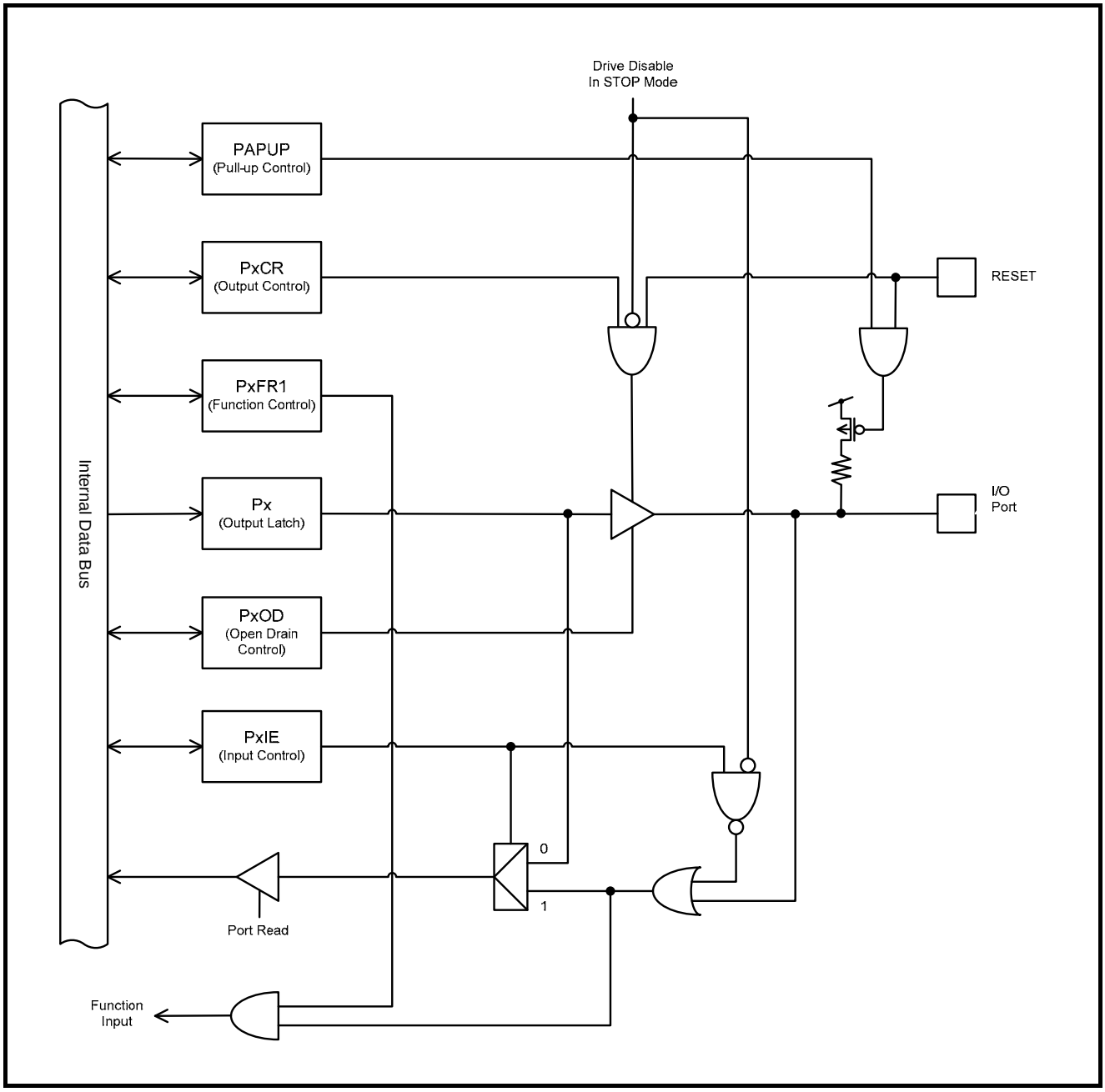
Type T3 is a general-purpose input/ output port with pull-up. It is used to input function data as well. Pull-up and output are disabled during reset.



9.3.5 Type T4

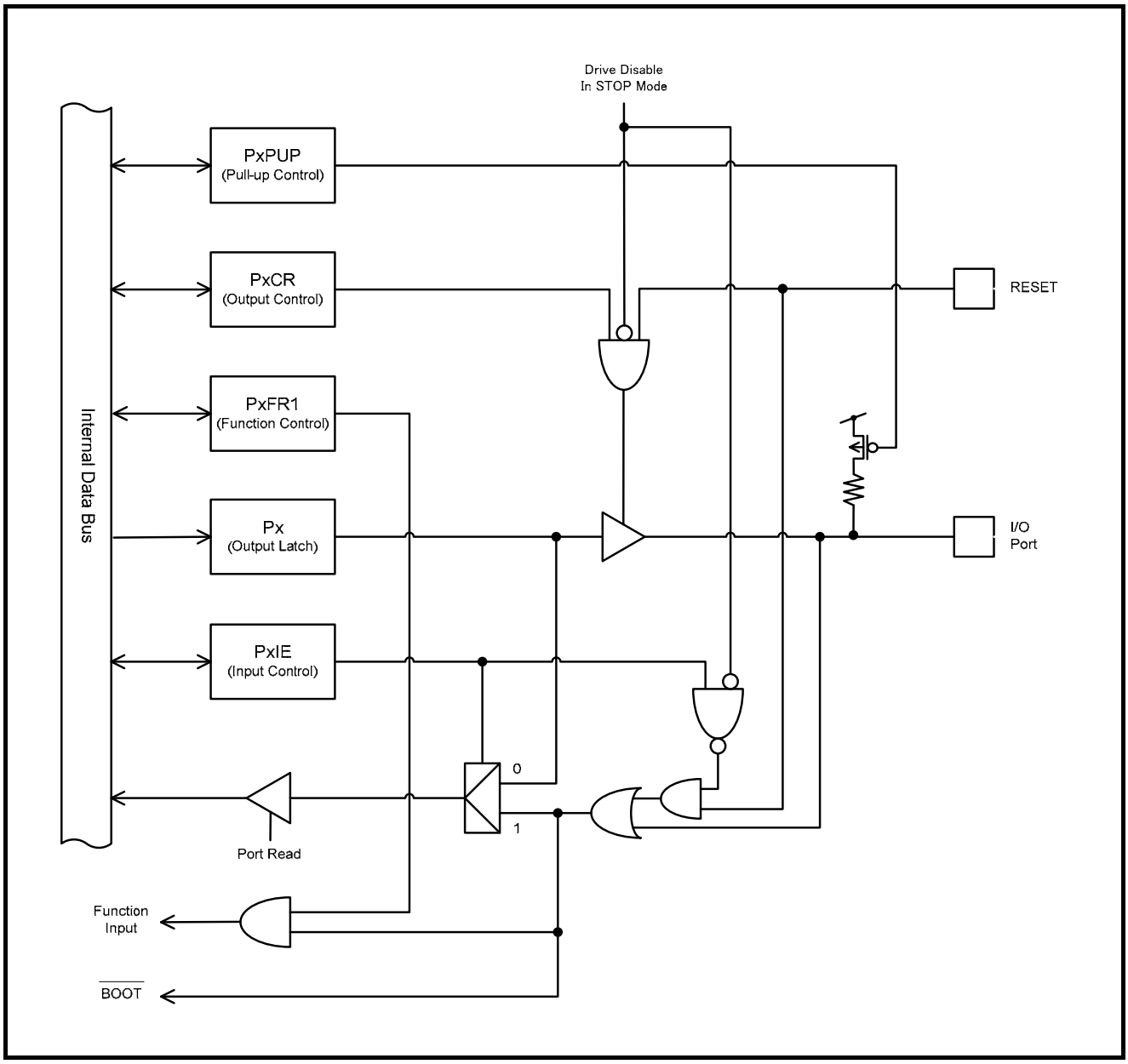
Type T4 is a general-purpose input/ output port with open drain. It is used to input function data as well.

Pull-up and output are disabled during reset.



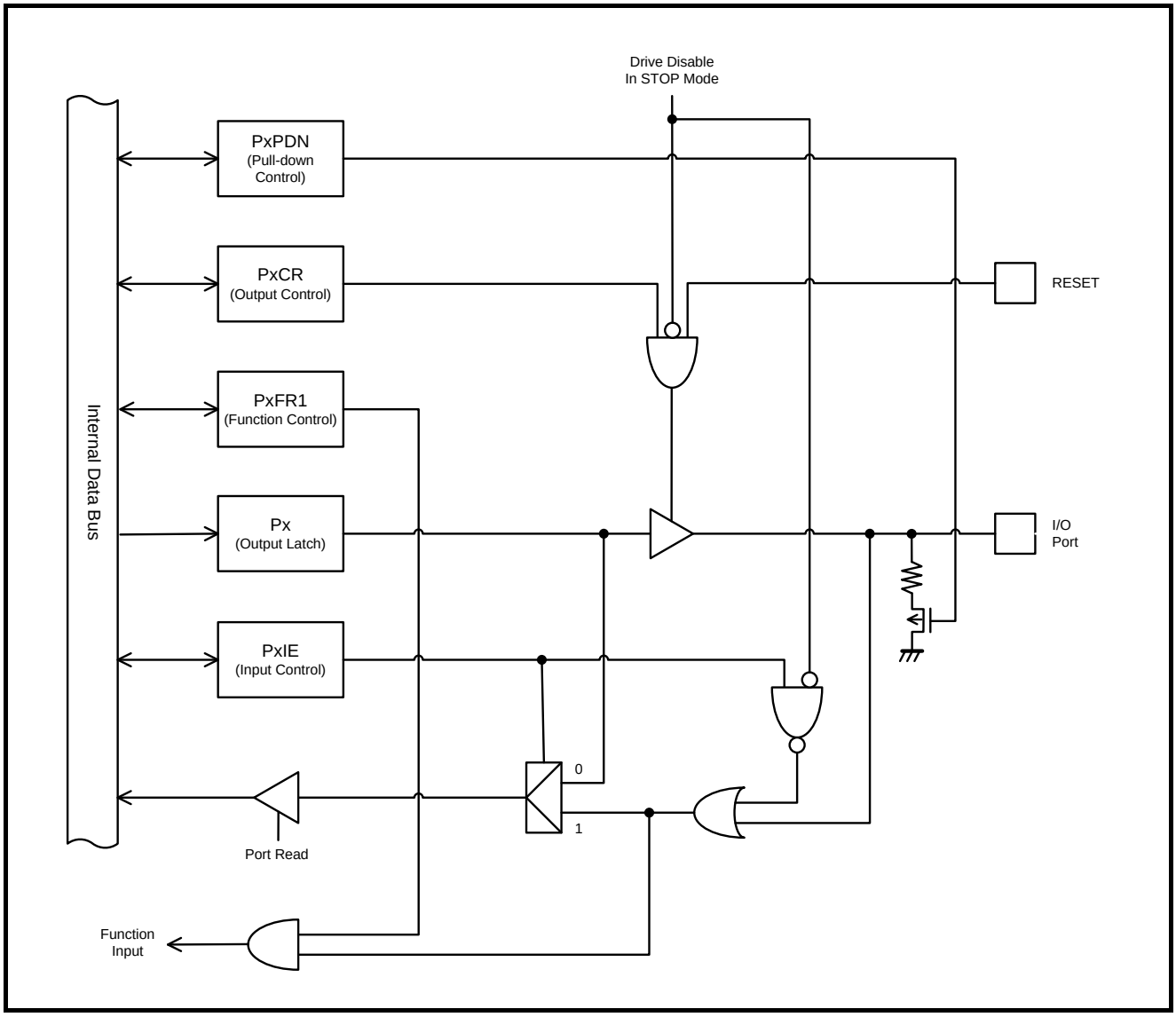
9.3.6 Type T5

Type T5 is a general-purpose input/ output port with pull-up. It is used to input function data as well. During reset, it functions as an input port for a $\overline{\text{BOOT}}$ signal and pull-up and output are disabled.



9.3.7 Type T6

Type T6 is a general-purpose port with pull-down. It is used to input function data as well.
Output is disabled during reset.



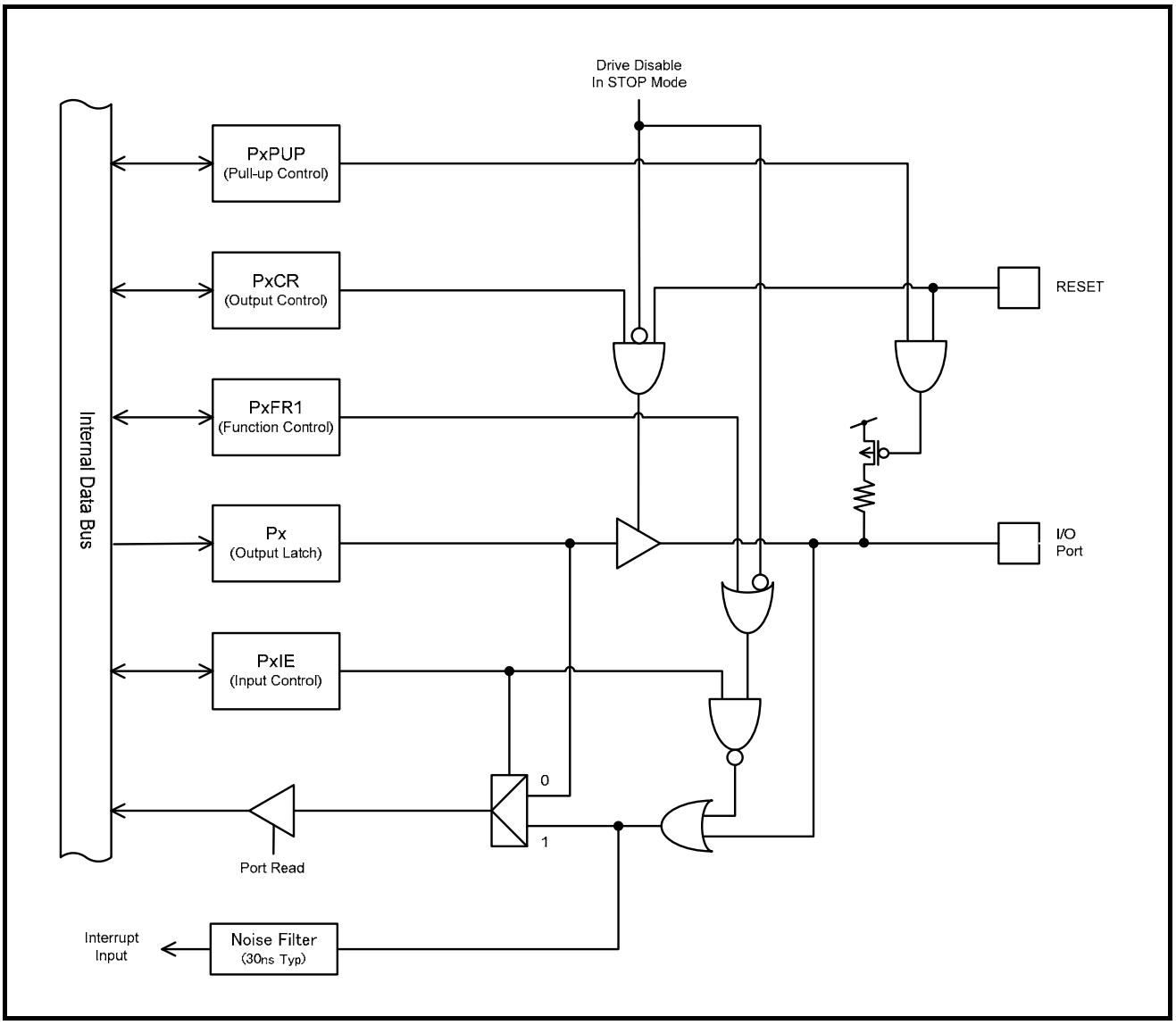
9.3.8 Type T7

Type T7 is a general-purpose input/ output port with pull-up. It is used to input interrupts as well.

Pull-up and output are disabled during reset.

To use the external interrupt input for releasing STOP mode, select this function in the PxFR register and enable input in the PxIE register. These settings enable the interrupt input even if the STBYCR2<DRVE> bit in the clock/mode control block is set to stop driving of pins during STOP mode.

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



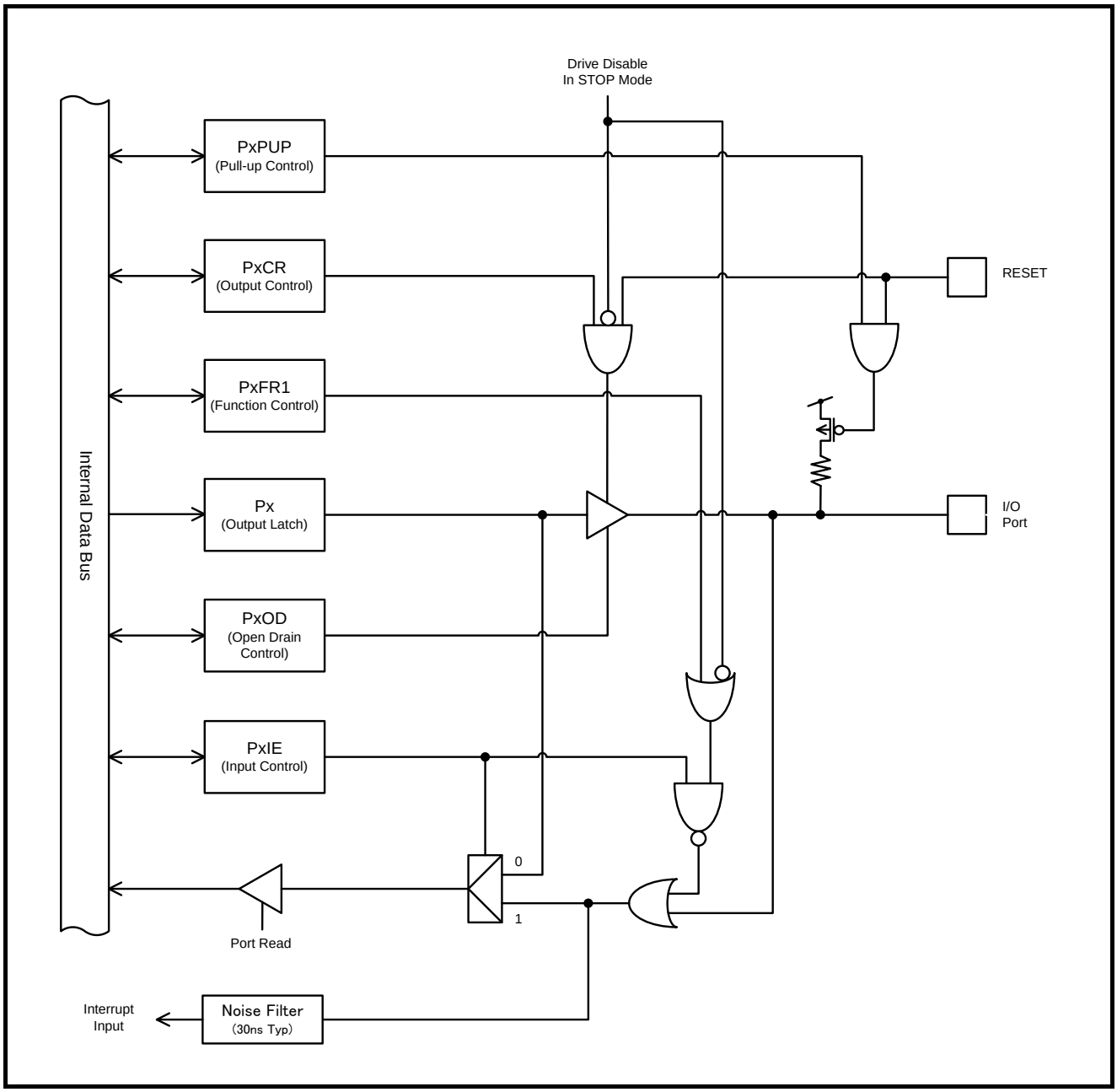
9.3.9 Type T8

Type T8 is a general-purpose input/ output port with pull-up and open drain. It is used to input interrupts as well.

Pull-up and output are disabled during reset.

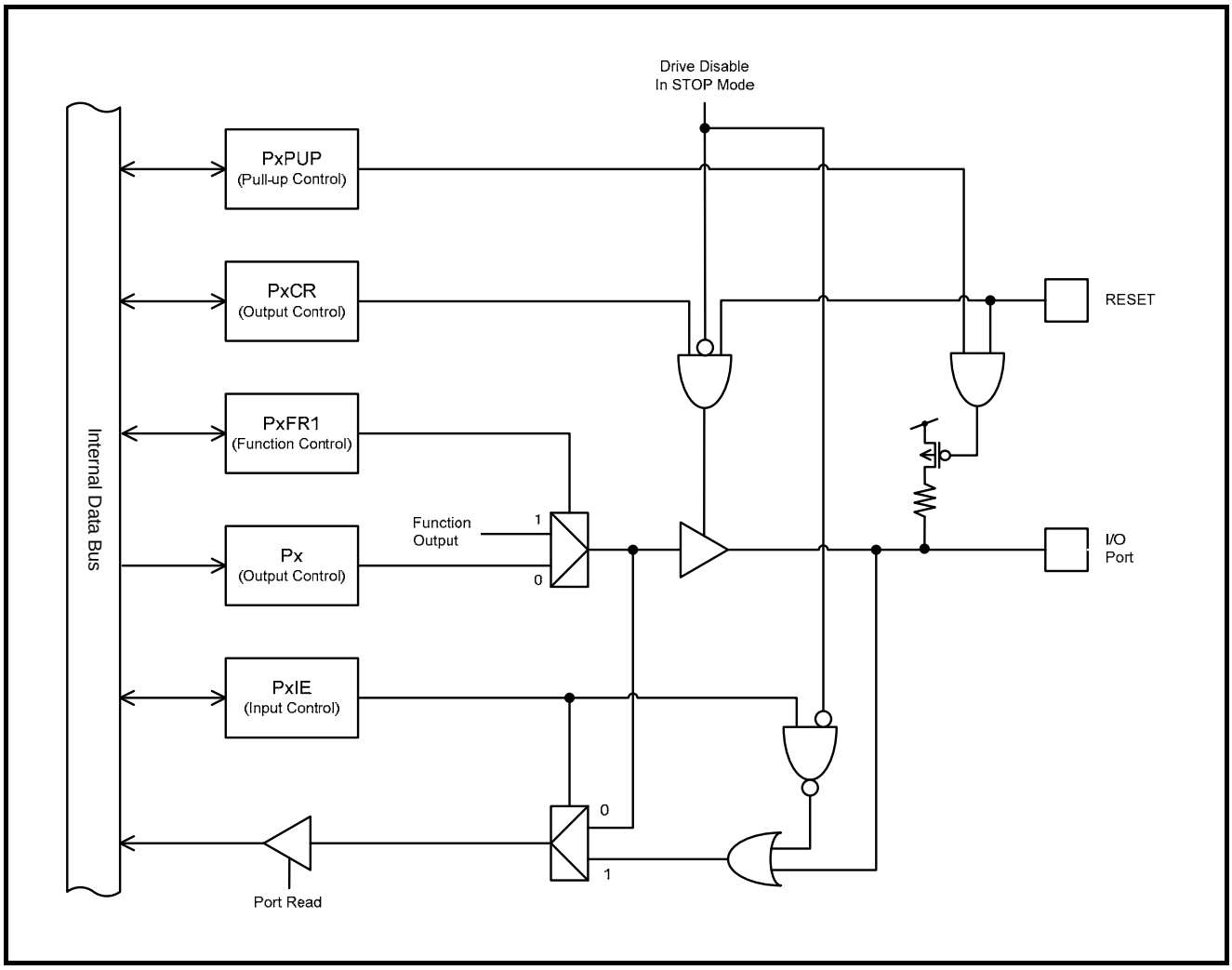
To use the external interrupt input for releasing STOP mode, select this function in the PxFR register and enable input in the PxIE register. These settings enable the interrupt input even if the STBYCR2<DRVE> bit in the clock/ mode control block is set to stop driving of pins during STOP mode.

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



9.3.10 Type T9

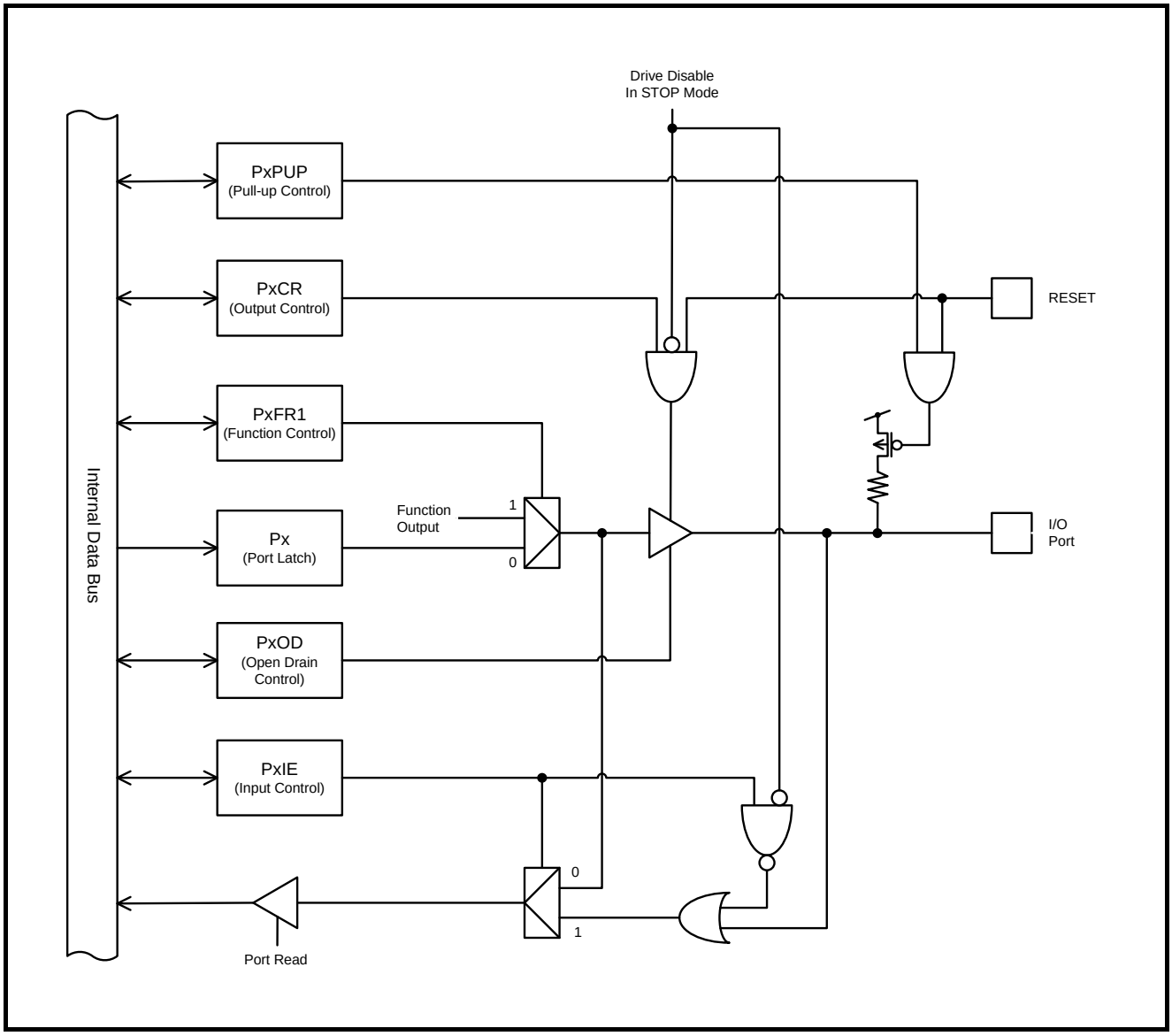
Type T9 is a general-purpose input/ output port with pull-up. It is used to output function data as well. Pull-up and output are disabled during reset.



9.3.11 Type T10

Type T10 is a general-purpose input/ output port with pull-up and open drain. It is used to output function data as well.

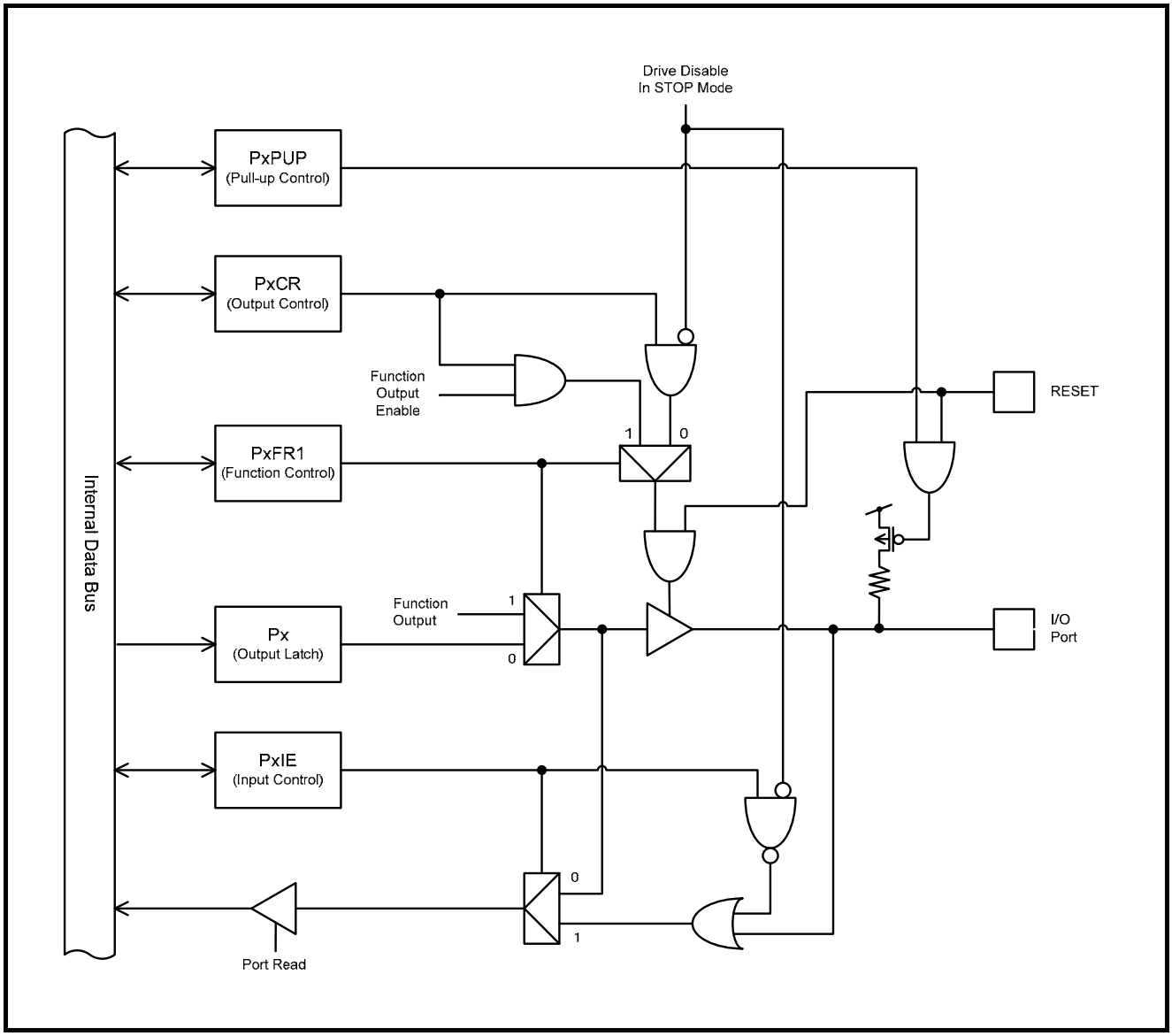
Pull-up and output are disabled during reset.



9.3.12 Type T11

Type T11 is a general-purpose input/ output port with pull-up. It is used to output function data as well. The function output is controlled by an enable signal. If enabled, the function data is output.

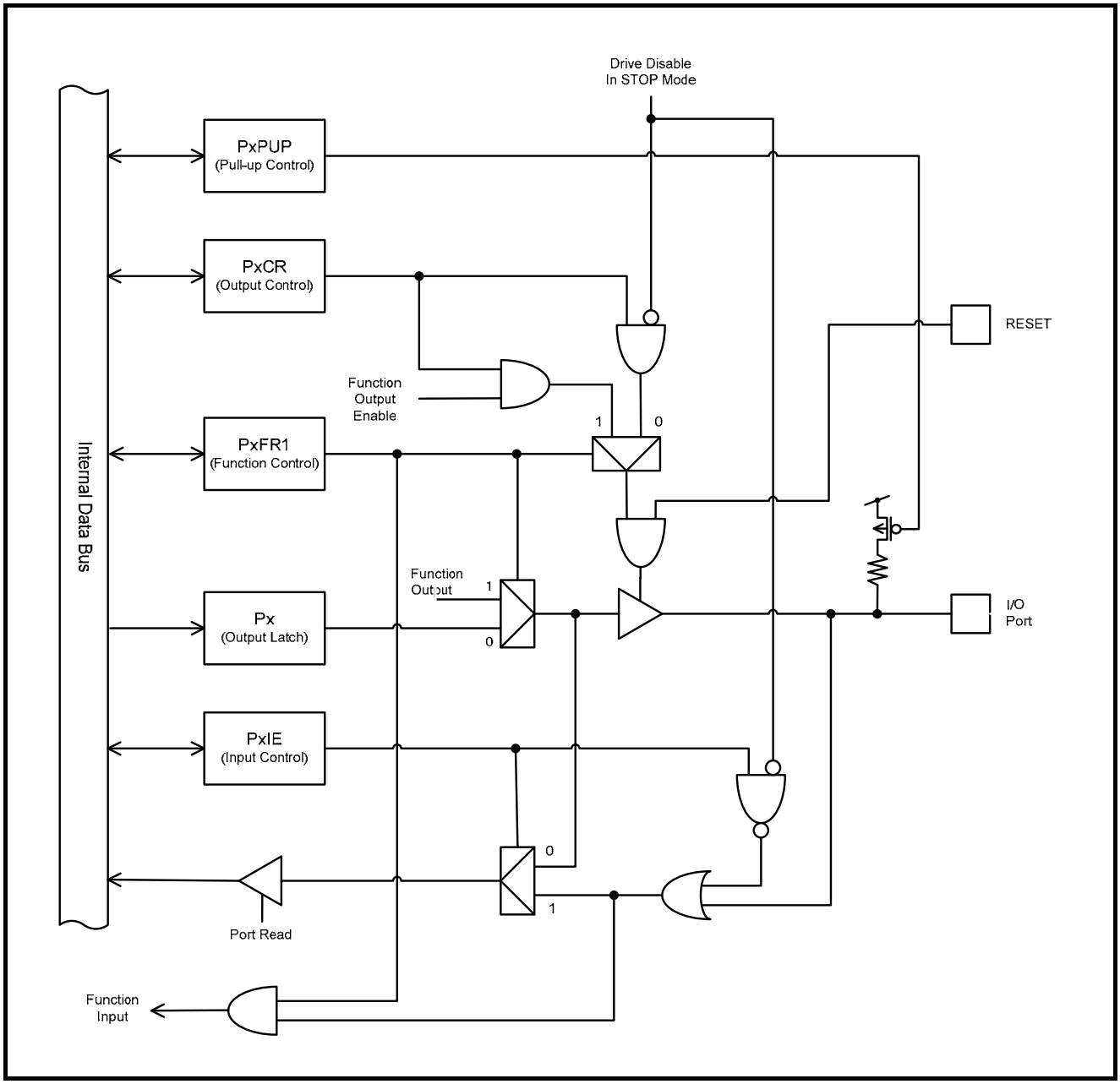
Pull-up and output are disabled during reset.



9.3.13 Type T12

Type T12 is a general-purpose input/ output port with pull-up. It is used to input/ output function data as well. The function output is controlled by an enable signal. If enabled, the function data is output.

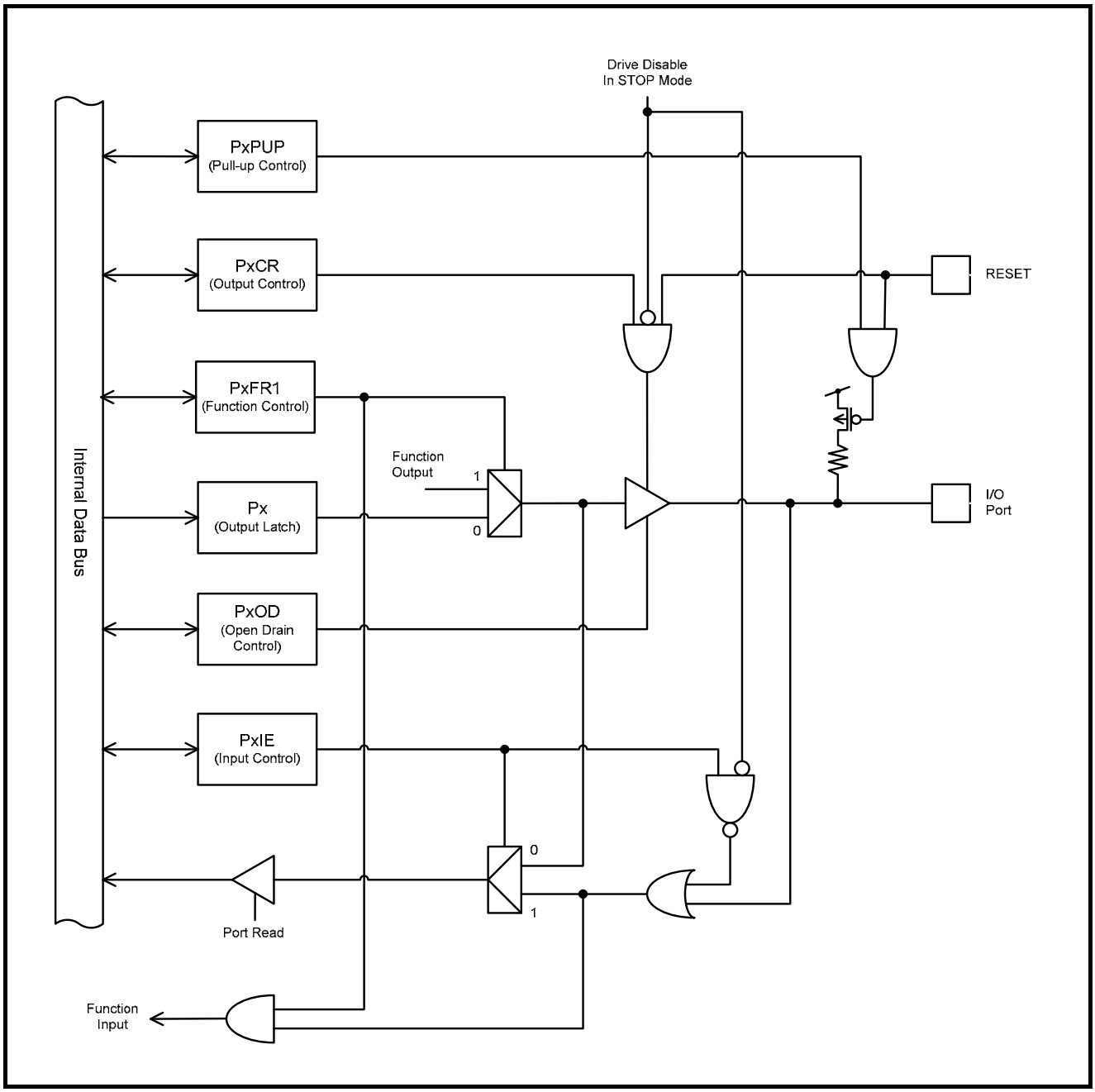
Output is disabled during reset.



9.3.14 Type T13

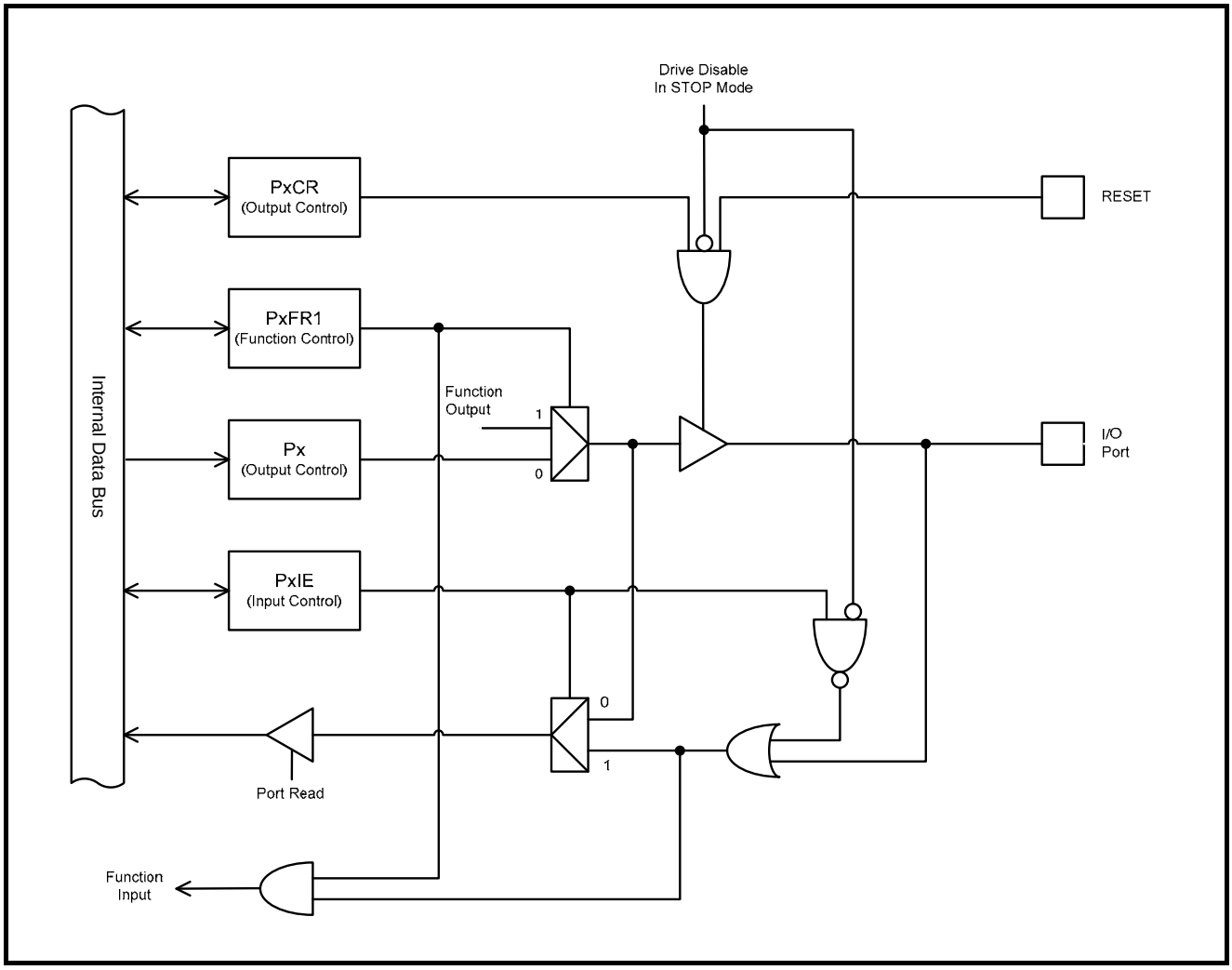
Type T13 is a general-purpose input/ output port with pull-up and open drain. It is used to input/output function data as well.

Pull-up and output are disabled during reset.



9.3.15 Type T14

Type T14 is a general-purpose input/ output port. It is used to input/output function data as well. Output is disabled during reset.

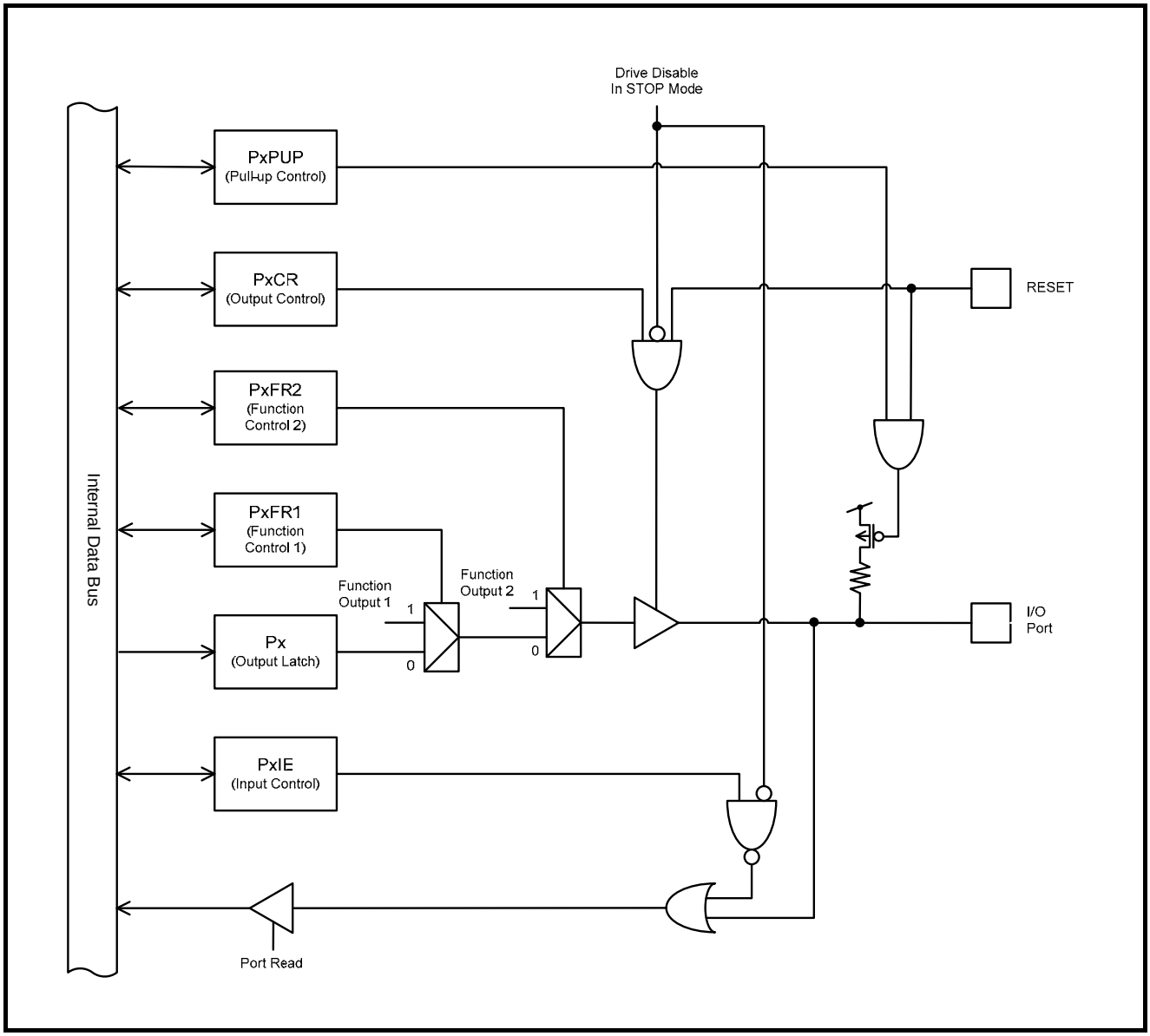


(Note) PK0 that uses Type 14 is an Nch open drain port.

9.3.16 Type T15

Type T15 is a general-purpose input/ output port with pull-up. It is used to output two kinds of function data as well.

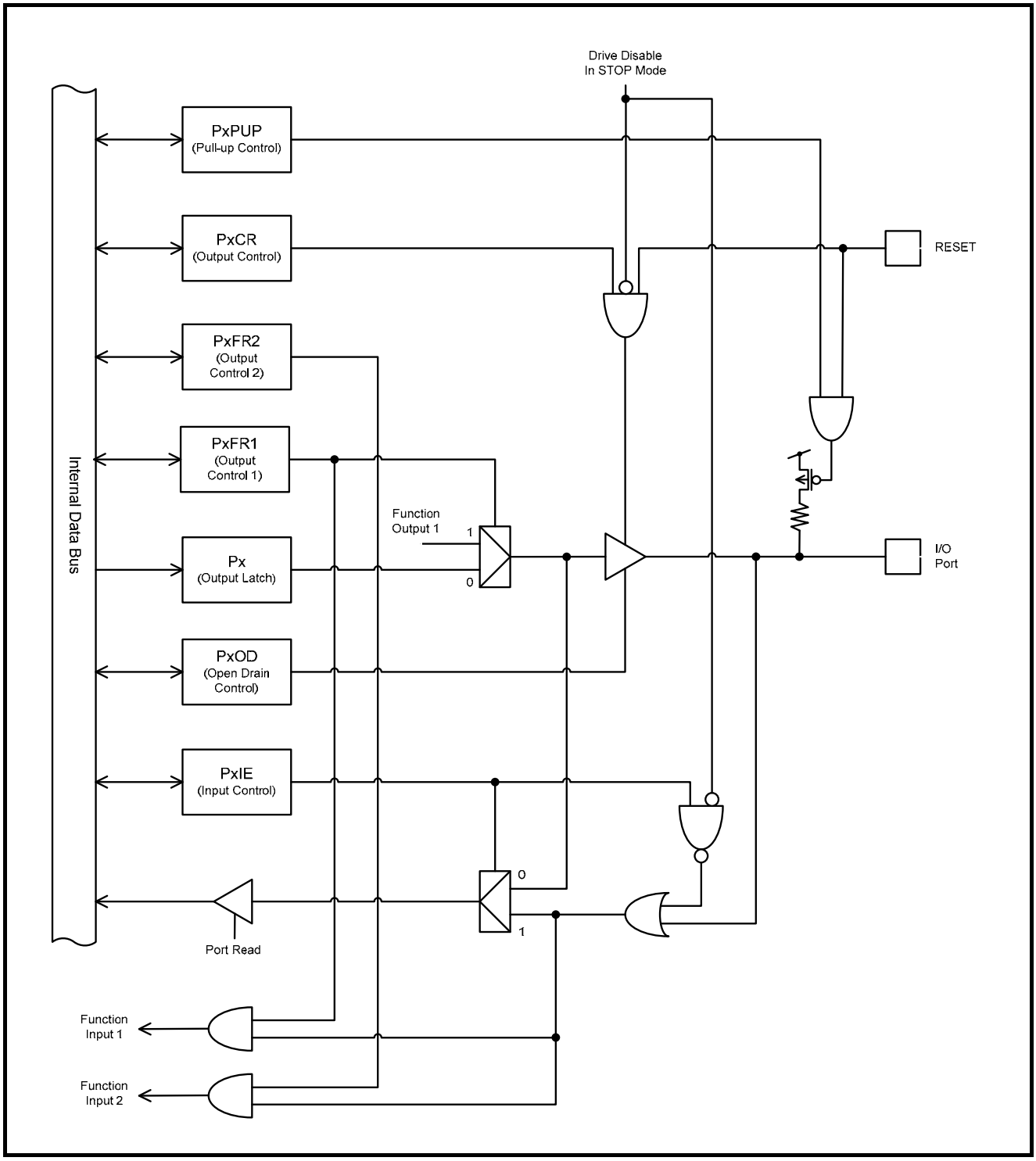
Pull-up and output are disabled during reset.



9.3.17 Type T16

Type T16 is a general-purpose input/ output port with pull-up and open drain. It is used to communicate two kinds of function data (function 1: input and output, function 2: input) as well.

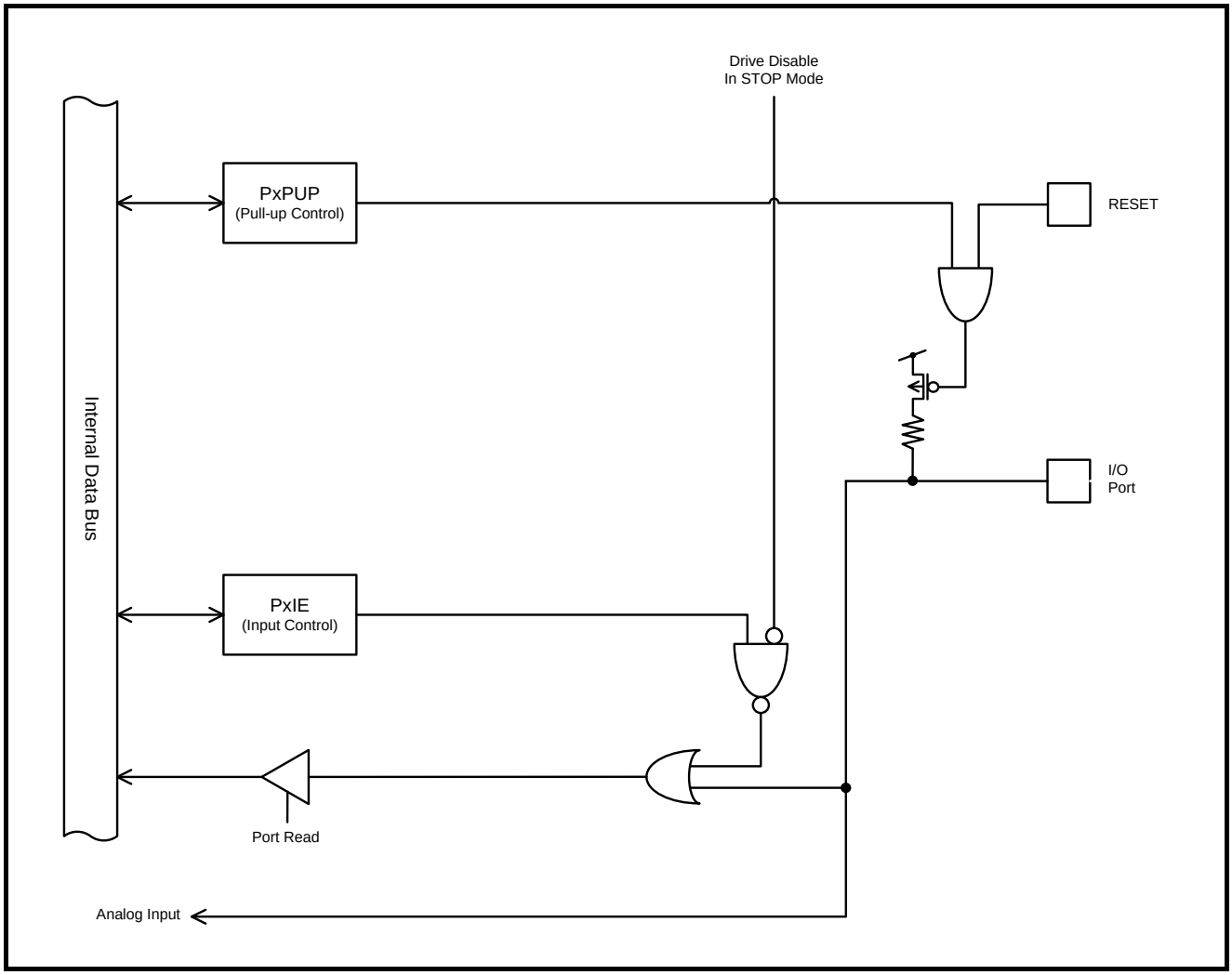
Pull-up and output are disabled during reset.



9.3.18 Type T17

Type 17 is a general-purpose input port with pull-up. It is used to input analog signals for A/D converter.

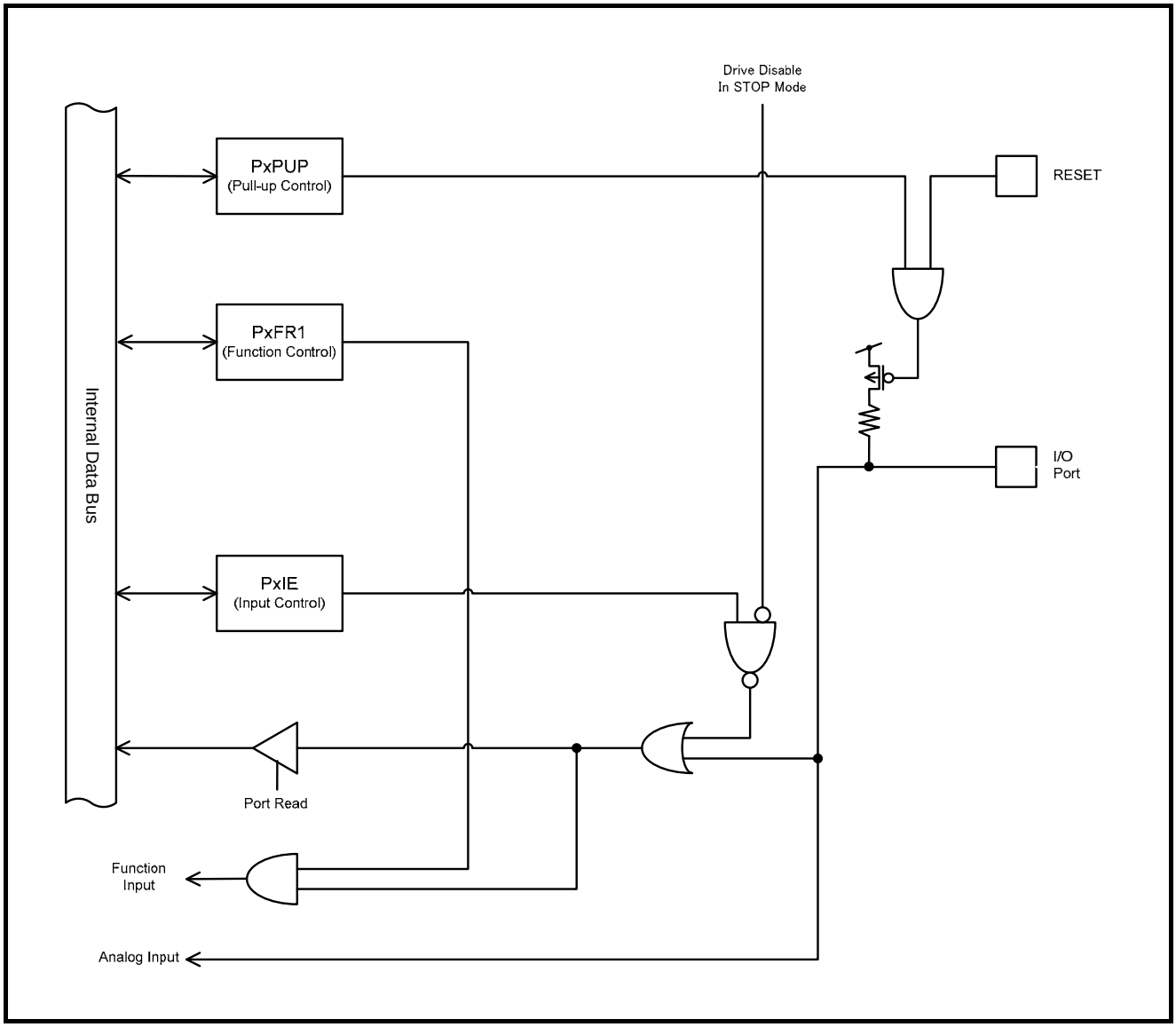
Pull-up is disabled during reset.



9.3.19 Type T18

Type 18 is a general-purpose input port with pull-up. It is used to input function data and analog signals for A/D converter as well.

Pull-up is disabled during reset.



10 16-bit Timer/Event Counters (TMRBs)

10.1 Outline

Each of the ten channels (TMRB0 through TMRB9) has a multi-functional 16-bit timer/event counter. TMRBs operate in the following four operation modes:

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable square-wave output mode (PPG)
- Timer synchronous mode (capable of setting output mode for each 4ch)

The use of the capture function allows TMRBs to perform the following three measurements

- Frequency measurement
- Pulse width measurement
- Time difference measurement

10.2 Differences in the Specifications

Each channel (TMRB0 through TMRB9) functions independently and the channels operate in the same way, except for the differences in their specifications as shown in Table 10-1 and Table 10-2 and the two-phase pulse count function. Therefore, the operational descriptions here are only for TMRB0.

The channels shown below are used as the capture or start trigger.

(1) The flip-flop output of TMRB 7 through TMRB 9 can be used as the capture trigger of other channels.
- TB7OUT => available for TMRB 0 through TMRB 1
- TB8OUT => available for TMRB 2 through TMRB 4
- TB9OUT => available for TMRB 5 through TMRB 6

(2) The start trigger of the timer synchronous mode (with TBxRUN)
- TMRB0 => can start TMRB 0 through TMRB 3 synchronously
- TMRB4 => can start TMRB 4 through TMRB 7 synchronously

Table 10-1 Differences in the Specifications of TMRB Modules (1)

Specification / Channel	External pins		Trigger	
	External clock/ capture trigger input pins	Timer flip-flop output pin	Timer for capture triggers	Timer for synchronous stat triggers
TMRB0	TB0IN0 (shared with PH0) TB0IN1 (shared with PH1)	TB0OUT (shared with PI0)	TB7OUT	–
TMRB1	TB1IN0 (shared with PH2) TB1IN1 (shared with PH3)	TB1OUT (shared with PI1)	TB7OUT	TB0PRUN,TB0RUN
TMRB2	TB2IN0 (shared with PH4) TB2IN1 (shared with PH5)	TB2OUT (shared with PI2)	TB8OUT	TB0PRUN,TB0RUN
TMRB3	TB3IN0 (shared with PH6) TB3IN1 (shared with PH7)	TB3OUT (shared with PI3)	TB8OUT	TB0PRUN,TB0RUN
TMRB4	TB4IN0 (shared with PI6) TB4IN1 (shared with PI7)	TB4OUT (shared with PI4)	TB8OUT	–
TMRB5	TB5IN0 (shared with PD0) TB5IN1 (shared with PD1)	TB5OUT (shared with PI5)	TB9OUT	TB4PRUN,TB4RUN
TMRB6	TB6IN0 (shared with PD2) TB6IN1 (shared with PD3)	TB6OUT (shared with PJ4)	TB9OUT	TB4PRUN,TB4RUN
TMRB7	–	TB7OUT (shared with PJ5)	–	TB4PRUN,TB4RUN
TMRB8	–	TB8OUT (shared with PG7)	–	–
TMRB9	–	TB9OUT (shared with PK2)	–	–

Table 10-2 Differences in the Specifications of TMRB Modules (2)

Specification / Channel	Interrupt	
	Capture interrupt	TMRB interrupt
TMRB0	INTCAP00 INTCAP01	INTTB0
TMRB1	INTCAP10 INTCAP11	INTTB1
TMRB2	INTCAP20 INTCAP21	INTTB2
TMRB3	INTCAP30 INTCAP31	INTTB3
TMRB4	INTCAP40 INTCAP40	INTTB4
TMRB5	INTCAP50 INTCAP51	INTTB5
TMRB6	INTCAP60 INTCAP61	INTTB6
TMRB7	–	INTTB7
TMRB8	–	INTTB8
TMRB9	–	INTTB9

10.3 Configuration

Each channel consists of a 16-bit up-counter, two 16-bit timer registers (one of which is double-buffered, except for TBRB0 with one double-buffered 16-bit timer register), 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.

(Note 1) TB7OUT is input to channels 0 and 1. TB8OUT is input to channels 2 through 4. TB9OUT is input to channels 5 and 6.

(Note 2) Please note that channels 7 through 9:
- do not output TBxOUT to an external pin.
- do not input TBnIN0 and TBnIN1 from an external pin.
- cannot use capture interrupt of INTCAPn0 and INTCAPn1.

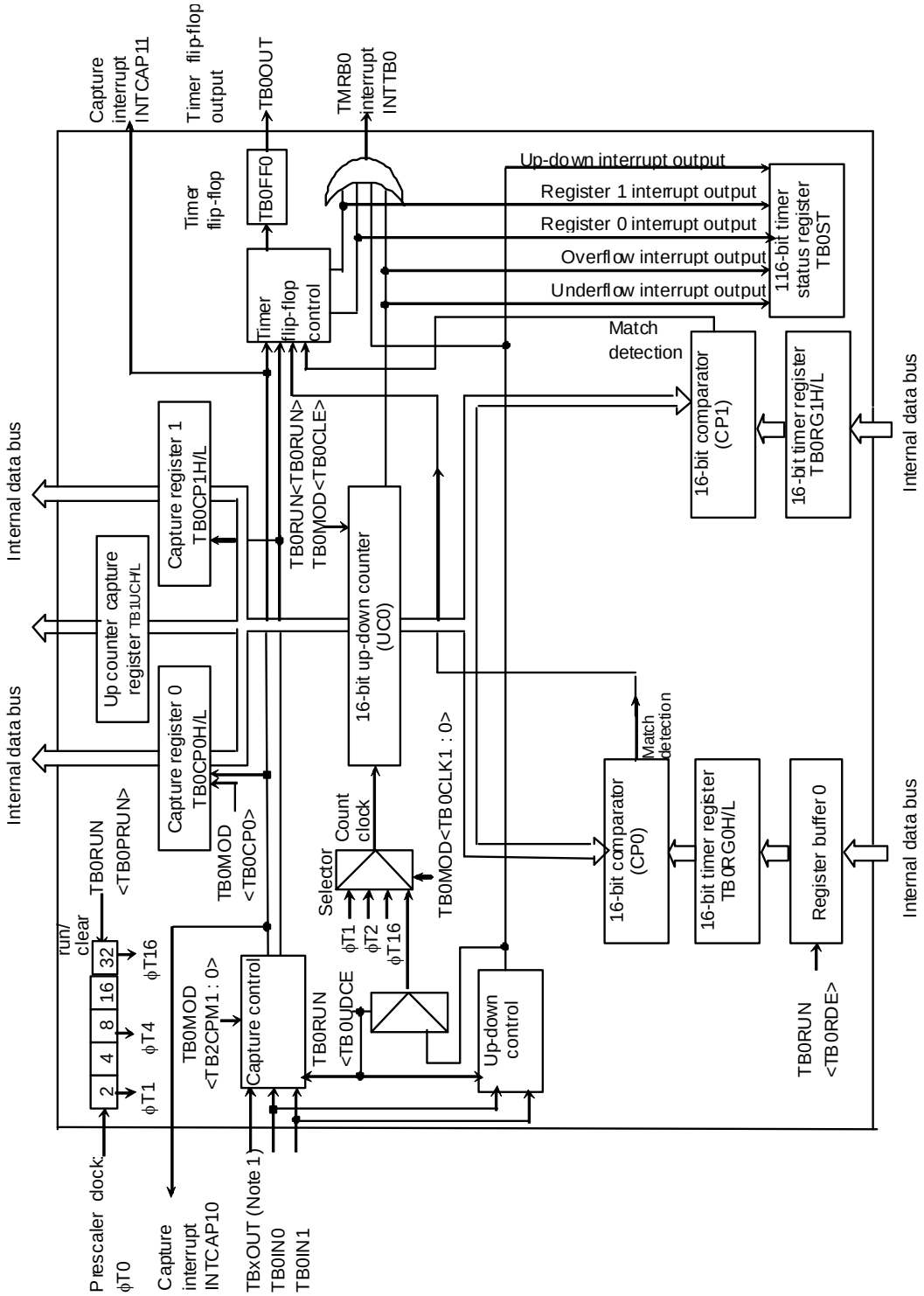


Fig. 10-1 TMRB1 Block Diagram (the same applies to channels 0 and 2 through 9)

10.4 Registers

10.4.1 TMRB registers

Table 10-3 shows the register names and addresses of each channel.

Table 10-3 TMRB registers (1/2)

Specification \ Channel		TMRB0		TMRB1		TMRB2		TMRB3	
Register names (addresses)	Timer enable register	TB0EN	0x4001_0000	TB1EN	0x4001_0040	TB2EN	0x4001_0080	TB3EN	0x4001_00C0
	Timer RUN register	TB0RUN	0x4001_0004	TB1RUN	0x4001_0044	TB2RUN	0x4001_0084	TB3RUN	0x4001_00C4
	Timer control register	TB0CR	0x4001_0008	TB1CR	0x4001_0048	TB2CR	0x4001_0088	TB3CR	0x4001_00C8
	Timer mode register	TB0MOD	0x4001_000C	TB1MOD	0x4001_004C	TB2MOD	0x4001_008C	TB3MOD	0x4001_00CC
	Timer flip-flop control register	TB0FFCR	0x4001_0010	TB1FFCR	0x4001_0050	TB2FFCR	0x4001_0090	TB3FFCR	0x4001_00D0
	Timer status register	TB0ST	0x4001_0014	TB1ST	0x4001_0054	TB2ST	0x4001_0094	TB3ST	0x4001_00D4
	Interrupt mask register	TB0IM	0x4001_0018	TB1IM	0x4001_0058	TB2IM	0x4001_0098	TB3IM	0x4001_00D8
	Timer up counter register	TB0UC	0x4001_001C	TB1UC	0x4001_005C	TB2UC	0x4001_009C	TB3UC	0x4001_00DC
	Timer register	TB0RG0 TB0RG1	0x4001_0020 0x4001_0024	TB1RG0 TB1RG1	0x4001_0060 0x4001_0064	TB2RG0 TB2RG1	0x4001_00A0 0x4001_00A4	TB3RG0 TB3RG1	0x4001_00E0 0x4001_00E4
	Capture register	TB0CP0 TB0CP1	0x4001_0028 0x4001_002C	TB1CP0 TB1CP1	0x4001_0068 0x4001_006C	TB2CP0 TB2CP1	0x4001_00A8 0x4001_00AC	TB3CP0 TB3CP1	0x4001_00E8 0x4001_00EC

Specification \ Channel		TMRB4		TMRB5		TMRB6		TMRB7	
Register names (addresses)	Timer enable register	TB4EN	0x4001_0100	TB5EN	0x4001_0140	TB6EN	0x4001_0180	TB7EN	0x4001_01C0
	Timer RUN register	TB4RUN	0x4001_0104	TB5RUN	0x4001_0144	TB6RUN	0x4001_0184	TB7RUN	0x4001_01C4
	Timer control register	TB4CR	0x4001_0108	TB5CR	0x4001_0148	TB6CR	0x4001_0188	TB7CR	0x4001_01C8
	Timer mode register	TB4MOD	0x4001_010C	TB5MOD	0x4001_014C	TB6MOD	0x4001_018C	TB7MOD	0x4001_01CC
	Timer flip-flop control register	TB4FFCR	0x4001_0110	TB5FFCR	0x4001_0150	TB6FFCR	0x4001_0190	TB7FFCR	0x4001_01D0
	Timer status register	TB4ST	0x4001_0114	TB5ST	0x4001_0154	TB6ST	0x4001_0194	TB7ST	0x4001_01D4
	Interrupt mask register	TB4IM	0x4001_0118	TB5IM	0x4001_0158	TB6IM	0x4001_0198	TB7IM	0x4001_01D8
	Timer up counter register	TB4UC	0x4001_011C	TB5UC	0x4001_015C	TB6UC	0x4001_019C	TB7UC	0x4001_01DC
	Timer register	TB4RG0 TB4RG1	0x4001_0120 0x4001_0124	TB5RG0 TB5RG1	0x4001_0160 0x4001_0164	TB6RG0 TB6RG1	0x4001_01A0 0x4001_01A4	TB7RG0 TB7RG1	0x4001_01E0 0x4001_01E4
	Capture register	TB4CP0 TB4CP1	0x4001_0128 0x4001_012C	TB5CP0 TB5CP1	0x4001_0168 0x4001_016C	TB6CP0 TB6CP1	0x4001_01A8 0x4001_01AC	TB7CP0 TB7CP1	0x4001_01E8 0x4001_01EC

Table 10-3 TMRB registers (2/2)

Specification \ Channel		TMRB8		TMRB9	
Register names (addresses)	Timer enable register	TB8EN	0x4001_0200	TB9EN	0x4001_0240
	Timer RUN register	TB8RUN	0x4001_0204	TB9RUN	0x4001_0244
	Timer control register	TB8CR	0x4001_0208	TB9CR	0x4001_0248
	Timer mode register	TB8MOD	0x4001_020C	TB9MOD	0x4001_024C
	Timer flip-flop control register	TB8FFCR	0x4001_0210	TB9FFCR	0x4001_0250
	Timer status register	TB8ST	0x4001_0214	TB9ST	0x4001_0254
	Interrupt mask register	TB8IM	0x4001_0218	TB9IM	0x4001_0258
	Timer up counter register	TB8UC	0x4001_021C	TB9UC	0x4001_025C
	Timer register	TB8RG0 TB8RG1	0x4001_0220 0x4001_0224	TB9RG0 TB9RG1	0x4001_0260 0x4001_0264
	Capture register	TB8CP0 TB8CP1	0x4001_0228 0x4001_022C	TB9CP0 TB9CP1	0x4001_0268 0x4001_026C

10.4.1.1 TMRBn enable register (channels 0 through 9)

TMRBn enable register (n=0~9)

TBnEN
(0x4001_0xx0)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit Symbol	TBnEN							
Read/Write	R/W	R						
After reset	0	0						
Function	TMRBn operation 0: Disabled 1: Enabled	“0” is read.						

＜TBnEN＞: 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.) 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.

10.4.1.2 TMRB RUN register (channels 0 through 9)

TMRBn RUN register (n=0~9)

TBnRUN
(0x4001_0xx4)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit Symbol						TBnPRUN		TBnRUN
Read/Write	R					R/W	R	R/W
After reset	0					0	0	0
Function	“0” is read.					Timer Run/Stop Control 0: Stop & clear 1: Count * The first bit can be read as “0.”		

＜TBnRUN＞ :Controls the TMRB0 count operation.

＜TBnPRUN＞:Controls the TMRB0 prescaler operation.

10.4.1.3 TMRB control register (channels 0 through 9)

TMRBn control register (n=0~9)

TBnCR
(0x4001_0xx8)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit Symbol	TBnWBF		TBnSYNC		I2TBn			
Read/Write	R/W	R/W	R/W	R	R/W	R		
After reset	0	0	0	0	0	0		
Function	Double Buffer 0: Disabled 1: Enabled	Write “0”.	Timer operation mode 0:individual 1:synchronous	“0” is read.	IDLE 0:Stop 1:Operation	“0” is read.		

＜I2TBm＞:Controls the operation in the IDLE mode.

＜TBnSYNC＞:Controls operation mode of timers.

“0”: timers operate individually.

“1”: timers operate synchronously.

＜TBmWBF＞:Controls the enabling/disabling of double buffering.

10.4.1.4 TMRB mode register (channels 0 thorough 9)

TMRBn mode register(n=0~9)

TBnMOD
(0x4001_0xxC)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit Symbol			TBnCP0	TBnCPM1	TBnCPM0	TBnCLE	TBnCLK1	TBnCLK0
Read/Write	R	R/W	W	R/W				
After reset	0	0	1	0	0	0	0	0
Function	"0" is read.	Write "0".	Capture control by software 0: Capture by software 1: Don't care	Capture timing 00: Disable Capture timing 00: Disable 01: TBnIN0 ↑ TBnIN1 ↑ 10: TBnIN0 ↑ TBnIN1 ↓ 11: TBnOUT ↑ TBnOUT ↓		Up-counter control 0: Clear/disable 1: clear/enable	Selects source clock 00: TBnIN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	

＜TBnCLK1:0＞:Selects the TMRBn timer count clock.

＜TBnCLE＞:Clears and controls the TMRBn up-counter.

"0": Disables clearing of the up-counter.

"1": Clears up-counter if there is a match with timer register 1 (TBnRG1).

＜TBnCPM1:0＞:Specifies TMRBn capture timing.

"00": Capture disable

"01": Takes count values into capture register 0 (TBnCP0) upon rising of TBnIN0 pin input. Takes count values into capture register 1 (TBnCP1) upon rising of TBnIN1 pin input.

"10": Takes count values into capture register 0 (TBnCP0) upon rising of TBnIN0 pin input. Takes count values into capture register 1 (TBnCP1) upon falling of TBnIN0 pin input.

"11":Takes count values into capture register 0 (TBnCP0) upon rising of 16-bit timer match output (TBxOUT) and into capture register 1 (TBnCP1) upon falling of TBxOUT (TMRB0 and TMRB1:TB7OUT, TMRB2 through TMRB4:TB8OUT, TMRB5 and TMRB6:TB9OUT).

＜TBnCP0＞:Captures count values by software and takes them into capture register 0 (TBnCP0).

(Note 1) The value read from bit 5 of TBnMOD is "1".

(Note 2) Input from TBnIN0 and TBnIN1 is available only for channels TMRB0 thorough 6.

10.4.1.5 TMRB flip-flop control register (channels 0 through 9)

TMRBn flip-flop control register (n=0~9)

TBnFFCR (0x4001_0xx0)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit Symbol			TBnC1T1	TBnC0T1	TBnE1T1	TBnE0T1	TBnFF0C1	TBnFF0C0
Read/Write	R		R/W				R/W	
After reset	1	1	0	0	0	0	1	1
Function	"11" is read.		TBnFF0 reverse trigger 0: Disable trigger 1: Enable trigger				TBnFF0 control 00: Invert 01: Set 10: Clear 11: Don't care * This is always read as "11."	
			When the up-counter value is taken into TBnCP1	When the up-counter value is taken into TBnCP0	When the up-counter matches TBnRG1	When the up-counter matches TBnRG0		

＜TBnFF0C1:0＞:Controls the timer flip-flop.

"00": Reverses the value of TBnFF0 (reverse by using software).

"01": Sets TBnFF0 to "1".

"10": Clears TBnFF0 to "0".

"11":Don't care

＜TBnE1:0＞:Reverses the timer flip-flop when the up-counter matches the timer register 0,1 (TBnRG0,1).

＜TBnC1:0＞:Reverses the timer flip-flop when the up-counter value is taken into the capture register 0,1 (TBnCP0,1).

10.4.1.6 TMRB status register (channels 0 through 9)

TMRBn status register (n=0~9)

TBnST (0x4001_0xx4)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit Symbol						INTTBOFn	INTTBn1	INTTBn0
Read/Write	R					R		
After reset	0					0	0	0
Function	“0” is read.					0: Interrupt not generated 1: Interrupt generated	0: Interrupt not generated 1: Interrupt generated	0: Interrupt not generated 1: Interrupt generated

＜INTTBn0＞:Interrupt generated if there is a match with timer register 0 (TBnRG0)

＜INTTBn1＞:Interrupt generated if there is a match with timer register 1 (TBnRG1)

＜INTTBOFn＞:Interrupt generated if an up-counter overflow occurs

(Note) If any interrupt is generated, the flag that corresponds to the interrupt is set to TBnST and the generation of interrupt is notified to the CPU. The flag is cleared by reading the TBnST register.

10.4.1.7 TMRB interrupt mask register (channels 0 through 9)

TMRBn interrupt mask register (n=0~9)

TBnIM
(0x4001_0xx8)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
bit Symbol						TBIMOFn	TBIMn1	TBIMn0
Read/Write	R					R		
After reset	0					0	0	0
Function	“0” is read					1: Interrupt is masked	1: Interrupt is masked	1: Interrupt is masked

＜TBIMn0＞: Interrupt is masked if there is a match with timer register 0 (TBnRG0)

＜TBIMn1＞:Interrupt is masked if there is a match with timer register 1 (TBnRG1).

＜TBIMOFn＞:Interrupt is masked if an up-and-down counter overflow occurs.

10.4.1.8 TMRB read capture register (channels 0 through 9)

TBnUC read capture register (n=0~9)

TBnUC
(0x4001_0xxC)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol	UCn15	UCn14	UCn13	UCn12	UCn11	UCn10	UCn9	UCn8
Read/Write	R							
After reset	0							
Function	Data obtained by read capture: 15-8 bit data							
	7	6	5	4	3	2	1	0
bit Symbol	UCn7	UCn6	UCn5	UCn4	UCn3	UCn2	UCn1	UCn0
Read/Write	R							
After reset	0							
Function	Data obtained by read capture: 7-0 bit							

10.4.1.9 TMRB timer register (channels 0 through 9)

TBnRG0 timer register (n=0~9)

TBnRG0
(0x4001_0xx0)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol	TBnRG015	TBnRG014	TBnRG013	TBnRG012	TBnRG011	TBnRG010	TBnRG09	TBnRG08
Read/Write	R/W							
After reset	0							
Function	Timer count value: 15-8 bit data							
	7	6	5	4	3	2	1	0
bit Symbol	TBnRG07	TBnRG06	TBnRG05	TBnRG04	TBnRG03	TBnRG02	TBnRG01	TBnRG00
Read/Write	R/W							
After reset	0							
Function	Timer count value: 7-0 bit data							

TBnRG1 timer register (n=0~9)

TBnRG1
(0x4001_0xx4)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol	TBnRG115	TBnRG114	TBnRG113	TBnRG112	TBnRG111	TBnRG110	TBnRG19	TBnRG18
Read/Write	R/W							
After reset	0							
Function	Timer count value: 15-8 bit data							
	7	6	5	4	3	2	1	0
bit Symbol	TBnRG17	TBnRG16	TBnRG15	TBnRG14	TBnRG13	TBnRG12	TBnRG11	TBnRG10
Read/Write	R/W							
After reset	0							
Function	Timer count value: 7-0 bit data							

10.4.1.10 TMRB capture register (channels 0 through 9)

TBnCP0capture register (n=0~9)

TBnCP0
(0x4001_0xx8)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol	TBnCP015	TBnCP014	TBnCP013	TBnCP012	TBnCP011	TBnCP010	TBnCP09	TBnCP08
Read/Write	R							
After reset	Undefined							
Function	Timer capture value: 15-8 bit data							
	7	6	5	4	3	2	1	0
bit Symbol	TBnCP07	TBnCP06	TBnCP05	TBnCP04	TBnCP03	TBnCP02	TBnCP01	TBnCP00
Read/Write	R							
After reset	Undefined							
Function	Timer capture value: 7-0 bit data							

TBnCP1 capture register (n=0~9)

TBnCP1
(0x4001_0xxC)

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
bit Symbol	TBnCP115	TBnCP114	TBnCP113	TBnCP112	TBnCP111	TBnCP110	TBnCP19	TBnCP18
Read/Write	R							
After reset	Undefined							
Function	Timer capture value: 15-8 bit data							
	7	6	5	4	3	2	1	0
bit Symbol	TBnCP17	TBnCP16	TBnCP15	TBnCP14	TBnCP13	TBnCP12	TBnCP11	TBnCP10
Read/Write	R							
After reset	Undefined							
Function	Timer capture value: 7-0 bit data							

10.5 Description of Operations for Each Circuit

The channels operate in the same way, except for the differences in their specifications as shown in Table 10-1 and Table 10-2. Therefore, the operational descriptions here are only for channel 0.

10.5.1 Prescaler

There is a 4-bit prescaler to generate the source clock for up-counter UC0. The prescaler input clock ϕT0 is fperiph/1, fperiph/2, fperiph/4, fperiph/8, fperiph/16 or fperiph/32 selected by SYSCR1<FPSEL> in the CG. The peripheral clock, fperiph, is either fgear, a clock selected by SYSCR1<FPSEL> in the CG, or fc, which is a clock before it is divided by the clock gear.

The operation or the stoppage of a prescaler is set with TB0RUN<TB0PRUN> where writing “1” starts counting and writing “0” clears and stops counting. Table 10-4 and Table 10-5 show prescaler output clock resolutions.

Table 10-4 Prescaler Output Clock Resolutions @fc = 40MHz

Release peripheral clock <FPSEL>	Clock gear value <GEAR2:0>	Select prescaler clock <PRCK1:0>	Prescaler output clock resolutions		
			ϕT1	ϕT4	ϕT16
0 (fgear)	000 (fc)	000(fperiph/1)	fc/2^1(0.05μs)	fc/2^3(0.2μs)	fc/2^5(0.8μs)
		001(fperiph/2)	fc/2^2(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)
		010(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		100(fperiph/16)	fc/2^5(0.8(s)	fc/27(3.2(s)	fc/29(12.8(s)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
	100(fc/2)	000(fperiph/1)	fc/2^1(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)
		001(fperiph/2)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		010(fperiph/4)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		011(fperiph/8)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		100(fperiph/16)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		101(fperiph/32)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
	101(fc/4)	000(fperiph/1)	fc/2^1(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		001(fperiph/2)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		010(fperiph/4)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		011(fperiph/8)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		100(fperiph/16)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		101(fperiph/32)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
	110(fc/8)	000(fperiph/1)	fc/2^1(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		001(fperiph/2)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		010(fperiph/4)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		011(fperiph/8)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		100(fperiph/16)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
		101(fperiph/32)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)	fc/2^{13}(204.8μs)
1 (fc)	000 (fc)	000(fperiph/1)	fc/2^1(0.05μs)	fc/2^3(0.2μs)	fc/2^5(0.8μs)
		001(fperiph/2)	fc/2^2(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)
		010(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		100(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
	100(fc/2)	000(fperiph/1)	—	fc/2^3(0.2μs)	fc/2^5(0.8μs)
		001(fperiph/2)	fc/2^2(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)
		010(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		100(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
	101(fc/4)	000(fperiph/1)	—	fc/2^3(0.2μs)	fc/2^5(0.8μs)
		001(fperiph/2)	—	fc/2^4(0.4μs)	fc/2^6(1.6μs)
		010(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		100(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
	110(fc/8)	000(fperiph/1)	—	—	fc/2^5(0.8μs)
		001(fperiph/2)	—	fc/2^4(0.4μs)	fc/2^6(1.6μs)
		010(fperiph/4)	—	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		100(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)

(Note 1)	**The prescaler output clock ϕTn must be selected so that ϕTn<fsys is satisfied (so that ϕTn is slower than fsys).**
(Note 2)	**Do not change the clock gear while the timer is operating.**
(Note 3)	**"—" denotes a setting prohibited.**

Table 10-5 Prescaler Output Clock Resolutions @ = 32MHz

Release peripheral clock <FPSEL	**Clock gear value <GEAR2:0>**	**Select prescaler clock <PRCK2:0>**	**Prescaler output clock resolutions**		
			ϕT1	ϕT4	ϕT16
0 (fgear)	000 (fc)	000(fperiph/1)	fc/2^1(0.0625μs)	fc/2^3(0.25μs)	fc/2^5(1.0μs)
		001(fperiph/2)	fc/2^2(0.125μs)	fc/2^4(0.5μs)	fc/2^6(2.0μs)
		010(fperiph/4)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
	100(fc/2)	000(fperiph/1)	fc/2^1(0.125μs)	fc/2^4(0.5μs)	fc/2^6(2.0μs)
		001(fperiph/2)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		010(fperiph/4)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		011(fperiph/8)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		100(fperiph/16)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		101(fperiph/32)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
	101(fc/4)	000(fperiph/1)	fc/2^1(0. 25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		001(fperiph/2)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		010(fperiph/4)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		011(fperiph/8)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		100(fperiph/16)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		101(fperiph/32)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)
	110(fc/8)	000(fperiph/1)	fc/2^1(0. 5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		001(fperiph/2)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		010(fperiph/4)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		011(fperiph/8)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		100(fperiph/16)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)
		101(fperiph/32)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)	fc/2^{13}(256.0μs)
1 (fc)	000 (fc)	000(fperiph/1)	fc/2^1(0.0625μs)	fc/2^3(0.25μs)	fc/2^5(1.0μs)
		001(fperiph/2)	fc/2^2(0.125μs)	fc/2^4(0.5μs)	fc/2^6(2.0μs)
		010(fperiph/4)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
	100(fc/2)	000(fperiph/1)	—	fc/2^3(0.25μs)	fc/2^5(1.0μs)
		001(fperiph/2)	fc/2^2(0.125μs)	fc/2^4(0.5μs)	fc/2^6(2.0μs)
		010(fperiph/4)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
	101(fc/4)	000(fperiph/1)	—	fc/2^3(0.25μs)	fc/2^5(1.0μs)
		001(fperiph/2)	—	fc/2^4(0.5μs)	fc/2^6(2.0μs)
		010(fperiph/4)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
	110(fc/8)	000(fperiph/1)	—	—	fc/2^5(1.0μs)
		001(fperiph/2)	—	fc/2^4(0.5μs)	fc/2^6(2.0μs)
		010(fperiph/4)	—	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)

(Note 1)	**The prescaler output clock ϕTn must be selected so that ϕTn<fsys is satisfied (so that ϕTn is slower than fsys).**
(Note 2)	**Do not change the clock gear while the timer is operating.**
(Note 3)	**"—" denotes a setting prohibited.**

10.5.2 Up-counter (UC0)

UC0 is a 16-bit binary counter.

- Source clock

UC0 source clock, specified by TB0MOD<TB0CLK1:0>, can be selected from either three types - ϕT1, ϕT4 and ϕT16 - of prescaler output clock or the external clock of the TB0IN0 pin.

- Count start/ stop

Counter operation is specified by TB0RUN<TB0RUN>. UC0 starts counting if <TB0RUN> = “1”, and stops counting and clears counter value if <TB0RUN> = “0”.

- Timing to clear UC0

1) When a match is detected

By setting TB0MOD<TB0CLE> = “1”, UC0 is cleared if when the comparator detects a match between counter value and the value set in TB0RG1. UC0 operates as a free-running counter if TB0MOD<TB0CLE> = “0”.

2) When UC0 stops

UC0 stops counting and clears counter value if TB0RUN <TB0RUN> = “0”.

- UC0 overflow

If UC0 overflow occurs, the INTTB0 overflow interrupt is generated.

10.5.3 Timer registers (TB0RG0, TB0RG1)

TB0RG0 and TB0RG1 are registers for setting values to compare with up-counter values and two registers are built into each channel. If the comparator detects a match between a value set in this timer register and that in a UC0 up-counter, it outputs the match detection signal.

- Configuration

TB0RG0 and TB0RG1 of this timer registers are paired with register buffer 0 - a double-buffered configuration. The two registers use TB0CR<TB0WBF> to control the enabling/disabling of double buffering. If <TB0WBF> = “0”, double buffering is disabled and if <TB0WBF> = “1”, it is enabled. If double buffering is enabled, data is transferred from register buffer 0 to the TB0RG0 and TB0RG1 timer registers when there is a match between UC0 and TB0RG1.

- Default setting

The values of TB0RG0 and TB0RG1 become undefined after a reset. A reset disables the double buffer.

- Register setting

1) When not using double-buffering

To write data to the timer registers, either a 2-byte data transfer instruction or a 1-byte data transfer instruction written twice in the order of low-order 8 bits followed by high-order 8 bits can be used.

2) When using double-buffering

TB0RG0/ TB0RG1 and the register buffers are assigned to the same address. If <TB0WBF> = “0,” the same value is written to TB0RG0, TB0RG1 and each register buffer; if <TB0WBF> = “1,” the value is only written to each register buffer. To write an initial value to the timer register, therefore, the register buffers must be set to “disable”. Then set <TB0WBF> = “1”and write the following data to the register.

10.5.4 Capture

This is a circuit that controls the timing of latching values from the UC0 up-counter into the TB0CP0 and TB0CP1 capture registers. The timing with which to latch data is specified by TB0MOD <TB0CPM1:0>.

Software can also be used to import values from the UC0 up-counter into the capture register; specifically, UC0 values are taken into the TB0CP0 capture register each time “0” is written to TB0MOD<TB0CP0>. To use this capability, the prescaler must be running (TB0RUN<TB0PRUN> = “1”).

10.5.5 Capture Registers (TB0CP0H/L, TB0CP1H/L)

These are 16-bit registers for latching values from the UC0 up-counter. To read data from the capture register, use a 16-bit data transfer instruction or read in the order of low-order bits followed by high-order bits.

10.5.6 Up-counter capture register (TB0UCH/L)

Other than the capturing functions shown above, the current count value of the UC0 can be captured by reading the TB0UC registers.

10.5.7 Comparators (CP0, CP1)

These are 16-bit comparators for detecting a match by comparing set values of the UC0 up-counter with set values of the TB0RG0 and TB0RG1 timer registers. If a match is detected, INTTB0 is generated.

10.5.8 Timer Flip-flop (TB0FF0)

The timer flip-flop (TB0FF0) 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 TB0FFCR<TB0C1T1, TB0C0T1, TB0E1T1, TB0E0T1>.

The value of TB0FF0 becomes undefined after a reset. The flip-flop can be reversed by writing “00” to TB0FFCR<TB0FF0C1:0>. It can be set to “1” by writing “01,” and can be cleared to “0” by writing “10.”

The value of TB0FF0 can be output to the timer output pin, TB0OUT (shared with PI0). To enable timer output, the port I related registers PICR and PIFR1 must be programmed beforehand.

10.5.9 Capture interrupt (INTCAP00, INTCAP01)

Interrupts INTCAP00 and INTCAP01 can be generated at the timing of latching values from the UC0 up-counter into the TB0CP0 and TB0CP1 capture registers. The interrupt timing is specified by the CPU.

10.6 Description of Operations for Each Mode

10.6.1 16-bit Interval Timer Mode

-Generating interrupts at periodic cycles

To generate the INTTB0 interrupt, specify a time interval in the TB0RG1 timer register.

10.6.2 16-bit Event Counter Mode

It is possible to make it the event counter by using an input clock as an external clock (TB0IN0 pin input).

The up-counter counts up on the rising edge of TB0IN0 pin input. It is possible to read the count value by capturing value using software and reading the captured value.

To use it as an event counter, put the prescaler in a “RUN” state (TB0RUN<TB0PRUN> = “1”).

10.6.3 16-bit Programmable Square Wave Output Mode (PPG)

Square waves with any frequency and any duty (programmable square waves) can be output. The output pulse can be either low-active or high-active.

Programmable square waves can be output from the TB0OUT pin by triggering the timer flip-flop (TB0FF) to reverse when the set value of the up-counter (UCO) matches the set values of the timer registers (TB0RG0 and TB0RG1). Note that the set values of TB0RG0 and TB0RG1 must satisfy the following requirement:

(Set value of TB0RG0) < (Set value of TB0RG1)

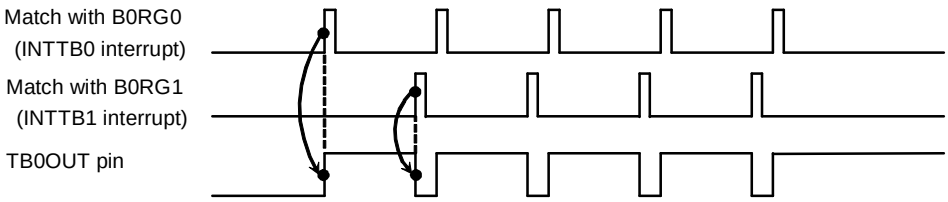


Fig. 10-2 Example of Output of Programmable Square Wave (PPG)

In this mode, by enabling the double buffering of TB0RG0, the value of register buffer 0 is shifted into TB0RG0 when the set value of the up-counter matches the set value of TB0RG1. This facilitates handling of small duties.

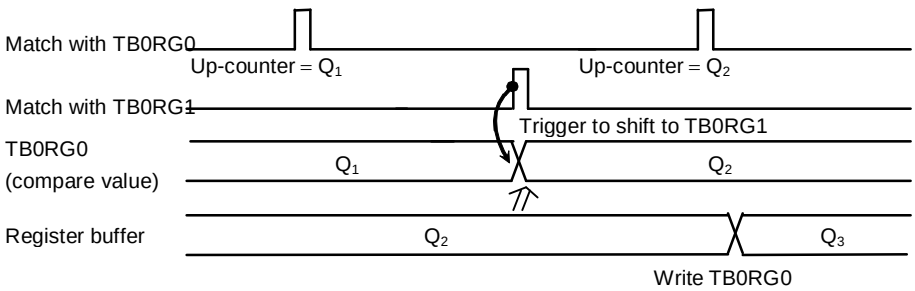


Fig. 10-3 Register Buffer Operation

The block diagram of this mode is shown below.

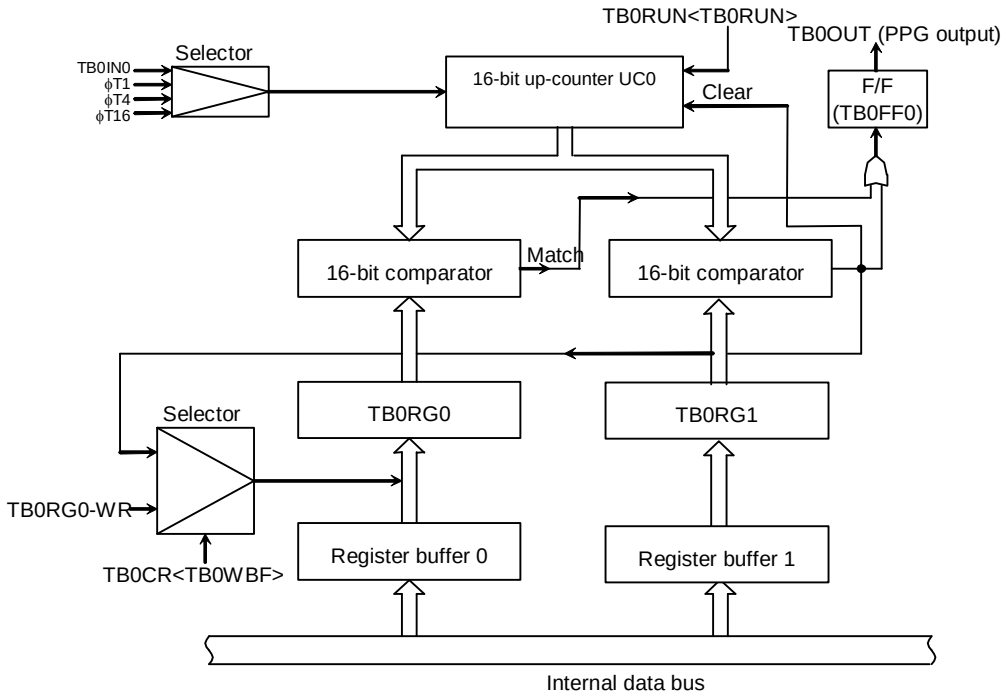


Fig. 10-4 Block Diagram of 16-bit PPG Mode

Each register in the 16-bit PPG output mode must be programmed as listed below.

		7	6	5	4	3	2	1	0	
TB0EN	←	1	X	X	X	X	X	X	X	Starts the TMRB0 module.
TB0RUN	←	X	X	X	X	X	0	X	0	Stops the TMRB0 module.
TB0RG0	←	*	*	*	*	*	*	*	*	Specifies a duty. (16 bits *32-bits for register)
	←	*	*	*	*	*	*	*	*	
TB0RG1	←	*	*	*	*	*	*	*	*	Specifies a cycle. (16 bits *32-bits for register)
	←	*	*	*	*	*	*	*	*	
TB0CR	←	1	0	X	0	0	0	0	0	Enables the TB0RG0 double buffering. (Changes the duty/cycle when the INTTB0 interrupt is generated)
TB0FFCR	←	X	X	0	0	1	1	1	0	Specifies to trigger TB0FF0 to reverse when a match with TB0RG0 or TB0RG1 is detected, and sets the initial value of TB0FF0 to "0."
TB0MOD	←	0	0	1	0	0	1	*	*	Designates the prescaler output clock as the input clock, and disables the capture function. (** = 01, 10, 11)
PICR	←	–	–	–	–	–	–	–	1	Assigns PI0 to TB0OUT
PIFR1	←	–	–	–	–	–	–	–	1	
TB0RUN	←	*	*	*	*	*	1	X	1	Starts TMRB0

X; Don't care –; no change

10.7 Timer synchronous mode

This mode enables the timers to start synchronously.

If the mode is used with PPG output, the output can be applied to drive a motor.

Use of the timer synchronous mode is specified in TBnCR<TBnSYNC>.

<TBnSYNC> =“0”: Timers operates individually.

<TBnSYNC> =“1”: Timers operate synchronously.

The channels are in two segments: channels TMRB0 through 3 and channels TMRB4 through 7.

If <TBnSYNC> =“1” is set, the start timing is synchronized with TMRB0 and TMRB4. The start timing of each channel, TBmRUN <TBmPRUN,TBmRUN> =”1,1”, is ignored.

(Note 1) The channels designated for synchronous output must be started by TBmRUN<TBmPRUN,TBmRUN>=”1,1” before the start triggered by TMRB0 and TMRB4.

(Note 2) Set TBnCR<TBnSYNC> to “0” unless the timer synchronous mode is used. The timer synchronous mode keeps the other channels operation waiting until TMRB0, TMRB4 and TMRB8 start operation.

(Note 3) TMRB0 and TMRB4 are the master clocks of the timer synchronous mode. Therefore, their TBnSYNC bit must be set to “0”.

(Note 4) This mode cannot be applied to TMRB8 and TMRB9.

TBnCR (0x4001_0xx8)

	7	6	5	4	3	2	1	0
bit Symbol	TBnWBF		TBnSYNC		I2TBn			
Read/Write	R/W	R/W	R/W	R	R/W	R	R	R
After reset	0	0	0	0	0	0	0	0
Function	Double Buffer 0:Disabled 1:Enabled	Write “0”.	Timer operation **0:Individual**	“0” is read.	IDLE 0:Stop 1:Operation	“0” is read.	“0” is read.	“0” is read.

Set the TBnSYNC bit of the timers, which are used as the slave clocks, to “1”.

TBnCR (0x4001_0xx8)

	7	6	5	4	3	2	1	0
bit Symbol	TBnWBF		TBnSYNC		I2TBn			
Read/Write	R/W	R/W	R/W	R	R/W	R	R	R
After reset	0	0	0	0	0	0	0	0
Function	Double Buffer 0:Disabled 1:Enabled	Write “0”.	Timer operation **1:Synchronous**	“0” is read.	IDLE 0:Stop 1:Operation	“0” is read.	“0” is read.	“0” is read.

10.8 Applications using the Capture Function

The capture function can be used to develop many applications, including those described below:

① One-shot pulse output triggered by an external pulse

② Frequency measurement

③ Pulse width measurement

④ Time difference measurement

① One-shot pulse output triggered by an external pulse

One-shot pulse output triggered by an external pulse is carried out as follows:

The 16-bit up-counter is made to count up by putting it in a free-running state using the prescaler output clock. An external pulse is input through the TB5IN0 pin. A trigger is generated at the rising of the external pulse by using the capture function and the value of the up-counter is taken into the capture registers (TB5CP0).

The CPU must be programmed so that an interrupt INTCAP50 is generated at the rising of an external trigger pulse. This interrupt is used to set the timer registers (TB5RG0) to the sum of the TB5CP0 value (c) and the delay time (d), (c + d), and set the timer registers (TB5RG1) to the sum of the TB5RG0 values and the pulse width (p) of one-shot pulse, (c + d + p).

TB5RG1 change must be completed before the next match.

In addition, the timer flip-flop control registers (TB5FFCR<TB5E1T1, TB5E0T1>) must be set to “11.” This enables triggering the timer flip-flop (TB5FF0) to reverse when UC5 matches TB5RG0 and TB5RG1. This trigger is disabled by the INTTB5 interrupt after a one-shot pulse is output.

Symbols (c), (d) and (p) used in the text correspond to symbols c, d and p in “Fig. 10-5 One-shot Pulse Output (With Delay).”

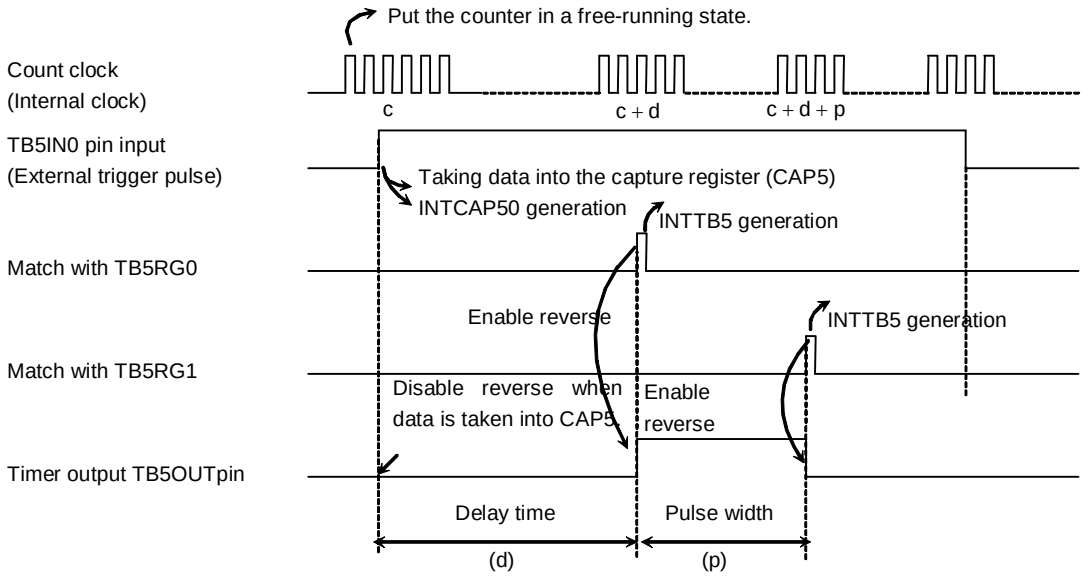


Fig. 10-5 One-shot Pulse Output (With Delay)

If a delay is not required, TB5FF0 is reversed when data is taken into TB5CP0, and TB5RG1 is set to the sum of the TB5CPO value (c) and the one-shot pulse width (p), (c + p), by generating the INT0 interrupt. TB5RG1 change must be completed before the next match. TB5FF0 is enabled to reverse when UC5 matches with TB5RG1, and is disabled by generating the INTTB5 interrupt.

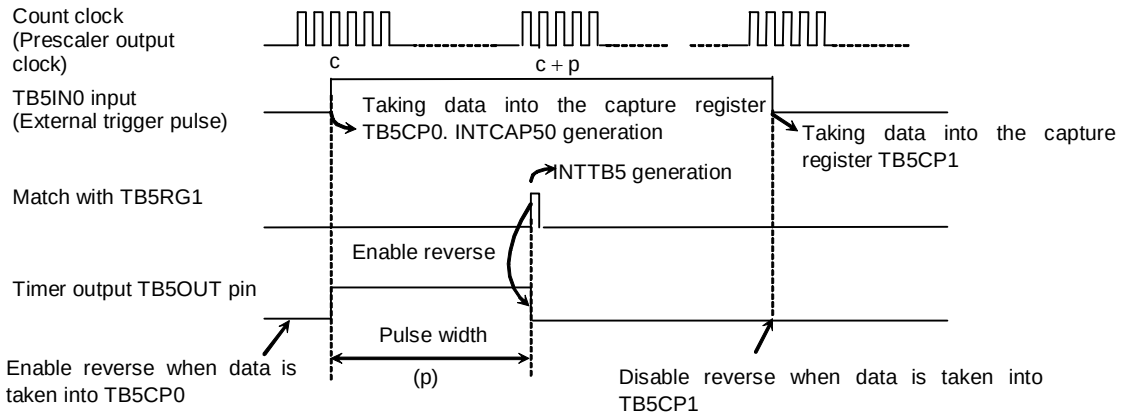


Fig. 10-6 One-shot Pulse Output Triggered by an External Pulse (Without Delay)

② Frequency measurement

The frequency of an external clock can be measured by using the capture function.

To measure frequency, another 16-bit timer (TMRB0) is used in combination with the 16-bit event counter mode. As an example, we explain with TMRB3 and TMRB8. TB8OUT of the 16-bit timer TMRB8 is used to specify the measurement time.

The TB3IN0 pin input is selected as the TMRB3 count clock to perform the count operation using an external input clock. TB3MOD<TB3CPM1:0> is set to “11.” This setting allows a count value of the 16-bit up-counter UC3 to be taken into the capture register (TB3CP0) upon rising of a timer flip-flop output (TB8OUT) of the 16-bit timer (TMRB8), and an UC3 counter value to be taken into the capture register (TB3CP1) upon falling of TB8OUT of the 16-bit timer (TMRB8).

A frequency is then obtained from the difference between TB3CP0 and TB3CP1 based on the measurement, by generating the INTTB8 16-bit timer interrupt.

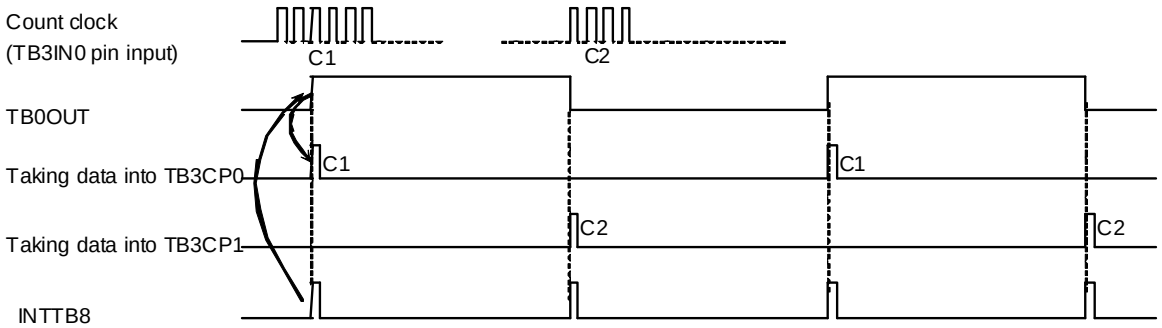


Fig. 10-7 Frequency Measurement

For example, if the set width of TB0FF level “1” of the 16-bit timer is 0.5 s and if the difference between TB3CP0 and TB3CP1 is 100, the frequency is 100 / 0.5 s = 200 Hz.

③ Pulse width measurement

By using the capture function, the “H” level width of an external pulse can be measured. Specifically, by putting it in a free-running state using the prescaler output clock, an external pulse is input through the TB5IN0 pin and the up-counter (UC5) is made to count up. A trigger is generated at each rising and falling edge of the external pulse by using the capture function and the value of the up-counter is taken into the capture registers (TB5CP0, TB5CP1). The CPU must be programmed so that INTCAP51 is generated at the falling edge of an external pulse input through the TB5IN0 pin.

The “H” level pulse width can be calculated by multiplying the difference between TB5CP0 and TB5CP1 by the clock cycle of an internal clock.

For example, if the difference between TB5CP0 and TB5CP1 is 100 and the cycle of the prescaler output clock is 0.5 *u*s, the pulse width is 100 × 0.5 *u*s = 50 *u*s.

Caution must be exercised when measuring pulse widths exceeding the UC5 maximum count time which is dependant upon the source clock used. The measurement of such pulse widths must be made using software.

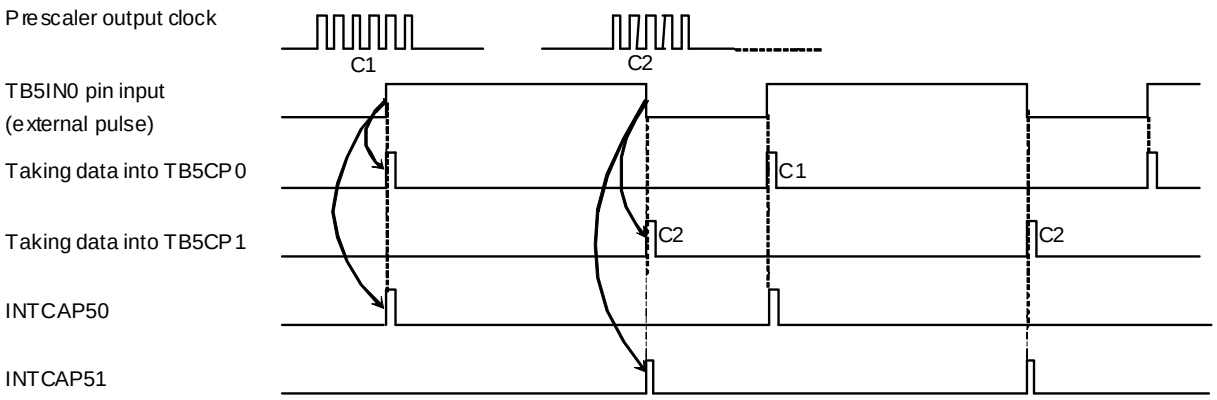


Fig. 10-8 Pulse Width Measurement

The “L” level width of an external pulse can also be measured. In such cases, the difference between C2 generated the first time and C1 generated the second time is initially obtained by performing the second stage of INTCAP50 interrupt processing as shown in “Fig. 10-8 Pulse Width Measurement” and this difference is multiplied by the cycle of the prescaler output clock to obtain the “L” level width.

④ Time Difference Measurement

The up-counter (UC5) is made to count up by putting it in a free-running state using the prescaler output clock. The value of UC5 is taken into the capture register (TB5CP0) at the rising edge of the TB5IN0 pin input pulse. The CPU must be programmed to generate INTCAP50 interrupt at this time.

The value of UC5 is taken into the capture register TB5CP1 at the rising edge of the TB5IN1 pin input pulse. The CPU must be programmed to generate INTCAP51 interrupt at this time.

The time difference can be calculated by multiplying the difference between TB5CP1 and TB5CP0 by the clock cycle of an internal clock.

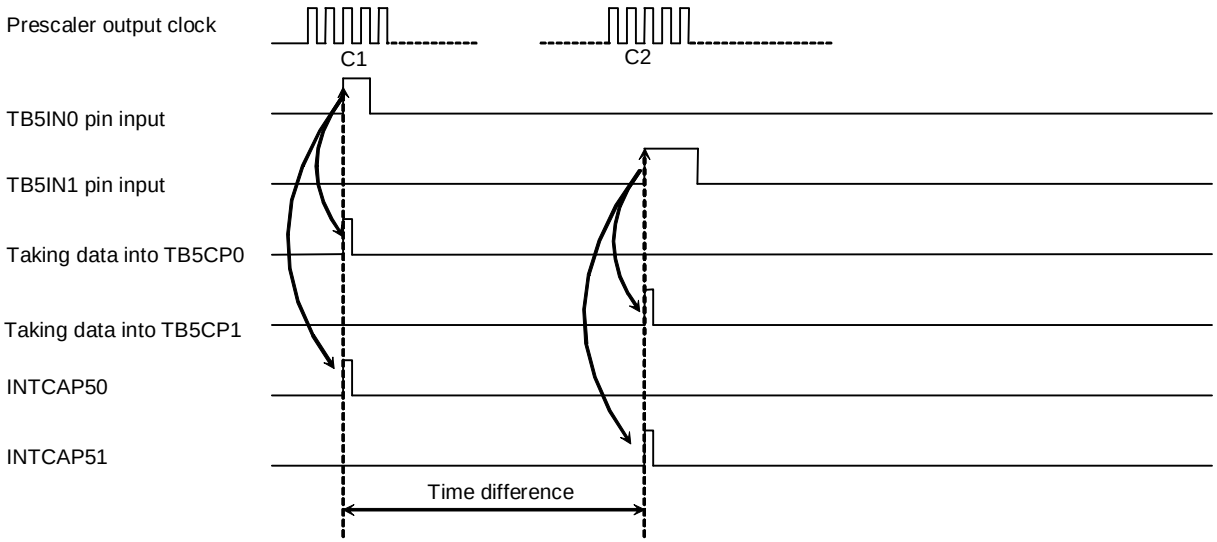


Fig. 10-9 Time Difference Measurement

11 Serial Channel (SIO)

11.1 Features

This device has three serial I/O channels: SIO0 to SIO2. Each channel operates in either the UART mode (asynchronous communication) or the I/O interface mode (synchronous communication) which is selected by the user.

I/O interface mode — Mode 0: This is the mode to transmit and receive I/O data and associated synchronization signals (SCLK) to extend I/O.

Asynchronous (UART) mode:
- Mode 1: TX/RX Data Length: 7 bits
- Mode 2: TX/RX Data Length: 8 bits
- Mode 3: TX/RX Data Length: 9 bits

In the above modes 1 and 2, parity bits can be added. The mode 3 has a wakeup function in which the master controller can start up slave controllers via the serial link (multi-controller system). Fig. 11-2 shows the block diagram of SIO0.

Each channel consists of a prescaler, a serial clock generation circuit, a receive buffer, its control circuit, a transmit buffer and its control circuit. Each channel functions independently.

As the SIOs 0 to 2 operate in the same way, only SIO0 is described here.

Table 11-1 Difference in the Specifications of SIO Modules

		Channel 0		Channel 1		Channel 2	
Pin name		TXD0 (PE0) RXD0 (PE1) $\overline{CTS0}$ /SCLK0 (PE2)		TXD1 (PE4) RXD1 (PE5) $\overline{CTS1}$ /SCLK1 (PE6)		TXD1 (PF0) RXD1 (PF1) $\overline{CTS1}$ /SCLK1 (PF2)	
Interrupt		INTRX0 INTTX0		INTRX1 INTTX1		INTRX2 INTTX2	
Register name (address)	Enable register	SC0EN	0x4002_0080	SC1EN	0x4002_00C0	SC2EN	0x4002_0100
	Transmit/ receive buffer register	SC0BUF	0x4002_0084	SC1BUF	0x4002_00C4	SC2BUF	0x4002_0104
	Control register	SC0CR	0x4002_0088	SC1CR	0x4002_00C8	SC2CR	0x4002_0108
	Mode control register 0	SC0MOD0	0x4002_008C	SC1MOD0	0x4002_00CC	SC2MOD0	0x4002_010C
	Baud rate generator control	SC0BRCR	0x4002_0090	SC1BRCR	0x4002_00D0	SC2BRCR	0x4002_0110
	Baud rate generator control 2	SC0BRADD	0x4002_0094	SC1BRADD	0x4002_00D4	SC2BRADD	0x4002_0114
	Mode control register 1	SC0MOD1	0x4002_0098	SC1MOD1	0x4002_00D8	SC2MOD1	0x4002_0118
	Mode control register 2	SC0MOD2	0x4002_009C	SC1MOD2	0x4002_00DC	SC2MOD2	0x4002_011C
	Receive FIFO configuration register	SC0RFC	0x4002_00A0	SC1RFC	0x4002_00E0	SC2RFC	0x4002_0120
	Transmit FIFO configuration register	SC0TFC	0x4002_00A4	SC1TFC	0x4002_00E4	SC2TFC	0x4002_0124
	Receive FIFO status register	SC0RST	0x4002_00A8	SC1RST	0x4002_00E8	SC2RST	0x4002_0128
	Transmit FIFO status register	SC0TST	0x4002_00AC	SC1TST	0x4002_00EC	SC2TST	0x4002_012C
	FIFO configuration register	SC0FCNF	0x4002_00B0	SC1FCNF	0x4002_00F0	SC2FCNF	0x4002_0130

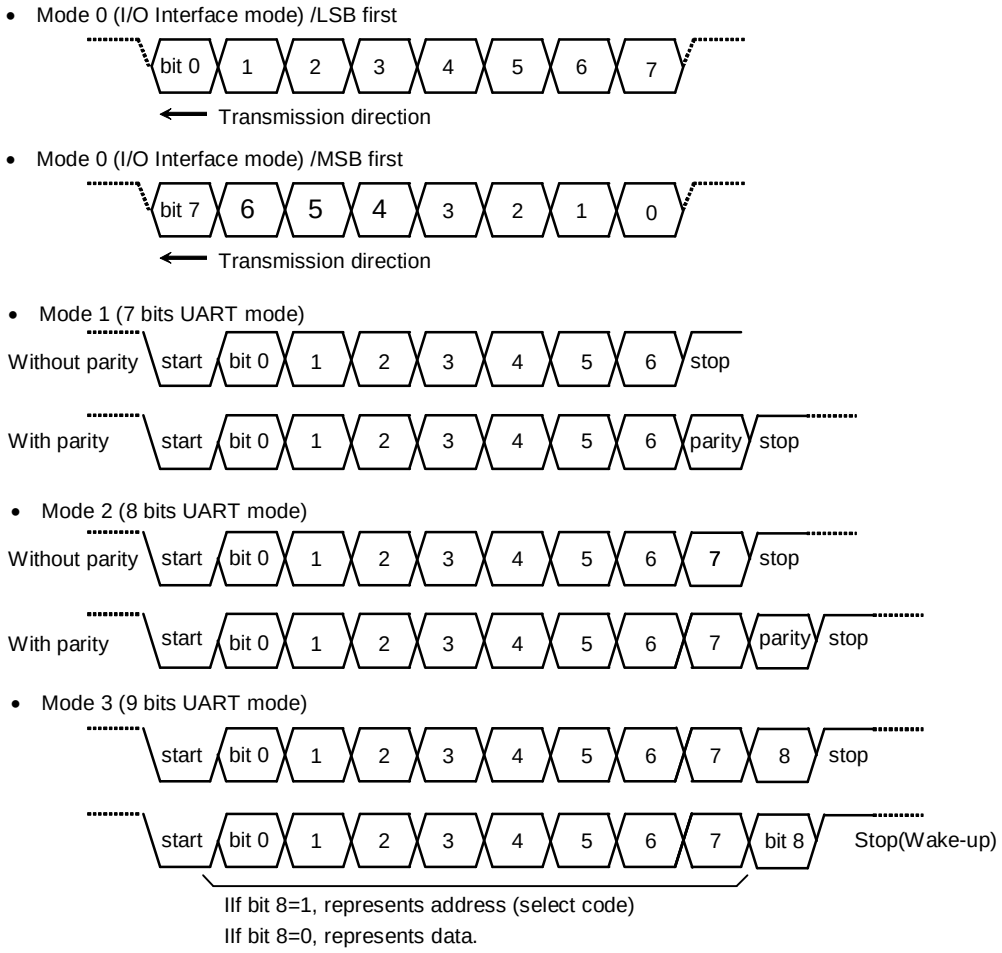


Fig. 11-1 Data Format

11.2 Block Diagram (Channel 0)

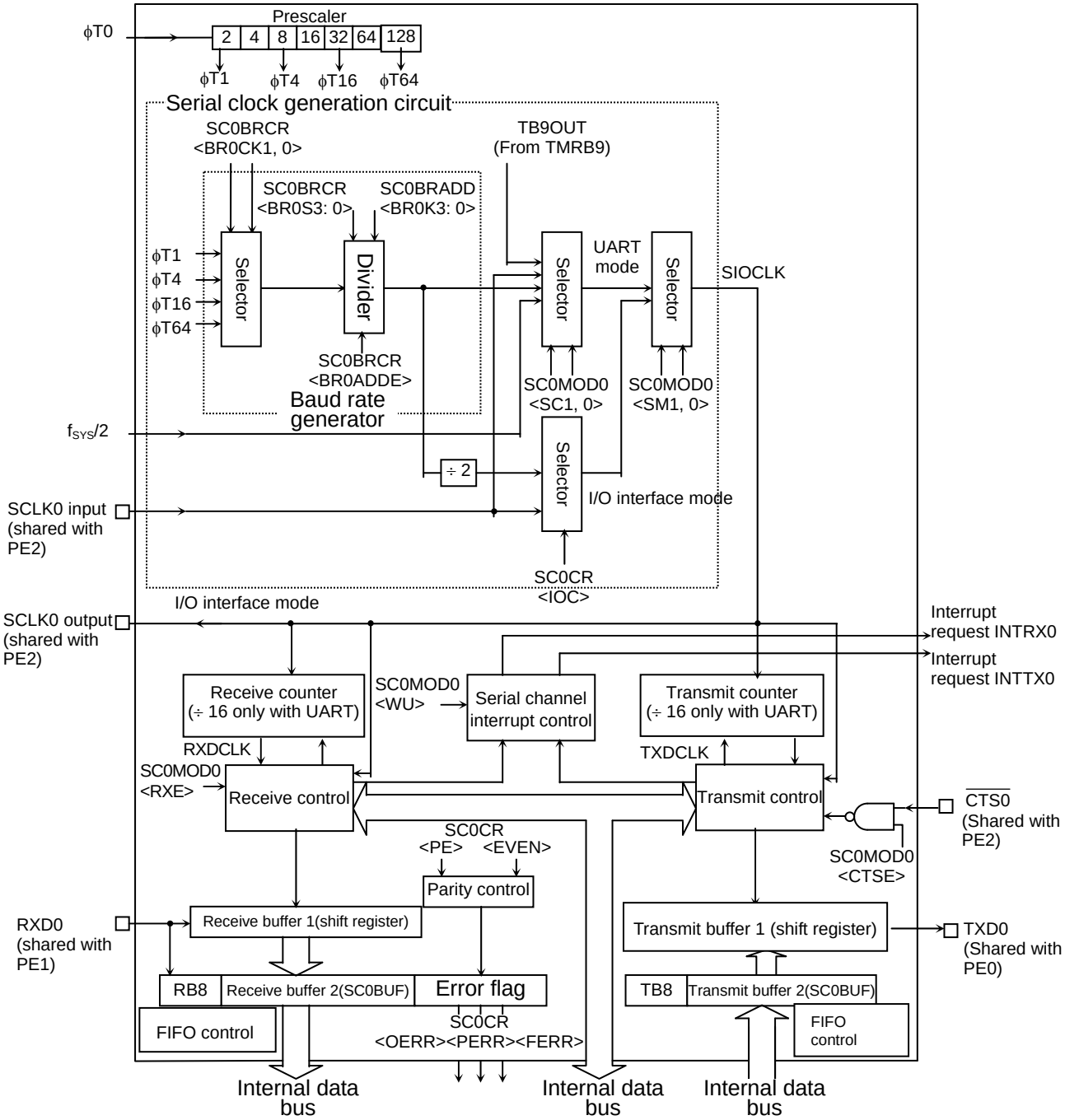


Fig. 11-2 SIO0 Block Diagram

11.3 Operation of Each Circuit (Channel 0)

11.3.1 Prescaler

The device includes a 7-bit prescaler to generate necessary clocks to drive SIO0. The input clock ϕT0 to the prescaler is selected by SYSCR1 of CG <PRCK2:0> to provide the frequency of either fperiph/1, fperiph/2, fperiph/4, fperiph/8, fperiph/16 or fperiph/32.

The clock frequency fperiph is either the clock “fgear,” to be selected by SYSCR1<FPSEL> of CG, or the clock “fc” before it is divided by the clock gear.

The prescaler becomes active only when the baud rate generator is selected for generating the serial transfer clock. Table 11-2 and Table 11-3 list the prescaler output clock resolution.

Table 11-2 Clock Resolution to the Baud Rate Generator @ = 40MHz

Clear peripheral clock <FPSEL>	Clock gear value <GEAR2:0>	Prescaler clock selection <PRCK2:0>	Prescaler output clock resolution			
			ϕT1	ϕT4	ϕT16	ϕT64
0 (fgear)	000 (fc)	000(fperiph/1)	fc/2^1(0.05μs)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		001(fperiph/2)	fc/2^2(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		010(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		100(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
	100(fc/2)	000(fperiph/1)	fc/2^2(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		001(fperiph/2)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		010(fperiph/4)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		011(fperiph/8)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		100(fperiph/16)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
		101(fperiph/32)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)	fc/2^{13}(204.8μs)
	101(fc/4)	000(fperiph/1)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		001(fperiph/2)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		010(fperiph/4)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		011(fperiph/8)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
		100(fperiph/16)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)	fc/2^{13}(204.8μs)
		101(fperiph/32)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)	fc/2^{14}(409.6μs)
	110(fc/8)	000(fperiph/1)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		001(fperiph/2)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		010(fperiph/4)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
		011(fperiph/8)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)	fc/2^{13}(204.8μs)
		100(fperiph/16)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)	fc/2^{14}(409.6μs)
		101(fperiph/32)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)	fc/2^{13}(204.8μs)	fc/2^{15}(819.2μs)
1 (fc)	000 (fc)	000(fperiph/1)	fc/2^1(0.05μs)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		001(fperiph/2)	fc/2^2(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		010(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		100(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
	100(fc/2)	000(fperiph/1)	-	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		001(fperiph/2)	fc/2^2(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		010(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		100(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
	101(fc/4)	000(fperiph/1)	-	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		001(fperiph/2)	-	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		010(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		100(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
	110(fc/8)	000(fperiph/1)	-	-	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		001(fperiph/2)	-	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		010(fperiph/4)	-	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		011(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		100(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		101(fperiph/32)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)

(Note 1)	**The prescaler output clock ϕTn must be selected so that the relationship "ϕTn < fsys" is satisfied (so that ϕTn is slower than fsys).**
(Note 2)	**Do not change the clock gear while SIO is operating.**
(Note 3)	**The horizontal lines in the above table indicate that the setting is prohibited.**

The serial interface baud rate generator uses four different clocks, i.e., ϕT1, ϕT4, ϕT16 and ϕT64, supplied from the prescaler output clock.

Table 11-3 Clock Resolution to the Baud Rate Generator @ = 32MHz

Clear periferal clock <FPSEL>	Clock gear value <GEAR2:0>	Prescaler clock selection <PRCK2:0>	Prescaler output clock resolution			
			ϕT1	ϕT4	ϕT16	ϕT64
0 (fgear)	000 (fc)	000(fperiph/1)	fc/2^1(0.0625μs)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		001(fperiph/2)	fc/2^2(0.125μs)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		010(fperiph/4)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)
	100(fc/2)	000(fperiph/1)	fc/2^1(0.0625μs)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		001(fperiph/2)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		010(fperiph/4)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		011(fperiph/8)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		100(fperiph/16)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)
		101(fperiph/32)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)	fc/2^{13}(256.0μs)
	101(fc/4)	000(fperiph/1)	fc/2^1(0.0625μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		001(fperiph/2)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		010(fperiph/4)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		011(fperiph/8)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)
		100(fperiph/16)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)	fc/2^{13}(256.0μs)
		101(fperiph/32)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)	fc/2^{14}(512.0μs)
	110(fc/8)	000(fperiph/1)	fc/2^1(0.0625μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		001(fperiph/2)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		010(fperiph/4)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)
		011(fperiph/8)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)	fc/2^{13}(256.0μs)
		100(fperiph/16)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)	fc/2^{14}(512.0μs)
		101(fperiph/32)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)	fc/2^{13}(256.0μs)	fc/2^{15}(1024μs)
1 (fc)	000 (fc)	000(fperiph/1)	fc/2^1(0.0625μs)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		001(fperiph/2)	fc/2^2(0.125μs)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		010(fperiph/4)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)
	100(fc/2)	000(fperiph/1)	-	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		001(fperiph/2)	fc/2^2(0.125μs)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		010(fperiph/4)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)
	101(fc/4)	000(fperiph/1)	-	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		001(fperiph/2)	-	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		010(fperiph/4)	fc/2^3(0.25μs)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)
	110(fc/8)	000(fperiph/1)	-	-	fc/2^5(1.0μs)	fc/2^7(4.0μs)
		001(fperiph/2)	-	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)
		010(fperiph/4)	-	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)
		011(fperiph/8)	fc/2^4(0.5μs)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)
		100(fperiph/16)	fc/2^5(1.0μs)	fc/2^7(4.0μs)	fc/2^9(16.0μs)	fc/2^{11}(64.0μs)
		101(fperiph/32)	fc/2^6(2.0μs)	fc/2^8(8.0μs)	fc/2^{10}(32.0μs)	fc/2^{12}(128.0μs)

(Note 1)	**The prescaler output clock ϕTn must be selected so that the relationship "ϕTn < fsys" is satisfied (so that ϕTn is slower than fsys).**
(Note 2)	**Do not change the clock gear while SIO is operating.**
(Note 3)	**The horizontal lines in the above table indicate that the setting is prohibited.**

The serial interface baud rate generator uses four different clocks, i.e., ϕT1, ϕT4, ϕT16 and ϕT64, supplied from the prescaler output clock.

11.3.2 Baud Rate Generator

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

The baud rate generator uses either the ϕT1, ϕT4, ϕT16 or ϕT64 clock supplied from the 7-bit prescaler. This input clock selection is made by setting the baud rate generator control register, SC0BRCR <BR0CK1:0>.

The baud rate generator contains built-in dividers for divide by 1, N + m/16 (N=2~15, m=0~15), and 16. The division is performed according to the settings of the baud rate generator control registers SC0BRCR <BR0ADDE> <BR0S3:0> and SC0BRADD <BR0K3:0> to determine the resulting transfer rate.

- UART mode

  1) If SC0BRCR <BR0ADDE> = 0,

     The setting of SC0BRADD <BR0K3:0> is ignored and the counter is divided by N where N is the value set to SC0BRCR <BR0S3:0>. (N = 1 to 16).

  2) If SC0BRCR <BR0ADDE> = 1,

     The N + (16 - K)/16 division function is enabled and the division is made by using the values N (set in SC0BRCR <BR0S3:0>) and K (set in SC0BRADD<BR0K3:0>). (N = 2 to 15, K = 1 to 15)

(Note) For the N values of 1 and 16, the above N+(16-K)/16 division function is inhibited. So, be sure to set SC0BRCR<BR0ADDE> to "0."

- I/O interface mode

  The N + (16 - K)/16 division function cannot be used in the I/O interface mode. Be sure to divide by N, by setting SC0BRCR <BR0ADDE> to "0".

- Baud rate calculation to use the baud rate generator:

  1) UART mode

$$\text{Baud rate} = \frac{\text{Baud rated generator input clock}}{\text{Frequency divided by the divide ratio}} /16$$

The highest baud rate out of the baud rate generator is 1.25 Mbps when ϕT1 is 20 MHz.

The fsys frequency, which is independent of the baud rate generator, can be used as the serial clock. In this case, the highest baud rate will be 2.5 Mbps when fsys is 40 MHz.

  2) I/O interface mode

$$\text{Baud rate} = \frac{\text{Baud rated generator input clock}}{\text{Frequency divided by the divide ratio}} /2$$

The highest baud rate will be generated when ϕT1 is 20 MHz. The divide ratio can be set to 1 if double buffer is used and the resulting output baud rate will be 10 Mbps. (If double buffering is not used, the highest baud rate will be 5.0 Mbps applying the divide ratio of "2").

- Example baud rate setting:

1) Division by an integer (divide by N):

Selecting fc = 39.321 MHz for fperiph, setting ϕT0 to fperiph/16, using the baud rate generator input clock ϕT1, setting the divide ratio N (SC0BRCR<BR0S3:0>) = 4, and setting SC0BRCR<BR0ADDE> = "0," the resulting baud rate in the UART mode is calculated as follows:

* Clocking conditions	System clock	:	High-speed (fc)
	High speed clock gear	:	x 1 (fc)
	Prescaler clock	:	f_{periph}/16 (f_{periph} = f_{sys})

Baud rate = $\frac{fc/32}{4}$ /16

$= 39.321 \times 10^6 \div 32 \div 4 \div 16 \fallingdotseq 19200$ (bps)

(Note) The divide by (N + (16-K)/16) function is inhibited and thus SC0BRADD <BR0K3:0> is ignored.

2) For divide by N + (16-K)/16 (only for UART mode):

Selecting fc = 9.6 MHz for fperiph, setting ϕT0 to fperiph/8, using the baud rate generator input clock ϕT1, setting the divide ratio N (SC0BRCR<BR0S3:0>) = 7, setting K (SC0BRADD<BR0K3:0>) = 3, and selecting SC0BRCR<BR0ADDE> = 1, the resulting baud rate is calculated as follows:

* Clocking conditions	System clock	:	High-speed (fc)
	High-speed clock gear	:	x 1 (fc)
	Prescaler clock	:	f_{periph}/4 (f_{periph} = f_{sys})

Baud rate = $\frac{fc/16}{7 + \frac{(16 - 3)}{16}}$ /16

$= 9.6 \times 10^6 \div 16 \div (7 + \frac{13}{16}) \div 16 = 4800$ (bps)

Also, an external clock input may be used as the serial clock. The resulting baud rate calculation is shown below:

- Baud rate calculation for an external clock input:

1) UART mode

Baud Rate = external clock input / 16

In this, the period of the external clock input must be equal to or greater than 2/fsys.

If fsys = 40 MHz, the highest baud rate will be 40÷2÷16=1.25 (Mbps).

2) I/O interface mode

Baud Rate = external clock input

When double buffering is used, it is necessary to satisfy the following relationship:

External clock input period > 6/fsys

Therefore, when fsys = 40 MHz, the baud rate must be set to a rate lower than 40÷6=6.66 (Mbps).

When double buffering is not used, it is necessary to satisfy the following relationship:

External clock input period > 8/fsys

Therefore, when fsys = 40 MHz, the baud rate must be set to a rate lower than 40÷8=5.0 (Mbps).

The baud rate examples for the UART mode are shown in Table 11-4 and Table 11-5.

Table 11-4 Selection of UART Baud Rate

(Using the baud rate generator with SC0BRCR <BR0ADDE> = 0)

Unit: (kbps)

fc [MHz]	Input clock / Divide ratio N (Set to SC0BRCR <BR0S3 : 0>)	ϕT1 (fc/4)	ϕT4 (fc/16)	ϕT16 (fc/64)	ϕT64 (fc/256)
9.830400	2	76.800	19.200	4.800	1.200
↑	4	38.400	9.600	2.400	0.600
↑	8	19.200	4.800	1.200	0.300
↑	0	9.600	2.400	0.600	0.150

(Note) This table shows the case where the system clock is set to fc, the clock gear is set to fc/1, and the prescaler clock is set to f_{periph}/2.

Table 11-5 Selection of UART Baud Rate

(The TMRB9 timer output (internal TB9OUT) is used with the timer input clock set to ϕT0.)

Unit: (Kbbs)

TB0RG \ fc	32 MHz	9.8304 MHz	8 MHz
1H	250	76.8	62.5
2H	125	38.4	31.25
3H		25.6	
4H	62.5	19.2	15.625
5H	50	15.36	12.5
6H		12.8	
8H	31.25	9.6	
AH	25	7.68	6.25
10H	15.625	4.8	
14H	12.5	3.84	3.125

Baud rate calculation to use the TMRB9 timer:

$$\text{Transfer rate} = \frac{\text{Clock frequency selected by SYSCR0<PRCK1:0>}}{\text{TB0RG}\times\underline{2}\times 16}$$

(When input clock to the timer TMRB9 is ϕT0)

(Note 1) In the I/O interface mode, the TMRB9 timer output signal cannot be used internally as the transfer clock.

(Note 2) This table shows the case where the system clock is set to fc, the clock gear is set to fc, and the prescaler clock is set to f_{periph}/4.

11.3.3 Serial Clock Generation Circuit

This circuit generates basic transmit and receive clocks.

- I/O interface mode

  In the SCLK output mode with the SC0CR <IOC> serial control register set to “0,” the output of the previously mentioned baud rate generator is divided by 2 to generate the basic clock.

  In the SCLK input mode with SC0CR <IOC> set to “1,” rising and falling edges are detected according to the SC0CR <SCLKS> setting to generate the basic clock.

- Asynchronous (UART) mode :

  According to the settings of the serial control mode register SC0MOD0 <SC1:0>, either the clock from the baud rate register, the system clock (f_{SYS}), the internal output signal of the TMRB9 timer, or the external clock (SCLKO pin) is selected to generate the basic clock, SIOCLK.

11.3.4 Receive Counter

The receive counter is a 4-bit binary counter used in the asynchronous (UART) mode and is up-counted by SIOCLK. Sixteen SIOCLK clock pulses are used in receiving a single data bit while the data symbol is sampled at the seventh, eighth, and ninth pulses. From these three samples, majority logic is applied to decide the received data.

11.3.5 Receive Control Unit

- I/O interface mode:

  In the SCLK output mode with SC0CR <IOC> set to “0,” the RXD0 pin is sampled on the rising edge of the shift clock output to the SCLK0 pin.

  In the SCLK input mode with SC0CR <IOC> set to “1,” the serial receive data RXD0 pin is sampled on the rising or falling edge of SCLK input depending on the SC0CR <SCLKS> setting.

- Asynchronous (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.

11.3.6 Receive Buffer

The receive buffer is of a dual structure to prevent overrun errors. The first receive buffer (a shift register) stores the received data bit-by-bit. When a complete set of bits have been stored, they are moved to the second receive buffer (SC0BUF). At the same time, the receive buffer full flag (SC0MOD2 “RBFLL”) is set to “1” to indicate that valid data is stored in the second receive buffer. However, if the receive FIFO is set enabled, the receive data is moved to the receive FIFO and this flag is immediately cleared.

If the receive FIFO has been disabled (SCOFCNF <CNFG> = 0 and SC0MOD1<FDPX1:0> =01), the INTRX0 interrupt is generated at the same time. If the receive FIFO has been enabled (SCNFCNF <CNFG> = 1 and SC0MOD1<FDPX1:0> = 01), an interrupt will be generated according to the SC0RFC <RIL1:0> setting.

The CPU will read the data from either the second receive buffer (SC0BUF) or from the receive FIFO (the address is the same as that of the receive buffer). If the receive FIFO has not been enabled, the receive buffer full flag <RBFLL> is cleared to “0” by the read operation. The next data received can be stored in the first receive buffer even if the CPU has not read the previous data from the second receive buffer (SC0BUF) or the receive FIFO.

If SCLK is set to generate clock output in the I/O interface mode, the double buffer control bit SC0MOD2 <WBUF> can be programmed to enable or disable the operation of the second receive buffer (SCOBUF).

By disabling the second receive buffer (i.e., the double buffer function) and also disabling the receive FIFO (SCOFCNF <CNFG> = 0 and <FDPX1:0> = 01), handshaking with the other side of communication can be enabled and the SCLK output stops each time one frame of data is transferred. In this setting, the CPU reads data from the first receive buffer. By the read operation of CPU, the SCLK output resumes.

If the second receive buffer (i.e., double buffering) is enabled but the receive FIFO is not enabled, the SCLK output is stopped when the first receive data is moved from the first receive buffer to the second receive buffer and the next data is stored in the first buffer filling both buffers with valid data. When the second receive buffer is read, the data of the first receive buffer is moved to the second receive buffer and the SCLK output is resumed upon generation of the receive interrupt INTRX. Therefore, no buffer overrun error will be caused in the I/O interface SCLK output mode regardless of the setting of the double buffer control bit SC0MOD2 <WBUF>.

If the second receive buffer (double buffering) is enabled and the receive FIFO is also enabled (SCNFCNF <CNFG> = 1 and <FDPX1:0> = 01/11), the SCLK output will be stopped when the receive FIFO is full (according to the setting of SCOFNCF <RFST>) and both the first and second receive buffers contain valid data. Also in this case, if SCOFCNF <RXTXCNT> has been set to “1,” the receive control bit RXE will be automatically cleared upon suspension of the SCLK output. If it is set to “0,” automatic clearing will not be performed.

(Note) In this mode, the SC0CR <OEER> flag is insignificant and the operation is undefined. Therefore, before switching from the SCLK output mode to another mode, the SC0CR register must be read to initialize this flag.

In other operating modes, the operation of the second receive buffer is always valid, thus improving the performance of continuous data transfer. If the receive FIFO is not enabled, an overrun error occurs when the data in the second receive buffer (SC0BUF) has not been read before the first receive buffer is full with the next receive data. If an overrun error occurs, data in the first receive buffer will be lost while data in the second receive buffer and the contents of SC0CR <RB8> remain intact. If the receive FIFO is enabled, the FIFO must be read before the FIFO is full and the second receive buffer is written by the next data through the first buffer. Otherwise, an overrun error will be generated and the receive FIFO overrun error flag will be set. Even in this case, the data already in the receive FIFO remains intact.

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 SC0CR <RB8>.

In the 9-bit UART mode, the slave controller can be operated in the wake-up mode by

setting the wake-up function SC0MOD0 <WU> to “1.” In this case, the interrupt INTRX0 will be generated only when SC0CR <RB8> is set to “1.”

11.3.7 Receive FIFO Buffer

In addition to the double buffer function already described, data may be stored using the receive FIFO buffer. By setting <CNFG> of the SC0FCNF register and <FDPX1:0> of the SC0MOD1 register, the 4-byte receive buffer can be enabled. Also, in the UART mode or I/O interface mode, data may be stored up to a predefined fill level. When the receive FIFO buffer is to be used, be sure to enable the double buffer function.

If data with parity bit is to be received in the UART mode, parity check must be performed each time a data frame is received.

11.3.8 Receive FIFO Operation

① I/O interface mode with SCLK output:

The following example describes the case a 4-byte data stream is received in the half duplex mode:

SC0RFC<7:6>=01: Clears receive FIFO and sets the condition of interrupt generation.

SC0RFC<1:0>=00: Sets the interrupt to be generated at fill level 4.

SC0FCNF <4:0>=10111: Automatically inhibits continued reception after reaching the fill level. The number of bytes to be used in the receive FIFO is the same as the interrupt generation fill level.

In this condition, 4-byte data reception may be initiated by setting the half duplex transmission mode and writing “1” to the RXE bit. After receiving 4 bytes, the RXE bit is automatically cleared and the receive operation is stopped (SCLK is stopped).

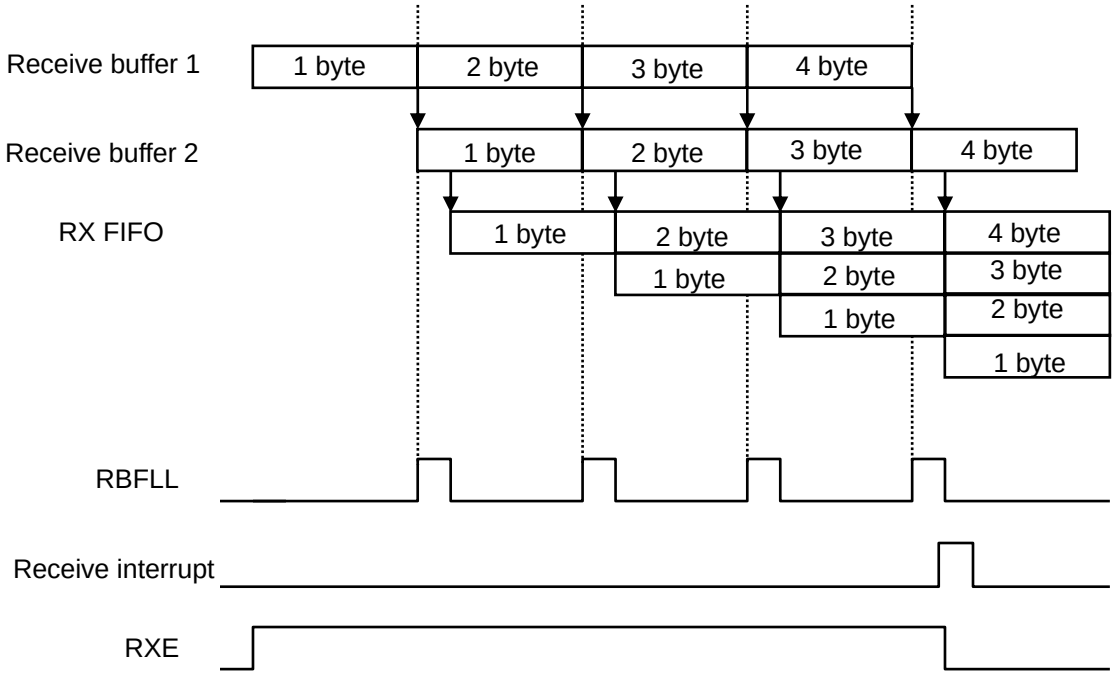


Fig. 11-3 Receive FIFO Operation

② I/O interface mode with SCLK input:

The following example describes the case a 4-byte data stream is received:

SC0RFC <7:6> = 10: Clears receive FIFO and sets the condition of interrupt generation

SC0RFC <1:0> = 00: Sets the interrupt to be generated at fill level 4.

SC0FCNF <1:0> = 10101: Automatically allows continued reception after reaching the fill level. The number of bytes to be used in the receive FIFO is the maximum allowable number.

In this condition, 4-byte data reception may be initiated by setting the half duplex transmission mode and writing “1” to the RXE bit. After receiving 4 bytes, receive FIFO interrupt is generated. This setting enables the next data reception as well. The next 4 bytes can be received before all the data is read from FIFO.

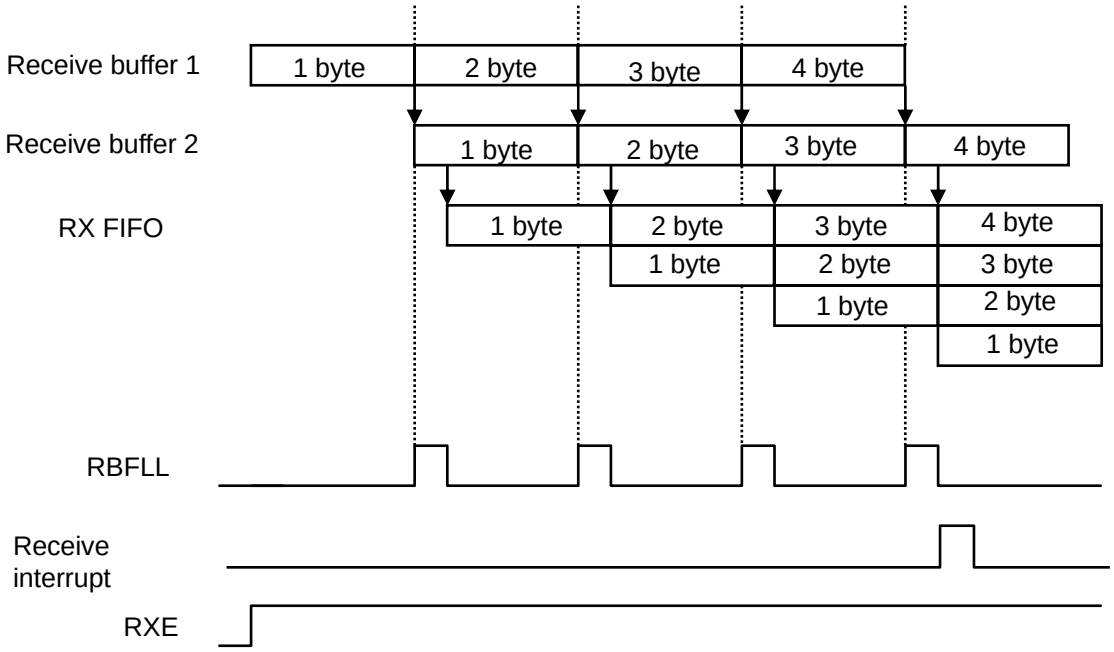


Fig. 11-4 Receive FIFO Operation

11.3.9 Transmit Counter

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

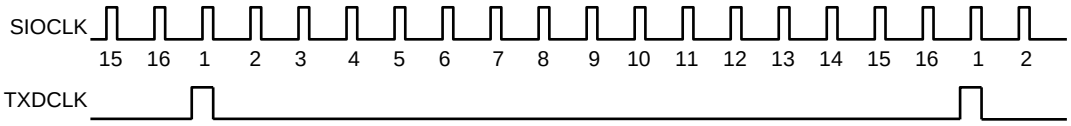


Fig. 11-5 Transmit Clock Generation

11.3.10 Transmit Control Unit

- I/O interface mode:

  In the SCLK output mode with SC0CR <IOC> set to “0,” each bit of data in the transmit buffer is output to the TXD0 pin on the rising edge of the shift clock output from the SCLK0 pin.

  In the SCLK input mode with SC0CR <IOC> set to “1,” each bit of data in the transmit buffer is output to the TXD0 pin on the rising or falling edge of the input SCLK signal according to the SC0CR <SCLKS> setting.

- Asynchronous (UART) mode:

  When the CPU writes data to the transmit buffer, data transmission is initiated on the rising edge of the next TXDCLK and the transmit shift clock (TXDSFT) is also generated.

- Handshake function

The $\overline{CTS}$ pin enables frame by frame data transmission so that overrun errors can be prevented. This function can be enabled or disabled by SC0MOD0 <CTSE>.

When the $\overline{CTS0}$ pin is set to the "H" level, the current data transmission can be completed but the next data transmission is suspended until the $\overline{CTS0}$ pin returns to the "L" level. However in this case, the INTTX0 interrupt is generated, the next transmit data is requested to the CPU, data is written to the transmit buffer, and it waits until it is ready to transmit data.

Although no $\overline{RTS}$ pin is provided, a handshake control function can be easily implemented by assigning a port for the $\overline{RTS}$ function. By setting the port to "H" level upon completion of data reception (in the receive interrupt routine), the transmit side can be requested to suspend data transmission.

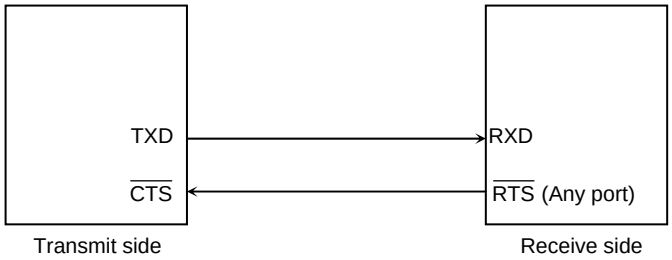


Fig. 11-6 Handshake Function

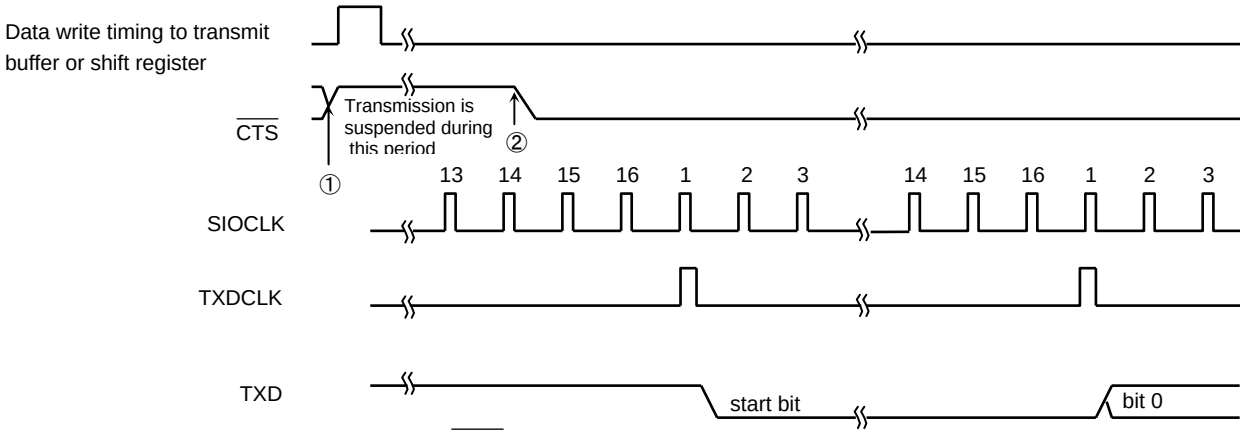


Fig. 11-7 $\overline{CTS}$ (Clear to Transmit) Signal Timing

(Note 1)	**If the $\overline{CTS}$ signal is set to "H" 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 $\overline{CTS}$ is set to "L."**

11.3.11 Transmit Buffer

The transmit buffer (SC0BUF) is in a dual structure. The double buffering function may be enabled or disabled by setting the double buffer control bit <WBUF> in serial mode control register 2 (SC0MOD2). If double buffering is enabled, data written to Transmit Buffer 2 (SCOBUF) is moved to Transmit Buffer 1 (shift register).

If the transmit FIFO has been disabled (SCOFCNF <CNFG> = 0 or 1 and SC0MOD1＜FDPX1:0＞＝01), the INTTX0 interrupt is generated at the same time and the transmit buffer empty flag <TBEMP> of SC0MOD2 is set to “1.” This flag indicates that Transmit Buffer 2 is now empty and that the next transmit data can be written. When the next data is written to Transmit Buffer 2, the <TBEMP> flag is cleared to “0.”

If the transmit FIFO has been enabled (SCNFCNF <CNFG> = 1 and SC0MOD1＜FDPX1:0＞＝10/11), any data in the transmit FIFO is moved to the Transmit Buffer 2 and <TBEMP> flag is immediately cleared to “0.” The CPU writes data to Transmit Buffer 2 or to the transmit FIFO.

If the transmit FIFO is disabled in the I/O interface SCLK input mode and if no data is set in Transmit Buffer 2 before the next frame clock input, which occurs upon completion of data transmission from Transmit Buffer 1, an under-run error occurs and a serial control register (SC0CR) <PERR> parity/under-run flag is set.

If the transmit FIFO is enabled in the I/O interface SCLK input mode, when data transmission from Transmit Buffer 1 is completed, the Transmit Buffer 2 data is moved to Transmit Buffer 1 and any data in transmit FIFO is moved to Transmit Buffer 2 at the same time.

If the transmit FIFO is disabled in the I/O interface SCLK output mode, when data in Transmit Buffer 2 is moved to Transmit Buffer 1 and the data transmission is completed, the SCLK output stops. So, no under-run errors can be generated.

If the transmit FIFO is enabled in the I/O interface SCLK output mode, the SCLK output stops upon completion of data transmission from Transmit Buffer 1 if there is no valid data in the transmit FIFO.

Note) In the I/O interface SCLK output mode, the SC0CR <PEER> flag is insignificant. In this case, the operation is undefined. Therefore, to switch from the SCLK output mode to another mode, SC0CR must be read in advance to initialize the flag.

If double buffering is disabled, the CPU writes data only to Transmit Buffer 1 and the transmit interrupt INTTX0 is generated upon completion of data transmission.

If handshaking with the other side is necessary, set the double buffer control bit <WBUF> to “0” (disable) to disable Transmit Buffer 2; any setting for the transmit FIFO should not be performed.

11.3.12 Transmit FIFO Buffer

In addition to the double buffer function already described, data may be stored using the transmit FIFO buffer. By setting <CNFG> of the SC0FCNF register and <FDPX1:0> of the SC0MOD1 register, the 4-byte transmit buffer can be enabled. In the UART mode or I/O interface mode, up to 4 bytes of data may be stored.
If data is to be transmitted with a parity bit in the UART mode, parity check must be performed on the receive side each time a data frame is received.

11.3.13 Transmit FIFO Operation

① I/O interface mode with SCLK output (normal mode):

The following example describes the case a 4-byte data stream is transmitted:

SC0TFC <7:6> = 01: Clears transmit FIFO and sets the condition of interrupt generation

SC0TFC <1:0> = 00: Sets the interrupt to be generated at fill level 0.

SC0FCNF <4:0> = 01011: Inhibits continued transmission after reaching the fill level.

In this condition, data transmission can be initiated by setting the transfer mode to half duplex, writing 4 bytes of data to the transmit FIFO, and setting the <TXE> bit to “1.” When the last transmit data is moved to the transmit buffer, the transmit FIFO interrupt is generated. When transmission of the last data is completed, the clock is stopped and the transmission sequence is terminated.

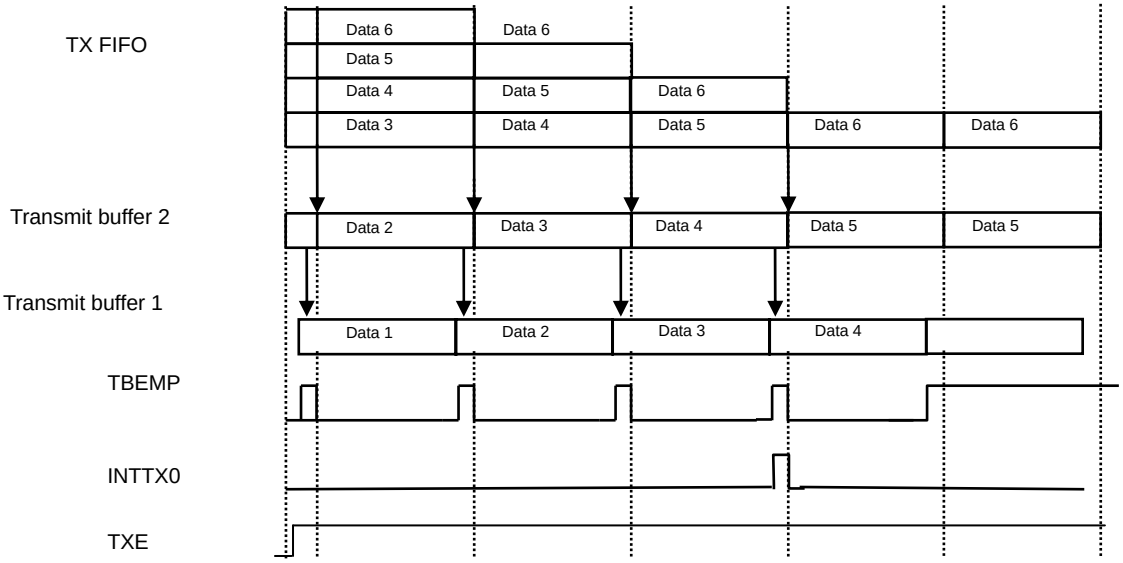


Fig. 11-8 Transmit FIFO Operation

② I/O interface mode with SCLK input (normal mode):

The following example describes the case a 4-byte data stream is transmitted:

SC0TFC <1:0> = 01: Clears the transmit FIFO and sets the condition of interrupt generation.

SC0TFC <7:6> = 000000: Sets the interrupt to be generated at fill level 0.

SC0FCNF <4:0> = 01001: Allows continued transmission after reaching the fill level.

In this condition, data transmission can be initiated along with the input clock by setting the transfer mode to half duplex, writing 4 bytes of data to the transmit FIFO, and setting the <TXE> bit to “1.” When the last transmit data is moved to the transmit buffer, the transmit FIFO interrupt is generated.

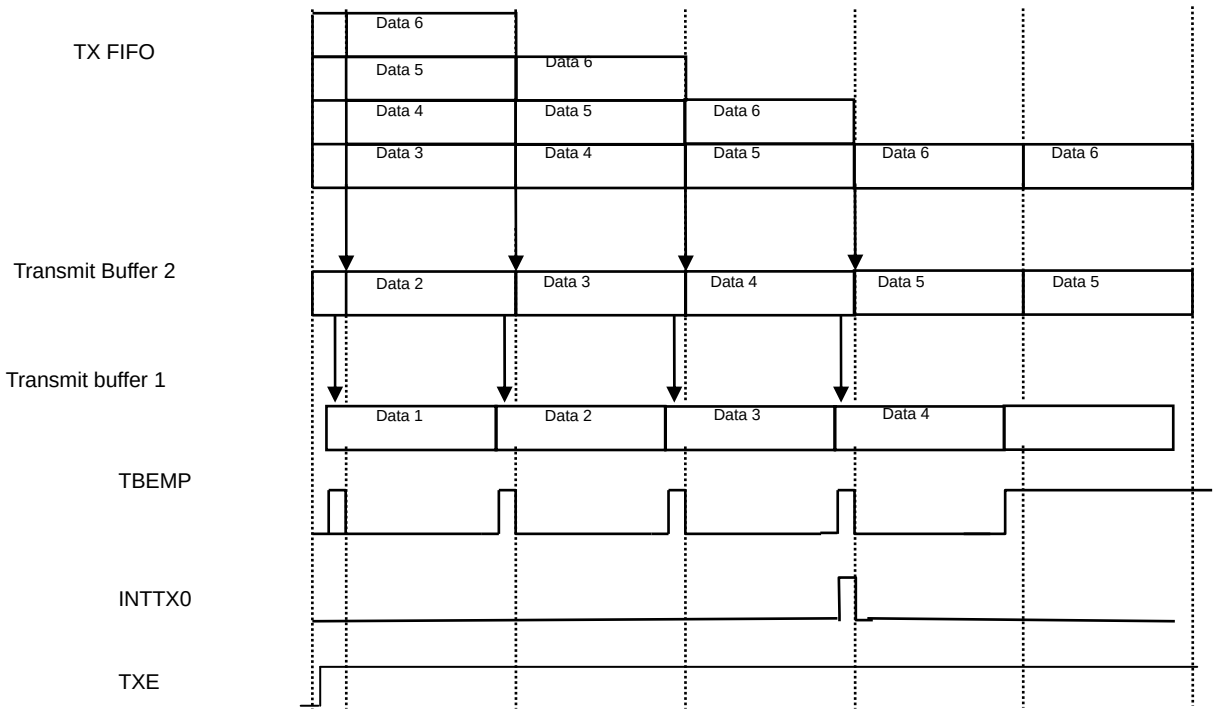


Fig. 11-9 Transmit FIFO Operation

11.3.14 Parity Control Circuit

If the parity addition bit <PE> of the serial control register SC0CR is set to “1,” data is sent with the parity bit. Note that the parity bit may be used only in the 7- or 8-bit UART mode. The <EVEN> bit of SC0CR selects either even or odd parity.

Upon data transmission, the parity control circuit automatically generates the parity with the data written to the transmit buffer (SC0BUF). After data transmission is complete, the parity bit will be stored in SC0BUF bit 7 <TB7> in the 7-bit UART mode and in bit 7 <TB8> in the serial mode control register SC0MOD in the 8-bit UART mode. The <PE> and <EVEN> settings must be completed before data is written to the transmit buffer.

Upon data reception, the parity bit for the received data is automatically generated while the data is shifted to receive buffer 1 and moved to receive buffer 2 (SC0BUF). In the 7-bit UART mode, the parity generated is compared with the parity stored in SC0BUF <RB7>, while in the 8-bit UART mode, it is compared with the bit 7 <RB8> of the SC0CR register. If there is any difference, a parity error occurs and the <PERR> flag of the SC0CR register is set.

In the I/O interface mode, the SC0CR <PERR> flag functions as an under-run error flag, not as a parity flag.

11.3.15 Error Flag

Three error flags are provided to inprove the reliability of received data.

1. Overrun error <OERR>: Bit 4 of the serial control register SC0CR

   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 receive data before the receive buffer has been read. If the receive FIFO is enabled, the received data is automatically moved to the receive FIFO and no overrun error will be generated until the receive FIFO is full (or until the usable bytes are fully occupied). This flag is set to “0” when it is read. In the I/O interface SCLK output mode, no overrun error is generated and therefore, this flag is inoperative and the operation is undefined.

2. Parity error/under-run error <PERR>: Bit 3 of the SC0CR register

   In the UART mode, this bit is set to “1” when a parity error is generated. A parity error is generated when the parity generated from the received data is different from the parity received. This flag is set to “0” when it is read.

   In the I/O interface mode, this bit indicates an under-run error. When the double buffer control bit <WBUF> of the serial mode control register SC0MOD2 is set to “1” in the SCLK input mode, if no data is set to the transmit double buffer before the next data transfer clock after completing the transmission from the transmit shift register, this error flag is set to “1” indicating an under-run error. If the transmit FIFO is enabled, any data content in the transmit FIFO will be moved to the buffer. When the transmit FIFO and the double buffer are both empty, an under-run error will be generated. Because no under-run errors can be generated in the SCLK output mode, this flag is inoperative and the operation is undefined. If Transmit Buffer 2 is disabled, the under-run flag <PERR> will not be set. This flag is set to “0” when it is read.

3. Framing error <FERR>: Bit 2 of the SC0CR register

   In the UART mode, this bit is set to “1” when a framing error is generated. This flag is set to “0” when it is read. 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 <SBLEN> (stop bit length) setting of the serial mode control register 2, SC0MOD2, the stop bit status is determined by only 1 bit on the receive side.

Operation mode	Error flag	Function
UART	OERR	Overrun error flag
	PERR	Parity error flag
	FERR	Framing error flag
I/O Interface (SCLK input)	OERR	Overrun error flag
	PERR	Underrun error flag (WBUF = 1)
		Fixed to 0 (WBUF = 0)
	FERR	Fixed to 0
I/O Interface (SCLK output)	OERR	Operation undefined
	PERR	Operation undefined
	FERR	Fixed to 0

11.3.16 Direction of Data Transfer

In the I/O interface mode, the direction of data transfer can be switched between “MSB first” and “LSB first” by the data transfer direction setting bit <DRCHG> of the SC0MOD2 serial mode control register 2. Don't switch the direction when data is being transferred.

11.3.17 Stop Bit Length

In the UART transmission mode, the stop bit length can be set to either 1 or 2 bits by bit 4 <SBLEN> of the SC0MOD2 register.

11.3.18 Status Flag

If the double buffer function is enabled (SC0MOD2 <WBUF> = “1”), the bit 6 flag <RBFLL> of the SC0MOD2 register indicates the condition of receive buffer full. When one frame of data has been received and transferred from buffer 1 to buffer 2, this bit is set to “1” to show that buffer 2 is full (data is stored in buffer 2). When the receive buffer is read by CPU/DMAC, it is cleared to “0.” If <WBUF> is set to “0,” this bit is insignificant and must not be used as a status flag. When double buffering is enabled (SC0MOD2 <WBUF> = “1”), the bit 7 flag <TBEMP> of the SC0MOD2 register indicates that Transmit Buffer 2 is empty. When data is moved from Transmit Buffer 2 to Transmit Buffer 1 (shift register), this bit is set to “1” indicating that Transmit Buffer 2 is now empty. When data is set to the transmit buffer by CPU/DMAC, the bit is cleared to “0.” If <WBUF> is set to “0,” this bit is insignificant and must not be used as a status flag.

11.3.19 Configurations of Transmit/Receive Buffer

		<WBUF> = 0	<WBUF> = 1
UART	Transmit buffer	Single	Double
	Receive buffer	Double	Double
I/O Interface (SCLK input)	Transmit buffer	Single	Double
	Receive buffer	Double	Double
I/O Interface (SCLK output)	Transmit buffer	Single	Double
	Receive buffer	Single	Double

11.3.20 Software reset

Software reset is generated by writing the bits 1 and 0 of SC0MOD2 <SWRST1:0> as “10” followed by “01”. As a result, SC0MOD0<RXE>, SC0MOD1<TXE>, SC0MOD2<TBEMP>,<RBFLL>,<TXRUN> of mode registers and SC0CR<OERR>, <PERR>, <FERR> of control registers and internal circuit is initialized. Other states are maintained.

11.3.21 Signal Generation Timing

① UART Mode:

Receive Side

Mode	9-bit	8-bit with parity	8-bit, 7-bit, and 7-bit with parity
Interrupt generation timing	Around the center of the 1st stop bit	Around the center of the 1st stop bit	Around the center of the 1st stop bit
Framing error generation timing	Around the center of the stop bit	Around the center of the stop bit	Around the center of the stop bit
Parity error generation timing	—	Around the center of the last (parity) bit	Around the center of the last (parity) bit
Overrun error generation timing	Around the center of the stop bit	Around the center of the stop bit	Around the center of the stop bit

Transmit Side

Mode	9-bit	8-bit with parity	8-bit, 7-bit, and 7-bit with parity
Interrupt generation timing (<WBUF> = 0)	Just before the stop bit is sent	Just before the stop bit is sent	Just before the stop bit is sent
Interrupt generation timing (<WBUF> = 1)	Immediately after data is moved to transmit buffer 1 (just before start bit transmission)	Immediately after data is moved to transmit buffer 1 (just before start bit transmission).	Immediately after data is moved to transmit buffer 1 (just before start bit transmission)

② I/O interface mode:

Receive Side

Interrupt generation timing (<WBUF> = 0)	SCLK output mode	Immediately after the rising edge of the last SCLK
	SCLK input mode	Immediately after the rising or falling edge of the last SCLK (for rising or falling edge mode, respectively).
Interrupt generation timing (<WBUF> = 1)	SCLK output mode	Immediately after the rising edge of the last SCLK (just after data transfer to receive buffer 2) or just after receive buffer 2 is read.
	SCLK input mode	Immediately after the rising edge or falling edge of the last SCLK (right after data is moved to receive buffer 2).
Overrun error generation timing	SCLK input mode	Immediately after the rising or falling edge of the last SCLK (for rising or falling edge mode, respectively)

Transmit Side

Interrupt generation timing (<WBUF> = 0)	SCLK output mode	Immediately after the rising edge of the last SCLK
	SCLK input mode	Immediately after the rising or falling edge of the last SCLK (for rising or falling edge mode, respectively)
Interrupt generation timing (<WBUF> = 1)	SCLK output mode	Immediately after the rising edge of the last SCLK or just after data is moved to Transmit Buffer 1
	SCLK input mode	Immediately after the rising or falling edge of the last SCLK or just after data is moved to Transmit Buffer 1
Under-run error generation timing	SCLK input mode	Immediately after the falling or rising edge of the next SCLK

(Note 1) Do not modify any control register when data is being sent or received (in a state ready to transmit or receive).
(Note 2) Do not stop the receive operation (by setting SC0MOD0 <RXE> = “0”) when data is being received.
(Note 3) Do not stop the transmit operation (by setting SC0MOD1 <TXE> = “0”) when data is being transmitted.

11.4 Register Description (Only for Channel 0)

The channel 0 registers are described here. Each register for all the channels operates in the same way.

11.4.1 Enable register

SC0EN

	7	6	5	4	3	2	1	0
bit Symbol								SIOE
Read/Write	R							R/W
After reset	0							0
Function	"0" is read.							SIO operation 0: disabled 1: enabled

<SIOE>: Specified the SIO operation.
To use the SIO, enable the SIO operation.
When the operation is disabled, no clock is supplied to the other registers in the SIO module. This can reduce the power consumption.
If the SIO operation is executed and then disabled, the settings will be maintained in each register.

11.4.2 Buffer register

SC0BUF works as a transmit buffer for WR operation and as a receive buffer for RD operation.

SC0BUF

	7	6	5	4	3	2	1	0
bit Symbol	TB7/RB7	TB6/RB6	TB5/RB5	TB4/RB4	TB3/RB3	TB2/RB2	TB1/RB1	TB0/RB0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	TB7~0 : Transmit buffer/FIFO RB7~0 : Receive buffer/FIFO							

<TB7:0> Transmit buffer (at WR operation).

<RB7:0> Receive buffer (at RD operation).

11.4.3 Control register

SC0CR

	7	6	5	4	3	2	1	0
bit Symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (Cleared to “0” when read)			R/W	
After reset	0	0	0	0	0	0	0	0
Function	Receive data bit 8 **(For UART)**	Parity **(For UART)** 0: Odd 1: Even	Add parity **(For UART)** 0: Disabled 1: Enabled	0: Normal operation 1: Error			0: SCLK0 1: SCLK0	**(For I/O interface)** 0: Baud rate generator 1: SCLK0 pin input
				Overrun	Parity/ underrun	Framing		

<RB8>: 9th bit of the received data in the 9 bits UART mode.

<EVEN>: Selects even or odd parity.
“0”: odd parity.
“1”: even parity.
The parity bit may be used only in the 7- or 8-bit UART mode.

<PE>: Controls enabling/ disabling parity.
The parity bit may be used only in the 7- or 8-bit UART mode.

<OERR>: Error flag **(see note)**
<PERR>: Indicate overrun error, parity error, underrun error and framing error.
<FERR>:

<SCLKS>: Selects edge for data transmission and reception.
“0”: Data transmit/receive at rising edges of SCLK0
“1”: Data transmit/receive at falling edges of SCLK0

<IOC>: Selects input clock in the I/O interface mode.
“0”: baud rate generator
“1”: SCLK0 pin input.

(Note) Any error flag is cleared when read.

11.4.4 Mode control register 0

SC0MOD0

	7	6	5	4	3	2	1	0
bit Symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Transmit data bit 8	Handshake function control 0: CTS diable 1: CTS enable	Receive control 0: Reception disabled 1: Reception enabled	Wake-up function 0: Reception disabled 1: Reception enabled	Serial transfer mode 00: I/O interface mode 01: 7-bit length UART mode 10: 8-bit length UART mode 11: 9-bit length UART mode		Serial transfer clock **(for UART)** 00: Timer TB7OUT 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clock (SCLK0 input)	

<TB8>: Writes the 9th bit of transmit data in the 9 bits UART mode.

<CTSE>: Controls handshake function.
Setting “1” enables handshake function using $\overline{CTS}$ pin.

<RXE>: Controls reception **(see note).**
Set <RXE> after setting each mode register (SC0MOD0, SC0MOD1 and SC0MOD2).

<WU>: Controls wake-up function.
This function is available only at 9-bit UART mode.

	9-bit UART mode	Other modes
0	Interrupt when received	don’t care
1	Interrupt only when RB9=1	

<SM1:0>: Specifies transfer mode.

<SC1:0>: Selects the serial transfer clock in the UART mode.
As for the I/O interface mode, the serial transfer clock can be set in the control register SC0CR.

(Note) With <RXE> set to “0,” set each mode register (SC0MOD0, SC0MOD1 and SC0MOD2). Then set <RXE> to “1.”

11.4.5 Mode control register 1

SC0MOD1

	7	6	5	4	3	2	1	0
bit Symbol	I2S0	FDPX1	FDPX0	TXE	SINT2	SINT1	SINT0	-
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	IDLE 0: Stop 1: Start	Transfer mode setting 00: Transfer prohibited 01: Half duplex(RX) 10:Half duplex(TX) 11:Full duplex		Transmit control 0:Disabled 1: Enabled	Interval time of continuous transmission **(for I/O interface)** 000: None 100:8SCLK 001:1SCLK 101:16SCLK 010:2SCLK 110:32SCLK 011:4SCLK 111:64SCLK			Write “0”.

<I2S0>: Specifies the IDLE mode operation.

<FDPX1:0>: Configures the transfer mode in the I/O interface mode. Also configures the FIFO if it is enabled. In the UART mode, it is used only to specify the FIFO configuration.

<TXE>: This bit enables transmission and is valid for all the transfer modes (**see note)**. If disabled while transmission is in progress, transmission is inhibited only after the current frame of data is completed for transmission.

<SINT2:0>: Specifies the interval time of continuous transmission when double buffering or FIFO is enabled in the I/O interface mode. This parameter is valid only for the I/O interface mode when SCLK0 pin input is not selected.

(Note) Specify the mode first and then specify the <TXE> bit.

11.4.6 Mode control register 2

SC0MOD2

	7	6	5	4	3	2	1	0
bit Symbol	TBEMP	RBFLL	TXRUN	SBLEN	DRCHG	WBUF	SWRST1	SWRST0
Read/Write	R			R/W				
After reset	1	0	0	0	0	0	0	0
Function	Transmit buffer empty flag 0: full 1: Empty	Receive Buffer full flag 0: Empty 1: full	In transmission flag 0: Stop 1: Start	STOP bit **(for UART)** 0: 1-bit 1: 2-bit	Setting transfer direction 0: LSB first 1: MSB first	W-buffer 0: Disabled 1: Enabled	SOFT RESET Overwrite "01" on "10"to reset.	

<TBEMP>: This flag shows that the transmit double buffers are empty. When data in the transmit double buffers is moved to the transmit shift register and the double buffers are empty, this bit is set to "1." Writing data again to the double buffers sets this bit to "0."
If double buffering is disabled, this flag is insignificant.

<RBFLL>: This is a flag to show that the receive double buffers are full. When a receive operation is completed and received data is moved from the receive shift register to the receive double buffers, this bit changes to "1" while reading this bit changes it to "0."
If double buffering is disabled, this flag is insignificant.

<TXRUN>: This is a status flag to show that data transmission is in progress.
<TXRUN> and <TBEMP> bits indicate the following status.

<TXRUN>	<TBEMP>	Status
1	-	Transmission in progress
0	1	Transmission completed
	0	Wait state with data in TX buffer

<SBLEN>: This specifies the length of stop bit transmission in the UART mode. On the receive side, the decision is made using only a single bit regardless of the <SBLEN> setting.

<DRCHG>: Specifies the direction of data transfer in the I/O interface mode. In the UART mode, it is fixed to LSB first.

<WBUF>: This parameter enables or disables the transmit/receive buffers to transmit (in both SCLK output/input modes) and receive (in SCLK output mode) data in the I/O interface mode and to transmit data in the UART. When receiving data in the I/O interface mode (I SCLK input) and UART mode, double buffering is enabled in both cases that 0 or 1 is set to <WBUF> bit.

<SWRST1:0>: Overwriting "01" in place of "10" generates a software reset. When this software reset is executed, the following bits and their internal circuits are initialized **(see note 1, 2 and 3)**.

Register	Bit
SC0MOD0	RXE
SC0MOD1	TXE
SC0MOD2	TBEMP,RBFLL,TXRUN,
SC0CR	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.
(Note 3) A software reset initializes other bits. Resetting a mode register and a control register are needed.

11.4.7 Baud rate generator control register(SC0BRCR) Baud rate generator control register 2(SC0BRADD)

SC0BRCR

	7	6	5	4	3	2	1	0
bit Symbol	-	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Write "0".	N + (16 − K)/16 divider function 0: disabled 1: enabled	Select input clock to the baud rate generator 00: φT1 01: φT4 10: φT16 11: φT64		Division ratio "N" 0000: 16 0001: 1 0010: 2 : 1111: 15			

SC0BRADD

	7	6	5	4	3	2	1	0
bit Symbol					BR0K3	BR0K2	BR0K1	BR0K0
Read/Write	R				R/W			
After reset	0				0	0	0	0
Function	"0" is read.				Specify K for the "N + (16 − K)/16" division 0000: Prohibited 0001: K=1 0010: K=2 : 1111: K=15			

<RB0ADDE>: Specifies N + (16-K)/16 division function.
N + (16-K)/16 division function can only be used in the UART mode.

<RB0CK1:0>: Specifies the baud rate generator input clock.

<RB0S3:0>: Specifies division ratio "N".

<RB0K3:0>: Specifies K for the "N+(16-K)/16" division.

The division ratio of the baud rate generator can be specified in the registers shown above. Table 11-6 lists the settings of baud rate generator division ratio.

Table 11-6 Setting division ratio

	BR0ADDE=0	BR0ADDE=1 **(Note 1)** (Only UART)
BR0S	Specify "N" **(Note 2) (Note 3)**	
BR0K	No setting required	Setting "K" **(Note 4)**
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 BR0K <BR0ADDE> to “1” after setting the K value to BR0K. The “N + (16 - K)/16” division function can only be used in the UART mode.**
(Note 2)	**The division ratio “1” of the baud rate generator can be specified only when** **・the “N + (16 - K)/16” division function is not used in the UART mode.** **・double buffering is used in the I/O interface mode.**
(Note 3)	**As a division ratio, 1 (“0001”) or 16 (“0000”) cannot be applied to N when using the “N + (16 - K)/16” division function.**
(Note 4)	**Specifying “K = 0” is prohibited.**

11.4.8 FIFO configuration register

SC0FCNF

	7	6	5	4	3	2	1	0
bit Symbol	Reserved	Reserved	Reserved	RFST	TFIE	RFIE	RXTXCNT	CNFG
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	**Be sure to write "000".**			Bytes used in RX FIFO 0: Maximum 1:Same as FILL level of RX FIFO	TX interrupt for TX FIFO 0: Disabled 1: Enabled	RX interrupt for RX FIFO 0:Disabled 1: Enabled	Automatic disable of RXE/TXE 0:None 1:Auto disable	FIFO enable 0: Disabled 1: Enabled

<RFST>: When RX FIFO is enabled, the number of RX FIFO bytes to be used is selected **(see note)**.
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 <RIL1:0>.

<TFIE>: When TX FIFO is enabled, transmit interrupts are enabled or disabled by this parameter.

<RFIE>: When RX FIFO is enabled, receive interrupts are enabled or disabled by this parameter.

<RXTXCNT>: Controls automatic disabling of transmission and reception.
The mode control register SCOMOD1 <FDPX1:0> is used to set the types of TX/RX.
Setting "1" enables to operate as follows.

Half duplex RX	When the RX FIFO is filled up to the specified number of valid bytes, SC0MOD0<RXE> is automatically set to "0" to inhibit further reception.
Half duplex TX	When the TX FIFO is empty, SC0MOD1<TXE> is automatically set to "0" to inhibit further transmission.
Full duplex	When either of the above two conditions is satisfied, TXE/RXE are automatically set to "0" to inhibit further transmission and reception.

<CNFG>: Enables FIFO.
If enabled, the SCOMOD1 <FDPX1:0> setting automatically configures FIFO as follows: (The type of TX/RX can be specified in the mode control register 1 SC0MOD1<FDPX1:0>).

Half duplex RX	RX FIFO 4byte
Half duplex TX	TX FIFO 4byte
Full duplex	RX FIFO 2byte＋TX FIFO 2byte

(Note) Regarding TX FIFO, the maximum number of bytes being configured is always available. The available number of bytes is the bytes already written to the TX FIFO.

11.4.9 RX FIFO configuration register

SC0RFC

	7	6	5	4	3	2	1	0
bit Symbol	RFCS	RFIS					RIL1	RIL0
Read/Write	W	R/W	R				R/W	
After reset	0	0	0				0	0
Function	RX FIFO clear 1: Clear "0" is read.	Select interrupt generation condition 0: when the data reaches to the specified fill level. 1: when the data reaches to the specified fill level or the data exceeds the specified fill level at the time data is read.	"0" is read.				FIFO fill level to generate RX interrupts 00:4byte(2 Byte at full duplex) 01:1byte 10:2byte 11:3byte	

<RFCS>: Clears RX FIFO
Setting "1" clears RX FIFO and "0" is always read.

<RFIS>: Specifies the condition of interrupt generation.
0: An interrupt is generated when it reaches to the specified fill level.
An interrupt is generated when it is reaches to the specified fill level or if it exceeds the specified fill level at the time data is read.

<RIL1:0>: Specifies FIFO fill level**(see note).**

	Other than full duplex	Full duplex
00	4byte	2byte
01	1byte	1byte
10	2byte	2byte
11	3byte	1byte

(Note) RIL1 is ignored when FDPX1:0 = 11 (full duplex)

11.4.10 TX FIFO configuration register

SC0TFC

	7	6	5	4	3	2	1	0
bit Symbol	TFCS	TFIS					TIL1	TIL0
Read/Write	W	R/W	R				R/W	
After reset	0	0	0				0	0
Function	TX FIFO clear 1:Clear Always reads “0”.	Select interrupt generation condition 0: when the data reaches to the specified fill level. 1: when the data reaches to the specified fill level or the data cannot reach the specified fill level at the time new data is read.	“0” is read.				FIFO fill level to generate TX interrupts. 00:Empty 01:1byte 10:2byte 11:3byte Note: TIL1 is ignored when FDPX1:0=11 (full duplex).	

<TFCS>: Clears TX FIFO.
Setting “1” clears TX FIFO and “0” is always read.

<TFIS>: Selects interrupt generation condition.
0: An interrupt is generated when the data reaches to the specified fill level.
1: An interrupt is generated when the data reaches to the specified fill level or the data cannot reach the
specified fill level at the time new data is read.

<TIL1:0>: Selects FIFO fill level **(see note).**

	Other than full duplex	Full duplex
00	Empty	Empty
01	1byte	1byte
10	2byte	Empty
11	3byte	1byte

(Note) TIL1 is ignored when FDPX1:0 = 11 (full duplex).

11.4.11 RX FIFO status register

SC0RST

	7	6	5	4	3	2	1	0
bit Symbol	ROR					RLVL2	RLVL1	RLVL0
Read/Write	R	R				R		
After reset	0	0				0	0	0
Function	RX FIFO Overrun 1: Generated	“0” is read.				Status of RX FIFO fill level 000:Empty 001:1Byte 010:2Byte 011:3Byte 100:4Byte		

<ROR>: Flags for RX FIFO overrun. When the overrun occurs, these bits are set to “1” **(see note).**

<RLVL2:0>: Shows the fill level of RX FIFO.

(Note) The <ROR> bit is cleared to “0” when receive data is read from the SC0BUF register.

11.4.12 TX FIFO status register

SC0TST

	7	6	5	4	3	2	1	0
bit Symbol	TUR					TLVL2	TLVL1	TLVL0
Read/Write	R	R				R		
After reset	1	0				0	0	0
Function	TX FIFO Under run 1:Generated Cleared by writing FIFO	“0” is read.				Status of TX FIFO fill level 000:Empty 001:1Byte 010:2Byte 011:3Byte 100:4Byte		

<TUR>: Flags for TX FIFO underrun. When the underrun occurs, these bits are set to “1” **(see note).**

<TLVL2:0>: Shows the fill level of TX FIFO.

(Note) The <TUR> bit is cleared to “0” when transmit data is written to the SC0BUF register.

11.5 Operation in Each Mode

11.5.1 Mode 0 (I/O interface mode)

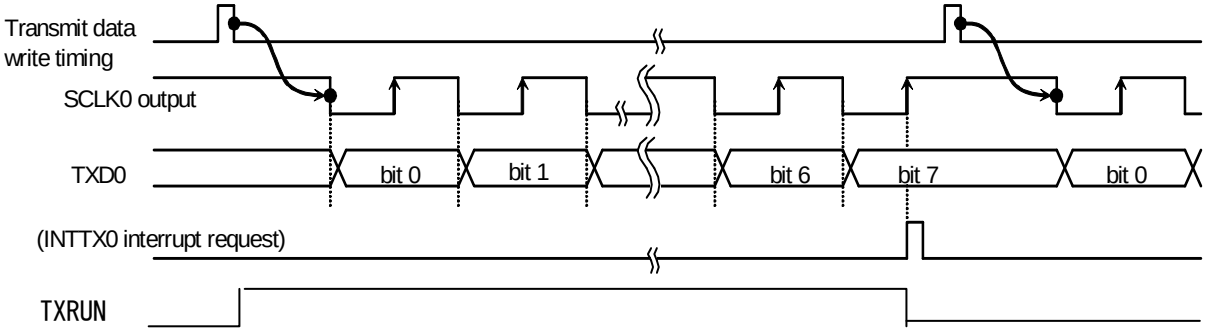
Mode 0 consists of two modes, the “SCLK output” mode to output synchronous clock and the “SCLK input” mode to accept synchronous clock from an external source. The following operational descriptions are for the case use of FIFO is disabled. For details of FIFO operation, refer to the previous sections describing receive/transmit FIFO functions.

① Transmitting data

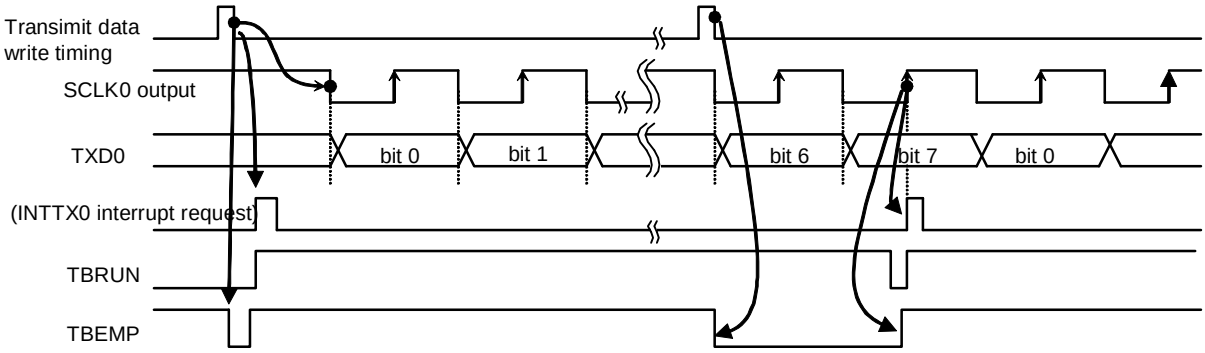
SCLK output mode

In the SCLK output mode, if SC0MOD2<WBUF> is set to “0” and the transmit double buffers are disabled, 8 bits of data are output from the TXD0 pin and the synchronous clock is output from the SCLK0 pin each time the CPU writes data to the transmit buffer. When all data is output, the INTTX0 interrupt is generated.

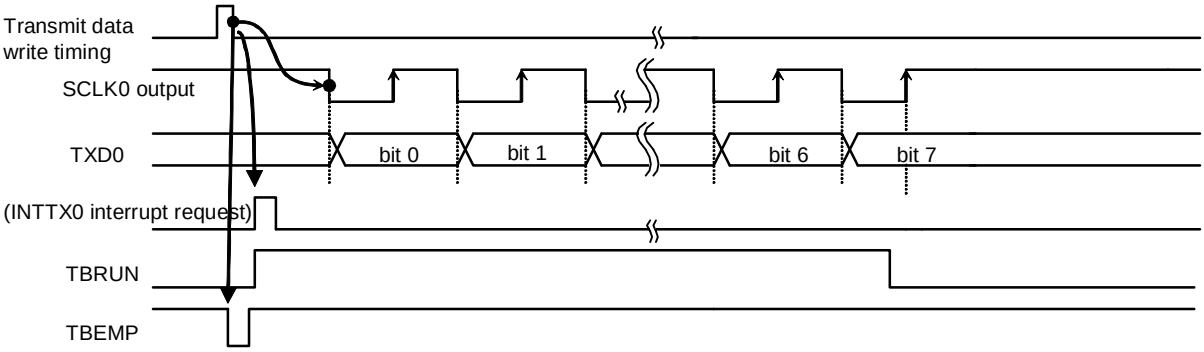
If SC0MOD2 <WBUF> is set to “1” and the transmit double buffers are enabled, data is moved from Transmit Buffer 2 to Transmit Buffer 1 when the CPU writes data to Transmit Buffer 2 while data transmission is halted or when data transmission from Transmit Buffer 1 (shift register) is completed. When data is moved from Transmit Buffer 2 to Transmit Buffer 1, the transmit buffer empty flag SC0MOD2 <TBEMP> is set to “1,” and the INTTX0 interrupt is generated. If Transmit Buffer 2 has no data to be moved to Transmit Buffer 1, the INTTX0 interrupt is not generated and the SCLK0 output stops.



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



<WBUF>=“1” (if double buffering is enabled and there is data in buffer 2)



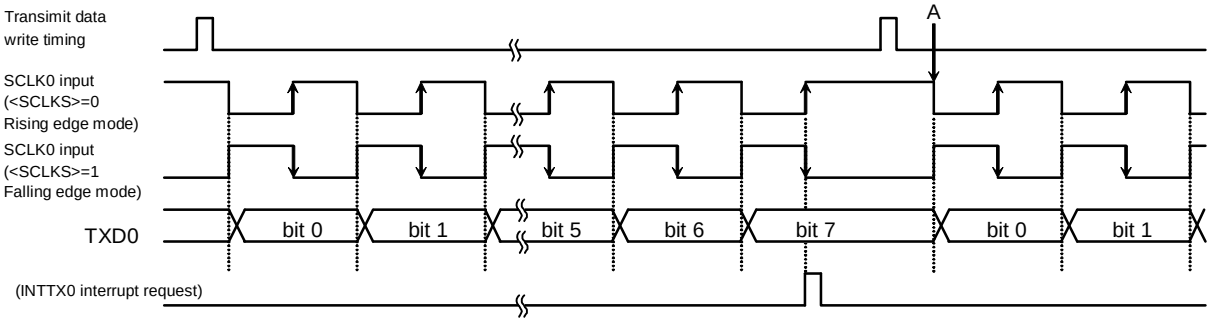
<WBUF>=“1” (if double buffering is enabled and there is no data in buffer 2)

Fig. 11-10 Transmit Operation in the I/O Interface Mode (SCLK0 Output Mode)

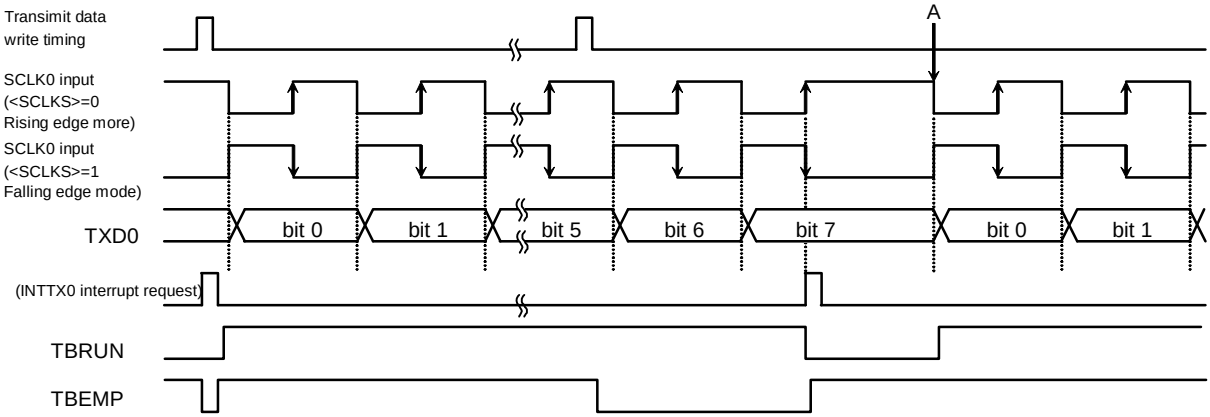
SCLK input mode

In the SCLK input mode, if SC0MOD2 <WBUF> is set to “0” and the transmit double buffers are disabled, 8-bit data that has been written in the transmit buffer is output from the TXD0 pin when the SCLK0 input becomes active. When all 8 bits are sent, the INTTX0 interrupt is generated. The next transmit data must be written before the timing point “A” as shown in Fig. 11-11.

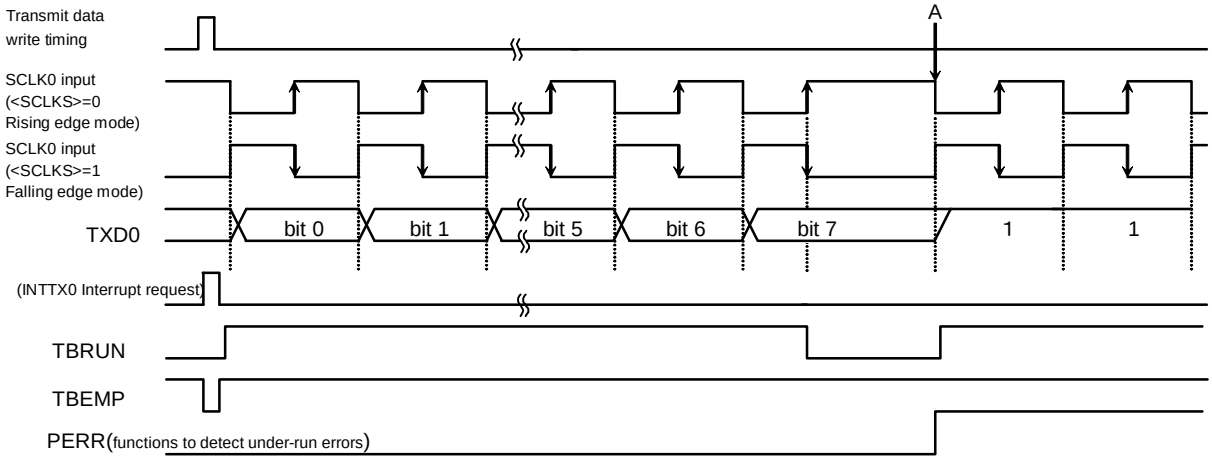
If SC0MOD2 <WBUF> is set to “1” and the transmit double buffers are enabled, data is moved from Transmit Buffer 2 to Transmit Buffer 1 when the CPU writes data to Transmit Buffer 2 before the SCLK0 becomes active or when data transmission from Transmit Buffer 1 (shift register) is completed. As data is moved from Transmit Buffer 2 to Transmit Buffer 1, the transmit buffer empty flag SC0MOD2 <TBEMP> is set to “1” and the INTTX0 interrupt is generated. If the SCLK0 input becomes active while no data is in Transmit Buffer 2, although the internal bit counter is started, an under-run error occurs and 8-bit dummy data (FFh) is sent.



<WBUF>=“0” (if double buffering is disabled



<WBUF>=“1” (if double buffering is enabled and there is data in buffer 2)



<WBUF>=“1” (if double buffering is enabled and there is no data in buffer 2)

Fig. 11-11 Transmit Operation in the I/O Interface Mode (SCLK0 Input Mode)

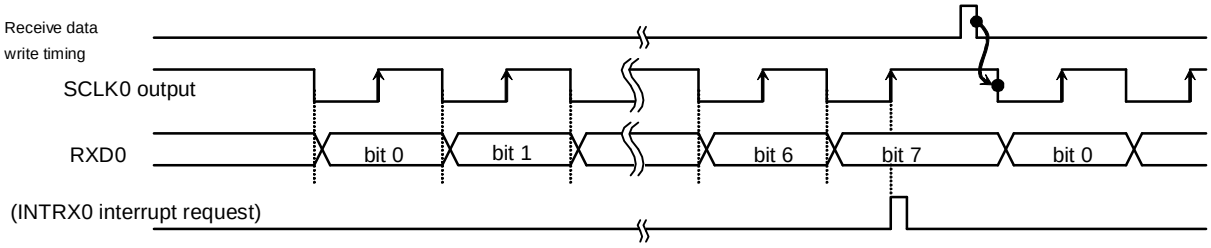
② Receiving data

SCLK output mode

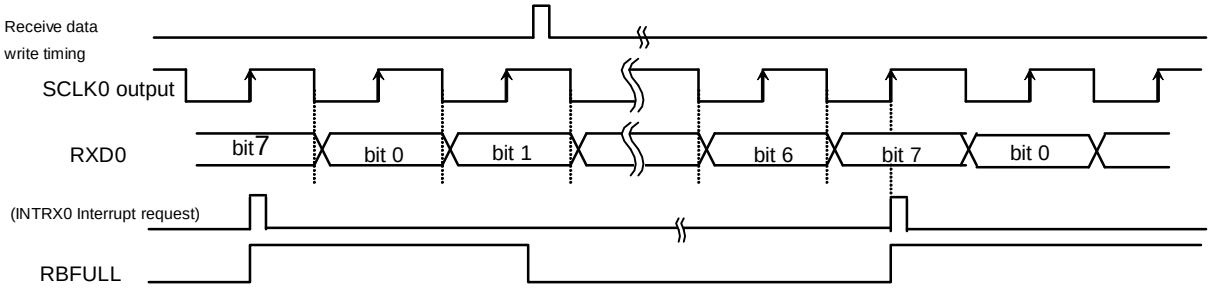
In the SCLK output mode, if SC0MOD2 <WBUF> = “0” and receive double buffering is disabled, a synchronous clock pulse is output from the SCLK0 pin and the next data is shifted into receive buffer 1 each time the CPU reads received data. When all the 8 bits are received, the INTRX0 interrupt is generated.

The first SCLK output can be started by setting the receive enable bit SC0MOD0 <RXE> to “1.” If the receive double buffering is enabled with SC0MOD2 <WBUF> set to “1,” the first frame received is moved to receive buffer 2 and receive buffer 1 can receive the next frame successively. As data is moved from receive buffer 1 to receive buffer 2, the receive buffer full flag SC0MOD2 <RBFULL> is set to “1” and the INTRX0 interrupt is generated.

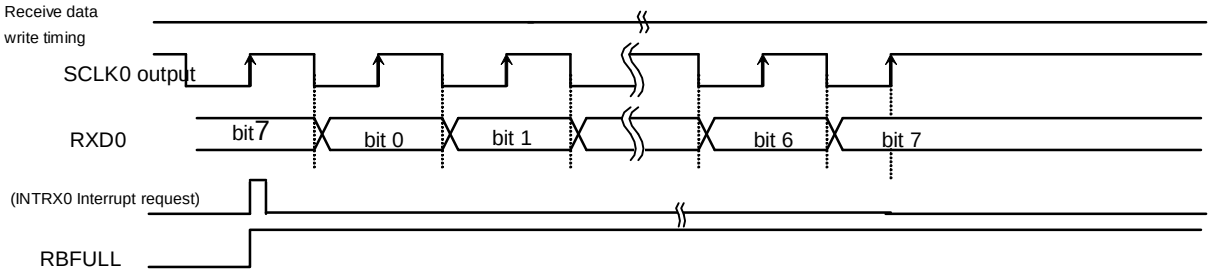
While data is in receive buffer 2, if CPU/DMAC cannot read data from receive buffer 2 before completing reception of the next 8 bits, the INTRX0 interrupt is not generated and the SCLK0 clock stops. In this state, reading data from receive buffer 2 allows data in receive buffer 1 to move to receive buffer 2 and thus the INTRX0 interrupt is generated and data reception resumes.



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



<WBUF>=“1” (if double buffering is enabled and data is read from buffer 2)



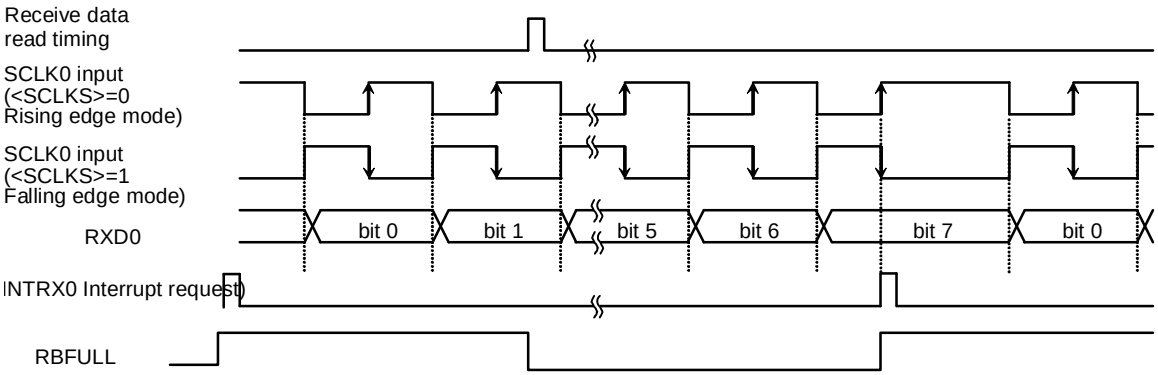
<WBUF>=“1” (if double buffering is enabled and data cannot be read from buffer 2)

Fig. 11-12 Receive Operation in the I/O Interface Mode (SCLK0 Output Mode)

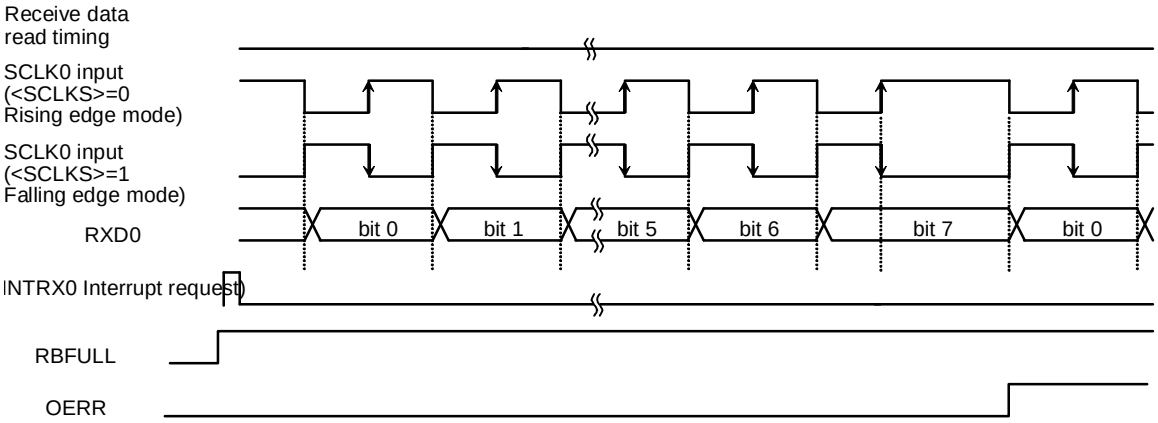
SCLK input mode

In the SCLK input mode, receiving double buffering is always enabled, the received frame can be moved to receive buffer 2 and receive buffer 1 can receive the next frame successively.

The INTRX receive interrupt is generated each time received data is moved to received buffer 2.



If data is read from buffer 2



If data cannot be read from buffer 2

Fig. 11-13 Receive Operation in the I/O Interface Mode (SCLK0 Input Mode)

(Note) To receive data, SC0MOD <RXE> must always be set to "1" (receive enable) in the SCLK output / SCLK input mode.

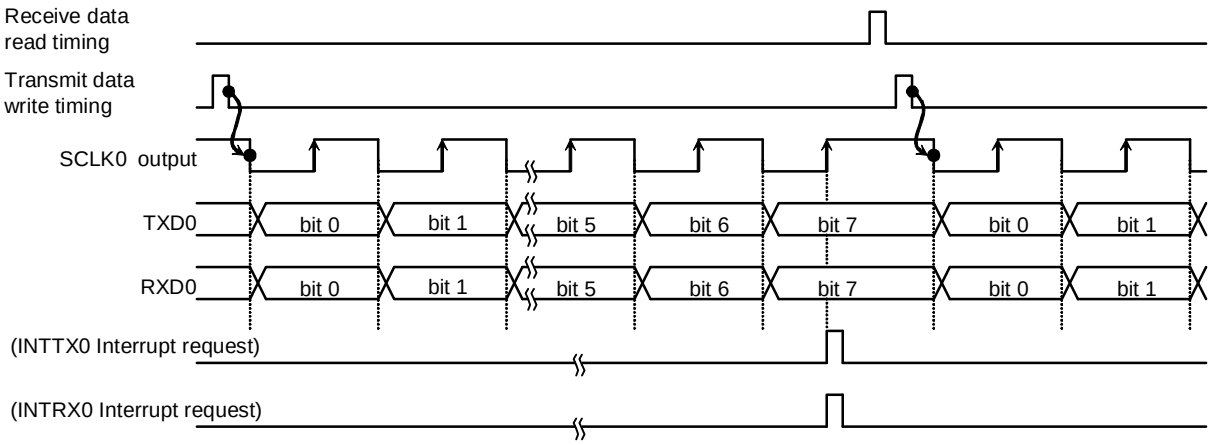
③ Transmit and receive (full-duplex)

The full-duplex mode is enabled by setting bit 6 <FDPX0> of the serial mode control register 1 (SC0MOD1) to “1”.

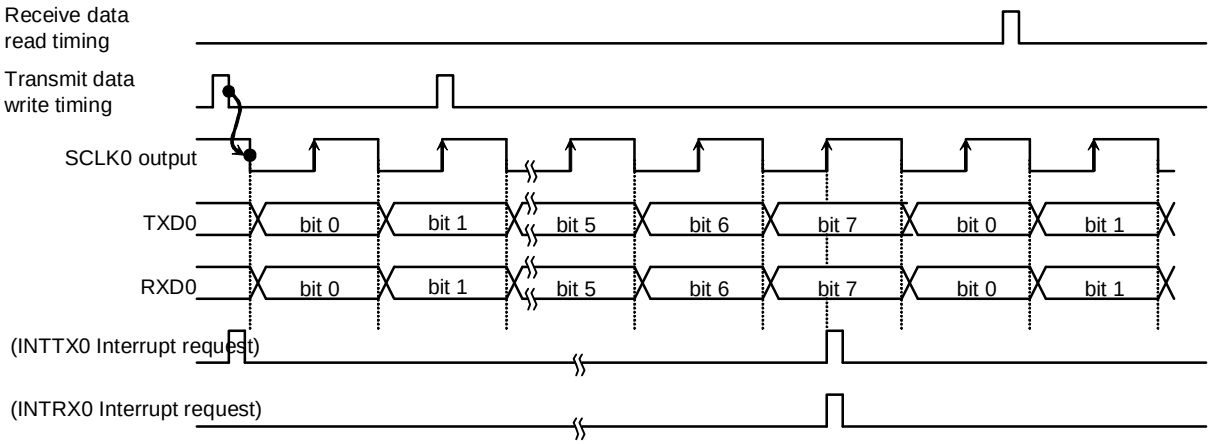
SCLK output mode

In the SCLK output mode, if SC0MOD2 <WBUF> is set to “0” and both the transmit and receive double buffers are disabled, SCLK is output when the CPU writes data to the transmit buffer. Subsequently, 8 bits of data are shifted into receive buffer 1 and the INTRX0 receive interrupt is generated. Concurrently, 8 bits of data written to the transmit buffer are output from the TXD0 pin, the INTTX0 transmit interrupt is generated when transmission of all data bits has been completed. Then, the SCLK output stops. In this, 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.

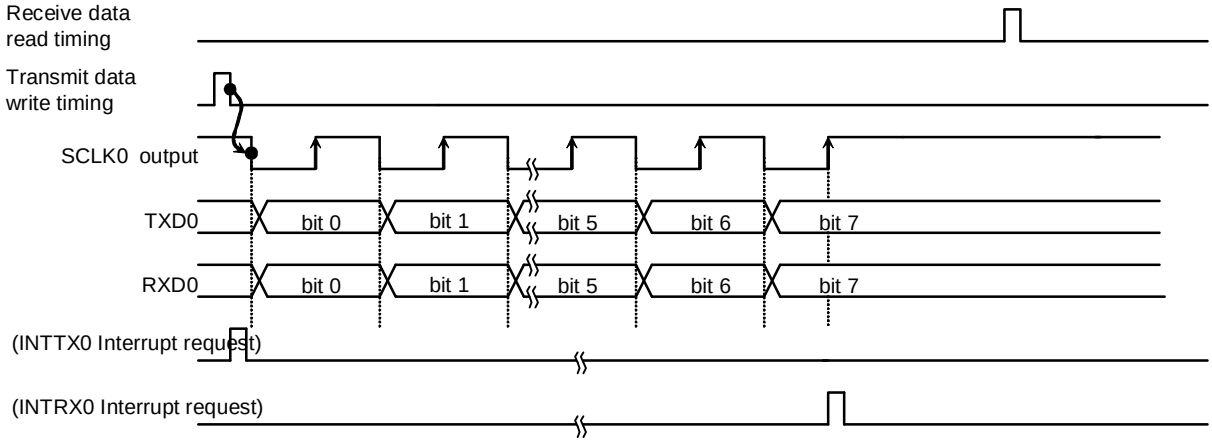
If SC0MOD2 <WBUF> = “1” and double buffering is enabled for both transmission and reception, SCLK is output when the CPU writes data to the transmit buffer. Subsequently, 8 bits of data are shifted into receive buffer 1, moved to receive buffer 2, and the INTRX0 interrupt is generated. While 8 bits of data is received, 8 bits of transmit data is output from the TXD0 pin. When all data bits are sent out, the INTTX0 interrupt is generated and the next data is moved from the Transmit Buffer 2 to Transmit Buffer 1. If Transmit Buffer 2 has no data to be moved to Transmit Buffer 1 (SC0MOD2 <TBEMP> = 1) or when receive buffer 2 is full (SC0MOD2 <RBFULL> = 1), the SCLK clock is stopped. When both conditions, receive data is read and transmit data is written, are satisfied, the SCLK output is resumed and the next round of data transmission is started.



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



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



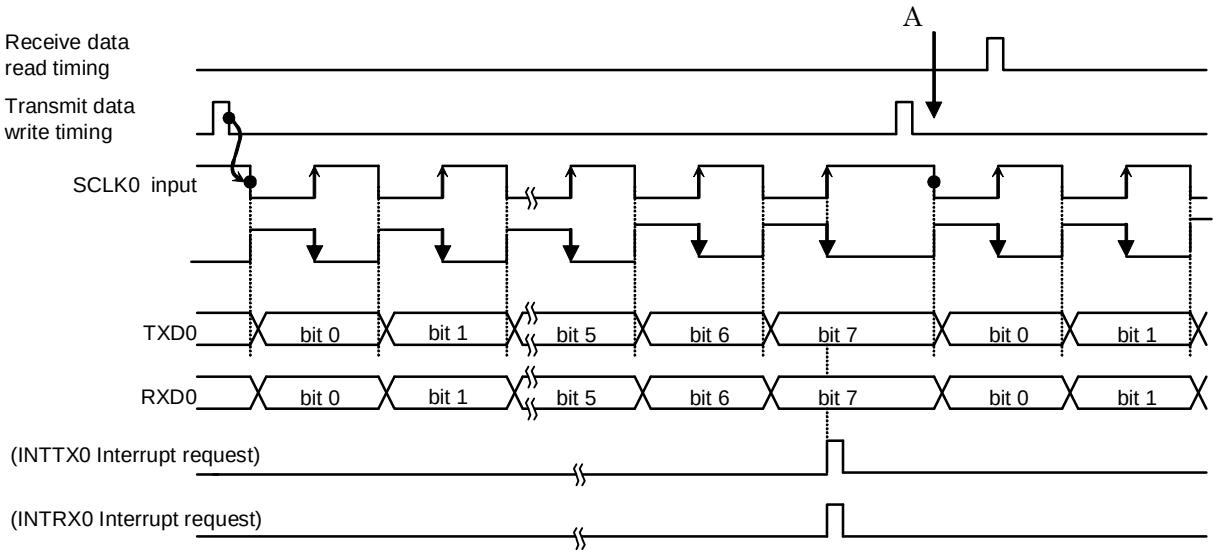
<WBUF>="1" (if double buffering is enabled)

Fig. 11-14 Transmit/Receive Operation in the I/O Interface Mode (SCLK0 Output Mode)

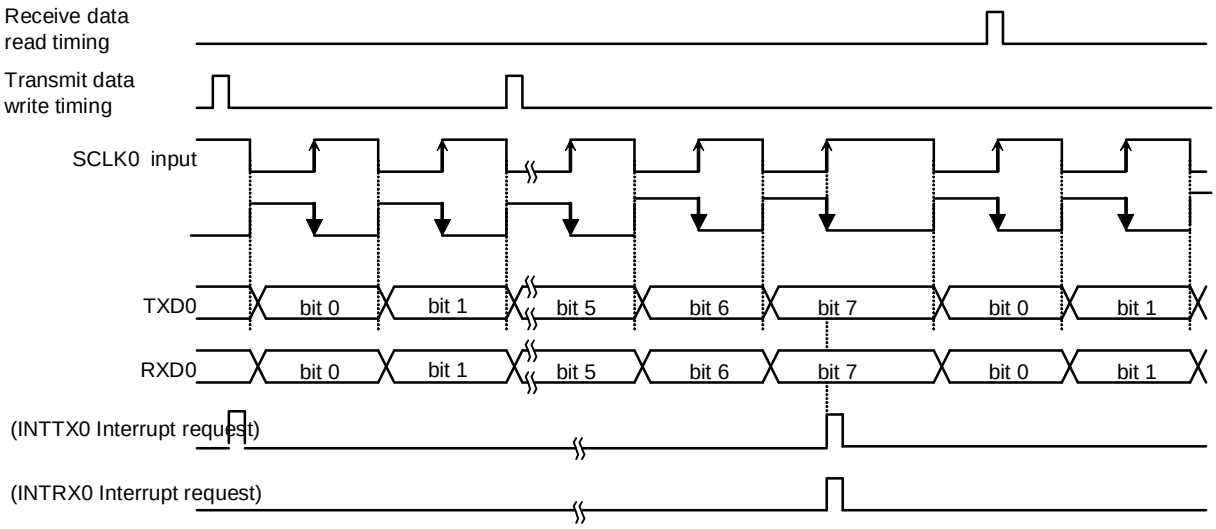
SCLK input mode

In the SCLK input mode with SC0MOD2 <WBUF> set to “0” and the transmit double buffers are disabled (double buffering is always enabled for the receive side), 8-bit data written in the transmit buffer is output from the TXD0 pin and 8 bits of data is shifted into the receive buffer when the SCLK input becomes active. The INTTX0 interrupt is generated upon completion of data transmission and the INTRX0 interrupt is generated at the instant the received data is moved from receive buffer 1 to receive buffer 2. Note that transmit data must be written into the transmit buffer before the SCLK input for the next frame (data must be written before the point A in Fig. 11-15). As double buffering is enabled for data reception, data must be read before completing reception of the next frame data.

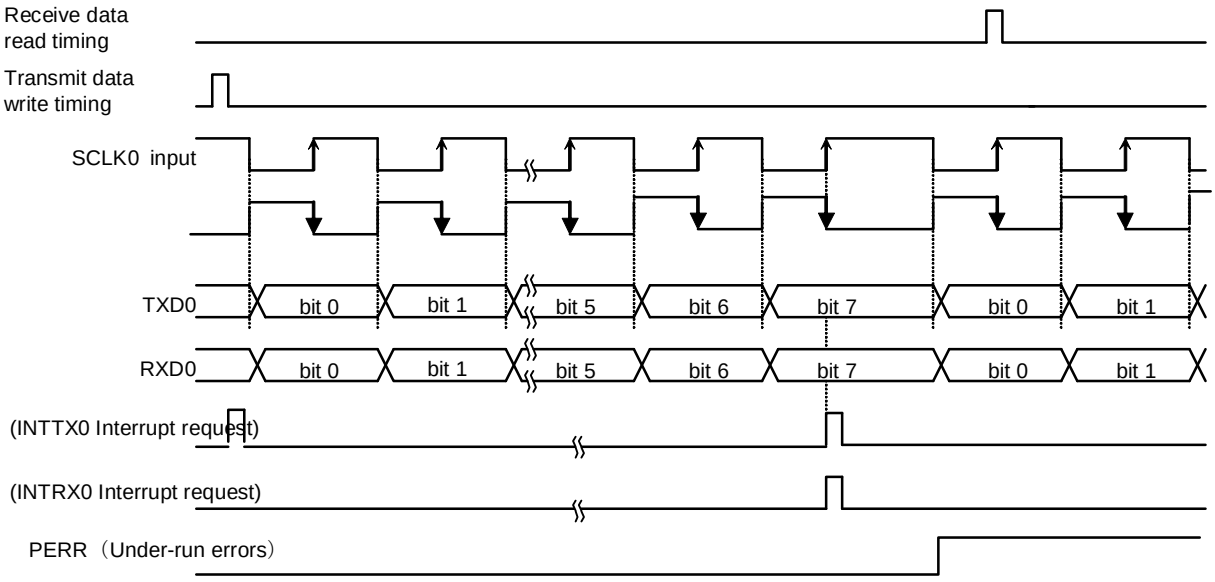
If SC0MOD2 <WBUF> = “1” and double buffering is enabled for both transmission and reception, the interrupt INTRX0 is generated at the timing Transmit Buffer 2 data is moved to Transmit Buffer 1 after completing data transmission from Transmit Buffer 1. At the same time, the 8 bits of data received is shifted to buffer 1, it is moved to receive buffer 2, and the INTRX0 interrupt is generated. Upon the SCLK input for the next frame, transmission from Transmit Buffer 1 (in which data has been moved from Transmit Buffer 2) is started while receive data is shifted into receive buffer 1 simultaneously. If data in receive buffer 2 has not been read when the last bit of the frame is received, an overrun error occurs. Similarly, if there is no data written to Transmit Buffer 2 when SCLK for the next frame is input, an under-run error occurs.



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



<WBUF>="1" (if double buffering is enabled with no errors)



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

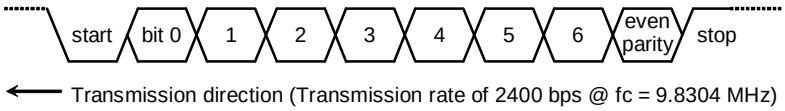
Fig. 11-15 Transmit/Receive Operation in the I/O Interface Mode (SCLK0 Input Mode)

11.5.2 Mode 1 (7-bit UART Mode)

The 7-bit UART mode can be selected by setting the serial mode control register (SC0MOD <SM1, 0>) to “01”.

In this mode, parity bits can be added to the transmit data stream; the serial mode control register (SC0CR <PE>) controls the parity enable/disable setting. When <PE> is set to “1” (enable), either even or odd parity may be selected using the SC0CR <EVEN> bit. The length of the stop bit can be specified using SC0MOD2<SBLEN>.

The following table shows the control register settings for transmitting in the following data format.



* Clocking conditions	System clock	:	high- speed (fc)
	High-speed clock gear	:	x1 (fc)
	Prescaler clock	:	$f_{periph/2}$ ($f_{periph = fsys}$)

11.5.3 Mode 2 (8-bit UART Mode)

The 8-bit UART mode can be selected by setting SC0MOD0 <SM1:0> to “10.” In this mode, parity bits can be added and parity enable/disable is controlled using SC0CR <PE>. If <PE> = “1” (enabled), either even or odd parity can be selected using SC0CR <EVEN>.

The control register settings for receiving data in the following format are as follows:

Transmission direction (transmission rate of 9600 bps @ fc = 9.8304 MHz)

* Clocking conditions	System clock	:	High-speed (fc)
	High-speed clock gear	:	x1 (fc)
	Prescaler clock	:	$f_{periph/4}$ ($f_{periph = fsys}$)

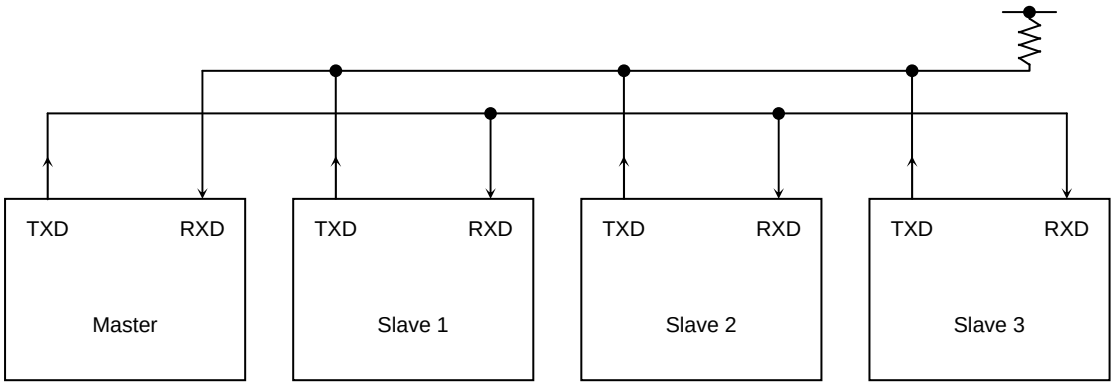
11.5.4 Mode 3 (9-bit UART)

The 9-bit UART mode can be selected by setting SC0MOD0 <SM1:0> to “11.” In this mode, parity bits must be disabled (SC0CR <PE> = “0”).

The most significant bit (9th bit) is written to bit 7 <TB8> of the serial mode control register 0 (SC0MOD0) for transmitting data. The data is stored in bit 7 <RB8> of the serial control register SC0CR. 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 SC0BUF. The stop bit length can be specified using SC0MOD2 <SBLEN>.

Wakeup function

In the 9-bit UART mode, slave controllers can be operated in the wake-up mode by setting the wake-up function control bit SC0MOD0 <WU> to “1.” In this case, the interrupt INTRX0 will be generated only when SC0CR <RB8> is set to “1”.



(Note)The TXD pin of the slave controller must be set to the open drain output mode using the ODE register.

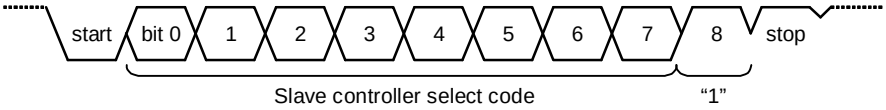
Fig. 11-16 Serial Links to Use Wake-up Function

Protocol

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

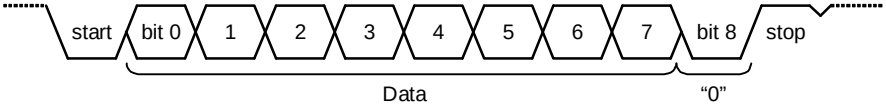
② Set SC0MOD <WU> to “1” for the slave controllers to make them ready to receive data.

③ The master controller is to transmit a single frame of data that includes the slave controller select code (8 bits). In this, the most significant bit (bit 8) <TB8> must be set to “1”.



④ Each slave controller receives the above data frame; if the code received matches with the controller's own select code, it clears the WU bit to “0”.

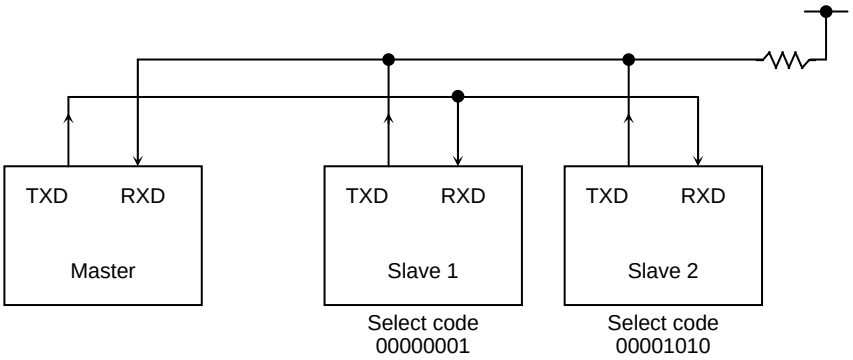
⑤ The master controller transmits data to the designated slave controller (the controller of which SC0MOD <WU> bit is cleared to “0”). In this, the most significant bit (bit 8) <TB8> must be set to “0”.



⑥ The slave controllers with the <WU> bit set to “1” ignore the receive data because the most significant bit (bit 8) <RB8> is set to “0” and thus no interrupt (INTRX0) is generated.

Also, the slave controller with the <WU> bit set to “0” can transmit data to the master controller to inform that the data has been successfully received.

An example: Using the internal clock f_{SYS} as the transfer clock, two slave controllers are serially linked as follows.



12 Serial Bus Interface (SBI)

The TMPM330 contains three Serial Bus Interface (SBI) channels, in which the following two operating modes are included:

- I^2C bus mode (with multi-master capability)
- Clock-synchronous 8-bit SIO mode

In the I^2C bus mode, the SBI is connected to external devices via SCL and SDA. In the clock-synchronous 8-bit SIO mode, the SBI is connected to external devices via SCK, SI and SO.

The following table shows the programming required to put the SBI in each operating mode.

		Pin name (PIN No.)	Port Function Register	Port Control Register	Port Input Register	Port Open Drain Output
Channel 0	I^2C bus mode	SCL : PG1 (27) SDA : PG0 (26)	PGFR1<1:0> = 11	PGCR<1:0> = 11	PGIE<1:0> = 11	PGOD<1:0> = 11
	Clock-synchronous 8-bit SIO mode	SCK : PG2 (28) SI : PG1 (27) SO : PG0 (26)	PGFR1<2:0> = 111	PGCR<2:0> = 101	PGIE<2:0> = 110	PGOD<2:0> = xxx
Channel 1	I^2C bus mode	SCL : PF5 (45) SDA : PF4 (44)	PFFR1<5:4> = 11	PFCR<5:4> = 11	PFIE<5:4> = 11	PFOD<5:4> = 11
	Clock-synchronous 8-bit SIO mode	SCK : PF6 (46) SI : PF5 (45) SO : PF4 (44)	PFFR1<6:4> = 111	PFCR<6:4> = 101	PFIE<6:4> = 110	PFOD<6:4> = xxx
Channel 2	I^2C bus mode	SCL : PG5 (16) SDA : PG4 (15)	PGFR1<5:4> = 11	PGCR<5:4> = 11	PGIE<5:4> = 11	PGOD<5:4> = 11
	Clock-synchronous 8-bit SIO mode	SCK : PG6 (17) SI : PG5 (16) SO : PG4 (15)	PGFR1<6:4> = 111	PGCR<6:4> = 101	PGIE<6:4> = 110	PGOD<6:4> = xxx

X: Don't care

12.1 Configuration

The configuration is shown in Fig. 12-1.

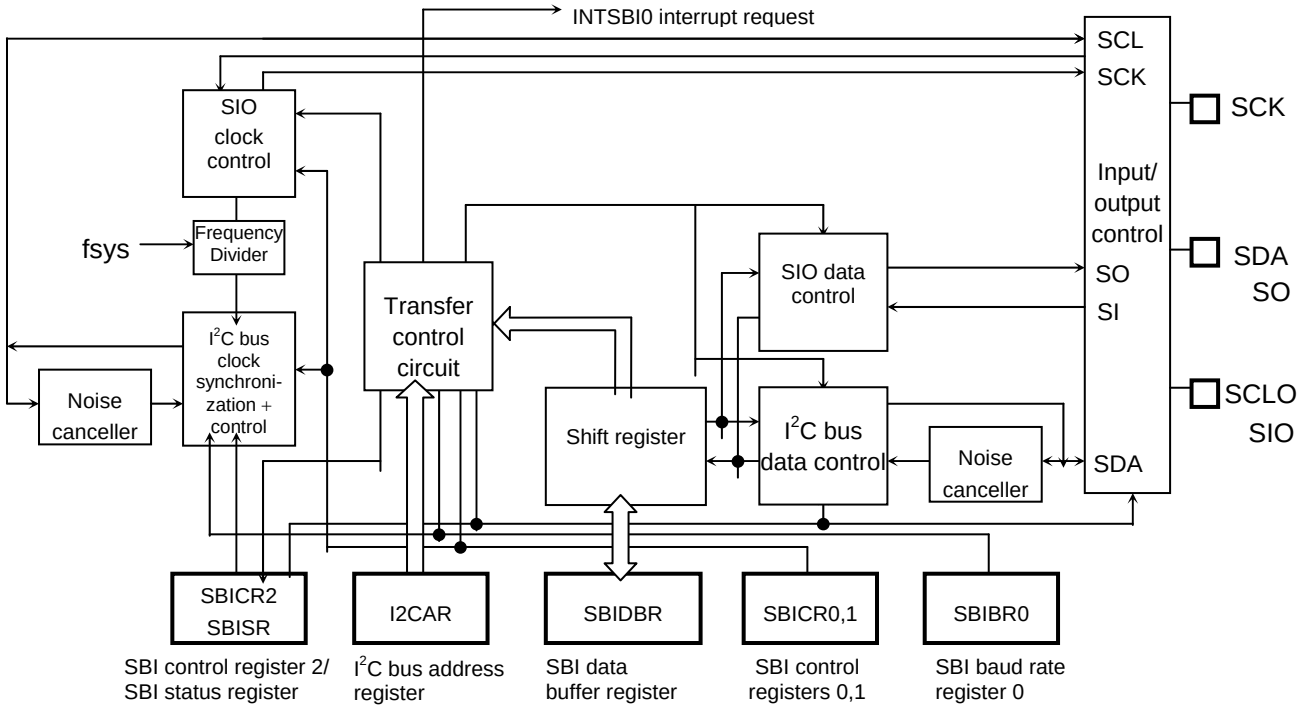


Fig. 12-1 SBI Block Diagram

12.2 Control

The following registers control the serial bus interface and provide its status information for monitoring.

- Serial bus interface control registers 0 (SBIxCR0)
- Serial bus interface control registers 1 (SBIxCR1)
- Serial bus interface control registers 2 (SBIxCR2)
- Serial bus interface buffer registers (SBIxDBR)
- I^2C bus address register (SBIxI2CAR)
- Serial bus interface status registers (SBIxSR)
- Serial bus interface baud rate registers 0 (SBIxBR0)

The functions of these registers vary, depending on the mode in which the SBI is operating. For a detailed description of the registers, refer to “12.5 Control in the I2C Bus Mode” and “0 Control in the Clock-synchronous 8-bit SIO **Mode**”.

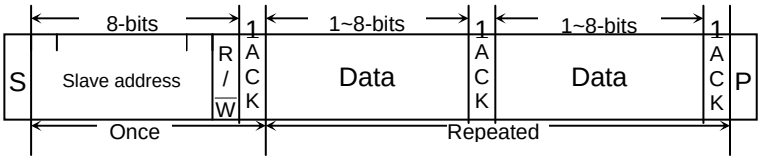
The addresses of each register are shown below.

		Channel 0		Channel 1		Channel 2	
Register name (address)	Serial bus interface control register 0	SBI0CR0	0x4002_0000	SBI1CR0	0x4002_0020	SBI2CR0	0x4002_0040
	Serial bus interface control register 1	SBI0CR1	0x4002_0004	SBI1CR1	0x4002_0024	SBI2CR1	0x4002_0044
	Serial bus interface control register 2	SBI0CR2 (reading)	0x4002_0010	SBI1CR2 (reading)	0x4002_0030	SBI2CR2 (reading)	0x4002_0050
	Serial bus interface status register	SBI0SR (writing)		SBI1SR (writing)		SBI2SR (writing)	
	Serial bus interface baud rate register 0	SBI0BR0	0x4002_0014	SBI1BR0	0x4002_0034	SBI2BR0	0x4002_0054
	Serial bus interface data buffer register	SBI0DBR	0x4002_0008	SBI1DBR	0x4002_0028	SBI2DBR	0x4002_0048
	I^2C bus address register	SBI0I2CAR	0x4002_000C	SBI1I2CAR	0x4002_002C	SBI2I2CAR	0x4002_004C

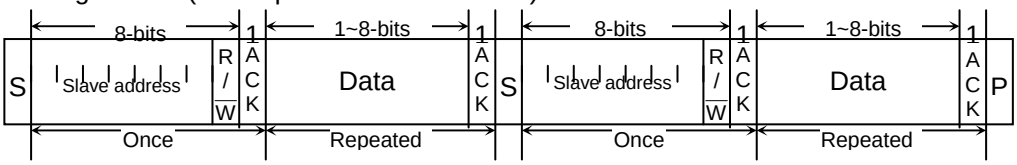
12.3 I2C Bus Mode Data Formats

Fig. 12-2 shows the data formats used in the I^2C bus mode.

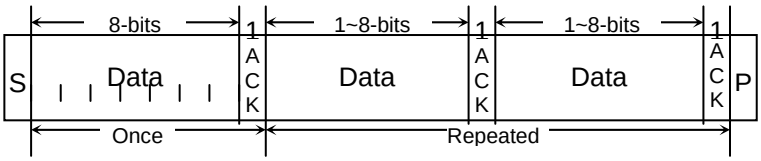
(a) Addressing format



(b) Addressing format (with repeated start condition)



(c) Free data format (master-transmitter to slave-receiver)



Note) S: Start condition
R/$\overline{W}$: Direction bit
ACK: Acknowledge bit
P: Stop condition

Fig. 12-2 I^2C Bus Mode Data Formats

12.4 Control Registers in the I2C Bus Mode

The following registers control the serial bus interface (SBI) in the I^2C bus mode and provide its status information for monitoring.

Serial bus control register 0

SBIxCR0

	7	6	5	4	3	2	1	0
bit Symbol	SBIEN							
Read/Write	R/W	R						
After reset	0	0						
Function	SBI operation 0: Disable 1: Enable	This can be read as “0.”						
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0.”							
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0.”							
	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0.”							

<SBIEN>: To use the SBI, enable the SBI operation (“1”) before setting each register in the SBI module.

Fig. 12-3 I^2C Bus Mode register

Serial bus control register 1

SBIxCR1

	7	6	5	4	3	2	1	0
Bit symbol	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0/ SWRMON
Read/Write	R/W			R/W	R	R/W		R/W
After reset	0	0	0	0	1	0	0	1
Function	Select the number of bits per transfer (Note 1)			Acknowledgment clock 0: Not generate 1: Generate	This can be read as "1."	Select internal SCL output clock frequency (Note 2) and reset monitor.		

	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R							
After reset	0							
Function	This can be read as "0."							

	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R							
After reset	0							
Function	This can be read as "0."							

	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R							
After reset	0							
Function	This can be read as "0."							

<Bit 2:0><SCK2:0>: Select internal SCL output clock frequency

On writing <SCK2:0>: Select internal SCL output clock frequency

000	n=5	196 kHz	System clock: fsys (=40 MHz)
001	n=6	149 kHz	
010	n=7	101 kHz	
011	n=8	61 kHz	Clock gear : fc/1
100	n=9	34 kHz	
101	n=10	18 kHz	Frequency = $\frac{fsys}{2^n + 72}$ [Hz]
110	n=11	9 kHz	
111		reserved	

<Bit 0>< SWRMON:0>: Software reset status monitor

On reading <SWRMON>: Software reset status monitor

0	Software reset operation is in progress.
1	Software reset operation is not in progress.

<Bit 7:5><BC2:0>: Select the number of bits per transfer

Select the number of bits per transfer

<BC2:0>	When <ACK> = 0		When <ACK> = 1	
	Number of clock cycles	Data length	Number of clock cycles	Data length
000	8	8	9	8
001	1	1	2	1
010	2	2	3	2
011	3	3	4	3
100	4	4	5	4
101	5	5	6	5
110	6	6	7	6
111	7	7	8	7

Fig. 12-4 I^2C Bus Mode register

(Note 1)	**Clear <BC2:0> to “000” before switching the operation mode to the clock-synchronous 8-bit SIO mode.**
(Note 2)	**For details on the SCL line clock frequency, refer to “12.5.3 Serial Clock.”**
(Note 3)	**After a reset, the <SCK0/SWRMON> bit is read as “1.” However, if the SIO mode is selected at the SBIxCR2 register, the initial value of the <SCK0> bit is “0.”**

Serial bus control register 2

SBIxCR2

	7	6	5	4	3	2	1	0
bit Symbol	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
Read/Write	W				W (Note 2)		W (Note 1)	
After reset	0	0	0	1	0	0	0	0
Function	Select master/slave 0: Slave 1: Master	Select transmit/ receive 0: Receive 1: Transmit	Start/stop condition generation 0: Stop condition generated 1: Start condition generated	Clear INTSBIn interrupt request 0: – 1: Clear interrupt request	Select serial bus interface operating mode (Note 2) 00: Port mode 01: SIO mode 10: I²C bus mode 11: (Reserved))		Software reset generation Write "10" followed by "01" to generate a reset.	

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as "0."							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as "0."							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as "0."							

<Bit 1:0><SWRST1:0>: Write "10" followed by "01" to generate a reset.
<Bit 3:2><SBIM1:0> : Select serial bus interface operating mode

Select serial bus interface operating mode (Note 2)

00	Port mode (disables serial bus interface output)
01	Clock –synchronous 8-bit SIO mode
10	I²C bus mode
11	(Reserved)

<Bit 4><PIN> : Clear INTSBIn interrupt request
<Bit 5><BB> : Start/stop condition generation
<Bit 6><TRX> : Select transmit/ receive
<Bit 7><MST> : Select master/slave

(Note 1) Reading this register causes it to function as the SBInSR register.

(Note 2) Ensure that the bus is free before switching the operating mode to the port mode. Ensure that the port is at the "H" level before switching the operating mode from the port mode to the I²C bus or clock-synchronous 8-bit SIO mode.

(Note 3) Ensure that serial transfer is completed before switching the mode.

Fig. 12-5 I²C Bus Mode register

Table 12-1 Base Clock Resolution

@fsys = 40 MHz

Clock gear value <GEAR1:0>	Base clock resolution
00 (fc)	fsys/2^2 (0.1 s)
01 (fc/2)	fsys/2^3 (0.2 s)
10 (fc/4)	fsys/2^4 (0.4 s)
11 (fc/8)	fsys/2^5 (0.8 s)

Serial bus interface status register

SBIxSR

	7	6	5	4	3	2	1	0
bit Symbol	MST	TRX	BB	PIN	AL	AAS	ADO	LRB
Read/Write	R							
After reset	0	0	0	1	0	0	0	0
Function	Master/ slave selection monitor 0: Slave 1: Master	Transmit/ receive selection monitor 0: Receive 1: Transmit	I^2C bus state monitor 0: Free 1: Busy	INTSBIn interrupt request monitor 0: Interrupt request generated 1: Interrupt request cleared	Arbitration lost detection 0: – 1: Detected	Slave address match detection 0: – 1: Detected	General call detection 0: – 1: Detected	Last received bit monitor 0: “0” 1: “1”

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0.”							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0.”							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0.”							

(Note) Writing to this register causes it to function as SBI0CR2.

Fig. 12-6 I^2C Bus Mode register

<Bit 0><LRB> : Last received bit monitor

<Bit 1><ADO> : General call detection

<Bit 2><AAS> : Slave address match detection

<Bit 3><AL> : Arbitration lost detection

<Bit 4><PIN> : INTSBIn interrupt request monitor

<Bit 5><BB> : I^2C bus state monitor

<Bit 6><TRX> : Transmit/ receive selection monitor

<Bit 7><MST> : Master/ slave selection monitor

Serial bus interface baud rate register 0

SBIxBR0

	7	6	5	4	3	2	1	0
bit Symbol		I2SBI0						
Read/Write	R	R/W	R					R/W
After reset	1	0	1					0
Function	This can be read as “1”.	IDLE 0: Stop 1:Operate	This can be read as “1”.					Be sure to write “0.” (Note)

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

<Bit 6><I2SBI0> : Operation at the IDLE mode

(Note) This is read as “1” at the SIO mode.

Serial bus interface data buffer register

SBIxDBR

	7	6	5	4	3	2	1	0
bit Symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (Receive)/W (Transmit)							
After reset	0							
Function	RX data/ TX data.							
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							
	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

(Note) The transmission data must be written in to the register from the MSB (bit 7).

I^{2}C bus address register

SBIxI2CAR

	7	6	5	4	3	2	1	0
bit Symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Set the slave address when the SBI acts as a slave device.							Specify address recognition mode

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

<Bit 0><ALS> : Specify address recognition mode

(Note) Please set the bit 0 <ALS> of I2C bus address register SBInI2CAR to “0”, except when you use a free data format. It operates as a free data format when setting it to “1”. Selecting the master fixes to transmission. Selecting the slave fixes to reception.

Fig. 12-7 I^{2}C Bus Mode Register

12.5 Control in the I2C Bus Mode

12.5.1 Setting the Acknowledgement Mode

Setting SBIxCR1<ACK> to “1” selects the acknowledge mode. When operating as a master, the SBI adds one clock for acknowledgment signals. As a transmitter, the SBI releases the SDA pin during this clock cycle to receive acknowledgment signals from the receiver. As a receiver, the SBI pulls the SDA pin to the “L” level during this clock cycle and generates acknowledgment signals.

By setting <ACK> to “0”, the non-acknowledgment mode is activated. When operating as a master, the SBI does not generate clock for acknowledgement signals.

12.5.2 Setting the Number of Bits per Transfer

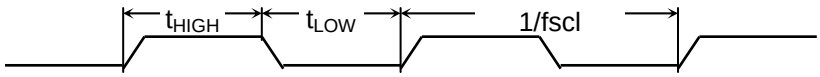
SBIxCR1 <BC2:0> specifies the number of bits of the next data to be transmitted or received.

Under the start condition, <BC2:0> is set to “000,” causing a slave address and the direction bit to be transferred in a packet of eight bits. At other times, <BC2:0> keeps a previously programmed value.

12.5.3 Serial Clock

① Clock source

SBIxCR1 <SCK2:0> specifies the maximum frequency of the serial clock to be output from the SCL pin in the master mode.



$t_{LOW} = 2^{n-1}/fsys + 58/ fsys$

$t_{HIGH} = 2^{n-1}/ fsys + 14/ fsys$

$fscl = 1/(t_{Low} + t_{HIGH})$

$= \dfrac{fsys}{2^n + 72}$

SBIxCR1 <SCK2:0>	n
000	5
001	6
010	7
011	8
100	9
101	10
110	11

Fig. 12-8 Clock Source

(Note)	**The highest speeds in the standard and high-speed modes are specified to 100KHz and 400KHz respectively following the communications standards. Note that the internal SCL clock frequency is determined by the fsys used and the calculation formula shown above.**

② Clock Synchronization

The I^2C bus is driven by using the wired-AND connection due to its pin structure. The first master that pulls its clock line to the “L” level overrides other masters producing the “H” level on their clock lines. This must be detected and responded by the masters producing the “H” level.

Clock synchronization assures correct data transfer on a bus that has two or more master.

For example, the clock synchronization procedure for a bus with two masters is shown below.

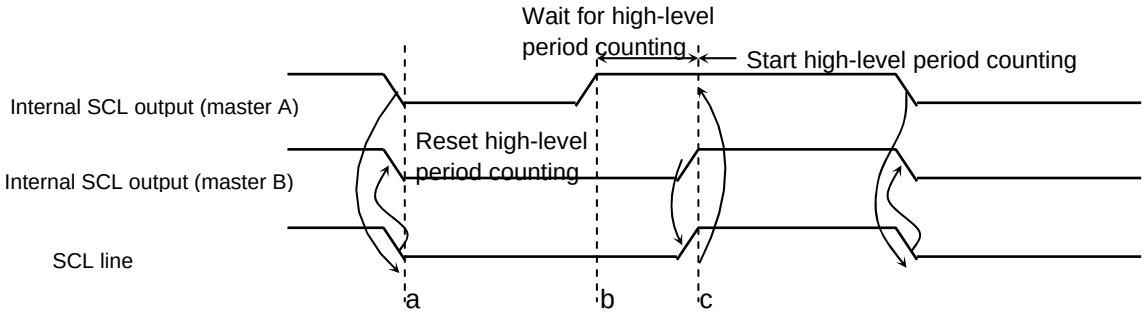


Fig. 12-9 Example of Clock Synchronization

At the point a, Master A pulls its internal SCL output to the “L” level, bringing the SCL bus line to the “L” level. Master B detects this transition, resets its “H” level period counter, and pulls its internal SCL output level to the “L” level.

Master A completes counting of its “L” level period at the point b, and brings its internal SCL output to the “H” level. However, Master B still keeps the SCL bus line at the “L” level, and Master A stops counting of its “H” level period counting. After Master A detects that Master B brings its internal SCL output to the “H” level and brings the SCL bus line to the “H” level at the point c, it starts counting of its “H” level period.

This way, the clock on the bus is determined by the master with the shortest “H” level period and the master with the longest “L” level period among those connected to the bus.

12.5.4 Slave Addressing and Address Recognition Mode

When the SBI is configured to operate as a slave device, the slave address <SA6:0> and <ALS> must be set at SBIxI2CAR. Setting <ALS> to “0” selects the address recognition mode.

12.5.5 Configuring the SBI as a Master or a Slave

Setting SBIxCR2<MST> to “1” configures the SBI to operate as a master device.

Setting <MST> to “0” configures the SBI as a slave device. <MST> is cleared to “0” by the hardware when it detects the stop condition on the bus or the arbitration lost.

12.5.6 Configuring the SBI as a Transmitter or a Receiver

Setting SBIxCR2 <TRX> to “1” configures the SBI as a transmitter. Setting <TRX> to “0” configures the SBI as a receiver.

At the slave mode, the SBI receives the direction bit ($R/\overline{W}$) from the master device on the following occasions:

- when data is transmitted in the addressing format
- when the received slave address matches the value specified at I2CCR
- when a general-call address is received; i.e., the eight bits following the start condition are all zeros

If the value of the direction bit ($R/\overline{W}$) is “1,” <TRX> is set to “1” by the hardware. If the bit is “0,” <TRX> is set to “0”.

As a master device, the SBI receives acknowledgement from a slave device. If the direction bit of “1” is transmitted, <TRX> is set to “0” by the hardware. If the direction bit is “0,” <TRX> changes to “1.” If the SBI does not receive acknowledgement, <TRX> retains the previous value.

<TRX> is cleared to “0” by the hardware when it detects the stop condition on the bus or the arbitration lost.

12.5.7 Generating Start and Stop Conditions

When SBIxSR<BB> is “0,” writing “1” to SBIxCR2 <MST, TRX, BB, PIN> causes the SBI to generate the start condition on the bus and output 8-bit data. <ACK> must be set to “1” in advance.

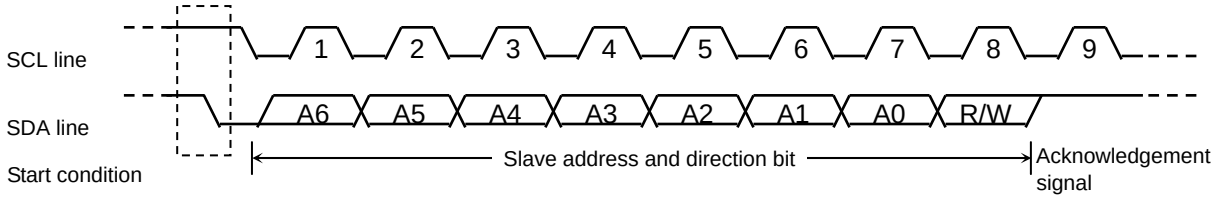


Fig. 12-10 Generating the Start Condition and a Slave Address

When <BB> is “1,” writing “1” to <MST, TRX, PIN> and “0” to <BB> causes the SBI to start a sequence for generating the stop condition on the bus. The contents of <MST, TRX, BB, PIN> should not be altered until the stop condition appears on the bus.

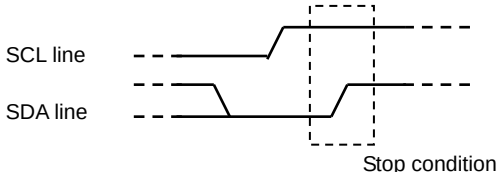


Fig. 12-11 Generating the Stop Condition

SBIxSR<BB> can be read to check the bus state. <BB> is set to “1” when the start condition is detected on the bus (the bus is busy), and set to “0” when the stop condition is detected (the bus is free).

12.5.8 Interrupt Service Request and Release

When a serial bus interface interrupt request (INTSBI0) is generated, SBIxCR2 <PIN> is cleared to “0.” While <PIN> is “0,” the SBI pulls the SCL line to the “L” level.

After transmission or reception of one data word, <PIN> is cleared to “0.” It is set to “1” when data is written to or read from SBIxDBR. It takes a period of t_{LOW} for the SCL line to be released after <PIN> is set to “1.”

In the address recognition mode (<ALS> = “0”), <PIN> is cleared to “0” when the received slave address matches the value specified at SBIxI2CAR or when a general-call address is received; i.e., the eight bits following the start condition are all zeros. When the program writes “1” to SBIxCR2<PIN>, it is set to “1.” However, writing “0” does clear this bit to “0”.

12.5.9 Serial Bus Interface Operating Modes

SBIxCR2 <SBIM1:0> selects an operating mode of the serial bus interface. <SBIM1:0> must be set to “10” to configure the SBI for the I^2C bus mode. Make sure that the bus is free before switching the operating mode to the port mode.

12.5.10 Lost-arbitration Detection Monitor

The I^2C bus has the multi-master capability (there are two or more masters on a bus), and requires the bus arbitration procedure to ensure correct data transfer.

A master that attempts to generate the start condition while the bus is busy loses bus arbitration, with no start condition occurring on the SDA and SCL lines. The I^2C-bus arbitration takes place on the SDA line.

The arbitration procedure for two masters on a bus is shown below. Up until the point a, Master A and Master B output the same data. At the point a, Master A outputs the “L” level and Master B outputs the “H” level. Then Master A pulls the SDA bus line to the “L” level because the line has the wired-AND connection. When the SCL line goes high at the point b, the slave device reads the SDA line data, i.e., data transmitted by Master A. At this time, data transmitted by Master B becomes invalid. This condition of Master B is called “Lost Arbitration”. Master B releases its SDA pin, so that it does not affect the data transfer initiated by another master. If two or more masters have transmitted exactly the same first data word, the arbitration procedure continues with the second data word.

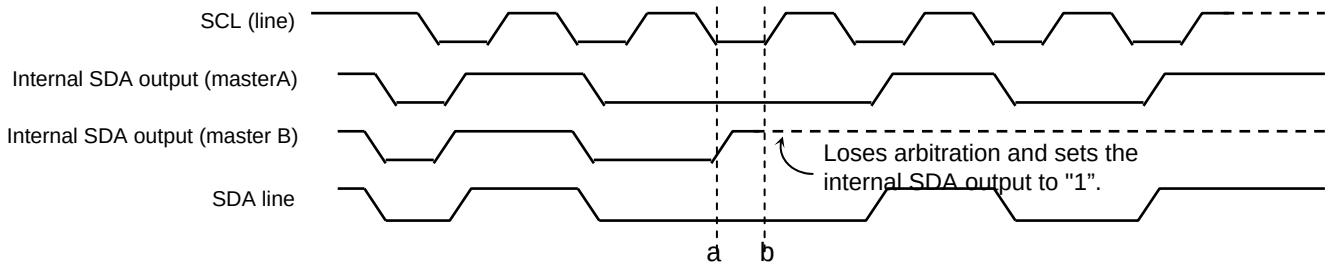


Fig. 12-12 Lost Arbitration

A master compares the SDA bus line level and the internal SDA output level at the rising of the SCL line. If there is a difference between these two values, Arbitration Lost occurs and SBIxSR <AL> is set to “1”.

When <AL> is set to “1,” SBIxSR <MST, TRX> are cleared to “0,” causing the SBI to operate as a slave receiver. <AL> is cleared to “0” when data is written to or read from SBIxDBR or data is written to SBIxCR2.

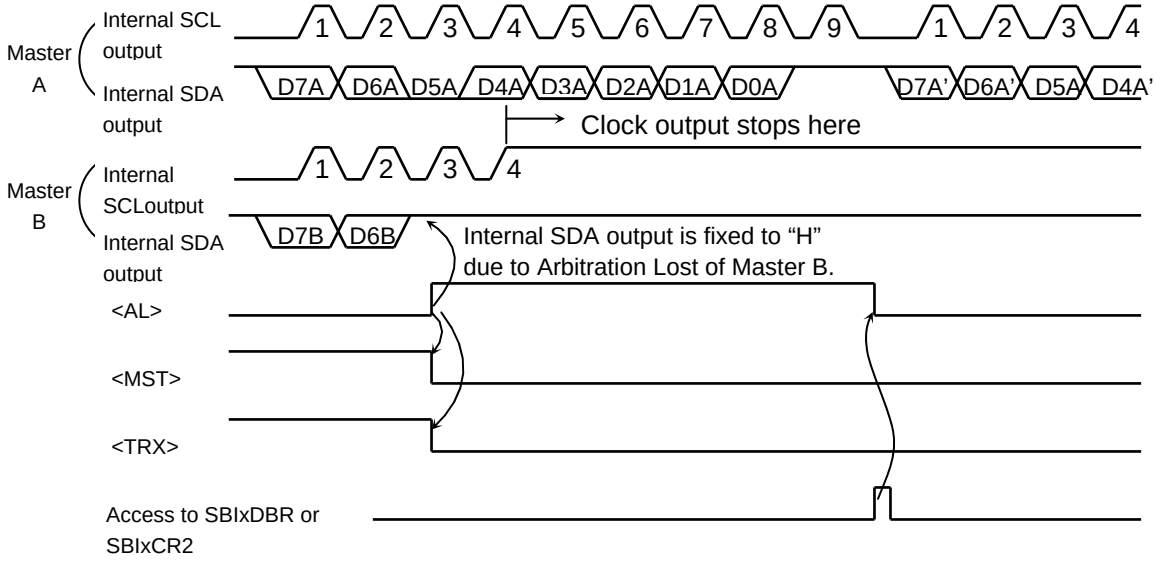


Fig. 12-13 Example of Master B Lost Arbitration (D7A = D7B, D6A = D6B)

12.5.11 Slave Address Match Detection Monitor

When the SBI operates as a slave device in the address recognition mode (SBIxI2CAR <ALS> = ”0”), SBIxSR <AAS> is set to “1” on receiving the general-call address or the slave address that matches the value specified at SBIxI2CAR. When <ALS> is “1,” <AAS> is set to “1” when the first data word has been received. <AAS> is cleared to “0” when data is written to or read from SBIxDBR.

12.5.12 General-call Detection Monitor

When the SBI operates as a slave device, SBIxSR <AD0> is set to “1” when it receives the general-call address; i.e., the eight bits following the start condition are all zeros. <AD0> is cleared to “0” when the start or stop condition is detected on the bus.

12.5.13 Last Received Bit Monitor

SBIxSR <LRB> is set to the SDA line value that was read at the rising of the SCL line. In the acknowledgment mode, reading SBIxSR <LRB> immediately after generation of the INTSBIx interrupt request causes ACK signal to be read.

12.5.14 Software Reset

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

Writing “10” followed by “01” to SBIxCR2 <SWRST1:0> generates a reset signal that initializes the serial bus interface circuit. After a reset, all control registers and status flags are initialized to their reset values. When the serial bus interface is initialized, <SWRST> is automatically cleared to “0”.

(Note)	**A software reset causes the SBI operating mode to switch from the I^2C mode to the port mode.**

12.5.15 Serial Bus Interface Data Buffer Register (SBIxDBR)

Reading or writing SBIxDBR initiates reading received data or writing transmitted data. When the SBI is acting as a master, setting a slave address and a direction bit to this register generates the start condition.

12.5.16 I2C Bus Address Register (SBIxI2CAR)

When the SBI is configured as a slave device, the SBIxI2CAR<SA6:0> bit is used to specify a slave address. If I2CAR <ALS> is set to “0,” the SBI recognizes a slave address transmitted by the master device and receives data in the addressing format. If <ALS> is set to “1,” the SBI does not recognize a slave address and receives data in the free data format.

12.5.17 IDLE Setting Register (SBIxBR0)

The SBIxBR0<I2SBI> register determines if the SBI operates or not when it enters the IDLE mode. This register must be programmed before executing an instruction to switch to the standby mode.

12.6 Data Transfer Procedure in the I2C Bus Mode

12.6.1 Device Initialization

First, program SBIxCR1<ACK, SCK2:0> by writing “0” to bits 7 to 5 and bit 3 in SBIxCR1.

Next, program SBIxI2CAR by specifying a slave address at <SA6:0> and an address recognition mode at <ALS>. (<ALS> must be set to”0” when using the addressing format).

Then program SBIxCR2 to initially configure the SBI in the slave receiver mode by writing “0” to <MST, TRX, BB> , “1” to <PIN> , “10” to <SBIM1:0> and “0” to bits 1 and 0.

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	0	0	X	0	X	X	X	Specifies ACK and SCL clock.
SBIxI2CAR	←	X	X	X	X	X	X	X	X	Specifies a slave address and an address recognition mode.
SBIxCR2	←	0	0	0	1	1	0	0	0	Configures the SBI as a slave receiver.

(Note) X: Don’t care

12.6.2 Generating the Start Condition and a Slave Address

① Master mode

In the master mode, the following steps are required to generate the start condition and a slave address.

First, ensure that the bus is free (<BB> = “0”). Then, write “1” to SBIxCR1 <ACK> to select the acknowledgment mode. Write to SBIxDBR a slave address and a direction bit to be transmitted.

When <BB> = “0,” writing “1111” to SBIxCR2 <MST, TRX, BB, PIN> generates the start condition on the bus. Following the start condition, the SBI generates nine clocks from the SCL pin. The SBI outputs the slave address and the direction bit specified at SBIxDBR with the first eight clocks, and releases the SDA line in the ninth clock to receive an acknowledgment signal from the slave device.

The INTSBIx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to ”0.” In the master mode, the SBI holds the SCL line at the “L” level while <PIN> is “0.” <TRX> changes its value according to the transmitted direction bit at generation of the INTSBIx interrupt request, provided that an acknowledgment signal has been returned from the slave device.

Settings in main routine

```
                    7 6 5 4 3 2 1 0
→ Reg.        ← SBISR
  Reg.        ← Reg. e 0x20
  if Reg.     ≠ 0x00                    Ensures that the bus is free.
  Then
  SBIxCR1     ← X X X 1 0 X X X         Selects the acknowledgement mode.
  SBIxDR1     ← X X X X X X X X         Specifies the desired slave address and direction.
  SBIxCR2     ← 1 1 1 1 1 0 0 0         Generates the start condition.
```

Example of INTSBI0 interrupt routine

Clears the interrupt request.
Processing
End of interrupt

② Slave mode

In the slave mode, the SBI receives the start condition and a slave address.

After receiving the start condition from the master device, the SBI receives a slave address and a direction bit from the master device during the first eight clocks on the SCL line. If the received address matches its slave address specified at SBIxI2CAR or is equal to the general-call address, the SBI pulls the SDA line to the “L” level during the ninth clock and outputs an acknowledgment signal.

The INTSBIx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to “0.” In the slave mode, the SBI holds the SCL line at the “L” level while <PIN> is “0”.

(Note) The user can only use a DMA transfer:
- **when there is only one master and only one slave and**
- **continuous transmission or reception is possible.**

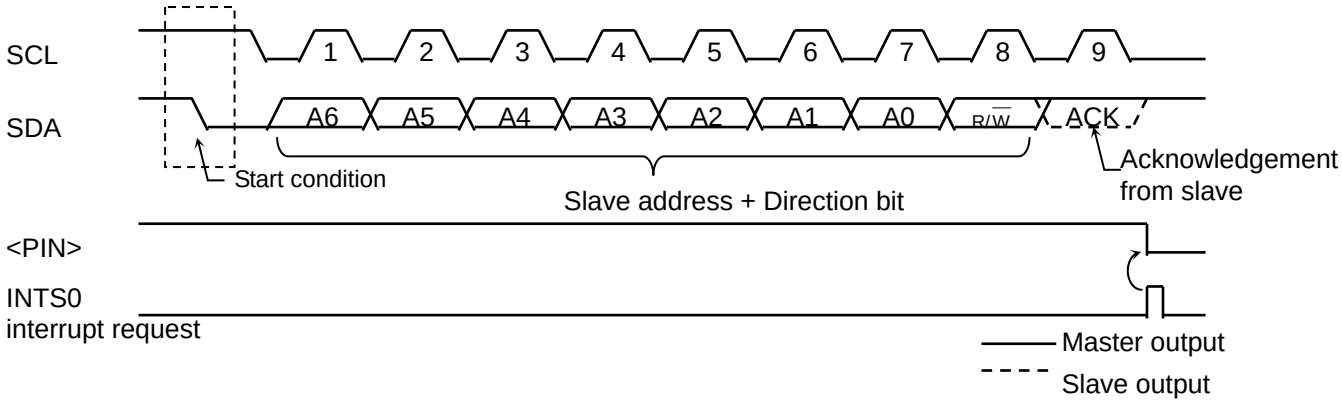


Fig. 12-14 Generation of the Start Condition and a Slave Address

12.6.3 Transferring a Data Word

At the end of a data word transfer, the INTSBIx interrupt is generated to test <MST> to determine whether the SBI is in the master or slave mode.

① Master mode (<MST> = “1”)

Test <TRX> to determine whether the SBI is configured as a transmitter or a receiver.

Transmitter mode (<TRX> = “1”)

Test <LRB>. If <LRB> is “1,” that means the receiver requires no further data. The master then generates the stop condition as described later to stop transmission.

If <LRB> is “0,” that means the receiver requires further data. If the next data to be transmitted has eight bits, the data is written into SBIxDBR. If the data has different length, <BC2:0> and <ACK> are programmed and the transmit data is written into SBIxDBR. Writing the data makes <PIN> to”1,” causing the SCL pin to generate a serial clock for transferring a next data word, and the SDA pin to transfer the data word. After the transfer is completed, the INTSBIx interrupt request is generated, <PIN> is set to “0,” and the SCL pin is pulled to the “L” level. To transmit more data words, test <LRB> again and repeat the above procedure.

INTSBIx interrupt

if MST = 0
Then go to the slave-mode processing
if TRX = 0
Then go to the receiver-mode processing
if LRB = 0
Then go to processing for generating the stop condition

SBIxCR1 ← X X X X 0 X X X	Specifies the number of bits to be transmitted and specify whether ACK is required.
SBIxDBR ← X X X X X X X X	Writes the transmit data.

End of interrupt processing

(Note) X: Don’t care

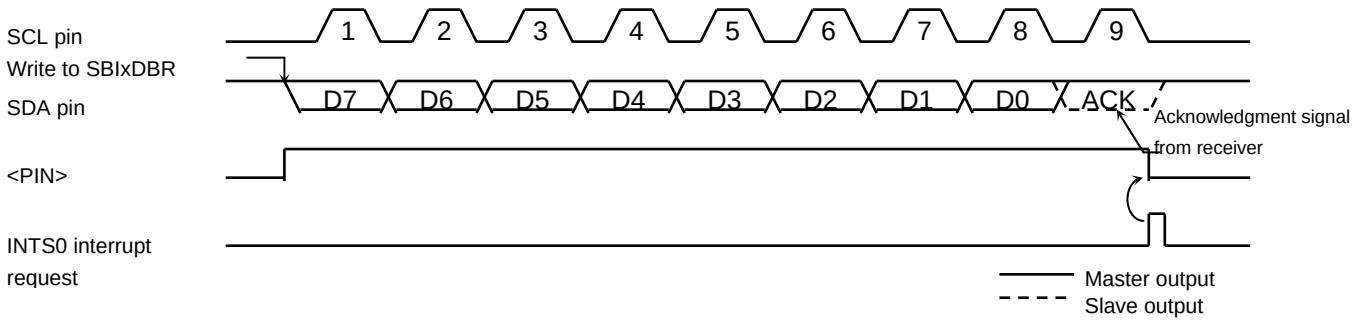


Fig. 12-15 <BC2:0> = “000” and <ACK> = “1” (Transmitter Mode)

Receiver mode (<TRX> = “0”)

If the next data to be transmitted has eight bits, the transmit data is written into SBIxDBR. If the data has different length, <BC2:0> and <ACK> are programmed and the received data is read from SBIxDBR to release the SCL line. (The data read immediately after transmission of a slave address is undefined.) On reading the data, <PIN> is set to “1,” and the serial clock is output to the SCL pin to transfer the next data word. In the last bit, when the acknowledgment signal becomes the “L” level, “0” is output to the SDA pin.

After that, the INTSBIx interrupt request is generated, and <PIN> is cleared to “0,” pulling the SCL pin to the “L” level. Each time the received data is read from SBIxDBR, one-word transfer clock and an acknowledgement signal are output.

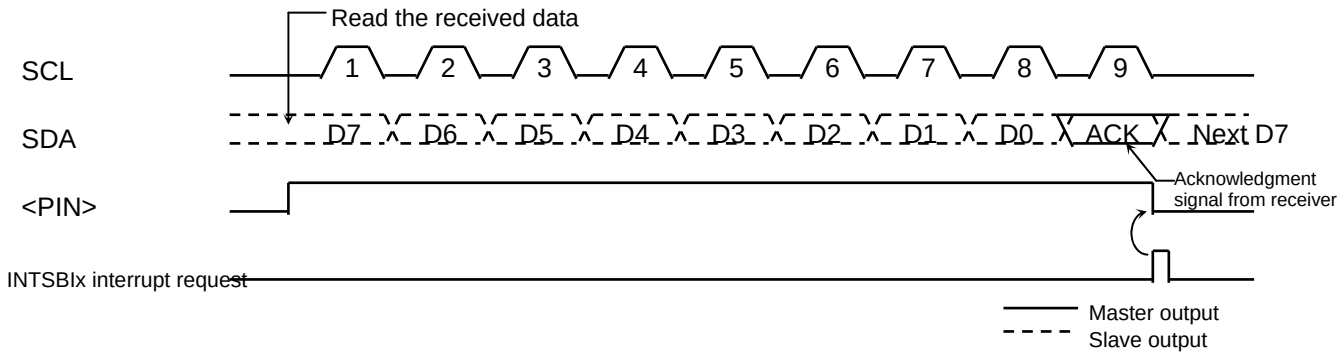


Fig. 12-16 <BC2:0> = “000” and <ACK> = “1” (Receiver Mode)

To terminate the data transmission from the transmitter, <ACK> must be set to “0” immediately before reading the data word second to last. This disables generation of an acknowledgment clock for the last data word. When the transfer is completed, an interrupt request is generated. After the interrupt processing, <BC2:0> must be set to “001” and the data must be read so that a clock is generated for 1-bit transfer. At this time, the master receiver holds the SDA bus line at the “H” level, which signals the end of transfer to the transmitter as an acknowledgment signal.

In the interrupt processing for terminating the reception of 1-bit data, the stop condition is generated to terminate the data transfer.

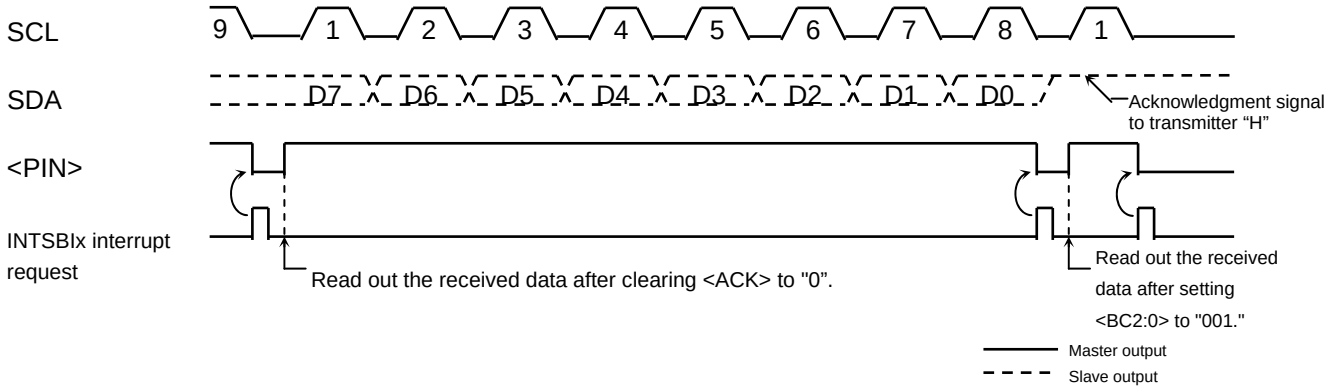


Fig. 12-17 Terminating Data Transmission in the Master Receiver Mode

Example: When receiving N data word

INTSBIx interrupt (after data transmission)

		7 6 5 4 3 2 1 0	
SBIxCR1	←	X X X X 0 X X X	Sets the number of bits of data to be received and specify whether ACK is required.
Reg.	←	SBI0DBR	Reads dummy data.
End of interrupt			

INTSBIx interrupt (first to (N-2)th data reception)

		7 6 5 4 3 2 1 0	
Reg.	←	SBIxDBR	Reads the first to (N-2)th data words.
End of interrupt			

INTSBIx interrupt ((N-1)th data reception)

		7 6 5 4 3 2 1 0	
SBIxCR1	←	X X X 0 0 X X X	Disables generation of acknowledgement clock.
Reg.	←	SBIxDBR	Reads the (N-1)th data word.
End of interrupt			

INTSBIx interrupt (Nth data reception)

		7 6 5 4 3 2 1 0	
SBIxCR1	←	0 0 1 0 0 X X X	Disables generation of acknowledgement clock.
Reg.	←	SBIxDBR	Reads the Nth data word.
End of interrupt			

INTSBIx interrupt (after completing data reception)

Processing to generate the stop condition.	Terminates the data transmission.
End of interrupt	

(Note) X: Don’t care

② Slave mode (<MST> = “0”)

In the slave mode, the SBI generates the INTSBIx interrupt request on four occasions: 1) when the SBI has received any slave address from the master, 2) when the SBI has received a general-call address, 3) when the received slave address matches its own address, and 4) when a data transfer has been completed in response to a general-call. Also, if the SBI detects Arbitration Lost in the master mode, it switches to the slave mode. Upon the completion of data word transfer in which Arbitration Lost is detected, the INTSBIx interrupt request is generated, <PIN> is cleared to “0,” and the SCL pin is pulled to the “L” level. When data is written to or read from SBIxDBR or when <PIN> is set to “1,” the SCL pin is released after a period of t_{LOW}.

In the slave mode, the normal slave mode processing or the processing as a result of Arbitration Lost is carried out.

SBIxSR <AL>, <TRX>, <AAS> and <AD0> are tested to determine the processing required.

Table 12-2 shows the slave mode states and required processing.

Example: When the received slave address matches the SBI's own address and the direction bit is “1” in the slave receiver mode.

INTSBIx interrupt

```
if TRX = 0
Then go to other processing
if AL = 1
Then go to other processing
if AAS = 0
Then go to other processing
SBIxCR1  ← X X X 1 0 X X X        Sets the number of bits to be transmitted.
SBIxDBR  ← X X X X 0 X X X        Sets the transmit data.

(Note) X: Don’t care
```

Table 12-2 Processing in Slave Mode

<TRX>	<AL>	<AAS>	<AD0>	State	Processing
1	1	1	0	Arbitration Lost is detected while the slave address was being transmitted and the SBI received a slave address with the direction bit “1” transmitted by another master.	Set the number of bits in a data word to <BC2:0> and write the transmit data into SBIxDBR.
	0	1	0	In the slave receiver mode, the SBI received a slave address with the direction bit “1” transmitted by the master.	
		0	0	In the slave transmitter mode, the SBI has completed a transmission of one data word.	Test LRB. If it has been set to “1,” that means the receiver does not require further data. Set <PIN> to 1 and reset <TRX> to 0 to release the bus. If <LRB> has been reset to “0,” that means the receiver requires further data. Set the number of bits in the data word to <BC2:0> and write the transmit data to the SBIxDBR.
0	1	1	1/0	Arbitration Lost is detected while a slave address is being transmitted, and the SBI receives either a slave address with the direction bit “0” or a general-call address transmitted by another master.	Read the SBIxDBR (a dummy read) to set <PIN> to 1, or write “1” to <PIN>.
		0	0	Arbitration Lost is detected while a slave address or a data word is being transmitted, and the transfer is terminated.	
	0	1	1/0	In the slave receiver mode, the SBI received either a slave address with the direction bit “0” or a general-call address transmitted by the master.	
		0	1/0	In the slave receiver mode, the SBI has completed a reception of a data word.	Set the number of bits in the data word to <BC2:0> and read the received data from SBIxDBR.

12.6.4 Generating the Stop Condition

When SBIxSR <BB> is “1,” writing “1” to SBIxCR2 <MST, TRX, PIN> and “0” to <BB> causes the SBI to start a sequence for generating the stop condition on the bus. Do not alter the contents of <MST, TRX, BB, PIN> until the stop condition appears on the bus.

If another device is holding down the SCL bus line, the SBI waits until the SCL line is released. After that, the SDA pin goes high, causing the stop condition to be generated.

		7	6	5	4	3	2	1	0	
SBIxCR2	←	1	1	0	1	1	0	0	0	Generates the stop condition.

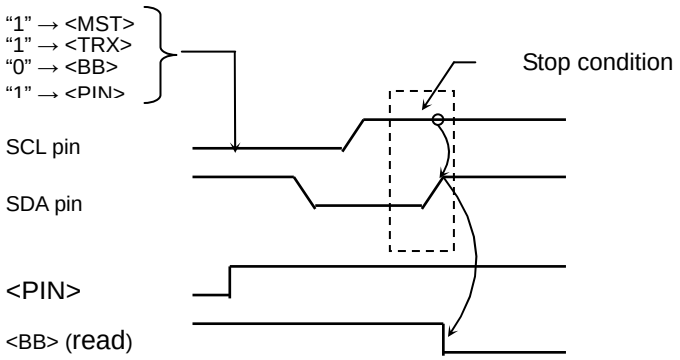


Fig. 12-18 Generating the Stop Condition

12.6.5 Restart Procedure

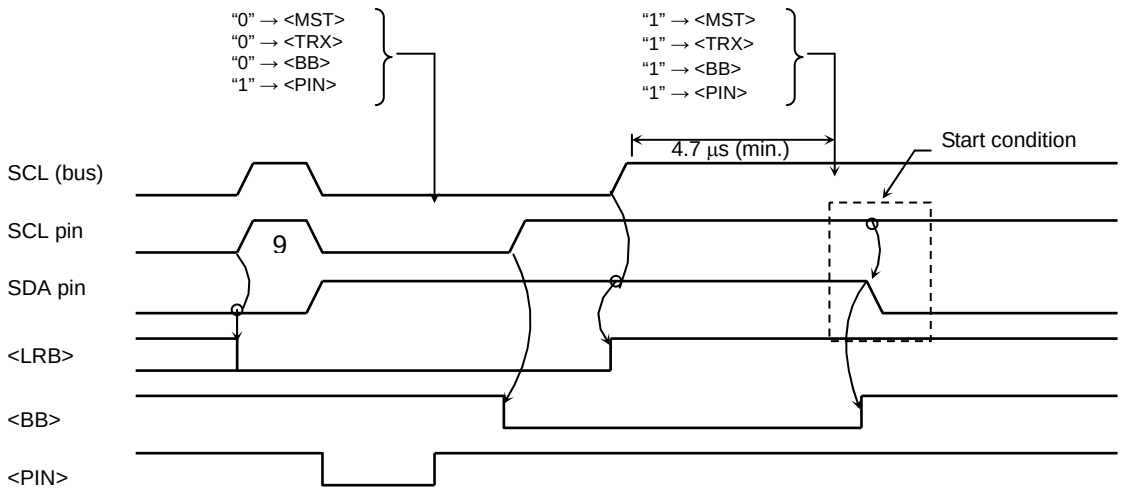
Restart is used when a master device changes the data transfer direction without terminating the transfer to a slave device. The procedure of generating a restart in the master mode is described below.

First, set SBIxCR2 <MST, TRX, BB> to “0” and write “1” to <PIN> to release the bus. At this time, the SDA pin is held at the “H” level and the SCL pin is released. Because no stop condition is generated on the bus, other devices recognize that the bus is busy. Then, test SBIxSR <BB> and wait until it becomes “0” to ensure that the SCL pin is released. Next, test <LRB> and wait until it becomes “1” to ensure that no other device is pulling the SCL bus line to the “L” level. Once the bus is determined to be free this way, use the above-mentioned steps 12.6.2 to generate the start condition.

To satisfy the setup time of restart, at least 4.7-μs wait period (in the standard mode) must be created by the software after the bus is determined to be free.

	7 6 5 4 3 2 1 0	
SBIxCR2	← 0 0 0 1 1 0 0 0	Releases the bus.
if SBIxSR<BB> ≠ 0		Checks that the SCL pin is released.
Then		
if SBIxSR<LRB> ≠1		Checks that no other device is pulling the SCL pin to the “L” level.
Then		
4.7 μs Wait		
SBIxCR1	← X X X 1 0 X X X	Selects the acknowledgment mode.
SBIxDBR	← X X X X X X X X	Sets the desired slave address and direction.
SBIxCR2	← 1 1 1 1 1 0 0 0	Generates the start condition.

(Note) X: Don’t care



(Note) Do not write <MST> to “0” when it is “0.” (Restart cannot be initiated.)

Fig. 12-19 Timing Chart of Generating a Restart

12.7 Control in the Clock-synchronous 8-bit SIO Mode

The following registers control the serial bus interface in the clock-synchronous 8-bit SIO mode and provide its status information for monitoring.

Serial bus interface control register 0

SBIxCR0

	7	6	5	4	3	2	1	0
bit Symbol	SBIEN							
Read/Write	R/W	R						
After reset	0	0						
Function	SBI operation 0: Disable 1: Enable	This can be read as “0.”						
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0.”							
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0.”							
	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0.”							

<SBIEN>: To use the SBI, enable the SBI operation (“1”) before setting each register of SBI module.

Fig. 12-20 SIO Mode Registers

Serial bus interface control register 1

SBIxCR1

	7	6	5	4	3	2	1	0
bit Symbol	SIOS	SIOINH	SIOM1	SIOM0		SCK2	SCK1	SCK0
Read/Write	W				R	W		R/W
After reset	0	0	0	0	1	0	0	0
Function	Start transfer 0: Stop 1: Start	Transfer 0: Continue 1: Forced termination	Select transfer mode 00: Transmit mode 01: (Reserved) 10: Transmit/receive mode 11: Receive mode		This can be read as "1".	Select serial clock frequency		

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as "0".							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as "0".							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as "0".							

On writing <SCK2:0>: Select serial clock frequency

000	n = 3	2.5 MHz
001	n = 4	1.25 MHz
010	n = 5	625 kHz
011	n = 6	313 kHz
100	n = 7	156 kHz
101	n = 8	78 kHz
110	n = 9	39 kHz
111	—	External clock

System clock : fsys (=40 MHz)
Clock gear : fc/1
Frequency = $\frac{fsys/2}{2^n}$ [Hz]

(Note) Set <SIOS> to "0" and <SIOINH> to "1" before programming the transfer mode and the serial clock.

Fig. 12-21 SIO Mode Registers

Serial bus interface data buffer register

SBIxDBR

	7	6	5	4	3	2	1	0
bit Symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (Receive)/W (Transmit)							
After reset	Undefined							
Function	RX data/ TX data							
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
リセット後	0							
機　能	This can be read as “0”.							
	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

Serial bus interface control register 2

SBIxCR2

	7	6	5	4	3	2	1	0
bit Symbol					SBIM1	SBIM0		
Read/Write	R				W		R	
After reset	1				0	0	1	
Function	This can be read as “1”.				Select serial bus interface operating mode 00: Port mode 01: Clock-synchronous 8-bit SIO mode 10: I^2C bus mode 11: (Reserved)		This can be read as “1”.	
	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							
	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							
	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

Fig. 12-22 SIO Mode Registers

SBIxSR

Serial bus interface register

	7	6	5	4	3	2	1	0
bit Symbol					SIOF	SEF		
Read/Write	R				R		R	
After reset	1				0	0	1	
Function	This can be read as “1”.				Serial transfer status monitor 0: Completed 1: In progress	Shift operation status monitor 0: Completed 1: In progress	This can be read as “1”.	

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”.							

SBIxBR0

Serial bus interface baud rate register 0

	7	6	5	4	3	2	1	0
bit Symbol		I2SBI						
Read/Write	R	R/W	R					W
After reset	1	0	1					0
Function	This can be read as “1”.	IDLE 0: Stop 1: Operate	This can be read as “1”.					Make sure to write “0.”

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”							

	23	22	21	20	19	18	17	16
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”							

	31	30	29	28	27	26	25	24
bit Symbol								
Read/Write	R							
After reset	0							
Function	This can be read as “0”							

Fig. 12-23 SIO Mode Registers

12.7.1 Serial Clock

① Clock source

Internal or external clocks can be selected by programming SBIxCR1 <SCK2:0>.

Internal clocks

In the internal clock mode, one of the seven frequencies can be selected as a serial clock, which is output to the outside through the SCK pin. At the beginning of a transfer, the SCK pin output becomes the “H” level.

If the program cannot keep up with this serial clock rate in writing the transmit data or reading the received data, the SBI automatically enters a wait period. During this period, the serial clock is stopped automatically and the next shift operation is suspended until the processing is completed.

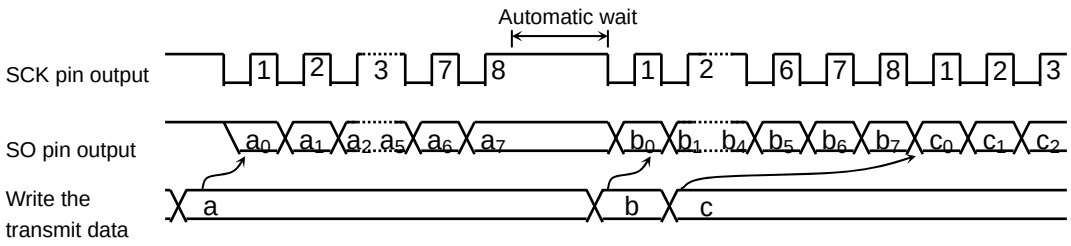


Fig. 12-24 Automatic Wait

External clock (<SCK2:0> = “111”)

The SBI uses an external clock supplied from the outside to the SCK pin as a serial clock. For proper shift operations, the serial clock at the “H” and “L” levels must have the pulse widths as shown below.

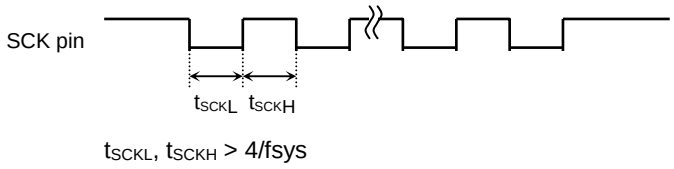


Fig. 12-25 Maximum Transfer Frequency of External Clock Input

② Shift Edge

Leading-edge shift is used in transmission. Trailing-edge shift is used in reception.

Leading-edge shift

Data is shifted at the leading edge of the serial clock (or the falling edge of the SCK pin input/output).

Trailing-edge shift

Data is shifted at the trailing edge of the serial clock (or the rising edge of the SCK pin input/output).

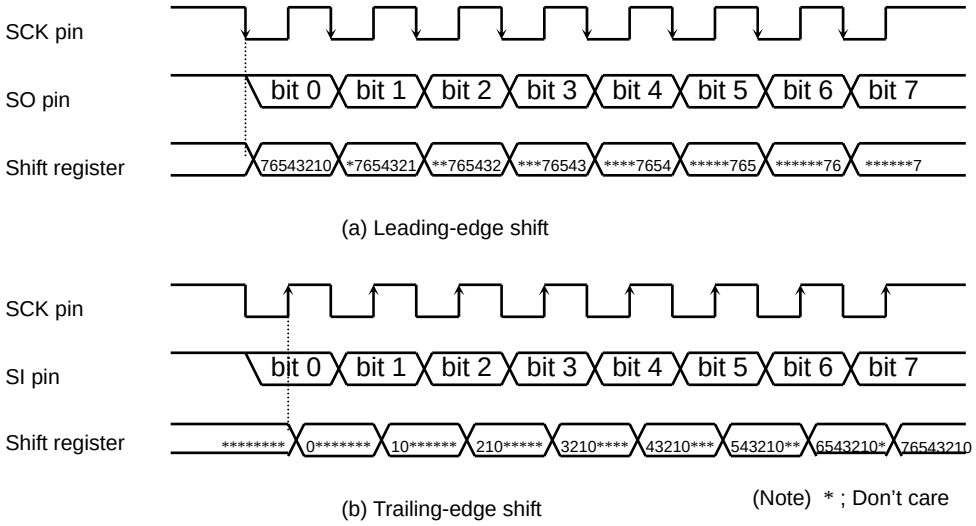


Fig. 12-26 Shift Edge

12.7.2 Transfer Modes

The transmit mode, the receive mode or the transmit/receive mode can be selected by programming SBIxCR1 <SIOM1:0>.

① 8-bit transmit mode

Set the control register to the transmit mode and write the transmit data to SBIxDBR.

After writing the transmit data, writing “1” to SBIxCR1 <SIOS> starts the transmission. The transmit data is moved from SBIxDBR to a shift register and output to the SO pin, with the least-significant bit (LSB) first, in synchronization with the serial clock. Once the transmit data is transferred to the shift register, SBIxDBR becomes empty, and the INTSBIx (buffer-empty) interrupt is generated, requesting the next transmit data.

In the internal clock mode, the serial clock will be stopped and automatically enter the wait state, if next data is not loaded after the 8-bit data has been fully transmitted. The wait state will be cleared when SBIxDBR is loaded with the next transmit data.

In the external clock mode, SBIxDBR must be loaded with data before the next data shift operation is started. Therefore, the data transfer rate varies depending on the maximum latency between when the interrupt request is generated and when SBIxDBR is loaded with data in the interrupt service program.

At the beginning of transmission, the same value as in the last bit of the previously transmitted data is output in a period from setting SBIxSR <SIOF> to “1” to the falling edge of SCK.

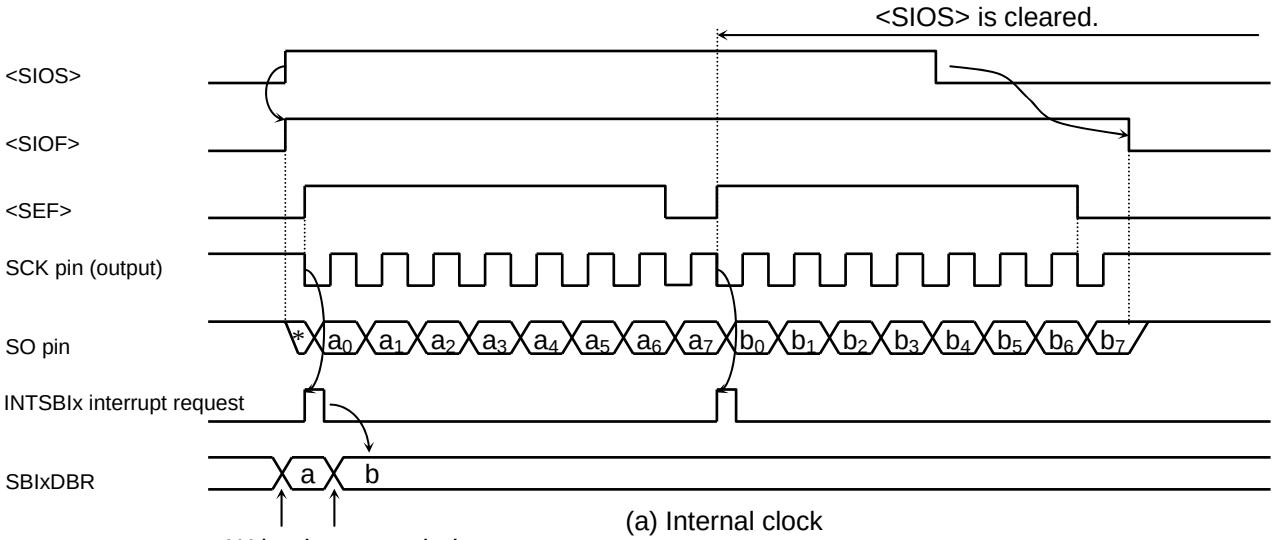
Transmission can be terminated by clearing <SIOS> to “0” or setting <SIOINH> to “1” in the INTSBIx interrupt service program. If <SIOS> is cleared, remaining data is output before transmission ends. The program checks SBIxSR <SIOF> to determine whether transmission has come to an end. <SIOF> is cleared to “0” at the end of transmission. If <SIOINH> is set to “1,” the transmission is aborted immediately and <SIOF> is cleared to “0”.

In the external clock mode, <SIOS> must be set to “0” before the next transmit data shift operation is started. Otherwise, operation will stop after dummy data is transmitted.

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	0	0	0	X	X	X	Selects the transmit mode.
SBIxDBR	←	X	X	X	X	X	X	X	X	Writes the transmit data.
SBIxCR1	←	1	0	0	0	0	X	X	X	Starts transmission.

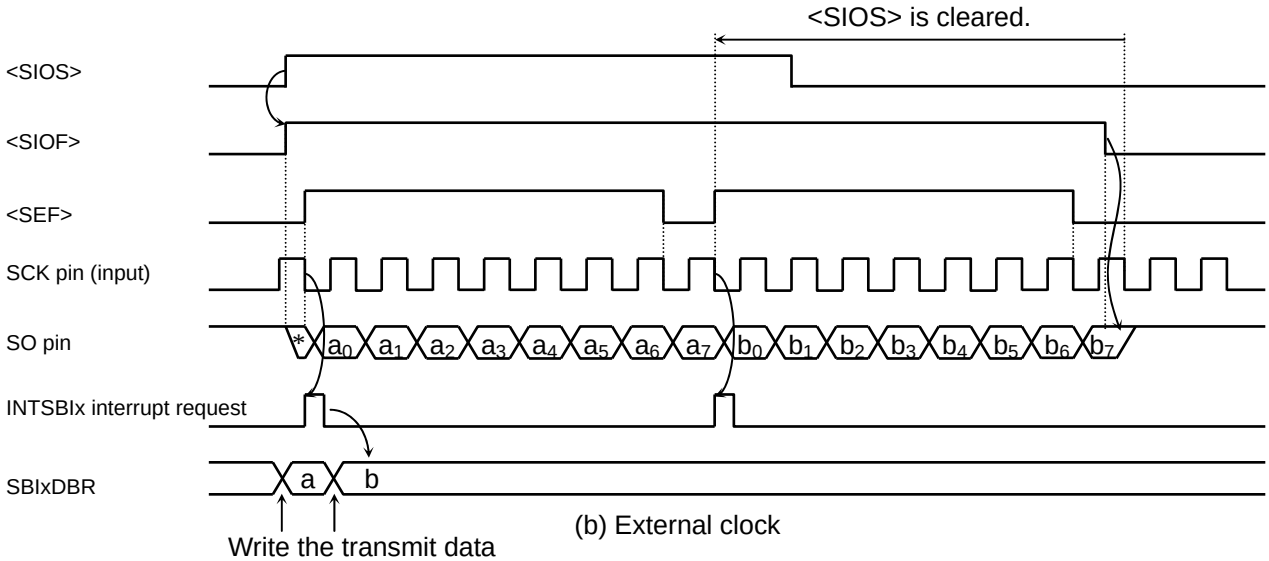
INTSBIx interrupt

SBIxDBR	←	X	X	X	X	X	X	X	X	Writes the transmit data.



Write the transmit data

(a) Internal clock



Write the transmit data

(b) External clock

Fig. 12-27 Transmit Mode

Example: Example of programming (external clock) to terminate transmission by <SIO>

7 6 5 4 3 2 1 0	
if SBISR<SIOF> ≠ 0	Recognizes the completion of the transmission.
Then	
if SCK ≠ 1	Recognizes “1” is set to the SCK pin by monitoring the port.
Then	
SBIxCR1 ← 0 0 0 0 0 1 1 1	Completes the transmission by setting <SIOS> = 0.

② 8-bit receive mode

Set the control register to the receive mode. Then writing “1” to SBIxCR1 <SIOS> enables reception. Data is taken into the shift register from the SI pin, with the least-significant bit (LSB) first, in synchronization with the serial clock. Once the shift register is loaded with the 8-bit data, it transfers the received data to SBIxDBR and the INTSBIx (buffer-full) interrupt request is generated to request reading the received data. The interrupt service program then reads the received data from SBIxDBR.

In the internal clock mode, the serial clock will be stopped and automatically be in the wait state until the received data is read from SBIxDBR.

In the external clock mode, shift operations are executed in synchronization with the external clock. The maximum data transfer rate varies, depending on the maximum latency between generating the interrupt request and reading the received data.

Reception can be terminated by clearing <SIOS> to “0” or setting <SIOINH> to “1” in the INTSBIx interrupt service program. If <SIOS> is cleared, reception continues until all the bits of received data are written to SBIxDBR. The program checks SBIxSR <SIOF> to determine whether reception has come to an end. <SIOF> is cleared to “0” at the end of reception. After confirming the completion of the reception, last received data is read. If <SIOINH> is set to “1,” the reception is aborted immediately and <SIOF> is cleared to “0.” (The received data becomes invalid, and there is no need to read it out.)

(Note) The contents of SBIxDBR will not be retained after the transfer mode is changed. The ongoing reception must be completed by clearing <SIOS> to “0” and the last received data must be read before the transfer mode is changed.

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	1	1	0	X	X	X	Selects the receive mode.
SBIxCR1	←	1	0	1	1	0	0	0	0	Starts reception.

INTSBIx interrupt

Reg.	← SBIxDBR	Reads the received data.

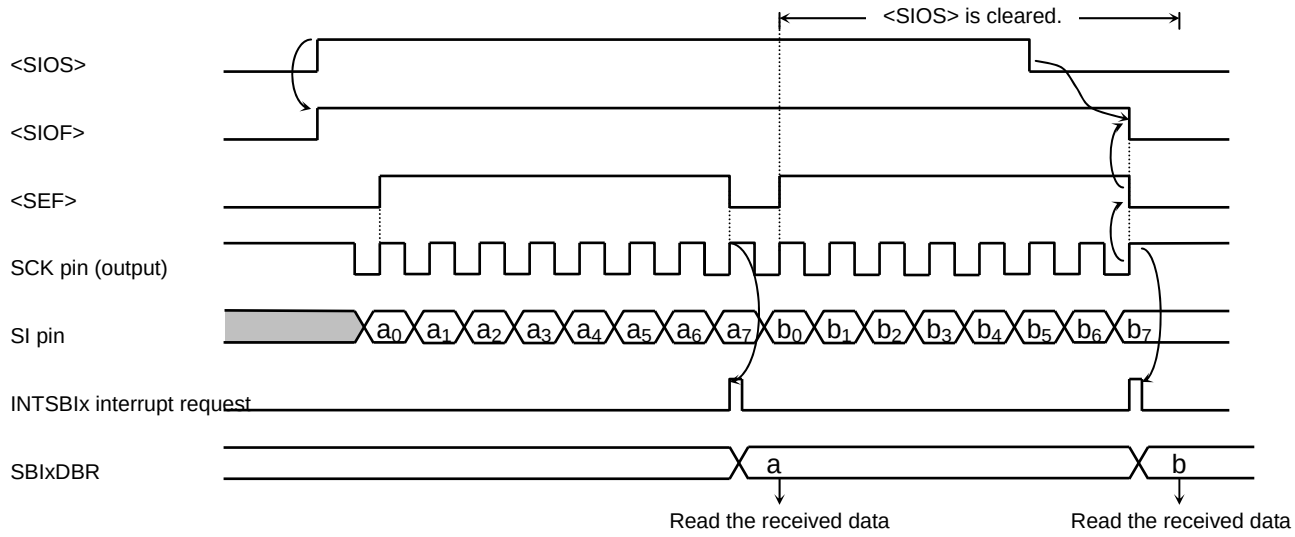


Fig. 12-28 Receive Mode (Example: Internal Clock)

③ 8-bit transmit/receive mode

Set the control register to the transfer/receive mode. Then writing the transmit data to SBIxDBR and setting SBIxCR1 <SIOS> to “1” enables transmission and reception. The transmit data is output through the SO pin at the falling of the serial clock, and the received data is taken in through the SI pin at the rising of the serial clock, with the least-significant bit (LSB) first. Once the shift register is loaded with the 8-bit data, it transfers the received data to SBIxDBR and the INTSBIx interrupt request is generated. The interrupt service program reads the received data from the data buffer register and writes the next transmit data. Because SBIxDBR is shared between transmit and receive operations, the received data must be read before the next transmit data is written.

In the internal clock operation, the serial clock will be automatically in the wait state until the received data is read and the next transmit data is written.

In the external clock mode, shift operations are executed in synchronization with the external serial clock. Therefore, the received data must be read and the next transmit data must be written before the next shift operation is started. The maximum data transfer rate for the external clock operation varies depending on the maximum latency between when the interrupt request is generated and when the transmit data is written.

At the beginning of transmission, the same value as in the last bit of the previously transmitted data is output in a period from setting <SIOF> to “1” to the falling edge of SCK.

Transmission and reception can be terminated by clearing <SIOS> to “0” or setting SBIxCR1 <SIOINH> to “1” in the INTSBIx interrupt service program. If <SIOS> is cleared, transmission and reception continue until the received data is fully transferred to SBIxDBR. The program checks SBIxSR <SIOF> to determine whether transmission and reception have come to an end. <SIOF> is cleared to “0” at the end of transmission and reception. If <SIOINH> is set, the transmission and reception are aborted immediately and <SIOF> is cleared to “0.”

(Note) The contents of SBIxDBR will not be retained after the transfer mode is changed. The ongoing transmission and reception must be completed by clearing <SIOS> to “0” and the last received data must be read before the transfer mode is changed.

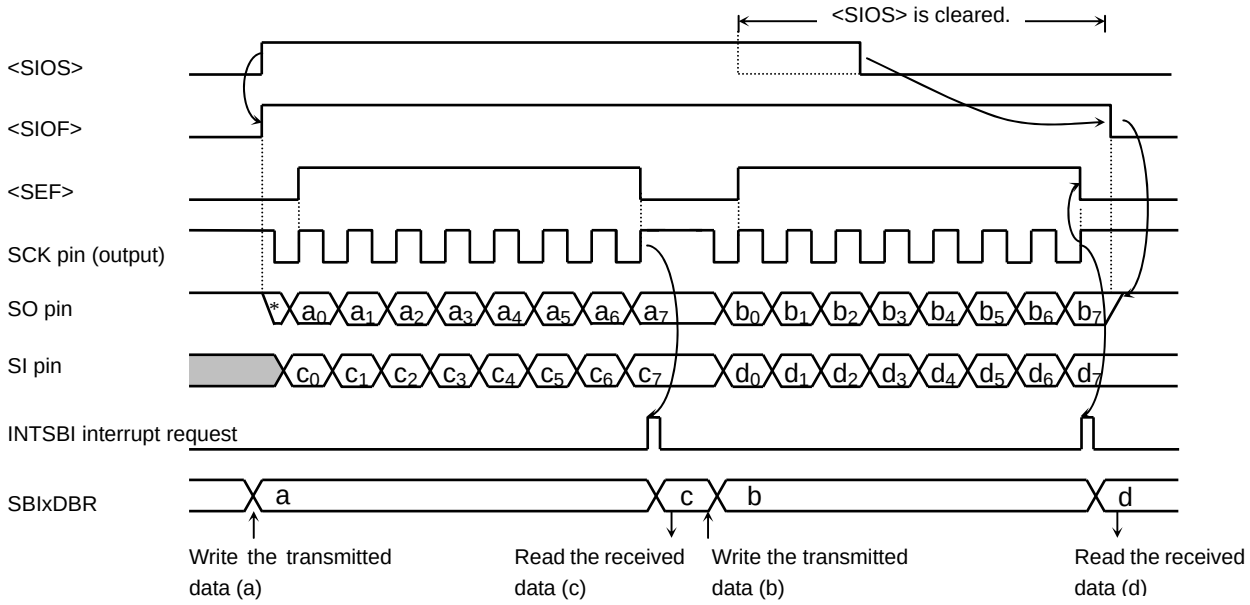


Fig. 12-29 Transmit/Receive Mode (Example: Internal Clock)

		7 6 5 4 3 2 1 0	
SBIxCR1	←	0 1 1 0 0 X X X	Selects the transmit mode.
SBIxDBR	←	X X X X X X X X	Writes the transmit data.
SBIxCR1	←	1 0 1 0 0 X X X	Starts reception/transmission.

INTSBIx interrupt

Reg.	←	SBIxDBR	Reads the received data.
SBIxDBR	←	X X X X X X X X	Writes the transmit data.

13 Consumer Electronics Control (CEC)

13.1 Outline

This IP enables to transmit or receive data that conforms to Consumer Electronics Control (hereafter referred to as CEC) protocol (conforms to HDMI 1.3a specifications).

13.1.1 Reception

- Clock sampling at 32KHz
  - Adjustable noise canceling time
- Data reception per 1byte
  - Flexible data sampling point
  - Data reception is available even when an address discrepancy is detected.
- Error detection
  - Cycle error（min./ max.）
  - ACK collision
  - Waveform error

13.1.2 Transmission

- Data transmission per 1byte
  - Triggered by auto-detection of bus free state
- Flexible waveform
  - Adjustable rising edge and cycle
- Error detection
  - Arbitration lost
  - ACK response error

<<IMPORTANT>> Be careful about the following in the receive operation.

When data reception at logical address discrepancy is disabled (CECRCR1<CECOTH>="0")	**The initiator must send the last block of data with the EOM bit set to "1". If the last block is sent with EOM="0", the subsequent operation cannot be guaranteed.**
When data reception at logical address discrepancy is enabled (CECRCR1<CECOTH>="1")	**If the initiator sends a new message beginning with the start bit without having sent the last block with EOM="1", a maximum cycle error is determined for the ACK bit and an interrupt is generated. Then, the receive operation is performed in the usual way.**

13.2 Registers

13.2.1 Control Registers and Addresses

The control registers and address for CEC are as follows.

Registers		Addresses
CEC Enable Register	CECEN	0x4004_0300
Logical Address Register	CECADD	0x4004_0304
Software Reset Register	CECRESET	0x4004_0308
Receive Enable Register	CECREN	0x4004_030C
Receive Buffer Register	CECRBUF	0x4004_0310
Receive Control Register 1	CECRCR1	0x4004_0314
Receive Control Register 2	CECRCR2	0x4004_0318
Receive Control Register 3	CECRCR3	0x4004_031C
Transmit Enable Register	CECTEN	0x4004_0320
Transmit Buffer Register	CECTBUF	0x4004_0324
Transmit Control Register	CECTCR	0x4004_0328
Receive Interrupt Status Register	CECRSTAT	0x4004_032C
Transmit Interrupt Status Register	CECTSTAT	0x4004_0330

13.2.2 CEC Enable Register [CECEN]

	7	6	5	4	3	2	1	0
bit Symbol	—						I2CEC	CECEN
Read/Write	R						R/W	R/W
After reset	0						0	0
Function	"0" is read.						CEC operation at IDLE 0: Disabled 1: Enabled	CEC operation 0: Disabled 1: Enabled

<I2CEC>: Controls the CEC operation at the IDLE mode.
Set this bit to "1" when using CEC at the IDLE mode.
The <I2CEC> and <CECEN> bits can be set simultaneously.

<CECEN>: Specifies the CEC operation.
Enable CEC before using.
When the CEC operation is disabled, no clocks are supplied to the CEC module except for the enable register. Thus power consumption can be reduced.
When CEC is disabled after it was enabled, each register setting is maintained.

13.2.3 Logical Address Register [CECADD]

	15	14	13	12	11	10	9	8
bit Symbol	CECADD 15	CECADD 14	CECADD 13	CECADD 12	CECADD 11	CECADD 10	CECADD 9	CECADD 8
Read/Write	R/W							
After reset	0							
Function	Logical address 15	Logical address 14	Logical address 13	Logical address 12	Logical address 11	Logical address 10	Logical address 9	Logical address 8

	7	6	5	4	3	2	1	0
bit Symbol	CECADD 7	CECADD 6	CECADD 5	CECADD 4	CECADD 3	CECADD 2	CECADD 1	CECADD 0
Read/Write	R/W							
After reset	0							
Function	Logical address 7	Logical address 6	Logical address 5	Logical address 4	Logical address 3	Logical address 2	Logical address 1	Logical address 0

<CECADD15:0>: Specifies the logical address assigned to CEC.
Multiple addresses can be set simultaneously since each bit corresponds with each address.

(Note) A broadcast message is received regardless of the register setting.
By allocating a logical address of a device to 15, logical "0" is sent as an ACK response to the broadcast message.

13.2.4 Software Reset Register [CECRESET]

	7	6	5	4	3	2	1	0
bit Symbol	—							CEC RESET
Read/Write	R							W
After reset	0							0
Function	"0" is read.							Software reset 0: Disabled 1: Enabled "0" is read.

<CECRESET>: Stops all the CEC operation and initializes the register.
Setting this bit to "1" affects as follows:
Reception: Stops immediately. The received data is discarded.
Transmission (including the CEC line): Stops immediately.
Register: All the registers other than CECEN are initialized.

13.2.5 Receive Enable Register [CECREN]

	7	6	5	4	3	2	1	0
bit Symbol	—							CECREN
Read/Write	R							R/W
After reset	0							Undefined
Function	"0" is read.							Reception control [Write] 0: Disabled 1: Enabled [Read] 0: Stopped 1: in operation

<CECREN>: Controls the reception operation of CEC.
Writing "0" or "1" to this bit enables or disables data reception. This bit becomes ready for data reception by writing "1".
The state of the reception circuit is monitored by reading this bit. It enables you to check if what you set has properly been reflected.

(Note 1) Enable the <CECREN> bit after setting the reception control register 1, 2 and 3.

(Note 2) It takes a little time to reflect the setting of the <CECREN> bit to the circuit. Stop transmission and reception before changing the settings or enabling the transmission and reception.

13.2.6 Receive Buffer Register [CECRBUF]

	15	14	13	12	11	10	9	8
bit Symbol	—						CECACK	CECEOM
Read/Write	R						R	R
After reset	0						0	0
Function	"0" is read.						ACK bit	EOM bit

	7	6	5	4	3	2	1	0
bit Symbol	CECRBUF 7	CECRBUF 6	CECRBUF 5	CECRBUF 4	CECRBUF 3	CECRBUF 2	CECRBUF 1	CECRBUF 0
Read/Write	R							
After reset	0							
Function	Received data							

<CECACK>: Reads the received ACK bit.

<CECEOM>: Reads the received EOM bit.

<CECRBUF7:0>: Reads one byte of data received. The bit 7 is the MSB.

(Note 1) Writing to this register is ignored.

(Note 2) Read this register as soon as a receive interrupt is generated. The subsequent reading data may not be ensured.

13.2.7 Receive Control Register 1 [CECRCR1]

	31	30	29	28	27	26	25	24
bit Symbol	—							CECACK DIS
Read/Write	R							R/W
After reset	0							0
Function	“0” is read.							Logical “0” as ACK response 0: send 1: not send

	23	22	21	20	19	18	17	16
bit Symbol	—		CECHNC 1	CECHNC 0	—	CECLNC 2	CECLNC 1	CECLNC 0
Read/Write	R		R/W		R	R/W		
After reset	0		0		0	0		
Function	“0” is read.		The number of “1” samplings for noise cancellation. 00: 1 01: 2 10: 3 11: 4		“0” is read.	The number of “0” samplings for noise cancellation. 000: 1 001: 2 010: 3 011: 4 100: 5 101: 6 110: 7 111: 8		

	15	14	13	12	11	10	9	8
bit Symbol	—	CECMIN2	CECMIN1	CECMIN0	—	CECMAX 2	CECMAX 1	CECMAX 0
Read/Write	R	R/W			R	R/W		
After reset	0	0			0	0		
Function	“0” is read.	Time to identify as min. cycle error 000: 2.05ms 001: 2.05ms+1cycle 010: 2.05ms+2cycles 011: 2.05ms+3cycles 100: 2.05ms-1cycle 101: 2.05ms-2cycles 110: 2.05ms-3cycles 111: 2.05ms-4cycles			“0” is read.	Time to identify as max. cycle error 000: 2.75ms 001: 2.75ms+1cycle 010: 2.75ms+2cycles 011: 2.75ms+3cycles 100: 2.75ms-1cycle 101: 2.75ms-2cycles 110: 2.75ms-3cycles 111: 2.75ms-4cycles		

	7	6	5	4	3	2	1	0
bit Symbol	—	CECDAT 2	CECDAT 1	CECDAT 0	CECTOU T1	CECTOU T0	CECRI HLD	CECOTH
Read/Write	R	R/W			R/W		R/W	R/W
After reset	0	0			0		0	0
Function	“0” is read.	Point of determining the data as 0 or 1. 000: 1.05ms 001: 1.05ms+2cycles 010: 1.05ms+4cycles 011: 1.05ms+6cycles 100: 1.05ms-2cycles 101: 1.05ms-4cycles 110: 1.05ms-6cycles 111: Reserved			Cycle to identify timeout 00: 1 bit cycle 01: 2 bit cycles 10: 3 bit cycles 11: Reserved		Error interrupt suspend 0: Yes 1: No	Data reception at logical address discrepancy 0: Yes 1: No

<CECACKDIS>: Specifies if logical “0” is sent or not as an ACK response to the data block when destination address corresponds with the address set in the logical address register. (The header block sends logical “0” as an ACK response regardless of the bit setting when detecting the addresses corresponding).

<CECHNC1:0>: Specifies the time of the noise cancellation for each sampling clock cycle when detecting “1”.
It is considered as noise if “1”s of the same number as the specified cycles are not sampled.

<CECLNC2:0>: Specifies the time of the noise cancellation for each sampling clock cycle when detecting “0”.
It is considered as noise if “0”s of the same number as the specified cycles are not sampled.

<CECMIN2:0>: Specifies the minimum time to identify a valid bit.
Enables to specify it for each sampling clock cycle between the ranges of -4 to +3 cycles from approx. 2.05 ms.
An interrupt is generated and “0” is output to CEC for approx. 3.6 ms when one bit cycle is shorter than the specified time.

<CECMAX2:0>: Specifies the maximum time to identify a valid bit.
Enables to specify it for each sampling clock cycle between the ranges of -4 to +3 cycles from approx. 2.75 ms.
An interrupt is generated when one bit cycle is longer than the specified time.

<CECDAT2:0>: Specifies the point of determining the data as 0 or 1.
Enables to specify it per two sampling clock cycles between the ranges of + or - 6 cycles from approx. 1.05 ms.

<CECTOUT1:0>: Specifies the time to determine a timeout. Enables to specify it between 1 bit and 3 bits for each bit cycle.
This setting is used to detect a timeout occurs when the <CECRIHLD> bit is valid.

<CECRIHLD>: Specifies if a receive error interrupt (maximum cycle error, buffer overrun and waveform error) is suspended or not.
Setting “1” generates no interrupt at the error detection. If data continues to an ACK bit, an ACK response is executed by a reversed logic. If the subsequent bits are interrupted, it is determined as a timeout, based on the setting in <CECTOUT>. After the ACK response or the timeout determination, an interrupt is generated.

<CECOTH>: Specifies if data is received or not when destination address does not correspond with the address set in the logical address register.

(Note 1) The settings in <CECHNC>, <CECLNC> and <CECDAT> are also used in receiving an ACK response at transmission.

(Note 2) Changing the configurations during transmission or reception may harm its proper operation. Before the change, set the CECREN <CECREN> bit to disable the reception and read the <CECREN> bit and the CECTEN <CECTEN> bit to ensure that the operation is stopped.

(Note 3) A broadcast message is received regardless of the <CECOTH> register setting.

13.2.8 Receive Control Register 2 [CECRCR2]

	15	14	13	12	11	10	9	8
bit Symbol	—	CEC SWAV32	CEC SWAV31	CEC SWAV30	—	CEC SWAV22	CEC SWAV21	CEC SWAV20
Read/Write	R	R/W			R	R/W		
After reset	0	0			0	0		
Function	“0” is read.	Max. cycle to detect start bit 000: 4.7ms 001: 4.7ms+1cycle 010: 4.7ms+2cycles 011: 4.7ms+3cycles 100: 4.7ms+4cycles 101: 4.7ms+5cycles 110: 4.7ms+6cycles 111: 4.7ms+7cycles			“0” is read.	Min. cycle to detect start bit 000: 4.3ms 001: 4.3ms-1cycle 010: 4.3ms-2cycles 011: 4.3ms-3cycles 100: 4.3ms-4cycles 101: 4.3ms-5cycles 110: 4.3ms-6cycles 111: 4.3ms-7cycles		

	7	6	5	4	3	2	1	0
bit Symbol	—	CEC SWAV12	CEC SWAV11	CEC SWAV10	—	CEC SWAV02	CEC SWAV01	CEC SWAV00
Read/Write	R	R/W			R	R/W		
After reset	0	0			0	0		
Function	“0” is read.	Max. time of start bit rising timing. 000: 3.9ms 001: 3.9ms+1cycle 010: 3.9ms+2cycles 011: 3.9ms+3cycles 100: 3.9ms+4cycles 101: 3.9ms+5cycles 110: 3.9ms+6cycles 111: 3.9ms+7cycles			“0” is read.	Min. time of start bit rising timing. 000: 3.5ms 001: 3.5ms-1cycle 010: 3.5ms-2cycles 011: 3.5ms-3cycles 100: 3.5ms-4cycles 101: 3.5ms-5cycles 110: 3.5ms-6cycles 111: 3.5ms-7cycles		

<CECSWAV3 2:0>: Specifies the cycles to detect a start bit.
<CECSWAV2 2:0>: <CECSWAV3> is for the maximum cycles. Enables to set it for each sampling clock cycle between the ranges of 0 to +7 cycles from default value (4.7 ms).
<CECSWAV2> is for the minimum cycles. Enables to set it for each sampling clock cycle between the ranges of 0 to +7 cycles from default value (4.3 ms).

<CECSWAV1 2:0>: Specifies the rising timing of a start bit in its detection.
<CECSWAV0 2:0>: <CECSWAV1> is for the maximum time of the rising timing. Enables to set it for each sampling clock cycle between the ranges of 0 to +7 cycles from default value (3.9 ms).
<CECSWAV0> is for the minimum time of the rising timing. Enables to set it for each sampling clock cycle between the ranges of 0 to -7 cycles from default value (3.5 ms).

(Note) Changing the configurations during transmission or reception may harm its proper operation. Before the change, set CECREN <CECREN> to disable the reception and read the <CECREN> bit to ensure that the operation is stopped.

13.2.9 Receive Control Register 3 [CECRCR3]

	23	22	21	20	19	18	17	16
bit Symbol	—	CEC WAV32	CEC WAV31	CEC WAV30	—	CEC WAV22	CEC WAV21	CEC WAV20
Read/Write	R	R/W			R	R/W		
After reset	0	0			0	0		
Function	"0" is read.	The latest rising timing of logical "0" determined as proper waveform. 000: 1.7ms 001: 1.7ms+1cycle 010: 1.7ms+2cycles 011: 1.7ms+3cycles 100: 1.7ms+4cycles 101: 1.7ms+5cycles 110: 1.7ms+6cycles 111: 1.7ms+7cycles			"0" is read.	The fastest rising timing of logical "0" determined as proper waveform. 000: 1.3ms 001: 1.3ms-1cycle 010: 1.3ms-2cycles 011: 1.3ms-3cycles 100: 1.3ms-4cycles 101: 1.3ms-5cycles 110: 1.3ms-6cycles 111: 1.3ms-7cycles		

	15	14	13	12	11	10	9	8
bit Symbol	—	CEC WAV12	CEC WAV11	CEC WAV10	—	CEC WAV02	CEC WAV01	CEC WAV00
Read/Write	R	R/W			R	R/W		
After reset	0	0			0	0		
Function	"0" is read.	The latest rising timing of logical "1" determined as proper waveform. 000: 0.8ms 001: 0.8ms+1cycle 010: 0.8ms+2cycles 011: 0.8ms+3cycles 100: 0.8ms+4cycles 101: 0.8ms+5cycles 110: 0.8ms+6cycles 111: 0.8ms+7cycles			"0" is read.	The fastest rising timing of logical "1" determined as proper waveform. 000: 0.4ms 001: 0.4ms-1cycle 010: 0.4ms-2cycles 111: 0.4ms-3cycles 100: 0.4ms-4cycles 101: 0.4ms-5cycles 110: 0.4ms-6cycles 111: 0.4ms-7cycles		

	7	6	5	4	3	2	1	0
bit Symbol	—							CEC WAVEN
Read/Write	R							R/W
After reset	0							0
Function	"0" is read.							Waveform error detection 1: Enabled 0: Disabled

(Note) **Changing the configurations during transmission or reception may harm its proper operation. Before the change, configure the <CECREN> bits to disable the reception and read the <CECREN> bit to ensure that the operation is stopped.**

<CECWAV3 2:0>: This setting is enabled when the <CECWAVEN> bit is set to “1”.
By setting these bits, an error is detected if rising edge of the received waveform comes later than that of proper logical “0”. Enables to set it for each sampling clock cycle between the ranges of 0 to 7 cycles from defined maximum tolerance (1.7 ms). The received waveform is considered to be an error if a rising edge is not detected from the start point of the bit to the value specified in <CECWAV3>.

<CECWAV2 2:0>:
<CECWAV1 2:0>: This setting is enabled when the <CECWAVEN> bit is set to “1”.
By setting these bits, an error is detected if rising edge of the received waveform comes faster than logical “0” and later than that of proper logical “1”.
Enables to set <CECWAV1> for each sampling clock cycle between the ranges of 0 to 7 cycles from defined maximum tolerance (0.8 ms) of logical “1” waveform.
Enables to set <CECWAV2> for each sampling clock cycle between the ranges of 0 to -7 cycles from defined minimum tolerance (1.3 ms) of logical “0” waveform.
The received waveform is considered to be an error if a rising edge is detected between the values specified in <CECWAV2> and <CECWAV1>.

<CECWAV0 2:0>: This setting is enabled when the <CECWAVEN> bit is set to “1”.
By setting these bits, an error is detected if rising edge of the received waveform comes faster than that of proper logical “1”. Enables to set <CECWAV0> for each sampling clock cycle between the ranges of 0 to -7 cycles from defined minimum tolerance (0.4 ms).
The received waveform is considered to be an error if a rising edge is not detected from a start point of the bit to the value specified in <CECWAV0>.

<CECWAVEN>: Detects a received waveform does not identical to the one defined and generates waveform error interrupt.
If enabled, an error is detected according to the setting of <CECWAV0> <CECWAV1> <CECWAV2> <CECWAV3>.

13.2.10 Transmit Enable Register [CECTEN]

	7	6	5	4	3	2	1	0
bit Symbol	—						CEC TRANS	CECTEN
Read/Write	R						R	W
After reset	0						0	Undefined
Function	"0" is read.						Transmission state 0: not in progress 1: in progress	Transmission control 0: Disabled 1: Enabled

<CECTRANS>: Indicates whether the transmission is in progress or not.
It indicates "1" upon starting the transmission of the start bit. It indicates "0" if transmission is completed or an interrupt is generated.
Writing to this bit is ignored.

<CECTEN>: Controls the CEC transmission.
Writing this bit enables or disables the transmission. Writing "1" to this bit initiates the transmission. This bit is automatically cleared by a transmit completion interrupt or an error interrupt.

(Note 1) Set <CECTEN> after setting the transmit buffer register and transmit control register.

(Note 2) Stop transmission and reception before changing the settings or enabling the transmission and reception.

13.2.11 Transmit Buffer Register [CECTBUF]

	15	14	13	12	11	10	9	8
bit Symbol	—							CECTEOM
Read/Write	R							R/W
After reset	0							0
Function	"0" is read.							EOM bit

	7	6	5	4	3	2	1	0
bit Symbol	CECTBUF 7	CECTBUF 6	CECTBUF 5	CECTBUF 4	CECTBUF 3	CECTBUF 2	CECTBUF 1	CECTBUF 0
Read/Write	R/W							
After reset	0							
Function	Transmitted data							

<CECTEOM>: Specifies the EOM bit to transmit.

<CECTBUF7:0>: Specifies a byte of data to transmit. The bit 7 is the MSB.

13.2.12 Transmit Control Register [CECTCR]

	23	22	21	20	19	18	17	16
bit Symbol	—	CECSTRS 2	CECSTRS 1	CECSTRS 0	—	CECSPRD 2	CECSPRD 1	CECSPRD 0
Read/Write	R	R/W			R	R/W		
After reset	0	0			0	0		
Function	“0” is read.	Rising timing of start bit 000: Reference value (RV) 001: RV –1cycle 010: RV –2cycle 011: RV –3cycle 100: RV –4cycle 101: RV –5cycle 110: RV –6cycle 111: RV –7cycle			“0” is read.	Start bit cycle 000: RV 001: RV –1cycle 010: RV –2cycle 011: RV –3cycle 100: RV –4cycle 101: RV –5cycle 110: RV –6cycle 111: RV –7cycle		

	15	14	13	12	11	10	9	8
bit Symbol	—	CECDTRS 2	CECDTRS 1	CECDTRS 0	CECDPRD 3	CECDPRD 2	CECDPRD 1	CECDPRD 0
Read/Write	R	R/W			R/W			
After reset	0	0			0			
Function	“0” is read.	Rising timing of data bit 000: RV 001: RV –1cycle 010: RV –2cycle 011: RV –3cycle 100: RV –4cycle 101: RV –5cycle 110: RV –6cycle 111: RV –7cycle			Data bit cycle 0000: RV 0001: RV –1cycle 0010: RV –2cycle 0011: RV –3cycle 0100: RV –4cycle 0101: RV –5cycle 0110: RV –6cycle 0111: RV –7cycle		1000: RV –8cycle 1001: RV –9cycle 1010: RV –10cycle 1011: RV –11cycle 1100: RV –12cycle 1101: RV –13cycle 1110: RV –14cycle 1111: RV –15cycle	

	7	6	5	4	3	2	1	0
bit Symbol	—			CECBRD	CECFREE 3	CECFREE 2	CECFREE 1	CECFREE 0
Read/Write	R			R/W	R/W			
After reset	0			0	0			
Function	“0” is read.			Broadcast transmission 0: No 1: Yes	Time of bus to be free 0000: 1 bit cycle 0001: 2 bit cycle 0010: 3 bit cycle 0011: 4 bit cycle 0100: 5 bit cycle 0101: 6 bit cycle 0110: 7 bit cycle 0111: 8 bit cycle		1000: 9 bit cycle 1001: 10 bit cycle 1010: 11 bit cycle 1011: 12 bit cycle 1100: 13 bit cycle 1101: 14 bit cycle 1110: 15 bit cycle 1111: 16 bit cycle	

<CECSTRS2:0>: Specifies the rising timing of a start bit.
Enables to set it for each sampling clock cycle between the ranges of 0 to -7 cycles from default value (3.7 ms approx.).

<CECSPRD2:0>: Specifies a cycle of a start bit.
Enables to set it for each sampling clock cycle between the ranges of 0 to -7 cycles from default value (4.5 ms approx.).

<CECDTRS2:0>: Specifies the rising timing of a data bit.
Enables to set it for each sampling clock cycle between the ranges of 0 to -7 cycles from default value (logical “1”: 0.6 ms approx., logical “0”: 1.5 ms approx.).

<CECDPRD3:0>: Specifies a cycle of a data bit.
Enables to set it for each sampling clock cycle between the ranges of 0 to -15 cycles from default value (2.4 ms approx.).

<CECBRD>: Set this bit to “1” when transmitting a broadcast message.

<CECFREE3:0>: Specifies time of a bus to be free that checked before transmission. Start transmission after checking the CEC line kept inactive during the specified cycles.

13.2.13 Receive Interrupt Status Register [CECRSTAT]

	7	6	5	4	3	2	1	0
bit Symbol	—	CECRIWAV	CECRIOR	CECRIACK	CECRIMIN	CECRIMAX	CECRISTA	CECRIEND
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
Function	"0" is read.	Interrupt flag 1: Wave form error	Interrupt flag 1: Receive buffer overrun	Interrupt flag 1: ACK collision	Interrupt flag 1: Min. cycle error	Interrupt flag 1: Max. cycle error	Interrupt flag 1: Start bit detection	Interrupt flag 1: Completion of 1 byte data reception

<CECRIWAV>: Indicates that waveform error is detected. The error occurs when waveform error detection is enabled in CECRCR3 <CECWAVEN>.

<CECRIOR>: Indicates the receive buffer receives next data before reading the data that had already been set.

<CECRIACK>: Indicates "0" is detected after the specified time to output ACK bit "0".

<CECRIMIN>: Indicates one bit cycle is shorter than the minimum cycle error detection time specified in CECRCR1<CECMIN>.

<CECRIMAX>: Indicates one bit cycle is longer than the maximum cycle error detection time specified in CECRCR1<CECMAX>.

<CECRISTA>: Indicates a start bit is detected.

<CECRIEND>: Indicates 1 byte of data reception is completed.

(Note) Writing to this bit is ignored.

13.2.14 Transmit Interrupt Status Register [CECTSTAT]

	7	6	5	4	3	2	1	0
bit Symbol	—			CECTIUR	CECTIACK	CECTIAL	CECTIEND	CECTISTA
Read/Write	R			R	R	R	R	R
After reset	0			0	0	0	0	0
Function	"0" is read.			Interrupt flag 1: Transmit buffer underrun	Interrupt flag 1: ACK error detection	Interrupt flag 1: Arbitation lost occurs	Interrupt flag 1: data transmission is completed	Interrupt flag 1: Start transmission

<CECTIUR>: Indicates next data has not set to the transmission buffer within a byte of data transmission.

<CECTIACK>: Indicates one of the following conditions occurs.
- When logical "0" is not detected in transmission to the specific address.
- When logical "1" is not detected in transmission of a broadcast message .

<CECTIAL>: Indicates "0" is detected while transmitting "1".

<CECTIEND>: Indicates data transmission including the EOM bit is completed.

<CECTISTA>: Indicates 1 byte of data transmission is started.

(Note) Writing to this bit is ignored.

13.3 Operations

13.3.1 Reception

13.3.1.1 Sampling Clock

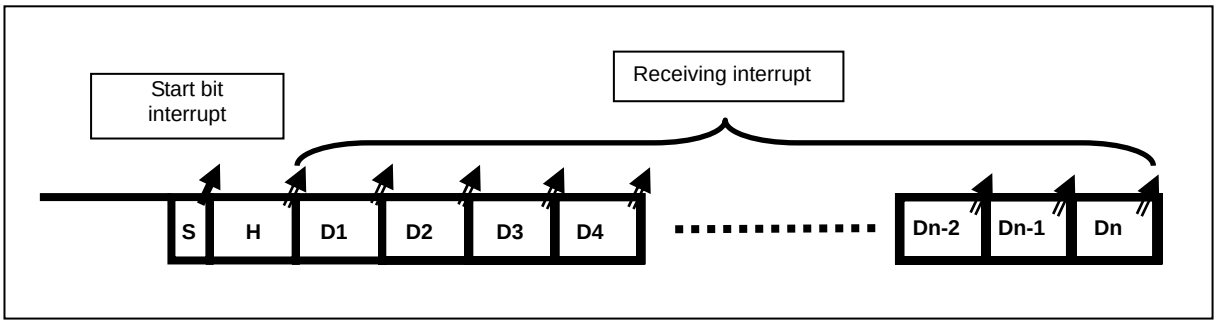
CEC lines are sampled by a low speed clock.

13.3.1.2 Basic Operation

If reception is enabled, detecting a start bit generates a start bit interrupt and starts receiving data per byte. The interrupt is generated after a byte of data (8 bit), the EOM bit and the ACK bit are received. The data, EOM and ACK bit are stored in a buffer.

Detecting an error during data reception causes an error interrupt, and CEC waits for the next start bit. The received data is discarded.

(Note) The initiator must send the last block of data with the EOM bit set to "1". If the last block is sent with EOM ="0", the subsequent operation cannot be guaranteed.



13.3.1.3 Preconfiguration

Before receiving data, reception settings to the Logical Address Register <CECADD>, the Receive Control Register 1 <CECRCR1>, the Receive Control Register 2 <CECRCR2> and the Receive Control Register 3 <CECRCR3> are required.

(1) Logical Address Configuration

Configure logical address assigned to this product to the CECADD register. Multiple addresses can be set simultaneously since every bit in this register corresponds with each address.

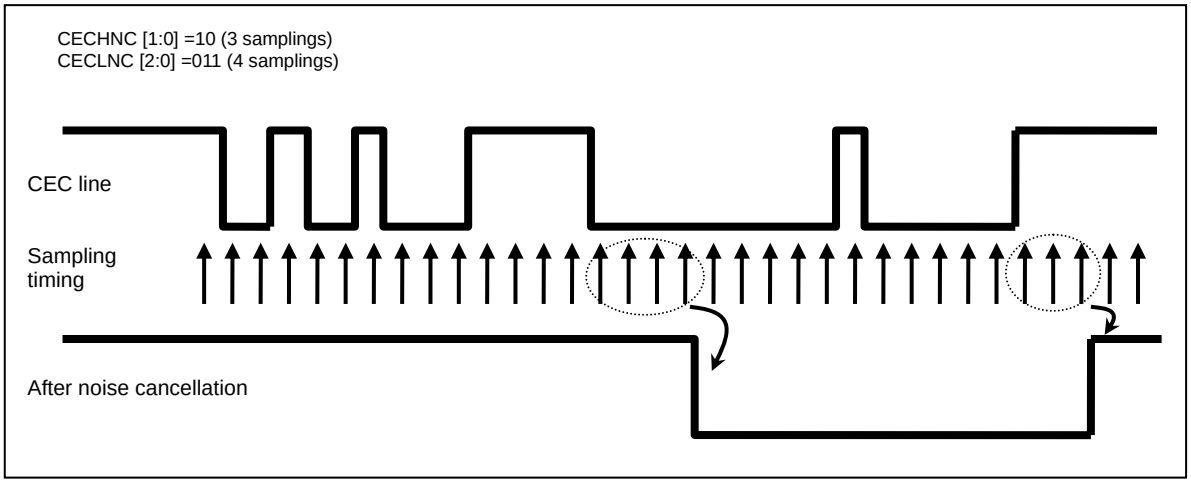
(Note) A broadcast message is received regardless of the CECADD register setting. By allocating a logical address of a device to 15, logical "0" is sent as an ACK response to the broadcast message.

(2) Noise Cancellation Time

The noise cancellation time is configurable with the <CECHNC1:0><CECLNC2:0> bits of the CECRCR1 register. You can configure the time to detect "1" and "0" respectively.
It is considered as noise if "1"s or "0"s of the same number as the specified value are not sampled.

A CEC line is monitored at each rising edge of a sampling clock. In the case that the CEC line is changed from "1" to "0", the change is fully recognized if "0"s of the same number as specified in the <CECLNC> bit are monitored. In the case that the CEC line is changed from "0" to "1", the change is fully recognized if "1"s of the same number as specified in the <CECHNC> bit are sampled.

The following illustrates the operation of a case that a noise cancelling is configured as <CECHNC 1:0>=10 (3 samplings) and <CECLNC 2:0>=011 (4 samplings). By cancelling the noise, a signal "1" shifts to "0" after "0" is sampled four times. The signal "0" shifts to "1" after "1" is sampled three times.



(3) Cycle Error

Configure the CECRCR1 <CECMIN2:0> <CECMAX2:0> bits to detect a cycle error.

You can specify the time to detect a cycle error for each sampling clock cycle between the ranges of -4 to +3 cycles from the maximum or minimum time set in the CEC standard.

Detecting an error during data reception causes an error interrupt, and CEC waits for the next start bit. The received data is discarded.

(4) Point of Determining Data

Configure the CECRCR1 <CECDAT> bit for the point of determining the data as “0” or “1”.

You can specify it per two sampling clock cycles between the ranges of + or - 6 cycles with approx. 1.05 ms from the bit start point.

(5) ACK Response

Configuring the CECRCR1 <CECACKDIS> bit enables you to specify if logical “0” is sent or not as an ACK response to the data block when destination address corresponds with the address set in the logical address register.

The header block sends logical “0” as an ACK response regardless of the bit setting when detecting the addresses corresponding.

(6) Receive Error Interrupt Suspend

Configure the CECRCR1 <CECRIHLD> bit to specify if a receive error interrupt (maximum cycle error, buffer overrun and waveform error) is suspended or not.
Setting “1” generates no interrupt at the error detection. If data continues to the ACK bit, an ACK response is executed by a reversed logic. If the subsequent bits are interrupted, it is determined as a timeout, based on the setting in <CECTOUT> of the CECRCR1 register. After the ACK response or the timeout determination, an interrupt is generated.

(7) Cycles to Identify Timeout

Configure the CECRCR１<CECTOUT> bit to specify the time to determine a timeout.
This is used when the setting of a receive error interrupt suspension, which is specified in CECRCR1 <CECRIHLD>, is valid.

(8) Data Reception at Logical Address Discrepancy

By setting CECRCR1 <CECOTH>, you can specify if data is received or not when destination address does not correspond with the address set in the logical address register.

In this case, data is received as usual, and an interrupt is generated by detecting an error. However, an ACK response of neither the header block nor the data block is sent.

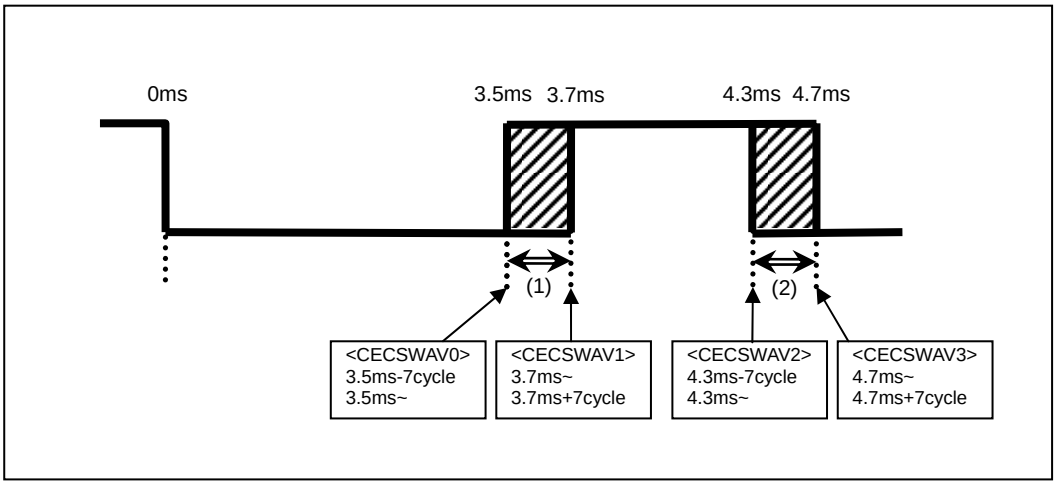
(Note 1)	**A broadcast message is received regardless of the <CECOTH> register setting.**
(Note 2)	**If the initiator sends a new message beginning with the start bit without having sent the last block with EOM="1", a maximum cycle error is determined for the ACK bit and an interrupt is generated. Then, the receive operation is performed in the usual way.**

(9) Start Bit Detection

Configuring the CECRCR2 register allows you to specify the rising timing and a cycle of the start bit detection respectively.

<CECSWAV0> is to specify the fastest start bit rising timing. <CECSWAV1> is to specify the latest start bit rising timing ((1) in the figure shown below).
<CECSWAV2> is to specify the minimum cycle of a start bit. <CECSWAV3> is to specify the maximum cycle of a start bit ((2) in the figure shown below).

If a rising edge during the period (1) and a falling edge during the period (2) are detected, the start bit is considered to be valid.



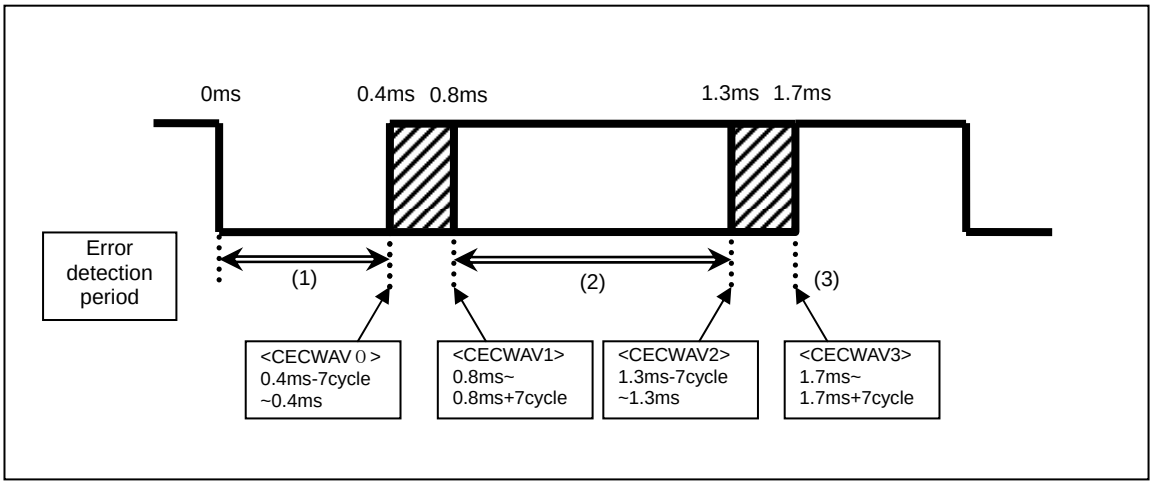
(10) Waveform Error Detection

To detect an error when a received waveform is out of the defined tolerance range, configure the CECRCR3 register.

An error is detected when the <CECWAVEN> bit of the CECRCR3 register is enabled. You can specify the detection time in the <CECWAV0> <CECWAV1> <CECWAV2> <CECWAV3> bits.

If the rising edge is detected during the period (1) or (2) shown below, or not detected in the timing described in (3), a waveform error interrupt is generated.

(1) A period between the beginning of a bit and the fastest logical “1” rising timing.
(2) A period between the latest logical “1” rising timing and the fastest logical “0” rising timing.
(3) The latest logical “0” rising timing.



13.3.1.4 Enabling Reception

After configuring the CECADD, CECRCR1, CECRCR2 and CECRCR3 registers, CEC is ready for reception by enabling the CECREN <CECREN> bit. Detecting a start bit initiates the reception.

(Note) Changing the configurations of the CECADD, CECRCR1, CECRCR2 and CECRCR3 registers during transmission or reception may harm its proper operation. Before the change of the registers shown below, set the CECREN <CECREN> bit to disable the reception and read the <CECREN> bit and the CECTEN <CECTEN> bit to ensure that the operation is stopped.

CECADD	**<CECADD15:0>**	**Logical address**
CECRCR1	**<CECHNC><CECLNC>**	**Noise cancellation time**
	<CECMIN><CECMAX>	**Time to identify cycle error**
	<CECOTH>	**Data reception at logical address discrepancy**
CECRCR2	**<CECSWAV0><CECSWAV1> <CECSWAV2><CECSWAV3>**	**Start bit detection**
CECRCR3	**<CECWAV0><CECWAV1> <CECWAV2><CECWAV3>**	**Waveform error detection (when enabled)**

13.3.1.5 Reception

After detecting a start bit, a start bit interrupt is generated, and the CECRSTAT <CECRISTA> bit is set.

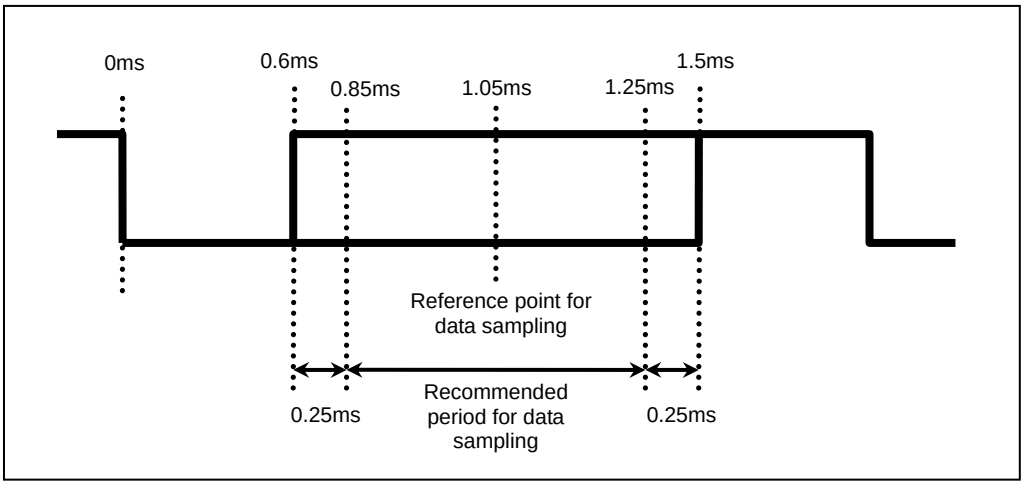
Upon receiving a byte of data, the EOM and ACK bits, they are stored in the CECRBUF register. A receive interrupt is generated and it causes the CECRSTAT <CECRIEND> bit to be set. Same as the other data, the ACK bit that monitored the CEC line is stored instead of the one generated in the CEC circuit.

The reception continues from the first data block until the final data block that has the EOM bit indicating “1”. After detecting the final data block, CEC waits for a next start bit.

13.3.1.6 Data Sampling Point

The figure shown below illustrates a data sampling timing.

With the CECRCR1 <CECDAT> bit, you can specify a data sampling point per two sampling clock cycles between the ranges of + or - 6 cycles from a reference point (approx. 1.05 ms).



13.3.1.7 ACK Response

Setting the CECRCR1 <CECACKDIS> bit enables to specify if logical “0” is sent or not as an ACK response to the data block when destination address corresponds with the address set in the logical address register. The header block sends logical “0” as an ACK response regardless of the bit setting when detecting the addresses corresponding.

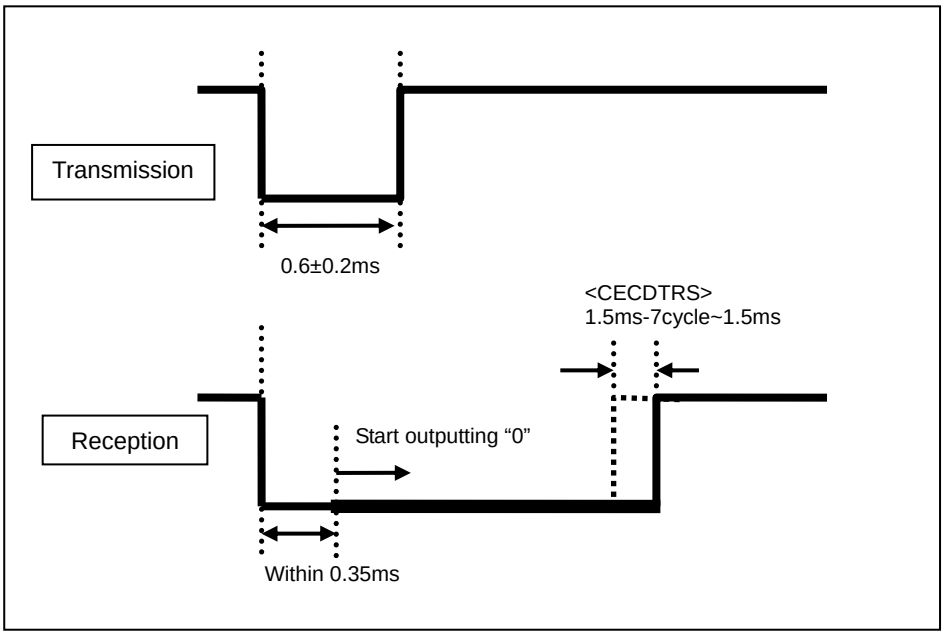
The following lists the ACK responses.
“Yes” indicates that CEC outputs “0” as a response to the ACK signal from a transmission device (ACK bit: logical “0”). “No” indicates that CEC does not output “0” as a response to the ACK signal from a transmission device (ACK bit: logical “1”).

Register setting		Header block address		Data block address	
		Conformity	Discrepancy	Conformity	Discrepancy
CECRCR1 <CECACKDIS>	“0” (responding logical “0”)	Yes	No	Yes	No
	“1” （not responding logical “0”）			No	No

The following describes the ACK response timing.

“0” is output within 0.35 ms from detecting the falling edge of the ACK bit output from the transmission device. The timing to stop output is the same as that of outputting logical “0” for transmission. The timing can be specified with the CECTCR <CECDTRS2:0> bit.

(Reference) The configuration of <CECDTRS2:0> is applied for transmission of the data bits and the EOM bit.



13.3.1.8 Detecting Error Interrupt

Detecting an error during data reception causes an error interrupt, and CEC waits for the next start bit. The received data is discarded.

It is possible to suspend a receive error interrupt (maximum cycle error, receive buffer overrun and waveform error), continue reception and send the reversed ACK response.

You can check the interrupt factor by monitoring the bit of the CECRSTAT register corresponding to the interrupt.

13.3.1.9 Details of Receive Error

(1) Cycle Error

Period between the falling edges of the two sequential bits is measured during reception. If the period does not comply with the specified minimum or maximum value, a cycle error interrupt is generated.

The maximum and minimum cycles are specified in the CECRCR1 <CECMIN2:0> <CECMAX2:0> bits. A cycle error can be detected for each sampling clock cycle between the ranges of -4 to +3 cycles from the minimum value (approx. 2.05 ms) or the maximum value (approx. 2.75 ms) defined by the CEC standard.

The CECRSTAT <CECRIMIN> bit or the <CECRIMAX> bit is set if a cycle error interrupt is generated.
The minimum cycle error causes CEC to output “0” for approx. 3.6 ms.

(Note) When CECRCR1<CECOTH>=“1” (to enable data reception at logical address discrepancy), if the initiator sends a new message beginning with the start bit without having sent the last block with EOM=“1”, a maximum cycle error is determined for the ACK bit and an interrupt is generated. Then, the receive operation is performed in the usual way.

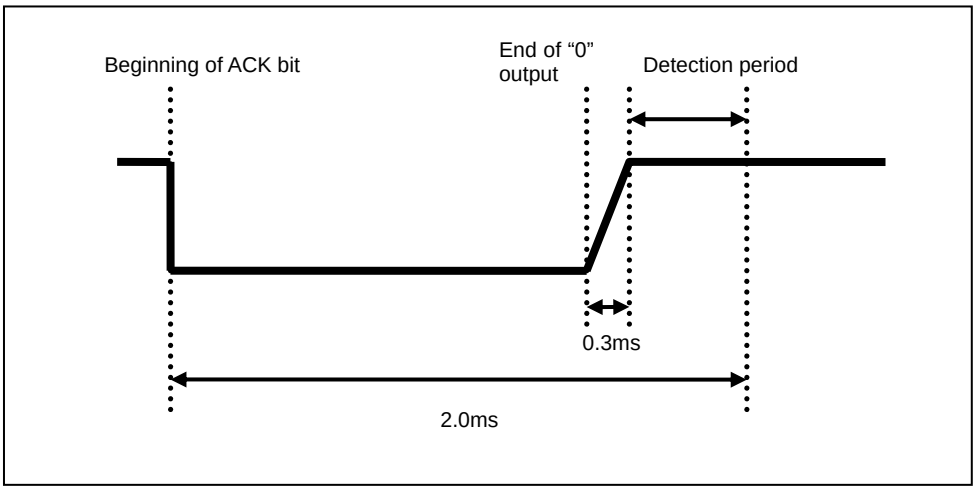
(2) ACK Collision

At an ACK response, detecting “0” after the specified period to output generates an ACK collision interrupt or a minimum cycle error interrupt.
The ACK collision interrupt sets the CECRSTAT <CECRIACK> bit. The minimum cycle error interrupt sets the CECRSTAT <CECRIMIN> bit.

The following describes the period and method of detection.

Detection starts approx. 0.3 ms after the end of the period of outputting “0” and ends approx 2.0 ms from the starting point (the falling edge) of the ACK bit.

At 0.3 ms from the end of the period of outputting “0”, CEC checks if the CEC line is “0” or not. If it is “0”, an ACK collision interrupt is generated. If it is “1”, and “0” is detected during the detection period, the minimum cycle error interrupt is generated. The minimum cycle error causes CEC to output “0” for approx. 3.6 ms.



(3) Receive Buffer Overrun

A receive buffer overrun interrupt is generated when the next data reception is completed before reading the data stored in the receive buffer.
The interrupt sets the CECRSTAT <CECRIOR> bit.

(4) Waveform Error

A waveform error occurs when waveform error detection is enabled in CECRCR3. Detecting a waveform, which does not identical to the defined, results in the waveform error. The interrupt is generated.
The interrupt sets the CECRSTAT <CECRIWAV> bit.

(5) Suspending Receive Error Interrupt

You can specify if a maximum cycle error, a buffer overrun and a waveform error are suspended or not without generating an interrupt at error detection. This can be set in the CECRCR1 <CECRIHLD> bit. To enable the setting, a timeout setting with the CECRCR1 <CECTOUT> bit is required.

Under suspend-enable condition, if CEC keeps receiving the next bit and the entire reception including the ACK bit is completed, CEC generates an interrupt after a reversed ACK response is executed. “1” is set to the bits of the CECRSTAT register: the <CECRIEND> bit that indicates the reception completion, and the bits corresponding to the detected errors.

If the reception of the next bit is interrupted, CEC starts to measure the timeout period, and an interrupt is generated after the timeout. “1” is set to the bits of the CECRSTAT register corresponding to the detected error.
The timeout is measured from the end of the last bit received as is the case with wait time of a bus to be free in transmission.

The information that the interrupts are suspended is held until the EOM bit is received or the timeout occurs. Thus, an interrupt is generated in each reception of a byte of data if multiple bytes are received while interrupts are suspended. “1” is set to the bits of the CECRSTAT register: the <CECRIEND> bit that indicates the reception completion, and the bits corresponding to the detected errors. The flags of the suspended interrupts and the reception completion are set to the bits of the CECRSTAT register.

(Note 1)	**A minimum cycle error interrupt is generated upon detecting a minimum cycle error in the next received bit while interrupts are suspended. “0” is output to CEC for approx. 3.6 ms.** **The flags of the suspended interrupts and the minimum cycle error are set to the bits of the CECRSTAT register.**
(Note 2)	**If an interrupt other than a minimum cycle error interrupt is generated while interrupts are suspended, CEC continues reception until the ACK response or the timeout.** **All the flags of the detected interrupts are set to the bits of the CECRSTAT register.**

13.3.1.10 Stopping Reception

Writing “0” to the CECREN <CECREN> bit disables data reception. The reception is stopped upon disabling the bit during reception. The received data is discarded.

(Note)	**If the reception is disabled while “0” is sent as a signal of minimum cycle error, the “0” output is stopped as well.**

13.3.2 Transmission

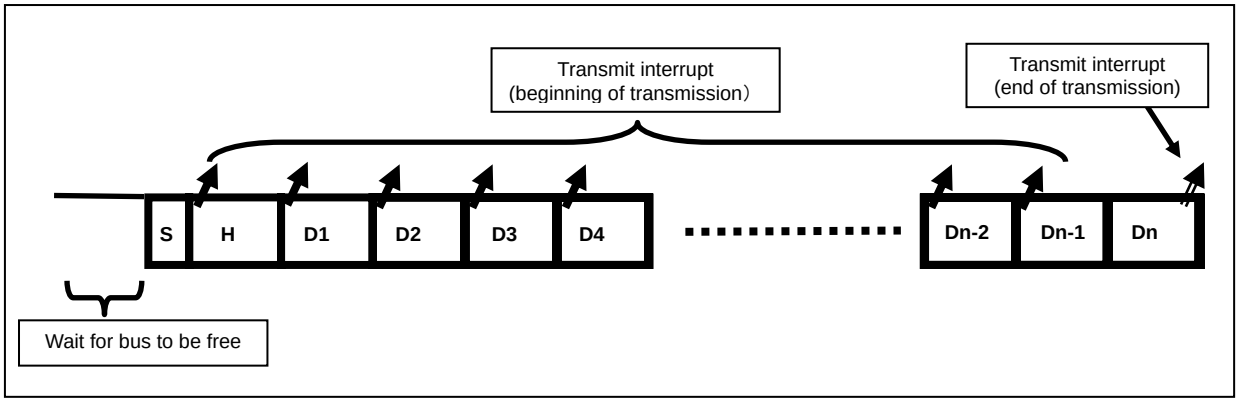
13.3.2.1 Basic Operation

Configure a start bit of transmission after configuring the data buffer. It enables the start bit to be transmitted after time that a bus is free is properly maintained.

Transmitting the first data bit subsequent to the start bit generates a transfer interrupt. It indicates that the next data can be set to a transmit buffer. The ACK bit is sent after a byte of data (8 bit) and the EOM bit are transmitted, and then the ACK response is detected.

The data transmission per byte continues until the data including the EOM bit that indicates “1” is stored in the transmit buffer. If its transmission is completed, a transmit completion interrupt is generated.

If an error is generated during transmission, an error interrupt is generates to stop transmission.
Even if reception is enabled, no reception is executed during transmission.



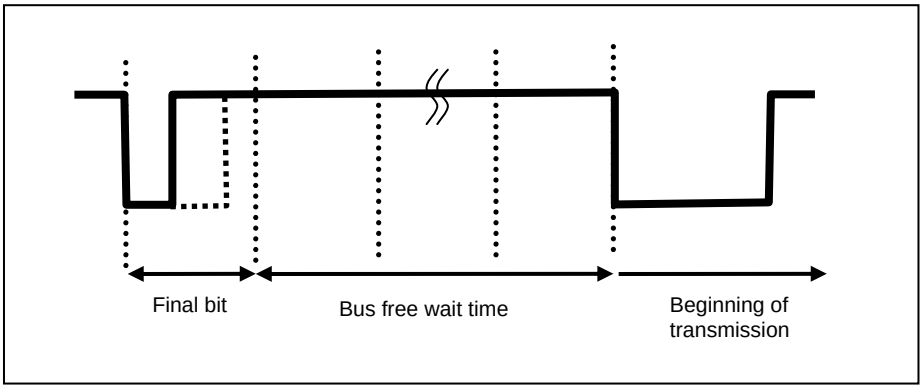
13.3.2.2 Preconfiguration

Before transmitting data, transmission settings to the Transmit Control Register CECTCR>and the transmit buffer <CECTBUF> are required.

(1) Bus Free Wait Time

Specify the bus free wait time in the CECTCR<CECFREE> bits. It can be specified in a range of 1 to 16 bit cycles.

Counting of the bus free wait time begins one bit cycle after the falling edge of the final bit. If the signal stays high for the specified number of bit cycles, transmission starts.



(2) Transmitting Broadcast Message

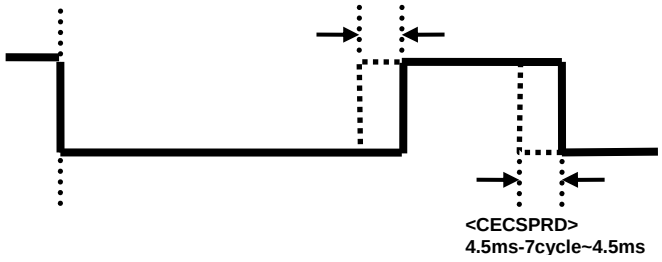
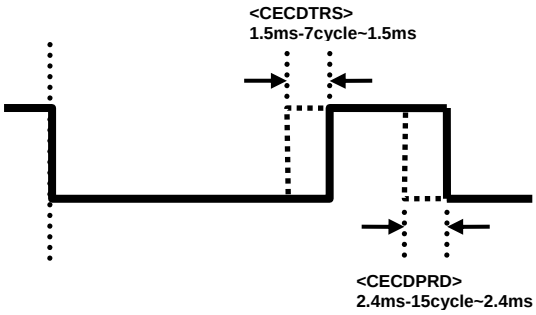
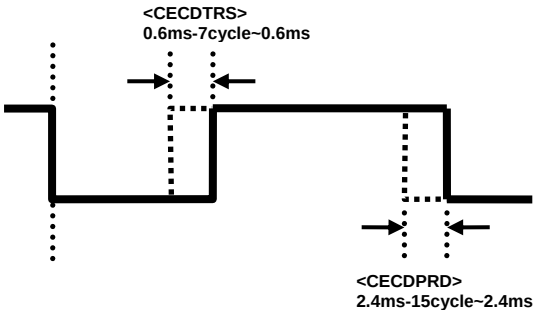
Set the CECTCR <CECBRD> bit when transmitting a broadcast message. If this bit is set, “0” response during an ACK cycle results in an error. If not, “1” response during an ACK cycle results in an error.

(3) Adjusting Transmission Waveform

Both start bit and data bit are capable of adjusting the rising timing and cycle. With the CECTCR <CECSTRS> <CECSPRD> <CECDTRS> <CECDPRD> bits, the timing can be specified between the defined fastest rising/cycle timing and the reference value.

The following figures show how the waveforms differ according to the configurations of the start bit, logical “0” and logical “1”.

(Reference) The configuration of <CECDTRS> is applied for waveform of an ACK response during reception. The ACK response and the logical “0” output show the same waveform.

Start bit	
Logical “0”	
Logical “1”	

(4) Preparing Transmission Data

Configure a byte of transmission data and EOM data with the CECTBUF register.

13.3.2.3 Starting Transmission

Transmission is ready by setting the CECTEN <CECTEN> bit to start transmission after setting the CECTCR and CECTBUF registers.

The <CECTEN> bit is never cleared to “0” until a transmit completion interrupt or an error interrupt occurs. Thus you don’t need to set this bit for each a byte of data transmission.

(Note)	**Changing the configurations of the CECTCR register during transmission or reception may harm its proper operation. Be careful if you change it during transmission.**

13.3.2.4 Transmission

Next to the setting for starting transmission, CEC checks if a bus is free. If “1” is sampled as a CEC line for specified bit cycles, start bit is transmitted. The CEC always checks if bus is free. Transmission starts anytime if bus is free for specified bit cycles.

After transmitting a start bit, a byte of data and the EOM data that are set in the buffer are sent to the shift register, and data transmission is started. When CEC starts transmitting the first bit of a byte of data, a transmit interrupt is generated. It sets the CECTSTAT <CECTISTA> bit. Subsequent to the transmit interrupt, a byte of next data can be set to the transmit buffer.

Then 8 bit data, the EOM bit and the ACK bit are transmitted, and the ACK response is checked. This is the end of a byte of data transmission.
Data transfer continues in the above sequence until “1” is set to the EOM bit.

If “1” is set to the EOM, a transmit completion interrupt is generated subsequent to the ACK bit check as described above. Generation of this interrupt, which means the end of a sequence of transmission operation, sets the CECTSTAT <CECTIEND> bit, and clears the CECTEN <CECTEN> bit.

13.3.2.5 ACK Transmission and ACK Error Criterion

A criterion of the ACK error differs depending on the CECTCR <CECBRD> bit.
If this bit is set, broadcast message is transmitted, and the ACK response of logical “0” is determined as an error. If this bit is not set, the ACK response of logical “1” is determined as an error.

13.3.2.6 Detecting Transmission Error

Error detection during transmission generates an interrupt and stops transmission. It clears the CECTEN <CECTEN> bit.

To identify an error factor, the CECTSTAT register has bits that correspond with each interrupt. You can identify the interrupt factor by checking these bits.

(Note) An attempt to stop transmission by an error may cause an improper waveform output to CEC. This is because output is stopped immediately after the error occurs.

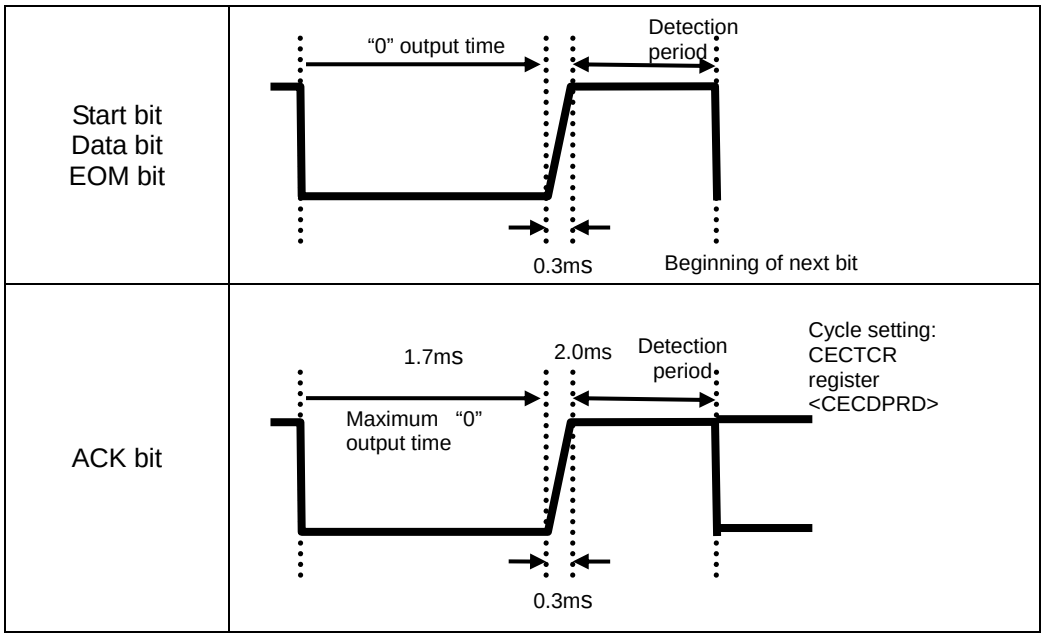
13.3.2.7 Details of Transmission Error

(1) Arbitration Lost

An arbitration lost error occurs when CEC detects “0” on completion of appropriate low duration.

Detecting an arbitration lost error sets the CECTSTAT <CECTIAL> bit.

Two types of the arbitration lost detection periods are shown below.



(2) ACK error

An ACK error interrupt occurs when an ACK response does not conform to the configuration specified in the CECTCR <CECBRD> bit.
When the ACK error interrupt occurs, the CECTSTAT <CECTIACK> bit is set.

The ACK error is detected in the following cases.

Configuration	Determined as an ACK error when
<CECBRD>=0 Broadcast transmission?: No	ACK response is logical “1”
<CECBRD>=1 Broadcast transmission?: Yes	ACK response is logical “0”

(3) Transmit Buffer Underrun

A transmit buffer underrun error is caused by the following sequence.
1. Data in the transmit buffer is transmit to the shift register.
2. An interrupt occurs.
3. A byte of data is transmitted.
4. No data is set to the transmit buffer before starting transmission of a byte of subsequent data.

When an underrun error occurs, the CECTSTAT <CECTIUR> bit is set.

(4) Order of ACK Error and Transmit Buffer Overrun

If interrupt factors of the ACK error and transmit buffer underrun are detected at the end of transmission of a byte of data, the transmit buffer underrun has priority. The transmit buffer underrun interrupt occurs first and then the ACK error interrupt occurs.

13.3.2.8 Stopping Transmission

To stop transmission, send data including the EOM bit that indicates “1”. This generates a transmit completion interrupt.
Please note that proper operation is not ensured if the start bit of transmission is set to “0” during transmission.

13.3.2.9 Retransmission

Transmission is stopped by error detection. To retry the transmission, configure the condition and data of starting the transmission.

13.3.3 Software Reset

The entire CEC function can be initialized by software.

Setting “1” to the software reset register CECRESET <CECRESET> bit causes the following operations.

- Reception : Immediately stops. The received data is discarded.
- Transmission : Immediately stops including output to the CEC line.
- Register : All the registers other than CECEN are initialized.

Please note that software reset during transmission may cause the CEC line waveform that does not identical to the defined.

14 Remote control signal preprocessor (RMC)

14.1 Basic operation

Remote control signal preprocessor (hereafter referred to as RMC) receives a remote control signal of which carrier is removed.

14.1.1 Reception of Remote Control Signal

- Sampled by 32KHz clock
- Noise canceller
- Leader detection
- Batch reception up to 72bit of data

14.2 Registers

14.2.1 Register Map

Addresses and names of RMC control registers are shown below.

Register		Address	
		Channel 0	Channel 1
Remote Control Enable Register	RMCxEN	0x4004_0400	0x4004_0440
Remote Control Receive Enable Register	RMCxREN	0x4004_0404	0x4004_0444
Remote Control Receive Data Buffer Register 1	RMCxRBUF1	0x4004_0408	0x4004_0448
Remote Control Receive Data Buffer Register 2	RMCxRBUF2	0x4004_040C	0x4004_044C
Remote Control Receive Data Buffer Register 3	RMCxRBUF3	0x4004_0410	0x4004_0450
Remote Control Receive Control Register 1	RMCxRCR1	0x4004_0414	0x4004_0454
Remote Control Receive Control Register 2	RMCxRCR2	0x4004_0418	0x4004_0458
Remote Control Receive Control Register 3	RMCxRCR3	0x4004_041C	0x4004_045C
Remote Control Receive Control Register 4	RMCxRCR4	0x4004_0420	0x4004_0460
Remote Control Receive Status Register	RMCxRSTAT	0x4004_0424	0x4004_0464

14.2.2 Remote Control Enable Register [RMCEN]

	7	6	5	4	3	2	1	0
bit Symbol	—						I2RMC	RMCEN
Read/Write	R						R/W	R/W
After reset	0						0	0
Function	”0” is read.						RMC in IDLE mode 0: Disabled 1: Enabled	RMC operation 0: Disabled 1: Enabled

<I2RMC>: Controls RMC operation in IDLE mode.
Set “1” to enable RMC in IDLE Mode.
This bit and the <RMCEN> bit can be set simultaneously.

<RMCEN>: Controls RMC operation.
To allow RMC to function, enable the RMCEN bit first. If the operation is disabled, all the clocks for RMC except for the enable register are stopped, and it can reduce power consumption.
If RMC is enabled and then disabled, the settings in each register remain intact.

14.2.3 Remote Control Receive Enable Register [RMCREN]

	7	6	5	4	3	2	1	0
bit Symbol	—							RMCREN
Read/Write	R							R/W
After reset	0							0
Function	”0” is read.							Reception 0: Disabled 1: Enabled

<RMCREN>: Controls reception of RMC.
Setting this bit to “1” enables reception.

(Note) Enable the <RMCREN> bit after setting the RMCxRCR1, RMCxRCR2 and RMCxRCR3.

14.2.4 Remote Control Receive Data Buffer Register 1 [RMCRBUF1]

	31	30	29	28	27	26	25	24
bit Symbol	RMCRBUF 31	RMCRBUF 30	RMCRBUF 29	RMCRBUF 28	RMCRBUF 27	RMCRBUF 26	RMCRBUF 25	RMCRBUF 24
Read/Write	R							
After reset	0							
Function	Received data							

	23	22	21	20	19	18	17	16
bit Symbol	RMCRBUF 23	RMCRBUF 22	RMCRBUF 21	RMCRBUF 20	RMCRBUF 19	RMCRBUF 18	RMCRBUF 17	RMCRBUF 16
Read/Write	R							
After reset	0							
Function	Received data							

	15	14	13	12	11	10	9	8
bit Symbol	RMCRBUF 15	RMCRBUF 14	RMCRBUF 13	RMCRBUF 12	RMCRBUF 11	RMCRBUF 10	RMCRBUF 9	RMCRBUF 8
Read/Write	R							
After reset	0							
Function	Received data							

	7	6	5	4	3	2	1	0
bit Symbol	RMCRBUF 7	RMCRBUF 6	RMCRBUF 5	RMCRBUF 4	RMCRBUF 3	RMCRBUF 2	RMCRBUF 1	RMCRBUF 0
Read/Write	R							
After reset	0							
Function	Received data							

<RMCRBUF31:0>: Reads 4 bytes of received data.

14.2.5 Remote Control Receive Data Buffer Register 2 [RMCRBUF2]

	31	30	29	28	27	26	25	24
bit Symbol	RMCRBUF 63	RMCRBUF 62	RMCRBUF 61	RMCRBUF 60	RMCRBUF 59	RMCRBUF 58	RMCRBUF 57	RMCRBUF 56
Read/Write	R							
After reset	0							
Function	Received data							
	23	22	21	20	19	18	17	16
bit Symbol	RMCRBUF 55	RMCRBUF 54	RMCRBUF 53	RMCRBUF 54	RMCRBUF 53	RMCRBUF 52	RMCRBUF 51	RMCRBUF 50
Read/Write	R							
After reset	0							
Function	Received data							
	15	14	13	12	11	10	9	8
bit Symbol	RMCRBUF 47	RMCRBUF 46	RMCRBUF 45	RMCRBUF 44	RMCRBUF 43	RMCRBUF 42	RMCRBUF 41	RMCRBUF 40
Read/Write	R							
After reset	0							
Function	Received data							
	7	6	5	4	3	2	1	0
bit Symbol	RMCRBUF 39	RMCRBUF 38	RMCRBUF 37	RMCRBUF 36	RMCRBUF 35	RMCRBUF 34	RMCRBUF 33	RMCRBUF 32
Read/Write	R							
After reset	0							
Function	Received data							

<RMCRBUF63:32>: Reads 4 bytes of received data.

14.2.6 Remote Control Receive Data Buffer Register 3 [RMCRBUF3]

	7	6	5	4	3	2	1	0
bit Symbol	RMCRBUF 71	RMCRBUF 70	RMCRBUF 69	RMCRBUF 68	RMCRBUF 67	RMCRBUF 66	RMCRBUF 65	RMCRBUF 64
Read/Write	R							
After reset	0							
Function	Received data							

<RMCRBUF71:64>: Reads a byte of received data.

(Note 1) Received data is stored from RMCRBUF1 <RMCRBUF0> to RMCRBUF3 <RMCRBUF71> in sequence.

(Note 2) The first received bit is stored in the MSB. The last received bit is stored in the LSB (bit 0).

If the remote control signal is received in the LSB first algorithm, the received data is stored in reverse sequence.

14.2.7 Remote Control Receive Control Register 1 [RMCRCR1]

	31	30	29	28	27	26	25	24
bit Symbol	RMCLC MAX7	RMCLC MAX6	RMCLC MAX5	RMCLC MAX4	RMCLC MAX3	RMCLC MAX2	RMCLC MAX1	RMCLC MAX0
Read/Write	R/W							
After reset	0							
Function	Maximum cycle of leader detection: RMCLCMAX×4/fs[s]							

	23	22	21	20	19	18	17	16
bit Symbol	RMCLC MIN7	RMCLC MIN6	RMCLC MIN5	RMCLC MIN4	RMCLC MIN3	RMCLC MIN2	RMCLC MIN1	RMCLC MIN0
Read/Write	R/W							
After reset	0							
Function	Minimum cycle of leader detection: RMCLCMIN×4/fs[s]							

	15	14	13	12	11	10	9	8
bit Symbol	RMCLL MAX7	RMCLL MAX6	RMCLL MAX5	RMCLL MAX4	RMCLL MAX3	RMCLL MAX2	RMCLL MAX1	RMCLL MAX0
Read/Write	R/W							
After reset	0							
Function	Maximum low width of leader detection: RMCLLMAX×4/fs[s]							

	7	6	5	4	3	2	1	0
bit Symbol	RMCLL MIN7	RMCLL MIN6	RMCLL MIN5	RMCLL MIN4	RMCLL MIN3	RMCLL MIN2	RMCLL MIN1	RMCLL MIN0
Read/Write	R/W							
After reset	0							
Function	Minimum low width of leader detection: RMCLLMIN×4/fs[s]							

<RMCLCMAX7:0>: Specifies a maximum cycle of leader detection.
Calculating formula of the maximum cycle: RMCLCMAX×4/fs[s].
RMC detects the first cycle as a leader if it is within the maximum cycle.

<RMCLCMIN7:0>: Specifies a minimum cycle of leader detection.
Calculating formula of the minimum cycle: RMCLCMIN×4/fs[s].
RMC detects the first cycle as a leader if it exceeds the minimum cycle.

<RMCLLMAX7:0>: Specifies a maximum low width of leader detection.
Calculating formula of the maximum low width: RMCLLMAX×4/fs[s]
RMC detects the first cycle as a leader if its low width is within the maximum low width.

<RMCLLMIN7:0>: Specifies a minimum low width of leader detection.
Calculating formula of the minimum low width: RMCLLMIN×4/fs[s]
RMC detects the first cycle as a leader if its low width exceeds the minimum low width.
If RMCRCR2<RMCLD> = 1, a value less than the specified is determined as data.

(Note) When you configure the register, you must follow the rule shown below.

Leader	Rules
Low width + high width	<RMCLCMAX7:0> > <RMCLCMIN7:0> <RMCLLMAX7:0> > <RMCLLMIN7:0> <RMCLCMIN7:0> > <RMCLLMAX7:0>
Only with high width	<RMCLCMAX7:0> > <RMCLCMIN7:0> <RMCLLMAX7:0> = 0x00000000 <RMCLLMIN7:0> = don’t care
No leader	<RMCLCMAX7:0> = 0x00000000 <RMCLCMIN7:0> = don’t care <RMCLLMAX7:0> = don’t care <RMCLLMIN7:0> = don’t care

14.2.8 Remote Control Receive Control Register 2 [RMCRCR2]

	31	30	29	28	27	26	25	24
bit Symbol	RMCLIEN	RMCEDIEN	—	—	—	—	RMCLD	RMCPHM
Read/Write	R/W	R/W	R				R/W	R/W
After reset	0	0	0				0	0
Function	Leader detection interrupt 0: Not generated 1 : Generated	Remote control input falling edge interrupt 0: Not generated 1 : Generated	”0” is read.				Receiving remote control signal with or without leader 0: Disabled 1: Enabled	Receive a remote control signal in phase method? 0: No (receive in cycle method) 1:Yes

	23	22	21	20	19	18	17	16
bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R							
After reset	0							
Function	”0” is read.							

	15	14	13	12	11	10	9	8
bit Symbol	RMCLL7	RMCLL6	RMCLL5	RMCLL4	RMCLL3	RMCLL2	RMCLL1	RMCLL0
Read/Write	R/W							
After reset	1							
Function	Excess low width that triggers reception completion and interrupt generation 00000000~11111110:RMCLL×1/fs[s] 11111111:not to use as the trigger							

	7	6	5	4	3	2	1	0
bit Symbol	RMCDMAX7	RMCDMAX6	RMCDMAX5	RMCDMAX4	RMCDMAX3	RMCDMAX2	RMCDMAX1	RMCDMAX0
Read/Write	R/W							
After reset	1							
Function	Maximum data bit cycle that triggers reception completion and interrupt generation 00000000~11111110:RMCDMAX×1/fs[s] 11111111: not to use as the trigger							

<RMCLIEN>: Enables to generate a leader detection interrupt by detecting a leader.

<RMCEDIEN>: Enables to generate a remote control input falling edge Interrupt.

<RMCLD>: Enables RMC to receive signals with or without a leader.

<RMCPHM>: Specifies data reception mode of a phase method. If you use the phase method of which signal cycle is fixed, set “1”.

<RMCLL7:0>: Specifies an excess low width. If an excess low width is detected, reception is completed and an interrupt is generated. The low width is not detected if <RMCLL7:0> = 11111111b. Calculating formula of an excess low width: RMCLLx1/fs[s].

<RMCDMAX7:0>: Specifies a threshold for detecting a maximum data bit cycle. It is detected when a data bit cycle exceeds the threshold. It is not detected when <RMCMAX7:0> = 11111111b. Calculating formula of the threshold: RMCDMAX x 1/fs[s].

14.2.9 Remote Control Receive Control Register 3 [RMCRCR3]

	15	14	13	12	11	10	9	8
bit Symbol	—	RMCDAT H6	RMCDAT H5	RMCDAT H4	RMCDAT H3	RMCDAT H2	RMCDAT H1	RMCDAT H0
Read/Write	R	W						
After reset	0	0						
Function	”0” is read.	Larger threshold to determine a signal pattern in a phase method RMCDATH×1/fs[s]						

	7	6	5	4	3	2	1	0
bit Symbol	—	RMCDAT L6	RMCDAT L5	RMCDAT L4	RMCDAT L3	RMCDAT L2	RMCDAT L1	RMCDAT L0
Read/Write	R	W						
After reset	0	0						
Function	”0” is read.	Threshold to determine 0 or 1/ smaller threshold to determine a signal pattern in a phase method RMCDATL×1/fs[s]						

<RMCDATH6:0>: Specifies a larger threshold (within a range of 1.5T and 2T) to determine a pattern of remote control signal in a phase method. If the measured cycle exceeds the threshold, the bit is determined as “10”. If not, the bit is determined as “01”. Calculating formula of the threshold: RMCDATHx1/fs[s].

<RMCDATL6:0>: Specifies two kinds of thresholds: a threshold to determine whether a data bit is 0 or 1; a smaller threshold (within a range of 1T and 1.5T) to determine a pattern of remote control signal in a phase method.
As for the determination of data bit, if the measured cycle exceeds the threshold, the bit is determined as “1”. If not, the bit is determined as “0”. Calculating formula of the threshold: RMCDATL×1/fs[s].
As for the determination of a remote control signal pattern in a phase method, if the measured cycle exceeds the threshold, the bit is determined as “01”. If not, the bit is determined as “00”. Calculating formula of the threshold to determine 0 or 1: RMCDATLx1/fs[s].

(Note) If the <RMCPHM> bit of the Remote Control Receive Control Register 2 is “0”, <RMCDATH6:0> are not enabled. The bits are enabled when <RMCPHM> is ”1”.

14.2.10 Remote Control Receive Control Register 4 [RMCRCR4]

	7	6	5	4	3	2	1	0
bit Symbol	RMCPO	—	—	—	RMCNC3	RMCNC2	RMCNC1	RMCNC0
Read/Write	R/W	R			R/W			
After reset	0	0			0			
Function	Remote control input signal 0: Not reversed 1: Reversed	”0” is read.			Noise cancellation time 0000: No cancellation 0001~1111:RMCNC×1/fs[s]			

<RMCPO>: Specifies whether a remote control input signal is reversed or not.

<RMCNC3:0>: Specifies time noises are cancelled by a noise canceller. If <RMCNC3:0> = 0000b, noises are not cancelled. Calculating formula of noise cancellation time: RMCNC x 1/fs[s].

14.2.11 Remote Control Receive Status Register [RMCRSTAT]

	15	14	13	12	11	10	9	8
bit Symbol	RMCRLIF	RMCLOIF	RMCDMAXIF	RMCEDIF	—	—	—	—
Read/Write	R	R	R	R	R			
After reset	0	0	0	0	0			
	Leader detection is interrupt factor? 0: No 1: Yes	Low width detection is interrupt factor? 0: No 1: Yes	Maximum data bit cycle detection is interrupt factor? 0: No 1: Yes	Remote control input falling edge interrupt is interrupt factor? 0: No 1: Yes	”0” is read.			
	7	6	5	4	3	2	1	0
bit Symbol	RMCRLDR	RMCRNUM6	RMCRNUM5	RMCRNUM4	RMCRNUM3	RMCRNUM2	RMCRNUM1	RMCRNUM0
Read/Write	R	R						
After reset	0	0						
Function	Leader detection 0: No 1: Yes	The number of received data bit 0000000:no data bit (only with leader) 0000001~1001000:1~72bit 1001001~1111111:73bit and more						

<RMCRLIF>: Indicates that leader detection is the interrupt factor.

<RMCLOIF>: Indicates that low width detection is the interrupt factor.

<RMCDMAXIF>: Indicates that maximum data bit cycle detection is the interrupt factor.

<RMCEDIF>: Indicates that a remote control input falling edge interrupt is the interrupt factor.

<RMCRLDR>: Detects a leader of a received remote control signal

<RMCRNUM6:0>: Indicates the number of bits received as remote control signal data. The number cannot be monitored during reception. On completion of reception, the number is stored.

(Note 1) This register is updated every time an interrupt is generated.
Writing to this register is ignored.

(Note 2) RMC keeps receiving 73 bit or more data unless reception is completed by detecting the maximum data bit cycle or the excess low width. If so, the received data in the data buffer may not be correct.

14.3 Operation Description

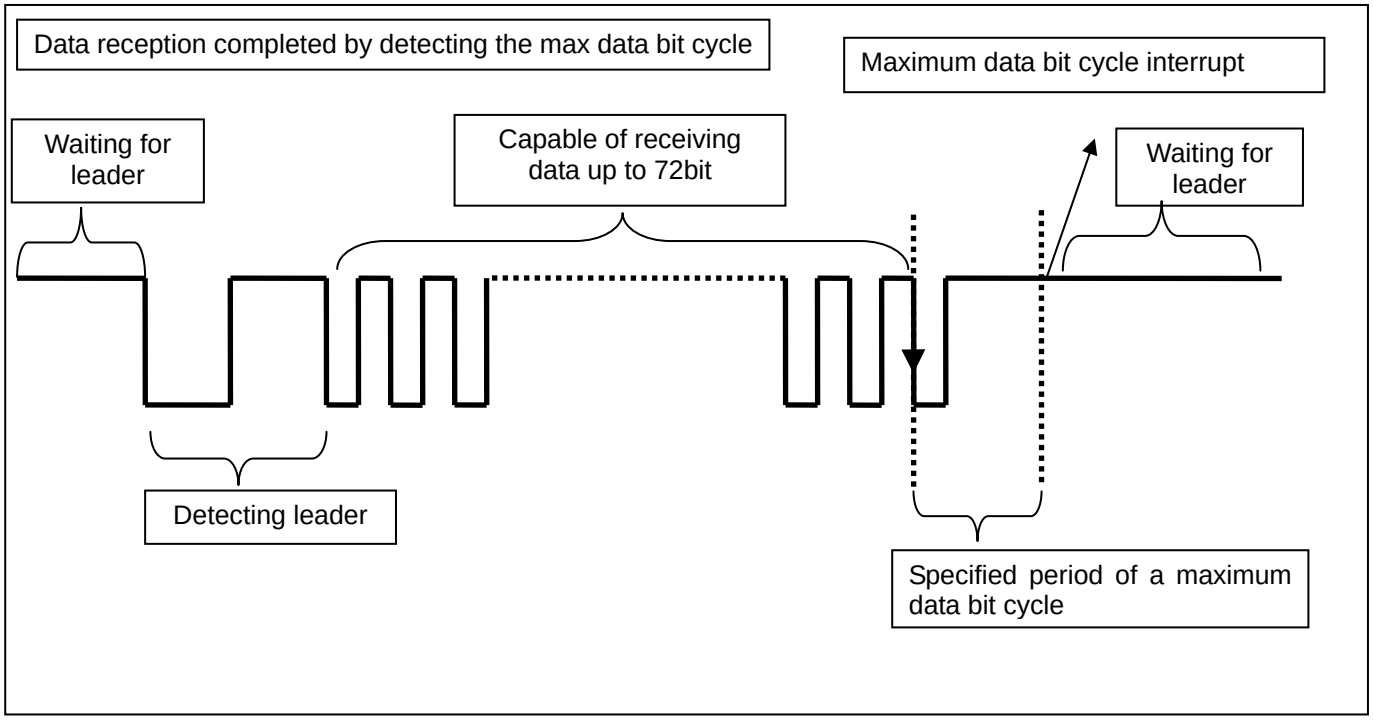
14.3.1 Reception of Remote Control Signal

14.3.1.1 Sampling Clock

A remote control signal is sampled by low-speed clock (fs).

14.3.1.2 Basic Operation

RMC starts to receive a data bit if a leader is detected while RMC is waiting for a leader. Based on a falling edge cycle, the data bit is determined as 0 or 1. By detecting a leader while RMC is waiting for a leader, a leader detection interrupt is generated, and the data bit reception starts. The data bit is determined as 0 or 1 based on a falling edge cycle. RMC is capable of receiving data up to 72bit. Reception is completed by detecting either a maximum data bit cycle or the excess low width. On completion of reception, RMC is waiting for the next leader, and the Remote Control Receive Data Buffer Registers and the Remote Control Receive Status Register are updated.



14.3.1.3 Preparation

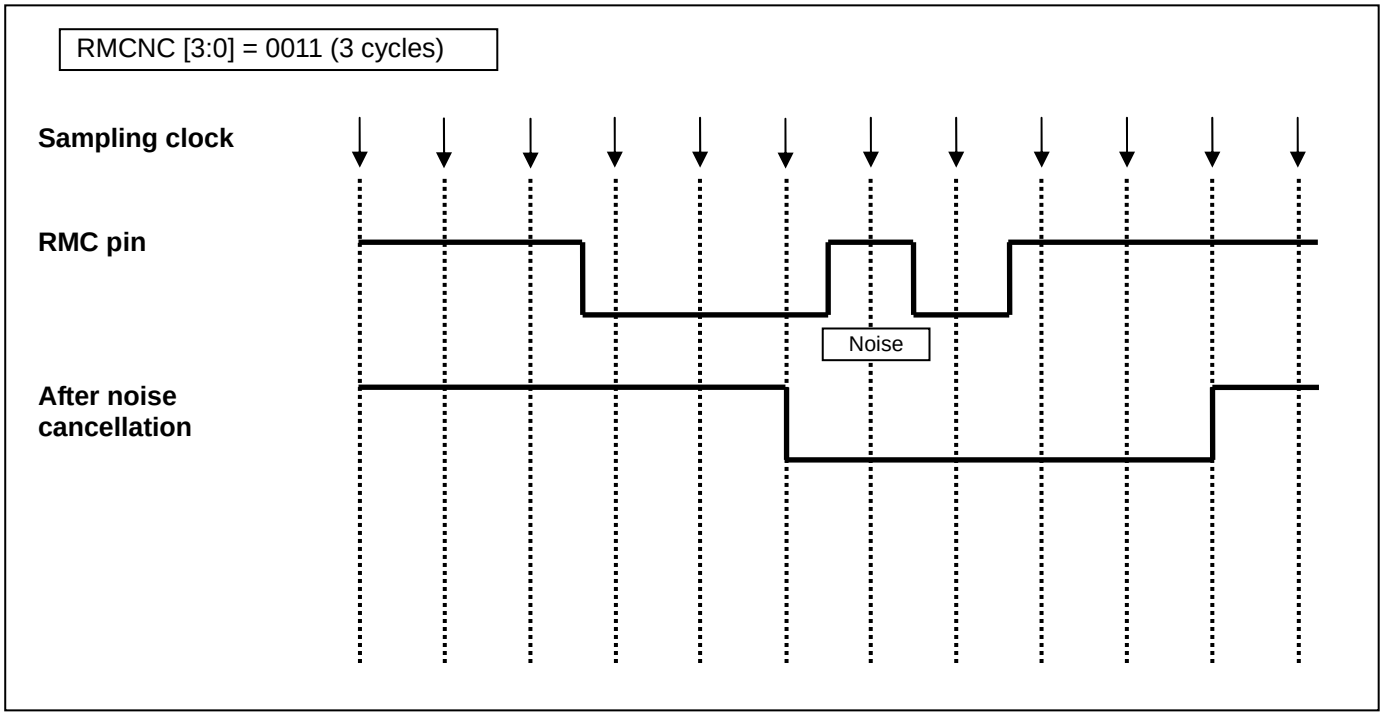
Configure reception operation of a remote control signal with the Remote Control Signal Receive Control Registers (RMCRCR1, RMCRCR 2 and RMCRCR3) before reception.

(1) Settings of Noise Cancelling Time

Configure noise cancelling time with the RMCRCR4 <RMCNC3:0> bit.

RMC monitors a remote control signal in each rising edge of a sampling clock. If “1” is monitored, RMC recognizes that the signal was changed to “0” after monitoring cycles of “0”s specified in RMCNC. If “0” is monitored, RMC recognizes that the signal was changed to “1” after monitoring cycles of “1”s specified in RMCNC.

The following figure shows how RMC operates according to the noise cancel setting of RMCNC [3:0] = 0011 (3 cycles). Subsequent to noise cancellation, the signal is changed from “1” to “0” upon monitoring 3 cycles of “0” s, and the signal is changed from “0” to “1” upon monitoring 3 cycles of “1” s.

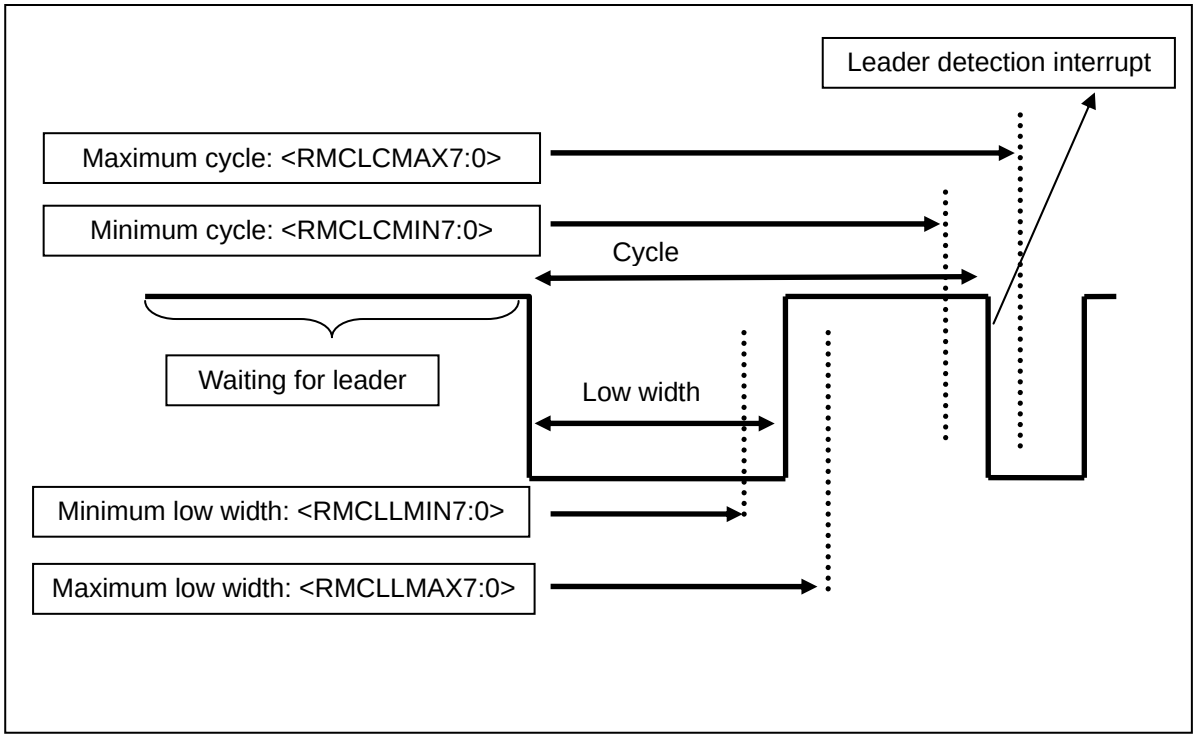


(2) Settings of Detecting Leader

To detect a leader, configure cycle and low width of the leader with the RMCRCR1 <RMCLLMIN7:0> <RMCLLMAX7:0> <RMCLCMIN7:0> <RMCLCMAX7:0> bits. When you configure the register, you must follow the rule shown below.

Leader	Rules
Low width + High width	<RMCLCMAX7:0> > <RMCLCMIN7:0> <RMCLLMAX7:0> > <RMCLLMIN7:0>
Only with high width	<RMCLCMIN7:0> > <RMCLLMAX7:0> <RMCLCMAX7:0> > <RMCLCMIN7:0> <RMCLLMAX7:0> = 0x00000000 <RMCLLMIN7:0> = don't care
No leader	<RMCLCMAX7:0> = 0x00000000 <RMCLCMIN7:0> = don't care <RMCLLMAX7:0> = don't care <RMCLLMIN7:0> = don't care

The following shows a leader waveform and the RMCRCR1 register settings.



If you want to generate an interrupt when detecting a leader, configure the RMCRCR2 <RMCLIEN> bit. A remote control signal without a leader cannot generate a leader detection interrupt.

(3) Settings of Data Bit Determination

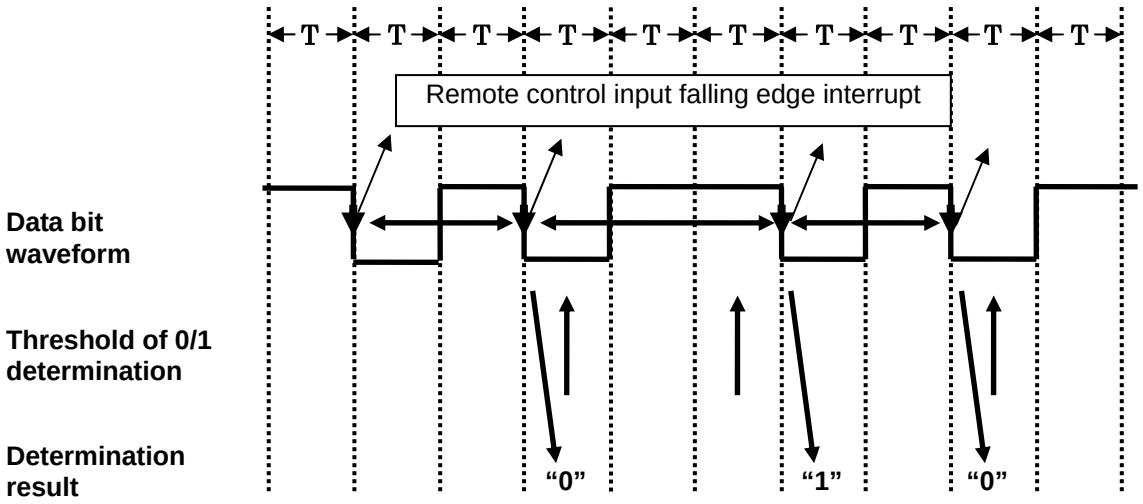
Based on a falling edge cycle, the data bit is determined as 0 or 1.

Configure a threshold of the determination with the RMCRCR3 <RMCDATL6:0> bit. If the cycle exceeds the threshold, the bit is determined as “1”. If not, the bit is determined as “0”.

By setting ”1” to the RMCRCR2 <RMCEDIEN> bit, a remote control signal input falling edge interrupt can be generated in each falling edge of the data bit. Using this interrupt together with a 16-bit timer enables the determination to be done by software.

The following shows how the data bit is determined as “0” or “1”.

Example: Threshold of 0/1 determination is set to 2.5T with the <RMCDATL6:0> bit.



As for data bit determination of a remote control signal in a phase method, see 14.3.1.10 “Receiving a Remote Control Signal in a Phase Method”.

(4) Settings of Reception Completion

To complete data reception, settings of detecting the maximum data bit cycle and excess low width are required. If multiple factors are specified, reception is completed by the factor detected first. Make sure to configure the reception completion settings.

1) Completed by a maximum data bit cycle

To complete reception by detecting a maximum data bit cycle, you need to configure the RMCRCR2 <RMCDMAX7:0> bits. If the falling edge of the data bit cycle isn’t monitored after time specified as threshold in the <RMCDMAX7:0> bits, a maximum data bit cycle is detected. The detection completes reception and generates an interrupt.

2) Completed by excess low width

To complete reception by detecting the low width, you need to configure the RMCRCR2 <RMCLL7:0> bits. After the falling edge of the data bit is detected, if the signal stays low longer than specified, excess low width is detected. The detection completes reception and generates an interrupt.

14.3.1.4 Enabling Reception

By enabling the RMCREN <RMCREN> bit after configuring the RMCRCR1, RMCRCR2, RMCRCR3 and RMCRCR4 registers, RMC is ready for reception. Detecting a leader initiates reception.

(Note)	**Changing the configurations of the RMCRCR1, RMCRCR2, RMCRCR3 and RMCRCR4 registers during reception may harm their proper operation. Be careful if you change them during reception.**

14.3.1.5 Reception

Detecting a leader sets the RMCRSTAT <RMCRLDR> bit. Simultaneously, a leader detection interrupt is generated if the RMCRCR2 <RMCLIEN> bit is set. When the interrupt is generated, the RMCRSTAT <RMCRLIF> bit is set.

Next to the leader detection, each data bit is determined as 0 or 1. The results are stored in the RMCRBUF1, RMCRBUF 2 and RMCRBUF 3 registers up to 72bit. By setting “1” to the RMCRCR2 <RMCEDIEN> bit, a remote control signal input falling edge interrupt can be generated in each falling edge of the data bit. When the interrupt is generated, the RMCRSTAT <RMCEDIF> bit is set.

Detecting the maximum data bit cycle or the excess low width completes reception and generates an interrupt.

To check the status of RMC after reception is completed, read the Remote Control Receive Status Register.

On completion of reception, RMC is waiting for the next leader.

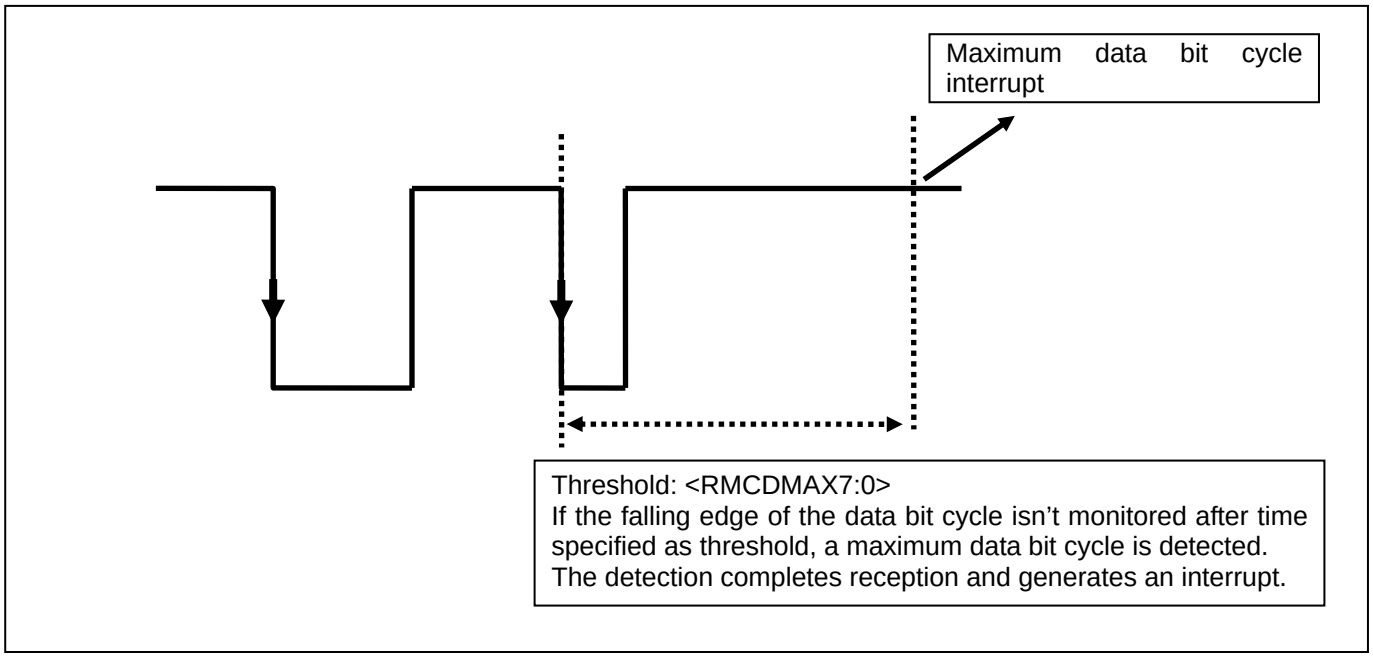
By setting RMC to receive a signal without a leader, RMC recognizes the received is data and starts reception without detecting a leader.

If the next data reception is completed before reading the preceding received data, the preceding data is overwritten by the next one.

14.3.1.6 Reception Completion

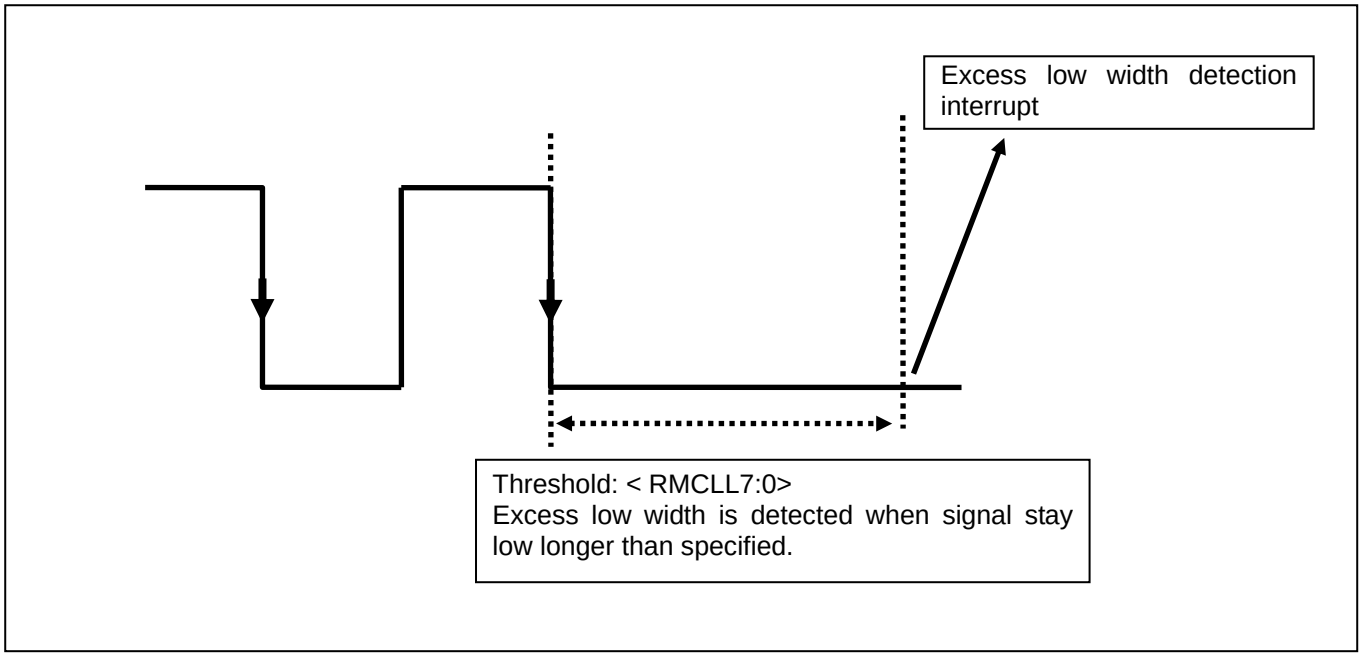
1) Completed by a maximum data bit cycle

Detecting a maximum data bit cycle completes reception and generates an interrupt. After the interrupt is generated, the RMCRSTAT <RMCDMAXIF> bit is set to “1”.



2) Completed by excess low width

Detecting excess low width completes reception and generates an interrupt. After the interrupt is generated, the RMCRSTAT <RMCLOIF> bit is set to “1”.



RMC keeps receiving 73 bit or more data unless reception is completed by detecting the maximum data bit cycle or the excess low width. If so, the received data in the data buffer may not be correct.

To check the status of RMC after reception is completed, read the Remote Control Receive Status Register. The status of RMC that each bit type indicates is shown below.

<RMCRLDR>	<RMCRNUM6:0>	RMC Status
0	0000001~1001000	Receiving remote control signal without a leader (Data bits: 1~72bit)
0	1001001~1111111	Receiving remote control signal without a leader (Data bits: 73bit and more)
1	0000000	Only with a leader
1	0000001~1001000	Receiving remote control signal with a leader (Data bits: 1~72bit)
1	1001001~1111111	Receiving remote control signal without a leader (Data bits: 73bit and more)

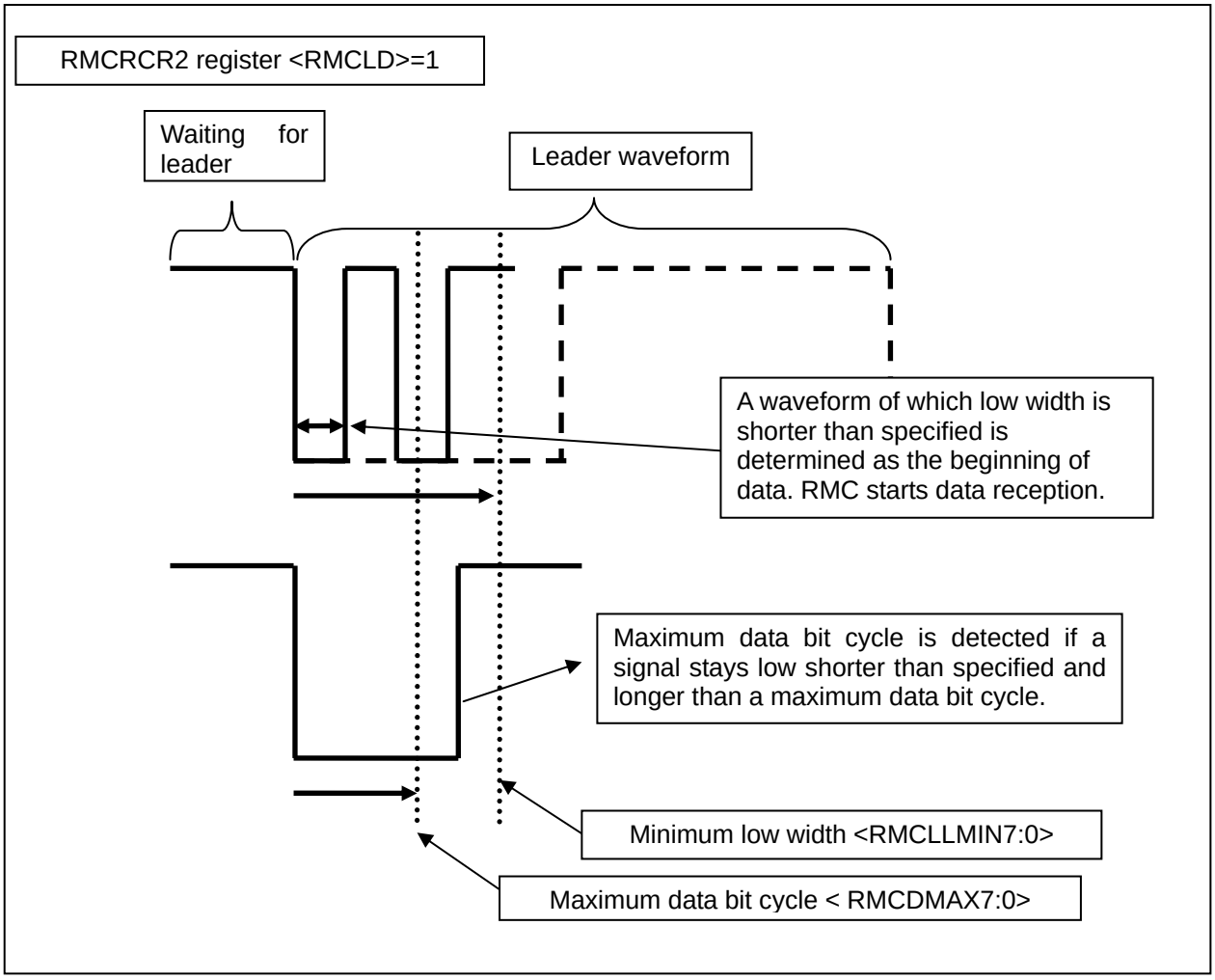
14.3.1.7 Stopping Reception

RMC stops reception by clearing the RMCREN <RMCREN> bit to “0” (reception disabled). Clearing this bit during reception stops reception immediately and the received data is discarded.

14.3.1.8 Receiving Remote Control Signal without Leader

Setting RMCRCR2 <RMCLD> enables RMC to receive signals with or without a leader. By setting RMCRCR2 <RMCLD>, RMC starts receiving data if it recognizes a signal of which low width is shorter than a maximum low width of leader detection specified in the RMCRCR1 <RMCLLMAX7:0> bits. RMC keeps receiving data until the final data bit is received.

If RMCRCR2 <RMCLD> is enabled, the same settings of error detection, reception completion and data bit determination of 0 or 1 are applied regardless of whether a signal has a leader or not. Thus receivable remote control signals are limited.



14.3.1.9 A Leader only with Low Width

The figure shown below illustrates a remote control signal that starts with a leader of which waveform only has low width. This signal starts with a leader that only has low width and a data bit cycle starts from the rising edge. To enable the signal, it must be sent after being reversed by setting the RMCRCR4 <RMCPO> bit to “1”. This is because RMC is configured to detect a data bit cycle from the falling edge.

A leader is detected by the low width. When you configure the RMCRCR1 register, you must follow the rule shown below.

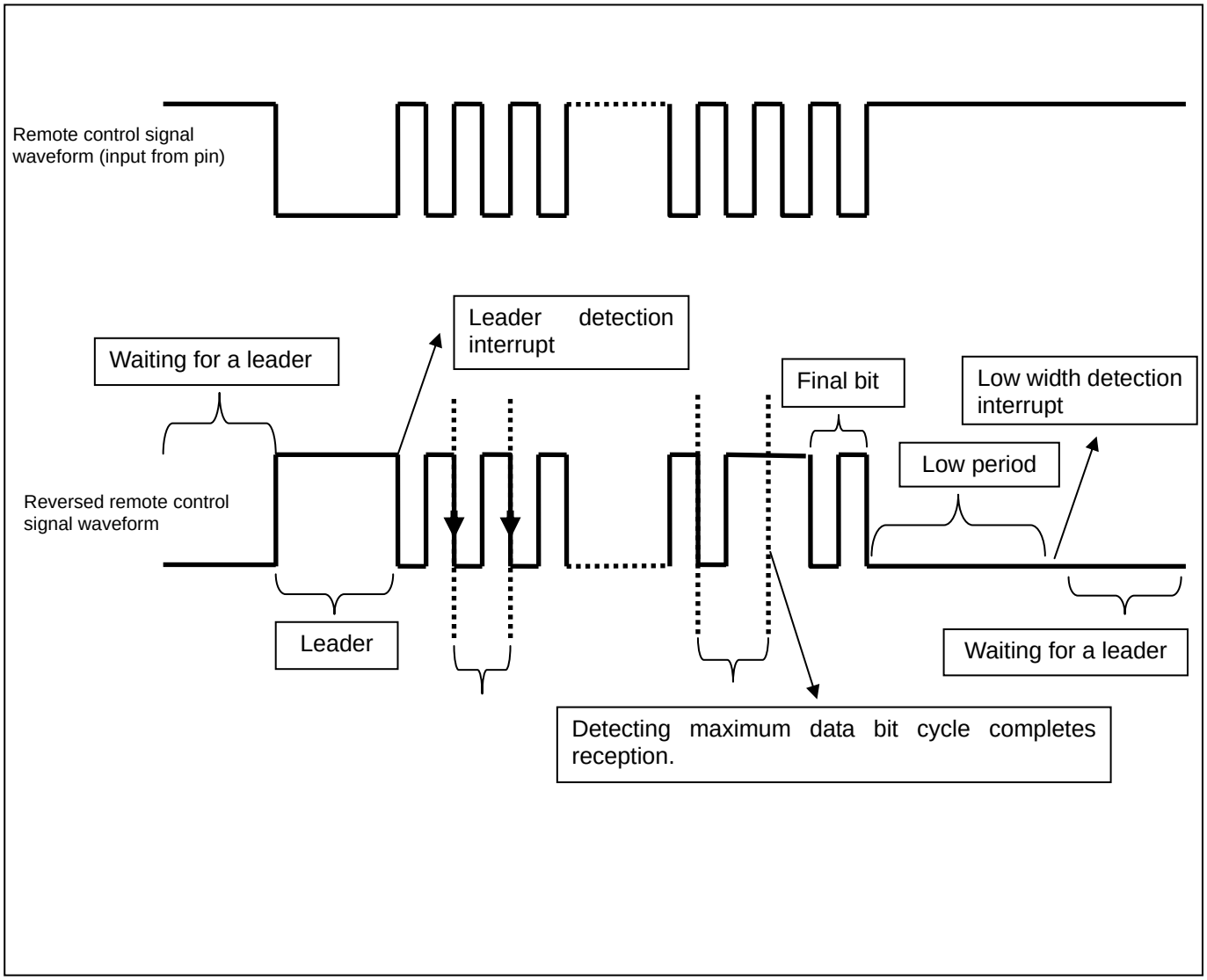
<RMCLLMAX7:0> = 0x00000000
<RMCLHMAX7:0> > <RMCLHMIN7:0>

If the rules are applied, RMC does not care about the value of <RMCLLMIN7:0>.

To determine the data bit as 0 or 1, configure a threshold of the determination with the RMCRCR3 <RMCDATL6:0> bit.

Configure a maximum data bit cycle with the <RMCDMAX7:0> bits of the Remote Control Receive Control Register 2.

To complete reception by detecting the maximum data bit cycle, you need to configure the RMCRCR2 <RMCDMAX7:0> bits. To complete reception by detecting the low width, you need to configure the RMCRCR2 <RMCLL7:0> bits. Detecting the maximum data bit cycle or the excess low width completes reception and generates an interrupt. RMC waits for the next leader.


Remote control signal waveform (input from pin)
Waiting for a leader
Leader detection interrupt
Final bit
Low width detection interrupt
Low period
Reversed remote control signal waveform
Leader
Waiting for a leader
Detecting maximum data bit cycle completes reception.

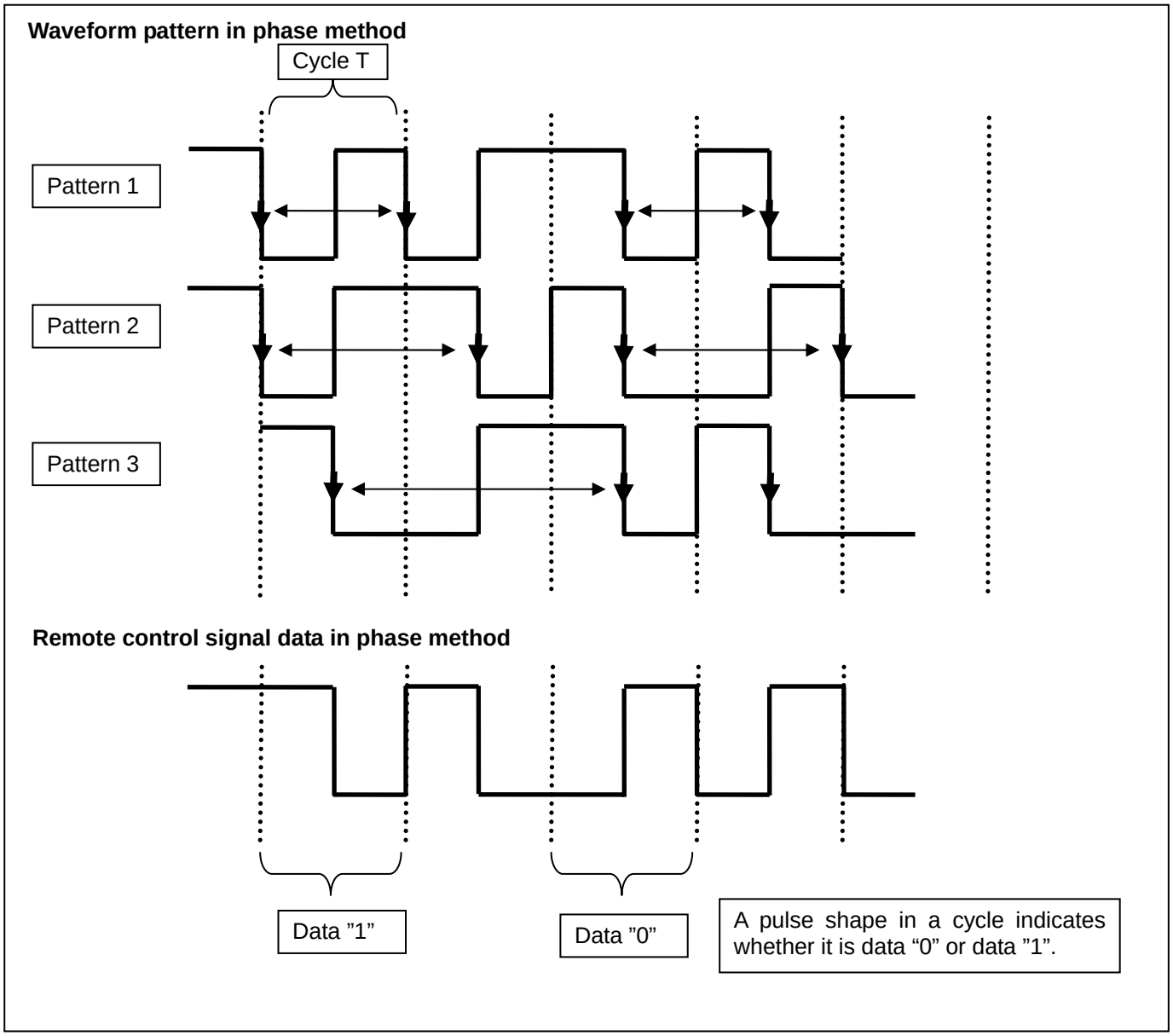
14.3.1.10 Receiving a Remote Control Signal in a Phase Method

RMC is capable of receiving a remote control signal in a phase method of which signal cycle is fixed. A signal in the phase method has three waveform patterns (see the figure shown below). By setting two thresholds a remote control signal pattern is determined. RMC converts the signal into data ”0” or “1”. On completion of reception, received data “0” and “1” are stored in the RMCRBUF1, RMCRBUF 2 and RMCRBUF3.

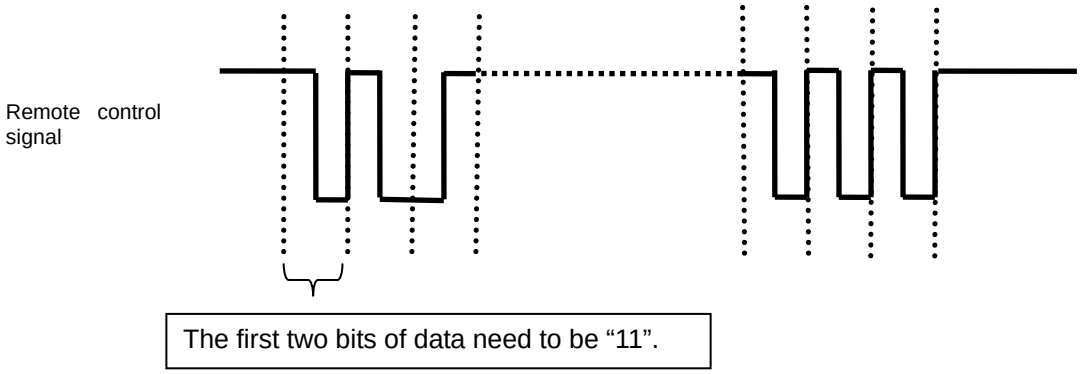
By setting RMCRCR2<RMCPHM>=”1”, RMC enables to receive a remote control signal in the phase method. Each threshold can be configured with the RMCRCR4 <RMCDATL6:0> bits and <RMCDATH6:0> bits. Two thresholds are used to distinguish three waveform patterns. On condition that a cycle between two falling edges is “T”, three patterns show cycles of 1T, 1.5T and 2T. Details of the two thresholds are shown below.

To determine a remote control signal in the phase method, three patterns of data waveform and preceding data are required. In addition, the signal needs to start from data “1”.

	Determined by	Threshold	Register bits to set
Threshold 1	Pattern 1 & pattern 2	1T~1.5T	RMCRCR2 register <RMCDATL6:0>
Threshold 2	Pattern 2 & pattern 3	1.5T~2T	RMCRCR2 register <RMCDATH6:0>



Remote control signal in phase method



15 Analog/Digital Converter

A 10-bit, sequential-conversion analog/digital converter (A/D converter) is built into the TMPM330. This A/D converter is equipped with 12 analog input channels.

Fig. 15-1 shows the block diagram of this A/D converter.

These 12 analog input channels (pins AN0 through AN11) are also used as input/ output ports.

(Note) If it is necessary to reduce a power current by operating the TMPM330 in IDLE or STOP mode and if either case shown below is applicable, you must first stop the A/D converter and then execute the instruction to put the TMPM330 into standby mode:

1) The TMPM330 must be put into IDLE mode when ADMOD1<I2AD> is "0."
2) The TMPM330 must be put into STOP mode.

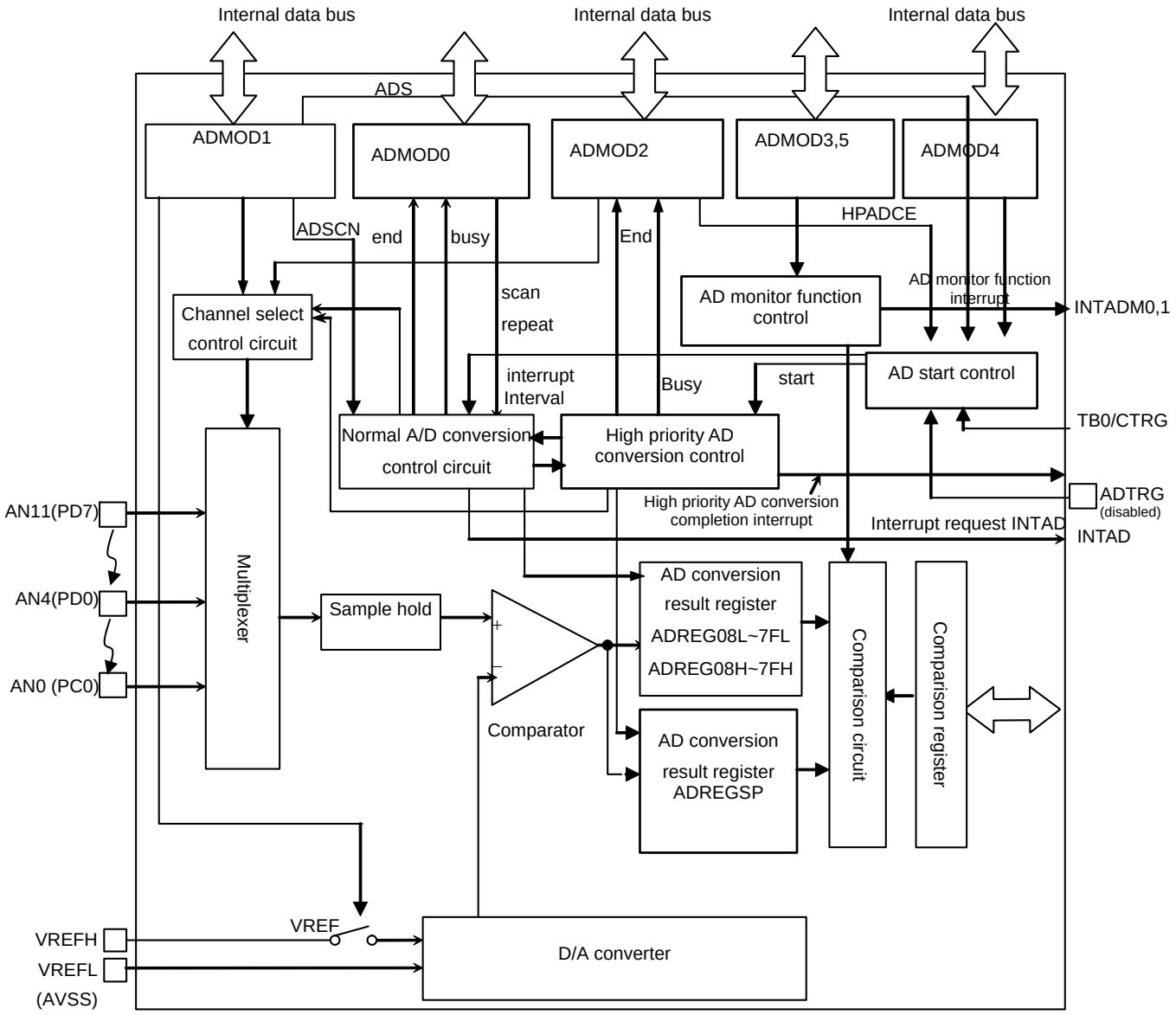


Fig. 15-1 A/D Converter Block Diagram

15.1 Registers

The control registers and addresses of the A/D converter are as follows.

Register name		Address
A/D Conversion Clock Setting Register	ADCLK	0x4003_0000
A/D Mode Control Register 0	ADMOD0	0x4003_0004
A/D Mode Control Register 1	ADMOD1	0x4003_0008
A/D Mode Control Register 2	ADMOD2	0x4003_000C
A/D Mode Control Register 3	ADMOD3	0x4003_0010
A/D Mode Control Register 4	ADMOD4	0x4003_0014
A/D Mode Control Register 5	ADMOD5	0x4003_0018
A/D Conversion Accuracy Setting Register (Note)	ADCBAS	0x4003_0020
Lower A/D Conversion Result Register 08L	ADREG08L	0x4003_0030
Upper A/D Conversion Result Register 08H	ADREG08H	0x4003_0031
Lower A/D Conversion Result Register 19L	ADREG19L	0x4003_0034
Upper A/D Conversion Result Register 19H	ADREG19H	0x4003_0035
Lower A/D Conversion Result Register 2AL	ADREG2AL	0x4003_0038
Upper A/D Conversion Result Register 2AH	ADREG2AH	0x4003_0039
Lower A/D Conversion Result Register 3BL	ADREG3BL	0x4003_003C
Upper A/D Conversion Result Register 3BH	ADREG3BH	0x4003_003D
Lower A/D Conversion Result Register 4CL	ADREG4CL	0x4003_0040
Upper A/D Conversion Result Register 4CH	ADREG4CH	0x4003_0041
Lower A/D Conversion Result Register 5DL	ADREG5DL	0x4003_0044
Upper A/D Conversion Result Register 5DH	ADREG5DH	0x4003_0045
Lower A/D Conversion Result Register 6EL	ADREG6EL	0x4003_0048
Upper A/D Conversion Result Register 6EH	ADREG6EH	0x4003_0049
Lower A/D Conversion Result Register 7FL	ADREG7FL	0x4003_004C
Upper A/D Conversion Result Register 7FH	ADREG7FH	0x4003_004D
Lower A/D Conversion Result Register SP	ADREGSPL	0x4003_0050
Upper A/D Conversion Result Register SP	ADREGSPH	0x4003_0051
Lower A/D Conversion Result Comparison Register 0	ADCMP0L	0x4003_0054
Upper A/D Conversion Result Comparison Register 0	ADCMP0H	0x4003_0055
Lower A/D Conversion Result Comparison Register 1	ADCMP1L	0x4003_0058
Upper A/D Conversion Result Comparison Register 1	ADCMP1H	0x4003_0059

Note) To assure conversion accuracy, the ADCBAS register must be configured as follows.

0x4003_0020 ＝ 0x58

ADCBAS

	7	6	5	4	3	2	1	0
bit Symbol								
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	1	0	1	1	0	0	0
Function	Write “0”.	Write “1”.	Write “0”.	Write “1”.	Write “1”.	Write “0”.	Write “0”.	Write “0”.

15.2 Registers Description

The A/D converter is controlled by A/D mode control registers (ADMOD0, ADMOD1, ADMOD2, ADMOD3, ADMOD4 and ADMOD5). Results of A/D conversion are stored in 16 upper and lower A/D conversion result registers ADREG08H/L through ADREG7FH/L. Results of top-priority conversion are stored in ADREGSPH/L.

Here are the descriptions of the registers.

A/D Mode Control Register 0

ADMOD0	7	6	5	4	3	2	1	0
Bit symbol	EOCFN	ADBFN		ITM1	ITM0	REPEAT	SCAN	ADS
Read/Write	R		R	R/W				
After reset	0	0	0	0	0	0	0	0
Function	Normal A/D conversion completion flag 0: Before or during conversion 1: Completion	Normal A/D conversion BUSY flag 0: Conversion stop 1: During conversion	"0" is read.	Specify interrupt in fixed channel repeat conversion mode	Specify interrupt in fixed channel repeat conversion mode.	Specify repeat mode 0: Single conversion mode 1: Repeat conversion mode	Specify scan mode 0: Fixed channel mode 1: Channel scan mode	Start A/D conversion 0: Don't care 1: Start conversion "0" is always read.

Specify A/D conversion interrupt in fixed channel repeat conversion mode

	Fixed channel repeat conversion mode <SCAN> = "0", <REPEAT> = "1"
00	Generate interrupt once every single conversion
01	Generate interrupt once every 4 conversions
10	Generate interrupt once every 8 conversions
11	Setting prohibited

Fig. 15-2 A/D Mode Control Register 0

(Note 1) Please specify the mode first and then specify the <ADS> bit.

A/D Mode Control Register 1

ADMOD1	7	6	5	4	3	2	1	0
Bit symbol	VREFON	I2AD	ADSCN	–	ADCH3	ADCH2	ADCH1	ADCH0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	VREF application control 0:OFF 1:ON	IDLE 0 : Stop 1 : Operation	Specify operation mode for channel scanning 0: 4ch scan 1: 8ch scan	Write "0."	Select analog input channel			

Select analog input channel

<SCAN> <ADCH3.2, 1, 0>	0 Fixed channel	1 Channel scan (ADSCN=0)	1 Channel scan (ADSCN=1)
0000	AN0	AN0	AN0
0001	AN1	AN0～AN1	AN0～AN1
0010	AN2	AN0～AN2	AN0～AN2
0011	AN3	AN0～AN3	AN0～AN3
0100	AN4	AN4	AN0～AN4
0101	AN5	AN4～AN5	AN0～AN5
0110	AN6	AN4～AN6	AN0～AN6
0111	AN7	AN4～AN7	AN0～AN7
1000	AN8	AN８	AN8
1001	AN9	AN8～AN9	AN8～AN9
1010	AN10	AN8～AN10	AN8～AN10
1011	AN11	AN8～AN11	AN8～AN11
1101	Setting prohibited		
1110			
1111			

(Note 1) Before starting AD conversion, write "1" to the <VREFON> bit, wait for 3 μs during which time the internal reference voltage should stabilize, and then write "1" to the ADMOD0<ADS> bit.

(Note 2) To go into standby mode upon completion of AD conversion, set <VREFON> to "0."

Fig. 15-3 A/D Mode Control Register 1

A/D Mode Control Register 2

ADMOD2

	7	6	5	4	3	2	1	0
Bit symbol	EOCFHP	ADBFHP	HPADCE	–	HPADCH3	HPADCH2	HPADCH1	HPADCH0
Read/Write	R	R	R/W					
After reset	0	0	0	0	0	0	0	0
Function	Top-priority AD conversion completion flag 0:Before or during conversion 1: Upon completion	Top-priority AD conversion BUSY flag 0: During conversion halts 1:During conversion	Activate top-priority conversion 0: Don't care 1: Start conversion. "0" is always read.	Write "0".	Select analog input channel when activating top-priority conversion.			

<HPADCH4,3.2, 1, 0>	Analog input channel when executing top-priority conversion
0000	AN0
0001	AN1
0010	AN2
0011	AN3
0100	AN4
0101	AN5
0110	AN6
0111	AN7
1000	AN8
1001	AN9
1010	AN10
1011	AN11
1100	AN12
1101	Setting prohibited
1110	
1111	

Fig. 15-4 A/D Mode Control Register 2

A/D Mode Control Register 3

ADMOD3

	7	6	5	4	3	2	1	0
Bit symbol			ADOBIC	REGS3	REGS2	REGS1	REGS0	ADOBSV
Read/Write	R/W	R	R/W					
After reset	0	0	0	0	0	0	0	0
Function	Write "0".	"0" can be read.	Make AD monitor function interrupt setting 0: Smaller than comparison Reg. 1: Larger than comparison Reg.	BIT for selecting the AD conversion result storage Reg. that is to be compared with the comparison Reg. if the AD monitor function is enabled				AD monitor function 0 : Disable 1 : Enable

<REGS.2, 1, 0>	AD conversion result storage Reg. to be compared
0000	ADREG08
0001	ADREG19
0010	ADREG2A
0011	ADREG3B
0100	ADREG4C
0101	ADREG5D
0110	ADREG6E
0111	ADREG7F
1XXX	ADREGSP

Fig. 15-5 A/D Mode Control Register 3

A/D Mode Control Register 5

ADMOD5

	7	6	5	4	3	2	1	0
Bit symbol			ADOBIC	REGS3	REGS2	REGS1	REGS0	ADOBSV
Read/Write	R		R/W					
After reset	0	0	0	0	0	0	0	0
Function	"0" can be read.		Make AD monitor function interrupt setting Smaller than comparison Reg. Larger than comparison Reg.	BIT for selecting the AD conversion result storage Reg. that is to be compared with the comparison Reg. if the AD monitor function is enabled .				AD monitor function 0 : Disable 1 : Enable

<REGS.2, 1, 0>	AD conversion result storage Reg. to be compared
0000	ADREG08
0001	ADREG19
0010	ADREG2A
0011	ADREG3B
0100	ADREG4C
0101	ADREG5D
0110	ADREG6E
0111	ADREG7F
1XXX	ADREGSP

Fig. 15-6 A/D Mode Control Register 5

A/D Mode Control Register 4

ADMOD4

	7	6	5	4	3	2	1	0
Bit symbol	HADHS	HADHTG	ADHS	ADHTG			ADRST1	ADRST0
Read/Write	R/W				R		W	W
After reset	0	0	0	0	0		–	–
Function	HW source for activating top-priority A/D conversion 0: External TRG 1:Match with TB5RG0	HW for activating top-priority A/D conversion 0: Disable 1: Enable	HW source for activating normal A/D conversion 0:External TRG 1:Match with TB6RG0	HW for activating normal A/D conversion 0: Disable 1: Enable	"0" can be read.		Overwriting 10 with 01 allows ADC to be software reset.	

(Note 1) If AD conversion is executed with the match triggers <ADHTG> and <HADHTG> of a 16-bit timer set to "1" by using a source for triggering H/W, A/D conversion can be activated at specified intervals by performing three steps shown below when the timer is idle:

1. **Select a source for triggering HW: <ADHS>, <HADHS>**
2. **Enable H/W activation of AD conversion: <ADHTG>, <HADHTG>**
3. **Start the timer.**

(Note 2)Do not make a top-priority AD conversion setting and a normal AD conversion setting simultaneously.

(Note 3) The external trigger cannot be used for H/W activation of AD conversion when it is used for H/W activation of top priority AD conversion.

(Note 4) A software reset initializes other bits. Resetting a mode register is needed.

(Note) The TMPM330 disables the external trigger used for H/W activation. Therefore "0" cannot be set to <HADHS> and <ADHS>.

Fig. 15-7 A/D Mode Control Register 4

Lower A/D Conversion Result Register 08

ADREG08L

	7	6	5	4	3	2	1	0
Bit symbol	ADR01	ADR00					OVR0	ADR0RF
Read/Write	R		R				R	R
After reset	0		0				0	0
Function	Store lower 2 bits of A/D conversion result		"0" can be read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1:Presence of conversion result

Upper A/D Conversion Result Register 08

ADREG08H

	7	6	5	4	3	2	1	0
Bit symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							

Lower A/D Conversion Result Register 19

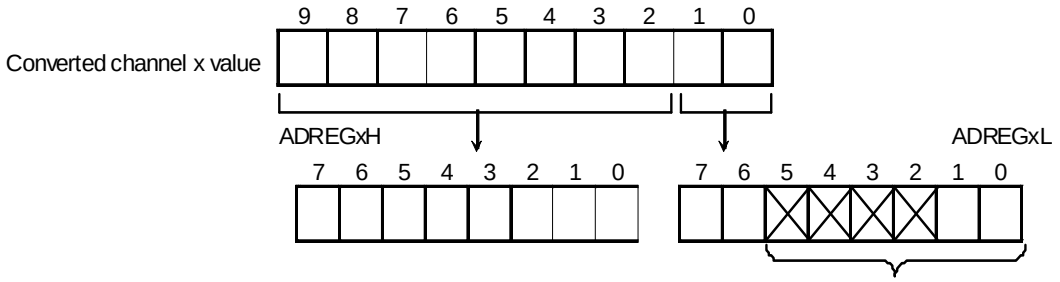
ADREG19L

	7	6	5	4	3	2	1	0
Bit symbol	ADR11	ADR10					OVR1	ADR1RF
Read/Write	R		R				R	R
After reset	0		0				0	0
Function	Store lower 2 bits of A/D conversion result		"0" can be read.				Over RUNflag 0: Not generate 1: Generate	A/D conversion result storage flag 1:Presence of conversion result

Upper A/D Conversion Result Register 19

ADREG19H

	7	6	5	4	3	2	1	0
Bit symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- Bit 0 of ADREG08L/ADREG19L is the A/D conversion result storage flag <ADRxRF>. This bit is set to "1" after an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0".
- Bit 1 of ADREG08L/ADREG19L is the over RUN flag <OVRx>. This bit is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH and ADREGxL) are read. A read of a flag will clear this bit to "0."
- When reading conversion result storage registers on a byte-by-byte basis, first read upper registers and then read lower registers.

Fig. 15-8 A/D Conversion Result Register

Lower A/D Conversion Result Register 2A

ADREG2AL

	7	6	5	4	3	2	1	0
Bit symbol	ADR21	ADR20					OVR2	ADR2RF
Read/Write	R		R				R	R
After reset	0		0				0	0
Function	Store lower 2 bits of A/D conversion result		"0" is read.				Over RUN flag 0:Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 2A

ADREG2AH

	7	6	5	4	3	2	1	0
bit Symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							

Lower A/D Conversion Result Register 3B

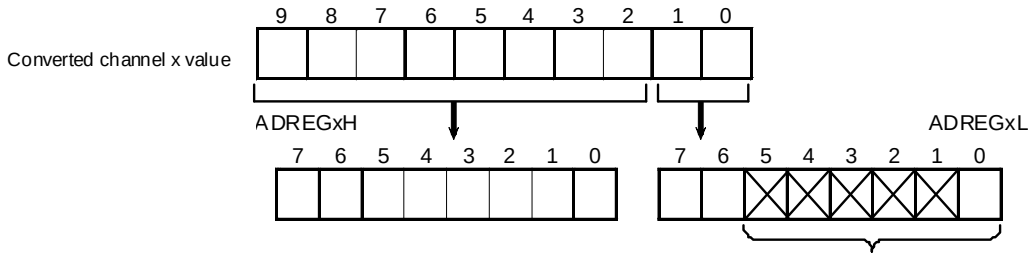
ADREG3BL

	7	6	5	4	3	2	1	0
bit Symbol	ADR31	ADR30					OVR3	ADR3RF
Read/Write	R		R				R	R
After reset	0		0				0	0
Function	Store lower 2 bits of A/D conversion result		"0" is read.				Over RUN flag 0:Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 3B

ADREG3BH

	7	6	5	4	3	2	1	0
bit Symbol	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- Bit 0 of the ADREG2AL/ADREG3BL is the A/D conversion result storage flag <ADRxRF>. It is set to "1" after an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0".
- Bit 1 of the ADREG2AL/ADREG3BL is the over RUN flag <OVRx>. It is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH,ADREGxL) are read. A read of a flag will clear this bit to "0.
- When reading conversion result storage registers on a byte-by-byte basis, first read upper registers and then read lower registers.

Fig. 15-9 A/D Conversion Result Register

Lower A/D Conversion Result Register 4C

ADREG4CL

	7	6	5	4	3	2	1	0
Bit symbol	ADR41	ADR40					OVR4	ADR4RF
Read/Write	R		R				R	R
After reset	0		0				0	0
Function	Store lower 2 bits of A/D conversion result		"0" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 4C

ADREG4CH

	7	6	5	4	3	2	1	0
Bit symbol	ADR49	ADR48	ADR47	ADR46	ADR45	ADR44	ADR43	ADR42
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							

Lower A/D Conversion Result Register 5D

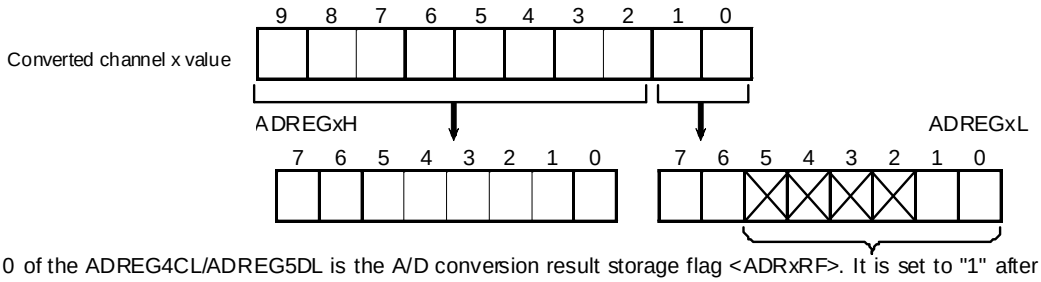
ADREG5DL

	7	6	5	4	3	2	1	0
Bit symbol	ADR51	ADR50					OVR5	ADR5RF
Read/Write	R		R				R	R
After reset	0		0				0	0
Function	Store lower 2 bits of A/D conversion result		"0" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag Presence of conversion result

Upper A/D Conversion Result Register 5D

ADREG5DH

	7	6	5	4	3	2	1	0
Bit symbol	ADR59	ADR58	ADR57	ADR56	ADR55	ADR54	ADR53	ADR52
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- Bit 0 of the ADREG4CL/ADREG5DL is the A/D conversion result storage flag <ADRxRF>. It is set to "1" after an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0".
- Bit 1 of the ADREG4CL/ADREG5DL is the over Run flag <OVRx>. It is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH and ADREGxL) are read. A read of a flag will clear this bit to "0".
- When reading conversion result storage registers on a byte-by-byte basis, first read upper registers and then read lower registers.

Fig. 15-10 A/D Conversion Result Register

Lower A/D Conversion Result Register 6E

ADREG6EL

	7	6	5	4	3	2	1	0
Bit Symbol	ADR61	ADR60					OVR6	ADR6RF
Read/Write	R		R				R	R
After reset	0		0				0	0
Function	Store lower 2 bits of A/D conversion result		"0" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 6E

ADREG6EH

	7	6	5	4	3	2	1	0
Bit Symbol	ADR69	ADR68	ADR67	ADR66	ADR65	ADR64	ADR63	ADR62
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							

Lower A/D Conversion Result Register 7F

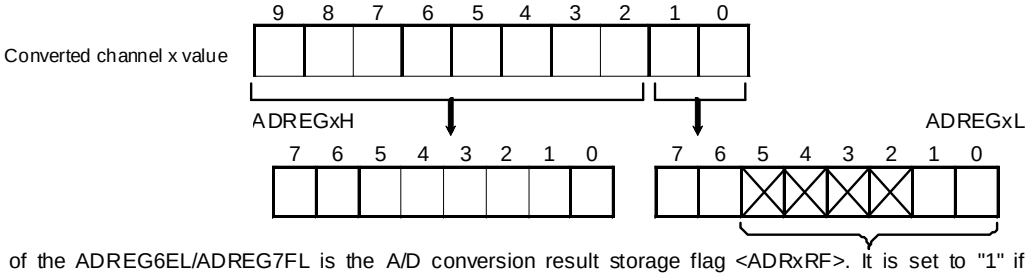
ADREG7FL

	7	6	5	4	3	2	1	0
Bit Symbol	ADR71	ADR70					OVR7	ADR7RF
Read/Write	R		R				R	R
After reset	0		0				0	0
Function	Store lower 2 bits of A/D conversion result		"0" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 7F

ADREG7FH

	7	6	5	4	3	2	1	0
Bit Symbol	ADR79	ADR78	ADR77	ADR76	ADR75	ADR74	ADR73	ADR72
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- Bit 0 of the ADREG6EL/ADREG7FL is the A/D conversion result storage flag <ADRxRF>. It is set to "1" if an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0".
- Bit 1 of the ADREG6EL/ADREG7FL is the over Run flag <OVRx>. It is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH and ADREGxL) are read. A read of a flag will clear this bit to "0".
- When reading conversion result storage registers on a byte-by-byte basis, first read upper registers and then read lower registers.

Fig. 15-11 A/D Conversion Result Register

Lower A/D Conversion Result Register SP

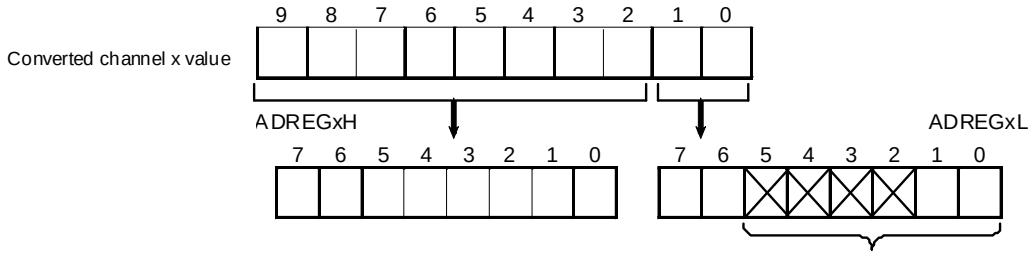
ADREGSPL

	7	6	5	4	3	2	1	0
Bit symbol	ADRSP1	ADRSP0					OVRSP	ADRSPRF
Read/Write	R		R				R	R
After reset	0		0				0	0
Function	Store lower 2 bits of A/D conversion result		"0" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register SP

ADREGSPH

	7	6	5	4	3	2	1	0
Bit symbol	ADRSP9	ADRSP8	ADRSP7	ADRSP6	ADRSP5	ADRSP4	ADRSP3	ADRSP2
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- Bit 0 of the ADREGSP is the A/D conversion result storage flag <ADRxRF>. It is set to "1" after an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0."
- Bit 1 of the ADREGSP is the over RUN flag <OVRx>. It is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH and ADREGxL) are read. A read of a flag will clear this bit to "0".
- When reading conversion result storage registers on a byte-by-byte basis, first read upper registers and then read lower registers.

Fig. 15-12 A/D Conversion Result Register

Lower A/D Conversion Result Comparison Register 0

ADCMP0L

	7	6	5	4	3	2	1	0
Bit symbol	ADR021	ADR020						
Read/Write	R/W		R					
After reset	0		0					
Function	Store lower 2 bits of A/D conversion result comparison		"0" is read.					

Upper A/D Conversion Result Comparison Register 0

ADCMP0H

	7	6	5	4	3	2	1	0
Bit symbol	ADR029	ADR028	ADR027	ADR026	ADR025	ADR024	ADR023	ADR022
Read/Write	R/W							
After reset	0							
Function	Store upper 8 bits of A/D conversion result comparison							

Lower A/D Conversion Result Comparison Register 1

ADCMP1L

	7	6	5	4	3	2	1	0
Bit symbol	ADR121	ADR120						
Read/Write	R/W		R					
After reset	0		0					
Function	Store lower 2 bits of A/D conversion result comparison		"0" is read.					

Upper A/D Conversion Result Comparison Register 1

ADCMP1H

	7	6	5	4	3	2	1	0
Bit symbol	ADR129	ADR128	ADR127	ADR126	ADR125	ADR124	ADR123	ADR122
Read/Write	R/W							
After reset	0							
Function	Store upper 8 bits of A/D conversion result comparison							

(Note) To set or change a value in this register, the AD monitor function must be disabled (ADMOD3, 5 <ADOBSVx>＝”0”).

Fig. 15-13 A/D Conversion Result Register

15.3 Conversion Clock

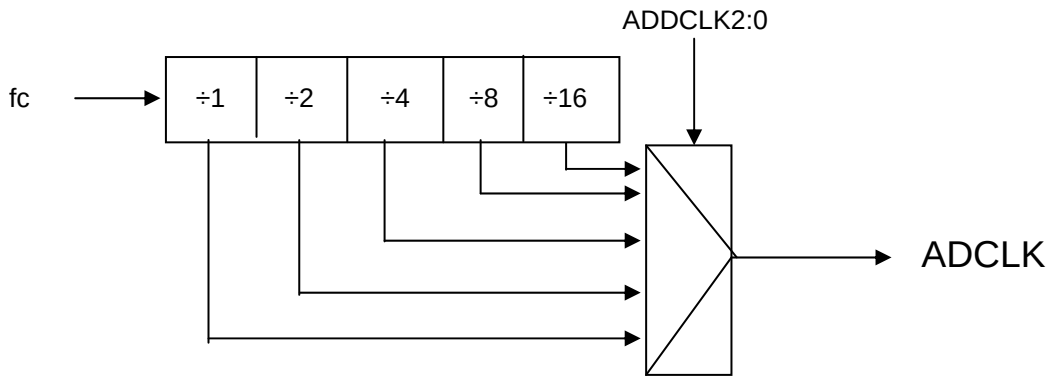
- The conversion time is calculated by the 46 conversion clock.

A/D Conversion Clock Setting Register

ADCLK

	7	6	5	4	3	2	1	0
Bit symbol	TSH3	tSH2	tSH1	tSH0		ADCLK2	ADCLK1	ADCLK0
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W
After reset	1	0	0	0	0	0	0	0
Function	Select the A/D sample hold time 1000: 8 conversion clock 1001: 16 conversion clock 1010: 24 conversion clock 1011: 32 conversion clock 0011: 64 conversion clock 1100: 128 conversion clock 1101: 512 conversion clock The setup other than those above: reserved				"0" is read.	Select the A/D prescaler output 000: fc 001: fc／2 010: fc／4 011: fc／8 100: fc／16 111:reserved		

Fig. 15-14 A/D Conversion Clock Setting Register



Example: If fsys = fc = 32 MHz/ 54MHz

Prescaler [ADDCLK2:0]	tconv.(conversion time) 40MHz	tconv.（conversion time） 32MHz
1	1.15μs	1.44μs
1/2	2.3μs	2.88μs
1/4	4.6μs	5.75μs

Variable S/H time

Conversion clock	S/H time 40MHz	tconv.（conversion time）
40MHz	Conversion clk*8 （0.2 μs）	1.15μs
	Conversion clk*16 （0.4 μs）	1.35μs
	Conversion clk*24 （0.6 μs）	1.55μs
	Conversion clk*32 （0.8 μs）	1.75μs
	Conversion clk*64 （1.6 μs）	2.55μs
	Conversion clk*128 （3.2 μs）	4.15μs
	Conversion clk*512 （12.8 μs）	13.75μs

(Note) Please do not change the analog to digital conversion clock setting during the analog to digital conversion.

Fig. 15-15 A/D Conversion Time

15.4 Description of Operations

15.4.1 Analog Reference Voltage

The "H" level of the analog reference voltage shall be applied to the VREFH pin, and the "L" level shall be applied to the VREFL pin. By writing "0" to the ADMOD1<VREFON> bit, a switched-on state of VREFH-VREFL can be turned into a switched-off state. To start AD conversion, make sure that you first write "1" to the <VREFON> bit, wait for 3 μs during which time the internal reference voltage should stabilize, and then write "1" to the ADMOD0<ADS> bit.

15.4.2 Selecting the Analog Input Channel

How the analog input channel is selected is different depending on A/D converter operation mode used.

(1) Normal AD conversion mode

- If the analog input channel is used in a fixed state (ADMOD0<SCAN>="0"):

One channel is selected from analog input pins AIN0 through AIN11 by setting ADMOD1<ADCH3 to 0> to an appropriate setting.

- If the analog input channel is used in a scan state (ADMOD0<SCAN>="1"):

One scan mode is selected from 12 scan modes by setting ADMOD1<ADCH3 to 0> and ADSCN to appropriate settings.

（2）Top-priority AD conversion mode

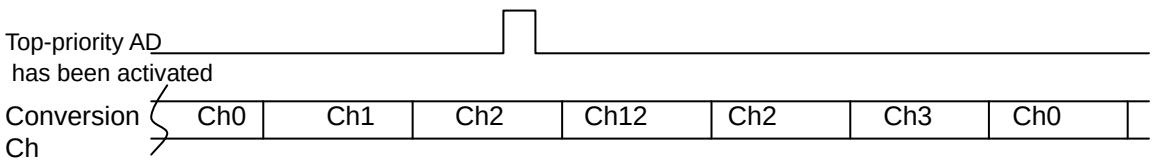
One channel is selected from analog input pins AIN0 through AIN11 by setting ADMOD2<HPADCH3 to 0> to an appropriate setting.

After a reset, ADMOD0<SCAN> is initialized to "0" and ADMOD1<ADCH3:0> is initialized to "0000." This initialization works as a trigger to select a fixed channel input through the AN0 pin. The pins that are not used as analog input channels can be used as ordinary input ports.

If top-priority AD conversion is activated during normal AD conversion, normal AD conversion is discontinued, top-priority AD conversion is executed and completed, and then normal AD conversion is resumed.

Example:

A case in which repeat-scan conversion is ongoing at channels AIN0 through AIN3 with ADMOD0<REPEAT:SCAN> set to "11" and ADMOD1<ADCH3:0> set to 0011, and top-priority AD conversion has been activated at AIN12 with ADMOD2<HPADCH3:0>=1100.



15.4.3 Starting A/D Conversion

Two types of A/D conversion are supported: normal AD conversion and top-priority AD conversion. Normal AD conversion is software activated by setting ADMOD0<ADS> to "1." Top-priority AD conversion is software activated by setting ADMOD2<HPADCE> to "1." 4 operation modes are made available to normal AD conversion. In performing normal AD conversion, one of these operation modes must be selected by setting ADMOD0<2:1> to an appropriate setting. For top-priority AD conversion, only one operation mode can be used: fixed channel single conversion mode. Normal AD conversion can be activated using the HW activation source selected by ADMOD3<ADHS>, and top-priority AD conversion can be activated using the HW activation source selected by ADMOD3<HADHS>. If this bit is "0," normal and top-priority AD conversions are activated in response to the input of a falling edge through the $\overline{\mathrm{ADTRG}}$ pin. If this bit is "1," normal AD conversion is activated in response to TB6RG0 generated by the 16-bit timer 6, and top-priority AD conversion is activated in response to TB5RG0 generated by the 16-bit timer 5. Software activation is still valid even after H/W activation has been authorized.

(Note) When an external trigger is used for the HW activation source of a top priority analog to digital conversion, an external trigger cannot usually be set for activating analog to digital conversion HW.

(Note) The TMPM330 disables the external trigger used for H/W activation. Therefore “0” cannot be set to <HADHS> and <ADHS>.

When normal A/D conversion starts, the A/D conversion Busy flag (ADMOD0<ADBF>) showing that A/D conversion is under way is set to "1." When top-priority A/D conversion starts, the A/D conversion Busy flag (ADMOD2<ADBFHP>) showing that A/D conversion is under way is set to "1." At that time, the value of the Busy flag for normal A/D conversion before the start of top-priority A/D conversion is retained. The value of the conversion completion flag EOCFN for normal A/D conversion before the start of top-priority A/D conversion can also be retained.

(Note) Normal A/D conversion must not be activated when top-priority A/D conversion is under way. In that case, the top-priority A/D conversion completion flag cannot be set, and the flag for previous normal A/D conversion cannot be cleared.

To reactivate normal A/D conversion while the conversion is under way, a software reset (ADMOD4<ADRST1:0>) must be performed before starting A/D conversion. The HW activation method must not be used to reactivate normal A/D conversion.

If ADMOD2<HPADCE> is set to "1" during normal A/D conversion, ongoing A/D conversion is discontinued and top-priority A/D conversion starts; specifically, A/D conversion (fixed channel single conversion) is executed for a channel designated by ADMOD2<3:0>. After the result of this top-priority A/D conversion is stored in the storage register ADREGSP, normal A/D conversion is resumed.

If HW activation of top-priority A/D conversion is authorized during normal A/D conversion, ongoing A/D conversion is discontinued when requirements for activation using a resource are met, and top-priority A/D conversion (fixed channel single conversion) starts for a channel designated by ADMOD2<3:0>. After the result of this top-priority A/D conversion is stored in the storage register ADREGSP, normal A/D conversion is resumed.

15.4.4 A/D Conversion Modes and A/D Conversion Completion Interrupts

For A/D conversion, the following four operation modes are supported. For normal A/D conversion, an operation mode can be selected by setting ADMOD0<2:1> to an appropriate setting. For top-priority A/D conversion, the fixed channel single conversion mode is automatically selected, irrespective of the ADMOD0<2:1> setting.

- Fixed channel single conversion mode
- Channel scan single conversion mode
- Fixed channel repeat conversion mode
- Channel scan repeat conversion mode

（1）Normal A/D conversion

An operation mode is selected with ADMOD0<REPEAT, SCAN>. As A/D conversion starts, ADMOD0<ADBFN> is set to "1." When specified A/D conversion is completed, the A/D conversion completion interrupt (INTAD) is generated, and ADMOD0<EOCF> showing the completion of A/D conversion is set to "1." If <REPEAT>="0," <ADBFN> returns to "0" concurrently with the setting of EOCF. If <REPEAT> is set to "1," <ADBFN> remains at "1" and A/D conversion continues.

1. Fixed channel single conversion mode

If ADMOD0 <REPEAT, SCAN> is set to "00," A/D conversion is performed in the fixed channel single conversion mode.

In this mode, A/D conversion is performed once for one channel selected. After A/D conversion is completed, ADMOD0<EOCF> is set to "1," ADMOD0<ADBF> is cleared to "0," and the interrupt request INTAD is generated. <EOCF> is cleared to "0" upon read.

2. Channel scan single conversion mode

If ADMOD0 <REPET,SCAN> is set to "01," A/D conversion is performed in the channel scan single conversion mode.

In this mode, A/D conversion is performed once for each scan channel selected. After A/D scan conversion is completed, ADMOD0<EOCF> is set to "1," ADMOD0<ADBF> is cleared to "0," and the interrupt request INTAD is generated. <EOCF> is cleared to "0" upon read.

3. Fixed channel repeat conversion mode

   If ADMOD0<REPEAT,SCAN> is set to "10," A/D conversion is performed in fixed channel repeat conversion mode.

   In this mode, A/D conversion is performed repeatedly for one channel selected. After A/D conversion is completed, ADMOD <EOCF> is set to "1." ADMOD0 <ADBF> is not cleared to "0." It remains at "1." The timing with which the interrupt request INTAD is generated can be selected by setting ADMOD0 <ITM1:0> to an appropriate setting. <EOCF> is set with the same timing as this interrupt INTAD is generated.

   <EOCF> is cleared to "0" upon read.

   With <ITM1:0> set to "00," an interrupt request is generated each time one A/D conversion is completed. In this case, the conversion results are always stored in the storage register ADREG08. After the conversion result is stored, EOCF changes to "1."

   With <ITM1:0> set to "01," an interrupt request is generated each time four A/D conversion are completed. In this case, the conversion results are sequentially stored in storage registers ADREG08 through ADREG3B. After the conversion results are stored in ADREG3B, <EOCF> is set to "1," and the storage of subsequent conversion results starts from ADREG08. <EOCF> is cleared to "0" upon read.

   With <ITM1:0> set to "10," an interrupt request is generated each time eight A/D conversions are completed. In this case, the conversion results are sequentially stored in storage registers ADREG08 through ADREG7F. After the conversion results are stored in ADREG7F, <EOCF> is set to "1," and the storage of subsequent conversion results starts from ADREG08.

   <EOCF> is cleared to "0" upon read.

4. Channel scan repeat conversion mode

   If ADMOD0 <REPEAT, SCAN> is set to "11," A/D conversion is performed in the channel scan repeat conversion mode.

   In this mode, A/D conversion is performed repeatedly for a scan channel selected. Each time one A/D scan conversion is completed, ADMOD0 <EOCF> is set to "1," and the interrupt request INTAD is generated. ADMOD0 <ADBF> is not cleared to "0." It remains at "1." <EOCF> is cleared to "0" upon read.

   To stop the A/D conversion operation in the repeat conversion mode (modes described in 3. and 4. above), write "0" to ADMOD0 <REPEAT>. When ongoing A/D conversion is completed, the repeat conversion mode terminates, and ADMOD0 <ADBF> is set to "0."

   Before switching from one mode to standby mode (such standby modes as IDLE, STOP, etc.), check that A/D conversion is not being executed. If A/D conversion is under way, you must stop it or wait until it is completed.

（2）Top-priority A/D conversion

Top-priority A/D conversion is performed only in fixed channel single conversion mode. The ADMOD0<REPEAT, SCAN> setting has no relevance to the top-priority A/D conversion operations or preparations. As activation requirements are met, A/D conversion is performed only once for a channel designated by ADMOD2<HPADCH3:0>. After the A/D conversion is completed, the top-priority A/D conversion completion interrupt is generated, ADMOD2<EOCFHP> is set to "1," and <ADBFHP> returns to "0." The EOCFHP Flag is cleared upon read.

Table 15-1 Relationships among A/D Conversion Modes, Interrupt Generation Timings and Flag Operations

Conversion mode	Interrupt generation timing	EOCF setting timing (see Note)	ADBF (after the interrupt is generated)	ADMOD0		
				ITM1:0	REPEAT	SCAN
Fixed channel single conversion	After conversion is completed	After conversion is completed	0	——	0	0
Fixed channel repeat conversion	Each time one conversion is completed	After one conversion is completed	1	00	1	0
	Each time four conversions are completed	After four conversions are completed	1	01		
	Each time eight conversions are completed	After eight conversions are completed	1	10		
Channel scan single conversion	After scan conversion is completed	After scan conversion is completed	0	——	0	1
Channel scan repeat conversion	Each time one scan conversion is completed	After one scan conversion is completed	1	——	1	1

(Note) EOCF is cleared upon read.

15.5 High-priority Conversion Mode

By interrupting ongoing normal A/D conversion, top-priority A/D conversion can be performed. Top-priority A/D conversion can be software activated by setting ADMOD2<HPADCE> to "1" or it can be activated using the HW resource by setting ADMOD3<7:6> to an appropriate setting. If top-priority A/D conversion has been activated during normal A/D conversion, ongoing normal A/D conversion is interrupted, and single conversion is performed for a channel designated by ADMOD2<3:0>. The result of single conversion is stored in ADREGSP, and the top-priority A/D conversion interrupt is generated. After top-priority A/D conversion is completed, normal A/D conversion is resumed; the status of normal A/D conversion immediately before being interrupted is maintained. Top-priority A/D conversion activated while top-priority A/D conversion is under way is ignored.

For example, if channel repeat conversion is activated for channels AN0 through AN8 and if <HPADCE> is set to "1" during AN3 conversion, AN3 conversion is suspended, and conversion is performed for a channel designated by <HPADC3:0>. After the result of conversion is stored in ADREGSP, channel repeat conversion is resumed, starting from AN3.

15.6 A/D Monitor Function

If ADCMPx<ADOBSVx> is set to "1," the A/D monitor function is enabled. If the value of the conversion result storage register specified by REGS<3:0> becomes larger or smaller ("larger" or "smaller" to be designated by ADOBIC) than the value of a comparison register, the A/D monitor function interrupt is generated. This comparison operation is performed each time a result is stored in a corresponding conversion result storage register, and the interrupt is generated if the conditions are met. Because storage registers assigned to perform the A/D monitor function are usually not read by software, overrun flag <OVRn> is always set and the conversion result storage flag <ADRnRF> is also set. To use the A/D monitor function, therefore, a flag of a corresponding conversion result storage register must not be used.

15.7 Storing and Reading A/D Conversion Results

A/D conversion results are stored in upper and lower A/D conversion result registers for normal A/D conversion (ADREG08H/L through ADRG7FH/L).

In fixed channel repeat conversion mode, A/D conversion results are sequentially stored in ADREG08H/L through ADREG7FH/L. If <ITM1:0> is so set as to generate the interrupt each time one A/D conversion is completed, conversion results are stored only in ADREG08H/L. If <ITM1:0> is so set as to generate the interrupt each time four A/D conversions are completed, conversion results are sequentially stored in ADREG08H/L through ADREG3BH/L.

Table 15-2 shows analog input channels and related A/D conversion result registers.

Table 15-2 Analog Input Channels and Related A/D Conversion Result Registers

Analog input channel (port A)	A/D conversion result register			
	Conversion modes other than shown to the right	Fixed channel repeat conversion mode (every one conversion)	Fixed channel repeat conversion mode (every four conversions)	Fixed channel repeat conversion mode (every eight conversions)
AN0	ADREG08H/L	ADREG08H/L fixed	ADREG08H/L → ADREG3BH/L	ADREG08H/L → ADREG4CH/L
AN1	ADREG19H/L			
AN2	ADREG2AH/L			
AN3	ADREG3BH/L			
AN4	ADREG4CH/L			
AN5	ADREG5DH/L			
AN6	ADREG6EH/L			
AN7	ADREG7FH/L			
AN8	ADREG08H/L			
AN9	ADREG19H/L			
AN10	ADREG2AH/L			
AN11	ADREG3BH/L			
AN12	ADREG4CH/L			

15.8 Data Polling

To process A/D conversion results without using interrupts, ADMOD0<EOCF> must be polled. If this flag is set, conversion results are stored in a specified A/D conversion result register. After confirming that this flag is set, read that conversion result storage register. In reading the register, make sure that you first read upper bits and then lower bits to detect an overrun. If OVRn is "0" and ADRnRF is "1" in lower bits, a correct conversion result has been obtained.

16 Watchdog Timer (Runaway Detection Timer)

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 timer detects a runaway, it generates a non-maskable interrupt to notify the CPU and outputs “0” from the output pin of the watchdog timer ($\overline{\text{WDTOUT}}$) to notify peripherals.

(Note) The TMPM330 does not include a watchdog timer out pin ($\overline{\text{WDTOUT}}$).

The watchdog timer can reset the CPU by connecting the $\overline{\text{WDTOUT}}$ pin to internal reset.

16.1 Configuration

Fig. 16-1 shows the block diagram of the watchdog timer

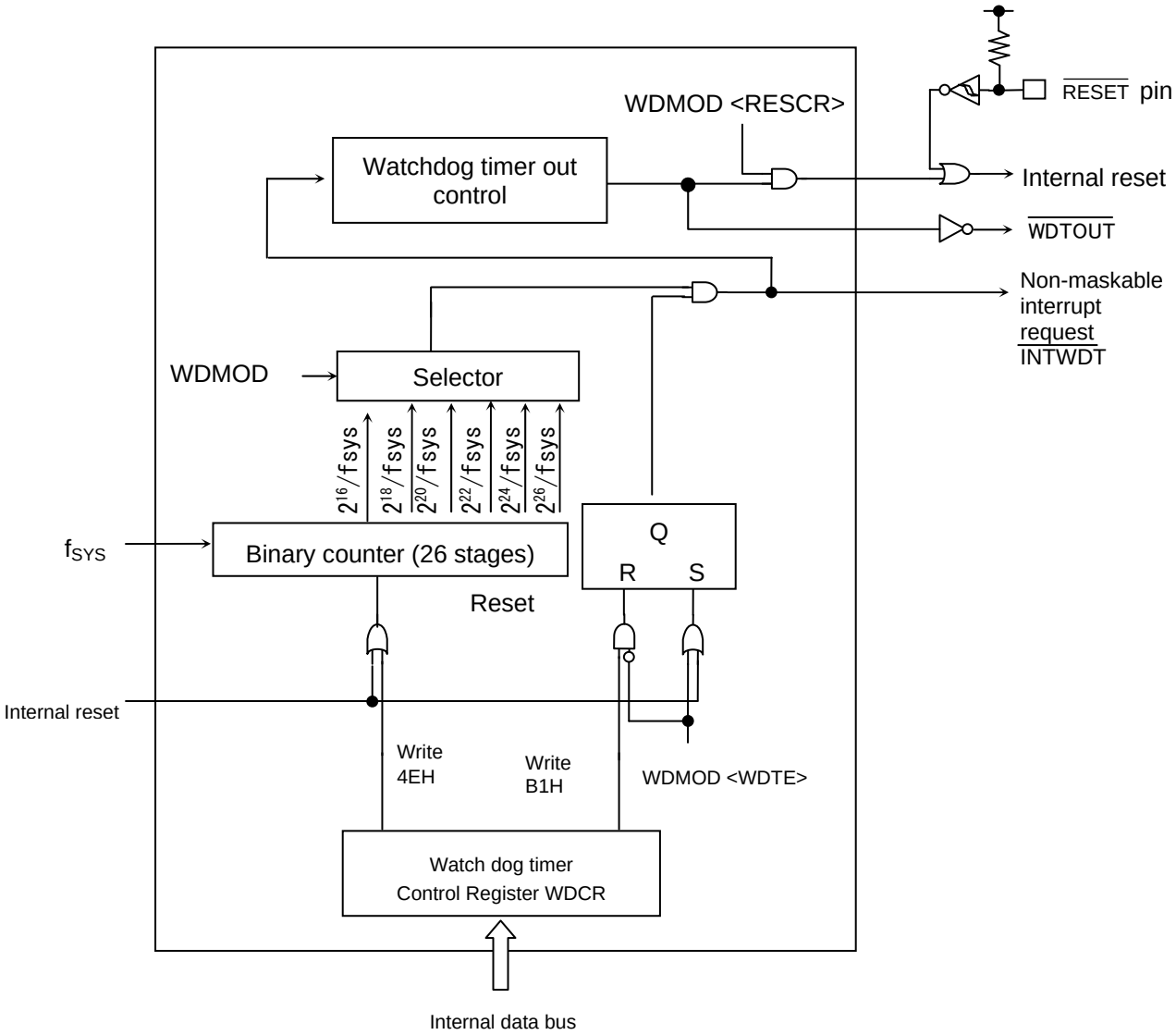


Fig. 16-1 Block Diagram of the Watchdog Timer

16.2 Watchdog Timer Interrupt

The watchdog timer consists of the binary counters that are arranged in 26 stages and work using the f_{SYS} system clock as an input clock. The outputs produced by these binary counters are 2^{16}, 2^{18}, 2^{20}, 2^{22}, 2^{24} are 2^{26}.By selecting one of these outputs with WDMOD <WDTP2:0>, a watchdog timer interrupt $\overline{\text{INTWDT}}$ can be generated and the $\overline{\text{WDTOUT}}$ is output when an overflow occurs, as shown in Fig. 16-2.

The $\overline{\text{INTWDT}}$ interrupt is one of the non-maskable interrupt factors. The $\overline{\text{INTWDT}}$ interrupt can be identified with the NMIFLG <NMIFLG0> bit in the clock generator

The output pin of the watchdog timer can reset the peripherals by outputting “0” caused by an overflow. The output is set to “1”if the watchdog timer is cleared (if the clear code 4EH is written to the WDCR register). The $\overline{\text{WDTOUT}}$ pin outputs “0” at normal mode unless the clear code is written to WDCR register.

(Note) The TMPM330 does not include a watchdog timer out pin ($\overline{\text{WDTOUT}}$).

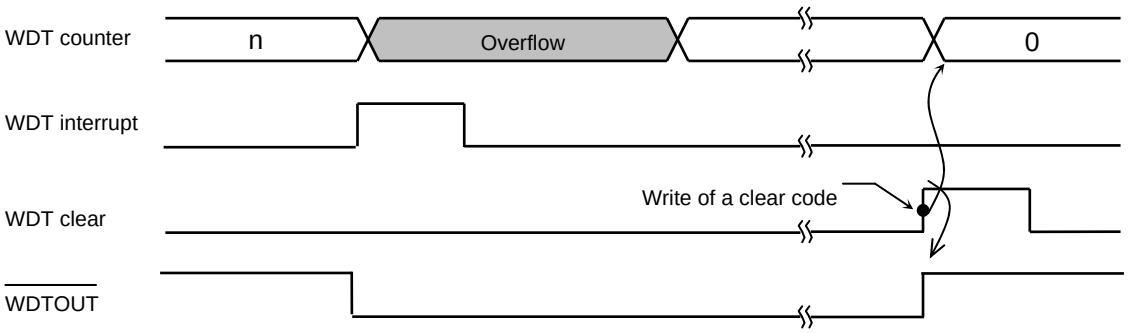


Fig. 16-2 Normal Mode

When an overflow occurs, resetting the chip itself is an option to choose. If the chip is reset, a reset is affected for a 32-state time, as shown in Fig. 16-3. If this reset is affected, the clock f_{SYS} that the clock gear generates by dividing the clock f_C of the high-speed oscillator by 1 is used as an input clock f_{SYS}.

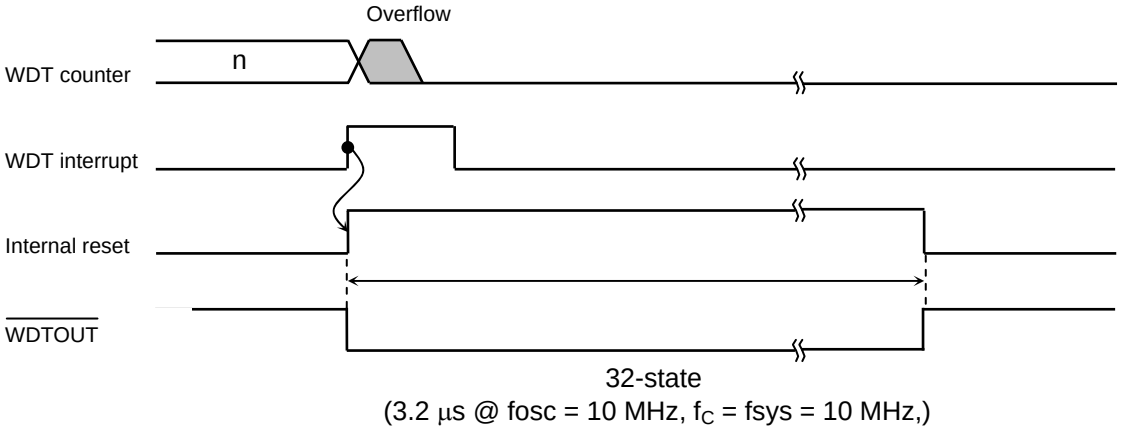


Fig. 16-3 Reset Mode

16.3 Control Registers

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

16.3.1 Watchdog Timer Mode Register (WDMOD)

1. Specifying the detection time of the watchdog timer <WDTP1: 0>

This is a 2-bit register for specifying the watchdog timer interrupt time for runaway detection. When a reset is effected, this register is initialized to WDMOD <WDTP1, 0> = "00." Fig. 16-4 shows the detection time of the watchdog timer.

2. Enabling/disabling the watchdog timer <WDTE>

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

To disable the watchdog timer, this bit must be set to "0" and, at the same time, the disable code (B1H) must be written to the WDCR register. This dual setting is intended to minimize the probability that the watchdog timer may inadvertently be disabled if a runaway occurs.

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

3. Watchdog timer out reset connection <RESCR>

Setting this bit to "1" enables the watch dog timer to be reset when a runaway is detected. Since a reset initializes this bit to "1," a counter overflow causes a reset.

16.3.2 Watchdog Timer Control Register (WDCR)

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

- Disabling control

By writing the disable code (B1H) to this WDCR register after setting WDMOD <WDTE> to "0," the watchdog timer can be disabled.

WDMOD	← 0 – – – – – – –	Clears WDTE to "0."
WDCR	← 1 0 1 1 0 0 0 1	Writes the disable code (B1H).

- Enabling control

Set WDMOD <WDTE> to "1".

- Watchdog timer clearing control

Writing the clear code (4EH) to the WDCR register clears the binary counter and allows it to resume counting.

WDCR	← 0 1 0 0 1 1 1 0	Writes the clear code (4EH)

(Note) Writing the disable code (BIH) clears the binary counter.

Watchdog Timer Mode Register

WDMOD (0x4004_0000)

	7	6	5	4	3	2	1	0
bit Symbol	WDTE	WDTP2	WDTP1	WDTP0		I2WDT	RESCR	
Read/Write	R/W	R/W			R	R/W		R/W
After reset	1	0	0	0		0	1	0
Function	WDT control 1: enable	Selects WDT detection time 000: $2^{16}/f_{SYS}$ 001: $2^{18}/f_{SYS}$ 010: $2^{20}/f_{SYS}$ 011: $2^{22}/f_{SYS}$ 100: $2^{24}/f_{SYS}$ 101: $2^{26}/f_{SYS}$ 110: Setting prohibited 111: Setting prohibited			"0" is read.	IDLE 0: Stop 1: Start	0: Generates NMI interrupt 1: Connects WDTOUT to reset	Write "0."

Watchdog timer out control

0	Generates NMI interrupt
1	Connects WDT out to reset

Detection time of watchdog timer @ fc = 40 MHz

SYSCR1 clock gear value <GEAR2:0>	Detection time of watchdog timer WDMOD<WDTP2:0>					
	000	001	010	011	100	101
000 (fc)	1.63 ms	6.55 ms	26.21 ms	104.86 ms	419.43 ms	1.68 s
100 (fc/2)	3.28 ms	13.11 ms	52.43 ms	209.72 ms	838.86 ms	3.36 s
101 (fc/4)	6.55 ms	26.21 ms	104.86 ms	419.43 ms	1.68 s	6.71 s
110 (fc/8)	13.11 ms	52.43 ms	209.72 ms	838.86 ms	3.36 s	13.42 s

Enable/disable control of the watchdog timer

0	Disable
1	Enable

Watchdog Timer Control Register

WDCR (0x4004_0004)

	7	6	5	4	3	2	1	0
bit Symbol	—							
Read/Write	W							
After reset	—							
Function	B1H : WDT disable code 4EH : WDT clear code							

Disable & clear of WDT

B1H	WDT disable code
4EH	WDT clear code
Others	—

Fig. 16-4 Watchdog Timer Registers

16.4 Control Register

The watchdog timer generates the $\overline{\text{INTWDT}}$ interrupt after a lapse of the detection time specified by the WDMOD <WDTP1, 0> register and outputs a signal at low level from the output pin of the watchdog timer ($\overline{\text{WDTOUT}}$). Before generating the $\overline{\text{INTWDT}}$ interrupt, the binary counter for the watchdog timer must be cleared to "0" using software (instruction). If the CPU malfunctions (runaways) due to noise or other disturbances and cannot execute the instruction to clear the binary counter, the binary counter overflows and the non-maskable interrupt is generated by the $\overline{\text{INTWDT}}$ interrupt. The CPU is able to recognize the occurrence of a malfunction (runaway) by identifying the non-maskable interrupt and to restore the faulty condition to normal by using a malfunction (runaway) countermeasure program. Additionally, it is possible to resolve the problem of a malfunction (runaway) of the CPU by connecting the watchdog timer out pin to reset pins of peripheral devices.

(Note) The TMPM330 does not include a watchdog timer out pin ($\overline{\text{WDTOUT}}$).

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

In STOP mode, the watchdog timer is reset and in an idle state. When the bus is released ($\overline{\text{BUSAK}}$ = "L"), it continues counting. In IDLE mode, its operation depends on the WDMOD <I2WDT> setting. Before putting it in IDLE mode, WDMOD <I2WDT> must be set to an appropriate setting, as required.

(Note) Releasing bus is disabled since no external bus feature is available on the TMPM330.

Example:

1. To clear the binary counter

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

2. To set the detection time of the watchdog timer to $2^{18}/f_{SYS.}$

		7	6	5	4	3	2	1	0
WDMOD	←	1	0	1	–	–	–	–	(

3. To disable the watchdog timer.

	7	6	5	4	3	2	1	0	
WDMOD (	0	(	(	(	(	(	(	(	Clears WDTE to "0"
WDCR (	1	0	1	1	0	0	0	1	Writes the disable code (B1H)

(Note 1) If the watchdog timer is operated when the high-frequency oscillator is idle, the system reset operation initiated by the watchdog timer becomes erratic due to the unstable oscillation of the high-frequency oscillator. Therefore, do not operate the watchdog timer when the high-frequency oscillator is idle.

(Note 2) The counter of the watchdog timer stops at the debug mode.

17 Real Time Clock (RTC)

17.1 Functions

1) Clock (hour, minute and second)
2) Calendar (month, week, date and leap year)
3) Selectable 12 (am/ pm) and 24 hour display
4) Time adjustment ＋ or － 30 seconds (by software)
5) Alarm (alarm output)
6) Alarm interrupt

17.2 Block Diagram

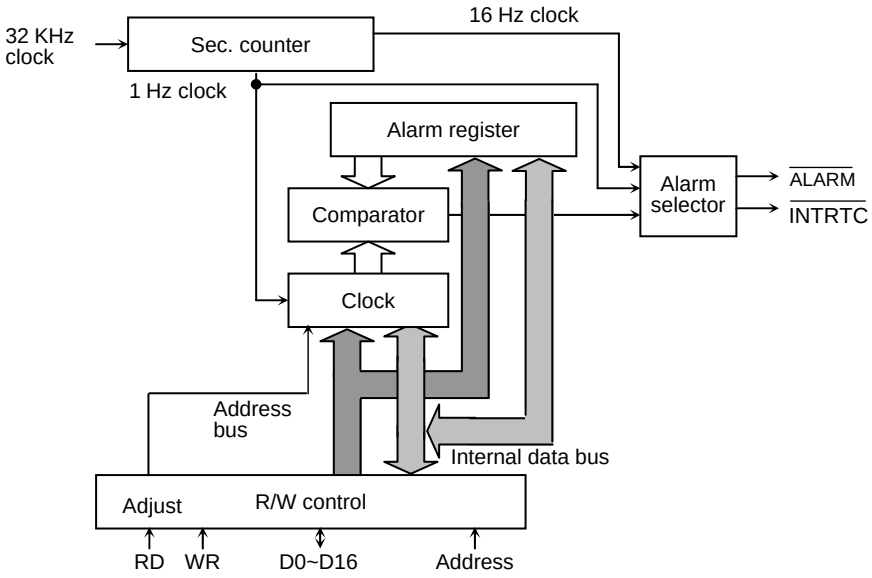


Fig. 17-1 Block Diagram

(Note 1) Western calendar year column:
This product uses only the final two digits of the year. The year following 99 is 00 years. Please take into account the first two digits when handling years in the western calendar.

(Note 2) Leap year:
A leap year is divisible by 4 excluding a year divisible by 100; the year divisible by 100 is not considered to be a leap year. Any year divisible by 400 is a leap year. This product is considered the year divisible by 4 to be a leap year and does not take into account the above exceptions. It needs adjustments for the exceptions.

17.3 Control Registers

Reset operation initializes the following registers:

- PAGER<PAGE>,<ADJUST>,<INTENA>
- RESTR<RSTALM>,<RSTTMR>,<DIS16HZ>,<DIS1HZ>

Other clock-related registers are not initialized by reset operation.

Before starting the RTC, set the time, month, day, day of the week, year and leap year in the relevant registers.

Caution is required in setting clock data, adjusting seconds or resetting the clock. Refer to “17.5.3 Entering the Low Power Consumption Mode”.

Table 17-1 PAGE0 (clock function) register

Symbol	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function	Read/Write
SECR	0x4004_0100H		40 sec	20 sec	10 sec	8 sec	4 sec	2 sec	1 sec	Second column	R/W
MINR	0x4004_0101H		40 min	20 min	10 min	8 min	4 min	2 min	1 min	Minute column	R/W
HOURR	0x4004_0102H			20 hours /PM/AM	10 hours	8 hours	4 hours	2 hours	1 hours	Hour column	R/W
DAYR	0x4004_0104H						W2	W1	W0	Day of the week column	R/W
DATER	0x4004_0105H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	0x4004_0106H				Oct.	Aug.	Apr.	Feb.	Jan.	Month column	R/W
YEARR	0x4004_0107H	Year 80	Year 40	Year 20	Year 10	Year 8	Year 4	Year 2	Year 1	Year column (lower two columns)	R/W
PAGER	0x4004_0108H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W, R/W
RESTR	0x4004_010CH	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write “0”.				Reset register	W only

(Note) Reading SECR, MINR, HOURR, DAYR, MONTHR, YEARR of PAGE0 captures the current state.

Table 17-2 PAGE1 (alarm function) registers

Symbol	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function	Read/Write
SECR	0x4004_0100H										
MINR	0x4004_0101H		40 min	20 min	10 min	8 min	4 min	2 min	1 min	Minute column	R/W
HOURR	0x4004_0102H			20 hours /PM/AM	10 hours	8 hours	4 hours	2 hours	1 hour	Hour column	R/W
DAYR	0x4004_0104H						W2	W1	W0	Day of the week column	R/W
DATER	0x4004_0105H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	0x4004_0106H								24/12	24-hour clock mode	R/W
YEARR	0x4004_0107H							Leap-year setting		Leap-year mode	R/W
PAGER	0x4004_0108H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W,R/W
RESTR	0x4004_010CH	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write “0”.				Reset register	W only

(Note 1) Reading SECR, MINR, HOURR, DAYR, MONTHR, YEARR of PAGE1 captures the current state.

(Note 2) SECR, MINR, HOURR, DAYR, MONTHR, YEARR of PAGE0 and YEARR of PAGE1 (for leap year) must be read twice and compare the data captured.

17.4 Detailed Description of Control Register

The RTC is not initialized by system reset. All registers must be initialized at the beginning of the program.

(1) Second column register (for PAGE0 only)

SECR	7	6	5	4	3	2	1	0
Bit symbol	—	SE6	SE5	SE4	SE3	SE2	SE1	SE0
Read/Write	R	R/W						
After reset	0	Undefined						
Function	“0” is read.	40 sec. column	20 sec. column	10 sec. column	8 sec. column	4 sec. column	2 sec. column	1 sec. column

0	0	0	0	0	0	0	0 sec
0	0	0	0	0	0	1	1 sec
0	0	0	0	0	1	0	2 sec
0	0	0	0	0	1	1	3 sec
0	0	0	0	1	0	0	4 sec
0	0	0	0	1	0	1	5 sec
0	0	0	0	1	1	0	6 sec
0	0	0	0	1	1	1	7 sec
0	0	0	1	0	0	0	8 sec
0	0	0	1	0	0	1	9 sec
0	0	1	0	0	0	0	10 sec
			:				
0	0	1	1	0	0	1	19 sec
0	1	0	0	0	0	0	20 sec
			:				
0	1	0	1	0	0	1	29 sec
0	1	1	0	0	0	0	30 sec
			:				
0	1	1	1	0	0	1	39 sec
1	0	0	0	0	0	0	40 sec
			:				
1	0	0	1	0	0	1	49 sec
1	0	1	0	0	0	0	50 sec
			:				
1	0	1	1	0	0	1	59 sec

Note) The setting other than listed above is prohibited.

(2) Minute column register (for PAGE0/1)

MINR		7	6	5	4	3	2	1	0
	Bit symbol	—	MI6	MI5	MI4	MI3	MI2	MI1	MI0
	Read/Write	R	R/W						
	After reset	0	Undefined						
	Function	“0” is read	40 min. column	20 min. column	10 min. column	8 min. column	4 min. column	2 min. column	1 min. column

0	0	0	0	0	0	0	0 min
0	0	0	0	0	0	1	1 min
0	0	0	0	0	1	0	2 min
0	0	0	0	0	1	1	3 min
0	0	0	0	1	0	0	4 min
0	0	0	0	1	0	1	5 min
0	0	0	0	1	1	0	6 min
0	0	0	0	1	1	1	7 min
0	0	0	1	0	0	0	8 min
0	0	0	1	0	0	1	9 min
0	0	1	0	0	0	0	10 min
			:				
0	0	1	1	0	0	1	19 min
0	1	0	0	0	0	0	20 min
			:				
0	1	0	1	0	0	1	29 min
0	1	1	0	0	0	0	30 min
			:				
0	1	1	1	0	0	1	39 min
1	0	0	0	0	0	0	40 min
			:				
1	0	0	1	0	0	1	49 min
1	0	1	0	0	0	0	50 min
			:				
1	0	1	1	0	0	1	59 min

Note) The setting other than listed above is prohibited.

(3) Hour column register (for PAGE0/1)

1. 24-hour clock mode (MONTHR<MO0>=“1”)

HOURR

	7	6	5	4	3	2	1	0
Bit symbol	—		HO5	HO4	HO3	HO2	HO1	HO0
Read/Write	R		R/W					
After reset	0		Undefined					
Function	“0”is read.		20 hour column	10 hour column	8 hour column	4 hour column	2 hour column	1 hour column

0	0	0	0	0	0	0 o’ clock
0	0	0	0	0	1	1 o’ clock
0	0	0	0	1	0	2 o’ clock
		:				
0	0	1	0	0	0	8 o’ clock
0	0	1	0	0	1	9 o’ clock
0	1	0	0	0	0	10 o’ clock
		:				
0	1	1	0	0	1	19 o’ clock
1	0	0	0	0	0	20 o’ clock
		:				
1	0	0	0	1	1	23 o’ clock

Note) The setting other than listed above is prohibited.

2. 12-hour clock mode (MONTHR<MO0>=“0”)

HOURR

	7	6	5	4	3	2	1	0
Bit symbol	—		HO5	HO4	HO3	HO2	HO1	HO0
Read/Write	R		R/W					
After reset	0		Undefined					
Function	“0”is read.		PM/AM	10 hour column	8 hour column	4 hour column	2 hour column	1 hour column

0	0	0	0	0	0	0 o’ clock (AM)
0	0	0	0	0	1	1 o’ clock
0	0	0	0	1	0	2 o’ clock
		:				
0	0	1	0	0	1	9 o’ clock
0	1	0	0	0	0	10 o’ clock
0	1	0	0	0	1	11 o’ clock
1	0	0	0	0	0	0 o’ clock (PM)
1	0	0	0	0	1	1 o’ clock

Note) The setting other than listed above is prohibited.

(4) Day of the week column register (for PAGE0/1)

DAYR

	7	6	5	4	3	2	1	0
Bit symbol	—					WE2	WE1	WE0
Read/Write	R					R/W		
After reset	0					Undefined		
Function	"0" is read.					W2	W1	W0

0	0	0	Sunday
0	0	1	Monday
0	1	0	Tuesday
0	1	1	Wednesday
1	0	0	Thursday
1	0	1	Friday
1	1	0	Saturday

Note) The setting other than listed above is prohibited.

(5) Day column register (PAGE0/1)

DATER

	7	6	5	4	3	2	1	0
Bit symbol	—		DA5	DA4	DA3	DA2	DA1	DA0
Read/Write	R		R/W					
After reset	0		Undefined					
Function	"0" is read.		Day 20	Day 10	Day 8	Day 4	Day 2	Day 1

0	0	0	0	0	0	0
0	0	0	0	0	1	1st day
0	0	0	0	1	0	2nd day
0	0	0	0	1	1	3rd day
0	0	0	1	0	0	4th day
		:				
0	0	1	0	0	1	9th day
0	1	0	0	0	0	10th day
0	1	0	0	0	1	11th day
		:				
0	1	1	0	0	1	19th day
1	0	0	0	0	0	20th day
		:				
1	0	1	0	0	1	29th day
1	1	0	0	0	0	30th day
1	1	0	0	0	1	31st day

Note 1) The setting other than listed above is prohibited.

Note 2) Do not set for non-existent days (e.g.: 30th Feb)

(6) Month column register (for PAGE0 only)

MONTHR

	7	6	5	4	3	2	1	0
Bit symbol	—			MO4	MO4	MO2	MO1	MO0
Read/Write	R			R/W				
After reset	0			Undefined				
Function	"0" is read.			10 months	8 months	4 months	2 months	1 month

0	0	0	0	1	January
0	0	0	1	0	February
0	0	0	1	1	March
0	0	1	0	0	April
0	0	1	0	1	May
0	0	1	1	0	June
0	0	1	1	1	July
0	1	0	0	0	August
0	1	0	0	1	September
1	0	0	0	0	October
1	0	0	0	1	November
1	0	0	1	0	December

Note) The setting other than listed above is prohibited.

(7) Selection of 24-hour clock or 12-hour clock (for PAGE1 only)

MONTHR

	7	6	5	4	3	2	1	0
Bit symbol	—							MO0
Read/Write	R							R/W
After reset	0							Undefined
Function	"0" is read.							1: 24-hour 0: 12-hour

（Note）Do not change the MONTHR<MO0> bit while the RTC is in operation.

(8) Year column register (for PAGE0 only)

YEARR

	7	6	5	4	3	2	1	0
Bit symbol	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0
Read/Write	R/W							
After reset	Undefined							
Function	80 years	40 years	20 years	10 years	8 years	4 years	2 years	1 year

0	0	0	0	0	0	0	0	00
0	0	0	0	0	0	0	1	01 years
0	0	0	0	0	0	1	0	02 years
0	0	0	0	0	0	1	1	03 years
0	0	0	0	0	1	0	0	04 years
0	0	0	0	0	1	0	1	05 years
:								
1	0	0	1	1	0	0	1	99 years

Note) The setting other than listed above is prohibited.

(9) Leap year register (for PAGE1 only)

YEARR

	7	6	5	4	3	2	1	0
Bit symbol	—						LEAP1	LEAP0
Read/Write	R						R/W	
After reset	0						Undefined	
Function	"0" is read.						00: leap year 01: one year after leap year 10: two years after leap year 11: three years after leap year	

0	0	Current year is a leap-year.
0	1	Current year is the year following a leap-year.
1	0	Current year is two years after a leap year.
1	1	Current year is three years after a leap year

(10) PAGE register (for PAGE0/1)

PAGER

A read-modify-write operation cannot be performed.

	7	6	5	4	3	2	1	0
Bit symbol	INTENA	—		ADJUST	ENATMR	ENAALM	—	PAGE
Read/Write	R/W	R		R/W	R/W		R	R/W
After reset	0	0		0	Undefined		0	0
Function	INTRTC 0: Disabled 1: Enabled	"0" is read.		[Write] 0: Don't care 1: Sets ADJUST request [Read] 0: No ADJUST request 1: ADJUST requested	Clock 0: Disabled 1: Enabled	ALARM 0: Disabled 1: Enabled	"0" is read.	PAGE selection

(Note) Keep the setting order of <ENATMR>, <ENAAML> and <INTENA> as shown in the example below. Ensure an interval of time between Clock/Alarm and interrupt.

Example: Clock setting/Alarm setting

		7	6	5	4	3	2	1	0	
PAGER	←	0	0	0	0	1	1	0	0	Enables Clock and alarm
PAGER	←	1	0	0	0	1	1	0	0	Enables interrupt

PAGE	0	Selects Page0
	1	Selects Page1

ADJUST	0	Don't care
	1	Adjusts seconds. The request is sampled when the sec. counter counts up. If the time elapsed is between 0 and 29 seconds, the sec. counter is cleared to "0". If the time elapsed is between 30 and 59 seconds, the min. counter is carried and sec. counter is cleared to "0". Reading this bit shows if ADJUST is requested or not.

(11) Reset register (for PAGE0/1)

RESTR

A read-modify-write operation cannot be performed.

	7	6	5	4	3	2	1	0
Bit symbol	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	–			
Read/Write	R/W				W			
After reset	1	1	0	0	Undefined			
Function	1 Hz 0: Enabled 1: Disabled	16 Hz 0: Enabled 1: Disabled	[Write] 0: Don't care 1: Clock reset [Read] 0: No RESET request 1: RESET requested	0: Don't care 1: Alarm reset	"0" is read.			

RSTALM	0	Unused
	1	Initializes alarm registers (Minute Column, Hour Column, Day Column and Day of the week Column) as follows. Minute: 00, Hour: 00, Day: 01, Day of the week: Sunday

RSTTMR	0	Unused
	1	Resets sec counter. Reading this bit shows if RESET is requested or not. The request is sampled using low-speed clock.

<DIS1HZ>	<DIS16HZ>	PAGER<ENAALM>	Interrupt source signal
1	1	1	Alarm
0	1	0	1Hz
1	0	0	16Hz
Others			Outputs "0".

17.5 Operational Description

The RTC incorporates a sec. counter that generates an 1Hz signal from a 32.768 KHz signal. The sec. counter operation must be taken into account when using the RTC.

17.5.1 Reading clock data

1. Using 1Hz interrupt

   The 1Hz interrupt is generated being synchronized with counting up of the sec. counter. Data can be read correctly if reading data after 1Hz interrupt occurred.

2. Using pair reading

   There is a possibility that the clock data may be read incorrectly if the internal counter operates carry during reading. To ensure correct data reading, read the clock data twice as shown below. A pair of data read successively needs to match.

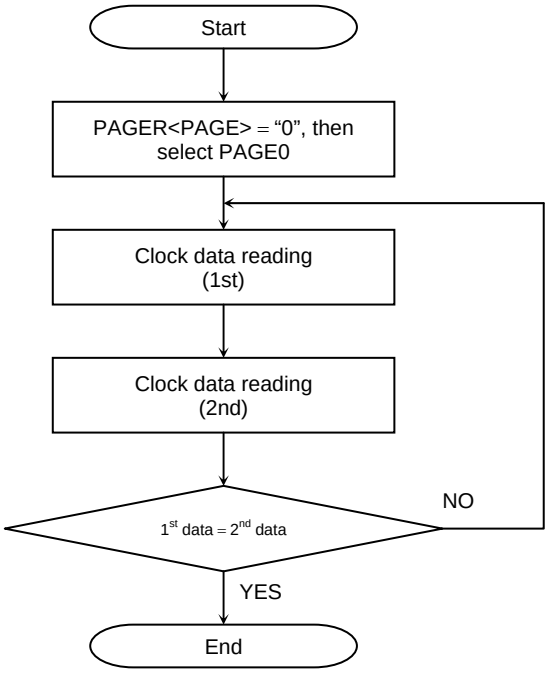


Fig. 17-2 Flowchart of the clock data reading

17.5.2 Writing clock data

A carry during writing ruins correct data writing. The following procedure ensures the correct data writing.

1. Using 1Hz interrupt

   The 1Hz interrupt is generated being synchronized with counting up of the sec. counter. If data is written in the time between 1Hz interrupt and subsequent one second count, it completes correctly.

2. Resetting counter

   Write data after resetting the sec. counter.

   The 1Hz-interrupt is generated one second after enabling the interrupt subsequent to counter reset. The time must be set within one second after the interrupt.

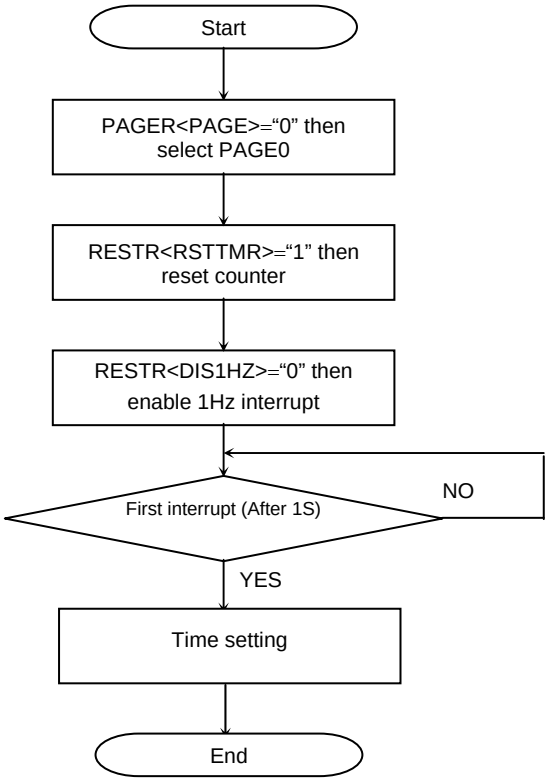


Fig. 17-3 Flowchart of the clock data writing

3. Disabling the clock

Writing “0” to PAGER<ENATMR> disables clock operation including a carry.

Stop the clock after the 1Hz-interrupt. The sec. counter keeps counting. Set the clock again and enable the clock within one second before next 1Hz-interrupt.

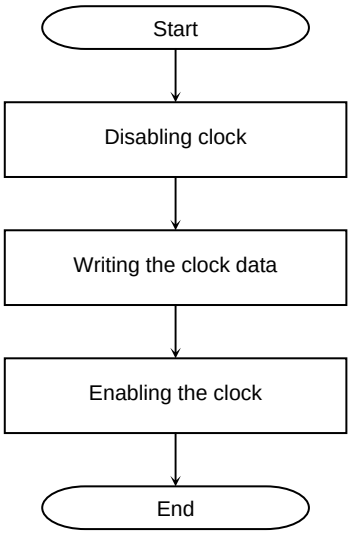


Fig. 17-4 Flowchart of the disabling clock

17.5.3 Entering the Low Power Consumption Mode

To enter SLEEP mode, in which the system clock stops, after changing clock data, adjusting seconds or resetting the clock, be sure to observe one of the following procedures:

1. After changing the clock setting registers, setting the PAGER<ADJUST> bit or setting the RESTR<RSTTMR> bit, wait for one second for an interrupt to be generated.
2. After changing the clock setting registers, setting the PAGER<ADJUST> bit or setting the RESTR<RSTTMR> bit, read the corresponding clock register values, <ADJUST> or <RSTTMR> to make sure that the setting you have made is reflected.

17.6 Alarm Function

By writing “1” to PAGER<PAGE>, the alarm function of the PAGE1 registers is enabled. One of the following three signals is output to the $\overline{\text{ALARM}}$ pin.

(1) “0” pulse (when the alarm register corresponds with the clock)

(2) 1Hz cycle “0” pulse

(3) 16Hz cycle “0” pulse

In any cases shown above, the INTRTC outputs one cycle pulse of low-speed clock. It outputs the INTRTC interrupt request simultaneously.

The INTRTC interrupt signal is falling edge triggered. Specify the falling edge as the active state in the CG Interrupt Mode Control Register.

(1) “0” pulse (when the alarm register corresponds with the clock)

“0” pulse is output to the $\overline{\text{ALARM}}$ pin when the values of the PAGE0 clock register and the PAGE1 alarm register correspond. The INTRTC interrupt is generated and the alarm is triggered.

<u>The alarm settings</u>

Initialize the alarm with alarm prohibited. Write “1” to RESTR<RSTALM>. It makes the alarm setting to be 00 minute, 00 hour, 01 day and Sunday.

Setting alarm for min., hour, date and day is done by writing data to the relevant PAGE1 register. Enable the alarm with the PAGER <ENAALM> bit. Enable the interrupt with the PAGER <INTENA> bit.

The following is an example program for outputting an alarm from the $\overline{\text{ALARM}}$ pin at noon (PM12:00) on Monday 5th.

		7	6	5	4	3	2	1	0	
PAGER	←	0	0	0	0	1	0	0	1	Disables alarm, sets PAGE1
RESTR	←	1	1	0	1	0	0	0	0	Initializes alarm
DAYR	←	0	0	0	0	0	0	0	1	Monday
DATAR	←	0	0	0	0	0	1	0	1	5th day
HOURR	←	0	0	0	1	0	0	1	0	Sets 12 o’clock
MINR	←	0	0	0	0	0	0	0	0	Sets 00 min.
PAGER	←	0	0	0	0	1	1	0	0	Enables alarm
PAGER	←	1	0	0	0	1	1	0	0	Enables interrupt

The above alarm works in synchronization with the low-speed clock. When the CPU is operating at high frequency oscillation, a maximum of one clock delay at 32 kHz (about 30µs) may occur for the time register setting to become valid.

(Note) To make the alarm work repeatedly (e.g. every Wednesday at 12:00), next alarm must be set during the INTRTC interrupt routine that is generated when the time set for the alarm matches the RTC count.

(2) 1Hz cycle “0” pulse

The RTC outputs a "0" pulse cycle of low-speed 1Hz clock to the $\overline{\text{ALARM}}$ pin by setting PAGER<INTENA>=1 after setting PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “0” and <DIS16HZ>= “1”. It generates an INTRTC interrupt simultaneously.

(3) 16Hz cycle “0” pulse

The RTC outputs a "0" pulse cycle of low-speed 16Hz clock to the $\overline{\text{ALARM}}$ pin by setting PAGER<INTENA>=1 after setting PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “1” and <DIS16HZ>= “0”. It generates an INTRTC interrupt simultaneously.

18 Flash Memory Operation

This section describes the hardware configuration and operation of the flash memory.

18.1 Flash Memory

18.1.1 Features

1) Memory capacity
The TMPM330FDFG/ TMPM330FYFG/ TMPM330FWFG devices contain flash memory. The memory sizes and configurations of each device are shown in the table below. Independent write access to each block is available. When the CPU is to access the internal flash memory, 32-bit data bus width is used.

2) Write/erase time
Writing is executed per page. The TMPM330FDFG/ TMPM330FYFG contain 128 words and the TMPM330FWFG contains 64 words in a page.
Page writing requires 1.25ms (typical) regardless of number of words.
A block erase requires 0.1 sec. (typical).
The following table shows write and erase time per chip.

Product Name	Memory Size	Block Configuration				# of Words	Write Time	Erase Time
		128KB	64KB	32KB	16KB			
TMPM330FDFG	512KB	3	1	2	-	128	1.28sec	0.4sec
TMPM330FYFG	256KB	1	1	2	-	128	0.64sec	0.4sec
TMPM330FWFG	128KB	-	1	1	2	64	0.64sec	0.2sec

(Note) The above values are theoretical values not including data transfer time.
The write time per chip depends on the write method to be used by the user.

3) Programming method
The onboard programming mode is available for the user to program (rewrite) the device while it is mounted on the user's board.

・The onboard programming mode

3-1) User boot mode
The user's original rewriting method can be supported.

3-2) Single boot mode
The rewriting method to use serial data transfer (Toshiba's unique method) can be supported.

Rewriting method
The flash memory included in this device is generally compliant with the applicable JEDEC standards except for some specific functions. Therefore, if the user is currently using an external flash memory device, it is easy to implement the functions into this device. Furthermore, the user is not required to build his/her own programs to realize complicated write and erase functions because such functions are automatically performed using the circuits already built-in the flash memory chip.
This device is also implemented with a read-protect function to inhibit reading flash memory data from any external writer device. On the other hand, rewrite protection is available only through command-based software programming; any hardware setting method to apply +12VDC is not supported. See chapter 19 for details of ROM protection and security function.

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

18.1.2 Block Diagram of the Flash Memory Section

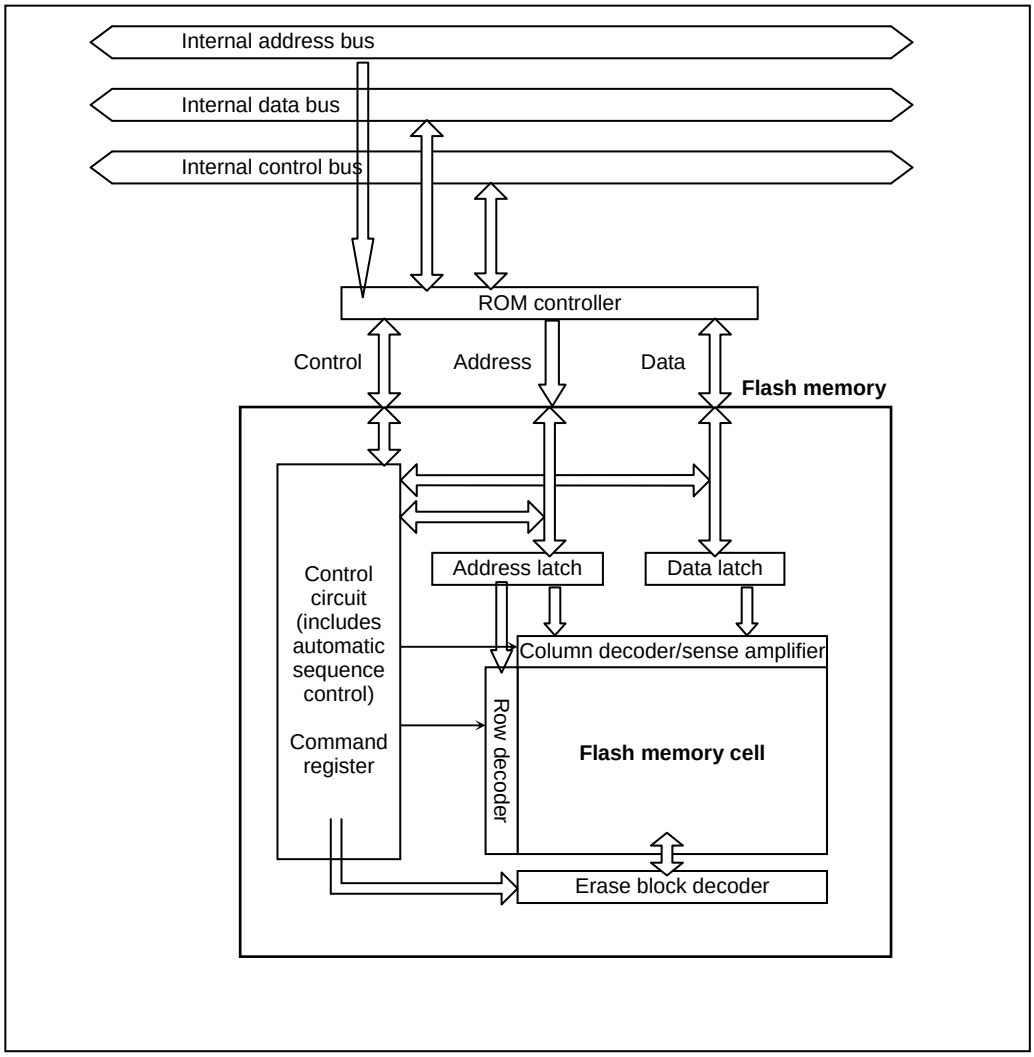


Fig. 18-1 Block Diagram of the Flash Memory Section

18.2 Operation Mode

This device has three operation modes including the mode not to use the internal flash memory.

Table 18-1 Operation Modes

Operation mode		Operation details
Single chip mode		After reset is cleared, it starts up from the internal flash memory.
	Normal mode	In this operation mode, two different modes, i.e., the mode to execute user application programs and the mode to rewrite the flash memory onboard the user's card, are defined. The former is referred to as "normal mode" and the latter "user boot mode. The user can uniquely configure the system to switch between these two modes. For example, the user can freely design the system such that the normal mode is selected when the port "A0" is set to "1" and the user boot mode is selected when it is set to "0." The user should prepare a routine as part of the application program to make the decision on the selection of the modes.
	User boot mode	
Single boot mode		After reset is cleared, it starts up from the internal Boot ROM (Mask ROM). In the Boot ROM, an algorithm to enable flash memory rewriting on the user's set through the serial port of this device is programmed. By connecting to an external host computer through the serial port, the internal flash memory can be programmed by transferring data in accordance with predefined protocols.

Among the flash memory operation modes listed in the above table, the User Boot mode and the Single Boot mode are the programmable modes. These two modes, the User Boot mode and the Single Boot mode, are referred to as "Onboard Programming" modes where onboard rewriting of internal flash memory can be made on the user's card.

Either the Single Chip or Single Boot operation mode can be selected by externally setting the level of the $\overline{\mathrm{BOOT}}$ (PH0) pin while the device is in reset status.

After the level is set, the CPU starts operation in the selected operation mode when the reset condition is removed. Regarding the $\overline{\mathrm{BOOT}}$ (PH0) pin, be sure not to change the levels during operation once the mode is selected.

The mode setting method and the mode transition diagram are shown below:

Table 18-2 Operation Mode Setting

Operation mode	Pin	
	$\overline{\mathrm{RESET}}$	$\overline{\mathrm{BOOT}}$ (PH0)
Single chip mode	0 → 1	1
Single boot mode	0 → 1	0

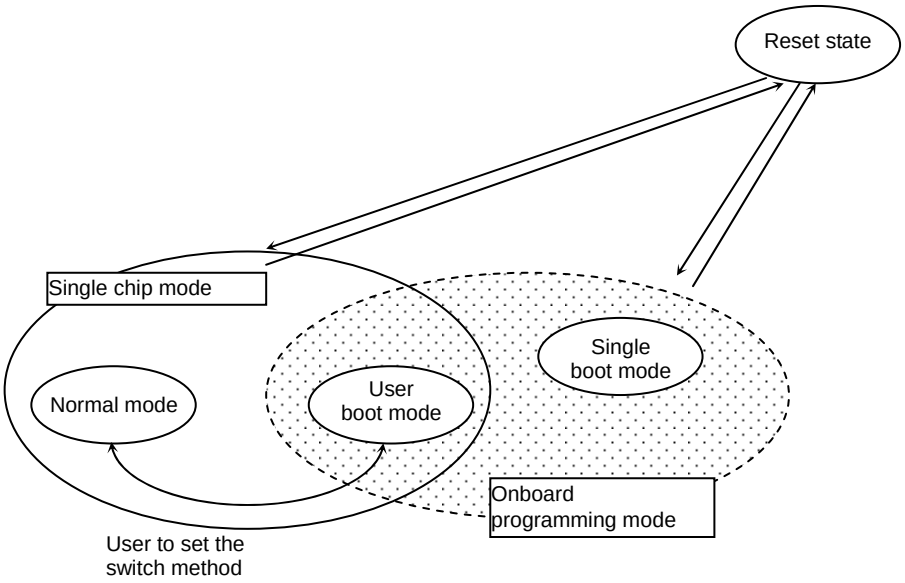


Fig. 18-2 Mode Transition Diagram

18.2.1 Reset Operation

To reset the device, ensure that the power supply voltage is within the operating voltage range, that the internal oscillator has been stabilized, and that the $\overline{\mathrm{RESET}}$ input is held at "0" for a minimum duration of 12 system clocks (0.3 μs with 40MHz operation; the "1/1" clock gear mode is applied after reset).

(Note 1) Regarding power-on reset of devices with internal flash memory;
for devices with internal flash memory, it is necessary to apply "0" to the $\overline{\mathrm{RESET}}$ inputs upon power on for a minimum duration of 300 microseconds regardless of the operating frequency.

(Note 2) While flash auto programming or deletion is in progress, at least 0.5 microseconds of reset period is required regardless of the system clock frequency. In this condition, it takes approx. 2 ms to enable reading after reset.

18.2.2 User Boot Mode (Single chip mode)

User Boot mode is to use flash memory programming routine defined by users. It is used when the data transfer buses for flash memory program code on the old application and for serial I/O are different. It operates at the single chip mode; therefore, a switch from normal mode in which user application is activated at the single chip mode to User Boot Mode for programming flash is required. Specifically, add a mode judgment routine to a reset program in the old application.

The condition to switch the modes needs to be set by using the I/O of M330 in conformity with the user's system setup condition. Also, flash memory programming routine that the user uniquely makes up needs to be set in the new application. This routine is used for programming after being switched to User Boot Mode. The execution of the programming routine must take place while it is stored in the area other than the flash memory since the data in the internal flash memory cannot be read out during delete/ writing mode. Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental corruption during subsequent Single-Chip (Normal mode) operations. All the interruption including a non-maskable are inhibited at User Boot Mode.

(1-A) and (1-B) are the examples of programming with routines in the internal flash memory and in the external memory. For a detailed description of the erase and program sequence, refer to 0 On-board Programming of Flash Memory (Rewrite/Erase).

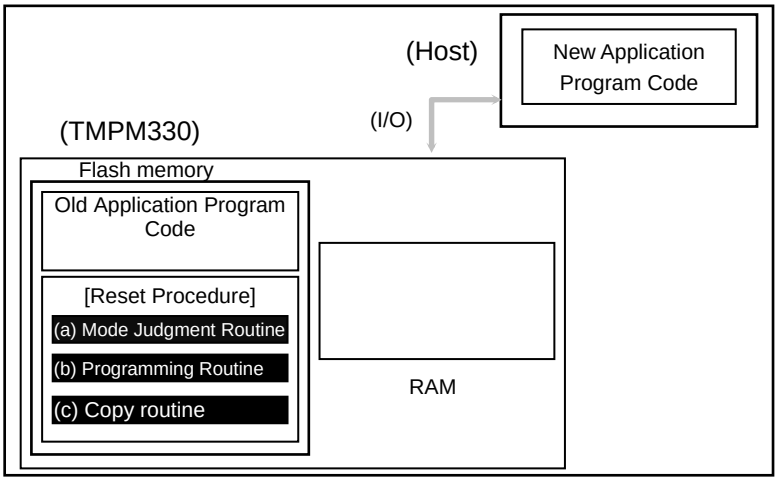
User Boot Mode

(1-A) Method 1: Storing a Programming Routine in the Flash Memory

(Step-1)

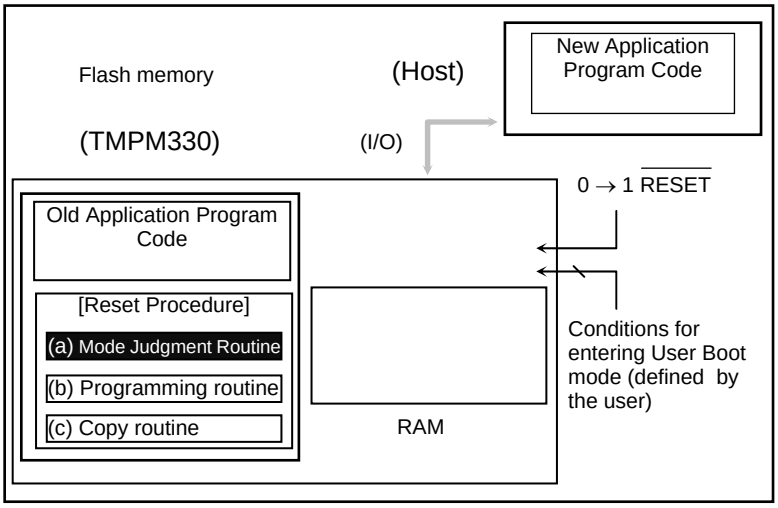
Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMPM330 on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

(a) Mode judgment routine: Code to determine whether or not to switch to User Boot mode
(b) Programming routine: Code to download new program code from a host controller and re-program the flash memory
(c) Copy routine: Code to copy the data described in (b) from the TMPM330 flash memory to either the TMPM330 on-chip RAM or external memory device.



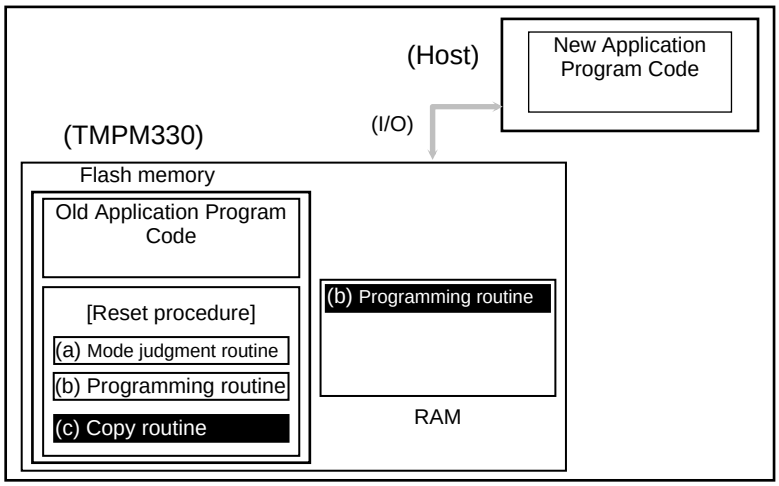
(Step-2)

After $\overline{\text{RESET}}$ is released, the reset procedure determines whether to put the TMPM330 flash memory in User Boot mode. If mode switching conditions are met, the flash memory enters User Boot mode. (All interrupts including NMI must be disabled while in User Boot mode.)



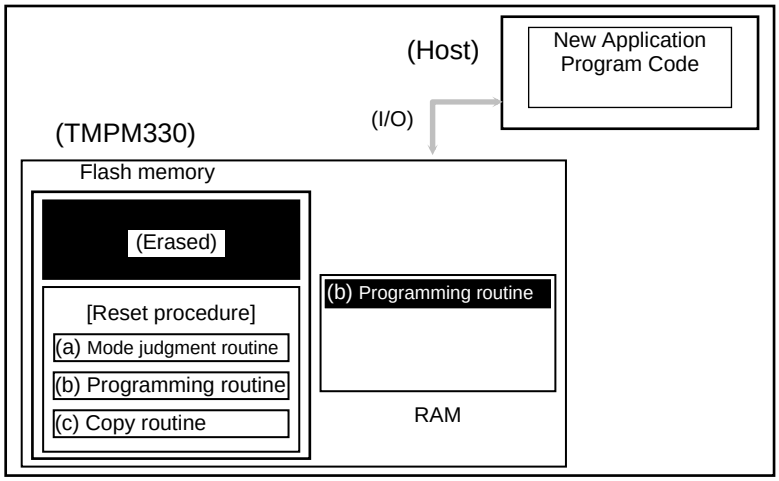
(Step-3)

Once transition to User Boot mode is occurred, execute the copy routine (c) to copy the flash programming routine (b) to the TMPM330 on-chip RAM.



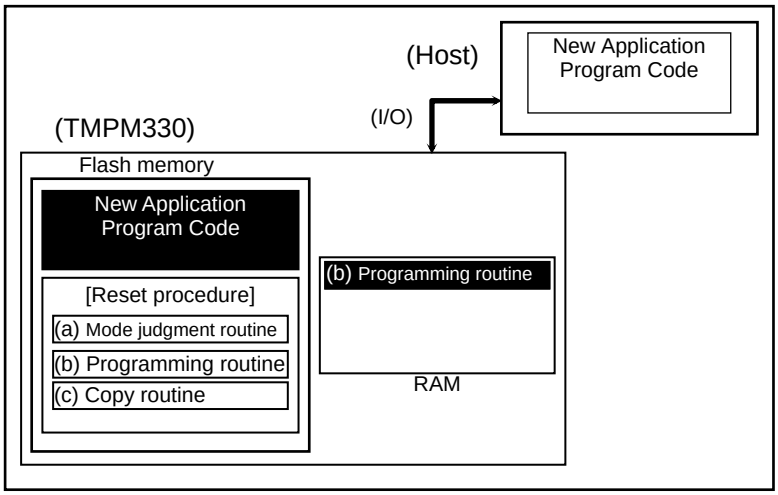
(Step-4)

Jump program execution to the flash programming routine in the on-chip RAM to erase a flash block containing the old application program code.



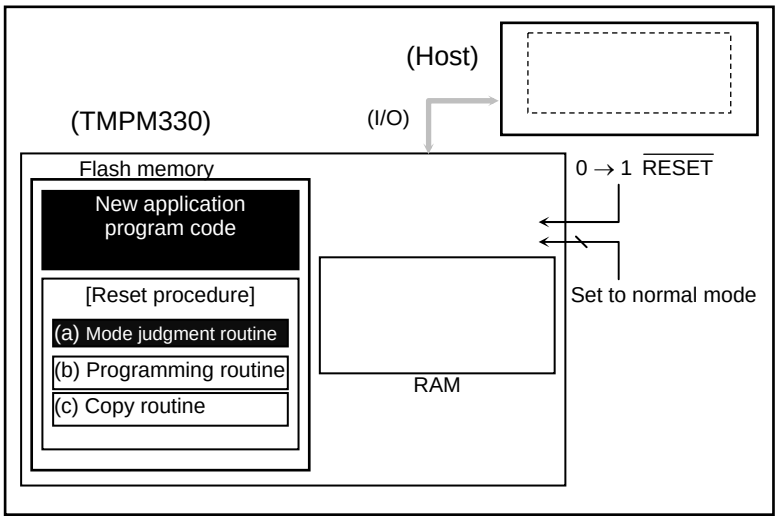
(Step-5)

Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. Once programming is complete, turn on the protection of that flash block.



(Step-6)

Set $\overline{\text{RESET}}$ to “0” to reset the TMPM330. Upon reset, the on-chip flash memory is put in Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.



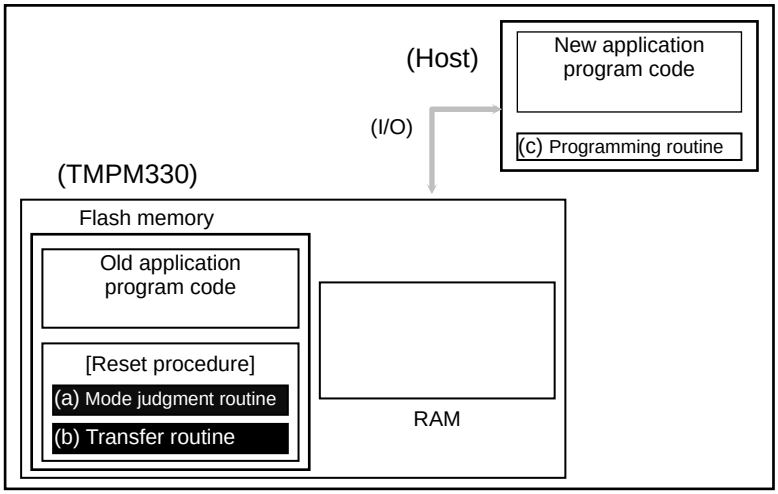
(1-B) Method 2: Transferring a Programming Routine from an External Host

(Step-1)

Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMPM330 on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

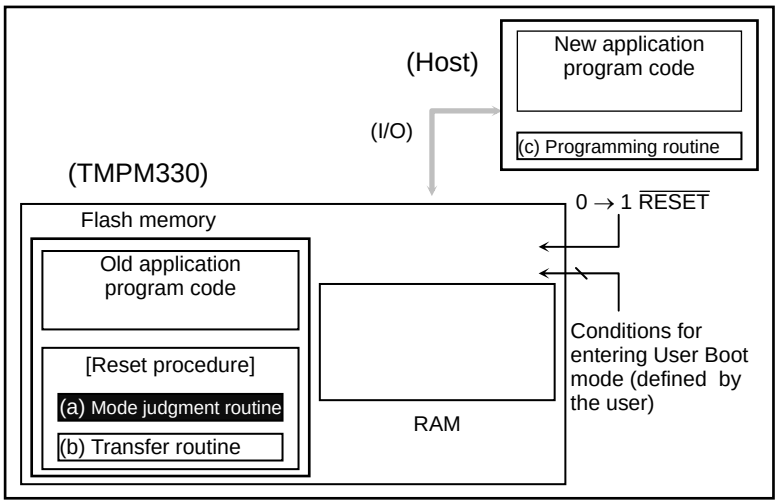
(a) Mode judgment routine: Code to determine whether or not to switch to User Boot mode
(b) Transfer routine: Code to download new program code from a host controller

Also, prepare a programming routine shown below on the host controller:
(c) Programming routine: Code to download new program code from an external host controller and re-program the flash memory



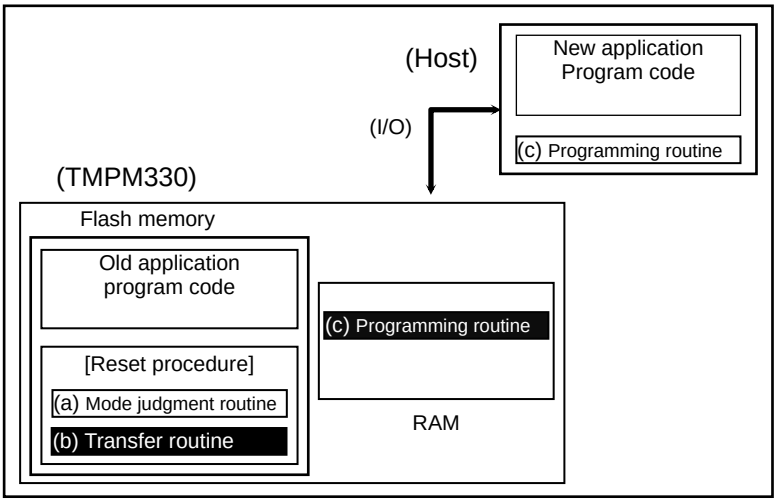
(Step-2)

After $\overline{\text{RESET}}$ is released, the reset procedure determines whether to put the TMPM330 flash memory in User Boot mode. If mode switching conditions are met, the flash memory enters User Boot mode. (All interrupts including NMI must be disabled while in User Boot mode).



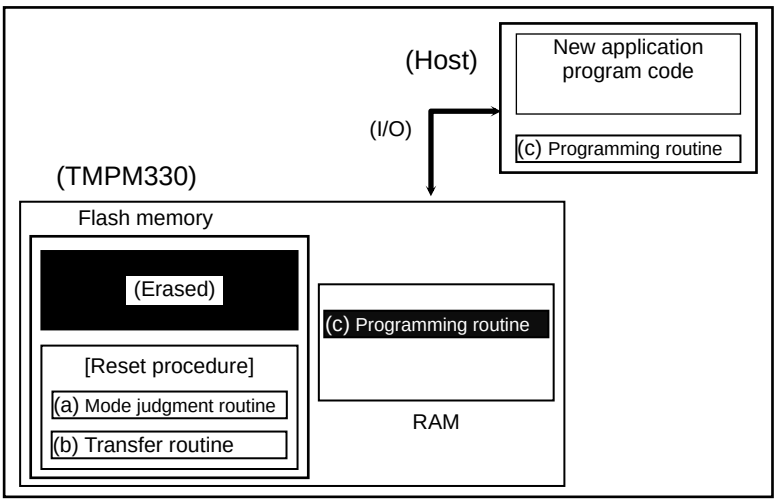
(Step-3)

Once User Boot mode is entered, execute the transfer routine (b) to download the flash programming routine (c) from the host controller to the TMPM330 on-chip RAM.



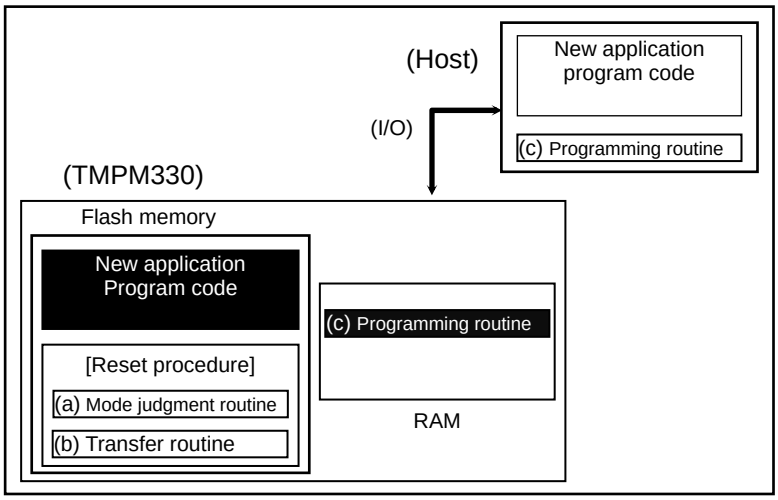
(Step-4)

Jump program execution to the flash programming routine in the on-chip RAM to erase a flash block containing the old application program code.



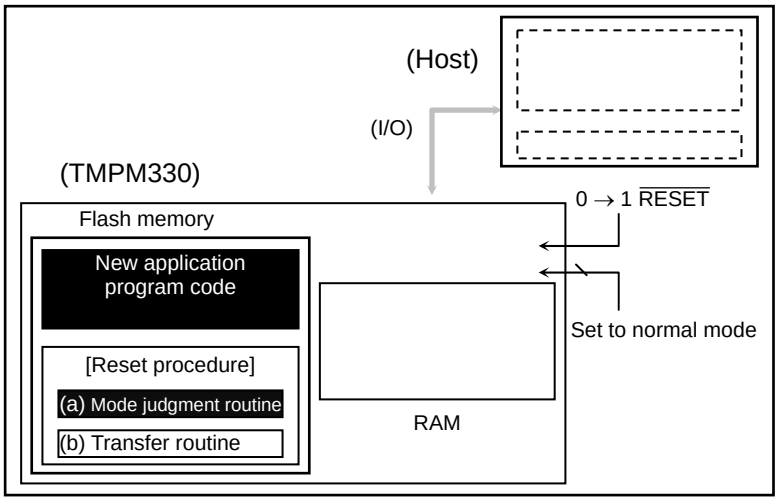
(Step-5)

Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. Once programming is complete, turn on the protection of that flash block.



(Step-6)

Set $\overline{\text{RESET}}$ to “0” low to reset the TMPM330. Upon reset, the on-chip flash memory is put in Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.



18.2.3 Single Boot Mode

In Single Boot mode, the flash memory can be re-programmed by using a program contained in the TMPM330 on-chip boot ROM. This boot ROM is a masked ROM. When Single Boot mode is selected upon reset, the boot ROM is mapped to the address region including the interrupt vector table while the flash memory is mapped to an address region different from it.

Single Boot mode allows for serial programming of the flash memory. Channel 0 of the SIO (SIO0) of the TMPM330 is connected to an external host controller. Via this serial link, a programming routine is downloaded from the host controller to the TMPM330 on-chip RAM. Then, the flash memory is re-programmed by executing the programming routine. The host sends out both commands and programming data to re-program the flash memory.

Communications between the SIO0 and the host must follow the protocol described later. To secure the contents of the flash memory, the validity of the application's password is verified before a programming routine is downloaded into the on-chip RAM. If password matching fails, the transfer of a programming routine itself is aborted.

As in the case of User Boot mode, all interrupts including the non-maskable interrupt (NMI) must be disabled in Single Boot mode while the flash memory is being erased or programmed. In Single Boot mode, the boot-ROM programs are executed in Normal mode.

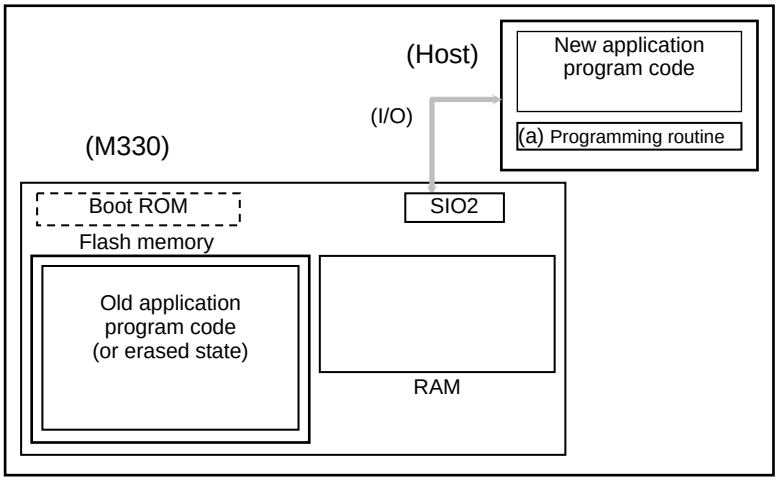
Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental corruption during subsequent Single-Chip (Normal mode) operations.

Single Boot Mode

(2-A) Using the Program in the On-Chip Boot ROM

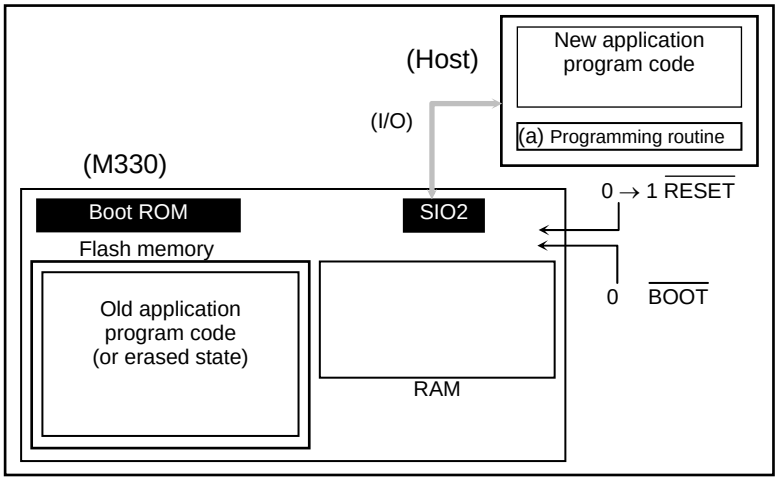
(Step-1)

The flash block containing the older version of the program code need not be erased before executing the programming routine. Since a programming routine and programming data are transferred via the SIO (SIO2), the SIO2 must be connected to a host controller. Prepare a programming routine (a) on the host controller.



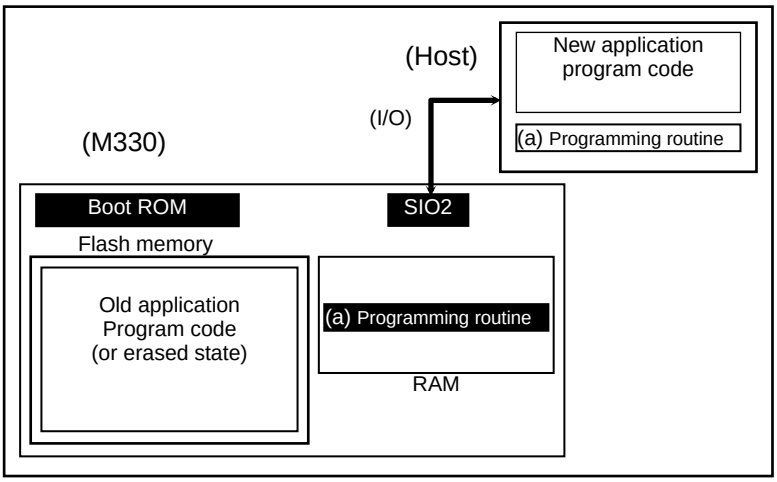
(Step-2)

Cancel the reset of the TMPM330 by setting the Single Boot mode pin to “0”, so that the CPU re-boots from the on-chip boot ROM. The 12-byte password transferred from the host controller via SIO2 is first compared to the contents of the special flash memory locations. (If the flash block has already been erased, the password is 0xFFFF).



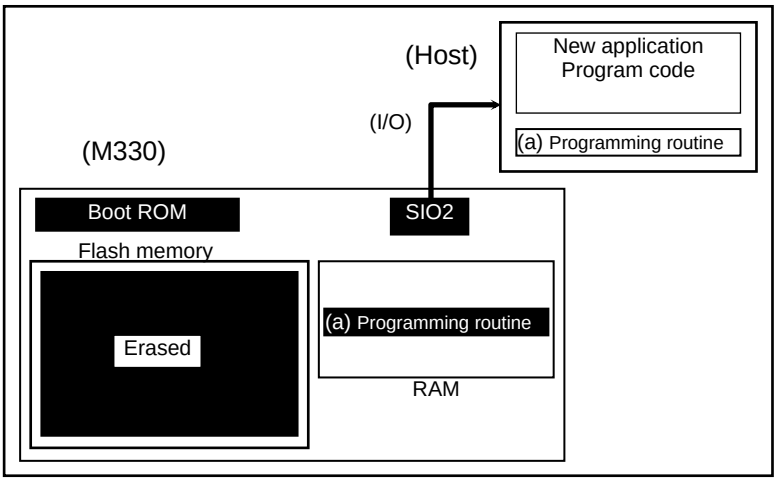
(Step-3)

If the password was correct, the boot program downloads, via the SIO0, the programming routine (a) from the host controller into the on-chip RAM of the TMPM330. The programming routine must be stored in the range from 0x2000_0400 to the end address of RAM.



(Step-4)

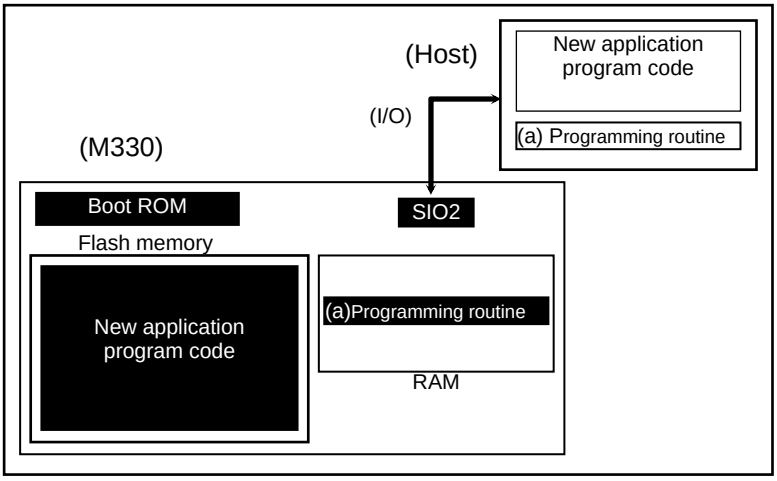
The CPU jumps to the programming routine (a) in the on-chip RAM to erase the flash block containing the old application program code. The Block Erase or Chip Erase command may be used.



(Step-5)

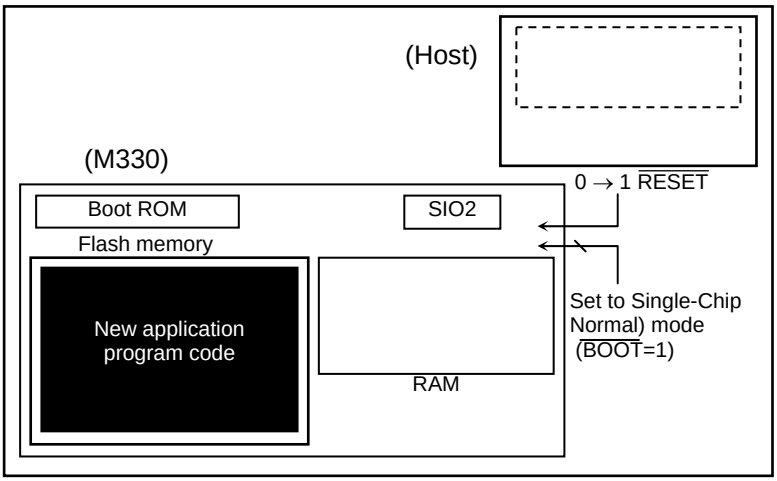
Next, the programming routine (a) downloads new application program code from the host controller and programs it into the erased flash block. Once programming is complete, protection of that flash block is turned on. It is not allowed to move program control from the programming routine (a) back to the boot ROM.

In the example below, new program code comes from the same host controller via the same SIO0 channel as for the programming routine. However, once the programming routine has begun to execute, it is free to change the transfer path and the source of the transfer. Create board hardware and a programming routine to suit your particular needs.



(Step-6)

When programming of the flash memory is complete, power off the board and disconnect the cable leading from the host to the target board. Turn on the power again so that the TMPM330 re-boots in Single-Chip (Normal) mode to execute the new program.



(1) Configuration for Single Boot Mode

To execute the on-board programming, boot the TMPM330 with Single Boot mode following the configuration shown below.

$\overline{BOOT}$ (PH0) = 0

$\overline{RESET}$ = 0 → 1

Set the $\overline{RESET}$ input to 0, and set the each $\overline{BOOT}$ (PH0) pins to values shown above, and then release RESET (high).

(2) Memory Map

Fig. 18-3 shows a comparison of the memory maps in Normal and Single Boot modes. In Single Boot mode, the internal flash memory is mapped to 0x3F80_0000 and later addresses, and the Internal boot ROM (Mask ROM) is mapped to 0x0000_0000 through 0x0000_1FFF.

The internal flash memory and RAM addresses of each device are shown below.

Product Name	Flash Size	RAM Size	Flash Address (Single Chip/ Single Boot Mode)	RAM Address
TMPM330FDFG	512KB	32KB	0x0000_0000 - 0x0007_FFFF 0x3F80_0000 - 0x3F87_FFFF	0x2000_0000 - 0x2000_7FFF
TMPM330FYFG	256KB	16KB	0x0000_0000 - 0x0003_FFFF (0x0007_FF00 – 0x0007_FF80)(Note) 0x3F80_0000 - 0x3F83_FFFF (0x3F87_FF00 – 0x3F87_FF80)(Note)	0x2000_0000 - 0x2000_3FFF
TMPM330FWFG	128KB	8KB	0x0000_0000 - 0x0001_FFFF 0x3F80_0000 - 0x3F81_FFFF	0x2000_0000 - 0x2000_1FFF

(Note) In addition to 256KB flash area, the TMPM330FYFG provides 128-word data/ password area (1 page) for Show Product Information command.

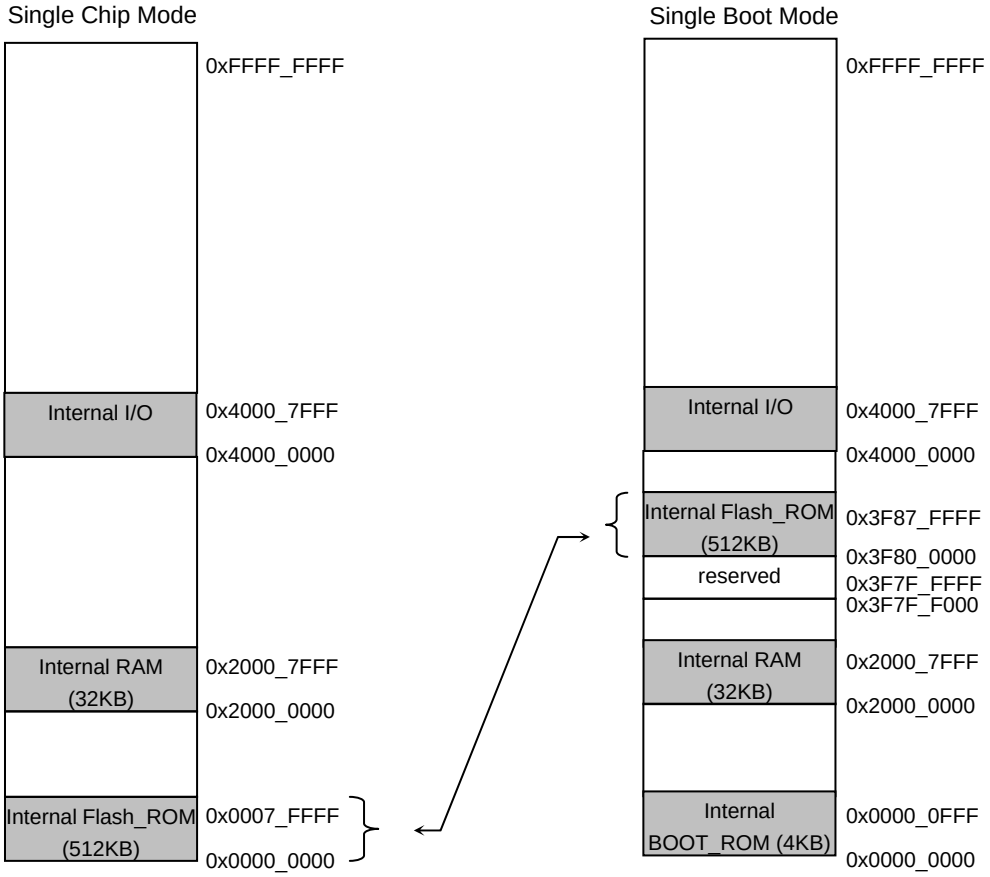


Fig. 18-3 Memory Maps for TMPM330FD

(3) Interface specification

In Single Boot mode, an SIO channel is used for communications with a programming controller. The same configuration is applied to a communication format on a programming controller to execute the on-board programming. Both UART (asynchronous) and I/O Interface (synchronous) modes are supported. The communication formats are shown below.

- UART communication

Communication channel	: SIO channel 0
Serial transfer mode	: UART (asynchronous), half -duplex, LSB fast
Data length	: 8 bit
Parity bits	: None
STOP bits	: 1 bit
Baud rate	: Arbitrary baud rate

- I/O interface mode

Communication channel	: SIO channel 0
Serial transfer mode	: I/O interface mode, full -duplex, LSB fast
Synchronization clock (SCLK0)	: Input mode
Handshaking signal	: PE4 configured as an output mode
Baud rate	: Arbitrary baud rate

Table 18-3 Required Pin Connections

Pins		Interface	
		UART	I/O Interface Mode
Power supply pins	REGVCC	○	○
	AVCC	○	○
	DVCC	○	○
	CVCC	○	○
	REGVSS	○	○
	AVSS	○	○
	DVSS	○	○
	CVSS	○	○
Mode-setting pin	$\overline{\text{BOOT}}$ (PH0)	○	○
Reset pin	$\overline{\text{RESET}}$	○	○
Communication pins	TXD0(PE0)	○	○
	RXD0(PE1)	○	○
	SCLK0(PE2)	x	○ (Input mode)
	PE4	x	○ (Output mode)

(4) Data Transfer Format

Table 18-4 and Table 18-6 to Table 18-9 illustrate the operation commands and data transfer formats at each operation mode. In conjunction with this section, refer to (6) Operation of Boot Program.

Table 18-4 Single Boot Mode Commands

Code	Command
10H	RAM transfer
20H	Show Flash Memory SUM
30H	Show Product Information
40H	Chip and protection bit erase

(5) Restrictions on internal memories

Single Boot Mode places restrictions on the internal RAM and ROM as shown in Table 18-5.

Table 18-5 Restrictions in Single Boot Mode

Memory	Details
Internal RAM	BOOT ROM is mapped to 0x2000_0000 to 0x2000_03FF. Store the RAM transfer program from 0x2000_0400 through the end address of RAM.
Internal ROM	The following addresses are assigned for storing software ID information and passwords. Storing program in these addresses is not recommendable. TMPM330FDFG : 0x3F87_FF00 - 0x3F87_FF0F TMPM330FYFG : 0x3F87_FF00 - 0x3F87_FF0F TMPM330FWFG : 0x3F81_FF00 - 0x3F81_FF0F

Table 18-6 Transfer Format for the RAM Transfer Command

	Byte	Data Transferred from the Controller to the TMPM330	Baud rate	Data Transferred from the TMPM330 to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode 86H For I/O Interface mode 30H	Desired baud rate (Note 2)	-
	2 byte	-		ACK for the serial operation mode byte For UART mode -Normal acknowledge 86H (The boot program aborts if the baud rate can not be set correctly.) For I/O Interface mode -Normal acknowledge 30H
	3 byte	Command code (10H)		-
	4 byte	-		ACK for the command code byte (Note 3) -Normal acknowledge 10H -Negative acknowledge × 1H -Communication error × 8H
	5 byte - 16 byte	Password sequence (12 bytes) 0x3F87_FF04~0x3F87_FF0F (FD/ FY) 0x3F81_FF04~0x3F81_FF0F (FW)		-
	17 byte	Check SUM value for bytes 5 - 16		-
	18 byte	-		ACK for the checksum byte (Note 3) -Normal acknowledge 10H -Negative acknowledge x1H -Communication error x8H
	19 byte	RAM storage start address 31 - 24		-
	20 byte	RAM storage start address 23 - 16		-
	21 byte	RAM storage start address 15 - 8		-
	22 byte	RAM storage start address 7 - 0		-
	23 byte	RAM storage byte count 15 - 8		-
	24 byte	RAM storage byte count 7 - 0		-
	25 byte	Check SUM value for bytes 19 - 24		-
	26 byte	-		ACK for the checksum byte (Note 3) -Normal acknowledge 10H -Negative acknowledge x1H -Communication error x8H
	27 byte ~ m byte	RAM storage data		-
	m + byte	Checksum value for bytes 27 - m		-
	m + byte	-		ACK for the checksum byte (Note 3) -Normal acknowledge 10H -Negative acknowledge x1H -Communication error x8H
RAM	m + byte	-		Jump to RAM storage start address

(Note 1) FD/ FY/ FW in the above table denotes the TMPM330FDFG, TMPM330FYFG and TMPM330FWFG respectively.

(Note 2) In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.

(Note 3) In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.

(Note 4) The 19th to 25th bytes must be within the RAM address range from 0x2000_0400 through the end address of RAM.

Table 18-7 Transfer Format for the Show Flash Memory Sum Command

	Byte	Data Transferred from the Controller to the TMPM330	Baud rate	Data Transferred from the TMPM330 to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode 86H For I/O Interface mode 30H	Desired baud rate (Note 1)	-
	2 byte	-		ACK for the serial operation mode byte For UART mode -Normal acknowledge 86H (The boot program aborts if the baud rate can not be set correctly.) For I/O Interface mode -Normal acknowledge 30H
	3 byte	Command code (20H)		-
	4 byte	-		ACK for the command code byte (Note 2) -Normal acknowledge 20H -Negative acknowledge x1H -Communication error x8H
	5 byte	-		SUM (upper byte)
	6 byte	-		SUM (lower byte)
	7 byte	-		Checksum value for bytes 5 and 6
	8 byte	(Wait for the next command code.)		-

(Note 1) In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.

(Note 2) In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.

Table 18-8 Transfer Format for the Show Product Information Command (1/2)

	Byte	Data Transferred from the Controller to the TMPM330	Baud rate	Data Transferred from the TMPM330 to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode 86H For I/O Interface mode 30H	Desired baud rate (Note 2)	-
	2 byte	-		ACK for the serial operation mode byte For UART mode -Normal acknowledge 86H (The boot program aborts if the baud rate can not be set correctly.) For I/O Interface mode -Normal acknowledge 30H
	3 byte	Command code (30H)		-
	4 byte	-		ACK for the command code byte (Note 3) -Normal acknowledge 30H -Negative acknowledge × 1H -Communication error × 8H
	5 byte	-		Flash memory data at address 0x3F87_FF00 (FD/ FY) 0x3F81_FF00 (FW)
	6 byte	-		Flash memory data at address 0x3F87_FF01 (FD/ FY) 0x3F81_FF01 (FW)
	7 byte	-		Flash memory data at address 0x3F87_FF02 (FD/ FY) 0x3F81_FF02 (FW)
	8 byte	-		Flash memory data at address 0x3F87_FF03 (FD/ FY) 0x3F81_FF03 (FW)
	9 byte - 20 byte	-		Product name (12-byte ASCII code) From the 9th byte: 'TMPM330FD_ _' (FD/ FY) 'TMPM330FW_ _' (FW)
	21 byte - 24 byte	-		Password comparison start address (4 bytes) From the 21st byte: 04H, FFH, 87H, 3FH (FD/ FY) 04H, FFH, 81H, 3FH (FW)
	25 byte - 28 byte	-		RAM start address (4 bytes) 00H, 00H, 00H and 20H from the 25th byte (FD/FY/FW)
	29 byte - 32 byte	-		Dummy data (4 bytes) 00H, 00H, 00H and 00H from the 29th byte (FD/FY/FW)
	33 byte - 36 byte	-		RAM end address (4 bytes) From the 33rd byte: FFH, 7FH, 00H, 20H (FD/ FY) (Note 4) FFH, 1FH, 00H, 20H (FW)
	37 byte- 40 byte	-		Dummy data (4 bytes) 00H, 00H, 00H and 00H from the 37th byte. (FD/FY/FW)
	41 byte - 44 byte	-		Dummy data (4 bytes) 00H, 00H, 00H and 00H from the 41st byte (FD/FY/FW)
	45 byte – 46 byte	-		Fuse information (2 bytes) 00H and 00H from the 45th byte. (FD/FY/FW)
	47 byte - 50 byte	-		Flash memory start address (4 bytes) 00H, 00H, 80H and 3FH from the 47th byte (FD/FY/FW)
	51 byte - 54 byte	-		Flash memory end address (4 bytes) From the 51st byte: FFH, FFH, 87H, 3FH (FD/ FY) (Note 5) FFH, FFH, 81H, 3FH (FW)
	55 byte- 56 byte	-		Flash memory block count (2 bytes) From the 55th byte: 06H, 00H (FD/FY) 04H, 00H (FW)

Table 18-8 Transfer Format for the Show Product Information Command (2/2)

	Byte	Data Transferred from the Controller to the TMPM330	Baud rate	Data Transferred from the TMPM330 to the Controller
Boot ROM	57 byte - 60 byte	-		Start address of a group of the same-size (16K) flash blocks (4 bytes) From 57th byte: 00H, 00H, 00H, 00H (FD/ FY) 00H, 00H, 80H, 3FH (FW)
	61 byte - 64 byte	-		Size (in halfwords) of the same-size (16K) flash blocks (4 bytes) 00H, 20H, 00H and 00H from the 61st byte
	65 byte	-		Number of flash blocks of the same size (1 byte) 00H (FD/ FY) 02H (FW)
	66 byte - 69 byte	-		Start address of a group of the same-size (32K) flash blocks (4 bytes) From 66th byte: 00H, 00H, 80H, 3FH (FD/ FY) 00H, 80H, 80H, 3FH (FW)
	70 byte - 73 byte			Size (in halfwords) of the same-size (32K) flash blocks (4 bytes) 00H, 40H, 00H and 00H from the 70th byte
	74 byte			Number of flash blocks of the same size (32K) (1 byte) 02H (FD/ FY) 01H (FW)
	75 byte - 78 byte			Start address of a group of the same-size (64K) flash blocks (4 bytes) 00H, 00H, 81H and 3FH from 75th byte
	79 byte - 82 byte			Size (in halfwords) of the same-size (64K) flash blocks (4 bytes) 00H, 80H, 00H and 00H from the 79th byte
	83 byte			Number of flash blocks of the same size (64K) (1 byte) From 83rd byte: 01H (FD/FY/FW)
	84 byte - 87 byte			Start address of a group of the same-size (128K) flash blocks (4 bytes) From 84th byte: 00H, 00H, 82H, 3FH (FD/ FY) 00H, 00H, 00H, 00H (FW)
	88 byte - 91 byte			Size (in halfwords) of the same-size (128K) flash blocks (4 bytes) 00H, 00H, 01H and 00H from the 88th byte
	92 byte			Number of flash blocks of the same size (128K) (1 byte) 03H (FD/ FY) (Note 6) 00H (FW)
	93 byte			Checksum value for bytes 5 - 92
	94 byte	(Wait for the next command code.)		-

(Note 1) FD/ FY/ FW in the above table denotes the TMPM330FDFG, TMPM330FYFG and TMPM330FWFG respectively.

(Note 2) In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.

(Note 3) In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.

(Note 4) The RAM actual end address of the TMPM330FYFG is 0x2000_3FFF.

(Note 5) The flash memory actual end address of the TMPM330FYFG is 0x3F83_FFFF.
In addition to 256KB flash area, the TMPM330FYFG provides 128-word data/ password area (0x3F87_FF00 – 0x3F87_FF80, 1 page) for Show Product Information command.

(Note 6) The actual number of the TMPM330FYFG flash blocks is one.

Table 18-9 Transfer Format for the Chip and Protection Bit Erase Command

	Byte	Data Transferred from the Controller to the TMPM330	Baud rate	Data Transferred from the TMPM330 to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode 86H For I/O Interface mode 30H	Desired baud rate (Note 1)	—
	2 byte	—		ACK for the serial operation mode byte For UART mode -Normal acknowledge 86H For I/O Interface mode -Normal acknowledge 30H (The boot program aborts if the baud rate can not be set correctly.)
	3 byte	Command code (40H)		—
	4 byte	—		ACK for the command code byte (Note 2) -Normal acknowledge 40H -Negative acknowledge × 1H -Communication error × 8H
	5 byte	Chip erase command code (54H)		—
	6 byte	—		ACK for the command code byte (Note 2) -Normal acknowledge 54H -Negative acknowledge × 1H -Communication error × 8H
	7 byte	—		ACK for the chip erase command code byte -Normal acknowledge 4FH -Negative acknowledge 4CH
	8 byte	(Wait for the next command code.)		—

(Note 1) In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.

(Note 2) In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.

(6) Operation of Boot Program

When Single Boot mode is selected, the boot program is automatically executed on startup. The boot program offers these three commands, of which the details are provided on the following subsections. The addresses described in this section are the virtual unless otherwise noted.

1. RAM Transfer command

The RAM Transfer command stores program code transferred from a host controller to the on-chip RAM and executes the program once the transfer is successfully completed. The user program RAM space can be assigned to the range from 0x2000_0400 to the end address of RAM, whereas the boot program area (0x2000_0000 ～ 0x2000_03FF) is unavailable. The user program starts at the assigned RAM address.

The RAM Transfer command can be used to download a flash programming routine of your own; this provides the ability to control on-board programming of the flash memory in a unique manner. The programming routine must utilize the flash memory command sequences described in Section 0.

Before initiating a transfer, the RAM Transfer command verifies a password sequence coming from the controller against that stored in the flash memory.

2. Show Flash Memory Sum command

The Show Flash Memory Sum command adds the entire contents of the flash memory together. The boot program does not provide a command to read out the contents of the flash memory. Instead, the Flash Memory Sum command can be used for software revision management.

3. Show Product Information command

The Show Product Information command provides the product name, on-chip memory configuration and the like. This command also reads out the contents of the flash memory locations at addresses shown below. In addition to the Show Flash Memory Sum command, these locations can be used for software revision management.

Product name	Area
TMPM330FDFG TMPM330FYFG	0x3F87_FF00 – 0x3F87_FF03
TMPM330FWFG	0x3F81_FF00 – 0x3F81_FF03

4. Chip and Protection Bit Erase command

This command erases the entire area of the flash memory automatically without verifying a password. All the blocks in the memory cell and their protection conditions are erased even when any of the blocks are prohibited from writing and erasing. When the command is completed, the SECBIT <SECBIT> bit is set to “1”.

This command serves to recover boot programming operation when a user forgets the password. Therefore password verification is not executed.

1) RAM Transfer Command (See Table 18-6)

1. The 1st byte specifies which one of the two serial operation modes is used. For a detailed description of how the serial operation mode is determined, see Determination of a Serial Operation Mode described later. If it is determined as UART mode, the boot program then checks if the SIO0 is programmable to the baud rate at which the 1st byte was transferred. During the first-byte interval, the RXE bit in the HSC0MOD register is cleare

- To communicate in UART mode
  Send, from the controller to the target board, 86H in UART data format at the desired baud rate. If the serial operation mode is determined as UART, then the boot program checks if the SIO0 can be programmed to the baud rate at which the first byte was transferred. If that baud rate is not possible, the boot program aborts, disabling any subsequent communications.

- To communicate in I/O Interface mode
  Send, from the controller to the target board, 30H in I/O Interface data format at 1/16 of the desired baud rate. Also send the 2nd byte at the same baud rate. Then send all subsequent bytes at a rate equal to the desired baud rate.

  In I/O Interface mode, the CPU sees the serial receive pin as if it were a general input port in monitoring its logic transitions. If the baud rate of the incoming data is high or the chip's operating frequency is high, the CPU may not be able to keep up with the speed of logic transitions. To prevent such situations, the 1st and 2nd bytes must be transferred at 1/16 of the desired baud rate; then the boot program calculates 16 times that as the desired baud rate. When the serial operation mode is determined as I/O Interface mode, the SIO0 is configured for SCLK Input mode. Beginning with the third byte, the controller must ensure that its AC timing restrictions are satisfied at the selected baud rate. In the case of I/O Interface mode, the boot program does not check the receive error flag; thus there is no such thing as error acknowledge (bit 3, x8H).

2. The 2nd byte, transmitted from the target board to the controller, is an acknowledge response to the 1st byte. The boot program echoes back the first byte: 86H for UART mode and 30H for I/O Interface mode.

- UART mode
  If the SIO0 can be programmed to the baud rate at which the 1st byte was transferred, the boot program programs the SC0BRCR and sends back 86H to the controller as an acknowledge. If the SIO0 is not programmable at that baud rate, the boot program simply aborts with no error indication. Following the 1st byte, the controller should allow for a time-out period of five seconds. If it does not receive 86H within the allowed time-out period, the controller should give up the communication. The boot program sets the RXE bit in the SC0MOD0 register to enable reception (1) before loading the SIO transmit buffer with 86H.

- I/O Interface mode
  The boot program programs the SC0MOD0 and SC0CR registers to configure the SIO0 in I/O Interface mode (clocked by the rising edge of SCLK0), writes 30H to the

SC0BUF. Then, the SIO0 waits for the SCLK0 signal to come from the controller. Following the transmission of the 1st byte, the controller should send the SCLK clock to the target board after a certain idle time (several microseconds). This must be done at 1/16 the desire baud rate. If the 2nd byte, which is from the target board to the controller, is 30H, then the controller should take it as a go-ahead. The controller must then deliver the 3rd byte to the target board at a rate equal to the desired baud rate. The boot program sets the RXE bit in the SC0MOD register to enable reception before loading the SIO transmit buffer with 30H.

3. The 3rd byte transmitted from the controller to the target board is a command. The code for the RAM Transfer command is 10H.

4. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H (bit 3) and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

   If the 3rd byte is equal to any of the command codes listed in Table 18-4, the boot program echoes it back to the controller. When the RAM Transfer command was received, the boot program echoes back a value of 10H and then branches to the RAM Transfer routine. Once this branch is taken, password verification is done. Password verification is detailed in a later section “Password”. If the 3rd byte is not a valid command, the boot program sends back x1H (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

5. The 5th to 16th bytes transmitted from the controller to the target board, are a 12-byte password. Each byte is compared to the contents of following addresses in the flash memory. The verification is started with the 5th byte and the smallest address in the designated area. If the password verification fails, the RAM Transfer routine sets the password error flag.

Product name	Area
TMPM330FDFG TMPM330FYFG	0x3F87_FF04 – 0x3F87_FF0F
TMPM330FWFG	0x3F81_FF04 – 0x3F81_FF0F

6. The 17th byte is a checksum value for the password sequence (5th to 16th bytes). To calculate the checksum value for the 12-byte password, add the 12 bytes together, drop the carries and take the two’s complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in a later section “Checksum Calculation”.

7. The 18th byte, transmitted from the target board to the controller, is an acknowledge

response to the 5th to 17th bytes. First, the RAM Transfer routine checks for a receive error in the 5th to 17th bytes. If there was a receive error, the boot program sends back 18H (bit 3) and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., all 1s). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 5th to 16th bytes must result in 00H (with the carry dropped). If it is not 00H, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 11H to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

Finally, the RAM Transfer routine examines the result of the password verification. The following two cases are treated as a password error. In these cases, the RAM Transfer routine sends back 11H (bit 0) to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

- Irrespective of the result of the password comparison, all the 12 bytes of a password in the flash memory are the same value other than FFH.

Not the entire password bytes transmitted from the controller matched those contained in the flash memory.

When all the above verification has been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

8. The 19th to 22nd bytes, transmitted from the controller the target board, indicate the start address of the RAM region where subsequent data (e.g., a flash programming routine) should be stored. The 19th byte corresponds to bits 31–24 of the address and the 22nd byte corresponds to bits 7–0 of the address.

9. The 23rd and 24th bytes, transmitted from the controller to the target board, indicate the number of bytes that will be transferred from the controller to be stored in the RAM. The 23rd byte corresponds to bits 15–8 of the number of bytes to be transferred, and the 24th byte corresponds to bits 7–0 of the number of bytes.

10. The 25th byte is a checksum value for the 19th to 24th bytes. To calculate the checksum value, add all these bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in a later section "Checksum Calculation".

11. The 26th byte, transmitted from the target board to the controller, is an acknowledge response to the 19th to 25th bytes of data. First, the RAM Transfer routine checks for a receive error in the 19th to 25th bytes. If there was a receive error, the RAM Transfer routine sends back 18H and returns to the command wait state (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., all 1s). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 19th to 25th bytes must result in 00H (with the carry dropped). If it is not 00H, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 11H to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

- The RAM storage start address must be within the range of 0x2000_0400 to the end address of RAM.

When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

12. The 27th to mth bytes from the controller are stored in the on-chip RAM of the TMPM330. Storage begins at the address specified by the 19th–22nd bytes and continues for the number of bytes specified by the 23rd–24th bytes.

13. The (m+1) th byte is a checksum value. To calculate the checksum value, add the 27th to mth bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in a later section "Checksum Calculation".

14. The (m+2) th byte is a acknowledge response to the 27th to (m+1) th bytes.

    First, the RAM Transfer routine checks for a receive error in the 27th to (m+1) th bytes. If there was a receive error, the RAM Transfer routine sends back 18H (bit 3) and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., all 1s). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

    Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 27th to (m+1) th bytes must result in 00H (with the carry dropped). If it is not 00H, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 11H (bit 0) to the controller and returns to the command wait state (i.e., the 3rd byte) again. When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

15. If the (m+2) th byte was a normal acknowledge response, a branch is made to the address specified by the 19th to 22nd bytes.

2) Show Flash Memory Sum Command (See Table 18-7)

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Flash Memory Sum command is 20H.

3. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

   If the 3rd byte is equal to any of the command codes listed in Table 18-4, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 20H and then branches to the Show Flash Memory Sum routine. If the 3rd byte is not a valid command, the boot program sends back x1H (bit 0) to the controller and returns to the command wait state (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

4. The Show Flash Memory Sum routine adds all the bytes of the flash memory together. The 5th and 6th bytes, transmitted from the target board to the controller, indicate the upper and lower bytes of this total sum, respectively. For details on sum calculation, see a later section “Calculation of the Show Flash Memory Sum Command”.

5. The 7th byte is a checksum value for the 5th and 6th bytes. To calculate the checksum value, add the 5th and 6th bytes together, drop the carry and take the two’s complement of the sum. Transmit this checksum value from the controller to the target board.

6. The 8th byte is the next command code.

3) Show Product Information Command (See Table 18-8)

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Product Information command is 30H.

3. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

   If the 3rd byte is equal to any of the command codes listed in Table 18-4, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 30H and then branches to the Show Flash Memory Sum routine. If the 3rd byte is not a valid command, the boot program sends back x1H (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

4. The 5th to 8th bytes, transmitted from the target board to the controller, are the data read from addresses shown below in the flash memory. Software version management is possible by storing a software ID in these locations.

Product name	Area
TMPM330FDFG TMPM330FYFG	0x3F87_FF00 – 0x3F87_FF03
TMPM330FWFG	0x3F81_FF00 – 0x3F81_FF03

5. The 9th to 20th bytes, transmitted from the target board to the controller, indicate the product name as shown below (where [ ] is a space) in ASCII code.

Product name	Code
TMPM330FDFG TMPM330FYFG	T, M, P, M, 3, 3, 0, F, D, _, [ ], _
TMPM330FWFG	T, M, P, M, 3, 3, 0, F, D, _, [ ], _

6. The 21st to 24th bytes, transmitted from the target board to the controller, indicate the start address of the flash memory area containing the password. Each product has own start address shown below.

Product name	Address
TMPM330FDFG TMPM330FYFG	04H, FFH, 87H, 3FH
TMPM330FWFG	04H, FFH, 81H, 3FH

7. The 25th to 28th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip RAM, i.e., 00H, 00H, FDH, FFH.

8. The 29th to 32nd bytes, transmitted from the target board to the controller, are dummy data (00H, 00H, 00H and 00H).

9. The 33rd to 36th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip RAM. Each product has own start address shown below.

Product name	Address
TMPM330FDFG TMPM330FYFG	FFH, 7FH, 00H, 20H
TMPM330FWFG	FFH, 1FH, 00H, 20H

(Note) The RAM actual end address of the TMPM330FYFG is 0x2000_3FFF.

10. The 37th to 40th bytes, transmitted from the target board to the controller, are 00H, 00H, 00H and 00H. The 41st to 44th bytes, transmitted from the target board to the controller, are FFH, EFH, FDH and FFH.

11. The 45th and 46th bytes transmitted are 01H, 00H.

12. The 47th to 50th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip flash memory, are 00H, 00H, 80H, and 3FH.

13. The 51st to 54th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip flash memory. Each product has own start address shown below.

Product name	Address
TMPM330FDFG TMPM330FYFG	FFH, FFH, 87H, 3FH
TMPM330FWFG	FFH, FFH, 81H, 3FH

(Note) The flash memory actual end address of the TMPM330FYFG is 0x3F83_FFFF.
In addition to 256KB flash area, the TMPM330FYFG provides 128-word data/ password area (0x3F87_FF00 – 0x3F87_FF80, 1 page) for Show Product Information command.

14. The 55th to 56th bytes, transmitted from the target board to the controller, indicate the number of flash blocks available. Each product transmits own number shown below.

Product name	Number of flash blocks
TMPM330FDFG	06H, 00H
TMPM330FYFG	06H, 00H
TMPM330FWFG	04H, 00H

15. The 57th to 83rd bytes, transmitted from the target board to the controller, contain information about the flash blocks. Flash blocks of the same size are treated as a group. Information about the flash blocks indicate the start address of a group, the size of the blocks in that group (in halfwords) and the number of the blocks in that group.

    The 57th to 65th bytes are the information about the 16-kbyte blocks. The 66th to 74th bytes are the information about the 32-kbyte blocks. The 75th to 83rd bytes are the information about the 64-kbyte blocks. The 84th to 92nd bytes are the information about the 128-kbyte blocks. See Table 18-8 for the values of bytes transmitted.

16. The 66th byte, transmitted from the target board to the controller, is a checksum value for the 5th to 92nd bytes. The checksum value is calculated by adding all these bytes together, dropping the carry and taking the two's complement of the total sum.

17. The 94th byte is the next command code.

4) Chip and Protection Bit Erase command (See Table 18-9)

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Product Information command is 40H.

3. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

   If the 3rd byte is equal to any of the command codes listed in Table 18-4, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 40H. If the 3rd byte is not a valid command, the boot program sends back x1H (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

4. The 5th byte, transmitted from the target board to the controller, is the Chip Erase Enable command code (54H).

5. The 6th byte, transmitted from the target board to the controller, is an acknowledge response to the 5th byte.
   Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.
   If the 5th byte is equal to any of the command codes to enable erasing, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 54H and then branches to the Chip Erase routine. If the 5th byte is not a valid command, the boot program sends back x1H (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

6. The 7th byte indicates whether the Chip Erase command is normally completed or not.
   At normal completion, completion code (4FH) is sent.
   When an error was detected, error code (4CH) is sent.

7. The 9th byte is the next command code.

5) Acknowledge Responses

The boot program represents processing states with specific codes. Table 18-10 to Table 18-13 show the values of possible acknowledge responses to the received data. The upper four bits of the acknowledge response are equal to those of the command being executed. Bit 3 of the code indicates a receive error. Bit 0 indicates an invalid command error, a checksum error or a password error. Bit 1 and bit 2 are always 0. Receive error checking is not done in I/O Interface mode.

Table 18-10 ACK Response to the Serial Operation Mode Byte

Return Value	Meaning
0x86	The SIO can be configured to operate in UART mode. (See Note)
0x30	The SIO can be configured to operate in I/O Interface mode.

(Note) If the serial operation mode is determined as UART, the boot program checks if the SIO can be programmed to the baud rate at which the operation mode byte was transferred. If that baud rate is not possible, the boot program aborts, without sending back any response.

Table 18-11 ACK Response to the Command Byte

Return Value	Meaning
0x?8 (See Note)	A receive error occurred while getting a command code.
0x?1 (See Note)	An undefined command code was received. (Reception was completed normally.)
0x10	The RAM Transfer command was received.
0x20	The Show Flash Memory Sum command was received.
0x30	The Show Product Information command was received.
0x40	The Chip Erase command was received.

(Note) The upper four bits of the ACK response are the same as those of the previous command code.

Table 18-12 ACK Response to the Checksum Byte

Return Value	Meaning
0xN8 (See Note)	A receive error occurred.
0xN1 (See Note)	A checksum or password error occurred.
0xN0 (See Note)	The checksum was correct.

(Note) The upper four bits of the ACK response are the same as those of the operation command code. It is 1 (N＝RAM transfer command data [7:4]) when password error occurs.

Table 18-13 ACK Response to Chip and Protection Bit Erase Byte

Return Value	Meaning
54H	The Chip Erase enabling command was received.
4FH	The Chip Erase command was completed.
4CH	The Chip Erase command was abnormally completed.

6) Determination of a Serial Operation Mode

The first byte from the controller determines the serial operation mode. To use UART mode for communications between the controller and the target board, the controller must first send a value of 86H at a desired baud rate to the target board. To use I/O Interface mode, the controller must send a value of 30H at 1/16 the desired baud rate. Fig. 18-4 shows the waveforms for the first byte.

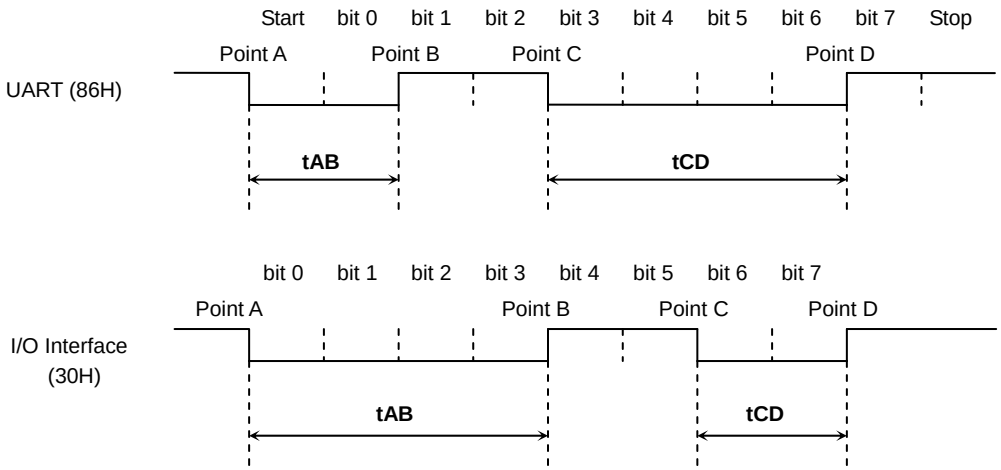


Fig. 18-4 Serial Operation Mode Byte

After $\overline{\text{RESET}}$ is released, the boot program monitors the first serial byte from the controller, with the SIO reception disabled, and calculates the intervals of tAB, tAC and tAD. Fig. 18-5 shows a flowchart describing the steps to determine the intervals of tAB, tAC and tAD. As shown in the flowchart, the boot program captures timer counts each time a logic transition occurs in the first serial byte. Consequently, the calculated tAB, tAC and tAD intervals are bound to have slight errors. If the transfer goes at a high baud rate, the CPU might not be able to keep up with the speed of logic transitions at the serial receive pin. In particular, I/O Interface mode is more prone to this problem since its baud rate is generally much higher than that for UART mode. To avoid such a situation, the controller should send the first serial byte at 1/16 the desired baud rate.

The flowchart in Fig. 18-5 shows how the boot program distinguishes between UART and I/O Interface modes. If the length of tAB is equal to or less than the length of tCD, the serial operation mode is determined as UART mode. If the length of tAB is greater than the length of tCD, the serial operation mode is determined as I/O Interface mode. Bear in mind that if the baud rate is too high or the timer operating frequency is too low, the timer resolution will be coarse, relative to the intervals between logic transitions. This becomes a problem due to inherent errors caused by the way in which timer counts are captured by software; consequently the boot program might not be able to determine the serial operation mode correctly. To prevent this problem, reset UART mode within the programming routine.

For example, the serial operation mode may be determined to be I/O Interface mode when the intended mode is UART mode. To avoid such a situation, when UART mode is utilized, the controller should allow for a time-out period within which it expects to receive an echo-back (86H) from the target board. The controller should give up the communication if it fails to get that echo-back within the allowed time. When I/O Interface mode is utilized, once the first serial byte has been transmitted, the controller should send the SCLK clock after a certain idle time to get an acknowledge response. If the received acknowledge response is not 30H, the controller should give up further communications.

When the intended mode is I/O interface mode, the first byte does not have to be 0x30 as long as tAB is greater than tCD as shown above. 0x91, 0xA1 or 0xB1 can be sent as the first byte code to determine the falling edges of Point A and Point C and the rising edges of Point B and Point D. If tAB is greater than tCD and SIO is selected by the resolution of the operation mode determination, the second byte code is 0x30 even though the transmitted code on the first byte is not 0x30 (The first byte code to determine I/O interface mode is described as 0x30).

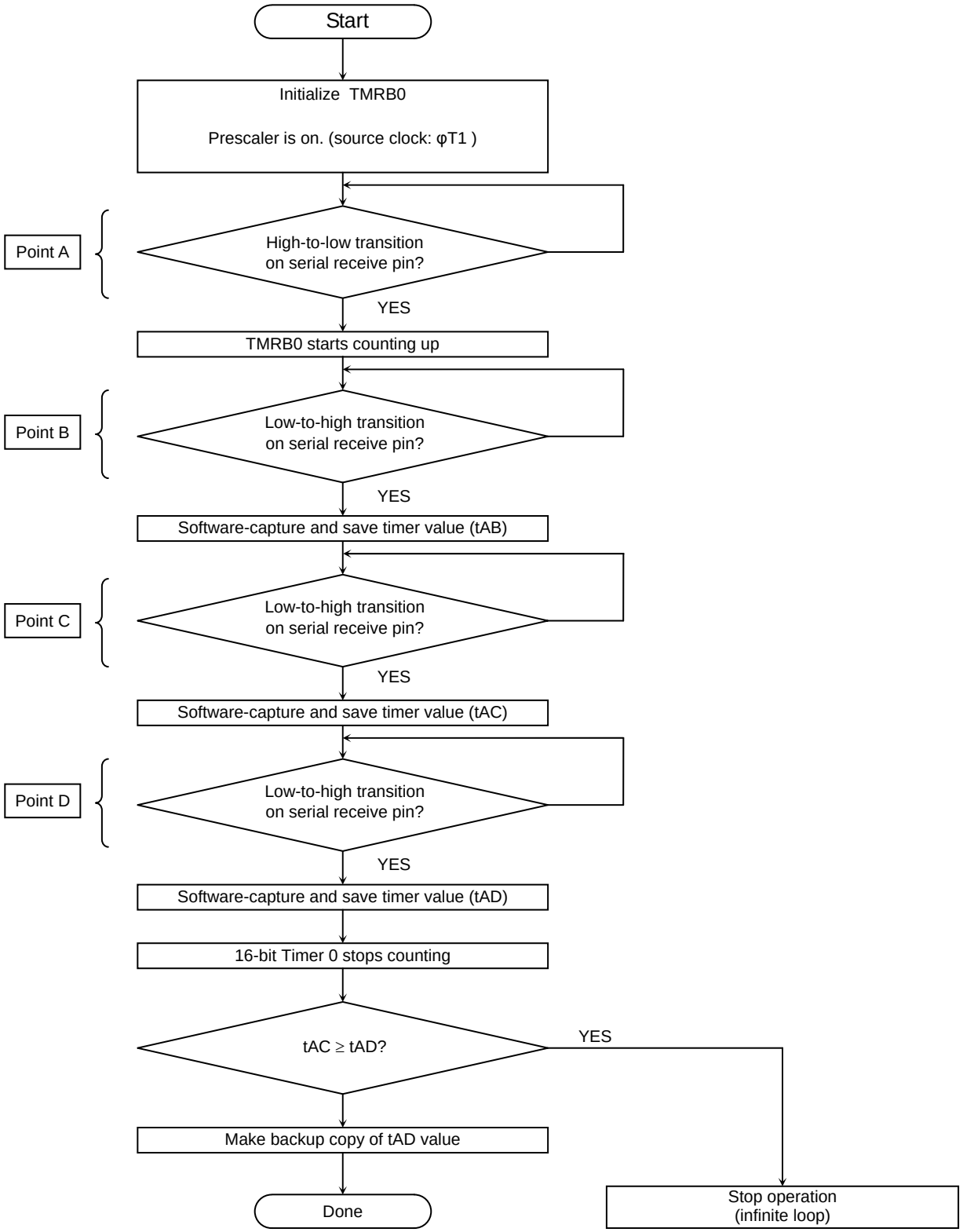


Fig. 18-5 Serial Operation Mode Byte Reception Flow

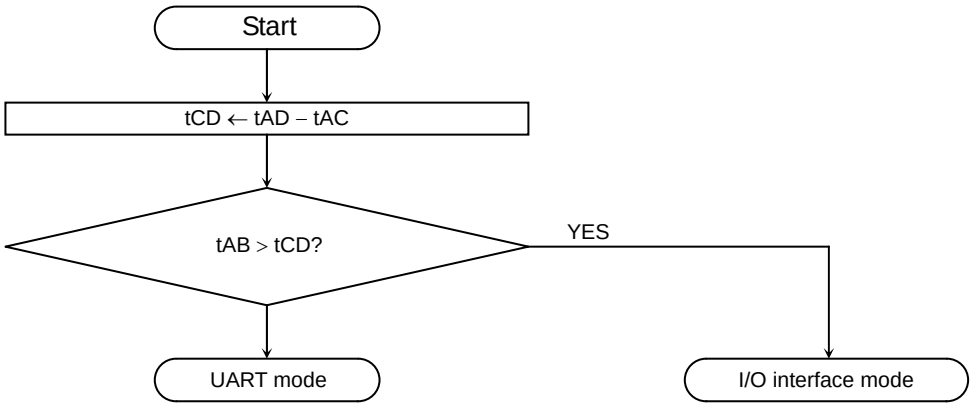


Fig. 18-6 Serial Operation Mode Determination Flow

7) Password

The RAM Transfer command (10H) causes the boot program to perform password verification. Following an echo-back of the command code, the boot program verifies the contents of the 12-byte password area within the flash memory. The following table shows the password area of each product.

Product name	Area
TMPM330FDFG TMPM330FYFG	0x3F87_FF04 – 0x3F87_FF0F
TMPM330FWFG	0x3F81_FF04 – 0x3F81_FF0F

If all these address locations contain the same bytes of data other than FFH, a password area error occurs as shown in Fig. 18-7. In this case, the boot program returns an error acknowledge (11H) in response to the checksum byte (the 17th byte), regardless of whether the password sequence sent from the controller is all FFHs.

The password sequence received from the controller (5th to 16th bytes) is compared to the password stored in the flash memory. All of the 12 bytes must match to pass the password verification. Otherwise, a password error occurs, which causes the boot program to reply an error acknowledge in response to the checksum byte (the 17th byte).

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

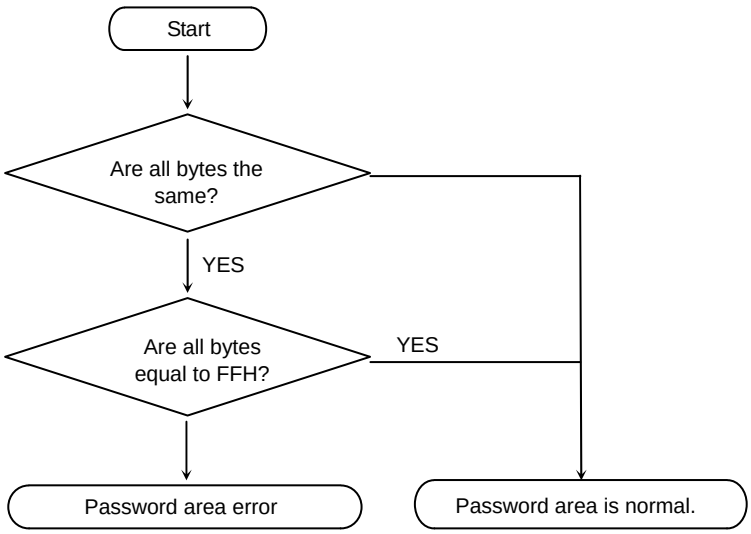


Fig. 18-7 Password Area Verification Flow

8) Calculation of the Show Flash Memory Sum Command

The result of the sum calculation “byte + byte + byte + …“ is responded by a word quantity. The Show Flash Memory Sum command adds all 512 Kbytes of the flash memory together and provides the total sum as a word quantity. The sum is sent to the controller, with the upper eight bits first, followed by the lower eight bits.

Example)

A1H
B2H
C3H
D4H

For the interest of simplicity, assume the depth of the flash memory is four locations. Then the sum of the four bytes is calculated as:

A1H + B2H + C3H + D4H = 02EAH

Hence, 02H is first sent to the controller, followed by EAH.

9) Checksum Calculation

The checksum byte for a series of bytes of data is calculated by adding the bytes together, dropping the carries, and taking the two’s complement of the total sum. The Show Flash Memory Sum command and the Show Product Information command perform the checksum calculation. The controller must perform the same checksum operation in transmitting checksum bytes.

Example) Assume the Show Flash Memory Sum command provides the upper and lower bytes of the sum as E5H and F6H. To calculate the checksum for a series of E5H and F6H:

Add the bytes together

E5H + F6H = 1DBH

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

0 − DBH = 25H

(7) General Boot Program Flowchart

Fig. 18-8 shows an overall flowchart of the boot program.

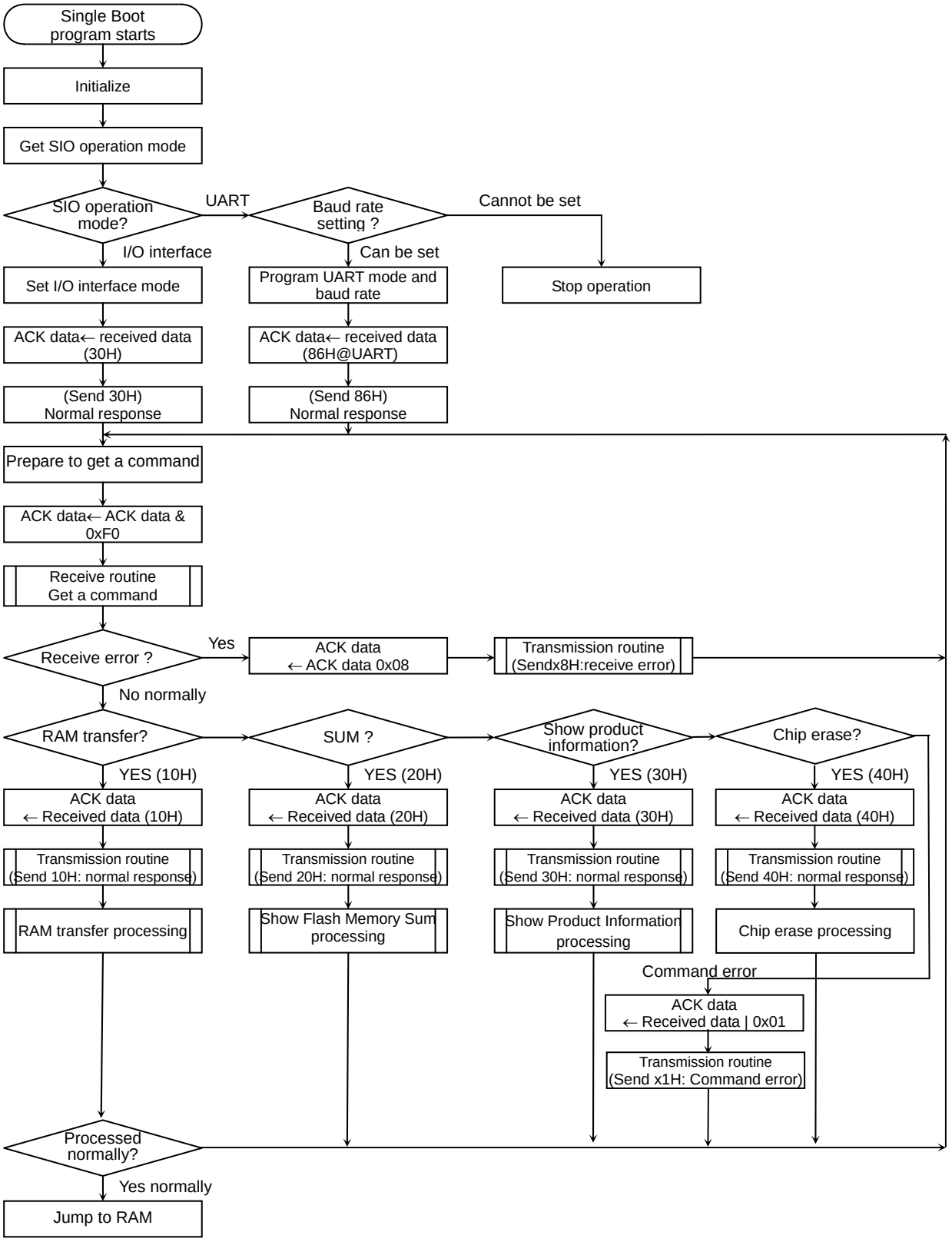


Fig. 18-8 Overall Boot Program Flow

18.3 On-board Programming of Flash Memory (Rewrite/Erase)

In on-board programming, the CPU is to execute software commands for rewriting or erasing the flash memory. The rewrite/erase control program should be prepared by the user beforehand. Because the flash memory content cannot be read while it is being written or erased, it is necessary to run the rewrite/erase program from the internal RAM or from an external memory device after shifting to the user boot mode.

18.3.1 Flash Memory

Except for some functions, writing and erasing flash memory data are in accordance with the standard JEDEC commands. In writing or erasing, use 32-bit data transfer command of the CPU to enter commands to the flash memory. Once the command is entered, the actual write or erase operation is automatically performed internally.

Table 18-14 Flash Memory Functions

Major functions	Description
Automatic page program	Writes data automatically per page.
Automatic chip erase	Erases the entire area of the flash memory automatically.
Automatic block erase	Erases a selected block automatically.
Protect function	By writing a 4-bit protection code, the write or erase function can be individually inhibited for each block.

Note that addressing of operation commands is different from the case of standard commands due to the specific interface arrangements with the CPU. Also note that the flash memory is written in 32-bit blocks. So, 32-bit (word) data transfer commands must be used in writing the flash memory.

(1) Block configuration

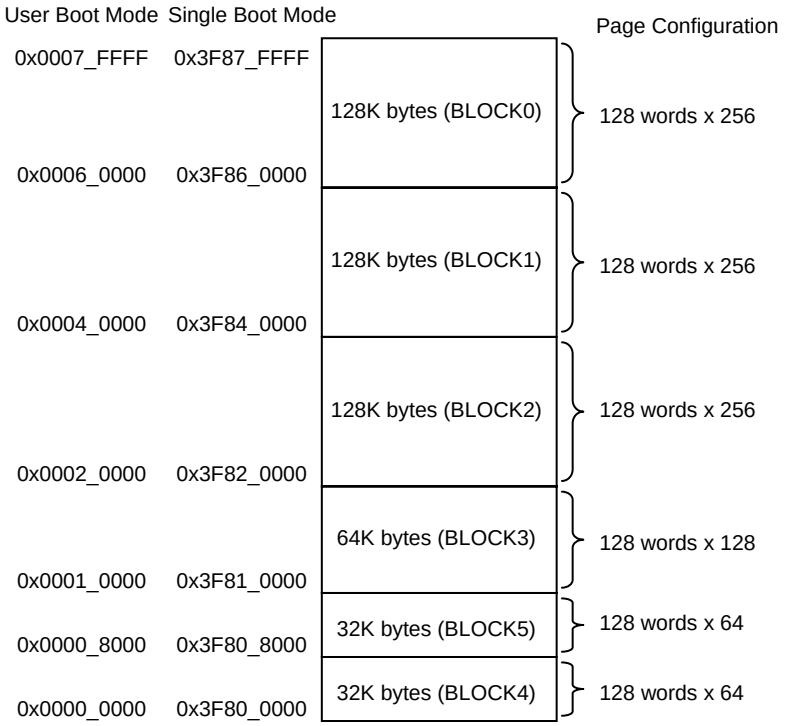


Fig. 18-9 Block Configuration of Flash Memory (TMPM330FDFG)

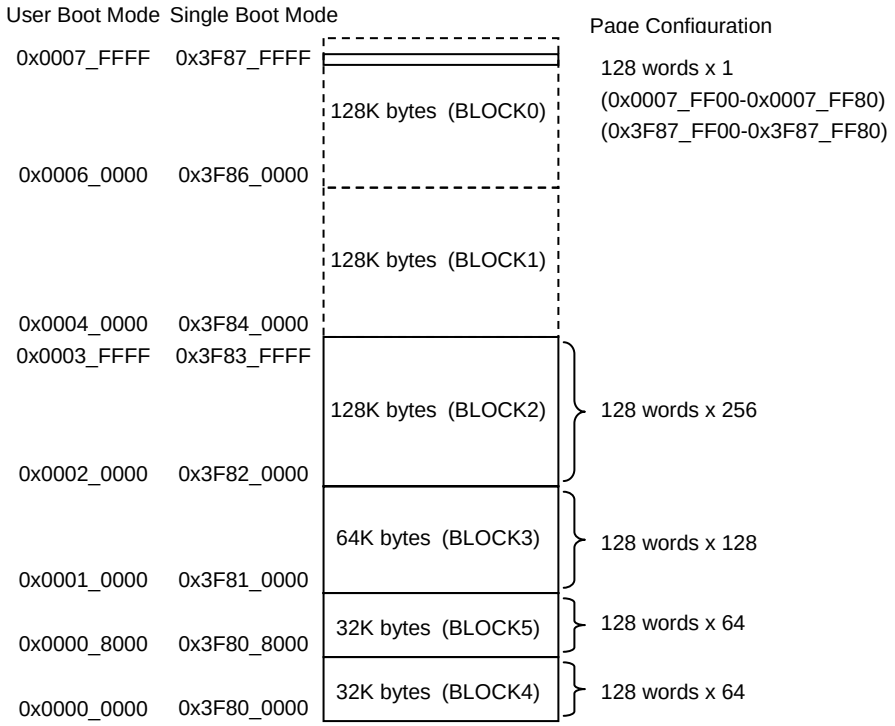


Fig. 18-10 Block Configuration of Flash Memory (TMPM330FYFG)

(Note)	**In addition to 256KB flash area, the TMPM330FYFG provides 128-word data/ password area (0x3F87_FF00 – 0x3F87_FF80, 1 page) for Show Product Information command. To erase the content, execute the automatic chip erase command or assign block 0 with the automatic block erase command.**

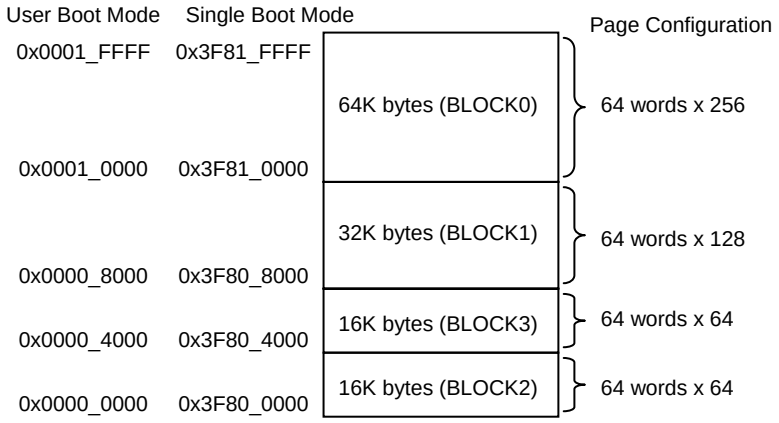


Fig. 18-11 Block Configuration of Flash Memory (TMPM330FWFG)

(2) Basic operation

Generally speaking, this flash memory device has the following two operation modes:

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

Transition to the automatic mode is made by executing a command sequence while it is in the memory read mode. In the automatic operation mode, flash memory data cannot be read and any commands stored in the flash memory cannot be executed. In the automatic operation mode, any interrupt or exception generation cannot set the device to the read mode except when a hardware reset is generated. During automatic operation, be sure not to cause any exceptions other than debug exceptions and reset while a debug port is connected. Any exception generation cannot set the device to the read mode except when a hardware reset is generated.

1) Read

When data is to be read, the flash memory must be set to the read mode. The flash memory will be set to the read mode immediately after power is applied, when CPU reset is removed, or when an automatic operation is normally terminated. In order to return to the read mode from other modes or after an automatic operation has been abnormally terminated, either the Read/reset command (a software command to be described later) or a hardware reset is used. The device must also be in the read mode when any command written on the flash memory is to be executed.

- **Read/reset command and Read command (software reset)**

When ID-Read command is used, the reading operation is terminated instead of automatically returning to the read mode. In this case, the Read/reset command can be used to return the flash memory to the read mode. Also, when a command that has not been completely written has to be canceled, the Read/reset command must be used. The Read command is used to return to the read mode after executing 32-bit data transfer command to write the data "0x0000_00F0" to an arbitrary address of the flash memory.

- **With the Read/reset command, the device is returned to the read mode after completing the third bus write cycle.**

2) Command write

This flash memory uses the command control method. Commands are executed by executing a command sequence to the flash memory. The flash memory executes automatic operation commands according to the address and data combinations applied (refer to Command Sequence).

If it is desired to cancel a command write operation already in progress or when any incorrect command sequence has been entered, the Read/reset command is to be executed. Then, the flash memory will terminate the command execution and return to the read

While commands are generally comprised of several bus cycles, the operation to apply 32-bit data transmit command to the flash memory is called "bus write cycle." The bus write cycles are to be in a specific sequential order and the flash memory will perform an automatic operation when the sequence of the bus write cycle data and address of a

command write operation is in accordance with a predefined specific sequence. If any bus write cycle does not follow a predefined command write sequence, the flash memory will terminate the command execution and return to the read mode.

(Note 1) Command sequences are executed from outside the flash memory area.

(Note 2) Each bus write cycle must be sequentially executed by 32-bit data transmit command. While a command sequence is being executed, access to the flash memory is prohibited. Also, don't generate any interrupt (except debug exceptions when a DSU probe is connected).If such an operation is made, it can result in an unexpected read access to the flash memory and the command sequencer may not be able to correctly recognize the command. While it could cause an abnormal termination of the command sequence, it is also possible that the written command is incorrectly recognized.

(Note 3) For the command sequencer to recognize a command, the device must be in the read mode prior to executing the command. Be sure to check before the first bus write cycle that the FLCS RDY/BSY bit is set to "1." It is recommended to subsequently execute a Read command.

(Note 4) Upon issuing a command, if any address or data is incorrectly written, be sure to perform a software reset to return to the read mode again.

(3) Reset

Hardware reset

A hardware reset is used to cancel the operational mode set by the command write operation when forcibly termination during auto programming/ erasing or abnormal termination during auto operations occurs. The flash memory has a reset input as the memory block and it is connected to the CPU reset signal. Therefore, when the RESET input pin of this device is set to V_{IL} or when the CPU is reset due to any overflow of the watch dog timer, the flash memory will return to the read mode terminating any automatic operation that may be in progress. It should also be noted that applying a hardware reset during an automatic operation can result in incorrect rewriting of data. In such a case, be sure to perform the rewriting again.

Refer to Section 18.2.1 "Reset Operation" for CPU reset operations. After a given reset input, the CPU will read the reset vector data from the flash memory and starts operation after the reset is removed.

(4) Commands

1) Automatic Page Programming

Writing to a flash memory device is to make "1" data cells to "0" data cells. Any "0" data cell cannot be changed to a "1" data cell. For making "0" data cells to "1" data cells, it is necessary to perform an erase operation.

The automatic page programming function of this device writes data of each page. The TMPM330FDFG/ TMPM330FYFG contain 128 words and the TMPM330FWFG contains 64 words in a page. A 128 word block is defined by a same [31:9] address and it starts from the address [8:0] = 0 and ends at the address [8:0] = 0x1FF. A 64 word block is defined by a same [31:8] address and it starts from the address [7:0] = 0 and ends at the address [7:0] = 0xFF. This programming unit is hereafter referred to as a "page."

Writing to data cells is automatically performed by an internal sequencer and no external control by the CPU is required. The state of automatic page programming (whether it is in writing operation or not) can be checked by the FLCS [0] <RDY/BSY> register.

Also, any new command sequence is not accepted while it is in the automatic page programming mode. If it is desired to interrupt the automatic page programming, use the hardware reset function. If the operation is stopped by a hardware reset operation, it is necessary to once erase the page and then perform the automatic page programming again because writing to the page has not been normally terminated.

The automatic page programming operation is allowed only once for a page already erased. No programming can be performed twice or more times irrespective of the data cell value whether it is "1" or "0." Note that rewriting to a page that has been once written requires execution of the automatic block erase or automatic chip erase command before executing the automatic page programming command again. Note that an attempt to rewrite a page two or more times without erasing the content can cause damages to the device.

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

The automatic page programming operation starts when the third bus write cycle of the command cycle is completed. On and after the fifth bus write cycle, data will be written sequentially starting from the next address of the address specified in the fourth bus write cycle (in the fourth bus write cycle, the page top address will be command written) (32 bits of data is input at a time). Be sure to use the 32-bit data transfer command in writing commands on and after the fourth bus cycle. In this, any 32-bit data transfer commands shall not be placed across word boundary. On and after the fifth bus write cycle, data is command written to the same page area. Even if it is desired to write the page only partially, it is required to perform the automatic page programming for the entire page. In this case, the address input for the fourth bus write cycle shall be set to the top address of the page. Be sure to perform command write operation with the input data set to "1" for the data cells not to be set to "0." For example, if the top address of a page is not to be written, set the input data of the fourth bus write cycle to 0xFFFFFFFF to command write the data.

Once the fourth bus cycle is executed, it is in the automatic programming operation. This condition can be checked by monitoring the register bit FLCS [0] <RDY/BSY> (See Table 18-15). Any new command sequence is not accepted while it is in automatic page programming mode. If it is desired to stop operation, use the hardware reset function. Be

careful in doing so because data cannot be written normally if the operation is interrupted. When a single page has been command written normally terminating the automatic page writing process, the FLCS [0] <RDY/BSY> bit is set to "1" and it returns to the read mode. When multiple pages are to be written, it is necessary to execute the page programming command for each page because the number of pages to be written by a single execution of the automatic page program command is limited to only one page. It is not allowed for automatic page programming to process input data across pages.

Data cannot be written to a protected block. When automatic programming is finished, it automatically returns to the read mode. This condition can be checked by monitoring FLCS [0] <RDY/BSY> (Table 18-15). If automatic programming has failed, the flash memory is locked in the mode and will not return to the read mode. For returning to the read mode, it is necessary to execute hardware reset to reset the flash memory or the device. In this case, while writing to the address has failed, it is recommended not to use the device or not to use the block that includes the failed address.

(Note) Software reset becomes ineffective in bus write cycles on and after the fourth bus write cycle of the automatic page programming command.

2) Automatic chip erase

The automatic chip erase operation starts when the sixth bus write cycle of the command cycle is completed.

This condition can be checked by monitoring FLCS [0] <RDY/BSY> (See Table 18-15). While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that data has been correctly erased. Any new command sequence is not accepted while it is in an automatic chip erase operation. If it is desired to stop operation, use the hardware reset function. If the operation is forced to stop, it is necessary to perform the automatic chip erase operation again because the data erasing operation has not been normally terminated.

Also, any protected blocks cannot be erased. If all the blocks are protected, the automatic chip erase operation will not be performed and it returns to the read mode after completing the sixth bus read cycle of the command sequence. When an automatic chip erase operation is normally terminated, it automatically returns to the read mode. If an automatic chip erase operation has failed, the flash memory is locked in the mode and will not return to the read mode.

For returning to the read mode, it is necessary to execute hardware reset to reset the device. In this case, the failed block cannot be detected. It is recommended not to use the device anymore or to identify the failed block by using the block erase function for not to use the identified block anymore.

3) Automatic block erase (fro aech block)

The automatic block erase operation starts when the sixth bus write cycle of the command cycle is completed.

This status of the automatic block erase operation can be checked by monitoring FLCS <RDY/BSY> (See Table 18-15). While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that data has been correctly erased. Any new command sequence is not accepted while it is in an automatic block erase operation. If it is desired to stop operation, use the hardware reset function. In this case, it is necessary to perform the automatic block erase operation again because the data erasing operation has not been normally terminated.

Also, any protected blocks cannot be erased. If an automatic block erase operation has failed, the flash memory is locked in the mode and will not return to the read mode. In this case, execute hardware reset to reset the device.

4) Automatic programming of protection bits (for each block)
This device is implemented with protection bits. This protection can be set for each block. See Table 18-20 for table of protection bit addresses. This device assigns 1 bit to 1 block as a protection bit. The applicable protection bit is specified by PBA in the seventh bus write cycle. By automatically programming the protection bits, write and/or erase functions can be inhibited (for protection) individually for each block. The protection status of each block can be checked by the FLCS <BLPRO> register to be described later. This status of the automatic programming operation to set protection bits can be checked by monitoring FLCS <RDY/BSY> (See Table 18-15). Any new command sequence is not accepted while automatic programming is in progress to program the protection bits. If it is desired to stop the programming operation, use the hardware reset function. In this case, it is necessary to perform the programming operation again because the protection bits may not have been correctly programmed. If all the protection bits have been programmed, all the FLCS <BLPRO> bits are set to "1" indicating that it is in the protected state (See Table 18-15). This disables subsequent writing and erasing of all blocks.

(Note) Software reset is ineffective in the seventh bus write cycle of the automatic protection bit programming command. The FLCS <RDY/BSY> bit turns to "0" after entering the seventh bus write cycle.

5) Automatic erasing of protection bits

Different results will be obtained when the automatic protection bit erase command is executed depending on the status of the protection bits and the security bits. It depends on the status of FLCS <BLPRO> whether all the <BLPRO> bits are set to "1" or not if SECBIT<SECBIT> is 0x1. Be sure to check the value of FLCS <BLPRO> before executing the automatic protection bit erase command. See chapter 17 for details.

(Note) The TMPM330FYFG is configured with block 2 through 5. Block 0 and 1 require a programming of protection bits when using security function.

・When all the FLCS <BLPRO> bits are set to "1" (all the protection bits are programmed):

When the automatic protection bit erase command is command written, the flash memory is automatically initialized within the device. When the seventh bus write cycle is completed, the entire area of the flash memory data cells is erased and then the protection bits are erased. This operation can be checked by monitoring FLCS <RDY/BSY>. If the automatic operation to erase protection bits is normally terminated, FLCS will be set to "0x00000001." While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that it has been correctly erased. For returning to the read mode while the automatic operation after the seventh bus cycle is in progress, it is necessary to use the hardware reset to reset the device. If this is done, it is necessary to check the status of protection bits by FLCS <BLPRO> after retuning to the read mode and perform either the automatic protection bit erase, automatic chip erase, or automatic block erase operation, as appropriate.

・When the FLCS <BLPRO> bits include "0" (not all the protection bits are programmed):

The protection condition can be canceled by the automatic protection bit erase operation. With this device, protection bits set by an individual block can be erased handling all the blocks at a time as shown in Table 18-21. The target bits are specified in the seventh bus write cycle and when the command is completed, the device is in a condition all the blocks are erased. The protection status of each block can be checked by FLCS <BLPRO> to be described later. This status of the programming operation for automatic protection bits can be checked by monitoring FLCS <RDY/BSY>. When the automatic operation to erase protection bits is normally terminated, the protection bits of FLCS <BLPRO> selected for erasure are set to "0."

In any case, any new command sequence is not accepted while it is in an automatic operation to erase protection bits. If it is desired to stop the operation, use the hardware reset function. When the automatic operation to erase protection bits is normally terminated, it returns to the read mode.

(Note) The FLCS <RDY/BSY> bit is "0" while in automatic operation and it turns to "1" when the automatic operation is terminated.

6) ID-Read

Using the ID-Read command, you can obtain the type and other information on the flash memory contained in the device. The data to be loaded will be different depending on the address [15:14] of the fourth and subsequent bus write cycles (recommended input data is 0x00). On and after the fourth bus write cycle, when an arbitrary flash memory area is read, the ID value will be loaded. Once the fourth bus write cycle of an ID-Read command has passed, the device will not automatically return to the read mode. In this condition, the set of the fourth bus write cycle and ID-Read commands can be repetitively executed. For returning to the read mode, use the Read/reset command or hardware reset command.

(5) Flash control/ status register

This resister is used to monitor the status of the flash memory and to indicate the protection status of each block.

Table 18-15 Flash Control Register

FLCS
0x4004_0520

	31	30	29	28	27	26	25	24
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read.							

	23	22	21	20	19	18	17	16
bit Symbol	-	-	BLPRO5	BLPRO4	BLPRO3	BLPRO2	BLPRO1	BLPRO0
Read/Write	R		R	R	R	R	R	R
After reset	0		(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)
Function	"0" is read.		Protection for Block 5 (Note 2) 0: disabled 1:enabled	Protection for Block 4 (Note 2) 0: disabled 1:enabled	Protection for Block 3 0: disabled 1:enabled	Protection for Block 2 0: disabled 1:enabled	Protection for Block 1 0: disabled 1:enabled	Protection for Block 0 0: disabled 1:enabled

	15	14	13	12	11	10	9	8
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read.							

	7	6	5	4	3	2	1	0
bit Symbol	-	-	-	-	-	-	-	RDY/BSY
Read/Write	R							R
After reset	0							1
Function	"0" is read.							Ready/Busy (Note 1) 0:Auto operating 1:Auto operation terminated

Bit 0: Ready/Busy flag bit

The RDY/BSY output is provided as a means to monitor the status of automatic operation. This bit is a function bit for the CPU to monitor the function. When the flash memory is in automatic operation, it outputs "0" to indicate that it is busy. When the automatic operation is terminated, it returns to the ready state and outputs "1" to accept the next command. If the automatic operation has failed, this bit maintains the "0" output. By applying a hardware reset, it returns to "1."

Bit [21:16]: Protection status bits

Each of the protection bits represents the protection status of the corresponding block. When a bit is set to "1," it indicates that the block corresponding to the bit is protected. When the block is protected, data cannot be written to it.

(Note 1) This command must be issued in the ready state. Issuing the command in the busy state may disable both correct command transmission and further command input. To exit from the condition, execute system reset. System reset requires at least 0.5 microseconds regardless of the system clock frequency. In this condition, it takes approx. 2 ms to enable reading after reset.

(Note 2) The FLCS bits [21:20] of TMPM330FWFG have no function. They are read as "0".

(Note 3) The value varies depending on protection applied.

Table 18-16 Security bit register

SECBIT
0x4004_0500

	31	30	29	28	27	26	25	24
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read							

	23	22	21	20	19	18	17	16
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read							

	15	14	13	12	11	10	9	8
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read							

	7	6	5	4	3	2	1	0
bit Symbol	-	-	-	-	-	-	-	SECBIT
Read/Write	R							R/W
After reset	0							1
Function	"0" is read							Security bits 0:disabled 1:enabled

(Note) This register is initialized only by power-on reset.

(6) List of Command Sequences

Table 18-17 Flash Memory Access from the Internal CPU

Command sequence	First bus cycle	Second bus cycle	Third bus cycle	Fourth bus cycle	Fifth bus cycle	Sixth bus cycle	Seventh bus cycle
	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.
	Data	Data	Data	Data	Data	Data	Data
Read	0xXX	–	–	–	–	–	–
	0xF0	–	–	–	–	–	–
Read/Reset	0x54XX	0xAAXX	0x54XX	RA	–	–	–
	0xAA	0x55	0xF0	RD	–	–	–
ID-Read	0x54XX	0xAAXX	0x54XX	IA	0xXX	–	–
	0xAA	0x55	0x90	0x00	ID	–	–
Automatic page programming	0x54XX	0xAAXX	0x54XX	PA	PA	PA	PA
	0xAA	0x55	0xA0	PD0	PD1	PD2	PD3
Automatic chip erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	–
	0xAA	0x55	0x80	0xAA	0x55	0x10	–
Auto Block erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	BA	–
	0xAA	0x55	0x80	0xAA	0x55	0x30	–
Protection bit programming	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x9A	0xAA	0x55	0x9A	0x9A
Protection bit erase	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x6A	0xAA	0x55	0x6A	0x6A

Supplementary explanation

- RA: Read address
- RD: Read data
- IA: ID address
- ID: ID data
- PA: Program page address
  PD: Program data (32 bit data)
  After the fourth bus cycle, enter data in the order of the address for a page.
- BA: Block address
- PBA: Protection bit address

(Note 1) Always set "0" to the address bits [1:0] in the entire bus cycle. (Recommendable setting values to bits [7:2] are ”0”.)
(Note 2) Bus cycles are "bus write cycles" except for the second bus cycle of the Read command, the fourth bus cycle of the Read/reset command, and the fifth bus cycle of the ID-Read command. Bus write cycles are executed by 32-bit data transfer commands. The address [31:16] in each bus write cycle should be the target flash memory address [31:16] of the command sequence. Use "Addr." in the table for the address [15:0].

(7) Address bit configuration for bus write cycles

Table 18-18 Address Bit Configuration for Bus Write Cycles

Address	Addr [31:19]	Addr [18]	Addr [17]	Addr [16]	Addr [15]	Addr [14]	Addr [13:11]	Addr [10]	Addr [9]	Addr [8]	Addr [7:0]

[TMPM330FDFG/ FYFG/ FWFG]

Normal commands	**Normal bus write cycle address configuration**				
	Flash area	“0” is recommended.	Command		Addr[1:0]=“0” (fixed) Others:0 (recommended)
ID -READ	**IA: ID address (Set the fourth bus write cycle address for ID-Read operation)**				
	Flash area	“0” is recommended.	ID address	Addr[1:0]=“0” (fixed) , Others:0 (recommended)	

[TMPM330FDFG/ TMPM330FYFG]

Block erase	**BA: Block address (Set the sixth bus write cycle address for block erase operation)**				
	Block selection (**Table 18-19**)		Addr[1:0]=“0” (fixed) , Others:0 (recommended)		
Auto page programming	**PA: Program page address (Set the fourth bus write cycle address for page programming operation)**				
	Page selection				Addr[1:0]=“0” (fixed) Others:0 (recommended)
Protection bit programming	**PBA: Protection bit address (Set the seventh bus erase cycle address for protection bit erasure)**				
	Flash area	Protection bit selection (**Table 18-20**)	Fixed to “0”.	Protection bit selection (**Table 18-20**)	Addr[1:0]=“0” (fixed) Others:0 (recommended)
Protection bit erase	**PBA: Protection bit address (Set the seventh bus erase cycle address for protection bit erasure)**				
	Flash area	Protection bit selection (**Table 18-21**)	“0” is recommended.	Protection bit selection (**Table 18-21**)	Addr[1:0]=“0” (fixed) Others:0 (recommended)

[TMPM330FWFG only]

Block erase	**BA: Block address (Set the sixth bus write cycle address for block erase operation)**				
	Block selection (**Table 18-19**)		Addr[1:0]=“0” (fixed) , Others:0 (recommended)		
Auto page programming	**PA: Program page address (Set the fourth bus write cycle address for page programming operation)**				
	Page selection				Addr[1:0]=“0” (fixed) Others:0 (recommended)
Protection bit programming	**PBA: Protection bit address (Set the seventh bus write cycle address for protection bit programming**				
	Flash area	Protection bit selection (**Table 18-20**)	Fixed to “0”.	Protection bit selection (**Table 18-20**)	Addr[1:0]=“0” (fixed) Others:0 (recommended)
Protection bit erase	**PBA: Protection bit address (Set the seventh bus erase cycle address for protection bit erasure)**				
	Flash area	Protection bit selection (**Table 18-21**)	Fixed to “0”.	Protection bit selection (**Table 18-21**)	Addr[1:0]=“0” (fixed) Others:0 (recommended)

(Note 1) Table 18-17 "Flash Memory Access from the Internal CPU" can also be used.
(Note 2) Address setting can be performed according to the "Normal bus write cycle address configuration" from the first bus cycle.
(Note 3) "0" is recommended" can be changed as necessary.

Table 18-19 Block Address Table

Block	Address (User boot mode)	Address (Single boot mode)	Size (Kbyte)
[TMPM330FDFG/ TMPM330FYFG]			
4	0x0000_0000-0x0000_7FFF	0x3F80_0000-0x3F80_7FFF	32
5	0x0000_8000-0x0000_FFFF	0x3F80_0000-0x3F80_FFFF	32
3	0x0001_0000-0x0001_FFFF	0x3F81_0000-0x3F81_FFFF	64
2	0x0002_0000-0x0003_FFFF	0x3F82_0000-0x3F83_FFFF	128
1	0x0004_0000-0x0005_FFFF	0x3F84_0000-0x3F85_FFFF	128
0	0x0006_0000-0x0007_FFFF	0x3F86_8000-0x3F87_FFFF	128
[TMPM330FWFG]			
2	0x0000_0000-0x0000_3FFF	0x3F80_0000-0x3F80_3FFF	16
3	0x0000_4000-0x0000_7FFF	0x3F80_4000-0x3F80_7FFF	16
1	0x0000_8000-0x0000_FFFF	0x3F80_8000-0x3F80_FFFF	32
0	0x0001_0000-0x0001_FFFF	0x3F81_0000-0x3F31_FFFF	64

As block address, specify any address in the block to be erased.

(Note) As for the addresses from the first to the fifth bus cycles, specify the upper 4 bit with the corresponding flash memory addresses of the blocks to be erased.

Table 18-20 Protection Bit Programming Address Table

Block	Protection bit	The seventh bus write cycle address						
		Address [18]	Address [17]	Address [16]	Address [15:11]	Address [10]	Address [9]	Address [8]
[TMPM330FDFG/ TMPM330FYFG]								
Block0	BLPRO0	0	0	1	Fixed to "0".	0	0	"0" is recommended.
Block1	BLPRO1	0	0	1		0	1	
Block2	BLPRO2	0	0	1		1	0	
Block3	BLPRO3	0	0	1		1	1	
Block4	BLPRO4	0	1	0		0	0	
Block5	BLPRO5	0	1	0		0	1	
[TMPM330FWFG]								
Block0	BLPRO0	0	0	Fixed to "0".			0	0
Block1	BLPRO1	0	0				0	1
Block2	BLPRO2	0	0				1	0
Block3	BLPRO3	0	0				1	1

Table 18-21 Protection Bit Erase Address Table

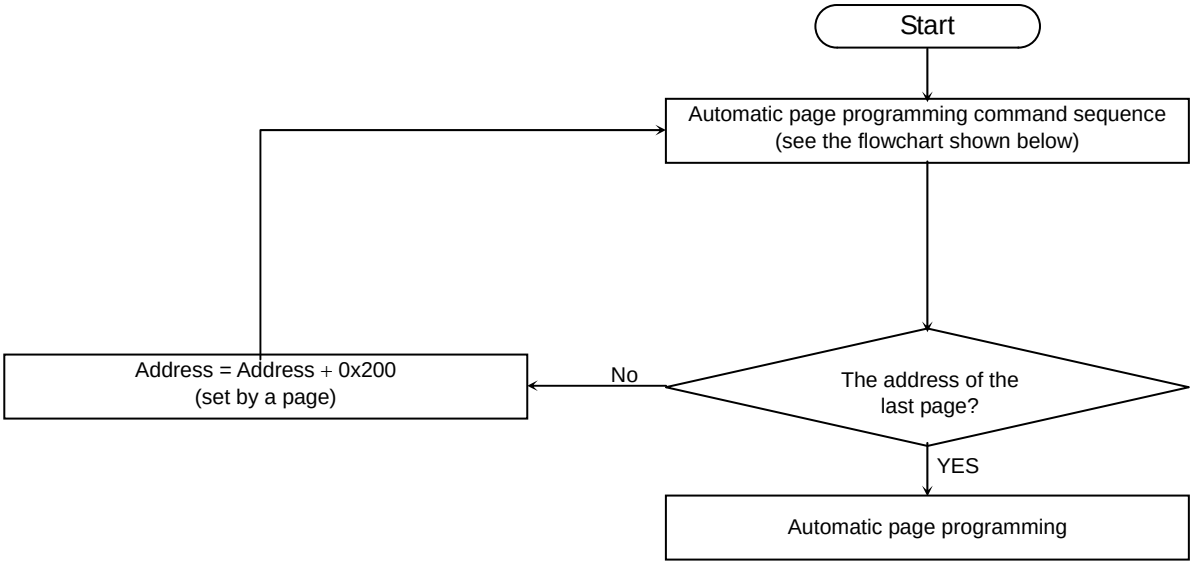
Block	Protection bit	The seventh bus write cycle address [18:17]	
		Address [18]	Address [17]
Block0～3	BLPRO0～3	0	0
Block4～5	BLPRO4～5	0	1

(Note) The protection bit erase command cannot erase by individual block.

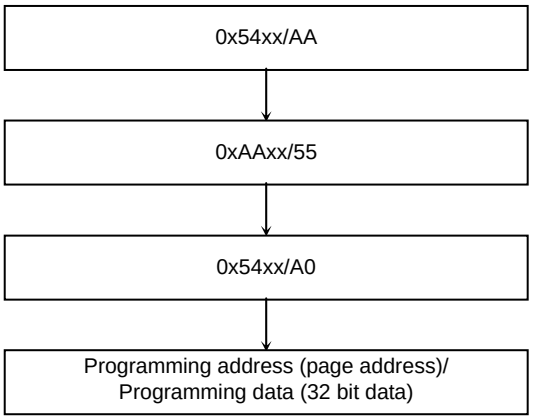
Table 18-22 The ID-Read command's fourth bus write cycle ID address (IA) and the data to be read by the following 32-bit data transfer command (ID)

IA [15:14]	ID [7: 0]	Code
00b	0x98	Manufacturer code
01b	0x5A	Device code
10b	Reserved	---
11b	0x12 (TMPM330FDFG) 0x12 (TMPM330FYFG) 0x11 (TMPM330FWFG)	Macro code

(8) Flowchart

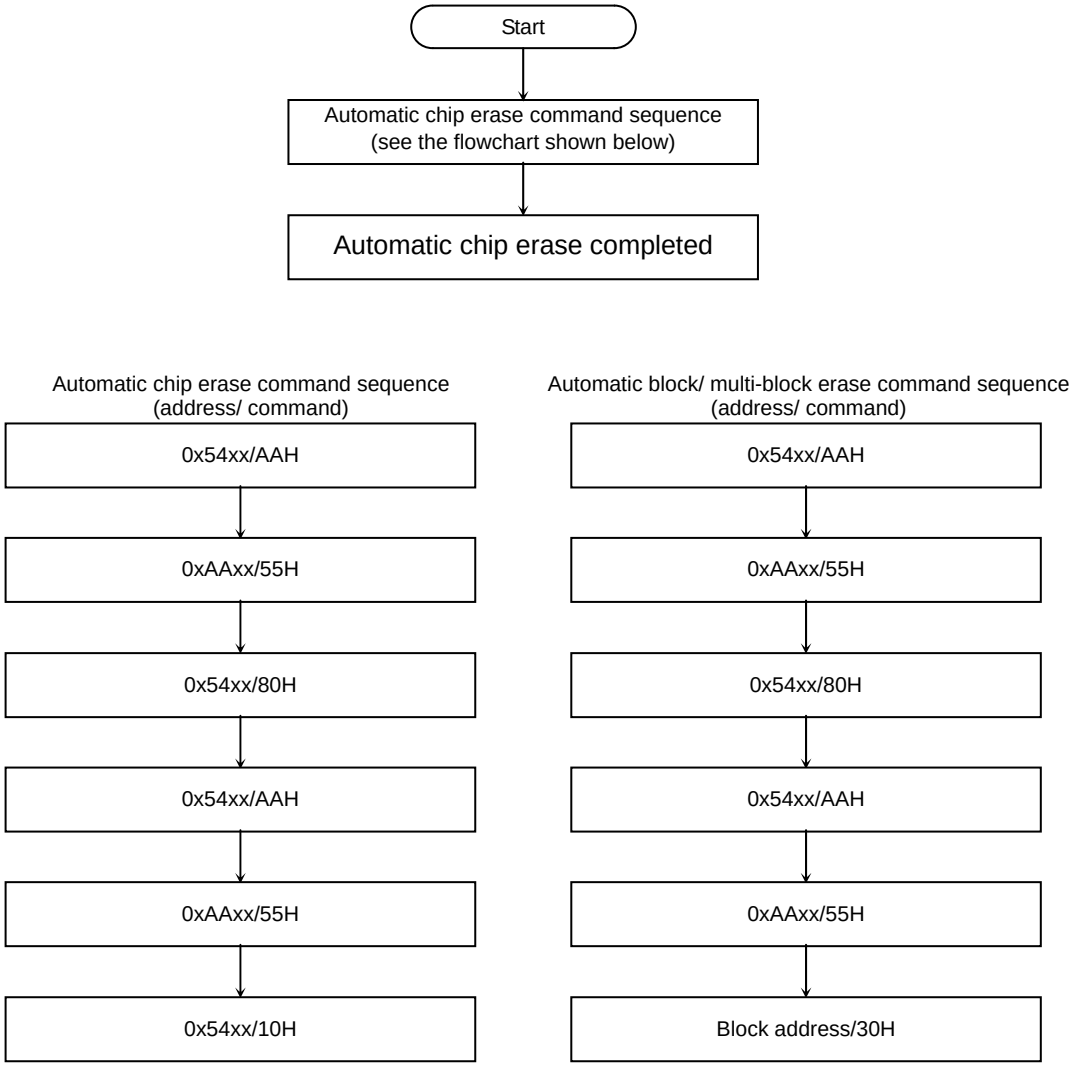


Automatic Page Programming Command Sequence (Address/ Command)



(Note) Command sequence is executed by 0x54xx or 0x55xx.

Fig. 18-12 Automatic Programming



(Note) Command sequence is executed by 0x54xx or 0x55xx.

Fig. 18-13 Automatic Erase

19 ROM protection

19.1 Outline

The TMPM330 offers two kinds of ROM protection/ security functions. One is a write/ erase-protection function for the internal flash ROM data. The other is a security function that restricts internal flash ROM data readout and debugging.

19.2 Features

19.2.1 Write/ erase-protection function

The write/ erase-protection function enables the internal flash to prohibit the writing and erasing operation for each block.

This function is available with a single chip mode, single boot mode and writer mode. To activate the function, write “1” to the corresponding bits to a block to protect. Writing “0” to the bits cancels the protection. The protection settings of the bits can be monitored by the FLCS <BLPRO> bit. See chapter 21 for programming details.

19.2.2 Security function

The security function restricts flash ROM data readout and debugging.

This function is available under the conditions shown below.

1) The SECBIT <SECBIT> bit is set to”1”.
2) All the protection bits (the FLCS<BLPRO> bits) used for the write/erase-protection function are set to “1”.

Note) The SECBIT <SECBIT> bit is set to “1” at a power-on reset right after power-on.

Table 19-1 shows details of the restrictions by the security function.

Table 19-1 Restrictions by the security function

Item	Details
1) ROM data readout	Data in the ROM area cannot be read out when writer mode is set. By executing readout, the company code 0x0098 is read. The ROM reading operation is available with a single chip mode and single boot mode.
2) Debug port	Communication of JTAG/SW and trace are prohibited.
3) Command for flash memory	Writing a command to the flash memory is prohibited. An attempt to erase the contents in the bits used for the write/erase-protection erases all the protection bits.

19.3 Register

The flash control register shows the status of the flash memory operation and the protection of each block.

Table 19-2 Flash control register

FLCS
0x4004_0520

	31	30	29	28	27	26	25	24
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read							

	23	22	21	20	19	18	17	16
bit Symbol	-	-	BLPRO5	BLPRO4	BLPRO3	BLPRO2	BLPRO1	BLPRO0
Read/Write	R		R	R	R	R	R	R
After reset	0		(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)	(Note 3)
Function	"0" is read		Protection for Block 5 (Note 2) 0: disabled 1:enabled	Protection for Block 4 (Note 2) 0: disabled 1:enabled	Protection for Block 3 0: disabled 1:enabled	Protection for Block 2 0: disabled 1:enabled	Protection for Block 1 0: disabled 1:enabled	Protection for Block 0 0: disabled 1:enabled

	15	14	13	12	11	10	9	8
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read							

	7	6	5	4	3	2	1	0
bit Symbol	-	-	-	-	-	-	-	RDY/BSY
Read/Write	R							R
After reset	0							1
Function	"0" is read							Ready/Busy (Note 1) 0:Auto operating 1:Auto operation terminated

Bit 0: Ready/Busy flag bit

The RDY/BSY output is provided as a means to monitor the status of automatic operation. This bit is a function bit for the CPU to monitor the function. When the flash memory is in automatic operation, it outputs "0" to indicate that it is busy. When the automatic operation is terminated, it returns to the ready state and outputs "1" to accept the next command. If the automatic operation has failed, this bit maintains the "0" output. By applying a hardware reset, it returns to "1."

Bit [21:16]: Protection status bits

Each of the protection bits (6 bits) represents the protection status of the corresponding block. When a bit is set to "1," it indicates that the block corresponding to the bit is protected. When the block is protected, data cannot be written to it.

(Note 1) This command must be issued in the ready state. Issuing the command in the busy state may disable both correct command transmission and further command input. To exit from the condition, execute system reset. System reset requires at least 0.5 microseconds regardless of the system clock frequency. In this condition, it takes approx. 2 ms to enable reading after reset.

(Note 2) The FLCS bits [21:20] of TMPM330FWFG have no function. They are read as "0".

(Note 3) The value varies depending on protection applied.

Table 19-3 Security bit register

SECBIT
0x4004_0500

	31	30	29	28	27	26	25	24
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read							

	23	22	21	20	19	18	17	16
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read							

	15	14	13	12	11	10	9	8
bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
After reset	0							
Function	"0" is read							

	7	6	5	4	3	2	1	0
bit Symbol	-	-	-	-	-	-	-	SECBIT
Read/Write	R							R/W
After reset	0							1
Function	"0" is read							Security bits 0:disabled 1:enabled

(Note) This register is initialized only by power-on reset.

19.4 Writing and erasing

19.4.1 Protection bits

Writing and erasing protection bits are available with a single chip mode, single boot mode and writer mode.
Writing to the protection bits is done on block-by-block basis.
Erasing of the protection bits is done by two groups of the blocks: block 0 through 3 and block 4 through 5. When the settings for all the blocks are “1”, erasing must be done after setting the SECBIT <SECBIT> bit to “0”. Setting “1” at that situation erases all the protection bits. To write and erase the protection bits, command sequence is used.
See chapter 18 for details.

19.4.2 Security bit

Rewriting of the security bits is available with a single chip mode and single boot mode. The SECBIT <SECBIT> bit that activates security function is set to “1” at a power-on reset right after power-on. The bit is rewritten by the following procedure.

1) Write the code 0xa74a9d23 to SECBIT register.
2) Write data within 16 clocks from the above.

Note) The above procedure is enabled only when using 32-bit data transfer command.

20 Special Function Registers

[1] Port registers

[2] 16-bit timer (TMRB)

[3] Serial bus interface (SBI)

[4] Serial interface (UART/SIO)

[5] 10-bit A/D converter (A/DC)

[6] Watchdog timer (WDT)

[7] Real time clock (RTC)

[8] Clock generator (CG)

[9] CEC

[10] Remote control signal preprocessor

[11] Flash

[12] Reserved area

(Note 1)	**As for the internal I/O areas (0x4000_0000~0x4007_FFFF), reading the areas not described in this chapter yields undefined value. Writing these areas is ignored.**
(Note 2)	**The <R0> areas are read as “0”. Writing the area is ignored.**
(Note 3)	**Access to the <reserved> areas is prohibited.**

20.1 Addresses

[1] Port [1/4]

<PORT A>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_0000	PA	0x4000_0010		0x4000_0020		0x4000_0030	PAPDN
0x4000_0001	<R0>	0x4000_0011		0x4000_0021		0x4000_0031	<R0>
0x4000_0002	<R0>	0x4000_0012		0x4000_0022		0x4000_0032	<R0>
0x4000_0003	<R0>	0x4000_0013		0x4000_0023		0x4000_0033	<R0>
0x4000_0004	PACR	0x4000_0014		0x4000_0024		0x4000_0034	
0x4000_0005	<R0>	0x4000_0015		0x4000_0025		0x4000_0035	
0x4000_0006	<R0>	0x4000_0016		0x4000_0026		0x4000_0036	
0x4000_0007	<R0>	0x4000_0017		0x4000_0027		0x4000_0037	
0x4000_0008	PAFR1	0x4000_0018		0x4000_0028		0x4000_0038	PAIE
0x4000_0009	<R0>	0x4000_0019		0x4000_0029		0x4000_0039	<R0>
0x4000_000A	<R0>	0x4000_001A		0x4000_002A		0x4000_003A	<R0>
0x4000_000B	<R0>	0x4000_001B		0x4000_002B		0x4000_003B	<R0>
0x4000_000C		0x4000_001C		0x4000_002C	PAPUP	0x4000_003C	
0x4000_000D		0x4000_001D		0x4000_002D	<R0>	0x4000_003D	
0x4000_000E		0x4000_001E		0x4000_002E	<R0>	0x4000_003E	
0x4000_000F		0x4000_001F		0x4000_002F	<R0>	0x4000_003F	

<PORT B>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_0040	PB	0x4000_0050		0x4000_0060		0x4000_0070	
0x4000_0041	<R0>	0x4000_0051		0x4000_0061		0x4000_0071	
0x4000_0042	<R0>	0x4000_0052		0x4000_0062		0x4000_0072	
0x4000_0043	<R0>	0x4000_0053		0x4000_0063		0x4000_0073	
0x4000_0044	PBCR	0x4000_0054		0x4000_0064		0x4000_0074	
0x4000_0045	<R0>	0x4000_0055		0x4000_0065		0x4000_0075	
0x4000_0046	<R0>	0x4000_0056		0x4000_0066		0x4000_0076	
0x4000_0047	<R0>	0x4000_0057		0x4000_0067		0x4000_0077	
0x4000_0048	PBFR1	0x4000_0058		0x4000_0068		0x4000_0078	PBIE
0x4000_0049	<R0>	0x4000_0059		0x4000_0069		0x4000_0079	<R0>
0x4000_004A	<R0>	0x4000_005A		0x4000_006A		0x4000_007A	<R0>
0x4000_004B	<R0>	0x4000_005B		0x4000_006B		0x4000_007B	<R0>
0x4000_004C		0x4000_005C		0x4000_006C	PBPUP	0x4000_007C	
0x4000_004D		0x4000_005D		0x4000_006D	<R0>	0x4000_007D	
0x4000_004E		0x4000_005E		0x4000_006E	<R0>	0x4000_007E	
0x4000_004F		0x4000_005F		0x4000_006F	<R0>	0x4000_007F	

<PORT C>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_0080	PC	0x4000_0090		0x4000_00A0		0x4000_00B0	
0x4000_0081	<R0>	0x4000_0091		0x4000_00A1		0x4000_00B1	
0x4000_0082	<R0>	0x4000_0092		0x4000_00A2		0x4000_00B2	
0x4000_0083	<R0>	0x4000_0093		0x4000_00A3		0x4000_00B3	
0x4000_0084		0x4000_0094		0x4000_00A4		0x4000_00B4	
0x4000_0085		0x4000_0095		0x4000_00A5		0x4000_00B5	
0x4000_0086		0x4000_0096		0x4000_00A6		0x4000_00B6	
0x4000_0087		0x4000_0097		0x4000_00A7		0x4000_00B7	
0x4000_0088		0x4000_0098		0x4000_00A8		0x4000_00B8	PCIE
0x4000_0089		0x4000_0099		0x4000_00A9		0x4000_00B9	<R0>
0x4000_008A		0x4000_009A		0x4000_00AA		0x4000_00BA	<R0>
0x4000_008B		0x4000_009B		0x4000_00AB		0x4000_00BB	<R0>
0x4000_008C		0x4000_009C		0x4000_00AC	PCPUP	0x4000_00BC	
0x4000_008D		0x4000_009D		0x4000_00AD	<R0>	0x4000_00BD	
0x4000_008E		0x4000_009E		0x4000_00AE	<R0>	0x4000_00BE	
0x4000_008F		0x4000_009F		0x4000_00AF	<R0>	0x4000_00BF	

[1] Port [2/4]

<PORT D>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_00C0	PD	0x4000_00D0		0x4000_00E0		0x4000_00F0	
0x4000_00C1	<R0>	0x4000_00D1		0x4000_00E1		0x4000_00F1	
0x4000_00C2	<R0>	0x4000_00D2		0x4000_00E2		0x4000_00F2	
0x4000_00C3	<R0>	0x4000_00D3		0x4000_00E3		0x4000_00F3	
0x4000_00C4		0x4000_00D4		0x4000_00E4		0x4000_00F4	
0x4000_00C5		0x4000_00D5		0x4000_00E5		0x4000_00F5	
0x4000_00C6		0x4000_00D6		0x4000_00E6		0x4000_00F6	
0x4000_00C7		0x4000_00D7		0x4000_00E7		0x4000_00F7	
0x4000_00C8	PDFR1	0x4000_00D8		0x4000_00E8		0x4000_00F8	PDIE
0x4000_00C9	<R0>	0x4000_00D9		0x4000_00E9		0x4000_00F9	<R0>
0x4000_00CA	<R0>	0x4000_00DA		0x4000_00EA		0x4000_00FA	<R0>
0x4000_00CB	<R0>	0x4000_00DB		0x4000_00EB		0x4000_00FB	<R0>
0x4000_00CC		0x4000_00DC		0x4000_00EC	PDPUP	0x4000_00FC	
0x4000_00CD		0x4000_00DD		0x4000_00ED	<R0>	0x4000_00FD	
0x4000_00CE		0x4000_00DE		0x4000_00EE	<R0>	0x4000_00FE	
0x4000_00CF		0x4000_00DF		0x4000_00EF	<R0>	0x4000_00FF	

<PORT E>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_0100	PE	0x4000_0110		0x4000_0120		0x4000_0130	
0x4000_0101	<R0>	0x4000_0111		0x4000_0121		0x4000_0131	
0x4000_0102	<R0>	0x4000_0112		0x4000_0122		0x4000_0132	
0x4000_0103	<R0>	0x4000_0113		0x4000_0123		0x4000_0133	
0x4000_0104	PECR	0x4000_0114		0x4000_0124		0x4000_0134	
0x4000_0105	<R0>	0x4000_0115		0x4000_0125		0x4000_0135	
0x4000_0106	<R0>	0x4000_0116		0x4000_0126		0x4000_0136	
0x4000_0107	<R0>	0x4000_0117		0x4000_0127		0x4000_0137	
0x4000_0108	PEFR1	0x4000_0118		0x4000_0128	PEOD	0x4000_0138	PEIE
0x4000_0109	<R0>	0x4000_0119		0x4000_0129	<R0>	0x4000_0139	<R0>
0x4000_010A	<R0>	0x4000_011A		0x4000_012A	<R0>	0x4000_013A	<R0>
0x4000_010B	<R0>	0x4000_011B		0x4000_012B	<R0>	0x4000_013B	<R0>
0x4000_010C	PEFR2	0x4000_011C		0x4000_012C	PEPUP	0x4000_013C	
0x4000_010D	<R0>	0x4000_011D		0x4000_012D	<R0>	0x4000_013D	
0x4000_010E	<R0>	0x4000_011E		0x4000_012E	<R0>	0x4000_013E	
0x4000_010F	<R0>	0x4000_011F		0x4000_012F	<R0>	0x4000_013F	

<PORT F>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_0140	PF	0x4000_0150		0x4000_0160		0x4000_0170	
0x4000_0141	<R0>	0x4000_0151		0x4000_0161		0x4000_0171	
0x4000_0142	<R0>	0x4000_0152		0x4000_0162		0x4000_0172	
0x4000_0143	<R0>	0x4000_0153		0x4000_0163		0x4000_0173	
0x4000_0144	PFCR	0x4000_0154		0x4000_0164		0x4000_0174	
0x4000_0145	<R0>	0x4000_0155		0x4000_0165		0x4000_0175	
0x4000_0146	<R0>	0x4000_0156		0x4000_0166		0x4000_0176	
0x4000_0147	<R0>	0x4000_0157		0x4000_0167		0x4000_0177	
0x4000_0148	PFFR1	0x4000_0158		0x4000_0168	PFOD	0x4000_0178	PFIE
0x4000_0149	<R0>	0x4000_0159		0x4000_0169	<R0>	0x4000_0179	<R0>
0x4000_014A	<R0>	0x4000_015A		0x4000_016A	<R0>	0x4000_017A	<R0>
0x4000_014B	<R0>	0x4000_015B		0x4000_016B	<R0>	0x4000_017B	<R0>
0x4000_014C	PFFR2	0x4000_015C		0x4000_016C	PFPUP	0x4000_017C	
0x4000_014D	<R0>	0x4000_015D		0x4000_016D	<R0>	0x4000_017D	
0x4000_014E	<R0>	0x4000_015E		0x4000_016E	<R0>	0x4000_017E	
0x4000_014F	<R0>	0x4000_015F		0x4000_016F	<R0>	0x4000_017F	

[1] Port 3/4]

<PORT G>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_0180	PG	0x4000_0190	<reserved>	0x4000_01A0		0x4000_01B0	
0x4000_0181	<R0>	0x4000_0191	<reserved>	0x4000_01A1		0x4000_01B1	
0x4000_0182	<R0>	0x4000_0192	<reserved>	0x4000_01A2		0x4000_01B2	
0x4000_0183	<R0>	0x4000_0193	<reserved>	0x4000_01A3		0x4000_01B3	
0x4000_0184	PGCR	0x4000_0194		0x4000_01A4		0x4000_01B4	
0x4000_0185	<R0>	0x4000_0195		0x4000_01A5		0x4000_01B5	
0x4000_0186	<R0>	0x4000_0196		0x4000_01A6		0x4000_01B6	
0x4000_0187	<R0>	0x4000_0197		0x4000_01A7		0x4000_01B7	
0x4000_0188	PGFR1	0x4000_0198		0x4000_01A8	PGOD	0x4000_01B8	PGIE
0x4000_0189	<R0>	0x4000_0199		0x4000_01A9	<R0>	0x4000_01B9	<R0>
0x4000_018A	<R0>	0x4000_019A		0x4000_01AA	<R0>	0x4000_01BA	<R0>
0x4000_018B	<R0>	0x4000_019B		0x4000_01AB	<R0>	0x4000_01BB	<R0>
0x4000_018C		0x4000_019C		0x4000_01AC	PGPUP	0x4000_01BC	
0x4000_018D		0x4000_019D		0x4000_01AD	<R0>	0x4000_01BD	
0x4000_018E		0x4000_019E		0x4000_01AE	<R0>	0x4000_01BE	
0x4000_018F		0x4000_019F		0x4000_01AF	<R0>	0x4000_01BF	

<PORT H>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_01C0	PH	0x4000_01D0	<reserved>	0x4000_01E0		0x4000_01F0	
0x4000_01C1	<R0>	0x4000_01D1	<reserved>	0x4000_01E1		0x4000_01F1	
0x4000_01C2	<R0>	0x4000_01D2	<reserved>	0x4000_01E2		0x4000_01F2	
0x4000_01C3	<R0>	0x4000_01D3	<reserved>	0x4000_01E3		0x4000_01F3	
0x4000_01C4	PHCR	0x4000_01D4		0x4000_01E4		0x4000_01F4	
0x4000_01C5	<R0>	0x4000_01D5		0x4000_01E5		0x4000_01F5	
0x4000_01C6	<R0>	0x4000_01D6		0x4000_01E6		0x4000_01F6	
0x4000_01C7	<R0>	0x4000_01D7		0x4000_01E7		0x4000_01F7	
0x4000_01C8	PHFR1	0x4000_01D8		0x4000_01E8		0x4000_01F8	PHIE
0x4000_01C9	<R0>	0x4000_01D9		0x4000_01E9		0x4000_01F9	<R0>
0x4000_01CA	<R0>	0x4000_01DA		0x4000_01EA		0x4000_01FA	<R0>
0x4000_01CB	<R0>	0x4000_01DB		0x4000_01EB		0x4000_01FB	<R0>
0x4000_01CC		0x4000_01DC		0x4000_01EC	PHPUP	0x4000_01FC	
0x4000_01CD		0x4000_01DD		0x4000_01ED	<R0>	0x4000_01FD	
0x4000_01CE		0x4000_01DE		0x4000_01EE	<R0>	0x4000_01FE	
0x4000_01CF		0x4000_01DF		0x4000_01EF	<R0>	0x4000_01FF	

<PORT I>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_0200	PI	0x4000_0210	<reserved>	0x4000_0220		0x4000_0230	
0x4000_0201	<R0>	0x4000_0211	<reserved>	0x4000_0221		0x4000_0231	
0x4000_0202	<R0>	0x4000_0212	<reserved>	0x4000_0222		0x4000_0232	
0x4000_0203	<R0>	0x4000_0213	<reserved>	0x4000_0223		0x4000_0233	
0x4000_0204	PICR	0x4000_0214		0x4000_0224		0x4000_0234	
0x4000_0205	<R0>	0x4000_0215		0x4000_0225		0x4000_0235	
0x4000_0206	<R0>	0x4000_0216		0x4000_0226		0x4000_0236	
0x4000_0207	<R0>	0x4000_0217		0x4000_0227		0x4000_0237	
0x4000_0208	PIFR1	0x4000_0218		0x4000_0228		0x4000_0238	PIIE
0x4000_0209	<R0>	0x4000_0219		0x4000_0229		0x4000_0239	<R0>
0x4000_020A	<R0>	0x4000_021A		0x4000_022A		0x4000_023A	<R0>
0x4000_020B	<R0>	0x4000_021B		0x4000_022B		0x4000_023B	<R0>
0x4000_020C		0x4000_021C		0x4000_022C	PIPUP	0x4000_023C	
0x4000_020D		0x4000_021D		0x4000_022D	<R0>	0x4000_023D	
0x4000_020E		0x4000_021E		0x4000_022E	<R0>	0x4000_023E	
0x4000_020F		0x4000_021F		0x4000_022F	<R0>	0x4000_023F	

[1] Port [4/4]

<PORT J>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_0240	PJ	0x4000_0250	<reserved>	0x4000_0260		0x4000_0270	
0x4000_0241	<R0>	0x4000_0251	<reserved>	0x4000_0261		0x4000_0271	
0x4000_0242	<R0>	0x4000_0252	<reserved>	0x4000_0262		0x4000_0272	
0x4000_0243	<R0>	0x4000_0253	<reserved>	0x4000_0263		0x4000_0273	
0x4000_0244	PJCR	0x4000_0254		0x4000_0264		0x4000_0274	
0x4000_0245	<R0>	0x4000_0255		0x4000_0265		0x4000_0275	
0x4000_0246	<R0>	0x4000_0256		0x4000_0266		0x4000_0276	
0x4000_0247	<R0>	0x4000_0257		0x4000_0267		0x4000_0277	
0x4000_0248	PJFR1	0x4000_0258		0x4000_0268		0x4000_0278	PJIE
0x4000_0249	<R0>	0x4000_0259		0x4000_0269		0x4000_0279	<R0>
0x4000_024A	<R0>	0x4000_025A		0x4000_026A		0x4000_027A	<R0>
0x4000_024B	<R0>	0x4000_025B		0x4000_026B		0x4000_027B	<R0>
0x4000_024C		0x4000_025C		0x4000_026C	PJPUP	0x4000_027C	
0x4000_024D		0x4000_025D		0x4000_026D	<R0>	0x4000_027D	
0x4000_024E		0x4000_025E		0x4000_026E	<R0>	0x4000_027E	
0x4000_024F		0x4000_025F		0x4000_026F	<R0>	0x4000_027F	

<PORT K>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4000_0280	PK	0x4000_0290		0x4000_02A0		0x4000_02B0	
0x4000_0281	<R0>	0x4000_0291		0x4000_02A1		0x4000_02B1	
0x4000_0282	<R0>	0x4000_0292		0x4000_02A2		0x4000_02B2	
0x4000_0283	<R0>	0x4000_0293		0x4000_02A3		0x4000_02B3	
0x4000_0284	PKCR	0x4000_0294		0x4000_02A4		0x4000_02B4	
0x4000_0285	<R0>	0x4000_0295		0x4000_02A5		0x4000_02B5	
0x4000_0286	<R0>	0x4000_0296		0x4000_02A6		0x4000_02B6	
0x4000_0287	<R0>	0x4000_0297		0x4000_02A7		0x4000_02B7	
0x4000_0288	PKFR1	0x4000_0298		0x4000_02A8		0x4000_02B8	PKIE
0x4000_0289	<R0>	0x4000_0299		0x4000_02A9		0x4000_02B9	<R0>
0x4000_028A	<R0>	0x4000_029A		0x4000_02AA		0x4000_02BA	<R0>
0x4000_028B	<R0>	0x4000_029B		0x4000_02AB		0x4000_02BB	<R0>
0x4000_028C	PKFR2	0x4000_029C		0x4000_02AC	PKPUP	0x4000_02BC	
0x4000_028D	<R0>	0x4000_029D		0x4000_02AD	<R0>	0x4000_02BD	
0x4000_028E	<R0>	0x4000_029E		0x4000_02AE	<R0>	0x4000_02BE	
0x4000_028F	<R0>	0x4000_029F		0x4000_02AF	<R0>	0x4000_02BF	

[2] 16-bit timer [1/4]

<TMRB0>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_0000	TB0EN	0x4001_0010	TB0FFCR	0x4001_0020	TB0RG0L	0x4001_0030	
0x4001_0001	<R0>	0x4001_0011	<R0>	0x4001_0021	TB0RG0H	0x4001_0031	
0x4001_0002	<R0>	0x4001_0012	<R0>	0x4001_0022	<R0>	0x4001_0032	
0x4001_0003	<R0>	0x4001_0013	<R0>	0x4001_0023	<R0>	0x4001_0033	
0x4001_0004	TB0RUN	0x4001_0014	TB0ST	0x4001_0024	TB0RG1L	0x4001_0034	
0x4001_0005	<R0>	0x4001_0015	<R0>	0x4001_0025	TB0RG1H	0x4001_0035	
0x4001_0006	<R0>	0x4001_0016	<R0>	0x4001_0026	<R0>	0x4001_0036	
0x4001_0007	<R0>	0x4001_0017	<R0>	0x4001_0027	<R0>	0x4001_0037	
0x4001_0008	TB0CR	0x4001_0018	TB0IM	0x4001_0028	TB0CP0L	0x4001_0038	
0x4001_0009	<R0>	0x4001_0019	<R0>	0x4001_0029	TB0CP0H	0x4001_0039	
0x4001_000A	<R0>	0x4001_001A	<R0>	0x4001_002A	<R0>	0x4001_003A	
0x4001_000B	<R0>	0x4001_001B	<R0>	0x4001_002B	<R0>	0x4001_003B	
0x4001_000C	TB0MOD	0x4001_001C	TB0UCL	0x4001_002C	TB0CP1L	0x4001_003C	
0x4001_000D	<R0>	0x4001_001D	TB0UCH	0x4001_002D	TB0CP1H	0x4001_003D	
0x4001_000E	<R0>	0x4001_001E	<R0>	0x4001_002E	<R0>	0x4001_003E	
0x4001_000F	<R0>	0x4001_001F	<R0>	0x4001_002F	<R0>	0x4001_003F	

<TMRB1>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_0040	TB1EN	0x4001_0050	TB1FFCR	0x4001_0060	TB1RG0L	0x4001_0070	
0x4001_0041	<R0>	0x4001_0051	<R0>	0x4001_0061	TB1RG0H	0x4001_0071	
0x4001_0042	<R0>	0x4001_0052	<R0>	0x4001_0062	<R0>	0x4001_0072	
0x4001_0043	<R0>	0x4001_0053	<R0>	0x4001_0063	<R0>	0x4001_0073	
0x4001_0044	TB1RUN	0x4001_0054	TB1ST	0x4001_0064	TB1RG1L	0x4001_0074	
0x4001_0045	<R0>	0x4001_0055	<R0>	0x4001_0065	TB1RG1H	0x4001_0075	
0x4001_0046	<R0>	0x4001_0056	<R0>	0x4001_0066	<R0>	0x4001_0076	
0x4001_0047	<R0>	0x4001_0057	<R0>	0x4001_0067	<R0>	0x4001_0077	
0x4001_0048	TB1CR	0x4001_0058	TB1IM	0x4001_0068	TB1CP0L	0x4001_0078	
0x4001_0049	<R0>	0x4001_0059	<R0>	0x4001_0069	TB1CP0H	0x4001_0079	
0x4001_004A	<R0>	0x4001_005A	<R0>	0x4001_006A	<R0>	0x4001_007A	
0x4001_004B	<R0>	0x4001_005B	<R0>	0x4001_006B	<R0>	0x4001_007B	
0x4001_004C	TB1MOD	0x4001_005C	TB1UCL	0x4001_006C	TB1CP1L	0x4001_007C	
0x4001_004D	<R0>	0x4001_005D	TB1UCH	0x4001_006D	TB1CP1H	0x4001_007D	
0x4001_004E	<R0>	0x4001_005E	<R0>	0x4001_006E	<R0>	0x4001_007E	
0x4001_004F	<R0>	0x4001_005F	<R0>	0x4001_006F	<R0>	0x4001_007F	

<TMRB2>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_0080	TB2EN	0x4001_0090	TB2FFCR	0x4001_00A0	TB2RG0L	0x4001_00B0	
0x4001_0081	<R0>	0x4001_0091	<R0>	0x4001_00A1	TB2RG0H	0x4001_00B1	
0x4001_0082	<R0>	0x4001_0092	<R0>	0x4001_00A2	<R0>	0x4001_00B2	
0x4001_0083	<R0>	0x4001_0093	<R0>	0x4001_00A3	<R0>	0x4001_00B3	
0x4001_0084	TB2RUN	0x4001_0094	TB2ST	0x4001_00A4	TB2RG1L	0x4001_00B4	
0x4001_0085	<R0>	0x4001_0095	<R0>	0x4001_00A5	TB2RG1H	0x4001_00B5	
0x4001_0086	<R0>	0x4001_0096	<R0>	0x4001_00A6	<R0>	0x4001_00B6	
0x4001_0087	<R0>	0x4001_0097	<R0>	0x4001_00A7	<R0>	0x4001_00B7	
0x4001_0088	TB2CR	0x4001_0098	TB2IM	0x4001_00A8	TB2CP0L	0x4001_00B8	
0x4001_0089	<R0>	0x4001_0099	<R0>	0x4001_00A9	TB2CP0H	0x4001_00B9	
0x4001_008A	<R0>	0x4001_009A	<R0>	0x4001_00AA	<R0>	0x4001_00BA	
0x4001_008B	<R0>	0x4001_009B	<R0>	0x4001_00AB	<R0>	0x4001_00BB	
0x4001_008C	TB2MOD	0x4001_009C	TB2UCL	0x4001_00AC	TB2CP1L	0x4001_00BC	
0x4001_008D	<R0>	0x4001_009D	TB2UCH	0x4001_00AD	TB2CP1H	0x4001_00BD	
0x4001_008E	<R0>	0x4001_009E	<R0>	0x4001_00AE	<R0>	0x4001_00BE	
0x4001_008F	<R0>	0x4001_009F	<R0>	0x4001_00AF	<R0>	0x4001_00BF	

[2] 16-bit timer [2/4]

<TMRB3>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_00C0	TB3EN	0x4001_00D0	TB3FFCR	0x4001_00E0	TB3RG0L	0x4001_00F0	
0x4001_00C1	<R0>	0x4001_00D1	<R0>	0x4001_00E1	TB3RG0H	0x4001_00F1	
0x4001_00C2	<R0>	0x4001_00D2	<R0>	0x4001_00E2	<R0>	0x4001_00F2	
0x4001_00C3	<R0>	0x4001_00D3	<R0>	0x4001_00E3	<R0>	0x4001_00F3	
0x4001_00C4	TB3RUN	0x4001_00D4	TB3ST	0x4001_00E4	TB3RG1L	0x4001_00F4	
0x4001_00C5	<R0>	0x4001_00D5	<R0>	0x4001_00E5	TB3RG1H	0x4001_00F5	
0x4001_00C6	<R0>	0x4001_00D6	<R0>	0x4001_00E6	<R0>	0x4001_00F6	
0x4001_00C7	<R0>	0x4001_00D7	<R0>	0x4001_00E7	<R0>	0x4001_00F7	
0x4001_00C8	TB3CR	0x4001_00D8	TB3IM	0x4001_00E8	TB3CP0L	0x4001_00F8	
0x4001_00C9	<R0>	0x4001_00D9	<R0>	0x4001_00E9	TB3CP0H	0x4001_00F9	
0x4001_00CA	<R0>	0x4001_00DA	<R0>	0x4001_00EA	<R0>	0x4001_00FA	
0x4001_00CB	<R0>	0x4001_00DB	<R0>	0x4001_00EB	<R0>	0x4001_00FB	
0x4001_00CC	TB3MOD	0x4001_00DC	TB3UCL	0x4001_00EC	TB3CP1L	0x4001_00FC	
0x4001_00CD	<R0>	0x4001_00DD	TB3UCH	0x4001_00ED	TB3CP1H	0x4001_00FD	
0x4001_00CE	<R0>	0x4001_00DE	<R0>	0x4001_00EE	<R0>	0x4001_00FE	
0x4001_00CF	<R0>	0x4001_00DF	<R0>	0x4001_00EF	<R0>	0x4001_00FF	

<TMRB4>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_0100	TB4EN	0x4001_0110	TB4FFCR	0x4001_0120	TB4RG0L	0x4001_0130	
0x4001_0101	<R0>	0x4001_0111	<R0>	0x4001_0121	TB4RG0H	0x4001_0131	
0x4001_0102	<R0>	0x4001_0112	<R0>	0x4001_0122	<R0>	0x4001_0132	
0x4001_0103	<R0>	0x4001_0113	<R0>	0x4001_0123	<R0>	0x4001_0133	
0x4001_0104	TB4RUN	0x4001_0114	TB4ST	0x4001_0124	TB4RG1L	0x4001_0134	
0x4001_0105	<R0>	0x4001_0115	<R0>	0x4001_0125	TB4RG1H	0x4001_0135	
0x4001_0106	<R0>	0x4001_0116	<R0>	0x4001_0126	<R0>	0x4001_0136	
0x4001_0107	<R0>	0x4001_0117	<R0>	0x4001_0127	<R0>	0x4001_0137	
0x4001_0108	TB4CR	0x4001_0118	TB4IM	0x4001_0128	TB4CP0L	0x4001_0138	
0x4001_0109	<R0>	0x4001_0119	<R0>	0x4001_0129	TB4CP0H	0x4001_0139	
0x4001_010A	<R0>	0x4001_011A	<R0>	0x4001_012A	<R0>	0x4001_013A	
0x4001_010B	<R0>	0x4001_011B	<R0>	0x4001_012B	<R0>	0x4001_013B	
0x4001_010C	TB4MOD	0x4001_011C	TB4UCL	0x4001_012C	TB4CP1L	0x4001_013C	
0x4001_010D	<R0>	0x4001_011D	TB4UCH	0x4001_012D	TB4CP1H	0x4001_013D	
0x4001_010E	<R0>	0x4001_011E	<R0>	0x4001_012E	<R0>	0x4001_013E	
0x4001_010F	<R0>	0x4001_011F	<R0>	0x4001_012F	<R0>	0x4001_013F	

<TMRB5>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_0140	TB5EN	0x4001_0150	TB5FFCR	0x4001_0160	TB5RG0L	0x4001_0170	
0x4001_0141	<R0>	0x4001_0151	<R0>	0x4001_0161	TB5RG0H	0x4001_0171	
0x4001_0142	<R0>	0x4001_0152	<R0>	0x4001_0162	<R0>	0x4001_0172	
0x4001_0143	<R0>	0x4001_0153	<R0>	0x4001_0163	<R0>	0x4001_0173	
0x4001_0144	TB5RUN	0x4001_0154	TB5ST	0x4001_0164	TB5RG1L	0x4001_0174	
0x4001_0145	<R0>	0x4001_0155	<R0>	0x4001_0165	TB5RG1H	0x4001_0175	
0x4001_0146	<R0>	0x4001_0156	<R0>	0x4001_0166	<R0>	0x4001_0176	
0x4001_0147	<R0>	0x4001_0157	<R0>	0x4001_0167	<R0>	0x4001_0177	
0x4001_0148	TB5CR	0x4001_0158	TB5IM	0x4001_0168	TB5CP0L	0x4001_0178	
0x4001_0149	<R0>	0x4001_0159	<R0>	0x4001_0169	TB5CP0H	0x4001_0179	
0x4001_014A	<R0>	0x4001_015A	<R0>	0x4001_016A	<R0>	0x4001_017A	
0x4001_014B	<R0>	0x4001_015B	<R0>	0x4001_016B	<R0>	0x4001_017B	
0x4001_014C	TB5MOD	0x4001_015C	TB5UCL	0x4001_016C	TB5CP1L	0x4001_017C	
0x4001_014D	<R0>	0x4001_015D	TB5UCH	0x4001_016D	TB5CP1H	0x4001_017D	
0x4001_014E	<R0>	0x4001_015E	<R0>	0x4001_016E	<R0>	0x4001_017E	
0x4001_014F	<R0>	0x4001_015F	<R0>	0x4001_016F	<R0>	0x4001_017F	

[2] 16-bit timer [3/4]

<TMRB6>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_0180	TB6EN	0x4001_0190	TB6FFCR	0x4001_01A0	TB6RG0L	0x4001_01B0	
0x4001_0181	<R0>	0x4001_0191	<R0>	0x4001_01A1	TB6RG0H	0x4001_01B1	
0x4001_0182	<R0>	0x4001_0192	<R0>	0x4001_01A2	<R0>	0x4001_01B2	
0x4001_0183	<R0>	0x4001_0193	<R0>	0x4001_01A3	<R0>	0x4001_01B3	
0x4001_0184	TB6RUN	0x4001_0194	TB6ST	0x4001_01A4	TB6RG1L	0x4001_01B4	
0x4001_0185	<R0>	0x4001_0195	<R0>	0x4001_01A5	TB6RG1H	0x4001_01B5	
0x4001_0186	<R0>	0x4001_0196	<R0>	0x4001_01A6	<R0>	0x4001_01B6	
0x4001_0187	<R0>	0x4001_0197	<R0>	0x4001_01A7	<R0>	0x4001_01B7	
0x4001_0188	TB6CR	0x4001_0198	TB6IM	0x4001_01A8	TB6CP0L	0x4001_01B8	
0x4001_0189	<R0>	0x4001_0199	<R0>	0x4001_01A9	TB6CP0H	0x4001_01B9	
0x4001_018A	<R0>	0x4001_019A	<R0>	0x4001_01AA	<R0>	0x4001_01BA	
0x4001_018B	<R0>	0x4001_019B	<R0>	0x4001_01AB	<R0>	0x4001_01BB	
0x4001_018C	TB6MOD	0x4001_019C	TB6UCL	0x4001_01AC	TB6CP1L	0x4001_01BC	
0x4001_018D	<R0>	0x4001_019D	TB6UCH	0x4001_01AD	TB6CP1H	0x4001_01BD	
0x4001_018E	<R0>	0x4001_019E	<R0>	0x4001_01AE	<R0>	0x4001_01BE	
0x4001_018F	<R0>	0x4001_019F	<R0>	0x4001_01AF	<R0>	0x4001_01BF	

<TMRB7>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_01C0	TB7EN	0x4001_01D0	TB7FFCR	0x4001_01E0	TB7RG0L	0x4001_01F0	
0x4001_01C1	<R0>	0x4001_01D1	<R0>	0x4001_01E1	TB7RG0H	0x4001_01F1	
0x4001_01C2	<R0>	0x4001_01D2	<R0>	0x4001_01E2	<R0>	0x4001_01F2	
0x4001_01C3	<R0>	0x4001_01D3	<R0>	0x4001_01E3	<R0>	0x4001_01F3	
0x4001_01C4	TB7RUN	0x4001_01D4	TB7ST	0x4001_01E4	TB7RG1L	0x4001_01F4	
0x4001_01C5	<R0>	0x4001_01D5	<R0>	0x4001_01E5	TB7RG1H	0x4001_01F5	
0x4001_01C6	<R0>	0x4001_01D6	<R0>	0x4001_01E6	<R0>	0x4001_01F6	
0x4001_01C7	<R0>	0x4001_01D7	<R0>	0x4001_01E7	<R0>	0x4001_01F7	
0x4001_01C8	TB7CR	0x4001_01D8	TB7IM	0x4001_01E8	TB7CP0L	0x4001_01F8	
0x4001_01C9	<R0>	0x4001_01D9	<R0>	0x4001_01E9	TB7CP0H	0x4001_01F9	
0x4001_01CA	<R0>	0x4001_01DA	<R0>	0x4001_01EA	<R0>	0x4001_01FA	
0x4001_01CB	<R0>	0x4001_01DB	<R0>	0x4001_01EB	<R0>	0x4001_01FB	
0x4001_01CC	TB7MOD	0x4001_01DC	TB7UCL	0x4001_01EC	TB7CP1L	0x4001_01FC	
0x4001_01CD	<R0>	0x4001_01DD	TB7UCH	0x4001_01ED	TB7CP1H	0x4001_01FD	
0x4001_01CE	<R0>	0x4001_01DE	<R0>	0x4001_01EE	<R0>	0x4001_01FE	
0x4001_01CF	<R0>	0x4001_01DF	<R0>	0x4001_01EF	<R0>	0x4001_01FF	

<TMRB8>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_0200	TB8EN	0x4001_0210	TB8FFCR	0x4001_0220	TB8RG0L	0x4001_0230	
0x4001_0201	<R0>	0x4001_0211	<R0>	0x4001_0221	TB8RG0H	0x4001_0231	
0x4001_0202	<R0>	0x4001_0212	<R0>	0x4001_0222	<R0>	0x4001_0232	
0x4001_0203	<R0>	0x4001_0213	<R0>	0x4001_0223	<R0>	0x4001_0233	
0x4001_0204	TB8RUN	0x4001_0214	TB8ST	0x4001_0224	TB8RG1L	0x4001_0234	
0x4001_0205	<R0>	0x4001_0215	<R0>	0x4001_0225	TB8RG1H	0x4001_0235	
0x4001_0206	<R0>	0x4001_0216	<R0>	0x4001_0226	<R0>	0x4001_0236	
0x4001_0207	<R0>	0x4001_0217	<R0>	0x4001_0227	<R0>	0x4001_0237	
0x4001_0208	TB8CR	0x4001_0218	TB8IM	0x4001_0228	TB8CP0L	0x4001_0238	
0x4001_0209	<R0>	0x4001_0219	<R0>	0x4001_0229	TB8CP0H	0x4001_0239	
0x4001_020A	<R0>	0x4001_021A	<R0>	0x4001_022A	<R0>	0x4001_023A	
0x4001_020B	<R0>	0x4001_021B	<R0>	0x4001_022B	<R0>	0x4001_023B	
0x4001_020C	TB8MOD	0x4001_021C	TB8UCL	0x4001_022C	TB8CP1L	0x4001_023C	
0x4001_020D	<R0>	0x4001_021D	TB8UCH	0x4001_022D	TB8CP1H	0x4001_023D	
0x4001_020E	<R0>	0x4001_021E	<R0>	0x4001_022E	<R0>	0x4001_023E	
0x4001_020F	<R0>	0x4001_021F	<R0>	0x4001_022F	<R0>	0x4001_023F	

[2] 16-bit timer [4/4]
<TMRB9>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4001_0240	TB9EN	0x4001_0250	TB9FFCR	0x4001_0260	TB9RG0L	0x4001_0270	
0x4001_0241	<R0>	0x4001_0251	<R0>	0x4001_0261	TB9RG0H	0x4001_0271	
0x4001_0242	<R0>	0x4001_0252	<R0>	0x4001_0262	<R0>	0x4001_0272	
0x4001_0243	<R0>	0x4001_0253	<R0>	0x4001_0263	<R0>	0x4001_0273	
0x4001_0244	TB9RUN	0x4001_0254	TB9ST	0x4001_0264	TB9RG1L	0x4001_0274	
0x4001_0245	<R0>	0x4001_0255	<R0>	0x4001_0265	TB9RG1H	0x4001_0275	
0x4001_0246	<R0>	0x4001_0256	<R0>	0x4001_0266	<R0>	0x4001_0276	
0x4001_0247	<R0>	0x4001_0257	<R0>	0x4001_0267	<R0>	0x4001_0277	
0x4001_0248	TB9CR	0x4001_0258	TB9IM	0x4001_0268	TB9CP0L	0x4001_0278	
0x4001_0249	<R0>	0x4001_0259	<R0>	0x4001_0269	TB9CP0H	0x4001_0279	
0x4001_024A	<R0>	0x4001_025A	<R0>	0x4001_026A	<R0>	0x4001_027A	
0x4001_024B	<R0>	0x4001_025B	<R0>	0x4001_026B	<R0>	0x4001_027B	
0x4001_024C	TB9MOD	0x4001_025C	TB9UCL	0x4001_026C	TB9CP1L	0x4001_027C	
0x4001_024D	<R0>	0x4001_025D	TB9UCH	0x4001_026D	TB9CP1H	0x4001_027D	
0x4001_024E	<R0>	0x4001_025E	<R0>	0x4001_026E	<R0>	0x4001_027E	
0x4001_024F	<R0>	0x4001_025F	<R0>	0x4001_026F	<R0>	0x4001_027F	

[3] Serial bus interface (SBI)

<SBI0>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4002_0000	SBI0CR0	0x4002_0010	SBI0CR2/SR	0x4002_0020	SBI1CR0	0x4002_0030	SBI1CR2/SR
0x4002_0001	<R0>	0x4002_0011	<R0>	0x4002_0021	<R0>	0x4002_0031	<R0>
0x4002_0002	<R0>	0x4002_0012	<R0>	0x4002_0022	<R0>	0x4002_0032	<R0>
0x4002_0003	<R0>	0x4002_0013	<R0>	0x4002_0023	<R0>	0x4002_0033	<R0>
0x4002_0004	SBI0CR1	0x4002_0014	SBI0BR0	0x4002_0024	SBI1CR1	0x4002_0034	SBI1BR0
0x4002_0005	<R0>	0x4002_0015	<R0>	0x4002_0025	<R0>	0x4002_0035	<R0>
0x4002_0006	<R0>	0x4002_0016	<R0>	0x4002_0026	<R0>	0x4002_0036	<R0>
0x4002_0007	<R0>	0x4002_0017	<R0>	0x4002_0027	<R0>	0x4002_0037	<R0>
0x4002_0008	SBI0DBR	0x4002_0018		0x4002_0028	SBI1DBR	0x4002_0038	
0x4002_0009	<R0>	0x4002_0019		0x4002_0029	<R0>	0x4002_0039	
0x4002_000A	<R0>	0x4002_001A		0x4002_002A	<R0>	0x4002_003A	
0x4002_000B	<R0>	0x4002_001B		0x4002_002B	<R0>	0x4002_003B	
0x4002_000C	SBI0I2CAR	0x4002_001C		0x4002_002C	SBI1I2CAR	0x4002_003C	
0x4002_000D	<R0>	0x4002_001D		0x4002_002D	<R0>	0x4002_003D	
0x4002_000E	<R0>	0x4002_001E		0x4002_002E	<R0>	0x4002_003E	
0x4002_000F	<R0>	0x4002_001F		0x4002_002F	<R0>	0x4002_003F	

<SBI1>

<SBI2>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4002_0040	SBI2CR0	0x4002_0050	SBI2CR2/SR	0x4002_0060		0x4002_0070	
0x4002_0041	<R0>	0x4002_0051	<R0>	0x4002_0061		0x4002_0071	
0x4002_0042	<R0>	0x4002_0052	<R0>	0x4002_0062		0x4002_0072	
0x4002_0043	<R0>	0x4002_0053	<R0>	0x4002_0063		0x4002_0073	
0x4002_0044	SBI2CR1	0x4002_0054	SBI2BR0	0x4002_0064		0x4002_0074	
0x4002_0045	<R0>	0x4002_0055	<R0>	0x4002_0065		0x4002_0075	
0x4002_0046	<R0>	0x4002_0056	<R0>	0x4002_0066		0x4002_0076	
0x4002_0047	<R0>	0x4002_0057	<R0>	0x4002_0067		0x4002_0077	
0x4002_0048	SBI2DBR	0x4002_0058		0x4002_0068		0x4002_0078	
0x4002_0049	<R0>	0x4002_0059		0x4002_0069		0x4002_0079	
0x4002_004A	<R0>	0x4002_005A		0x4002_006A		0x4002_007A	
0x4002_004B	<R0>	0x4002_005B		0x4002_006B		0x4002_007B	
0x4002_004C	SBI2I2CAR	0x4002_005C		0x4002_006C		0x4002_007C	
0x4002_004D	<R0>	0x4002_005D		0x4002_006D		0x4002_007D	
0x4002_004E	<R0>	0x4002_005E		0x4002_006E		0x4002_007E	
0x4002_004F	<R0>	0x4002_005F		0x4002_006F		0x4002_007F	

[4] Serial interface (UART/SIO)

<SIO0>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4002_0080	SC0EN	0x4002_0090	SC0BRCR	0x4002_00A0	SC0RFC	0x4002_00B0	SC0FCNF
0x4002_0081	<R0>	0x4002_0091	<R0>	0x4002_00A1	<R0>	0x4002_00B1	<R0>
0x4002_0082	<R0>	0x4002_0092	<R0>	0x4002_00A2	<R0>	0x4002_00B2	<R0>
0x4002_0083	<R0>	0x4002_0093	<R0>	0x4002_00A3	<R0>	0x4002_00B3	<R0>
0x4002_0084	SC0BUF	0x4002_0094	SC0BRADD	0x4002_00A4	SC0TFC	0x4002_00B4	
0x4002_0085	<R0>	0x4002_0095	<R0>	0x4002_00A5	<R0>	0x4002_00B5	
0x4002_0086	<R0>	0x4002_0096	<R0>	0x4002_00A6	<R0>	0x4002_00B6	
0x4002_0087	<R0>	0x4002_0097	<R0>	0x4002_00A7	<R0>	0x4002_00B7	
0x4002_0088	SC0CR	0x4002_0098	SC0MOD1	0x4002_00A8	SC0RST	0x4002_00B8	
0x4002_0089	<R0>	0x4002_0099	<R0>	0x4002_00A9	<R0>	0x4002_00B9	
0x4002_008A	<R0>	0x4002_009A	<R0>	0x4002_00AA	<R0>	0x4002_00BA	
0x4002_008B	<R0>	0x4002_009B	<R0>	0x4002_00AB	<R0>	0x4002_00BB	
0x4002_008C	SC0MOD0	0x4002_009C	SC0MOD2	0x4002_00AC	SC0TST	0x4002_00BC	
0x4002_008D	<R0>	0x4002_009D	<R0>	0x4002_00AD	<R0>	0x4002_00BD	
0x4002_008E	<R0>	0x4002_009E	<R0>	0x4002_00AE	<R0>	0x4002_00BE	
0x4002_008F	<R0>	0x4002_009F	<R0>	0x4002_00AF	<R0>	0x4002_00BF	

<SIO1>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4002_00C0	SC1EN	0x4002_00D0	SC1BRCR	0x4002_00E0	SC1RFC	0x4002_00F0	SC1FCNF
0x4002_00C1	<R0>	0x4002_00D1	<R0>	0x4002_00E1	<R0>	0x4002_00F1	<R0>
0x4002_00C2	<R0>	0x4002_00D2	<R0>	0x4002_00E2	<R0>	0x4002_00F2	<R0>
0x4002_00C3	<R0>	0x4002_00D3	<R0>	0x4002_00E3	<R0>	0x4002_00F3	<R0>
0x4002_00C4	SC1BUF	0x4002_00D4	SC1BRADD	0x4002_00E4	SC1TFC	0x4002_00F4	
0x4002_00C5	<R0>	0x4002_00D5	<R0>	0x4002_00E5	<R0>	0x4002_00F5	
0x4002_00C6	<R0>	0x4002_00D6	<R0>	0x4002_00E6	<R0>	0x4002_00F6	
0x4002_00C7	<R0>	0x4002_00D7	<R0>	0x4002_00E7	<R0>	0x4002_00F7	
0x4002_00C8	SC1CR	0x4002_00D8	SC1MOD1	0x4002_00E8	SC1RST	0x4002_00F8	
0x4002_00C9	<R0>	0x4002_00D9	<R0>	0x4002_00E9	<R0>	0x4002_00F9	
0x4002_00CA	<R0>	0x4002_00DA	<R0>	0x4002_00EA	<R0>	0x4002_00FA	
0x4002_00CB	<R0>	0x4002_00DB	<R0>	0x4002_00EB	<R0>	0x4002_00FB	
0x4002_00CC	SC1MOD0	0x4002_00DC	SC1MOD2	0x4002_00EC	SC1TST	0x4002_00FC	
0x4002_00CD	<R0>	0x4002_00DD	<R0>	0x4002_00ED	<R0>	0x4002_00FD	
0x4002_00CE	<R0>	0x4002_00DE	<R0>	0x4002_00EE	<R0>	0x4002_00FE	
0x4002_00CF	<R0>	0x4002_00DF	<R0>	0x4002_00EF	<R0>	0x4002_00FF	

<SIO2>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4002_0100	SC2EN	0x4002_0110	SC2BRCR	0x4002_0120	SC2RFC	0x4002_0130	SC2FCNF
0x4002_0101	<R0>	0x4002_0111	<R0>	0x4002_0121	<R0>	0x4002_0131	<R0>
0x4002_0102	<R0>	0x4002_0112	<R0>	0x4002_0122	<R0>	0x4002_0132	<R0>
0x4002_0103	<R0>	0x4002_0113	<R0>	0x4002_0123	<R0>	0x4002_0133	<R0>
0x4002_0104	SC2BUF	0x4002_0114	SC2BRADD	0x4002_0124	SC2TFC	0x4002_0134	
0x4002_0105	<R0>	0x4002_0115	<R0>	0x4002_0125	<R0>	0x4002_0135	
0x4002_0106	<R0>	0x4002_0116	<R0>	0x4002_0126	<R0>	0x4002_0136	
0x4002_0107	<R0>	0x4002_0117	<R0>	0x4002_0127	<R0>	0x4002_0137	
0x4002_0108	SC2CR	0x4002_0118	SC2MOD1	0x4002_0128	SC2RST	0x4002_0138	
0x4002_0109	<R0>	0x4002_0119	<R0>	0x4002_0129	<R0>	0x4002_0139	
0x4002_010A	<R0>	0x4002_011A	<R0>	0x4002_012A	<R0>	0x4002_013A	
0x4002_010B	<R0>	0x4002_011B	<R0>	0x4002_012B	<R0>	0x4002_013B	
0x4002_010C	SC2MOD0	0x4002_011C	SC2MOD2	0x4002_012C	SC2TST	0x4002_013C	
0x4002_010D	<R0>	0x4002_011D	<R0>	0x4002_012D	<R0>	0x4002_013D	
0x4002_010E	<R0>	0x4002_011E	<R0>	0x4002_012E	<R0>	0x4002_013E	
0x4002_010F	<R0>	0x4002_011F	<R0>	0x4002_012F	<R0>	0x4002_013F	

[5] 10-bit A/D converter (A/DC)

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4003_0000	ADCLK	0x4003_0010	ADMOD3	0x4003_0020	ADCBAS	0x4003_0030	ADREG08L
0x4003_0001	<R0>	0x4003_0011	<R0>	0x4003_0021	<R0>	0x4003_0031	ADREG08H
0x4003_0002	<R0>	0x4003_0012	<R0>	0x4003_0022	<R0>	0x4003_0032	<R0>
0x4003_0003	<R0>	0x4003_0013	<R0>	0x4003_0023	<R0>	0x4003_0033	<R0>
0x4003_0004	ADMOD0	0x4003_0014	ADMOD4	0x4003_0024	<reserved>	0x4003_0034	ADREG19L
0x4003_0005	<R0>	0x4003_0015	<R0>	0x4003_0025	<reserved>	0x4003_0035	ADREG19H
0x4003_0006	<R0>	0x4003_0016	<R0>	0x4003_0026	<reserved>	0x4003_0036	<R0>
0x4003_0007	<R0>	0x4003_0017	<R0>	0x4003_0027	<reserved>	0x4003_0037	<R0>
0x4003_0008	ADMOD1	0x4003_0018	ADMOD5	0x4003_0028	<reserved>	0x4003_0038	ADREG2AL
0x4003_0009	<R0>	0x4003_0019	<R0>	0x4003_0029	<reserved>	0x4003_0039	ADREG2AH
0x4003_000A	<R0>	0x4003_001A	<R0>	0x4003_002A	<reserved>	0x4003_003A	<R0>
0x4003_000B	<R0>	0x4003_001B	<R0>	0x4003_002B	<reserved>	0x4003_003B	<R0>
0x4003_000C	ADMOD2	0x4003_001C		0x4003_002C	<reserved>	0x4003_003C	ADREG3BL
0x4003_000D	<R0>	0x4003_001D		0x4003_002D	<reserved>	0x4003_003D	ADREG3BH
0x4003_000E	<R0>	0x4003_001E		0x4003_002E	<reserved>	0x4003_003E	<R0>
0x4003_000F	<R0>	0x4003_001F		0x4003_002F	<reserved>	0x4003_003F	<R0>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4003_0040	ADREG4CL	0x4003_0050	ADREGSPL	0x4003_0060		0x4003_0070	
0x4003_0041	ADREG4CH	0x4003_0051	ADREGSPH	0x4003_0061		0x4003_0071	
0x4003_0042	<R0>	0x4003_0052	<R0>	0x4003_0062		0x4003_0072	
0x4003_0043	<R0>	0x4003_0053	<R0>	0x4003_0063		0x4003_0073	
0x4003_0044	ADREG5DL	0x4003_0054	ADCMP0L	0x4003_0064		0x4003_0074	
0x4003_0045	ADREG5DH	0x4003_0055	ADCMP0H	0x4003_0065		0x4003_0075	
0x4003_0046	<R0>	0x4003_0056	<R0>	0x4003_0066		0x4003_0076	
0x4003_0047	<R0>	0x4003_0057	<R0>	0x4003_0067		0x4003_0077	
0x4003_0048	ADREG6EL	0x4003_0058	ADCMP1L	0x4003_0068		0x4003_0078	
0x4003_0049	ADREG6EH	0x4003_0059	ADCMP1H	0x4003_0069		0x4003_0079	
0x4003_004A	<R0>	0x4003_005A	<R0>	0x4003_006A		0x4003_007A	
0x4003_004B	<R0>	0x4003_005B	<R0>	0x4003_006B		0x4003_007B	
0x4003_004C	ADREG7FL	0x4003_005C		0x4003_006C		0x4003_007C	
0x4003_004D	ADREG7FH	0x4003_005D		0x4003_006D		0x4003_007D	
0x4003_004E	<R0>	0x4003_005E		0x4003_006E		0x4003_007E	
0x4003_004F	<R0>	0x4003_005F		0x4003_006F		0x4003_007F	

[6] Watchdog timer (WDT)

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4004_0000	WDMOD	0x4004_0010		0x4004_0020		0x4004_0030	
0x4004_0001	<R0>	0x4004_0011		0x4004_0021		0x4004_0031	
0x4004_0002	<R0>	0x4004_0012		0x4004_0022		0x4004_0032	
0x4004_0003	<R0>	0x4004_0013		0x4004_0023		0x4004_0033	
0x4004_0004	WDCR	0x4004_0014		0x4004_0024		0x4004_0034	
0x4004_0005	<R0>	0x4004_0015		0x4004_0025		0x4004_0035	
0x4004_0006	<R0>	0x4004_0016		0x4004_0026		0x4004_0036	
0x4004_0007	<R0>	0x4004_0017		0x4004_0027		0x4004_0037	
0x4004_0008		0x4004_0018		0x4004_0028		0x4004_0038	
0x4004_0009		0x4004_0019		0x4004_0029		0x4004_0039	
0x4004_000A		0x4004_001A		0x4004_002A		0x4004_003A	
0x4004_000B		0x4004_001B		0x4004_002B		0x4004_003B	
0x4004_000C		0x4004_001C		0x4004_002C		0x4004_003C	
0x4004_000D		0x4004_001D		0x4004_002D		0x4004_003D	
0x4004_000E		0x4004_001E		0x4004_002E		0x4004_003E	
0x4004_000F		0x4004_001F		0x4004_002F		0x4004_003F	

[7] Real time clock (RTC)

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4004_0100	SECR	0x4004_0110		0x4004_0120		0x4004_0130	
0x4004_0101	MINR	0x4004_0111		0x4004_0121		0x4004_0131	
0x4004_0102	HOURR	0x4004_0112		0x4004_0122		0x4004_0132	
0x4004_0103	<R0>	0x4004_0113		0x4004_0123		0x4004_0133	
0x4004_0104	DAYR	0x4004_0114		0x4004_0124		0x4004_0134	
0x4004_0105	DATER	0x4004_0115		0x4004_0125		0x4004_0135	
0x4004_0106	MONTHR	0x4004_0116		0x4004_0126		0x4004_0136	
0x4004_0107	YEARR	0x4004_0117		0x4004_0127		0x4004_0137	
0x4004_0108	PAGER	0x4004_0118		0x4004_0128		0x4004_0138	
0x4004_0109	<R0>	0x4004_0119		0x4004_0129		0x4004_0139	
0x4004_010A	<R0>	0x4004_011A		0x4004_012A		0x4004_013A	
0x4004_010B	<R0>	0x4004_011B		0x4004_012B		0x4004_013B	
0x4004_010C	RESTR	0x4004_011C		0x4004_012C		0x4004_013C	
0x4004_010D	<reserved>	0x4004_011D		0x4004_012D		0x4004_013D	
0x4004_010E	<R0>	0x4004_011E		0x4004_012E		0x4004_013E	
0x4004_010F	<R0>	0x4004_011F		0x4004_012F		0x4004_013F	

[8] Clock generator (CG)

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4004_0200	SYSCR0	0x4004_0210	CKSEL	0x4004_0220	IMCGA0	0x4004_0230	<reserved>
0x4004_0201	SYSCR1	0x4004_0211	<R0>	0x4004_0221	IMCGA1	0x4004_0231	<reserved>
0x4004_0202	SYSCR2	0x4004_0212	<R0>	0x4004_0222	IMCGA2	0x4004_0232	<reserved>
0x4004_0203	<R0>	0x4004_0213	<R0>	0x4004_0223	IMCGA3	0x4004_0233	<reserved>
0x4004_0204	OSCCR0	0x4004_0214	ICRCG	0x4004_0224	IMCGB0	0x4004_0234	<reserved>
0x4004_0205	OSCCR1	0x4004_0215	<R0>	0x4004_0225	IMCGB1	0x4004_0235	<reserved>
0x4004_0206	<R0>	0x4004_0216	<R0>	0x4004_0226	IMCGB2	0x4004_0236	<reserved>
0x4004_0207	<R0>	0x4004_0217	<R0>	0x4004_0227	IMCGB3	0x4004_0237	<reserved>
0x4004_0208	STBYCR0	0x4004_0218	NMIFLG	0x4004_0228	IMCGC0	0x4004_0238	<reserved>
0x4004_0209	STBYCR1	0x4004_0219	<R0>	0x4004_0229	IMCGC1	0x4004_0239	<reserved>
0x4004_020A	STBYCR2	0x4004_021A	<R0>	0x4004_022A	IMCGC2	0x4004_023A	<reserved>
0x4004_020B	<R0>	0x4004_021B	<R0>	0x4004_022B	IMCGC3	0x4004_023B	<reserved>
0x4004_020C	PLLSEL	0x4004_021C	RSTFLG	0x4004_022C	IMCGD0	0x4004_023C	<reserved>
0x4004_020D	<R0>	0x4004_021D	<R0>	0x4004_022D	<reserved>	0x4004_023D	<reserved>
0x4004_020E	<R0>	0x4004_021E	<R0>	0x4004_022E	<reserved>	0x4004_023E	<reserved>
0x4004_020F	<R0>	0x4004_021F	<R0>	0x4004_022F	<reserved>	0x4004_023F	<reserved>

[9] CEC

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4004_0300	CECEN	0x4004_0310	CECRBUF	0x4004_0320	CECTEN	0x4004_0330	CECTSTAT
0x4004_0301	<R0>	0x4004_0311	“	0x4004_0321	<R0>	0x4004_0331	<R0>
0x4004_0302	<R0>	0x4004_0312	<R0>	0x4004_0322	<R0>	0x4004_0332	<R0>
0x4004_0303	<R0>	0x4004_0313	<R0>	0x4004_0323	<R0>	0x4004_0333	<R0>
0x4004_0304	CECADD	0x4004_0314	CECRCR1	0x4004_0324	CECTBUF	0x4004_0334	
0x4004_0305	“	0x4004_0315	“	0x4004_0325	“	0x4004_0335	
0x4004_0306	<R0>	0x4004_0316	“	0x4004_0326	<R0>	0x4004_0336	
0x4004_0307	<R0>	0x4004_0317	“	0x4004_0327	<R0>	0x4004_0337	
0x4004_0308	CECRESET	0x4004_0318	CECRCR2	0x4004_0328	CECTCR	0x4004_0338	
0x4004_0309	<R0>	0x4004_0319	“	0x4004_0329	“	0x4004_0339	
0x4004_030A	<R0>	0x4004_031A	<R0>	0x4004_032A	“	0x4004_033A	
0x4004_030B	<R0>	0x4004_031B	<R0>	0x4004_032B	<R0>	0x4004_033B	
0x4004_030C	CECREN	0x4004_031C	CECRCR3	0x4004_032C	CECRSTAT	0x4004_033C	
0x4004_030D	<R0>	0x4004_031D	“	0x4004_032D	<R0>	0x4004_033D	
0x4004_030E	<R0>	0x4004_031E	“	0x4004_032E	<R0>	0x4004_033E	
0x4004_030F	<R0>	0x4004_031F	<R0>	0x4004_032F	<R0>	0x4004_033F	

[10] Remote control signal preprocessor (RMC)

<RMC0>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4004_0400	RMC0EN	0x4004_0410	RMC0RBUF3	0x4004_0420	RMC0RCR4	0x4004_0430	
0x4004_0401	<R0>	0x4004_0411	<R0>	0x4004_0421	<R0>	0x4004_0431	
0x4004_0402	<R0>	0x4004_0412	<R0>	0x4004_0422	<R0>	0x4004_0432	
0x4004_0403	<R0>	0x4004_0413	<R0>	0x4004_0423	<R0>	0x4004_0433	
0x4004_0404	RMC0REN	0x4004_0414	RMC0RCR1	0x4004_0424	RMC0RSTAT	0x4004_0434	
0x4004_0405	<R0>	0x4004_0415	“	0x4004_0425	“	0x4004_0435	
0x4004_0406	<R0>	0x4004_0416	“	0x4004_0426	<R0>	0x4004_0436	
0x4004_0407	<R0>	0x4004_0417	“	0x4004_0427	<R0>	0x4004_0437	
0x4004_0408	RMC0RBUF1	0x4004_0418	RMC0RCR2	0x4004_0428		0x4004_0438	
0x4004_0409	“	0x4004_0419	“	0x4004_0429		0x4004_0439	
0x4004_040A	“	0x4004_041A	“	0x4004_042A		0x4004_043A	
0x4004_040B	“	0x4004_041B	“	0x4004_042B		0x4004_043B	
0x4004_040C	RMC0RBUF2	0x4004_041C	RMC0RCR3	0x4004_042C		0x4004_043C	
0x4004_040D	“	0x4004_041D	“	0x4004_042D		0x4004_043D	
0x4004_040E	“	0x4004_041E	<R0>	0x4004_042E		0x4004_043E	
0x4004_040F	“	0x4004_041F	<R0>	0x4004_042F		0x4004_043F	

<RMC1>

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4004_0440	RMC1EN	0x4004_0450	RMC1RBUF3	0x4004_0460	RMC1RCR4	0x4004_0470	
0x4004_0441	<R0>	0x4004_0451	<R0>	0x4004_0461	<R0>	0x4004_0471	
0x4004_0442	<R0>	0x4004_0452	<R0>	0x4004_0462	<R0>	0x4004_0472	
0x4004_0443	<R0>	0x4004_0453	<R0>	0x4004_0463	<R0>	0x4004_0473	
0x4004_0444	RMC1REN	0x4004_0454	RMC1RCR1	0x4004_0464	RMC1RSTAT	0x4004_0474	
0x4004_0445	<R0>	0x4004_0455	“	0x4004_0465	“	0x4004_0475	
0x4004_0446	<R0>	0x4004_0456	“	0x4004_0466	<R0>	0x4004_0476	
0x4004_0447	<R0>	0x4004_0457	“	0x4004_0467	<R0>	0x4004_0477	
0x4004_0448	RMC1RBUF1	0x4004_0458	RMC1RCR2	0x4004_0468		0x4004_0478	
0x4004_0449	“	0x4004_0459	“	0x4004_0469		0x4004_0479	
0x4004_044A	“	0x4004_045A	“	0x4004_046A		0x4004_047A	
0x4004_044B	“	0x4004_045B	“	0x4004_046B		0x4004_047B	
0x4004_044C	RMC1RBUF2	0x4004_045C	RMC1RCR3	0x4004_046C		0x4004_047C	
0x4004_044D	“	0x4004_045D	“	0x4004_046D		0x4004_047D	
0x4004_044E	“	0x4004_045E	<R0>	0x4004_046E		0x4004_047E	
0x4004_044F	“	0x4004_045F	<R0>	0x4004_046F		0x4004_047F	

[11]Flash

ADR	Register name	ADR	Register name	ADR	Register name	ADR	Register name
0x4004_0500	SECBIT	0x4004_0510		0x4004_0520	FLCS	0x4004_0530	
0x4004_0501	“	0x4004_0511		0x4004_0521	“	0x4004_0531	
0x4004_0502	“	0x4004_0512		0x4004_0522	“	0x4004_0532	
0x4004_0503	“	0x4004_0513		0x4004_0523	“	0x4004_0533	
0x4004_0504	<reserved>	0x4004_0514		0x4004_0524	<reserved>	0x4004_0534	
0x4004_0505	<reserved>	0x4004_0515		0x4004_0525	<reserved>	0x4004_0535	
0x4004_0506	<reserved>	0x4004_0516		0x4004_0526	<reserved>	0x4004_0536	
0x4004_0507	<reserved>	0x4004_0517		0x4004_0527	<reserved>	0x4004_0537	
0x4004_0508		0x4004_0518		0x4004_0528	<reserved>	0x4004_0538	
0x4004_0509		0x4004_0519		0x4004_0529	<reserved>	0x4004_0539	
0x4004_050A		0x4004_051A		0x4004_052A	<reserved>	0x4004_053A	
0x4004_050B		0x4004_051B		0x4004_052B	<reserved>	0x4004_053B	
0x4004_050C		0x4004_051C		0x4004_052C		0x4004_053C	
0x4004_050D		0x4004_051D		0x4004_052D		0x4004_053D	
0x4004_050E		0x4004_051E		0x4004_052E		0x4004_053E	
0x4004_050F		0x4004_051F		0x4004_052F		0x4004_053F	

[12] Reserved Area

ADR	Register name
0x4004_0540	<reserved>
0x4004_0541	<reserved>
0x4004_0542	<reserved>
0x4004_0543	<reserved>
0x4004_0544	<reserved>
0x4004_0545	<reserved>
0x4004_0546	<reserved>
0x4004_0547	<reserved>
0x4004_0548	
0x4004_0549	
0x4004_054A	
0x4004_054B	
0x4004_054C	
0x4004_054D	
0x4004_054E	
0x4004_054F	

ADR	Register name
0x4004_0550	<reserved>
0x4004_0551	<reserved>
0x4004_0552	<reserved>
0x4004_0553	<reserved>
0x4004_0554	
0x4004_0555	
0x4004_0556	
0x4004_0557	
0x4004_0558	
0x4004_0559	
0x4004_055A	
0x4004_055B	
0x4004_055C	
0x4004_055D	
0x4004_055E	
0x4004_055F	

ADR	Register name
0x4004_0560	<reserved>
0x4004_0561	<reserved>
0x4004_0562	<reserved>
0x4004_0563	<reserved>
0x4004_0564	<reserved>
0x4004_0565	<reserved>
0x4004_0566	<reserved>
0x4004_0567	<reserved>
0x4004_0568	<reserved>
0x4004_0569	<reserved>
0x4004_056A	<reserved>
0x4004_056B	<reserved>
0x4004_056C	<reserved>
0x4004_056D	<reserved>
0x4004_056E	<reserved>
0x4004_056F	<reserved>

ADR	Register name
0x4004_0570	<reserved>
0x4004_0571	<reserved>
0x4004_0572	<reserved>
0x4004_0573	<reserved>
0x4004_0574	<reserved>
0x4004_0575	<reserved>
0x4004_0576	<reserved>
0x4004_0577	<reserved>
0x4004_0578	<reserved>
0x4004_0579	<reserved>
0x4004_057A	<reserved>
0x4004_057B	<reserved>
0x4004_057C	<reserved>
0x4004_057D	<reserved>
0x4004_057E	<reserved>
0x4004_057F	<reserved>

ADR	Register name
0x4004_0580	<reserved>
0x4004_0581	<reserved>
0x4004_0582	<reserved>
0x4004_0583	<reserved>
0x4004_0584	<reserved>
0x4004_0585	<reserved>
0x4004_0586	<reserved>
0x4004_0587	<reserved>
0x4004_0588	<reserved>
0x4004_0589	<reserved>
0x4004_058A	<reserved>
0x4004_058B	<reserved>
0x4004_058C	<reserved>
0x4004_058D	<reserved>
0x4004_058E	<reserved>
0x4004_058F	<reserved>

ADR	Register name
0x4004_0590	<reserved>
0x4004_0591	<reserved>
0x4004_0592	<reserved>
0x4004_0593	<reserved>
0x4004_0594	
0x4004_0595	
0x4004_0596	
0x4004_0597	
0x4004_0598	
0x4004_0599	
0x4004_059A	
0x4004_059B	
0x4004_059C	
0x4004_059D	
0x4004_059E	
0x4004_059F	

ADR	Register name
0x4004_05A0	
0x4004_05A1	
0x4004_05A2	
0x4004_05A3	
0x4004_05A4	
0x4004_05A5	
0x4004_05A6	
0x4004_05A7	
0x4004_05A8	
0x4004_05A9	
0x4004_05AA	
0x4004_05AB	
0x4004_05AC	
0x4004_05AD	
0x4004_05AE	
0x4004_05AF	

ADR	Register name
0x4004_05B0	
0x4004_05B1	
0x4004_05B2	
0x4004_05B3	
0x4004_05B4	
0x4004_05B5	
0x4004_05B6	
0x4004_05B7	
0x4004_05B8	
0x4004_05B9	
0x4004_05BA	
0x4004_05BB	
0x4004_05BC	
0x4004_05BD	
0x4004_05BE	
0x4004_05BF	

ADR	Register name
0x4004_0700	<reserved>
0x4004_0701	<reserved>
0x4004_0702	<reserved>
0x4004_0703	<reserved>
0x4004_0704	<reserved>
0x4004_0705	<reserved>
0x4004_0706	<reserved>
0x4004_0707	<reserved>
0x4004_0708	
0x4004_0709	
0x4004_070A	
0x4004_070B	
0x4004_070C	
0x4004_070D	
0x4004_070E	
0x4004_070F	

ADR	Register name
0x4004_0710	
0x4004_0711	
0x4004_0712	
0x4004_0713	
0x4004_0714	
0x4004_0715	
0x4004_0716	
0x4004_0717	
0x4004_0718	
0x4004_0719	
0x4004_071A	
0x4004_071B	
0x4004_071C	
0x4004_071D	
0x4004_071E	
0x4004_071F	

ADR	Register name
0x4004_0720	
0x4004_0721	
0x4004_0722	
0x4004_0723	
0x4004_0724	
0x4004_0725	
0x4004_0726	
0x4004_0727	
0x4004_0728	
0x4004_0729	
0x4004_072A	
0x4004_072B	
0x4004_072C	
0x4004_072D	
0x4004_072E	
0x4004_072F	

ADR	Register name
0x4004_0730	
0x4004_0731	
0x4004_0732	
0x4004_0733	
0x4004_0734	
0x4004_0735	
0x4004_0736	
0x4004_0737	
0x4004_0738	
0x4004_0739	
0x4004_073A	
0x4004_073B	
0x4004_073C	
0x4004_073D	
0x4004_073E	
0x4004_073F	

21 Electrical Characteristics

21.1 Absolute Maximum Ratings

Parameter		Symbol	Rating	Unit
Supply voltage		DVCC (I/O)	- 0.3~3.9	V
		AVCC (A/D)	- 0.3~3.9	
		CVCC (CLK)	- 0.3~3.9	
		REGVCC	- 0.3~3.9	
Input voltage		V_{IN}	- 0.3~V_{CC} + 0.3	V
Low-level output current	Per pin	I_{OL}	5	mA
	Total	ΣI_{OL}	50	
High-level output current	Per pin	I_{OH}	- 5	
	Total	ΣI_{OH}	50	
Power consumption (Ta = 85°C)		PD	600	mW
Soldering temperature (10s)		T_{SOLDER}	260	°C
Storage temperature		T_{STG}	- 40~125	°C
Operating Temperature	Except during Flash W/E	T_{OPR}	- 20 ~ 85	°C
	During Flash W/E		0 ~ 70	
Write/erase cycles		N_{EW}	100	cycle

(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.

21.2 DC Electrical Characteristics (1/2)

Ta＝-20~85°C

Parameter		Symbol	Rating		Min.	Typ. (Note 1)	Max.	Unit
Supply voltage	DVCC = AVCC = CVCC = REGVCC (Note 2) DVSS = AVSS = 0V	DVCC AVCC CVCC REGVCC	fosc = 8 ~ 10MHz fsys = 1 ~ 40MHz Fs = 30 ~ 34KHz		2.7	-	3.6	V
Low-level input voltage	PC, PD	V_{IL1}	2.7V ≤ AVCC ≤ 3.6V		-0.3	-	0.3 AVCC	V
	Normal port	V_{IL2}	2.7V ≤ DVCC ≤ 3.6V				0.3 DVCC	
	Schmitt-Triggered port	V_{IL3}					0.2 DVCC	
	X1	V_{IL4}	2.7V ≤ CVCC ≤ 3.6V				0.1 CVCC	
High-level input voltage	PC, PD	V_{IH1}	2.7V ≤ AVCC ≤ 3.6V		0.7 AVCC	-	AVCC+0.3	V
	Normal port	V_{IH2}	2.7V ≤ DVCC ≤ 3.6V		0.7 DVCC		DVCC+0.3	
	Schmitt-Triggered port	V_{IH3}			0.8 DVCC			
	X1	V_{IH4}	2.7V ≤ CVCC ≤ 3.6V		0.9 CVCC		CVCC+0.3	
Low-level output voltage		V_{OL}	I_{OL} = 2mA	DVCC≥2.7V	-	-	0.4	V
High-level output voltage		V_{OH}	I_{OH} = -2mA	DVCC≥2.7V	2.4	-	-	V
Input leakage current	Except CEC	I_{LI1}	0.0 ≤ V_{IN} ≤DVCC 0.0 ≤ V_{IN} ≤AVCC		-	0.02	± 5	μA
	CEC	I_{LI2}	0.0 ≤ V_{IN} ≤3.6					
Output leakage current	Except CEC	I_{LO1}	0.2 ≤ V_{IN} ≤ DVCC - 0.2 0.2 ≤ V_{IN} ≤ AVCC - 0.2		-	0.05	± 10	
	CEC	I_{LO2}	0.2 ≤ V_{IN} ≤3.4					
Pull-up resister at Reset		RRST	DVCC = 2.7V~3.6V		-	50	150	kΩ
Schmitt-Triggered port		VTH	2.7V ≤ DVCC ≤ 3.6V		0.3	0.6	-	V
Programmable pull-up/ pull-down resistor		PKH	DVCC = 2.7V~3.6V		-	50	150	kΩ
Pin capacitance (Except power supply pins)		C_{IO}	fc = 1MHz		-	-	10	pF

(Note 1) Ta=25℃, DVCC = REGVCC = AVCC = 3.3V, unless otherwise noted.

(Note 2) The same voltage must be supplied to DVCC, AVCC, CVCC and REGVCC.

21.3 DC Electrical Characteristics (2/2)

(1) TMPM330FDFG/ TMPM330FYFG

DVCC = AVCC = CVCC = REGVCC = 2.7V~3.6V, Ta = -20~85°C

Parameter	Symbol	Rating	Min.	Typ. (Note 1)	Max.	Unit
NORMAL (Note 2) Gear 1/1	I_{CC}	fsys = 40 MHz (fosc = 10 MHz)	-	32	42	mA
IDLE (Note 3)			-	8	13	
SLOW		fs = 32.768kHz	-	2.5	6	
SLEEP			-	55	950	μA
STOP		-	-	45	900	

(Note 1) Ta=25℃, DVCC = AVCC = CVCC = REGVCC = 3.3V, unless otherwise noted.

(Note 2) ICC NORMAL: Measured with the dhrystone ver. 2.1 operated in FLASH. All functions operats excluding A/D.

(Note 3) ICC IDLE: Measured with all functions stopped. The currents flow through DVCC, AVCC, CVCC15 and REGVCC are included.

(2) TMPM330FWFG

DVCC = AVCC = CVCC = REGVCC = 2.7V~3.6V, Ta = -20~85°C

Parameter	Symbol	Rating	Min.	Typ. (Note 1)	Max.	Unit
NORMAL (Note 2) Gear 1/1	I_{CC}	fsys = 40 MHz (fosc = 10 MHz)	-	26	32	mA
IDLE (Note 3)			-	8	12.5	
SLOW		fs = 32.768kHz	-	2.5	5.5	
SLEEP			-	55	900	μA
STOP		-	-	45	850	

(Note 1) Ta=25℃, DVCC = AVCC = CVCC = REGVCC = 3.3V, unless otherwise noted.

(Note 2) ICC NORMAL: Measured with the dhrystone ver. 2.1 operated in FLASH. All functions operats excluding A/D.

(Note 3) ICC IDLE: Measured with all functions stopped. The currents flow through DVCC, AVCC, CVCC15 and REGVCC are included.

21.4 10-bit ADC Electrical Characteristics

DVCC = AVCC = CVCC = REGVCC = VREFH = 2.7V~3.6V,
AVSS = DVSS, Ta＝－20~85℃
AVCC load capacitance ≥3.3μF, VREFH load capacitance ≥3.3μF

Parameter		Symbol	Rating	Min	Typ	Max	Unit
Analog reference voltage (+)		VREFH	-	2.7	3.3	3.6	V
Analog input voltage		VAIN	-	AVSS	-	VREFH	V
Analog supply current	A/D conversion	IREF	DVSS = AVSS	-	2.5	5.5	mA
	Non-A/D conversion			-	± 0.02	± 5	μA
Supply current	A/D conversion	-	Non-IREF	-	-	3	mA
INL error		-	AIN resistance ≤600Ω AIN load capacitance ≤30pF Conversion time ≥1.15μs	-	± 2	± 3	LSB
DNL error				-	± 1	± 2	
Offset error				-	± 2	± 4	
Full-scale error				-	± 2	± 4	
INL error		-	AIN resistance ≤600Ω AIN load capacitance ≤30pF Conversion time ≥1.15μs	-	± 2	± 3	
DNL error				-	± 1	± 2	
Offset error				-	± 2	± 4	
Full-scale error				-	± 2	± 4	
INL error		-	AIN resistance ≤1.3kΩ AIN load capacitance ≥0.1μF Conversion time ≥1.15μs	-	± 2	± 3	
DNL error				-	± 1	± 2	
Offset error				-	± 2	± 4	
Full-scale error				-	± 2	± 4	
INL error		-	AIN resistance ≤10kΩ AIN load capacitance ≥0.1μF Conversion time ≥2.30μs	-	± 2	± 3	
DNL error				-	± 1	± 2	
Offset error				-	± 2	± 4	
Full-scale error				-	± 2	± 4	

(Note) 1LSB = (VREFH - AVSS) / 1024[V]

21.5 AC Electrical Characteristics

21.5.1 Serial Channel Timing (SIO)

(1) I/O Interface mode

In the table below, the letter x represents the SIO operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

1) SCLK input mode

Parameter	Symbol	Equation		40 MHz		Unit
		Min	Max	Min	Max	
SCLK Clock High width (input)	t_{SCH}	3x	-	75	-	ns
SCLK Clock Low width (input)	t_{SCL}	3x	-	75	-	
SCLK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	-	150	-	
TxD to SCLK rise or fall (Note 1)	t_{OSS}	$t_{SCY}/2 – 2x - 45$	-	-20 (Note 2)	-	
TxD hold or fall after SCLK rising (Note 1)	t_{OHS}	$t_{SCY}/2$	-	75	-	
RxD valid to SCLK rise or fall (Note 1)	t_{SRD}	30	-	30	-	
RxD hold or fall after SCLK rising (Note 1)	t_{HSR}	x + 30	-	55	-	

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.

(Note 1) SCLK rise or fall: Measured relative to the programmed active edge of SCLK.

(Note 2) Keep this value positive by adjusting SCLK cycle.

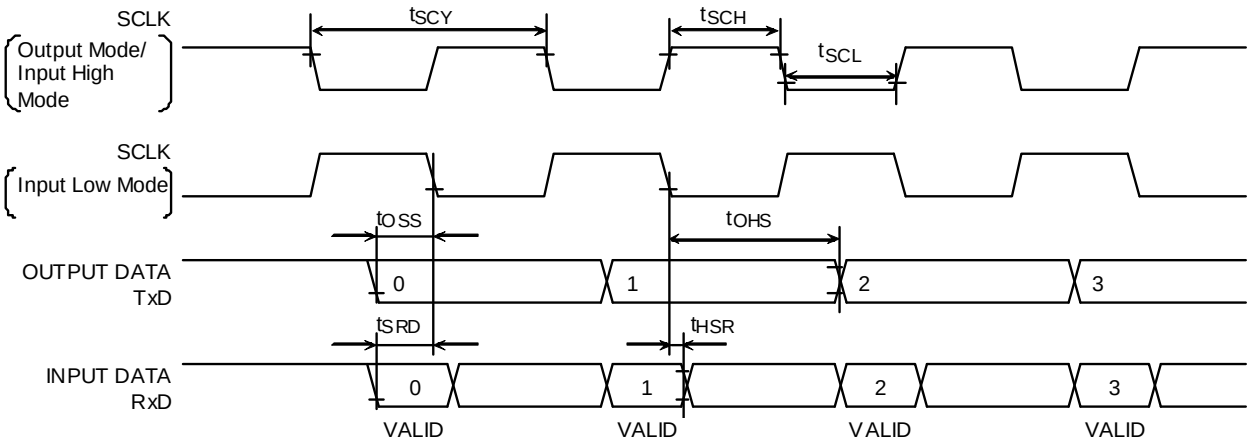
2) SCLK output mode

Parameter	Symbol	Equation		40 MHz		Unit
		Min	Max	Min	Max	
SCLK cycle (programmable)	t_{SCY}	4x	-	100	-	ns
TxD to SCLK rise	t_{OSS}	t_{SCY} /2 - 20	-	30	-	
TxD hold after SCLK rising	t_{OHS}	t_{SCY} /2 - 20	-	30	-	
RxD valid to SCLK rise	t_{SRD}	45	-	45	-	
RxD hold after SCLK rising	t_{HSR}	0	-	0	-	

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.



21.5.2 SBI（I2C）

(1) I2C Mode

In the table below, the letter x represents the I2C operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

n denotes the value of n programmed into the SCK (SCL output frequency select) field in the SBInCR.

Parameter	Symbol	Equation		Standard Mode		Fast Mode		Unit
		Min	Max	Min	Max	Min	Max	
SCL clock frequency	t_{SCL}	0	-	0	100	0	400	kHz
Hold time for START condition	$t_{HD:STA}$	-	-	4.0	-	0.6	-	μs
SCL low width (Input) (Note 1)	t_{LOW}	-	-	4.7	-	1.3	-	μs
SCL high width (Input) (Note 2)	t_{HIGH}	-	-	4.0	-	0.6	-	μs
Setup time for a repeated START condition	$t_{SU;STA}$	(Note 5)	-	4.7	-	0.6	-	μs
Data hold time (Input) (Note 3, 4)	$t_{HD;DAT}$	-	-	0.0	-	0.0	-	μs
Data setup time	$t_{SU;DAT}$	-	-	250	-	100	-	ns
Setup time for a stop condition	$t_{SU;STO}$	-	-	4.0	-	0.6	-	μs
Bus free time between stop condition and start condition	t_{BUF}	(Note 5)	-	4.7	-	1.3	-	μs

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.

(Note 1) SCL clock low width (output) is calculated with: $(2^{n-1}+58)/x$

(Note 2) SCL clock high width (output) is calculated with: $(2^{n-1}+12)/x$

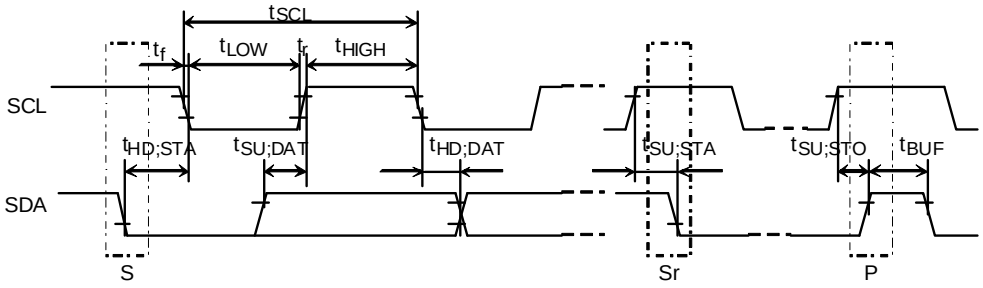
Notice: On I2C-bus specification, Maximum Speed of Standard Mode is 100KHz, Fast mode is 400Khz. Internal SCL Frequency setting should comply with Note1 & Note2 shown above.

(Note 3) The output data hold time is equal to 12x of internal SCL.

(Note 4) The Philips I2C-bus specification states that a device must internally provide a hold time of at least 300 ns for the SDA signal to bridge the undefined region of the falling edge of SCL. However, this SBI does not satisfy this requirement. Also, the output buffer for SCL does not incorporate slope control of the falling edges; therefore, the equipment manufacturer should design so that the input data hold time shown in the table is satisfied, including tr/tf of the SCL and SDA lines.

(Note 5) Software-dependent.

(Note 6) The Philips I2C-bus specification instructs that if the power supply to a Fast-mode device is switched off, the SDA and SCL I/O pins must be floating so that they don't obstruct the bus lines. However, this SBI does not satisfy this requirement.



S: Start condition
Sr: Repeated start condition
P: Stop condition

(2) Clock-Synchronous 8-Bit SIO mode

In the table below, the letter x represents the I2C operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

The electrical specifications below are for an SCK signal with a 50% duty cycle.

1) SCK Input Mode

Parameter	Symbol	Equation		40MHz		Unit
		Min	Max	Min	Max	
SCK Clock High width (input)	t_{SCH}	4x	-	100	-	ns
SCK Clock Low width (input)	t_{SCL}	4x	-	100	-	
SCK cycle	t_{SCY}	$t_{SCH} + t_{SCL}$	-	200	-	
TxD to SCK rise	t_{OSS}	$t_{SCY}/2 - 3x - 45$	-	-20 (Note)	-	
TxD hold after SCK rising	t_{OHS}	$t_{SCY}/2 + x$	-	125	-	
RxD valid to SCK rise	t_{SRD}	30 - x	-	5	-	
RxD hold after SCK rising	t_{HSR}	30	-	30	-	

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.

(Note) Keep this value positive by adjusting SCK cycle.

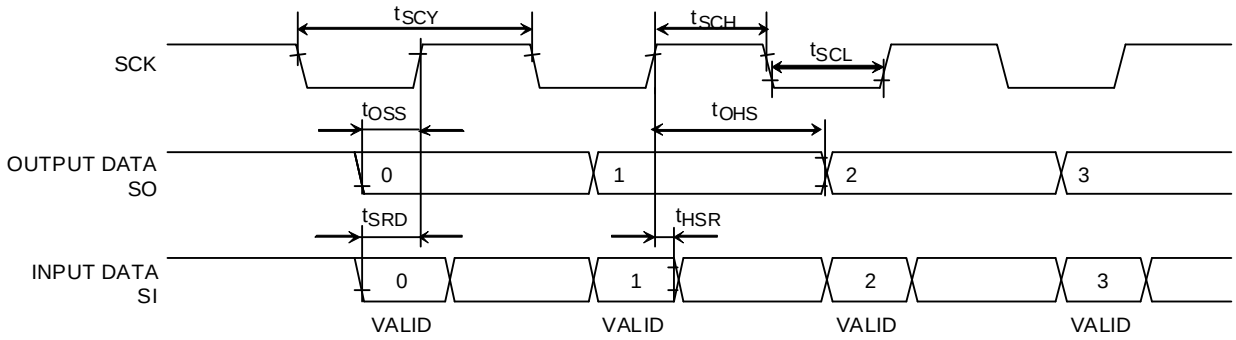
2) SCK Output Mode

Parameter	Symbol	Equation		40MHz		Unit
		Min	Max	Min	Max	
SCK cycle (programmable)	t_{SCY}	16x	-	400	-	ns
TxD to SCK rise	t_{OSS}	$t_{SCY}/2 – 20$	-	180	-	
TxD hold after SCK rising	t_{OHS}	$t_{SCY}/2 – 20$	-	180	-	
RxD valid to SCK rise	t_{SRD}	45	-	45	-	
RxD data hold after SCK rising	t_{HSR}	0	-	0	-	

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.


tSCY
tSCH
tSCL
SCK
tOSS
tOHS
OUTPUT DATA
SO
0
1
2
3
tSRD
tHSR
INPUT DATA
SI
0
1
2
3
VALID
VALID
VALID
VALID

21.5.3 Event Counter

In the table below, the letter x represents the TMRB operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		40MHz		Unit
		Min	Max	Min	Max	
Clock low pulse width	t_{VCKL}	2x + 100	-	150	-	ns
Clock high pulse width	t_{VCKH}	2x + 100	-	150	-	ns

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.

21.5.4 Capture

In the table below, the letter x represents the TMRB operation clock cycle time which is identical to the fsys cycle time. It varies depending on the programming of the clock gear function.

Parameter	Symbol	Equation		40MHz		Unit
		Min	Max	Min	Max	
Low pulse width	t_{CPL}	2x + 100	-	150	-	ns
High pulse width	t_{CPH}	2x + 100	-	150	-	ns

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.

21.5.5 General Interrupts

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

Parameter	Symbol	Equation		40MHz		Unit
		Min	Max	Min	Max	
Low pulse width for INT0~7	t_{CPL}	x + 100	-	125	-	ns
High pulse width for INT0~7	t_{CPH}	x + 100	-	125	-	ns

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.

21.5.6 $\overline{\text{NMI}}$, STOP Release Interrupts

Parameter	Symbol	Min	Max	Unit
Low pulse width for $\overline{\text{NMI}}$ and INT0~7	t_{INTBL}	100	-	ns
High pulse width for INT0~7	t_{INTBH}	100	-	ns

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.

21.5.7 SCOUT Pin AC Characteristic

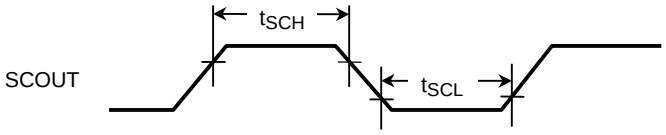
Parameter	Symbol	Equation		40MHz		Unit
		Min	Max	Min	Max	
High pulse width	t_{SCH}	0.5T - 5		7.5		ns
Low pulse width	t_{SCL}	0.5T - 5		7.5		ns

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.

(Note) In the above table, the letter T represents the cycle time of the SCOUT output clock.



21.5.8 Debug Communication

(1) SWD Interface

Parameter	Symbol	Min	Max	Unit
CLK cycle	T_{dck}	83.33	-	ns
DATA to CLK rising	T_{dos}	12	-	ns
DATA hold after CLK rising	T_{doh}	4	-	ns
DATA hold after CLK falling	T_{dih}	-	6	ns

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.

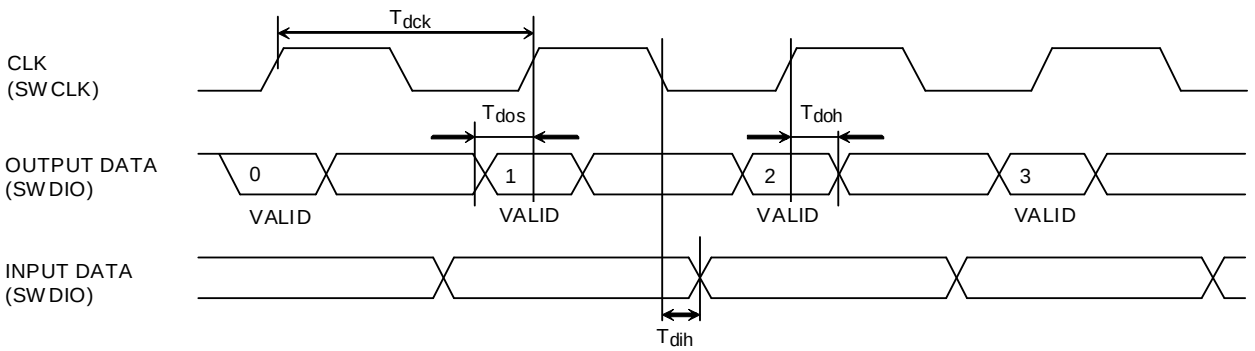
(2) JTAG Interface

Parameter	Symbol	Min	Max	Unit
CLK cycle	T_{dck}	100	-	ns
DATA to CLK rising	T_{dos}	3	-	ns
DATA hold after CLK rising	T_{doh}	25	-	ns
DATA hold after CLK falling	T_{dih}	-	15	ns

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.



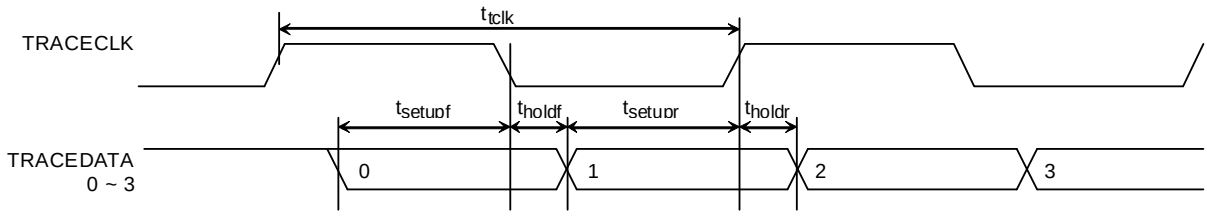
21.5.9 TRACE Output

Parameter	Symbol	Min	Max	Unit
TRACECLK cycle	T_{dclk}	50	-	ns
TRACEDATA valid to DCLK rise	t_{setupr}	1.8	-	ns
TRACEDATA hold after DCLK rising	t_{holdr}	0.5	-	ns
TRACEDATA valid to DCLK fall	t_{setupf}	1.8	-	ns
TRACEDATA hold after DCLK falling	t_{holdf}	0.5	-	ns

AC measurement condition

/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: Refer to low-level input voltage and high-level input voltage in 21.2 DC Electrical Characteristics.



21.6 Recommended Oscillation Circuit

The TMPM330 has been evaluated by the oscillator vender below. Use this information when selecting external parts.

(Note)The load value of the oscillator is the sum of loads (C1 and C2) and the floating load of the actual assembled board. There is a possibility of operating error when using C1 and C2 values in the table below. When designing the board, design the minimum length pattern around the oscillator. We also recommend that oscillator evaluation be carried out using the actual board.

(1) Connection example

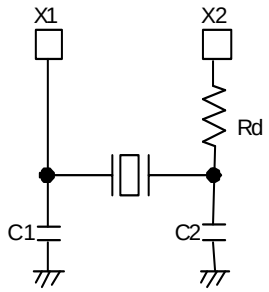


Fig. 21-1 High-frequency oscillation connection

(2) Recommended ceramic oscillator

The TMPM330 recommends the high-frequency oscillator by Murata Manufacturing Co., Ltd.
Please refer to the following URL for details.
http://www.murata.co.jp

(3) Recommended ceramic oscillator

The TMPM330 recommends the high-frequency oscillator by KYOCERA KINSEKI Corporation.
Please refer to the following URL for details.
http://www.kinseki.co.jp

21.7 Handling Precaution

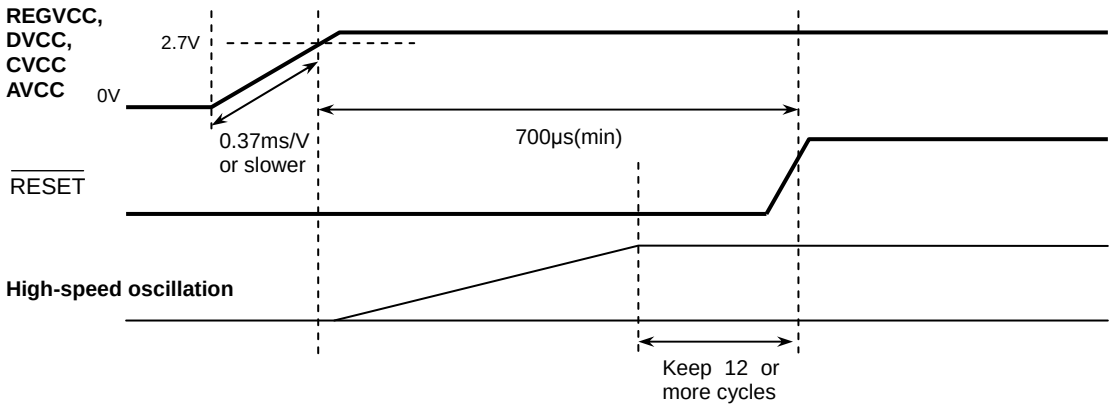
21.7.1 Solderability of lead free products

Test parameter	Test condition	Note
Solderability	Use of Sn-37Pb solder Bath Solder bath temperature = 230°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	Pass: solderability rate until forming ≥ 95%
	Use of Sn-3.0Ag-0.5Cu solder bath Solder bath temperature = 245°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux (use of lead free)	

21.7.2 Power-on sequence

The power supply must be raised (from 0V to 2.7V) at a speed of 0.37ms/V or slower.

The power-on sequence must include the time for the internal regulator and oscillator to be stable. In the TMPM330, the internal regulator requires at least 700 μs to be stable. The time required to achieve stable oscillation varies with system. At cold reset, the external reset pin must be kept low for a duration of time sufficiently long enough for the internal regulator and oscillator to be stable.



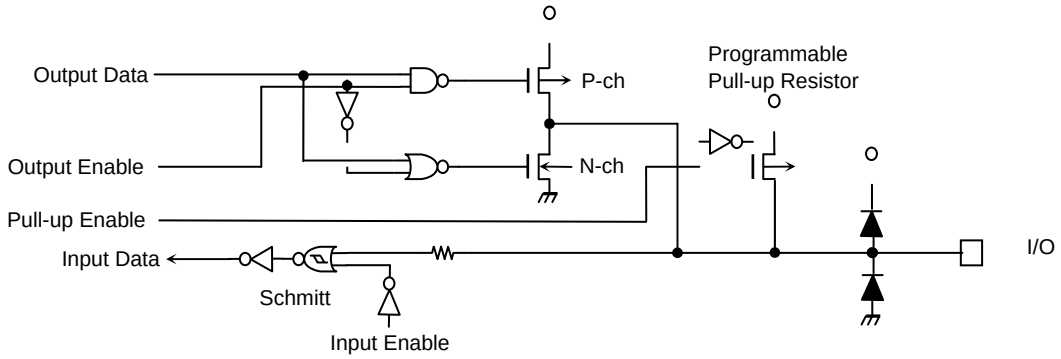
22 Port Section Equivalent Circuit Schematic

- Reading the schematics

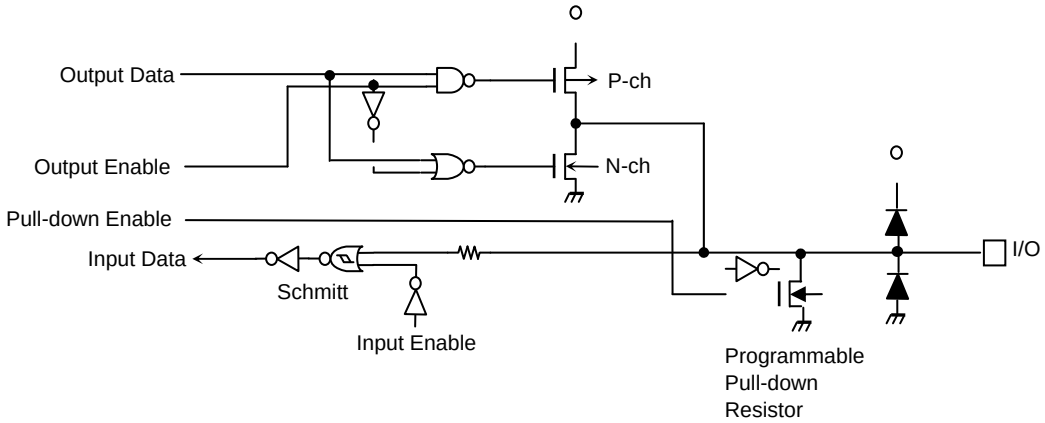
Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

The input protection resistance ranges from several tens of ohms to several hundreds of ohms. Damping resistors X2 and XT2 are shown with a typical value.

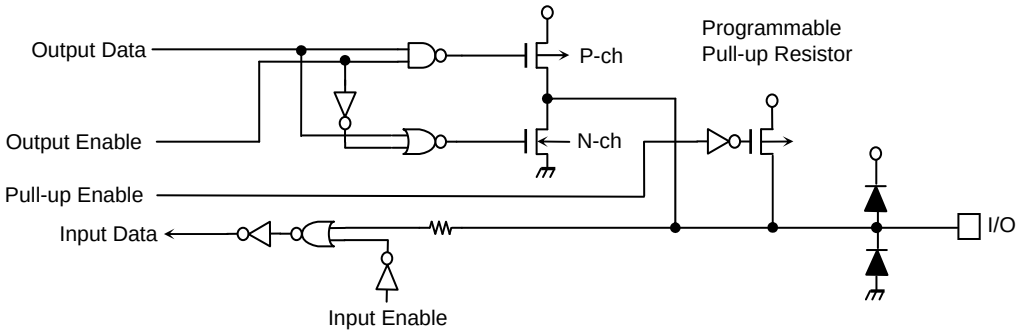
- PA0, PB1-2, PE1-3, PE5-6, PF1-7, PG0-6, PH0-7, PI6-7, PJ0-3, PJ6-7



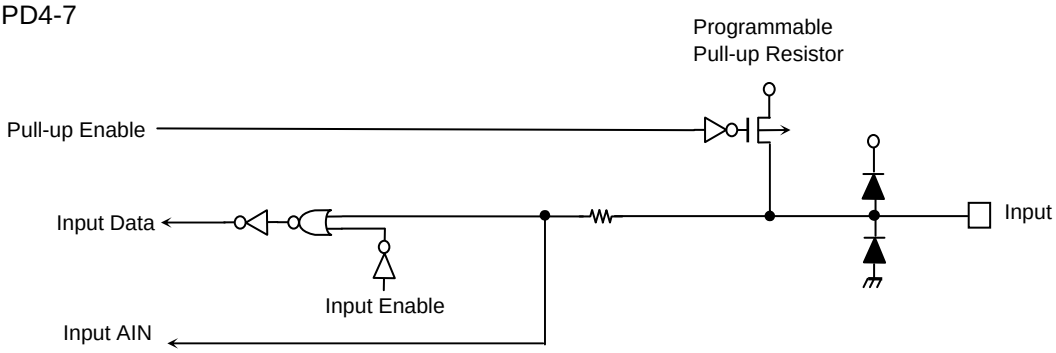
- PA1



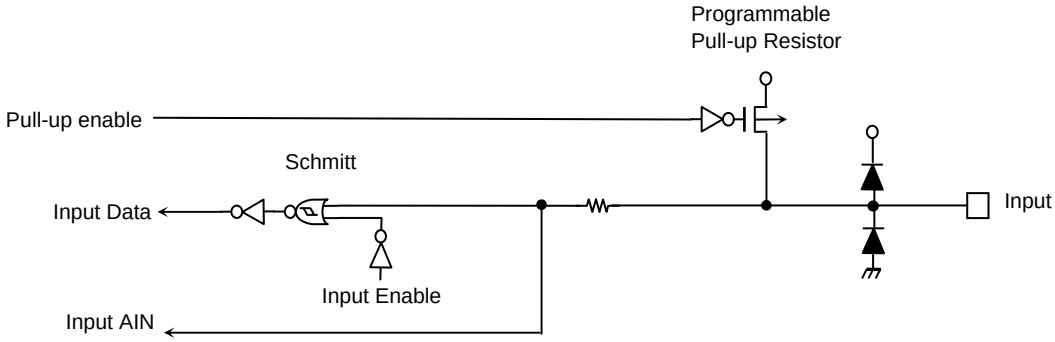
- PA2-7, PB0, PB3-7, PE0, PE4, PF0, PG7, PI0-5, PJ4-5, PK1-2



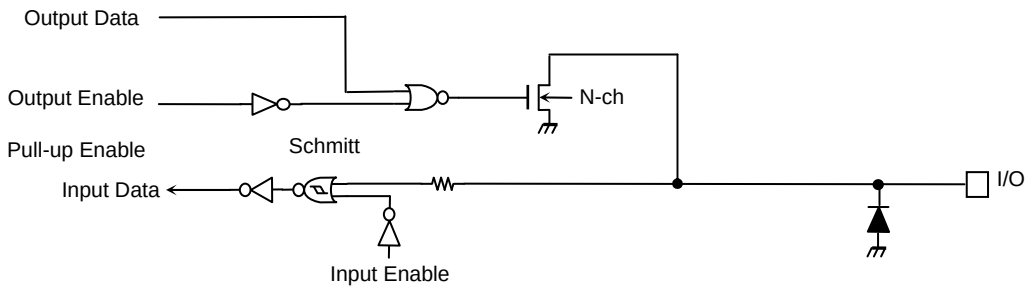
- PC0-3, PD4-7



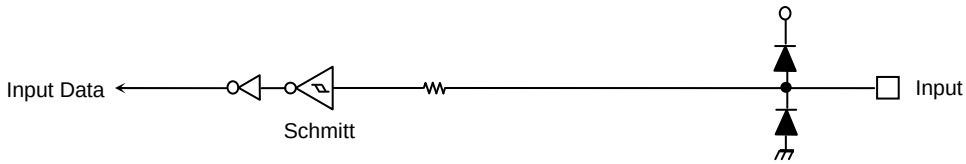
- PD0-3



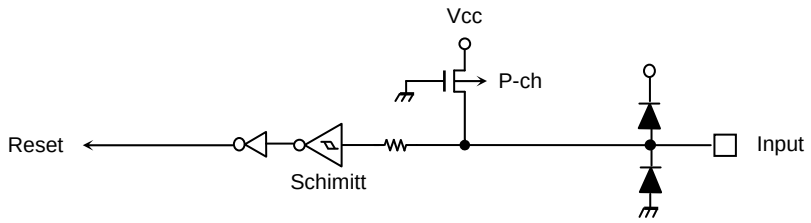
- PK0



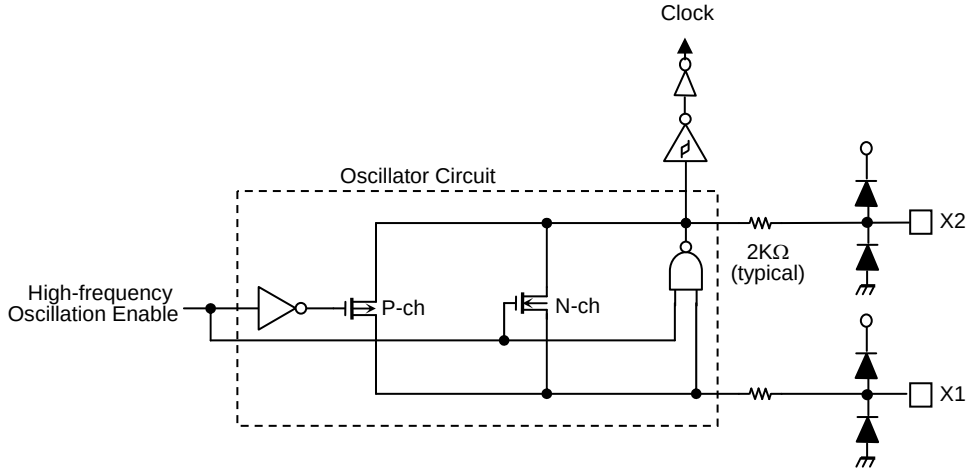
- $\overline{\text{NMI}}$, MODE



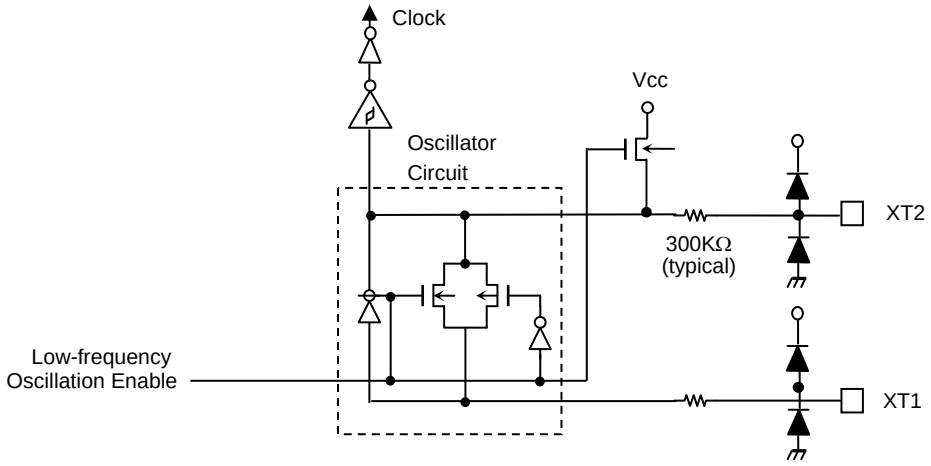
- $\overline{\text{RESET}}$



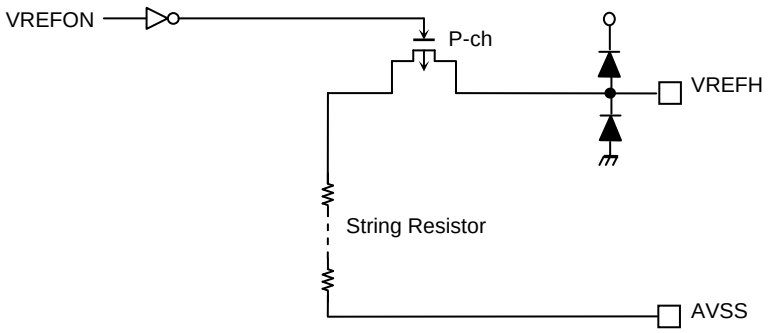
- X1, X2



- XT1, XT2



- VREFH, AVSS

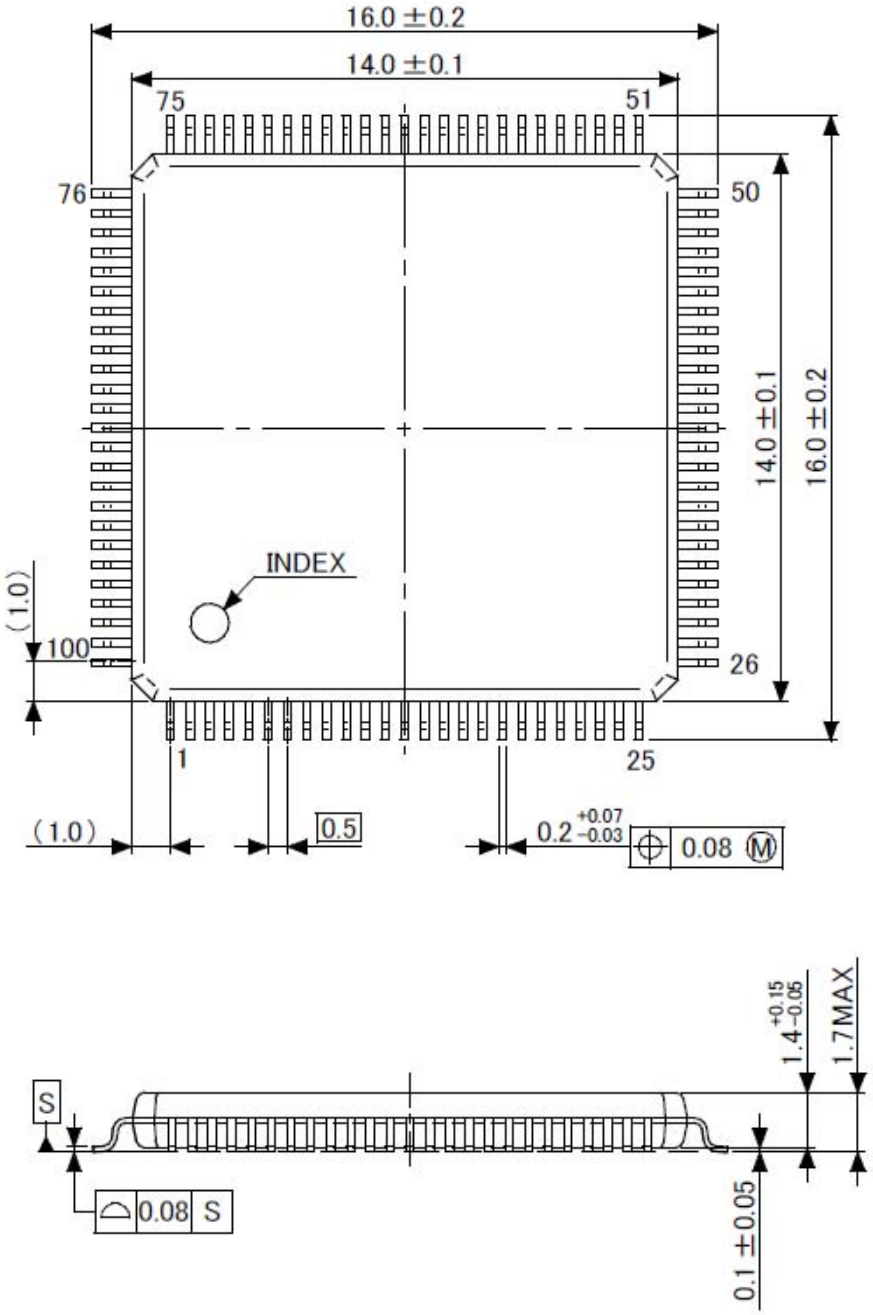


23 Package

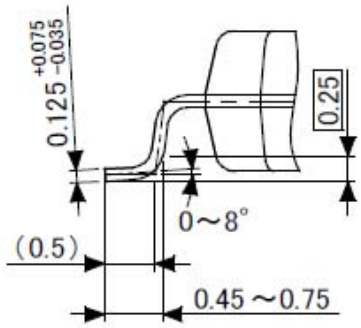
Type: LQFP100-P-1414-0.50H

Dimensions

Unit : mm



Pin detail



RESTRICTIONS ON PRODUCT USE

- Toshiba Corporation, and its subsidiaries and affiliates (collectively “TOSHIBA”), reserve the right to make changes to the information in this document, and related hardware, software and systems (collectively “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 creating and producing designs and using, 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 that 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 intended for use in general electronics applications (e.g., computers, personal equipment, office equipment, measuring equipment, industrial robots and home electronics appliances) or for specific applications as expressly stated in this document. Product is neither intended nor warranted for use in equipment 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 or serious public impact (“Unintended Use”). Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, 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, devices related to electric power, and equipment used in finance-related fields. Do not use Product for Unintended Use unless specifically permitted in this document.

- 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 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.